WCRT Analysis of CAN Messages in Gateway-Integrated In-Vehicle Networks

Guoqi Xie, *Member, IEEE*, Gang Zeng, *Member, IEEE*, Ryo Kurachi, Hiroaki Takada, *Member, IEEE*, Zhetao Li, *Member, IEEE*, Renfa Li, *Senior Member, IEEE*, and Keqin Li, *Fellow, IEEE*

***Abstract*—In modern automobiles, gateways are commonly used to connect several subsystems of controller area networks (CANs) to achieve integrated architecture and distributed cross-bus functionalities. To analyze the real-time property of such in-vehicle networks, worst-case response time (WCRT) analysis for gateway-integrated CAN messages has been studied recently. However, the WCRTs obtained are quite pessimistic. In this study, we examine in detail the various actual arriving orders of gateway messages and then propose an explorative WCRT computation method. It is proved that the obtained WCRT results are safe WCRT bounds. Experimental results for real message set demonstrate as much as 24% reduction of WCRT compared with those obtained using the state-of-the-art methods.**

***Index Terms*—Controller area network (CAN), gateway, integrated architecture, in-vehicle networks, worst case response time (WCRT).**

I. Introduction

A. Background

In a typical car of today, different suppliers develop different distributed subsystems (e.g., the body subsystem, the powertrain subsystem, and the entertainment subsystem, etc.) in a nearly autonomous manner [1], [2]. Current automotive electrical and electronic (E/E) architecture has evolved to an integrated architecture [1], [2], where some distributed automotive functionalities require the exchange of messages between different subsystems. For example, the engine controller of the powertrain subsystem need to take input signals from the body subsystem, and the dashboard of the body subsystem displays information from other subsystems [3], [4]. Controller area network (CAN) is a simple, efficient, and robust broadcast communications bus and is the most widespread networking standard in current in-vehicle networks [5]–[7]. The media access control (MAC) of CAN is based on carrier sensing multiple access with collision resolution (CSMA/CR) with bit-wise arbitration. That is, CAN is based on the distributed priority scheduling without coordination or collision overhead and is ideally suited for dynamic real-time distributed systems because of its event-triggered, non-destructive, and strictly deterministic medium arbitration [8]. Considering the size, weight, and power consumption (SWaP) for cost and high performance benefits, CAN has been divided by means of a gateway in such integrated automotive architecture. The gateway is an important node that is required to copy thousands of CAN messages from one bus to the other bus(es) to complete these distributed cross-bus functionalities [9], [10]. Obtaining the worst case end-to-end delay, namely, the worst case response time (WCRT) analysis is necessary for safe design.

The WCRT of a message is the maximum response time among all possible real response time values when the message is transmitted on a specific hardware platform with a given message set. WCRT analysis is a method to find out the WCRT value from all possible real response time values. Since exact WCRT analysis is a NP-hard problem, it is impossible to obtain accurate WCRT in polynomial time [11]. Therefore, deriving an approximate WCRT upper bound is common method [11]–[14]. The development life cycle of safety-critical systems usually involves the analysis, design, implementation, and testing phases [15]. In the analysis phase, the estimation of WCRT of each message is necessary because its results will be taken as input to build a safe gateway-integrated CAN network in the design phase.

B. Motivation

Finding the exact WCRT requires to solve the combinatorial explosion problem [16], which cannot be calculated out in limited time. Therefore, the general WCRT analysis is to estimate

Manuscript received June 27, 2016; revised March 15, 2017 and June 5, 2017; accepted July 30, 2017. Date of publication August 9, 2017; date of current version November 10, 2017. This work was supported in part by the National Key Research and Development Plan of China under Grant 2016YFB0200405, the National Natural Science Foundation of China under Grant 61702172, Grant 61672217, Grant 61432005, Grant 61379115, Grant 61402170, Grant 61370097, Grant 61502162, and Grant 61502405, the CERNET Innovation Project under Grant NGII20161003, and the China Postdoctoral Science Foundation under Grant 2016M592422. The review of this paper was coordinated by Prof. M. Dianati. (*Corresponding author: Zhetao Li*).

G. Xie and R. Li are with the College of Computer Science and Electronic Engineering, Hunan University, Key Laboratory for Embedded and Network Computing of Hunan Province, Hunan 410082, China (e-mail: xgqman@hnu.edu.cn; lirenfa@hnu.edu.cn).

G. Zeng, R. Kurachi, and H. Takada are with the Nagoya University, Aichi 4648603, Japan (e-mail: sogo@ertl.jp; kurachi@nces.is.nagoya-u.ac.jp; hiro@ertl.jp).

Z. Li is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410006, China, and also with the College of Information Engineering, Xiangtan University, Hunan 411105, China (e-mail: liztchina@hotmail.com).

K. Li is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410006, China, and also with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2017.2737035

a tight WCRT upper bound, which is larger than or equal to the exact WCRT, within a pseudo-polynomial computational time [12]–[14]. In this situation, the difference between the estimated WCRT and the exact WCRT can be used to indicate the precision or the pessimism of the estimation. WCRT upper bound is a safe value in the sense that all possible real response time values do not exceed it. For example, if the exact WCRT is 8, and the obtained WCRT upper bound is 12, then there is pessimism of 4.

The classical WCRT analysis proposed in [12] for a single CAN bus can be applied to gateway-integrated networks, but it can only obtain quite pessimistic results as it overlooks the influence of the gateway. Optimized WCRT is obtained by using the multiple first input first output (FIFO) queues in the gateway with experiments [13]. To analyze the real-time property and reduce the pessimism of such integrated in-vehicle networks, WCRT analysis for gateway-integrated CAN messages has been studied recently. The influence from the gateway for the WCRTs of messages in gateway-integrated CANs was first systematically summarized, and then the concepts of the busy sequence and the arriving order were presented [14]. However, their obtained WCRTs are still pessimistic. Considering that the WCRTs obtained by [13], [14] are quite pessimistic, this work aims to obtain tighter and safe WCRT upper bound with less pessimism than those in [13], [14] by further analyzing the influence of the gateway on the actually possible existing arriving orders of gateway messages. Tighter means that the WCRT upper bound obtained using the proposed method is less than those obtained using the methods in [13], [14], but it is still larger than or equal to the exact WCRT to ensure the safety. For example, if tighter WCRT upper bound obtained by the proposed method is 10, then it can reduce the pessimism and ensure the safety because $8 < 10 < 12$.

C. Our Contributions

The main contributions of this study are as follows.

1) We examine in detail the various arriving orders of gateway messages from a practical perspective. We can obtain the worst case arriving order, which is a real existing arriving order, or the pessimistic worst case arriving order if the worst case arriving order does not exist.
2) We obtain the critical instants and present the explorative WCRT computation method to obtain safe WCRT upper bound on the basis of the obtained worst case arriving order or pessimistic worst case arriving order.
3) We verify that our method can obtain tighter WCRT of each message compared with the state-of-the-art methods reported in [13] and [14] based on a real CAN message set provided by automaker.

II. Related Work

CAN is an event-triggered bus with static priority non-preemptive protocol, and the WCRT analysis for CAN messages builds on the static priority task scheduling of single processor systems. In [17], Joseph *et al.* first proposed the response time analysis for static priority task sets on single processor systems, which based on the classical Liu and Layland's (L&L) task model and critical instants theorem [18]. The concept of a busy period was introduced by Lehoczky [19], which is fundamental in analyzing the WCRT. Harbour *et al.* modified the definition of the busy period and define the priority level-*i* busy period [20]. In subsequent works, Tindell and Burns [21] and Tindell *et al.* [22], [23] provided a method of calculating the maximum queuing delay and hence the worst-case response times of all CAN messages based on the priority level-*i* busy period. Davis *et al.* [12] first found that the analysis method of [21]–[23] may result in computed WCRTs that are optimistic, and thereby provided a revised and improved WCRT computation method by correcting the calculation of maximum queuing delay (please see more details about maximum queuing delay in Section IV-A).

Another important WCRT analysis method is the interference function and maximum interference function (IF&MIF), which was first introduced by Takada and Sakamura [24]. IF&MIF initially aims at computing the WCRT bound for static priority generalized multiframe tasks. The interference function (IF) and maximum interference function (MIF) are functions that represent the time in an interval and the maximum time in an interval, respectively, in which a set of high-priority tasks interferes with the task under analysis. IF&MIF was also extended to FIFO-based offset assigned CAN messages [25] and mixed CAN messages (i.e., the messages that are periodic and event sporadic triggered.) with offsets [26]. Other related works about WCRT analysis methods for CAN messages include stochastic and statistical methods. For example, Zeng *et al.* proposed a stochastic analysis framework to compute CAN message response time in automotive systems with unsynchronized ECUs and periodic messages [27], and proposed the statistical analysis to compute the probability distribution of CAN message response times when only partial information is available about the functionality and architecture of a vehicle [28], respectively.

Network calculus is an another approach that has been applied in analyzing the WCRT for in-vehicle networks, including CAN [29], Ethernet audio video bridging (AVB) [30], and avionics full-duplex switched Ethernet (AFDX) [31]. In general, network calculus can be divided into deterministic network calculus and stochastic network calculus. Deterministic network calculus is a mature and reliable method and its purpose is to get the WCRT upper bound. All the aforementioned literatures [29]–[31] use the deterministic network calculus to analyze the WCRT upper bound.

WCRT analysis for gateway-integrated CAN messages has been studied recently. In [13], Davis *et al.* calculated the WCRT of CAN messages with gateway by reusing the classical priority level-*i* busy period of [12], and pointed out that it is preferable to use multiple FIFO queues for the gateway if it is not possible to implement a priority queue. In [3], Azketa *et al.* presented an optimized WCRT analysis method for CAN messages that takes into account the pipelining of the packets in the gateway. Considering that the messages released sporadically in the gateway, in [14], Yong Xie *et al.* introduced the concepts of busy sequence and minimum distance constraint to analyze the arriving order to obtain safe WCRTs (please refer to Sections IV-C and IV-D for more details about busy sequence and arriving order).

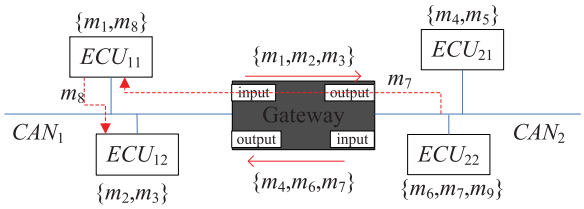


Fig. 1. System model of in-vehicle networks.

III. Models

A. In-Vehicle CANs Model

Similar to [14], we assume that the gateway-integrated in-vehicle networks consist of two CAN buses with the same bandwidth in this study. As shown in Fig. 1, several ECUs are mounted on each CAN bus, and each ECU contains a message set required to be released (queued). ECU is configured with a priority queue. These two buses are connected by a discrete channel CAN gateway [32]. The gateway is configured with two input priority queues and two output priority queues, respectively. Considering that the speed of gateway processing is much faster than that of message transmission, the message can immediately go through the gateway with a negligible processing time [9]. Hence, similar to [14], we assume that no processing time exists in the gateway. In other words, when one message m_i completes its transmission in its source bus and arrives at the gateway, it will be immediately released in the gateway for transmission in its destination bus.

B. Message Model

We use $MS = \{m_1, m_2, ..., m_{|MS|}\}$ to represent a message set, where $|MS|$ is the size of the message set, m_i=$(P_i, T_i, C_i, D_i, CAN_{\mathrm{S},i}, CAN_{\mathrm{D},i})$ is the ith message of the MS, i is the unique identifier of a message, and P_i represents the priority of m_i. If two messages m_i and m_j meet $i < j$, then m_i has a higher priority than m_j. T_i is considered as the period of m_i for a message [33]; and C_i and D_i are the transmission time and deadline of m_i, respectively. Generally, $T_i = D_i$. Finally, $CAN_{\mathrm{S},i}$ and $CAN_{\mathrm{D},i}$ represent the source-end, destination-end buses of m_i, respectively. There are no queuing jitter and offset for each message, which means that all messages in the same ECU can be released simultaneously [14]. Because the maximum transmission rate of CAN bus is 1 Mbps (i.e., the minimum transmission time of 1 bit on the CAN bus is 1 μs), we define the minimum time unit of each message as 1 μs [3], [12]–[14], [27], [28]. Table I lists the message set ($|MS| = 9$) of Fig. 1. This message set is considered as a motivating example in this study. The motivating example will be used to explain the proposed analysis method in the paper. For simplicity, the time units of all parameters are ignored in the example.

C. Message Types and Groups

Two types of messages are transmitted via one bus.

1) The message transmitted only via one bus is called a non-gateway message (e.g., m_5, m_8 and m_9 of the example).
2) The message released on the source-bus forwarding to the other destination-end bus through the gateway is called a gateway message (e.g., m_1, m_2, m_3, m_4, m_6, and m_7 of the example).

To analyze the WCRT of m_i (m_i is the message under analysis), we should group messages according to their relative priorities to m_i, and whether they go through the gateway. We define two groups of messages as follows.

One group is defined for messages in the source-end $CAN_{\mathrm{S},i}$ of m_i and the other group is defined for messages in the destination-end $CAN_{\mathrm{D},i}$ of m_i. Table II shows the notations of these message set and other important definitions.

Here, we use the message m_7 in Fig. 1 as an example, we have $shp_{\mathrm{NGW}}(7) = \{m_5\}$, $shp_{\mathrm{GW}}(7) = \{m_4, m_6\}$, $slp_{\mathrm{NGW}}(7) =$

TABLE I
A Motivating Example of the Message Set

m_i	$CAN_{\mathrm{S},i}$	$CAN_{\mathrm{D},i}$	P_i	T_i	C_i	D_i
m_1	CAN_1	CAN_2	1	14	2	14
m_2	CAN_1	CAN_2	2	16	1	16
m_3	CAN_1	CAN_2	3	13	1	13
m_4	CAN_2	CAN_1	4	16	1	16
m_5	CAN_2	CAN_2	5	10	1	10
m_6	CAN_2	CAN_1	6	18	2	18
m_7	CAN_2	CAN_1	7	20	1	20
m_8	CAN_1	CAN_1	8	14	2	14
m_9	CAN_2	CAN_2	9	20	1	20

TABLE II
Important Notations in This Study

Notation	Definition
$shp_{\mathrm{NGW}}(i)$	Set of high-priority non-gateway messages released in $CAN_{\mathrm{S},i}$
$shp_{\mathrm{GW}}(i)$	Set of high-priority gateway messages released in $CAN_{\mathrm{S},i}$
$shp(i)$	$shp_{\mathrm{GW}}(i) \cup shp_{\mathrm{NGW}}(i)$
$slp_{\mathrm{NGW}}(i)$	Set of low-priority non-gateway messages released in $CAN_{\mathrm{S},i}$
$slp_{\mathrm{GW}}(i)$	Set of low-priority gateway messages released in $CAN_{\mathrm{S},i}$
$dhp_{\mathrm{NGW}}(i)$	Set of high-priority non-gateway messages released in $CAN_{\mathrm{D},i}$
$dhp_{\mathrm{GW}}(i)$	Set of high-priority gateway messages released in $CAN_{\mathrm{D},i}$
$dhp(i)$	$dhp_{\mathrm{GW}}(i) \cup dhp_{\mathrm{NGW}}(i)$
$dlp_{\mathrm{NGW}}(i)$	Set of low-priority non-gateway messages released in $CAN_{\mathrm{D},i}$
$dlp_{\mathrm{GW}}(i)$	Set of low-priority gateway messages released in $CAN_{\mathrm{D},i}$
$CAN_{\mathrm{S},i}$	Source-end bus of the message m_i
$CAN_{\mathrm{D},i}$	Destination-end bus of the message m_i
C_i	Transmission time of the message m_i
D_i	Deadline of the message m_i
B_i	Blocking of the message m_i
$R_{\mathrm{S},i}$	Source-end WCRT of the message m_i
$R_{\mathrm{D},i}$	Destination-end WCRT of the message m_i
$R_{\mathrm{E2E},i}$	End-to-end WCRT of the message m_i
T_i	Period or inter-arrival time of the message m_i when in the ECU
$T_i^{\min}$	Minimum inter-arrival time of the message m_i in the gateway
$order_{\mathrm{GW}}^{k}(i)$	kth arriving order of $dhp_{\mathrm{GW}}(i)$.
$order_{\mathrm{GW}}^{\mathrm{SR}}(i)$	Simultaneously released arriving order of $dhp_{\mathrm{GW}}(i)$.
$order_{\mathrm{GW}}^{\mathrm{S}}(i)$	Successive arriving order of $dhp_{\mathrm{GW}}(i)$.
$order_{\mathrm{GW}}^{\mathrm{B}}(i)$	Busy arriving order of $dhp_{\mathrm{GW}}(i)$.
$order_{\mathrm{GW}}^{\mathrm{W}}(i)$	Worst case arriving order of $dhp_{\mathrm{GW}}(i)$.
$order_{\mathrm{GW}}^{\mathrm{E}}(i)$	Real existing arriving order of $dhp_{\mathrm{GW}}(i)$.
$order_{\mathrm{GW}}^{\mathrm{PW}}(i)$	Pessimistic worst case arriving order of $dhp_{\mathrm{GW}}(i)$.
$A(m_{i,k})$	Arriving instant of the kth job of the message m_i in the gateway.

$\{m_9\}$, $slp_{\mathrm{GW}}(7) = \{\}$, $dhp_{\mathrm{NGW}}(7) = \{\}$, $dhp_{\mathrm{GW}}(7) = \{m_1, m_2, m_3\}$, $dlp_{\mathrm{NGW}}(7) = \{m_8\}$, and $dlp_{\mathrm{GW}}(7) = \{\}$.

In this study, we first analyze the WCRT of m_i on its source-end bus $CAN_{\mathrm{S},i}$, and it is denoted by $R_{\mathrm{S},i}$. If m_i is a gateway message, we then further analyze the WCRT of m_i on its destination-end bus $CAN_{\mathrm{D},i}$, and it is denoted by $R_{\mathrm{D},i}$. Finally, the end-to-end WCRT $R_{\mathrm{E2E},i}$ for gateway message m_i is the sum of $R_{\mathrm{S},i}$ and $R_{\mathrm{D},i}$, namely,

$$R_{\mathrm{E2E},i} = R_{\mathrm{S},i} + R_{\mathrm{D},i}. \tag{1}$$

Using the above division approach (source-end and destination-end) for end-to-end WCRT analysis is tractable and safe. The reason is that if the individual WCRT upper bounds for source-end and destination-end can be obtained, respectively, the sum of them will be the end-to-end WCRT upper bound.

IV. Preliminaries

A. Minimum Inter-Arrival Time

m_i is affected by two types of messages according to classical WCRT theory in its source-end (note that all the analysis and explanations about m_i is based on the fact that m_i is transmitted on its source-end bus unless otherwise particularly indicated in this study).

1) Low-priority messages in $slp_{\mathrm{NGW}}(i)$, $slp_{\mathrm{GW}}(i)$, and $dlp_{\mathrm{GW}}(i)$ could block m_i. These messages only cause a priority inversion once to m_i with blocking [12], [14].
2) High-priority messages in $shp_{\mathrm{NGW}}(i)$, $shp_{\mathrm{GW}}(i)$, and $dhp_{\mathrm{GW}}(i)$ would cause interference to m_i. These messages can be classified into two types. One is m_x ($x \in shp_{\mathrm{GW}}(i) \cup shp_{\mathrm{NGW}}(i)$) which can be released periodically with its period T_x because it is released in the same bus $CAN_{\mathrm{S},i}$ with m_i [12], [14]. The other is m_y ($y \in dhp_{\mathrm{GW}}(i)$) which is released sporadically in the gateway [14] because its released instant in the gateway is its finish instant on $CAN_{\mathrm{D},i}$. If $m_{y,1}$ and $m_{y,2}$ experience a time of $R_{\mathrm{S},y}$ and C_y in $CAN_{\mathrm{S},y}$, respectively, then m_y arrives at the gateway with minimum inter-arrival time $T_y^{\min}$ between these two jobs (a job is an instance of a message). $T_y^{\min}$ is calculated by

$$T_y^{\min} = T_y - R_{\mathrm{s},y} + C_y, \tag{2}$$

where $R_{\mathrm{s},y}$ is the WCRT of m_y in its resource-end. For example, as shown in Fig. 2(a), m_3 is released periodically with its period $T_3 = 13$ in ECU_{12}. In Fig. 2(b), $m_{3,1}$ and $m_{3,2}$ experience the response time of $R_{\mathrm{S},3} = 6$ and $C_3 = 1$, respectively. Hence, the minimum inter-arrival for m_3 is $T_3^{\min} = 13 - 6 + 1 = 8$ in the gateway, as shown in Fig. 2(c).

B. Maximum Queuing Delay

If the gateway message m_y ($y \in dhp_{\mathrm{GW}}(i)$) is considered as periodic message with equivalent period of $T_y^{\min}$, and all high-priority messages of $shp_{\mathrm{NGW}}(i)$, $shp_{\mathrm{GW}}(i)$, and $dhp_{\mathrm{GW}}(i)$ are released simultaneously with m_i, we can reuse the existing WCRT analysis method in [12] and [13] to calculate the maximum queuing delay w_i using the following iteration (3) and (4) for a single CAN bus:

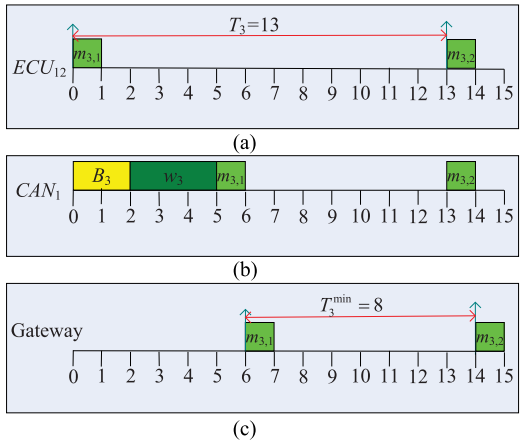


Fig. 2. Minimum inter-arrival time of m_3 in the gateway. (a) m_3 is released periodically in ECU_{12} with period $T_3 = 13$. (b) $m_{3,1}$ and $m_{3,2}$ are transmitted on CAN_1. (c) Minimum inter-arrival time of m_3 is 8.

$$w_i^{n+1} = B_i + \sum_{\forall x \in shp(i)} \left\lceil \frac{w_i^n + \tau_{\mathrm{bit}}}{T_x} \right\rceil C_x + \sum_{\forall y \in dhp_{\mathrm{GW}}(i)} \left\lceil \frac{w_i^n + \tau_{\mathrm{bit}}}{T_y^{\min}} \right\rceil C_y, \tag{3}$$

where $shp(i) = shp_{\mathrm{GW}}(i) \cup shp_{\mathrm{NGW}}(i)$, and τ_{bit} represents the transmission time for a single bit, and B_i is the blocking calculated by

$$B_i = \max(^{\max(C_k)}_{\forall k \in slp_{\mathrm{NGW}}(i) \cup slp_{\mathrm{GW}}(i) \cup dlp_{\mathrm{GW}}(i)}, C_i). \tag{4}$$

Iteration starts with a suitable initial value, such as $w_i^0 = C_i$, and continues until either $w_i^{n+1} + C_i > D_i$ in which case the message is not schedulable, or $w_i^{n+1} = w_i^n$ and $w_i^{n+1} + C_i \leqslant D_i$ in which case the message is schedulable [12], [13]. Finally, WCRT of m_i in the source-end can be calculated by

$$R_{\mathrm{S},i} = w_i^{n+1} + C_i. \tag{5}$$

C. Busy Sequence

To calculate $R_{\mathrm{S},i}$ of (5), we should acquire $T_y^{\min}$ of (3) in advance, which depends on $R_{\mathrm{s},y}$ according to (2). Hence, for a given message set, we should analyze these messages according to their priorities from the highest to the lowest priority message. For example, we should analyze m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, and m_9 step by step in the motivating example. However, using (3) to compute the w_i is much pessimistic, because the minimum inter-arrival time $T_y^{\min}$ for m_y is a very rare event in the actual situation. To address the above problem, a more practical situation is considered in [14]. If the inter-arrival time between the first and the second arriving jobs in the gateway is $T_y^{\min}$, then the inter-arrival time between any other adjacent jobs should be T_y. If m_y arrives at the gateway with the above arriving sequence, then this sequence is called the busy sequence of m_y [14].

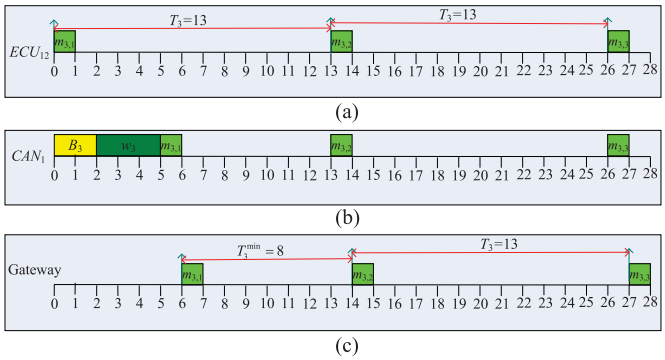


Fig. 3. Busy sequence of m_3. (a) m_3 is released periodically in ECU_{12} with period $T_3 = 13$. (b) $m_{3,1}$, $m_{3,2}$, and $m_{3,3}$ are transmitted on CAN_1. (c) Busy sequence ofm_3.

Fig. 3 shows an example of the busy sequence of m_3, we can see that the inter-arrival time between $m_{3,1}$ and $m_{3,2}$, and between $m_{3,2}$ and $m_{3,3}$ in the gateway are $T_3^{\min} = 8$ and $T_3 = 13$, respectively (Fig. 3(c)). The sequence is a busy sequence because the kth ($k \geqslant 2$) arriving job experiences the minimum response time of C_y in $CAN_{\mathrm{S},y}$. Moreover, the busy sequence is important for WCRT calculation because it contains the maximum number of arriving jobs of a gateway message in any given interval. Hence, if each message in $dhp_{\mathrm{GW}}(i)$ arrives at the gateway with their busy sequences, and all the messages in $dhp_{\mathrm{GW}}(i)$ are released simultaneously in the gateway, then $dhp_{\mathrm{GW}}(i)$ causes the maximum interference to m_i with less pessimism than using (3)–(5).

D. Arriving Order

In actual cases, the busy sequences of all gateway messages cannot simultaneously exist. The reason is that the released instant of message m_y ($y \in dhp_{\mathrm{GW}}(i)$) in the gateway is the transmitted finish instant of m_y on $CAN_{\mathrm{D},y}$, while the actual transmission sequence of messages is the serialized sequence on $CAN_{\mathrm{D},i}$. Hence, at most only a single message in $dhp_{\mathrm{GW}}(i)$ can be released simultaneously with m_i in the gateway. For this reason, the arriving order is defined as follows [14]. Assume that we have $dhp_{\mathrm{GW}}(i) = \{m_\alpha, m_\beta, ..., m_\gamma\}$ and we use $order^k_{\mathrm{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{\gamma,1})$ to represent the kth arriving order of $dhp_{\mathrm{GW}}(i)$. $order^k_{\mathrm{GW}}(i)$ consists of all the first message jobs in $dhp_{\mathrm{GW}}(i)$, which are ordered according to their arriving instants at the gateway.

Assume that the size of $dhp_{\mathrm{GW}}(i)$ is n, then there are $n!$ arriving orders. For the motivating example, we have $dhp_{\mathrm{GW}}(8) = \{m_4, m_6, m_7\}$. Hence, there are $3! = 6$ arriving orders for m_8 in the gateway: $order^1_{\mathrm{GW}}(8) = (m_{4,1}, m_{6,1}, m_{7,1})$, $order^2_{\mathrm{GW}}(8) = (m_{4,1}, m_{7,1}, m_{6,1})$, $order^3_{\mathrm{GW}}(8) = (m_{6,1}, m_{4,1}, m_{7,1})$, $order^4_{\mathrm{GW}}(8) = (m_{6,1}, m_{7,1}, m_{4,1})$, $order^5_{\mathrm{GW}}(8) = (m_{7,1}, m_{4,1}, m_{6,1})$, and $order^6_{\mathrm{GW}}(8) = (m_{7,1}, m_{6,1}, m_{4,1})$.

Any arriving order can cause interference to m_i, and the arriving order causing the maximum interference to m_i should be found out [14]. To this end, an exhaustive search can be employed, and it can generate less pessimistic WCRT than the method using (3)-(5). However, it remains a pessimistic method because of the following reasons.

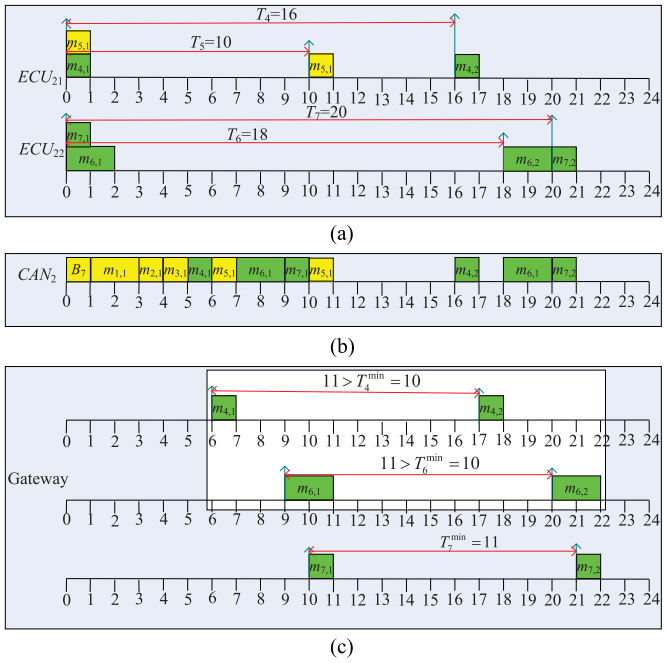


Fig. 4. Messages in $dhp_{\mathrm{GW}}(8)$ cannot arrive at the gateway with their busy sequences simultaneously. (a) Periodically released instants of m_4, m_6, and m_7 on ECUs. (b) m_4, m_6, and m_7 are transmitted on CAN_2. (c) m_7 meets its busy sequence, but both m_4 and m_6 cannot meet their busy sequences.

First, all the messages in $dhp_{\mathrm{GW}}(i)$ do not always arrive at the gateway with its busy sequence. For example, if we analyze m_8 in CAN_1 of Fig. 1, then $dhp_{\mathrm{GW}}(8) = \{m_4, m_6, m_7\}$. To make m_7 arrive at the gateway with its busy sequence, $m_{7,1}$ must be released simultaneously with $m_{4,1}$, $m_{5,1}$ and $m_{6,1}$ in their ECUs on CAN_2, as shown in Fig. 4(a). In this situation, gateway messages $m_{4,1}$, $m_{6,1}$, and $m_{7,1}$ have the same blocking time $B_7 = 1$, as shown in Fig. 4(b). However, a conflict is that both $m_{4,1}$ and $m_{6,1}$ should be blocked with $B_4 = B_6 = 2$, if they are released in the gateway with its busy sequence. Hence, although $m_{7,1}$ meets its busy sequence, both $m_{4,1}$ and $m_{6,1}$ cannot meet their busy sequences, as shown in Fig. 4(c). In other words, messages in $dhp_{\mathrm{GW}}(8)$ cannot arrive at the gateway with their busy sequences simultaneously.

Second, the arriving order $order^k_{\mathrm{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{\gamma,1})$ does not always arrive at the gateway one by one. In fact, the actual sequence on $CAN_{\mathrm{D},i}$ contains non-gateway message jobs. Moreover, $m_{\alpha,2}$ may arrive at the gateway earlier than $m_{\gamma,1}$. For example, we have an arriving order $order^1_{\mathrm{GW}}(8) = (m_{4,1}, m_{6,1}, m_{7,1})$, but the actual transmitted sequence on CAN_2 may be $(m_{4,1}, m_{5,1}, m_{6,1}, m_{4,2}, m_{7,1})$, where $m_{4,2}$ arrives at the gateway earlier than $m_{7,1}$.

Besides the pessimism, a problem of the exhaustive search is that it is not applicable for large-scale message set because all the $n!$ arriving orders should be calculated. To solve this problem, a simplified search is proposed in [14]. That is, for an arriving order, except for the first job, the other jobs are assumed to arrive at the gateway simultaneously. For example, as shown in Fig. 5, for $order^1_{\mathrm{GW}}(8) = (m_{4,1}, m_{6,1}, m_{7,1})$, $m_{4,1}$ first arrives at the gateway, and then $m_{6,1}$ and $m_{7,1}$ arrive at the gateway simultaneously by moving $m_{7,1}$ from time instant 10 to 9. Accordingly,

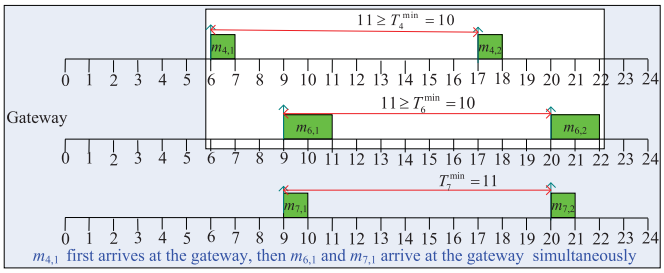


Fig. 5. Example of simplified arriving order and simplified search.

the other jobs $m_{7,k}$ ($k > 2$) are also moved. By this treatment, $order^1_{\mathrm{GW}}(8)$ is completely equal to $order^2_{\mathrm{GW}}(8)$. The simplified search can reduce the search space from $n!$ to n; however, this approach obviously introduces additional pessimism.

V. Arriving Order Analysis

To cope with the aforementioned problems, we should further analyze the arriving order in the gateway with more actual consideration.

A. Simultaneously Released Arriving Order

Definition 1 (Simultaneously Released Arriving Order): For the arriving order $order^k_{\mathrm{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{\gamma,1})$ on the gateway, if all these jobs are released simultaneously on the destination-end bus $CAN_{\mathrm{D},i}$, then this arriving order is called the simultaneously released arriving order on the gateway and is represented by $order^{\mathrm{SR}}_{\mathrm{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{\gamma,1})$ ($\alpha < \beta < ... < \gamma$).

Obviously, only one simultaneously released arriving order exists for $dhp_{\mathrm{GW}}(i)$. For example, as shown in Fig. 4(c), the simultaneously released arriving order for $dhp_{\mathrm{GW}}(8)$ is $order^{\mathrm{SR}}_{\mathrm{GW}}(8) = (m_{4,1}, m_{6,1}, m_{7,1})$, where $m_{4,1}$, $m_{6,1}$, and $m_{7,1}$ released simultaneously in their ECUs. We also list the simultaneously released arriving orders for $dhp_{\mathrm{GW}}(7)$, which is $order^{\mathrm{SR}}_{\mathrm{GW}}(7) = (m_{1,1}, m_{2,1}, m_{3,1})$, as shown in Fig. 6(c).

B. Successive Arriving Order

Definition 2 (Successive Arriving Order): For the arriving order $order^k_{\mathrm{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{\gamma,1})$ on the gateway, if it can cause successive interference to m_i between the arriving instant of $m_{\alpha,1}$ and the finish instant of $m_{\gamma,1}$, then this order is called the successive arriving order on the gateway and is denoted as $order^{\mathrm{S}}_{\mathrm{GW}}(i)$.

Theorem 1: Assume that a message job sequence consists of the messages of $dhp_{\mathrm{GW}}(i)$ that transmitted on $CAN_{\mathrm{D},i}$, and is denoted as $seq_{\mathrm{GW}}(i) = (m_{s(1)}, m_{s(2)}, m_{s(3)}, ..., m_{s(m)})$. If

$$B_i + \sum_{\forall x \in shp(i)} C_x + C_{s(1)} \geqslant \max_{\forall y \in dhp_{\mathrm{GW}}(i)} C_y, \tag{6}$$

where $shp(i) = shp_{\mathrm{GW}}(i) \cup shp_{\mathrm{NGW}}(i)$ as mentioned earlier, then $seq_{\mathrm{GW}}(i)$ can trigger a successive arriving order.

Proof: Assume that the arriving instant for $m_{s(1)}$ is $A(m_{s(1)})$ in the gateway. Considering that the arriving instant for each job in the gateway is its earliest transmitted finish instant in its source-end bus, then the earliest arriving instant for $m_{s(2)}$ should be

$$A(m_{s(2)}) = A(m_{s(1)}) + C_{s(2)}.$$

■

Similarly, we have

$$A(m_{s(3)}) = A(m_{s(2)}) + C_{s(3)},$$

..., and

$$A(m_{s(m)}) = A(m_{s(m-1)}) + C_{s(m)}. \tag{7}$$

Then, the earliest finish instant for for $F(m_{s(k)})$ should be

$$F(m_{s(m)}) = A(m_{s(m)}) + C_{s(m)}. \tag{8}$$

We define the minimum slack $S(m_{(k)})$ between the current job $m_{(k)}$ ($k \geqslant 2$) and its predecessor job $m_{(k-1)}$, which is calculated by

$$\begin{aligned} S(m_{s(k)}) &= A(m_{s(k)}) - F(m_{s(k-1)}) \\ &= A(m_{s(k)}) - (A(m_{s(k-1)}) + C_{s(k-1)}) \\ &= A(m_{(k-1)}) + C_{s(k)} - A(m_{(k-1)}) - C_{s(k-1)} \\ &= C_{s(k)} - C_{s(k-1)}. \end{aligned}$$

The subsequence $(m_{s(1)}, m_{(2)})$ can trigger a successive arriving order to m_i, because

$$\begin{aligned} &B_i + \sum_{\forall x \in shp(i)} C_x - S(m_{(2)}) \\ &= B_i + \sum_{\forall x \in shp(i)} C_x + C_{s(1)} - C_{s(2)} \geqslant 0, \end{aligned}$$

according to (6).

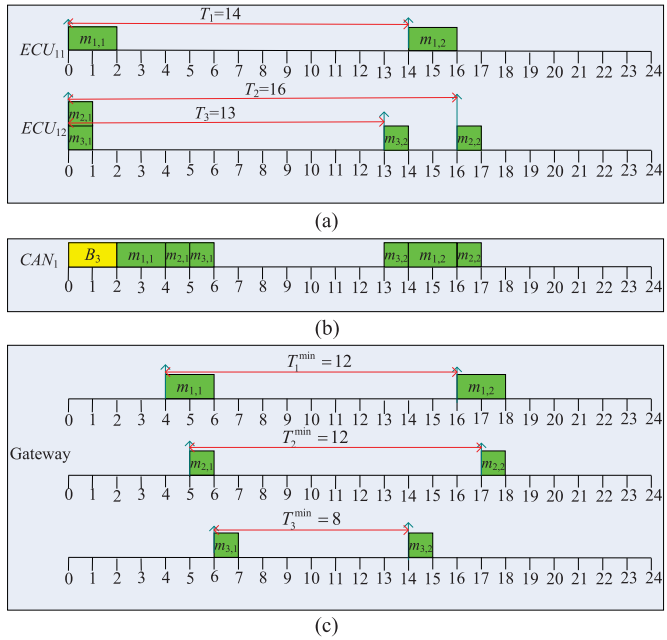


Fig. 6. Arriving order $order^{\mathrm{SR}}_{\mathrm{GW}}(7) = (m_{1,1}, m_{2,1}, m_{3,1})$. (a) Periodically released instants of m_1, m_2, and m_3 on ECUs. (b) m_1, m_2, and m_3 are transmitted on CAN_1. (c) Arriving order of m_1, m_2, and m_3 on the gateway.

The subsequence $(m_{(1)}, m_{(2)}, m_{(3)})$ can trigger a successive arriving order to m_i, because

$$B_i + \sum_{\forall x \in shp(i)} C_x - (S(m_{s(2)}) + S(m_{(3)})) = B_i + \sum_{\forall x \in shp(i)} C_x + C_{s(1)} - C_{s(3)} \geqslant 0.$$

The subsequence $(m_{(1)}, m_{(2)}, m_{s(3)}, ..., m_{s(m)})$ can trigger a successive arriving order to m_i, because

$$B_i + \sum_{\forall x \in shp(i)} C_x - (S(m_{s(2)}) + S(m_{s(3)}) + , , , + S(m_{(m)})) = B_i + \sum_{\forall x \in shp(i)} C_x + C_{s(1)} - C_{s(m)} \geqslant 0.$$

Given that all the subsequences can trigger a successive arriving order to m_i, $seq_{\text{GW}}(i)$ can trigger a successive arriving order to m_i. ■

For example, although a slack between $m_{4,1}$ and $m_{6,1}$ exists in Fig. 4(c), the sequence $(m_{4,1}, m_{6,1}, m_{7,1})$ in Fig. 4(b) still can trigger a successive arriving order to m_8, because $B_8 + C_1 + C_2 + C_3 + C_4 \geqslant C_6$ ((6)); moreover, $B_8 + C_1 + C_2 + C_3 = 6$, which is larger than the slack $S(m_2) = 2$ and can fill this slack. The sequence $(m_{1,1}, m_{2,1}, m_{3,1})$ in Fig. 6(b) can also trigger a successive arriving order to m_7, because $B_7 + C_4 + C_5 + C_6 + C_1 \geqslant C_1$ ((6)), and no slack exists in Fig. 6(c).

C. Busy Arriving Order

Definition 3 (Busy Arriving Order): If all the messages in the simultaneously released arriving order $order^{\text{SR}}_{\text{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{u,1}, m_{v,1}, m_{\gamma,1})$ $(\alpha < \beta < ... < u < v < \gamma)$ arrive at the gateway with their busy sequences, then this order is called the busy arriving order and is denoted by $order^{\text{B}}_{\text{GW}}(i)$.

Theorem 2: If all the messages in the simultaneously released arriving order $order^{\text{SR}}_{\text{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{u,1}, m_{v,1}, m_{\gamma,1})$ have the same blocking (i.e., $B_\alpha = B_\beta = ... = B_u = B_v = B_\gamma$), then $order^{\text{SR}}_{\text{GW}}(i)$ is the $order^{\text{B}}_{\text{GW}}(i)$.

Proof: As the order is the simultaneously released arriving order $order^{\text{SR}}_{\text{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{u,1}, m_{v,1}, m_{\gamma,1})$, $m_{\gamma,1}$ must experience the WCRT in its source-end with the blocking B_γ, and must be released simultaneously with $m_{\alpha,1}$, $m_{\beta,1}, ..., m_{u,1}$, and $m_{v,1}$ in the destination-end of m_i. That is, the inter-arrival time between the first and the second arriving jobs of the lowest priority message m_γ in the gateway is $T^{\min}_\gamma$, and the inter-arrival time between any other adjacent jobs should be T_γ; hence, m_γ arrives at the gateway with its busy sequence where blocking is B_γ. ■

Given that $m_{v,1}$ is released simultaneously with $m_{\alpha,1}$, $m_{\beta,1}$,..., and $m_{u,1}$ according to the condition, if $m_{v,1}$ experiences the maximum blocking B_v, then $m_{v,1}$ also arrives at the gateway with its busy sequence. However, the blocking for $m_{v,1}$ is B_γ according to the condition. Hence, if $B_v = B_\gamma$, then $m_{v,1}$ also arrives at the gateway with its busy sequence.

Similar to $m_{v,1}$, if $B_\gamma = B_u = ... = B_\beta = B_\alpha$, then $m_{u,1}$,..., $m_{\beta,1}$, and $m_{\alpha,1}$ also arrive at the gateway with their busy sequences. Hence, if $B_\alpha = B_\beta = ... = B_u = B_v = B_\gamma$, then all the messages in the $order^{\text{SR}}_{\text{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., , m_{u,1}, m_{v,1}, m_{\gamma,1})$ arrive at the gateway with their busy sequences, and the simultaneously released arriving order is the busy arriving order. ■

For example, as shown in Fig. 4(c), $order^{\text{SR}}_{\text{GW}}(8) = (m_{4,1}, m_{6,1}, m_{7,1})$ is not the busy arriving order, because $B_4 = B_6 = 2$, and $B_7 = 1$, such that the inter-arrival time between $m_{4,1}$ and $m_{4,2}$ is 11, which is larger than $T^{\min}_4 = 10$. In contrast, as shown in Fig. 6(c), $order^{\text{SR}}_{\text{GW}}(7) = (m_{1,1}, m_{2,1}, m_{3,1})$ is the busy arriving order, because $B_1 = B_2 = B_3 = 2$, such that all messages arrive at the gateway with their busy sequences.

D. Worst Case Arriving Order

Definition 4 (Worst Case Arriving Order): If the busy arriving order $order^{\text{B}}_{\text{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{\gamma,1})$ is also a successive arriving order, then this order is the worst case arriving order, and is represented by $order^{\text{W}}_{\text{GW}}(i)$.

According to Definition 4, the simultaneously released arriving order $order^{\text{SR}}_{\text{GW}}(8) = (m_{4,1}, m_{6,1}, m_{7,1})$ of Fig. 4(c) is not the worst case arriving order because it is the successive arriving order but not the busy arriving order. The simultaneously released arriving order $order^{\text{SR}}_{\text{GW}}(7) = (m_{1,1}, m_{2,1}, m_{3,1})$ of Fig. 6(c) is the worst case arriving order because it is not only the successive arriving order, but also the busy arriving order. Hence, we have $order^{\text{W}}_{\text{GW}}(7) = (m_{1,1}, m_{2,1}, m_{3,1})$.

Theorem 3: The caused interference to m_i by the worst case arriving order $order^{\text{W}}_{\text{GW}}(i)$ is larger than or equal to that by any other real existing arriving order (denoted as $order^{\text{E}}_{\text{GW}}(i)$) of $dhp_{\text{GW}}(i)$.

Proof: We use $A(order^{\text{W}}_{\text{GW}}(i), m_{y,k})$ and $A(order^{\text{E}}_{\text{GW}}(i), m_{y,k})$ to represent the relative arriving instant of $m_{y,k}$ for $order^{\text{W}}_{\text{GW}}(i)$ and $order^{\text{E}}_{\text{GW}}(i)$, respectively. If $A(order^{\text{W}}_{\text{GW}}(i), m_{y,k}) \leqslant A(order^{\text{E}}_{\text{GW}}(i), m_{y,k})$ $(k \geqslant 2)$ for any message m_y in $dhp_{\text{GW}}(i)$, then Theorem 3 is proved. ■

Given that the worst case arriving order $order^{\text{W}}_{\text{GW}}(i)$ is a successive arriving order, even $A(order^{\text{W}}_{\text{GW}}(i), m_{y,1}) \leqslant A(order^{\text{E}}_{\text{GW}}(i), m_{y,1})$ for m_y is not satisfied, all the first arriving jobs of the $order^{\text{W}}_{\text{GW}}(i)$ can cause successive interference to m_i. Hence, we should skip the first jobs and continue to analyze the interference of the kth $(k \geqslant 2)$ jobs of all the messages of $dhp_{\text{GW}}(i)$.

We first consider the second job of each message. Given that the $order^{\text{W}}_{\text{GW}}(i)$ is the busy arriving order, $A(order^{\text{W}}_{\text{GW}}(i), m_{y,2})$ is calculated by

$$A\left(order^{\text{W}}_{\text{GW}}(i), m_{y,2}\right) = T_y + C_y - R_{S,\text{HP}}, \tag{9}$$

where $R_{\text{S,HP}}$ represents the source-end WCRT for the highest priority message of the $dhp_{\text{GW}}(i)$. For example, as shown in Fig. 6(c), $A(order^{\text{W}}_{\text{GW}}(7), m_{3,2}) = T_7 + C_7 - R_{\text{S},1} = 13 + 1 - 4 = 10$, which is from instant 4 to instant 14. Consequently, we use the proof by contradiction.

Case 1: The $order^{\text{E}}_{\text{GW}}(i)$ is the busy arriving order, but not a successive arriving order, then we have

$$A\left(order^{\text{E}}_{\text{GW}}(i), m_{y,2}\right) = T_y + C_y - R_{S,\text{HP}} = A\left(order^{\text{W}}_{\text{GW}}(i), m_{y,2}\right).$$

Although $A(order^{\mathrm{E}}_{\mathrm{GW}}(i), m_{y,2}) = A(order^{\mathrm{W}}_{\mathrm{GW}}(i), m_{y,2})$, all the first arriving jobs of the $order^{\mathrm{E}}_{\mathrm{GW}}(i)$ may not cause successive interference to m_i according to Theorem 1.

Case 2: The $order^{\mathrm{E}}_{\mathrm{GW}}(i)$ is the simultaneously released arriving order, but not the busy arriving order, then regardless of whether $order^{\mathrm{E}}_{\mathrm{GW}}(i)$ is the successive arriving order, we have

$$\begin{aligned} A\left(order^{\mathrm{E}}_{\mathrm{GW}}(i), m_{y,2}\right) &\geqslant T_y + C_y - R_{\mathrm{S,HP}} \\ &= A\left(order^{\mathrm{W}}_{\mathrm{GW}}(i), m_{y,2}\right). \end{aligned}$$

Case 3: The $order^{\mathrm{E}}_{\mathrm{GW}}(i)$ is not the simultaneously released arriving order, so it is not the busy arriving order. Then, regardless of whether $order^{\mathrm{E}}_{\mathrm{GW}}(i)$ is the successive arriving order, we have

$$\begin{aligned} A\left(order^{\mathrm{E}}_{\mathrm{GW}}(i), m_{y,2}\right) &> T_y + C_y - R_{\mathrm{S,HP}} \\ &= A\left(order^{\mathrm{W}}_{\mathrm{GW}}(i), m_{y,2}\right). \end{aligned}$$

Similar to the analysis of the second job, we have the same results for the kth ($k \geqslant 3$) job of each message. Hence, Theorem 3 is proved. ■

For example, the caused interference to m_7 by the worst case arriving order $order^{\mathrm{W}}_{\mathrm{GW}}(7) = (m_{1,1}, m_{2,1}, m_{3,1})$ is larger than or equal to that by other arriving orders $(m_{1,1}, m_{3,1}, m_{2,1})$, $(m_{2,1}, m_{1,1}, m_{3,1})$, $(m_{2,1}, m_{3,1}, m_{1,1})$, $(m_{3,1}, m_{1,1}, m_{2,1})$, and $(m_{3,1}, m_{2,1}, m_{1,1})$.

E. Pessimistic Worst Case Arriving Order

Given that meeting the worst case arriving order for $dhp_{\mathrm{GW}}(i)$ requires strict constraint conditions, it may not exist in the actual situation. In this case, we first let the messages be released simultaneously in $CAN_{\mathrm{D},i}$ to form the simultaneously released arriving order $order^{\mathrm{SR}}_{\mathrm{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, m_{u,1}, m_{v,1}, ..., m_{\gamma,1})$ ($\alpha < \beta < ... < u < v < \gamma$), and then change the order to the worst case arriving order by moving some related jobs with the following rules.

Rule 1: The simultaneously released arriving order $order^{\mathrm{SR}}_{\mathrm{GW}}(i)$ can be modified to construct a successive arriving order $order^{\mathrm{S}}_{\mathrm{GW}}(i)$ by executing the following operations.

We know that $(m_{\alpha,1}, m_{\beta,1})$ is a successive subsequence, if

$$B_i + \sum_{\forall x \in shp(i)} C_x + C_\alpha - C_\beta \geqslant 0,$$

according to (6) of Theorem 1. Hence, if

$$B_i + \sum_{\forall x \in shp(i)} C_x + C_\alpha - C_\beta < 0,$$

then we can move the arriving instant of $m_{\beta,1}$ from $A(m_{\beta,1})$ to $A(m_{\beta,1}) - (B_i + \sum_{\forall x \in shp(i)} C_x + C_\alpha - C_\beta)$, such that the $(m_{\alpha,1}, m_{\beta,1})$ becomes a successive subsequence.

The other jobs in $(m_{u,1}, m_{v,1}, ..., m_{\gamma,1})$ can also be operated similar to $m_{\beta,1}$ to make all the subsequences successive. Finally, $order^{\mathrm{SR}}_{\mathrm{GW}}(i)$ is changed to the successive arriving order $order^{\mathrm{S}}_{\mathrm{GW}}(i)$.

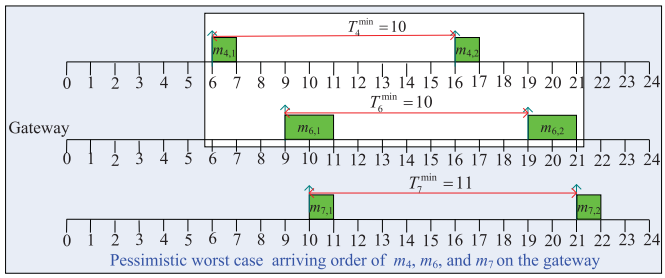


Fig. 7. Pessimistic worst case arriving order $order^{\mathrm{PW}}_{\mathrm{GW}}(8)$.

Rule 2: The successive arriving order $order^{\mathrm{S}}_{\mathrm{GW}}(i)$ can be changed to the busy arriving order $order^{\mathrm{B}}_{\mathrm{GW}}(i)$ by executing the following operations.

We know that for any message m_y in $dhp_{\mathrm{GW}}(i)$, if the inter-arrival time between $m_{y,1}$ and $m_{y,2}$ is $T_y^{\min}$ ($T_y^{\min} = T_y - R_{\mathrm{S},y} + C_y$), and the inter-arrival time between $m_{y,k}$ and $m_{y,k+1}$ ($k \geqslant 2$) is T_y, then it arrives at the gateway with its busy sequence. Hence, if we change

$$A(m_{y,2}) = A(m_{y,1}) + T_y^{\min},$$
$$A(m_{y,3}) = A(m_{y,2}) + T_y,$$

and

$$A(m_{y,k+1}) = A(m_{y,k}) + T_y$$

step by step, then m_y arrives at the gateway with its busy sequence.

When all the other messages in $dhp_{\mathrm{GW}}(i)$ arrive at the gateway with their busy sequences by using the same operation with m_y, the successive arriving order $order^{\mathrm{S}}_{\mathrm{GW}}(i)$ is changed to the busy arriving order $order^{\mathrm{B}}_{\mathrm{GW}}(i)$.

If *Rule 1* and/or *Rule 2* is employed, then the given arriving order is changed to the worst arriving order. Considering that the worst arriving order does not exist in the actual situation, the modified order is called the pessimistic worst case arriving order, and is denoted as $order^{\mathrm{PW}}_{\mathrm{GW}}(i)$. Obviously, Theorem 3 is also suitable for the pessimistic worst case arriving order.

For example, we have known that the simultaneously released arriving order $order^{\mathrm{SR}}_{\mathrm{GW}}(8) = (m_{4,1}, m_{6,1}, m_{7,1})$ of Fig. 4(c) is not the worst case arriving order because it is the successive arriving order but not the busy arriving order (explained in Section V-D). To address the above problem, we move $m_{4,2}$ from instant 17 to instant 16, and move $m_{6,2}$ from instant 20 to instant 19, respectively, as shown in Fig. 7. Accordingly, the other jobs $m_{4,k}$ and $m_{6,k}$ ($k \geqslant 2$) are also moved, and then the pessimistic worst case arriving order $order^{\mathrm{PW}}_{\mathrm{GW}}(8)$ is formed.

VI. WCRT Analysis

A. Finding the Critical Instant

Theorem 4: If the first arriving job of the arriving order $order^{\mathrm{W}}_{\mathrm{GW}}(i) = (m_{\alpha,1}, m_{\beta,1}, ..., m_{\gamma,1})$ (or $order^{\mathrm{PW}}_{\mathrm{GW}}(i)$) is released simultaneously with all the messages in the $shp(i) = shp_{\mathrm{GW}}(i) \cup shp_{\mathrm{NGW}}(i)$, then the released instant is the source-end critical instant CI_i, of m_i.

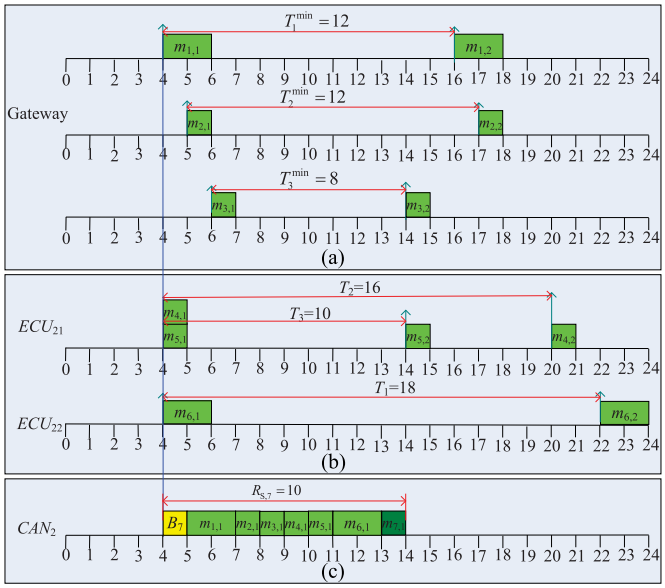


Fig. 8. Critical instant for m_7 in CAN_2. (a) Worst case arriving order of m_1, m_2, and m_3 on the gateway. (b) m_4, m_5, and m_6 are released at instant 4 on ECUs. (c) $R_{S,7} = 10$ on CAN_2.

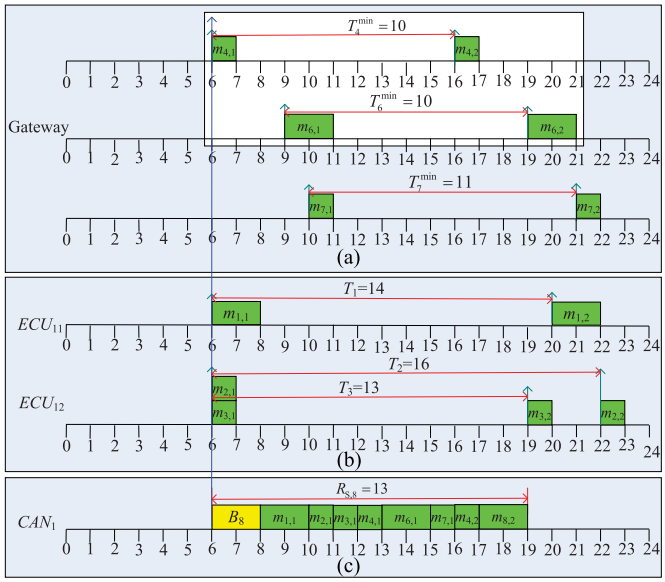


Fig. 9. Critical instant for m_8 in CAN_1. (a) Pessimistic worst case arriving order of m_4, m_6, and m_7. (b) m_1, m_2, and m_3 are released at instant 6 on ECUs. (c) $R_{S,8} = 13$ on CAN_1.

Proof: The arriving order $order^{\mathrm{W}}_{\mathrm{GW}}(i)$ (or $order^{\mathrm{PW}}_{\mathrm{GW}}(i)$) can cause successive interference to m_i between the arriving instant of $m_{\alpha,1}$ and the last instant of $m_{\gamma,1}$ according to Definition 2. Hence, when $CI_i = A(m_{\alpha,1})$ is the released instant, the minimum delay to m_i should be

$$w^{\min} = B_i + \sum_{\forall x \in shp(i)} C_x + \sum_{\forall y \in dhp_{\mathrm{GW}}(i)} C_y. \qquad (10)$$

■

Considering that the minimum delay to m_i is $w^{\min}$, we should also consider the expected interval between instants CI_i to $CI_i + w^{\exp}$, where $w^{\exp}$ is any expected delay meeting $w^{\exp} > w^{\min}$.

On the one hand, according to the proof of Theorem 3, the interval $w^{\exp}$ starting from CI_i contains a larger or equal number of arriving jobs in $dhp_{\mathrm{GW}}(i)$ than (to) the w^{exp} starting from any other instants. Hence, CI_i is the released instant from which the maximum interference from $dhp_{\mathrm{GW}}(i)$ can be obtained.

On the other hand, CI_i is also the released instant from which the maximum interference from $shp(i)$ can be obtained according to classical WCRT analysis theory [12].

Given that CI_i is the released instant that can cause the maximum interference to m_i from both $dhp_{\mathrm{GW}}(i)$ and $shp(i)$, it is the source-end critical instant of m_i. ■

For example, as shown in Fig. 8, the critical instant for m_7 in CAN_2 is 4, at which m_1, m_4, m_5, and m_6 are released simultaneously.

The critical instant for m_8 in CAN_1 is 6, at which m_4, m_1, m_2, and m_3 are released simultaneously, as shown Fig. 9.

B. WCRT Computation

Considering that the critical instant CI_i of m_i has been obtained in Section VI.A, we can obtain a safe WCRT bound using some methods. However, the messages in $dhp_{\mathrm{GW}}(i)$ are released sporadically; a period should be given if we use the method of maximum queuing delay (Section IV.B) to calculate the WCRT; if the minimum inter-arrival time of each message is considered as the period, then the obtained results are much pessimistic. To obtain tighter WCRT, we present an explorative method solving the problem.

The idea of calculating the $R_{\mathrm{S},i}$ using the explorative method aims to verify whether an expected delay (i.e., expected interval, which is started from CI_i) can be interfered by high-priority jobs from all the message jobs; these messages jobs include the message jobs of the worst case arriving order (i.e., $order^{\mathrm{W}}_{\mathrm{GW}}(i)$ or $order^{\mathrm{PW}}_{\mathrm{GW}}(i)$) in the gateway and of the high-priority messages jobs from the source-end $CAN_{\mathrm{S},i}$). Our main objective is to fill this interval with possible high-priority jobs from all these message jobs. For example, m_7 in its source-end bus CAN_2 will be interfered by m_1, m_2, and m_3 in $order^{\mathrm{W}}_{\mathrm{GW}}(7)$ from the gateway and by m_4, m_5, and m_6 from CAN_2. Considering that the minimum delay (i.e., $w^{\min}$) to m_i has been determined in (10) by employing obtained $CI_i = A(m_{\alpha,1})$, the expected delay should be explored from $w^{\min} + 1$ to $w^{\min} + 2$, $w^{\min} + 3$,... with increasing 1 time unit until it cannot be filled by more high-priority jobs. The explorative steps are as follows:

1) Set the initial expected delay to be $w^{\exp} = w^{\min} + 1$.
2) Explore whether the expected delay $w^{\exp}$ is reachable or not. For each high-priority message m_z in $shp(i) \cup dhp_{\mathrm{GW}}(i)$, the interference to m_i in the delay $w^{\exp}$ is I_i, which can be calculated by multiplying the released number of m_z in the expected interval $w^{\exp}$ by its transmission time C_z.
3) If $B_i + I_i \geqslant w^{\exp}$, then increase $w^{\exp}$ to $w^{\exp} = w^{\exp} + 1$. Repeat Step 2.
4) If $B_i + I_i < w^{\exp}$ (i.e., the $w^{\exp}$ is unreachable), then the maximum interfering delay is $w^{\exp} - 1$, and the source-

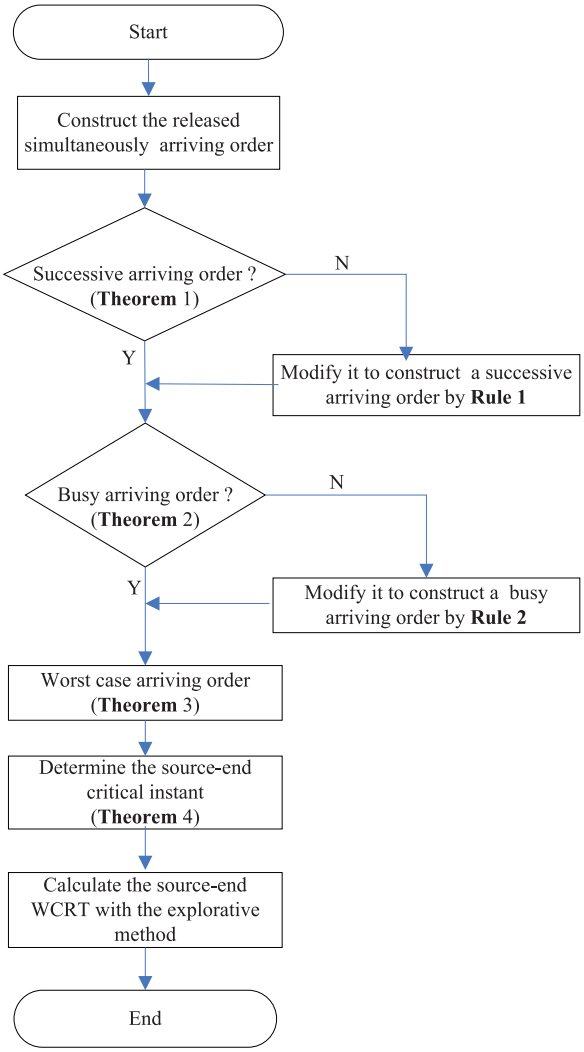


Fig. 10. Flow chart of calculating the source-end WCRTs of messages.

end WCRT of m_i should be

$$R_{\mathrm{S},i} = w^{\exp} - 1 + C_i.$$

We present the flow chart for the WCRT computation of the messages in their source-end in Fig. 10.

For example, $order^{\mathrm{W}}_{\mathrm{GW}}(7) = (m_{1,1}, m_{2,1}, m_{3,1})$ in Fig. 8(a) are released simultaneously with $m_{4,1}$, $m_{5,1}$, and $m_{6,1}$ in Fig. 8(b), then the obtained source-end WCRT for m_7 is $R_{\mathrm{S},7} = 10$, as shown in Fig. 8(c). Given that $order^{\mathrm{W}}_{\mathrm{GW}}(7)$ exists in the actual situation, the obtained result is the exact source-end WCRT for m_7. To obtain the source-end WCRT for m_8, $order^{\mathrm{PW}}_{\mathrm{GW}}(8) = (m_{4,1}, m_{6,1}, m_{7,1})$ in Fig. 9(a) are released simultaneously with $m_{1,1}$, $m_{2,1}$, and $m_{3,1}$ in Fig. 9(b). Finally, the obtained WCRT is $R_{\mathrm{S},8} = 10$, as shown in Fig. 9(c). Given that $order^{\mathrm{PW}}_{\mathrm{GW}}(8)$ does not exist in the actual situation, the obtained result is a pessimistic source-end WCRT for m_8.

C. Comparison With the State-of-the-Art Methods

To illustrate the difference between the explorative method of this study and the state-of-the-art methods [13], [14], we show

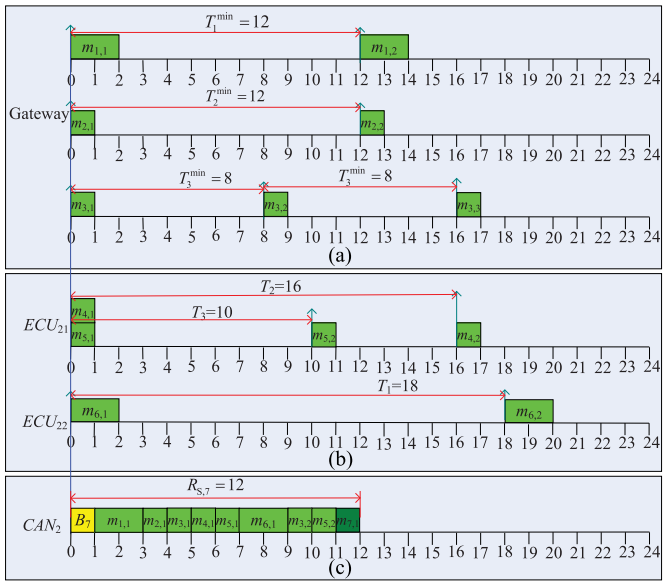


Fig. 11. Example of showing the source-end WCRT for m_7 with method in [13]. (a) m_1, m_2, and m_3 are released on the gateway. (b) m_4, m_5, and m_6 are released on ECUs. (c) $R_{S,7} = 12$ on CAN_2.

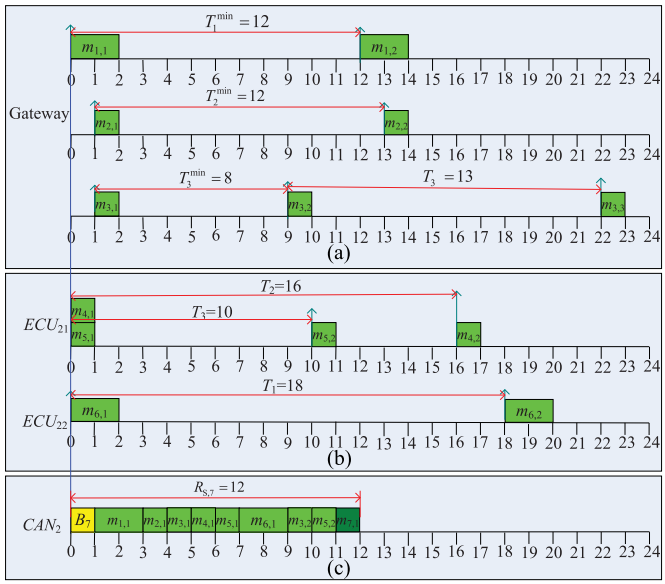


Fig. 12. Example of showing the source-end WCRT for m_7 with method in [14]. (a) m_1, m_2, and m_3 are released on the gateway. (b) m_4, m_5, and m_6 are released on ECUs. (c) $R_{S,7} = 12$ on CAN_2.

the source-end WCRT for m_7 with methods in [13] (Fig. 11) and [14] (Fig. 12), respectively.

We first consider the method of [13]. As shown in Fig. 11, the released messages (i.e., m_1, m_2, and m_3) in the gateway are released periodically with $T_1^{\min} = 12$, $T_2^{\min} = 12$, and $T_3^{\min} = 8$, respectively. The reason is that m_1, m_2, and m_3 are assumed released simultaneously in the gateway when using the method of [13]. As a result, $m_{3,2}$ interferes m_7, and then $m_{5,2}$ interferes m_7. Finally, the obtained source-end WCRT for m_7 is 12.

We then consider the method of [14]. As shown in Fig. 12, the released messages (i.e., m_1, m_2, and m_3) in the gateway are released with their busy sequences, respectively. Given that the

simplified search is used (i.e., for an arriving order, except for the first job, the other jobs arrive at the gateway simultaneously), the offsets for m_2 and m_3 are 1. As a result, $m_{3,2}$ interferes m_7, and then $m_{5,2}$ interferes m_7. Finally, the obtained source-end WCRT for m_7 is also 12. The source-end WCRT for m_7 with our proposed method is shown in Fig. 8. The released messages (i.e., m_1, m_2, and m_3) in the gateway are released with their busy sequences, respectively. Given that the worst case arriving order $order^{\mathrm{W}}_{\mathrm{GW}}(7) = (m_{1,1}, m_{2,1}, m_{3,1})$ is used, both the offsets for m_2 and m_3 are 1 and 2. As a result, both $m_{3,2}$ and $m_{5,2}$ cannot interfere m_7. Finally, the obtained source-end WCRT for m_7 is 10, which is a tighter WCRT than those obtained by [13] and [14].

D. Extending to the Destination-End

As mentioned earlier, the end-to-end WCRT $R_{\mathrm{E2E},i}$ of the gateway message m_i is the sum of its source-end WCRT $R_{\mathrm{S},i}$ and its destination-end WCRT $R_{\mathrm{D},i}$. The destination-end WCRT computation can use the similar arriving order and explorative method as the source-end. However, an important difference is that m_i itself is a gateway message and cannot be released simultaneously with any message in $shp_{\mathrm{GW}}(i)$. For example, assume that we are going to analyze the gateway message m_7. m_7 completes its transmission with WCRT value of 10 in its source-end bus CAN_2, and then is released in the gateway. m_4 and m_6 are high-priority gateway messages also released in CAN_2. Given that m_4, m_6, and m_7 arrive at the gateway with an arriving order, any two messages of them cannot be released simultaneously in the gateway. Therefore, the critical instant in the destination-end is the time instant when m_i is released simultaneously with messages in $dhp(i) = dhp_{\mathrm{GW}}(i) \cup dhp_{\mathrm{NGW}}(i)$. At the same time other messages in $shp_{\mathrm{GW}}(i)$ are released in the gateway with offsets.

Assume that a message job sequence consists of the first jobs of messages in $shp_{\mathrm{GW}}(i)$ is $(m_{s(1)}, m_{s(2)}, m_{s(3)}, ..., m_{s(m)})$, we can obtain the minimum offset ((11)) of each message in $shp_{\mathrm{GW}}(i)$) based on (7) in Theorem 1:

$$\begin{cases} O(m_{s(1)}) = 0 \\ O(m_{s(x)}) = O(m_{s(x-1)}) + C_{s(x)} \quad (x > 1). \end{cases} \tag{11}$$

For example, as shown in Fig. 13(b), the sequence is (m_7, m_4, m_6), and we have $O(m_7) = 0$, $O(m_4) = 0 + 1 = 1$, and $O(m_6) = 1 + 2 = 3$.

Without loss of generality, we use an example to illustrate how to calculate the destination-end WCRT for m_7 in Fig. 13. The main steps are given below.

1) The arriving order is (m_7, m_4, m_6), where m_7 should be the first arriving message, because the critical instant for m_7 is its released instant in the gateway, as shown in Fig. 13(b).
2) m_7 is only released simultaneously with m_1, m_2, and m_3 of CAN_1, as shown in Fig. 13(a).
3) (m_4, m_6) arrives at the gateway with the pessimistic worst case arriving order. As a result, the offsets for m_4 and m_6 are 1 and 3, respectively, as shown in Fig. 13(b).

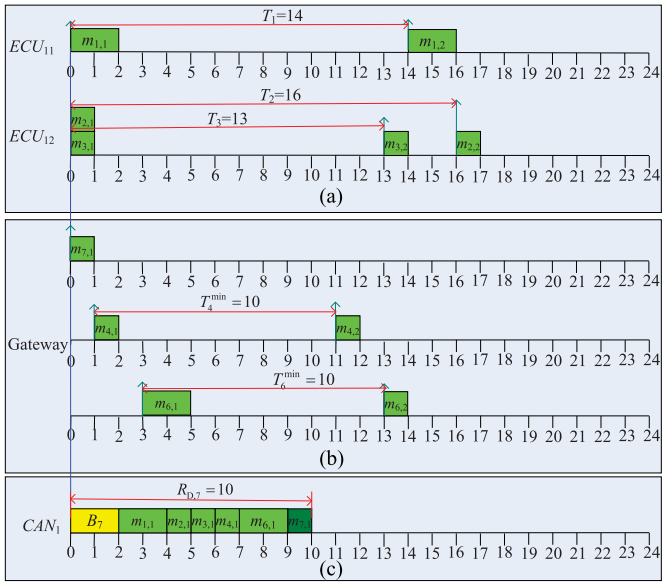


Fig. 13. Example of showing the destination-end WCRT for m_7 with the explorative method. (a) m_1, m_2, and m_3 are released at instant 0 on ECUs. (b) m_7, m_4, and m_6 are released on the gateway. (c) $R_{D,7} = 10$ on CAN_1.

TABLE III
COMPUTED WCRTS FOR THE MESSAGES OF THE EXAMPLE

Message		Analysis method of [13]			Analysis method of [14]			Analysis method of this study		
m_i	GW/NGW	$R_{\mathrm{S},i}$	$R_{\mathrm{D},i}$	$R_{\mathrm{E2E},i}$	$R_{\mathrm{S},i}$	$R_{\mathrm{D},i}$	$R_{\mathrm{E2E},i}$	$R_{\mathrm{S},i}$	$R_{\mathrm{D},i}$	$R_{\mathrm{E2E},i}$
m_1	GW	4	4	8	4	4	8	4	4	8
m_2	GW	5	5	10	5	5	10	5	5	10
m_3	GW	6	6	12	6	6	12	6	6	12
m_4	GW	7	7	14	7	7	14	7	7	14
m_5	NGW	8	–	8	8	–	8	8	–	8
m_6	GW	11	9	20	11	9	20	**10**	9	**19**
m_7	GW	12	13	25	12	13	25	**10**	**10**	**20**
m_8	NGW	24	–	24	20	–	20	**13**	–	**13**
m_9	NGW	16	–	16	16	–	16	16	–	16

4) The obtained destination-end WCRT for m_7 is 10, as shown in Fig. 13(c).
5) Given that the end-to-end WCRT $R_{\mathrm{E2E},i}$ of m_i is the sum of its source-end WCRT $R_{\mathrm{S},i}$ and its destination-end WCRT $R_{\mathrm{D},i}$, the obtained end-to-end WCRT is also safe.

Table III shows the source-end, destination-end, and end-to-end WCRTs for the messages of the examples, which are obtained by using different methods. As can be seen, the proposed explorative method can obtain tighter WCRT bounds than the state-of-the-art methods of [13] and [14]. For example, the deadlines of m_7 and m_8 are 20 and 14, respectively (Table I). If we use the methods of [13] and [14] to calculate the WCRTs, both m_7 and m_8 are unschedulable, because $R_{\mathrm{E2E},7} = 25 > 20$ and $R_{\mathrm{E2E},8} = 24 > 14$ using the method of [13], and $R_{\mathrm{E2E},7} = 25 > 20$ and $R_{\mathrm{E2E},8} = 20 > 14$ using the method of [14]. However, m_6 and m_7 are schedulable if we use the proposed method of this study, because $R_{\mathrm{E2E},7} = 20 = 20$ and $R_{\mathrm{E2E},8} = 13 < 14$.

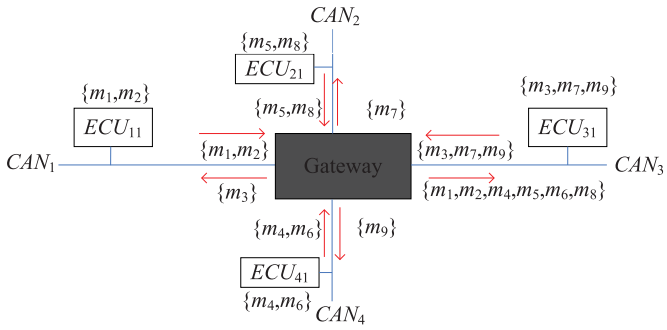


Fig. 14. Example of CAN clusters with four buses.

E. Extending to CAN Clusters

In todays cars, the trend towards increased bandwidth requirements and leds to the use of several CAN clusters of in-vehicle networks [9], [34]–[36], in which more than four or five CAN buses are interconnected by a gateway [37]. Fig. 14 shows an example of CAN clusters with four buses. The differences between dual-buses and CAN clusters are as follows:

1) The gateway messages released in the source-end bus only be transferred to one destination bus through the gateway in dual-buses, whereas the gateway messages released in one CAN buses can be transferred to multiple buses through the gateway in a CAN cluster. For example, the gateway messages released in CAN_3 are transferred to CAN_1 (m_3), CAN_2 (m_7), and CAN_4 (m_9).
2) Only one worst case arriving order (or pessimistic worst case arriving order) exists for the messages under analysis in dual-buses, whereas for CAN clusters with n buses, $n-1$ worst case arriving order (or pessimistic worst case arriving order) exist. For example, if the message under analysis m_9 released in CAN_3 in Fig. 14, the worst case arriving orders are (m_1, m_2) from CAN_1, (m_5, m_8) from CAN_2, and (m_4, m_6) from CAN_4, respectively.

Although the differences exists, our proposed method can still be applied to CAN clusters, only if we assume that all worst case arriving orders arrive at the gateway simultaneously. Even if this assumption may not exit in the actual situation, it is a simple and safe WCRT analysis method.

VII. Experiments

To evaluate the applicability and scalability of the proposed method, we used a real CAN message set provided by automaker [14]. The message set contains 65 messages that are assigned into 14 ECUs, and the bus load is about 53% in 500 Kbps. These messages are divided into two CAN buses, where The first CAN bus has 32 messages, including 16 gateway messages and 16 non-gateway messages in CAN_1. The second CAN bus contains 33 messages, including 17 gateway messages and 16 non-gateway messages in CAN_2. The time unit of each message is 1 μs, which is the minimum time unit of CAN messages of in-vehicle networks as explained earlier. The size of each message belongs to the set of { 150 μs, 170 μs, 190 μs, 210 μs, 230 μs, 250 μs, 270 μs } in 500 Kbps CAN bus. We implemented a tool using Java on a standard desktop computer (2.6 GHz Intel CPU and 4 GB memory) to calculate the WCRT of each message.

TABLE IV
WCRTs (μs) on Small-Scale Gateway-Integrated CANs

Small-scale message set $\|MS\| = 6510,000\ \mu s \leqslant T_i \leqslant 1,000,000\ \mu s$ $150\ \mu s \leqslant C_i \leqslant 270\ \mu s$					
m_i	$R^1_{\text{E2E},i}$ [13]	$R^2_{\text{E2E},i}$ [14]	$R^3_{\text{E2E},i}$	RP_1	RP_2
m_{1013}	13010	13010	9760	24.98%	24.98%
m_{982}	9830	9830	8310	15.46%	15.46%
m_{985}	21140	21140	17990	14.90%	14.90%
m_{980}	17630	17630	15880	9.93%	9.93%
m_{953}	8280	8030	7240	12.56%	9.84%

As mentioned in Section II, the preceding research before [12] may obtain a response value that is less than the exact WCRT. In this case, the safety of the obtain response value cannot be ensured. Therefore, the preceding research before [12] are not suitable for comparison due to their unsafe results. In addition, most literatures except [13] and [14], focus on the estimation of WCRT of CAN messages on a single CAN bus. To the best of knowledge, [13] and [14] are the state-of-the-art studies in WCRT analysis methods for CAN messages in gateway-integrated in-vehicle networks. Therefore, our method is compared with [13] and [14].

Experiment 1: This experiment is to compare the obtained WCRTs of messages using different methods on small-scale gateway-integrated CANs ($|MS| = 65$, 10,000 $\mu s \leqslant T_i \leqslant 1,000,000\ \mu$s, 150 $\mu s \leqslant C_i \leqslant 270\ \mu$s). We use $R^1_{\text{E2E},i}$, $R^2_{\text{E2E},i}$ and $R^3_{\text{E2E},i}$ to represent the results obtained using methods presented in [13], [14], and in this study, respectively. The reduced percentage of the pessimism is calculated by

$$RP_1 = (R^1_{\text{E2E},i} - R^3_{\text{E2E},i})/R^1_{\text{E2E},i}$$

and

$$RP_2 = (R^2_{\text{E2E},i} - R^3_{\text{E2E},i})/R^2_{\text{E2E},i},$$

respectively.

Table IV shows the top 5 end-to-end WCRTs of the messages according to the descending orders of RP_1 and RP_2. We can see that the computed WCRT of the explore algorithm are always less than that of the methods of [13] and [14]. Although we only list the top five tests, the other tests demonstrate the same regular pattern as these tests. That is, tighter WCRT bound are obtained using our proposed method than using the state-of-the art methods. As shown in the Table IV, the proposed method can reduce the WCRT pessimism as much as 24.98%.

Experiment 2: This experiment is to show the performance of the proposed method on large-scale gateway-integrated CANs. The message set is generated from the original small-scale message set, and has four times the number of messages as in the latter ($|MS| = 260$, 40,000 $\mu s \leqslant T_i \leqslant 4,000,000\ \mu$s, 150 $\mu s \leqslant C_i \leqslant 270\ \mu$s).

Table V shows the top 5 end-to-end WCRTs of the messages according to the descending orders of RP_2. In the experimental results, for any message m_i with R_{E2E} results, they meet

TABLE V
WCRTs (μs) ON LARGE-SCALE GATEWAY-INTEGRATED CANs

Large-scale message set $\|MS\| = 26040,000$ μs $\leqslant T_i \leqslant 4,000,000$ μs, 150 μs $\leqslant C_i \leqslant 270$ μs					
m_i	$R^1_{\text{E2E},i}$ [13]	$R^2_{\text{E2E},i}$ [14]	$R^3_{\text{E2E},i}$	RP_1	RP_2
m_{4977}	–	51430	39280	–	23.62%
m_{4980}	–	103060	89660	–	13.00%
m_{4929}	–	88120	76710	–	12.94%
m_{3928}	28350	28350	25310	10.72%	10.72%
m_{3930}	28730	28730	25690	10.58%	10.58%

TABLE VI
WCRTs (μs) ON LARGE-SCALE CAN CLUSTERS WITH FOUR BUSES

Large-scale message set $\|MS\| = 52040,000$ μs $\leqslant T_i \leqslant 4,000,000$ μs, 150 μs $\leqslant C_i \leqslant 270$ μs					
m_i	$R^1_{\text{E2E},i}$ [13]	$R^2_{\text{E2E},i}$ [14]	$R^3_{\text{E2E},i}$	RP_1	RP_2
m_{9970}	–	61550	46240	–	24.87%
m_{8980}	–	113060	92760	–	17.96%
m_{6932}	–	42730	35690	–	16.47%
m_{8970}	–	92820	80260	–	13.53%
m_{7938}	–	66740	58810	–	11.88%

$R^1_{\text{E2E},i} \geqslant R^2_{\text{E2E},i} \geqslant R^3_{\text{E2E},i}$, which indicates that the explorative method can generate tighter WCRT bound than the state-of-the-art methods. For the large-scale message set, the method proposed in [13] cannot work out the concrete results of partial messages (denoted with "-", e.g., m_{4977}, m_{4980}, and m_{4929}). The reason is that the period of any gateway message m_y is assumed to be $T_y^{\min}$ and the iteration of (3) cannot stop until the message is not schedulable. As can be seen, the WCRT of a low-priority message (m_{4977}) can be reduced by 23.62% using our method.

The above two experiments show that different scale message set has roughly the same trend in pessimism and reveal that our method can reduce as much as 24% WCRT comparing with the state-of-the-art methods.

Experiment 3: This experiment is to show the capability of the proposed method to deal with CAN clusters with multiple buses. The message set is generated from the original small-scale message set, and has eight times the number of messages as in the latter ($|MS| = 520$, 40,000 μs $\leqslant T_i \leqslant 4,000,000$ μs, 150 μs $\leqslant C_i \leqslant 270$ μs).

To our knowledge, there is no WCRT analysis method for CAN clusters; however, the method described in VI.E can also be applied to [13] and [14] to deal with CAN clusters.

Table VI shows the top five end-to-end WCRTs of the messages according to the descending orders of RP_2. In the experimental results, the obtained WCRTs of any message m_i meet $R^1_{\text{E2E},i} \geqslant R^2_{\text{E2E},i} \geqslant R^3_{\text{E2E},i}$. The method proposed in [13] cannot work out the concrete results of top five messages. The reason is explained earlier. As can be seen, the WCRT of a low-priority message (m_{9970}) can be reduced by 24.87% using our method.

The above experiment demonstrates that our method can deal with large-scale and multiple CAN buses and can also obtain tighter WCRT than the state-of-the-art methods in this environments.

TABLE VII
COMPUTATIONAL TIME (MS) OF DIFFERENT SCALE MESSAGE SETS

	The state-of-the-art method of [14]	The explorative method of this study
Small-scale dual-buses with $\|MS\| = 65$	163	19
Large-scale dual-buses with $\|MS\| = 260$	308881	1011
Large-scale CAN cluster with $\|MS\| = 520$	710542	2094

Experiment 4: Besides the pessimism of WCRT bound, computational time is also an important factor. Table VII shows the computational times of the above different scale message sets. We can see that our method consumes much less time than the state-of-the-art method of [14] in all the different scale message sets. Moreover, with the increase of the message set scale, the state-of-the-art method is quite time-consuming, while our method is still very short. For example, for the large-scale message set with 520 messages on CAN clusters, the state-of-the-art method needs 710542 ms (nearly 12 minutes), while our method is merely 2094 ms (about 2 s). These results indicates that our method is applicable even for large-scale CAN clusters.

VIII. CONCLUSION

We have developed an explorative WCRT computation method for messages of gateway-integrated CANs by considering the actual possible arriving order of gateway message to obtain the worst case arriving order or the pessimistic arriving order. We prove that the obtained WCRT results are safe WCRT bounds. Experimental results of real message sets reveal that our method can reduce as much as 24% WCRT comparing with the state-of-the-art methods on different scale message sets. Furthermore, our method is a heuristic low complexity method and is applicable even for large-scale CAN cluster. We believe that our results could improve the gateway-integrated in-vehicle networks design via reducing the unnecessarily conservation during design phase. Because the proposed WCRT estimation method is based on single domain (source-end or destination-end) analysis, the accuracy of estimation can be improved further. A future research direction would be to develop a holistic analysis method considering source-end and destination-end together for tighter WCRT upper bound estimation.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the anonymous reviewers whose constructive comments have helped to improve the manuscript.

REFERENCES

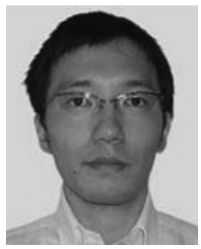
[1] R. Obermaisser, C. El Salloum, B. Huber, and H. Kopetz, "From a federated to an integrated automotive architecture," *IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst.*, vol. 28, no. 7, pp. 956–965, Jul. 2009.

[2] M. D. Natale and A. Sangiovanni-Vincentelli, "Moving from federated to integrated architectures in automotive: The role of standards, methods and tools," *Proc. IEEE*, vol. 98, no. 4, pp. 603–620, Mar. 2010.

[3] E. Azketa, J. J. Gutiérrez, J. C. Palencia, M. G. Harbour, L. Almeida, and M. Marcos, "Schedulability analysis of multi-packet messages in segmented CAN," in *Proc. 17th Conf. Emerging Technol. Factory Autom.*, 2012, pp. 1–8.
[4] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in automotive communication systems," *Proc. IEEE*, vol. 93, no. 6, pp. 1204–1223, Jun. 2005.
[5] R. Bosch, "Can specification version 2.0," *Rober Bousch GmbH, Postfach*, vol. 300240, 1991.
[6] Z. Shuai, H. Zhang, J. Wang, J. Li, and M. Ouyang, "Combined AFS and DYC control of four-wheel-independent-drive electric vehicles over CAN network with time-varying delays," *IEEE Trans. Veh. Technol.*, vol. 63, no. 2, pp. 591–602, Feb. 2014.
[7] S. Tuohy, M. Glavin, C. Hughes, E. Jones, M. Trivedi, and L. Kilmartin, "Intra-vehicle networks: A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 534–545, Apr. 2015.
[8] M. Sojka, P. Písa, O. Spinka, and Z. Hanzálek, "Measurement automation and result processing in timing analysis of a linux-based CAN-to-CAN gateway," in *Proc. IEEE 6th Int. Conf. Intell. Data Acquisition. Adv. Comput. Syst.*, vol. 2., 2011, pp. 963–968.
[9] G. Xie, G. Zeng, R. Kurachi, H. Takada, and R. Li, "Gateway modeling and response time analysis on CAN clusters of automobiles," in *Proc. IEEE 17th Int. Conf. High Perform. Comput. Commun.*, 2015, pp. 1147–1153.
[10] J. Sommer, L. Burgstahler, and V. Feil, "An analysis of automotive multi-domain CAN systems," *Proc. 12th EUNICE Open Eur. Summer School*, 2006, pp. 1–8.
[11] N. Guan, C. Gu, M. Stigge, Q. Deng, and W. Yi, "Approximate response time analysis of real-time task graphs," in *Proc. IEEE Real-Time Syst. Symp.*, 2015, pp. 304–313.
[12] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien, "Controller area network (CAN) schedulability analysis: Refuted, revisited and revised," *Real-Time Syst.*, vol. 35, no. 3, pp. 239–272, Apr. 2007.
[13] R. I. Davis, S. Kollmann, V. Pollex, and F. Slomka, "Schedulability analysis for controller area network (CAN) with FIFO queues priority queues and gateways," *Real-Time Syst.*, vol. 49, no. 1, pp. 73–116, Jan. 2013.
[14] X. Yong, Z. Gang, C. Yang, R. Kurachi, H. Takada, and L. Renfa, "Worst case response time analysis for messages in controller area network with gateway," *IEICE Trans. Inf. Syst.*, vol. 96, no. 7, pp. 1467–1477, Jul. 2013.
[15] G. Xie, Y. Chen, Y. Liu, Y. Wei, R. Li, and K. Li, "Resource consumption cost minimization of reliable parallel applications on heterogeneous embedded systems," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1629–1640, Aug. 2017.
[16] M. Stigge and W. Yi, "Combinatorial abstraction refinement for feasibility analysis," in *Proc. IEEE 34th Real-Time Syst. Symp.*, 2013, pp. 340–349.
[17] M. Joseph and P. Pandya, "Finding response times in a real-time system," *Comput. J.*, vol. 29, no. 5, pp. 390–395, Jan. 1986.
[18] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *J. ACM*, vol. 20, no. 1, pp. 46–61, Jan. 1973.
[19] J. P. Lehoczky, "Fixed priority scheduling of periodic task sets with arbitrary deadlines." in *Proc. 11th IEEE Real-Time Syst. Symp.*, vol. 90, 1990, pp. 201–209.
[20] M. Gonzalez, H. Mark, H. Klein, and J. P. Lehoczky, "Fixed priority scheduling of periodic tasks with varying execution priority," in *Proc. 12th IEEE Real-Time Syst. Symp.*, 1991, pp. 116–128.
[21] K. Tindell and A. Burns, "Guaranteeing message latencies on control area network (CAN)," in *Proc. 1st Int. CAN Conf.*, 1994, pp. 1–11.
[22] K. Tindell, H. Hansson, and A. J. Wellings, "Analysing real-time communications: Controller area network (CAN)," in *Proc. Real-Time Syst. Symp.*, 1994, pp. 259–263.
[23] K. Tindell, A. Burns, and A. J. Wellings, "Calculating controller area network (CAN) message response times," *Control Eng. Pract.*, vol. 3, no. 8, pp. 1163–1169, Aug. 1995.
[24] H. Takada and K. Sakamura, "Schedulability of generalized multiframe task sets under static priority assignment," in *Proc. 14th Int. Workshop Real-Time Comput. Syst. Appl.*, 1997, pp. 80–86.
[25] Y. Chen, R. Kurachi, G. Zeng, and H. Takada, "The worst-case response time analysis for fifo-based offset assigned CAN messages," *J. Inf. Process.*, vol. 20, no. 2, pp. 451–462, Jul. 2012.
[26] S. Mubeen, J. Maki-Turja, and M. Sjodin, "Worst-case response-time analysis for mixed messages with offsets in controller area network," in *Proc. IEEE 17th Conf. Emerging Technol. Factory Autom.*, 2012, pp. 1–10.
[27] H. Zeng, M. Di Natale, P. Giusto, and A. Sangiovanni-Vincentelli, "Stochastic analysis of CAN-based real-time automotive systems," *IEEE Trans. Ind. Informat.*, vol. 5, no. 4, pp. 388–401, Sep. 2009.
[28] H. Zeng, M. D. Natale, P. Giusto, and A. Sangiovanni-Vincentelli, "Using statistical methods to compute the probability distribution of message response time in controller area network," *IEEE Trans. Ind. Informat.*, vol. 6, no. 4, pp. 678–691, Nov. 2010.
[29] T. Herpel, K.-S. Hielscher, U. Klehmet, and R. German, "Stochastic and deterministic performance evaluation of automotive CAN communication," *Comput. Netw.*, vol. 53, no. 8, pp. 1171–1185, Jun. 2009.
[30] R. Queck, "Analysis of ethernet AVB for automotive networks using network calculus," in *Proc. IEEE Int. Conf. Veh. Electron. Saf.*, Jul. 2012, pp. 61–67.
[31] H. Charara, J.-L. Scharbarg, J. Ermont, and C. Fraboul, "Methods for bounding end-to-end delays on an AFDX network," in *Proc. 18th Euromicro Conf. Real-Time Syst.*, 2006, pp. 10–19.
[32] J. Taube, F. Hartwich, and H. Beikirch, "Comparison of CAN gateway modules for automotive and industrial control applications," in *Proc. 10th Int. CAN Conf.*, 2005, pp. 1–8.
[33] K. Schmidt and E. G. Schmidt, "Systematic message schedule construction for time-triggered CAN," *IEEE Trans. Veh. Technol.*, vol. 56, no. 6, pp. 3431–3441, Nov. 2007.
[34] N. Navet, S. Louvart, J. Villanueva, S. Campoy-Martinez, and J. Migge, "Timing verification of automotive communication architectures using quantile estimation," in *Proc. Eur. Congr. Embedded Real-Time Softw. Syst.*, 2014, pp. 1–10.
[35] G. Xie, G. Zeng, L. Liu, R. Li, and K. Li, "High performance real-time scheduling of multiple mixed-criticality functions in heterogeneous distributed embedded systems," *J. Syst. Archit.*, vol. 70, pp. 3–14, Oct. 2016.
[36] G. Xie, G. Zeng, Z. Li, R. Li, and K. Li, "Adaptive dynamic scheduling on multi-functional mixed-criticality automotive cyber-physical systems," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 6676–6692, Aug. 2017.
[37] N. Navet and H. Perrault, "CAN in automotive applications: A look forward," *Electron. World*, vol. 119, no. 1924, pp. 60–63, Apr. 2013.

Guoqi Xie (M'15) received his Ph.D. degree in computer science and engineering from Hunan University, China, in 2014. He is currently an Assistant Professor and a Yuelu Scholar at Hunan University. He was a Postdoctoral Researcher at Nagoya University, Japan, from 2014 to 2015, and at Hunan University, from 2015 to 2017. He has received the best paper award from ISPA 2016. His major interests include embedded and cyber-physical systems, parallel and distributed systems, software engineering, and methodology. He is a member of IEEE, ACM, and CCF.

Gang Zeng (M'03) received the Ph.D. degree in information science from Chiba University, Chiba, Japan, in 2006. He is currently an Associate Professor at the Graduate School of Engineering, Nagoya University, Nagoya, Japan. From 2006 to 2010, he was a Researcher, and then an Assistant Professor at the Center for Embedded Computing Systems (NCES), the Graduate School of Information Science, Nagoya University. His research interests mainly include power-aware computing and real-time embedded system design. He is a member of IPSJ.



Ryo Kurachi received the Ph.D. degree in information science from Nagoya University, Nagoya, Japan, in 2012. He is a Designated Assistant Professor at the Graduate School of Information Science, Nagoya University. From 2000 to 2006, he was working for AISIN AW CO., Ltd. Since 2007, he has been a Researcher of the Center for Embedded Computing Systems, Nagoya University. His research interests include in-vehicle networks and real-time scheduling theory.

Hiroaki Takada received the Ph.D. degree in information science from the University of Tokyo, Tokyo, Japan, in 1996. He is a Professor at the Graduate School of Information Science, Nagoya University. He is also the Executive Director of the Center for Embedded Computing Systems. He received the Ph.D. degree in information science from the University of Tokyo, Tokyo, Japan, in 1996. He was a Research Associate at the University of Tokyo from 1989 to 1997, and was a Lecturer and then an Associate Professor at Toyohashi University of Technology, Japan, from 1997 to 2003. His research interests include real-time operating systems, real-time scheduling theory, and embedded system design. He is a member of ACM, IPSJ, IEICE, JSSST, and JSAE.

Zhetao Li received the Ph.D. degree in computer science from Hunan University, Changsha, China, in 2010. He is a Professor at Hunan University and Xiangtan University, Xiangtan, China. His main research interests include in-vehicle networks, Internet of things (IOT), compressive sensing, and social computing.

Renfa Li (M'05–SM'10) received the Ph.D. degree in electronic engineering from Huazhong University of Science and Technology, Wuhan, China, in 2003. He is a Professor of computer science and electronic engineering, and the Dean of College of Computer Science and Electronic Engineering, Hunan University, China. He is the Director of the Key Laboratory for Embedded and Network Computing of Hunan Province, China. His major interests include computer architectures, embedded computing systems, cyber-physical systems, and Internet of things. He is a member of the council of CCF, and a senior member of ACM.

Keqin Li (M'90–SM'96–F'15) received the Ph.D. degree in computer science from the University of Houston, Houston, TX, USA, in 1990. He is a SUNY Distinguished Professor of computer science. His current research interests include parallel computing and high-performance computing, distributed computing, energy-efficient computing and communication, heterogeneous computing systems, cloud computing, big data computing, CPU–GPU hybrid and cooperative computing, multicore computing, storage and file systems, wireless communication networks, sensor networks, peer-to-peer file sharing systems, mobile computing, service computing, Internet of things and cyber-physical systems. He has published more than 470 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He is currently or has served on the editorial boards of the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON CLOUD COMPUTING, the IEEE TRANSACTIONS ON SERVICES COMPUTING, the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING.