Frontiers of Information Technology & Electronic Engineering
www.zju.edu.cn/jzus; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

A rectangle bin packing optimization approach to the signal scheduling problem in the FlexRay static segment*

Rui ZHAO‡[1], Gui-he QIN[1], Jia-qiao LIU[2]

([1]*Department of Computer Science and Technology, Jilin University, Changchun 130000, China*)

([2]*Technology Development Department, FAW-Volkswagen Automotive Company Ltd., Changchun 130000, China*)

E-mail: zhaor1022@163.com; qingh@jlu.edu.cn; jiaqiao.liu@faw-vw.com

Received July 21, 2015; Revision accepted Dec. 14, 2015; Crosschecked Feb. 26, 2016

Abstract: As FlexRay communication protocol is extensively used in distributed real-time applications on vehicles, signal scheduling in FlexRay network becomes a critical issue to ensure the safe and efficient operation of time-critical applications. In this study, we propose a rectangle bin packing optimization approach to schedule communication signals with timing constraints into the FlexRay static segment at minimum bandwidth cost. The proposed approach, which is based on integer linear programming (ILP), supports both the slot assignment mechanisms provided by the latest version of the FlexRay specification, namely, the single sender slot multiplexing, and multiple sender slot multiplexing mechanisms. Extensive experiments on a synthetic and an automotive X-by-wire system case study demonstrate that the proposed approach has a well optimized performance.

Key words: FlexRay, Real-time applications, Rectangle bin packing, Schedule optimization, Slot multiplexing
http://dx.doi.org/10.1631/FITEE.1500232 **CLC number:** TP393

1 Introduction

To meet the growing customer demands for safe and intelligent vehicles, today's automobiles use several electronic control units (ECUs) to execute various types of distributed applications. These ECUs require signal exchanges among each other via in-vehicle networks to support their functions. Currently, the most popular in-vehicle network protocol is the control area network (CAN) (Robert Bosch GmbH, 1991). However, because of its low data rate and event-triggered nature (Navet *et al*., 2005), CAN is incompatible with the latest real-time applications, such as X-by-wire (Bertoluzzo *et al*., 2004), which require predictable and high bandwidth communication. Therefore, the FlexRay protocol (International Organization for Standardization, 2013) has been developed by a consortium of automotive manufacturers and suppliers to address the aforementioned issues. FlexRay provides both time-triggered static and event-triggered dynamic segments, and offers a bandwidth of 10 Mb/s. This communication protocol is expected to be the core of the next-generation in-vehicle communication networks.

Similar real-time applications will be available in the future (Lee *et al*., 2003), and the amount of signal data on the FlexRay static segment will increase significantly. Therefore, the optimal signal scheduling that satisfies the timing constraints of each signal with minimum bandwidth cost becomes a critical issue in guaranteeing the operation of time-critical applications and in complying with high data volume demands of future automotive.

Communication in FlexRay takes place over a set of periodic cycles. Each cycle contains four segments, namely, static (ST) segment, dynamic (DYN) segment, symbol window (SW), and network idle time (NIT). ST segment employs the time-division

‡ Corresponding author

* Project supported by the Program for Changjiang Scholars and Innovative Research Team in the University of Ministry of Education of China (No. IRT1017)

ORCID: Rui ZHAO, http://orcid.org/0000-0002-3757-8047

multiple access (TDMA) scheme and is composed of a set of equal length slots. Signal scheduling in the ST segment involves assigning a special TDMA slot (identified by the pair of slot number and cycle counter number) for every instance of every signal according to both the timing and FlexRay protocol-related constraints.

To support the efficient use of an ST slot, FlexRay provides a proprietary slot multiplexing mechanism that allows the alteration of frame contents being sent to this slot from cycle to cycle. Specifically, FlexRay 3.0 allows an ECU to configure a transmit/receive buffer for a set of slots sharing the same slot identifier in a configurable set of communication cycles. The configuration of the set of communication cycles is supported by the cycle counter filter criterion, which could cover all cycles. Therefore, FlexRay allows configuration of multiple buffers for the same slot, where each buffer corresponds to a frame content, thus enabling slot multiplexing. Slot multiplexing can be classified into single sender slot multiplexing and multiple sender slot multiplexing. This classification is based on whether the FlexRay slots of the same slot identifier but different cycle numbers are allowed for different ECUs.

Several studies on optimal scheduling in the FlexRay network have been performed. Pop *et al.* (2008) proposed a method to determine the timing properties of messages transmitted to FlexRay and optimization techniques to define a FlexRay bus parameter configuration that can guarantee that all time constraints of the messages are satisfied. Park and Sunwoo (2011) proposed another FlexRay network parameter optimization method to determine the lengths of the static slot and the communication cycle. Hu *et al.* (2014) proposed a holistic scheduling algorithm to handle real-time applications in a FlexRay network that schedules tasks and messages in a flexible way to enhance schedulability. Hua *et al.* (2014) proposed a holistic scheduling scheme to schedule a mixture of periodic and aperiodic tasks to guarantee that all periodic deadlines are met and that the response time for the aperiodic tasks can be as small as possible in the FlexRay network. However, schedule optimization that complies only with the timing constraints of signals is not sufficient due to the considerable increase in the volume of signal data on the FlexRay bus.

Similar to other time-triggered communication protocols, all ST slots constantly have the same length that is independent of the slot content. To optimize the use of FlexRay bandwidth, signals with different transmission periods are allowed to pack into the same slot in FlexRay. Based on this idea, the optimal scheduling that meets the time constraints of signals while minimizing bandwidth cost has been studied. Schmidt and Schmidt (2009) provided the frame packing and message scheduling methods. In their study, signals are first packed into message frames while maximizing the utilization; the obtained messages are then scheduled in the FlexRay ST segment while using a minimum number of slots. Kang *et al.* (2013) developed a frame packing algorithm to minimize the bandwidth consumption of the FlexRay ST segment. Tanasa *et al.* (2011) proposed a reliable frame packing method to ensure that none of the signals violate their deadlines while maintaining the desired reliability goal at minimum bandwidth cost. Zeng *et al.* (2011) presented an optimization framework that includes signal-to-frame packing and frame-to-slot assignment and task schedule, with the goal of minimizing the number of slots used. However, all the aforementioned studies merely focused on traditional signals packing and neglected the slot multiplexing mechanism, which is the most effective measure to improve the use efficiency of the FlexRay ST segment.

Recent studies have considered using the slot multiplexing mechanism on signal scheduling of FlexRay to further improve bandwidth utilization. Tanasa *et al.* (2012) proposed a new approach for the timing analysis of the event-triggered DYN segment while accounting for slot multiplexing. Schneider *et al.* (2011) proposed definitions for both sustainability and extensibility from the FlexRay slot multiplexing perspective. However, both studies (Schneider *et al.*, 2011; Tanasa *et al.,* 2012) considered the use of slot multiplexing only in the FlexRay DYN segment. Lukasiewycz *et al.* (2009) applied single slot multiplexing to the scheduling of the ST segment with the objective of increasing the utilization of the FlexRay bus. Grenier *et al.* (2008) studied the configuration of the FlexRay network ST segment. They provided solutions to verify the freshness constraints of the signals exchanged in the ST segment and then proposed a method to construct the optimal

communication schedule using the single slot multiplexing mechanism. However, these studies neglected the more efficient multiple sender slot multiplexing mechanism (i.e., the FlexRay slots of the same slot identifier but different cycle numbers can be assigned to different ECUs), thereby severely limiting FlexRay performance on large-scale systems. To follow the upward trend of data volume growth on automotive networks, FlexRay 3.0 has improved the communication slot assignment mechanism, which allows multiple sender slot multiplexing to be used in the ST segment of FlexRay.

Moreover, in most of the abovementioned studies, the aspect of signal optimization scheduling was based on the assumption that the signal offset is 0. However, such an assumption is not in accordance with the actual situation of automotive applications; i.e., the time of signal is constrained by both lower bound (offset) and upper bound (deadline).

In this study, we propose a rectangle bin packing optimization approach to schedule communication signals into the FlexRay ST segment. Our approach computes an optimal communication slot assignment which ensures that each of the signals complies with its lower and upper bounds at minimum bandwidth cost. The proposed approach supports both the slot assignment mechanisms provided by the latest version of the FlexRay specification, including single slot sender multiplexing and multiple sender slot multiplexing, and signal packing. This approach is based on the integer linear programming (ILP) formulation. To date, this is the first study to use the multiple sender slot multiplexing technique on the optimal scheduling in the FlexRay ST segment.

2 Problem formulation

2.1 System model

The target model is a typical time-triggered in-vehicle system consisting of ECUs $\{E_1, E_2, \ldots, E_N\}$ connected by the FlexRay bus. Each ECU E_p is composed of a host, a FlexRay communication controller (CC), and a controller-host interface (CHI) between them (Fig. 1a). The CC runs independently of the host and implements the FlexRay protocol services. The host processes time-triggered particular tasks that exchange data signals via messages transferred on the bus.

For the system, we denote $S=\{s_1, s_2, \ldots, s_{n_s}\}$ the set of signals to be sent on the bus. Each signal s_i is characterized by the tuple $(E_i, B_i, T_i, O_i, D_i)$, where E_i is the identifier of the ECU that produces the signal instance, B_i the size in byte, T_i the period, O_i the offset that is the latest time at which its first instance is produced and the offset is expressed with regard to the start of the first FlexRay communication cycle, and $D_i \le T_i$ the signal deadline, which is the maximum allowable duration between the production of the signal on the sender side and the completion of the frame transmission carrying this signal on the FlexRay bus. Observing that all signals must be scheduled in multiples of the FlexRay cycle duration, it is necessary to use the FlexRay cycle as the greatest common divisor (GCD) of the signal periods or an integer divisor of that value.

We further define the application cycle H_{app} of the entire time-triggered system as the least common multiple (LCM) of the periods of all signals. It is sufficient to analyze the behavior of the signal scheduling in the FlexRay ST segment in only one application cycle, on the assumption that the first instance of each signal is ready for transmission before the first bus cycle. However, we consider that the signal has an offset relative to the beginning of the first FlexRay cycle in accordance with the actual situation of the automotive applications. The allowable latest time instant O_i+D_i of the first instance of a signal s_i, which is relative to the start of the first FlexRay cycle, may be greater than that of the application cycle H_{app}. Therefore, we should analyze the signal scheduling within the hyperperiod H, which is expressed as a multiple of the application cycle, with the constraints as follows:

$$H = \min\left\{n \cdot H_{\text{app}} | n \in \mathbb{N}, n \cdot H_{\text{app}} \ge \max_{s_i \in S}\{O_i + D_i\}\right\}. \tag{1}$$

In one hyperperiod H, each signal s_i occurs H/T_i times. Let s_i^j denote the jth instance of each signal s_i. The signal scheduling problem consists of determining the transmitting time on the FlexRay bus for every instance s_i^j of each signal s_i during one hyperperiod H.

2.2 FlexRay communication protocol

This section describes the entire process in which the signals generated by the host are transmitted to the FlexRay bus. Consider the example in Fig. 1 where three ECUs, E_1, E_2, and E_3, send signals s_1, s_2, s_3, s_4, and s_5 using a FlexRay bus. The parameters of these signals are listed in the overall set of the signals (Fig. 1b). Given these parameters, the values of application cycle and hyperperiod are 4 ms and 8 ms, respectively.

The communication that takes place in FlexRay is based on periodic cycles (Fig. 1c depicts eight communication cycles). Each cycle contains two major time intervals with different bus access policies, i.e., the ST and DYN segments. The ST segment is used to transmit the periodic and safety-critical data, whereas the DYN segment is mainly for the maintenance and diagnosis data. The focus of this study is optimal scheduling of the ST segment. The ST segment is composed of several slots; each slot is filled with a frame carrying the data signals of the applications. All the ST slots are of identical, statically configured duration, and all frames are of identical, statically configured length based on the characteristics of a particular automotive application. We denote l_{cycle}, l_{ST}, and l_{slot} the durations of the cycle, ST segment, and static slot in milliseconds, respectively. n_{ST} denotes the number of slots in the ST segment. Each frame consists of header and trailer segments, and a payload segment that is statically configured to carry b_{slot} bytes. In the example in Fig. 1, we assume that the values of l_{cycle}, l_{ST}, and l_{slot} are 1, 0.55, and 0.0275 ms, respectively. The value of b_{slot} is 16 bytes. We need only to study the signal scheduling within H/l_{cycle} of the FlexRay communication cycle because it will repeat for all hyperperiods. In this example, the value of H/l_{cycle} is 8.

Within the ST segment, a TDMA scheme is applied to coordinate transmissions. A specific TDMA slot (communication slot) in the ST segment of a specific communication cycle is assigned to a unique ECU for transmission by assigning the corresponding slot number and communication cycle number to the ECU. Therefore, signal scheduling in the ST segment intends to assign an individual slot (slot number and cycle number) for every instance of every signal based on both its timing and FlexRay protocol-related constraints. For each ECU, the set of all the transmission times (all pairs of slot and cycle numbers) assigned to all instances of all signals is called the schedule table. For each of these slot identifiers used in the schedule table, the CHI reserves one or more buffers depending on whether the system uses the slot multiplexing mechanism. The schedule table has to support all combinations of sets of slots through which the transmit buffers can be configured.

Signals are selected and placed into the associated ST buffer in the CHI according to the schedule table. At the beginning of each TDMA slot of each cycle, the CC verifies if the valid payload presents the

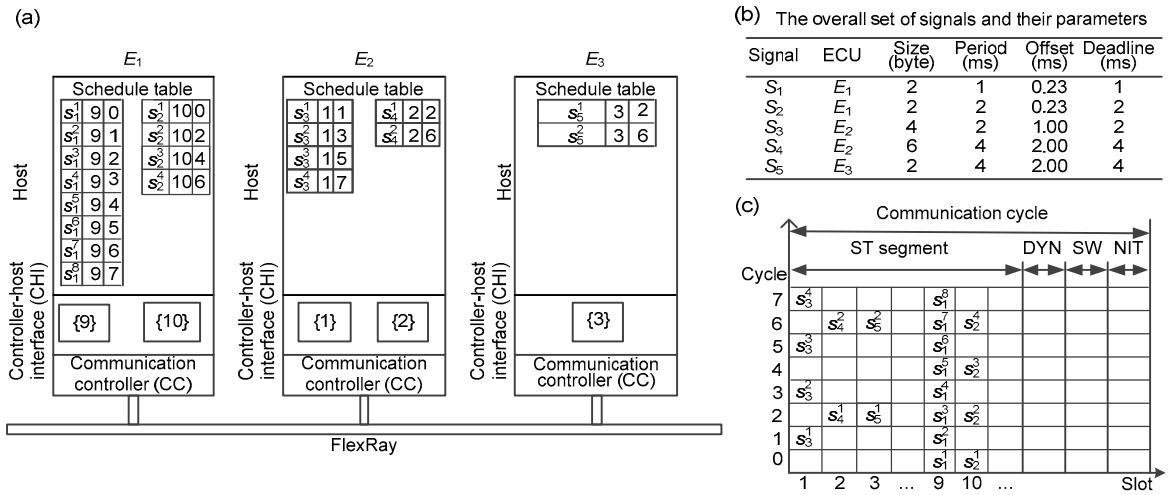

Fig. 1 Example of a FlexRay system model

(a) Schematic of ECUs connected by the FlexRay bus; (b) Overall set of signals; (c) Transmission process of each signal instance in the time-triggered system. ST: static; DYN: dynamic; SW: symbol window; NIT: network idle time

associated ST buffer, and transmits it in a FlexRay frame format to the bus.

Fig. 1a shows an example for scheduling the signals set in Fig. 1b in the FlexRay ST segment, without slot multiplexing. In this example, each ECU holds the schedule table with the transmission time (slot identifier and cycle number) of every instance of every signal. Consider the schedule table of ECU E_2, where the columns in the table show the signal instance, assigned slot identifier, and cycle number from left to right. For example, an entry $s_3^1|1|1$ specifies that the first instance s_3^1 of signal s_3 should be sent in slot 1 of cycle 1. Given a hyperperiod of 8 ms (eight FlexRay cycles), signal s_3 needs to be sent four times with a period of 2 ms, and s_4 needs to be sent twice with a period of 4 ms. As shown in the schedule table, the four instances s_3^1, s_3^2, s_3^3, and s_3^4 generated by signal s_3 are scheduled in slot 1 of cycles 1, 3, 5, and 7 in turn, and the instances s_4^1 and s_4^2 of another signal s_4 generated by ECU E_2 are scheduled in slot 2 of cycles 2 and 6. This slot assignment can meet the timing constraints of the signals. For example, slot 1 of cycle 1 assigned to the first instance s_3^1 of signal s_3 occurs within the time interval between its offset O_3 and its allowable latest time instant O_3+D_3 relative to the start of the first FlexRay cycle.

For each of the slot identifiers used in the schedule table, the CHI reserves a buffer that is configured for all slots that share the slot identifier in all the communication cycles. Fig. 1a depicts the associated buffers in CHI. Buffers {1} and {2} are configured for slots 1 and 2, respectively, in all cycles. Before the transmission time for a signal instance, the host needs to place it in its associated CHI buffer. At the right time, the CC will read this buffer and transmit the payload data within it to the FlexRay bus. For example, signal instance s_3^1 is placed into buffer 1 before the slot 1 of cycle 1 starts. When the slot comes, the CC reads buffer 1 and transmits s_3^1 in a FlexRay frame format to the bus. Each signal instance is explicitly sent according to the schedule table of the ECU to which it belongs; five slots are used in this slot assignment scheme. Fig. 1c presents the detailed transmission process of each signal instance in the time-triggered system within the eight FlexRay communication cycles, which corresponds to the schedule tables in Fig. 1a. In this example, the ST slots are not effectively used because the slot multiplexing mechanism is not used in scheduling signals. In this case, an ECU will transmit the same frame content in all slots with this number in all cycles. For example, ECU E_2 transmits signal s_3 in slot 1 in only some cycles. However, assigning slot 1 in the other idle cycles to other signals with the same ECU but different periods and offsets (i.e., s_4) or with different ECUs (i.e., s_5) is not allowed because such a situation will lead to different frame content.

2.3 Slot multiplexing mechanism

Although the slot assignment scheme in Fig. 1a can meet the timing constraints of each signal instance in the aforementioned time-triggered system, this case is not the optimal solution in terms of bandwidth utilization. With the significant increase in the number of signals in the FlexRay ST segment, an optimal signal scheduling that satisfies the timing constraints of each of the signals while optimizing the minimum number of slots used is essential for future automotive networks with large volumes of data.

To support the efficient use of the ST bandwidth, FlexRay provides a proprietary slot multiplexing mechanism that allows the alteration of frame content being sent to this slot from cycle to cycle. Specifically, FlexRay 3.0 allows the ECU to configure a transmit/receive buffer for a single slot or for a set of slots that share the same slot identifier in a configurable set of communication cycles. The configuration of the set of communication cycles can be defined using $\text{Cycle}_{\text{Repetition}}$ and $\text{Cycle}_{\text{Offset}}$. The transmit buffer is configured for this slot in each cycle with

$$\text{Cycle number}=(\text{Cycle}_{\text{Offset}}+n\cdot\text{Cycle}_{\text{Repetition}})\bmod 64,$$

with $\text{Cycle}_{\text{Offset}}$ selected from the set of {1, 2, 4, 5, 8, 10, 16, 20, 32, 40, 50, 64}, $\text{Cycle}_{\text{Offset}}$ selected from the set of {0, 1, …, 63} with $\text{Cycle}_{\text{Offset}}<\text{Cycle}_{\text{Repetition}}$, and variable n=0, 1, …, 63.

The configuration of the set of communication cycles is supported by the cycle counter filter criterion, which could cover all the cycles. A transmit buffer is identified by a tuple (slot identifier, $\text{Cycle}_{\text{Offset}}$, $\text{Cycle}_{\text{Repetition}}$). FlexRay supports the idea that multiple buffers are configured for the same slot identifier

with each corresponding to a frame content, enabling slot multiplexing. Slot multiplexing can be classified into single sender slot multiplexing and multiple sender slot multiplexing. This classification is based on whether the slots that have the same slot identifier but different communication cycle numbers can be assigned to different ECUs.

2.3.1 Single sender slot multiplexing mechanism

The single sender slot multiplexing in FlexRay allows an ECU to transmit different frame content in the same slot in different cycles. Fig. 2a shows an example of applying the single sender slot multiplexing mechanism to schedule the same signals set as in Fig. 1a. The schedule table of E_2 shows that signals s_3 and s_4 share the same slot number, where the four instances s_3^1, s_3^2, s_3^3, and s_3^4, generated by signal s_3, are still scheduled in slot 1 of cycles 1, 3, 5, and 7. In this case, the instances s_4^1 and s_4^2 of signal s_4 are scheduled in the same slot of cycles 1 and 5. Thus, two kinds of frame content will present in slot 1. One content, which consists of signals s_3 and s_4, is sent every four FlexRay cycles starting from slot 1 of cycle 1, and the associated buffer is identified by {1, 1, 4}. The other content, which consists of a single signal s_3, is sent every four FlexRay cycles starting from slot 1 of cycle 3, and the associated buffer is identified by {1, 3, 4}. At the beginning of slot 1 in cycle 1 or 5, the CC will be read by the buffer identified as {1, 1, 4}, and the frame content within it will be transmitted to the bus. At the beginning of slot 1 in cycle 3 or 7, the CC will be read by the buffer identified as {1, 3, 4}, and the frame content within it will be transmitted to the bus. Fig. 2b provides the detailed transmission process of each signal instance corresponding to the schedule tables in Fig. 2a. This example uses only three slot numbers by using the single sender slot multiplexing mechanism, thereby making effective use of available bandwidth.

2.3.2 Multiple sender multiplexing mechansim

The multiple sender slot multiplexing in FlexRay further allows several ECUs to transmit different frame content in the same slot in different cycles. Fig. 3a shows an example of applying the multiple sender slot multiplexing mechanism to schedule the same signal set as in Fig. 1a. In contrast to the slot assignment scheme in Fig. 2a, signal s_5, generated by ECU E_3, and signals s_3 and s_4, generated by E_2, share slot 1, where the two instances s_5^1 and s_5^2 of signal s_5 are scheduled in slot 1 of cycles 2 and 6. The slot assignments for the other signals remain the same. Therefore, a new frame content is available in slot 1, which consists of signal s_5. The frame is sent every four FlexRay cycles starting from slot 1 of cycle 2; the associated buffer is then identified by {1, 2, 4}. In this example, the number of slots used is further reduced to two through the use of the multiple sender slot multiplexing mechanism.

The aforementioned examples illustrate that the optimal signal scheduling in the FlexRay network can be reached only with the extensive and effective use of the slot multiplexing mechanisms, particularly the multiple sender slot multiplexing added in the latest version of the FlexRay specification. Given that most current chips support only the single sender slot multiplexing mechanism, this study separately adopts each mechanism to optimize signal scheduling in the FlexRay ST segment.

2.4 Problem statement

Our problem statement is formulated as follows: Given the system model described in Section 2, based on its signal set $S=\{s_1, s_2, \ldots, s_{n_s}\}$, we employ two types of mechanisms, namely, the single sender slot multiplexing and multiple sender slot multiplexing, to construct an optimal slot assignment solution, respectively, such that the timing constraints of all the signals in the system are satisfied, including both the lower and upper bounds, and that the total number of slots used is minimized.

3 Signal scheduling optimization method

The goal of signal scheduling in the FlexRay ST segment is to determine the pair of a slot number and a cycle number for every instance s_i^j of each signal s_i during one hyperperiod H. Therefore, a signal instance is the basic unit to which our optimization method allocates bandwidth source. To formulate the optimization problem, we define $I=\{s_1^1, s_1^2, ..., s_1^{H/T_1}, s_2^1, ..., s_{i-1}^{H/T_{i-1}}, s_i^1, ..., s_i^{H/T_i}\}$ as the set of all instances of all signals within a hyperperiod H. For simplicity, signal instances are identified and denoted by a single

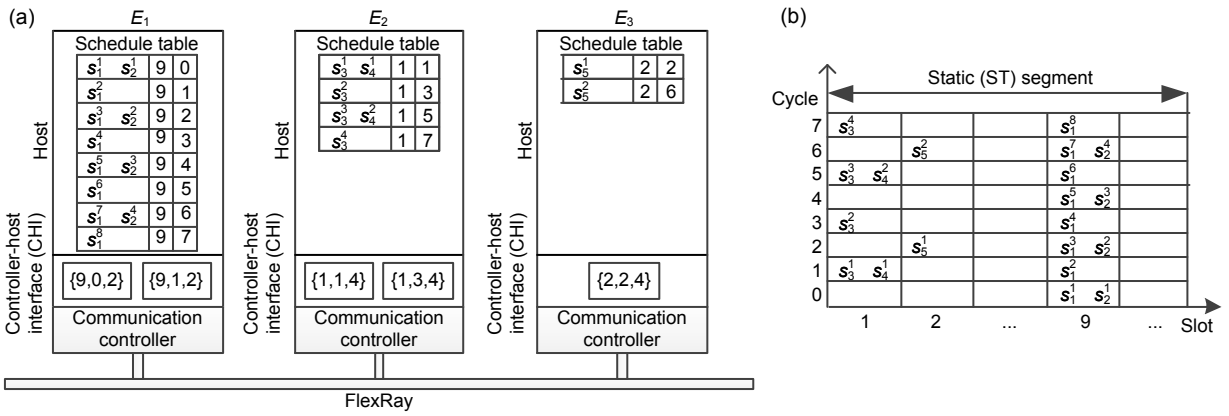

Fig. 2 Example of single sender slot multiplexing mechanism

(a) Schematic of ECUs connected by the FlexRay bus; (b) Transmission process of each signal instance in the time-triggered system

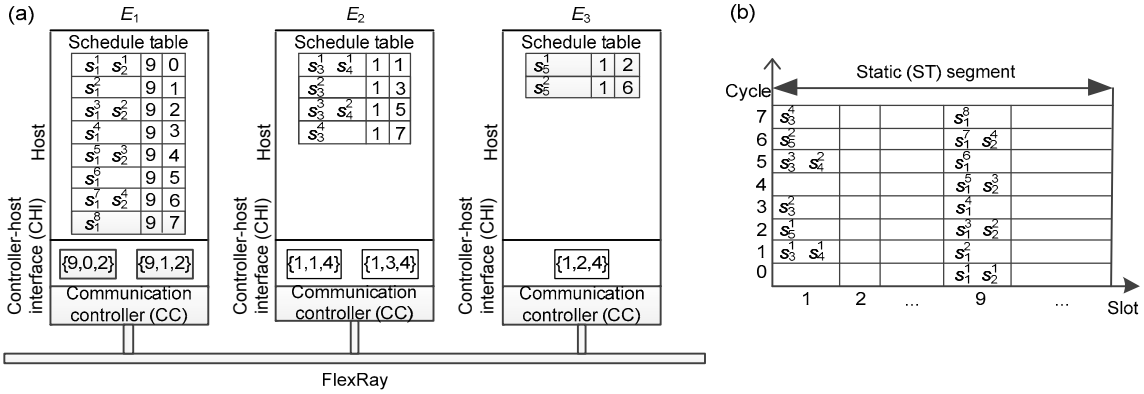

Fig. 3 Example of multiple sender slot multiplexing mechanism

(a) Schematic of ECUs connected by the FlexRay bus; (b) Transmission process of each signal instance in the time-triggered system

index τ_k. Each signal instance $\tau_k \in I$ is characterized by a tuple $(s_k, q_k, e_k, b_k, t_k, o_k, d_k)$, where s_k is the signal to which it belongs, q_k the sequence number of the instance τ_k within its own signal, e_k the identifier of the ECU that produces the signal to which it belongs, b_k the size of the signal to which it belongs, t_k the period of the signal to which it belongs, o_k the offset relative to the start of the first FlexRay communication cycle, which is calculated based on the offset O_{s_k} of the signal to which it belongs and its sequence number q_k as $O_{s_k}+(q_k-1)t_k$, and d_k the deadline of the signal to which it belongs.

In set I, the instances that belong to the same signal are placed together and sorted by the sequence of instances that occur, such that the first instance of each signal is ordered to the front. We assume that the total number of signal instances is n_k. Consequently, we assign an individual slot (the pair of slot number and cycle number) for each signal instance $\tau_k \in I$ with the objective of minimizing the number of slots used.

This slot assignment problem is similar to a two-dimensional rectangular bin packing problem, where, given a set of rectangular items having individual height h and width w, the objective is to pack them without overlapping into a minimum number of identical rectangular bins, having height H and width W (Lodi *et al.*, 2004; Puchinger and Raidl, 2007). In our problem, each signal instance $\tau_k \in I$ represents the rectangular item with a height of 1 and width of b_k, and each ST slot corresponds to one bin with the width of the payload size b_{slot} and height of the number of cycles H/l_{cycle} within one hyperperiod H. Our objective is to determine both the slot number (bin number) and cycle number (the position in the corresponding bin) for each signal instance. However,

slot assignment is more complex than general bin packing, because multiple constraints restrict the manner in which the bins can be covered, including both timing and FlexRay protocol-related constraints.

3.1 Integer linear programming formulation

In this section, we present the ILP approach to solve the particularly constrained two-dimensional bin packing problem that considers both single sender and multiple sender slot multiplexing mechanisms.

3.1.1 Variables

Our ILP formulation relies on the following binary variables:

$$\alpha_s=\begin{cases}1, & \text{slot } s \text{ is used},\\ 0, & \text{otherwise},\end{cases}$$

$$\beta_{k,s,l}=\begin{cases}1, & \text{instance } \tau_k \text{ is placed in slot } s \text{ at cycle } l,\\ 0, & \text{otherwise},\end{cases}$$

where s=1, 2, …, n_{slot}, l=1, 2, …, n_l, and $n_l=H/l_{\text{cycle}}$ denotes the height of the slot with unit of FlexRay cycle.

$$\gamma_{e_p,s,l}=\begin{cases}1, & \text{cycle } l \text{ of slot } s \text{ belongs to ECU } E_p,\\ 0, & \text{otherwise},\end{cases}$$

$$\delta_{e_p,s}=\begin{cases}1, & \text{slot } s \text{ belongs to ECU } E_p,\\ 0, & \text{otherwise},\end{cases}$$

where E_p=1, 2, …, n and s=1, 2, …, n_{slot}.

3.1.2 Optimization objective

The goal is to minimize the number of slots used in the FlexRay ST segment:

$$\min\sum_{s=1}^{n_{\text{slot}}}\alpha_s. \tag{2}$$

3.1.3 Constraints

The core of the slot multiplexing mechanism is its capability to allow altered frame content to be sent to a slot from cycle to cycle. Therefore, signals with different periods generated by the same ECU or several ECUs can be scheduled to the same slot depending on whether the slot multiplexing used is based on the single sender slot multiplexing mechanism or multiple sender slot multiplexing mechanism. The various instances of signals would constitute several kinds of frame content; the CHI supports slot multiplexing by providing configurable buffers with each corresponding to a kind of frame content. From a signal instance perspective, each instance $\tau_k\in I$ can be placed in any slot in any cycle, as long as both its timing and FlexRay protocol related constraints are satisfied. We present the constraints related to timing requirements (Eqs. (9)–(15)) and the FlexRay protocol (Eqs. (16)–(21)) in the following paragraphs:

1. Timing constraints

Section 2 discussed the requirement that a feasible slot assignment solution must ensure that the timing constraints of each signal be met. This process requires all the instances of a given signal be assigned to a slot after its offset and before its deadline. Consider the example in Fig. 1 where E_1–E_3 send signals through the FlexRay bus. The durations of FlexRay (l_{cycle}) and ST segment (l_{slot}) are 1 ms and 0.0275 ms, respectively, the number of slots (n_{slot}) in the ST segment is 20, and the hyperperiod is 8 ms, which includes eight FlexRay cycles. In this example, we further consider a signal s_2 of an offset O_2=0.23 ms, a deadline D_2=2 ms, and a period T_2= 2 ms. This signal generates four instances within one hyperperiod, which are denoted as s_2^1, s_2^2, s_2^3, and s_2^4, to describe their sending process more intuitively. The first instance s_2^1 is generated at O_2=0.23 ms and the subsequent instances s_2^1, s_2^3, and s_2^4 are generated at 2.23, 4.23, and 6.23 ms, in turn, in the interval of the period of the signal that generates it. The first instance s_2^1 must be sent before the time instant O_2+D_2=2.23 ms; the allowable latest time instants of s_2^2, s_2^3, and s_2^4 are 4.23, 6.23, and 8.23 ms, respectively (Fig. 4 depicts the first two instances of signal s_2).

Therefore, each instance can be assigned only to an ST slot in a range of the first available slot in the cycle in which it is generated and the last available slot in the cycle to which it must be sent. This range is called the ‘feasible region’ of this instance on the FlexRay bus, as shown in Fig. 4. We analyze the feasible region on the FlexRay bus for the first instance s_2^1. The cycle in which s_2^1 is generated is $\lfloor O_2/l_{\text{cycle}}\rfloor$=

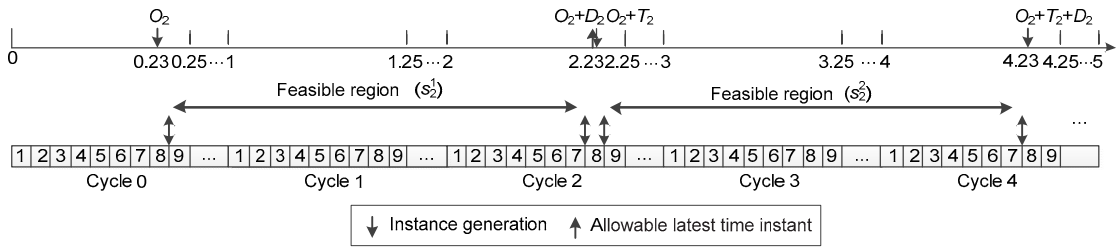

Fig. 4 Feasible region for the instances generated by a signal

$\lfloor 0.23 \text{ ms}/1 \text{ ms} \rfloor=0$, and the first available slot in this cycle is $\lfloor (O_2 \bmod l_{\text{cycle}})/l_{\text{slot}} \rfloor+1=\lfloor (0.23 \text{ ms} \bmod 1 \text{ ms})/0.0275 \text{ ms} \rfloor+1=9$. The cycle in which the deadline of s_2^1 is going to expire is $\lfloor (O_2+D_2)/l_{\text{cycle}} \rfloor=\lfloor 2.23 \text{ ms}/1 \text{ ms} \rfloor=2$, and the last available slot in this cycle is $\lfloor ((O_2+D_2) \bmod l_{\text{cycle}})/l_{\text{slot}} \rfloor-1=\lfloor (2.23 \text{ ms} \bmod 1 \text{ ms})/0.0275 \text{ ms} \rfloor-1=8$.

Similarly, we calculate the first and the last available cycles and slots for the other instances, namely, s_2^2, s_2^3, and s_2^4. Table 1 lists the results. We note that the interval between two neighboring feasible regions is an integral multiple of the FlexRay cycle duration (l_{cycle}) because l_{cycle} is the greatest common divisor of all signal periods or an integer divisor of that value. Therefore, as long as we can assign a slot within its feasible region for the first instance of a given signal, and then simply assign the same slot of the different FlexRay cycles for the other instances in the interval of the period of the signal that generates them, all timing requirements of all instances will be met and the different frame content that should be generated will be reduced. As a result, the buffer resources will be economized and the complexity of the system design will be decreased.

Table 1 Feasible regions for all the instances generated by signal s_2

Instance	First slot	First cycle	Last slot	Last cycle
s_2^1	9	0	7	2
s_2^2	9	2	7	4
s_2^3	9	4	7	6
s_2^4	9	6	7	8

The following presents how the feasible region is computed for the first instance of all signals in the general case. For any such instance, $\tau_k \in I$ with $q_k=1$:

(a) The cycle sc_k in which this instance is generated and the cycle fc_k in which its deadline is about to miss are computed as follows:

$$\begin{cases} \text{sc}_k = \lfloor o_k / l_{\text{cycle}} \rfloor, \\ \text{fc}_k = \lfloor (o_k + d_k)/l_{\text{cycle}} \rfloor, \end{cases} \tag{3}$$

(b) The first available slot ss_k in the cycle sc_k and the last available slot fs_k in the cycle fc_k are computed as follows:

$$\begin{cases} \text{ss}_k = \lfloor (o_k \bmod l_{\text{cycle}})/l_{\text{slot}} \rfloor + 1, \\ \text{fs}_k = \lfloor ((o_k + d_k) \bmod l_{\text{cycle}})/l_{\text{slot}} \rfloor. \end{cases} \tag{4}$$

Under normal conditions, we simply need to send this instance between the first available slot ss_k in the cycle sc_k and the last available slot fs_k in the cycle fc_k. However, two types of special cases must be considered. One of these cases is when the value of the first available slot ss_k is greater than the total number of slots n_{slot} in the ST segment, because a FlexRay cycle also includes other segments (e.g., DYN segment). Therefore, this instance can be sent only after the first slot in the next cycle. We comprehensively calculate the allowable first slot ss'_k and the cycle sc'_k as follows:

$$\text{ss}'_k = \begin{cases} \text{ss}_k, & \text{ss}_k \le n_{\text{slot}}, \\ 0, & \text{ss}_k > n_{\text{slot}}, \end{cases} \tag{5}$$

$$\text{sc}'_k = \begin{cases} \text{sc}_k, & \text{ss}_k \le n_{\text{slot}}, \\ \text{sc}_k + 1, & \text{ss}_k > n_{\text{slot}}. \end{cases} \tag{6}$$

The other case is when the value of the available last slot fs_k is greater than the total number of slots n_{slot} in the ST segment or is smaller than 1; thus, this instance can be sent only before the last slot in the previous cycle. We comprehensively calculate the

allowable last slot fs'_k and the cycle fc'_k as follows:

$$\mathrm{fs}'_k=\begin{cases}\min(\mathrm{fs}_k, n_{\mathrm{slot}}), & \mathrm{fs}_k \geq 1,\\ n_{\mathrm{slot}}, & \mathrm{fs}_k < 1,\end{cases} \tag{7}$$

$$\mathrm{fc}'_k=\begin{cases}\mathrm{fc}_k, & \mathrm{fs}_k \geq 1,\\ \mathrm{fc}_k-1, & \mathrm{fs}_k < 1.\end{cases} \tag{8}$$

Therefore, for any instance $\tau_k \in I$ with q_k=1, we need to assign a slot between the allowable first slot ss'_k in the start cycle sc'_k and the allowable last slot fs'_k in the finish cycle fc'_k. Three further scenarios are present in this feasible region:

(c) If the start cycle sc'_k is equal to the finish cycle fc'_k, then the instance $\tau_k \in I$ with q_k=1 is sent between the first slot ss'_k and the last slot fs'_k in this cycle:

$$\sum_{s=\mathrm{ss}'_k}^{\mathrm{fs}'_k} \beta_{k,s,\mathrm{sc}'_k}=1,\ \forall k \mid q_k=1 \vee \mathrm{sc}'_k=\mathrm{fc}'_k. \tag{9}$$

(d) If the start cycle sc'_k is equal to the finish cycle fc'_k+1, then the instance $\tau_k \in I$ with q_k=1 is sent between the first slot ss'_k and the slot n_{slot} in the start cycle sc'_k or between slot 1 and the last slot fs'_k of the finish cycle fc'_k:

$$\sum_{s=\mathrm{ss}'_k}^{n_{\mathrm{slot}}} \beta_{k,s,\mathrm{sc}'_k}+\sum_{s=1}^{\mathrm{fs}'_k} \beta_{k,s,\mathrm{fc}'_k}=1,$$
$$\forall k \mid q_k=1 \vee \mathrm{sc}'_k=\mathrm{fc}'_k+1. \tag{10}$$

(e) If the start cycle sc'_k is equal to or greater than the finish cycle sc'_k+2, then the instance $\tau_k \in I$ with q_k=1 is sent between the first slot ss'_k and the slot n_{slot} in the start cycle sc'_k or between slot 1 and slot n_{slot} in all cycles after sc'_k and before sc'_k, or between slot 1 and the last slot fs'_k of the finish cycle fc'_k:

$$\sum_{s=\mathrm{ss}'_k}^{n_{\mathrm{slot}}} \beta_{k,s,\mathrm{sc}'_k}+\sum_{s=1}^{n_{\mathrm{slot}}}\sum_{l=\mathrm{sc}'_k+1}^{\mathrm{fc}'_k-1} \beta_{k,s,l}+\sum_{s=1}^{\mathrm{fs}'_k} \beta_{k,s,\mathrm{fc}'_k}=1,$$
$$\forall k \mid q_k=1 \vee \mathrm{sc}'_k \geq \mathrm{fc}'_k+2. \tag{11}$$

As noted earlier, once the slot assigned to the first instance of a given signal is determined, the other instances just need to be placed in the same slot of the different FlexRay cycles in turn, in the interval of the period of the signal that generates them, to ensure that their timing requirements be met and to save the buffer resources. This constraint can be formulated as follows:

$$\beta_{k,s,l}=\beta_{k+n,(l+n\cdot p_k) \bmod H},$$
$$\forall k \mid q_k=1,\ n=1,2,...,(H/p_k)-1. \tag{12}$$

2. FlexRay protocol related constraints

The signal instance $\tau_k \in I$ can be placed only once into exactly one slot s at the specific cycle l. This constraint can be formulated as follows:

$$\sum_{s=1}^{n_{\mathrm{slot}}}\sum_{l=0}^{n_l-1} \beta_{k,s,l}=1,\quad \forall k=1,2,...,n_k. \tag{13}$$

The slot s has to be used if at least one instance τ_k is packed in it:

$$\alpha_s=\beta_{k,s,l},$$
$$\forall k=1,2,...,n_k,\ s=1,2,...,n_{\mathrm{slot}},\ l=1,2,...,n_l. \tag{14}$$

The length of all signal instances within each cycle of every slot must not exceed slot payload capacity b_{slot}:

$$\sum_{k=1}^{n_k} b_k \beta_{k,s,l} \leq b_{\mathrm{slot}},\ \forall s=1,2,...,n_{\mathrm{slot}},\ l=1,2,...,n_l. \tag{15}$$

The single sender slot multiplexing mechanism allows only the signals generated by the same ECU and shares one slot identifier through multiplexing cycles. Hence, the mechanism requires all slots sharing the slot identifier in all communication cycles to be owned by one specific ECU. Constraint (16) ensures that if an instance is placed in the slot in a specific cycle, then its source ECU must own the entire slot. Constraint (17) ensures that every slot identifier belongs to one ECU at the most. Constraint (18) limits the slot ownership to null if no instance is placed in this slot:

$$\beta_{k,s,l} \le \delta_{e_k,s},\ \forall s = 1,2,...,n_{\text{slot}}, \tag{16}$$

$$\sum_{e_p=1}^{n} \delta_{e_p,s} \le 1,\ \forall s = 1,2,...,n_{\text{slot}}, \tag{17}$$

$$\delta_{e_k,s} \le \sum_{k=1}^{n_k} \sum_{l=1}^{n_l-1} \beta_{k,s,l},\ \forall s = 1,2,...,n_{\text{slot}}. \tag{18}$$

The multiple sender slot multiplexing mechanism allows only the signals generated by several ECUs and shares one slot identifier through multiplexing cycles, and the mechanism requires only a specific slot of a cycle be owned by one specific ECU. Similarly, constraint (19) ensures that if an instance is placed at a specific slot in a specific cycle, then its source ECU must own this individual slot. Constraint (20) ensures that every slot of a cycle slot should belong to at most one ECU. Constraint (21) also limits the slot ownership to null if no instance is placed in this individual slot:

$$\beta_{k,s,l} \le \gamma_{e_k,s,l},\ \forall s = 1,2,...,n_s,\ l = 1,2,...,n_l, \tag{19}$$

$$\sum_{e_p=1}^{n} \gamma_{e_k,s,l} \le 1,\ \forall s = 1,2,...,n_s,\ l = 1,2,...n_l, \tag{20}$$

$$\gamma_{e_k,s,l} \le \sum_{k=1}^{n_k} \beta_{k,s,l},\ \forall s = 1,2,...,n_s,\ l = 1,2,...,n_l. \tag{21}$$

Constraints (9)–(15) together with constraints (16)–(18) define the ILP formulation based on the single sender slot multiplexing mechanism; constraints (9)–(15) together with constraints (19)–(21) define the ILP formulation for the multiple sender slot multiplexing. The ILP formulation combined with the minimization goal (Eq. (2)) determines the optimal slot assignment for each signal instance $\tau_k \in I$.

3.2 Efficient integer linear programming

In this section, we further increase the efficiency for the above ILP by reducing the search space, thereby reducing the runtime.

1. Lower bound for the number of slots needed: The minimum number of slots that are needed for sending all instances in set I is obtained as follows:

$$L_{\text{s}} = \left\lceil \frac{\sum_{\tau_k \in I} b_k}{b_{\text{slot}} \cdot n_l} \right\rceil. \tag{22}$$

2. Upper bound for the slot identifier: We bound the maximum slot identifier U_{s} needed for scheduling all signal instances in set I to reduce the runtime of our ILP. The value of U_{s} must not exceed the total amount of slots n_{slot} in the ST segment, and this value must be larger than both the maximum slot identifier $\max_{\tau_k \in I} \left\lceil (o_k \bmod l_{\text{cycle}}) / l_{\text{slot}} \right\rceil$, in which all instances are generated, and the minimum number of needed slots L_{s}. This upper bound is thus calculated as follows:

$$U_{\text{s}} = \min\left[n_{\text{slot}}, \max\left(L_{\text{s}}, \max_{\tau_k \in I} \left\lceil (o_k \bmod l_{\text{cycle}}) / l_{\text{slot}} \right\rceil \right) \right]. \tag{23}$$

Therefore, based on the two bounds, we replace n_{slot} with the upper bound U_{s} in all constraints in the aforementioned ILP and then add a new constraint:

$$\sum_{s=1}^{U_{\text{s}}} \alpha_s \ge L_{\text{s}}. \tag{24}$$

4 Experimental results and discussion

We conducted extensive experiments by running our proposed ILP approach on synthetic test cases and an automotive X-by-wire system case study. The ILP solver for the special two-dimensional bin packing problem was the CPLEX solver version 12.5. The experiments were conducted using a Windows 7 computer running on an Intel Core i5 2.80 GHz processor with 8 GB memory.

The synthetic case studies were based on the following FlexRay configuration, which is in accordance with the configuration in the latest BMW X5 SUV: the duration of the communication cycle (l_{cycle}) is 5 ms, with 3 ms for the ST segment and 2 ms for the DYN segment, SW, and NIT. The ST segment is composed of 91 slots, with each slot having a payload of 16 bytes. The duration of the slots is 0.032 ms. The test cases were generated by varying randomly the signal parameters, such as periods, deadlines, offsets, and lengths, to cover a wide range of possible combinations. The periods of the signals vary between $1\times l_{\text{cycle}}$ and $8\times l_{\text{cycle}}$ and both the deadlines and offsets of the signals vary between $1\times l_{\text{cycle}}$

and $8\times l_{\text{cycle}}$ and are smaller than the corresponding period. The lengths of the signals vary between 1 byte and 8 bytes. We considered a system composed of eight ECUs, where each ECU generated 5, 10, 15, 20, and 25 signals, respectively. We experimented with 20 examples for each case.

Fig. 5 shows the results of the scheduling using both the ILP approach based on the single sender slot multiplexing mechanism (ILP_SS) and the multiple sender slot multiplexing mechanism (ILP_MS). Fig. 5a shows the number of slots required by the ILP_MS, ILP_SS, and the non-optimization approach for the test cases. Fig. 5b shows the corresponding runtime for both the ILP_SS and ILP_MS. As observed in Fig. 5a, both ILP_MS and ILP_SS outperform the non-optimization approach in all test cases, and ILP_MS typically achieves the best results. As the number of signals increases, the slots saved by either ILP_MS or ILP_SS become even more significant. As shown in Fig. 5b, the runtime of both ILP_MS and ILP_SS depends on the characteristics of real-time applications, i.e., the number of signals. In our experiments, a timeout of 3600 s was used. For the cases with 40, 80, and 120 signals on the FlexRay bus (each ECU generating 5, 10, and 15 signals), the solver returns the optimal result in a significantly limited amount of time. For the case with 160 signals, the solver returns the optimal result in longer running times. For the case with 200 signals, the solver returns a result that uses one more slot than the lower bound.

To show the advantage of the proposed approach, we also considered an automotive X-by-wire system case study. We compared the performance of the ILP_MS approach with the existing methods described by Lukasiewycz *et al.* (2009) and Tanasa *et al.* (2011) in this real-life case. These three approaches represent three different ideas on optimal scheduling in the FlexRay ST segment.

The X-by-wire system consists of 11 ECUs and a total of 128 signals. Table 2 shows the signal characteristics, including the ECU that generated the signal, the offset, the period and deadline of the signal, and the size of the signal. The last column depicts the number of signals with those characteristics that were generated from the same ECU. The FlexRay bus configuration is as follows: the durations of the communication cycle, the ST segment, and the ST slot are 1 ms, 0.8 ms, and 0.032 ms, respectively. The

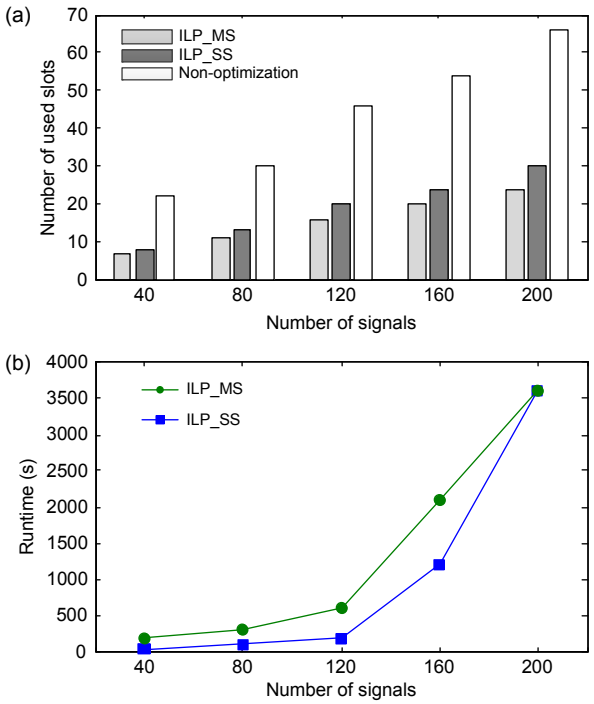

Fig. 5 Results of the synthetic case study

(a) The number of slots required by the ILP_MS, ILP_SS, and non-optimization approach for the test cases; (b) The corresponding runtime for both the ILP_SS and ILP_MS. ILP_SS: integer linear programming (ILP) approach based on the single sender slot multiplexing mechanism; ILP_MS: ILP approach based on the multiple sender slot multiplexing mechanism

ST segment is composed of 25 slots, with each slot having a payload of 16 bytes.

The results of scheduling using the three approaches are presented in Table 3. Tanasa *et al.* (2011) studied the reliability and schedulability problems of signal transmission in the FlexRay network with the objective of minimizing the bandwidth cost. Their approach consists of the following components: packing signals into frames, computing the number of times each frame must be retransmitted to guarantee the desired reliability level, and scheduling each frame to a separate slot in the ST segment such that the deadlines are met. In this case study, we focused on the parts of their approach that involved schedulability, without considering frame retransmissions. Using their approach, all signals are packed into 24 frames, and these obtained frames are assigned to 24 slots accordingly. Lukasiewycz *et al.* (2009) tried to improve bandwidth utilization by employing the single slot multiplexing mechanism on signal

Table 2 Signal list for an X-by-wire case study

ECU	Offset (μs)	Period (μs)	Deadline (μs)	Size (bit)	Multiplicity
ECU_1	7640	8000	8000	32	4
ECU_2	7650	8000	8000	32	4
ECU_3	105	1000	1000	32	5
ECU_3	530	1000	1000	32	5
ECU_3	530	1000	1000	16	4
ECU_3	530	1000	1000	8	2
ECU_4	120	1000	1000	32	1
ECU_4	565	1000	1000	32	1
ECU_4	565	1000	1000	16	4
ECU_4	565	1000	1000	8	2
ECU_5	160	1000	1000	32	5
ECU_5	160	1000	1000	16	2
ECU_6	200	1000	1000	32	6
ECU_6	450	1000	1000	16	2
ECU_6	450	1000	1000	8	2
ECU_7	1800	8000	8000	16	2
ECU_7	5800	8000	8000	16	10
ECU_8	1990	8000	8000	16	2
ECU_8	5990	8000	8000	16	10
ECU_9	2010	8000	8000	16	2
ECU_9	6010	8000	8000	16	10
ECU_{10}	2040	8000	8000	16	1
ECU_{10}	6040	8000	8000	16	10
ECU_{11}	4260	8000	8000	16	24
ECU_{11}	3490	8000	8000	16	7
ECU_{11}	3490	8000	8000	8	1

Table 3 Comparison of the optimal approaches proposed by Tanasa *et al.* (2011), Lukasiewycz *et al.* (2009), and this study

Approach	Number of frames	Number of slots used
ILP in Tanasa *et al.* (2011)	24	24
ILP in Lukasiewycz *et al.* (2009)	24	17
ILP_MS in this study	24	12

scheduling in the FlexRay ST segment based on traditional signals packing. It is important to note that the signal offsets in their approach are simply assumed to be zero. To compare the optimal performance levels, we added the constraints related to the offsets of the signals in their ILP approach. The results obtained by their approach show that the number of obtained frames is still 24, while the number of slots that have to be allocated for these frame transmissions is reduced from 24 to 17, because it allows an ECU to transmit different frame content in the same slot in different cycles. By comparison, our ILP_MS approach combines the more efficient multiple sender slot multiplexing mechanism (i.e., it allows several ECUs to transmit different frame content in the same slot in different cycles) and signals packing to optimize bandwidth utilization; thus, it produces the best results with the same number of frames, but with only 12 allocated slots.

5 Conclusions

We presented a rectangle bin packing optimization approach based on ILP that is capable of scheduling communication signals with timing constraints into the FlexRay ST segment at minimum bandwidth cost. The proposed approach supports both slot assignment mechanisms provided by the latest version of the FlexRay specification, namely, the single sender slot multiplexing and multiple sender slot multiplexing mechanisms, by using their respective special constraints and rules. We conducted extensive experiments on synthetic test cases. Our experimental results showed that the proposed approach, based on whether single sender slot multiplexing or multiple sender slot multiplexing mechanism, can achieve a well optimized performance. Additionally, we examined an automotive X-by-wire system case to emphasize the superior performance of the proposed approach compared to those of existing optimal scheduling approaches.

References

Bertoluzzo, M., Buja, G., Zuccollo, A., 2004. Design of drive-by-wire communication network for an industrial vehicle. IEEE Int. Conf. on Industrial Informatics, p.155-160. http://dx.doi.org/10.1109/INDIN.2004.1417320

Grenier, M., Havet, L., Navet, N., 2008. Configuring the communication on FlexRay: the case of the static segment. 4th European Congress on Embedded Real Time Software, p.1-18.

Hu, M.L., Luo, J., Wang, Y., *et al.*, 2014. Holistic scheduling of real-time applications in time-triggered in-vehicle networks. *IEEE Trans. Ind. Inf.*, **10**(3):1817-1828. http://dx.doi.org/10.1109/TII.2014.2327389

Hua, Y., Liu, X., He, W.B., *et al.*, 2014. Design and implementation of holistic scheduling and efficient storage for FlexRay. *IEEE Trans. Parall. Distrib. Syst.*, **25**(10): 2529-2539. http://dx.doi.org/10.1109/TPDS.2013.205

International Organization for Standardization, 2013. Road Vehicles—FlexRay Communications System—Part 2: Data Link Layer Specification, ISO 17458-2:2013. International Organization for Standardization, Geneva.

Kang, M., Park, K., Jeong, M.K., 2013. Frame packing for minimizing the bandwidth consumption of the FlexRay static segment. *IEEE Trans. Ind. Electron.*, **60**(9):4001-4008. http://dx.doi.org/10.1109/TIE.2012.2208433

Lee, K.C., Kim, M.H., Lee, S., *et al.*, 2003. IEEE-1451-based smart module for in-vehicle networking systems of intelligent vehicles. *IEEE Trans. Ind. Electron.*, **51**(6):1150-1158. http://dx.doi.org/10.1109/TIE.2004.837879

Lodi, A., Martello, S., Vigo, D., 2004. Models and bounds for two-dimensional level packing problems. *J. Combin. Optim.*, **8**(3):363-379. http://dx.doi.org/10.1023/B:JOCO.0000038915.62826.79

Lukasiewycz, M., Glaß, M., Teich, J., *et al.*, 2009. FlexRay schedule optimization of the static segment. Proc. 7th IEEE/ACM Int. Conf. on Hardware/Software Codesign and System Synthesis, p.363-372. http://dx.doi.org/10.1145/1629435.1629485

Navet, N., Song, Y., Simonot-Lion, F., *et al.*, 2005. Trends in automotive communication systems. *Proc. IEEE*, **93**(6):1204-1223. http://dx.doi.org/10.1109/JPROC.2005.849725

Park, I., Sunwoo, M., 2011. FlexRay network parameter optimization method for automotive applications. *IEEE Trans. Ind. Electron.*, **58**(4):1449-1459. http://dx.doi.org/10.1109/TIE.2010.2049713

Pop, T., Pop, P., Eles, P., *et al.*, 2008. Timing analysis of the FlexRay communication protocol. *Real-Time Syst.*, **39**(1-3):205-235. http://dx.doi.org/10.1007/s11241-007-9040-3

Puchinger, J., Raidl, G.R., 2007. Models and algorithms for three-stage two-dimensional bin packing. *Eur. J. Oper. Res.*, **127**(3):1304-1327. http://dx.doi.org/10.1016/j.ejor.2005.11.064

Robert Bosch GmbH, 1991. Controller Area Network. Available from http://www.can.bosch.com/ [Accessed on Mar. 22, 2015]

Schmidt, K., Schmidt, E.G., 2008. Message scheduling for the FlexRay protocol: the static segment. *IEEE Trans. Veh. Techn.*, **58**(5):2170-2179. http://dx.doi.org/10.1109/TVT.2008.2008654

Schneider, R., Goswami, D., Chakraborty, S., *et al.*, 2011. On the quantification of sustainability and extensibility of FlexRay schedules. Proc. 48th Design Automation Conf., p.375-380. http://dx.doi.org/10.1145/2024724.2024814

Tanasa, B., Bordoloi, U.D., Eles, P., *et al.*, 2011. Reliability-aware frame packing for the static segment of FlexRay. 9th ACM Int. Conf. on Embedded Software, p.175-184. http://dx.doi.org/10.1145/2038642.2038670

Tanasa, B., Bordoloi, U.D., Kosuch, S., *et al.*, 2012. Schedulability analysis for the dynamic segment of FlexRay: a generalization to slot multiplexing. IEEE 18th Real-Time and Embedded Technology and Applications Symp., p.185-194. http://dx.doi.org/10.1109/RTAS.2012.10

Zeng, H.B., Natale, M.D., Ghosal, A., *et al.*, 2011. Schedule optimization of time-triggered systems communicating over the FlexRay static segment. *IEEE Trans. Ind. Inform.*, **7**(1):1-17. http://dx.doi.org/10.1109/TII.2010.2089465