Multi-Traffic Resource Optimization for Real-Time Applications with 5G Configured Grant Scheduling

YUNGANG PAN, Department of Computer and Information Science, Linköping University, Linköping, Sweden
ROUHOLLAH MAHFOUZI, Department of Computer and Information Science, Linköping University, Linköping, Sweden
SOHEIL SAMII, Department of Computer and Information Science, Linköping University, Linköping, Sweden
PETRU ELES, Department of Computer and Information Science, Linköping University, Linköping, Sweden
ZEBO PENG, Department of Computer and Information Science, Linköping University, Linköping, Sweden

The fifth-generation (5G) technology standard in telecommunications is expected to support ultra-reliable low latency communication to enable real-time applications such as industrial automation and control. 5G configured grant (CG) scheduling features a pre-allocated periodicity-based scheduling approach, which reduces control signaling time and guarantees service quality. Although this enables 5G to support hard real-time periodic traffics, synthesizing the schedule efficiently and achieving high resource efficiency, while serving multiple communications, are still an open problem. In this work, we study the trade-off between scheduling flexibility and control overhead when performing CG scheduling. To address the CG scheduling problem, we first formulate it using satisfiability modulo theories (SMT) so that an SMT solver can be used to generate optimal solutions. To enhance scalability, we propose two heuristic approaches. The first one as the baseline, Co1, follows the basic idea of the 5G CG scheduling scheme that minimizes the control overhead. The second one, CoU, enables increased scheduling flexibility while considering the involved control overhead. The effectiveness and scalability of the proposed techniques and the superiority of CoU compared to Co1 have been evaluated using a large number of generated benchmarks as well as a realistic case study for industrial automation.

CCS Concepts: • **Computer systems organization → Embedded and cyber-physical systems**; **Embedded systems;**

This research has been supported by ELLIIT (Excellence Center at Linköping-Lund in Information Technology) and by SSF (Swedish Foundation for Strategic Research) under project no. FUS21-0033 Adaptive Software for the Heterogeneous Edge-Cloud Continuum.
Authors' Contact Information: Yungang Pan, Department of Computer and Information Science, Linköping University, Linköping, Östergötland, Sweden; e-mail: yungang.pan@liu.se; Rouhollah Mahfouzi, Department of Computer and Information Science, Linköping University, Linköping, Östergötland, Sweden; e-mail: rouhollah.mahfouzi@liu.se; Soheil Samii, Department of Computer and Information Science, Linköping University, Linköping, Östergötland, Sweden; e-mail: soheil.samii@liu.se; Petru Eles, Department of Computer and Information Science, Linköping University, Linköping, Östergötland, Sweden; e-mail: petru.eles@liu.se; Zebo Peng, Department of Computer and Information Science, Linköping University, Linköping, Östergötland, Sweden; e-mail: zebo.peng@liu.se.

ACM 1539-9087/2024/06-ART63
https://doi.org/10.1145/3664621

Additional Key Words and Phrases: 5G, URLLC, deterministic periodic traffic, configured grant scheduling, satisfiability modulo theories, resource optimization

ACM Reference Format:
Yungang Pan, Rouhollah Mahfouzi, Soheil Samii, Petru Eles, and Zebo Peng. 2024. Multi-Traffic Resource Optimization for Real-Time Applications with 5G Configured Grant Scheduling. *ACM Trans. Embedd. Comput. Syst.* 23, 4, Article 63 (June 2024), 31 pages. https://doi.org/10.1145/3664621

1 Introduction

The 5G standard for broadband cellular networks has been designed to support a wider range of applications compared to prior wireless communication techniques [1]. Apart from higher mobile bandwidth, **ultra-reliable low latency communication (URLLC)** is one of the main and vital features of 5G in supporting real-time applications and services required by, for example, industrial automation and manufacturing [2]. URLLC aims to support critical services with stringent latency and reliability requirements. In many industrial applications, deterministic periodic traffic is one of the most common traffic types [4], which generates data packets periodically with extremely strict timing constraints. Motion control, robotic communication, and factory automation are such examples.

Dynamic scheduling is typically used to coordinate the communication of data between the user and the **base station** (**BS**) via a hand-shaking process. The hand-shaking process creates a large amount of bi-directional message exchange, which exacerbates the latency problem, especially for industrial applications. In some situations, the communication of a data packet would take 10 ms, which is unacceptable for a large class of applications [21]. To support URLLC, the 5G standard [6] has introduced a new medium access mechanism, called **configured grant (CG)** scheduling, to mitigate the latency problem. CG scheduling assigns dedicated resources in a periodic manner to serve periodic traffic flows [1].

For one single traffic flow, a CG schedule consists of usually one single configuration, which is obtained in a straightforward manner. This configuration captures, among others, the transmission period, the frequency resources, and the time slots assigned to each data packet. For multiple traffic flows, the scheduling problem becomes difficult and complex, since resources have to be shared and used efficiently. At the same time, the stringent latency requirements of all the real-time traffic flows have to be satisfied. This article presents a CG scheduling technique that supports multiple traffic flows efficiently.

In order to provide flexibility in the search for an efficient solution, we would like to support multiple configurations in a proactive manner and not only a single one for each traffic flow. Although such a higher degree of scheduling flexibility helps to improve resource utilization, it also implies an increased amount of control signaling overhead. The trade-off between scheduling flexibility and control signaling overhead is thus an important factor that impacts CG scheduling. All the aforementioned considerations collectively motivate the core problem of this work: *how to solve the complex multi-traffic resource optimization problem with CG scheduling?*

In this article, we propose approaches to solve the CG scheduling problem. Detailed problem formulation, effective heuristic approaches, and extensive experiments are presented. In particular, we make the following contributions:

- We formulate and solve the CG scheduling problem with resource optimization in a systematic way considering the trade-off between scheduling flexibility and control overhead.
- We first establish a detailed specification of the CG scheduling problem based on **satisfiability modulo theories** (**SMT**) for a rigorous problem formulation.

Table 1. QoS Requirements of Selected 5G Industrial Applications [4, 16]

Applications	Transmission period	Payload	Latency requirement
Robotic motion control	2 ms	80 bytes	1 ms
Machine control	1 ~ 10 ms	40 ~ 250 bytes	< 50% transmission period
Assembly robots	4 ~ 8 ms	40 ~ 250 bytes	2 ~ 4 ms
Control-to-control communication	~ 4 ms	< 1 Kbytes	< 50% transmission period

— To enhance scalability, we propose the heuristic algorithm CoU to generate schedules efficiently. As a comparison, we develop another heuristic algorithm Co1 as the baseline.
— We implement and evaluate the proposed algorithms with a large number of synthetic benchmarks and a realistic case study. The results indicate that the proposed heuristic approaches provide good scalability and resource efficiency and demonstrate the superiority of the heuristic algorithm CoU compared to Co1.

The rest of this article is organized as follows: Sections 2 and 3 introduce the background and the related work. Section 4 presents the trade-off between scheduling flexibility and control overhead. Sections 5 and 6 describe the system model and the detailed problem specification. Section 7 proposes the heuristic algorithms. Section 8 presents the experimental settings and results. Finally, Section 9 concludes the article.

2 Background

2.1 5G Use Cases

5G URLLC is an important enabler for the incoming Industrial 4.0 era, which envisions a new production environment facilitated by a connected and digitalized process [24]. The community has identified a series of use cases and applications including, for example, factory automation, monitoring, and maintenance [4]. Among these use cases, deterministic periodic traffic is the most common communication type, especially for control-centric industrial applications. Deterministic periodic applications generate data packets periodically and each data packet has to be received within a given deadline. Some typical **quality-of-service (QoS)** requirements of deterministic periodic traffics are summarized in Table 1. For example, the robotic motion control application features a transmission period of 2 ms with a stringent latency requirement of 1 ms. The typical payload size is 80 bytes. Usually, there exist various use cases in the production environment, as illustrated in Figure 1. In a 5G-enabled factory automation framework, multiple periodic traffic flows with various latency requirements co-exist [4], which challenges the scheduling capability over the wireless communication infrastructure. To support such complex traffic flows, one critical challenge is how to provide an efficient way for multiple traffic flow scheduling considering the complicated and stringent requirements [27].

2.2 Configured Grant Scheduling

Very often, dynamic scheduling is widely adopted in 5G for data transmission as shown in Figure 2(a). In dynamic scheduling for uplink traffic, each actual data transmission is preceded by (1) a scheduling request indicating the transmission requirement of the packet, (2) the scheduling process performed by the BS, and (3) the corresponding uplink scheduling grant representing the time-frequency resources for data transmission. The aforementioned process repeats for each uplink transmission, which involves extremely large control signaling overhead. One uplink data transmission requires two control signaling messages. Although dynamic scheduling provides good scheduling flexibility, the involved control messages may cause control channel congestion, thus degrading the system performance, especially when transmitting small packets [14, 20].

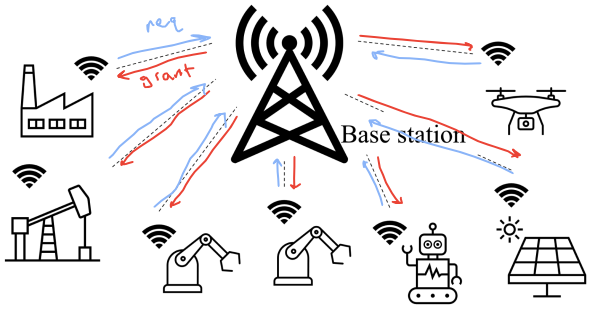


Fig. 1. The 5G system supports industrial automation applications.

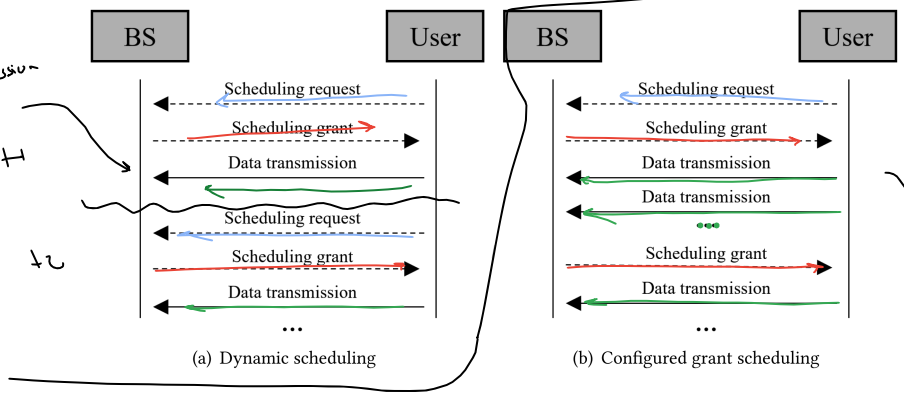


Fig. 2. Simplified signaling diagram of (a) dynamic scheduling and (b) CG scheduling for uplink transmissions.

Hence, the reduction of control signaling overhead must be considered in order to improve resource utilization. CG scheduling has first been introduced as part of **semi-persistent scheduling (SPS)** in the 4G standard to facilitate the scheduling of periodic **voice over Internet protocol (VoIP)** packets [17]. SPS assigns periodically fixed time-frequency resources to the user. Taking the VoIP application as an example, the resources for voice packets of 160 bytes are allocated every 20 ms. The control messages are only employed when there is a need to change the transport format and/or the time-frequency resources. Hence, tremendous control overhead can be mitigated compared to the typical dynamic scheduling scheme, especially for periodic traffic flows.

Similarly, to facilitate the scheduling of periodic traffic flows, 5G introduces the CG scheduling scheme to assign dedicated resources for data transmission based on the requirements of a particular traffic [5]. It is this kind of periodic real-time traffic that benefits from the drastically reduced control overhead implied by CG scheduling as compared to dynamic scheduling. The configured grant scheduling and SPS share the same basic idea. Compared to the conventional dynamic scheduling scheme, tremendous control overhead (a large number of scheduling requests/scheduling grant messages) can be avoided to improve resource efficiency as shown in Figure 2(b). The working process of CG scheduling is as follows: (1) the user sends a scheduling request, which indicates its transmission period, payload, and latency requirement among others, to the BS; (2) after performing the scheduling process by the BS, a scheduling grant (called *configuration* in this article) will be sent back to the user; the configuration provides the scheduling information needed for the following data transmissions, such as the **modulation and coding scheme (MCS)**, the time-frequency resources, and the period between every two adjacent data transmissions [6]; and

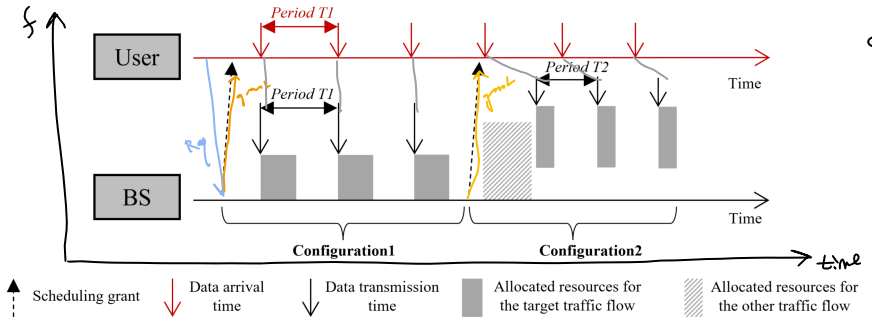


Fig. 3. Illustration of the CG scheduling scheme.

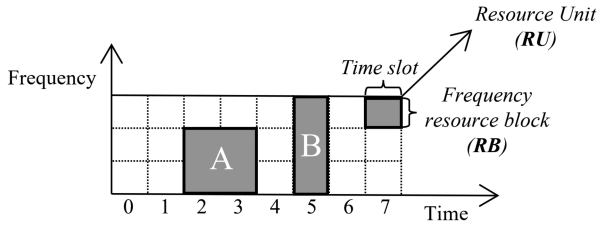


Fig. 4. Illustration of the wireless resource grid model.

(3) multiple data packets are transmitted to the BS according to the current configuration until a new configuration arrives. A new configuration is needed when the user changes its data transmission requirements or the BS updates its scheduling decision proactively.

A detailed illustration of the CG scheduling scheme is given in Figure 3. The target traffic flow generates packets periodically with period T1. Two configurations are employed for the target traffic flow, each of them indicating a certain data transmission scheme. As aforementioned, configuration represents the corresponding allocated time-frequency resources (gray rectangles in the figure) and the periods between two adjacent data transmissions that is, period T1 and period T2 in the figure. Specifically, configuration 1 is valid for the first three data transmissions, in which the period of data transmissions (i.e., period T1) is equal to the period of data arrivals (i.e., period T1). After that, we assume that an additional traffic flow request has to be accommodated, and a new configuration is triggered for our target traffic flow. Configuration 2 is now employed, which is enabled for the last three data transmissions with a different time-frequency resource allocation scheme and a shorter period (i.e., period T2) compared to configuration 1.

In terms of resource allocation, the wireless resource grid model used in this article is illustrated in Figure 4. The resource grid is separated into multiple time slots and multiple frequency **resource blocks** (**RBs**) evenly. One basic unit of the resource grid, called **resource unit** (**RU**) in this article, lasts for one time slot and spans one frequency RB. In Figure 4, there are two different resource allocation patterns, "A" and "B". Pattern "A" is formed by two time slots and two RBs and it comprises four RUs, while pattern "B" lasts for a single time slot and spans three RBs, meaning that three RUs are allocated. For configuration 1 in Figure 3, the resource allocation scheme follows pattern "A", while for configuration 2, pattern "B" is employed.

Note that in 5G protocols, the configuration message is conveyed by an information element called "ConfiguredGrantConfig [6]". In "ConfiguredGrantConfig", the following fields are defined: (1) resourceAllocation (i.e., frequency resource allocation information); (2) periodicity; (3) timeDomainOffset; and (4) cg-nrofSlots, which indicates the number of allocated slots for each data

transmission. Regarding "resourceAllocation", we adopt "resource allocation type 1" when allocating the frequency resources, which requires that the allocation of RBs must be consecutive. When allocating the time resources, "timeDomainOffset" and "cg-nrofSlots" require that the allocated time slots must be consecutive as well. Hence, the allocated time-frequency resources in Figure 3 and Figure 4 are in the form of rectangles.

3 Related Work

Resource allocation is the process by which wireless resources are allocated to the user for data transmission and control signaling. The scheduling techniques are extensively studied in order to provide efficient wireless resource management [10, 25]. Usually, statistical metrics are regarded as the scheduling optimization goal, such as the total throughput, the transmission delay, the average spectral efficiency, and the average queue length. Typical scheduling algorithms include *first-come-first-served*, *round-robin*, *maximum throughput*, and *proportional fair*. In this article, we are concerned about the schedulability of each traffic, which is crucial for URLLC applications, especially industrial real-time applications. For example, maximizing the overall system throughput does not mean that every traffic can be scheduled under given end-to-end delay constraints and every data packet can get the chance to be transmitted such that the deadlines are satisfied.

The pioneering research concerning CG scheduling has been proposed in the context of SPS for the 4G standard can be originated to [11–13, 15, 17]. Karadag et al. introduce SPS in the context of the scheduling of periodic machine-to-machine traffic with the optimization goal of minimizing the frequency resource consumption [19]. However, the impact of control messaging on schedulability and delays is not considered. In order to support various legacy and emerging requirements of periodic traffics, a recursive periodicity shifting SPS scheme is proposed to mitigate the misalignment issue between the periodic resource assignments and the periodic arriving packets in [18]. It concerns the scheduling time instances for periodic traffic under the SPS framework. However, its techniques can only be applied to serve one traffic. CG scheduling algorithms for multiple periodic traffics have been discussed recently, such as in [8, 9]. However, the developed approaches only consider the scenario with a single configuration, while ignoring the trade-off between scheduling flexibility and control overhead [8, 9].

In conclusion, neither the impact of control messages on the schedulability of multi-traffic communication nor the trade-off between control overhead and the potential flexibility provided by CG scheduling in the context of resource optimization has been considered in previous work except for our very first proposal in [7], which sketches the basic idea of considering schedulability and control overhead together.

4 Motivation

In this section, an illustration of the trade-off between the scheduling flexibility of data transmissions[1] and the involved control message overhead is given. Generally speaking, when there is only one **traffic flow** (**TF**), the schedule only needs one unique configuration. However, things become complex for multiple TFs. Figure 5 represents three alternatives for the transmission of two periodic TFs. In order to explore the trade-off between scheduling flexibility and control overhead, only one configuration is allowed for the first two schedules as shown in Figure 5(a) and (b), while the schedule in Figure 5(c) consists of two configurations. Since each new configuration has to be introduced by an additional control message, the number of configurations in a schedule is crucial when exploring the abovementioned trade-off.

[1]In this article, we prefer to use "data transmission" to represent the specific schedule information while the "data packet" refers to the actual packet/payload to be transmitted.

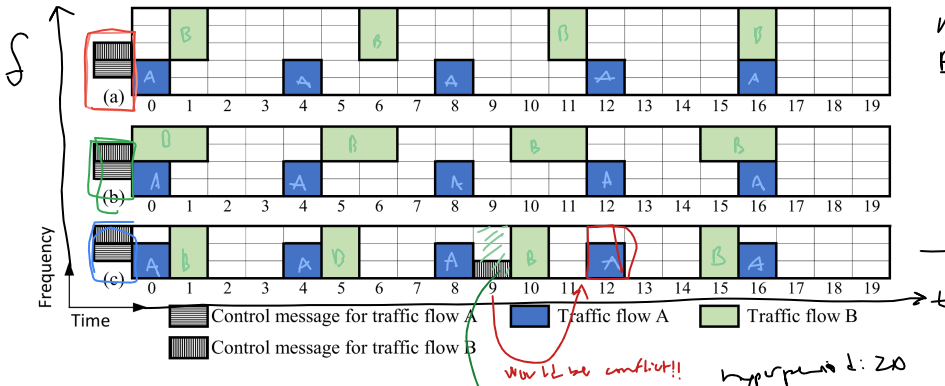


Fig. 5. Illustration of the trade-off between scheduling flexibility and control overhead for two periodic traffic flows.

In Figure 5, the horizontal axis represents the time slots and the vertical axis represents the frequency RBs. In this resource gird, there are 20 time slots, which is the hyperperiod of the two periodic TFs. The parameters of TF A are transmission period = 4 time slots, payload = 2 RUs, and latency requirement = 1 time slot, while for TF B, the parameters are transmission period = 5 time slots, payload = 3 RUs, and latency requirement = 2 time slots. The data transmissions of TF A and TF B are depicted with blue and green colors, respectively. In Figure 5(a), one configuration is used for TF A and one configuration for TF B. Five RBs are needed in total, as shown by the five rows. Taking the second data transmission of TF B as an example, the data transmission is transmitted in time slot 6 over 3 RBs, which means that it consumes 1×3 RUs. A schedule using fewer RBs is illustrated in Figure 5(b). As opposed to Figure 5(a), the schedule only needs 4 RBs. In this case, the second data packet of TF B is transmitted in the time slots 5 and 6 over 2 RBs, and it consumes 2×2 RUs (one of the RUs will not be used and will be wasted because the payload size is 3 RUs).

Now, if we want to further reduce the number of needed RBs, we need schedules that consist of two configurations for TF B as shown in Figure 5(c). Since two configurations are utilized, we need an additional control message for the newly involved second configuration compared to the cases with only a single configuration in Figure 5(a) and (b). In Figure 5(c), the first configuration for TF B represents the schedule for the first two data transmissions with the period of 4 time slots. Then we need the second configuration that is defined by a new control message in time slot 9. The third and fourth data transmissions of TF B are transmitted with the period of 5 time slots. The abovementioned two configurations have different offsets and periods such that all data transmissions can be successful. For the first configuration, the offset is slot 1 and the period is 4 time slots, while for the second configuration, the offset is slot 10 and the period is 5 time slots. Let us assume for a moment that the third data transmission is performed according to the scheduling decision in the first configuration. In this case (due to period = 4 time slots), the allocated RUs will be in slot 9. This will fail the third data transmission, since in slot 9, the third data packet of TF B has not arrived (it arrives in slot 10). Hence, a second configuration is needed.

In the above example, we illustrate the trade-off between scheduling flexibility and control overhead. Generally speaking, fewer configurations can lead to reduced resource efficiency as shown in Figure 5(a) and (b). This is because reduced scheduling flexibility means the lack of capability to handle the complex resource allocation interleaving effects among multiple TFs. On the other side, more configurations introduce additional control messages. The involved control messages consume resources and may affect resource efficiency if not considered carefully. Furthermore, the stringent latency requirement makes the problem even more complicated since

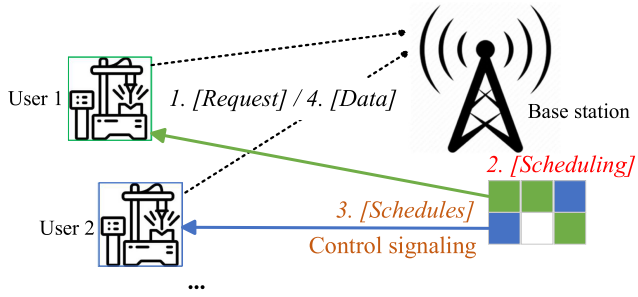


Fig. 6. The system overview.

schedules have to consider the deadlines for all data packets. Hence, the scheduling problem of multiple periodic TFs is very complex. Given the heterogeneous requirements of TFs including the transmission periods, the payloads, and the deadline constraints, there are three concerns when performing the scheduling as follows: (1) the period of the schedule may not equal the transmission period of the TF (e.g., the first configuration of TF B is 4 instead of the transmission period 5 as shown in Figure 5(c)); (2) various resource allocations can be adopted (e.g., two time slots × two RBs vs. one time slot × three RBs of TF B in Figure 5(b) and (c)); and (3) deadlines for all data packets have to be considered to ensure the per-traffic scheduling guarantee while taking into account the scheduling of control messages.

5 System Model and Problem Statement

5.1 System Overview

As illustrated in Figure 6, we consider systems consisting of one BS and multiple users. The typical working process of the system is as follows: first, the users' requirements are collected by the BS. The requirements from the user include all information needed for the scheduling such as the transmission period, the payload, and the latency requirement. As the result of the scheduling, which is done by the BS, one or several successive configurations are generated for each user. Their first configuration is transferred to each user via control signaling. Then, the data transmission starts. The periodically generated data packets are transmitted to the BS according to the configurations generated by the BS. All notations used in this article are summerized in Table 2 and for abbreviations, please see Appendix A.

5.2 Traffic Model

We assume multiple periodic traffic flows (TFs)[2] exist in the system. The set of all TFs is denoted by $\mathcal{F}$. Each TF $f_i \in \mathcal{F}$ is characterized by an initial offset ϕ_i,[3] a transmission period T_i, a payload B_i, and a maximum latency requirement D_i with $\phi_i + D_i \leq T_i$. In terms of transmission latency, we consider both the queuing time and transmission time. Here, the queuing time represents the time interval between the data arrival and the start time of the actual data transmission. The latency requirement D_i represents the maximum allowed sum of the queuing time and transmission time. The packets of TF f_i are generated at time $\phi_i + j \times T_i$ ($j \in \mathbb{N}_0$) and should be transmitted to BS by the corresponding deadline $\phi_i + j \times T_i + D_i$. For simplicity, in the next, all time-related parameters are normalized to the time duration of one single time slot. The indices of time slots start from

[2]Unless explicitly denoted, in this article, we assume one user corresponds to one TF, without loss of generality.
[3]We do not assume that all TFs are synchronized. The initial offset of each TF can be different, which may be due to the application's specific requirements.

Table 2. Notations

Notations	Descriptions
$\mathcal{F}$	The set of all traffic flows
f_i	Traffic flow i
ϕ_i	The initial offset of f_i
T_i	The transmission period of f_i
B_i	The payload of f_i
D_i	The maximum latency requirement of f_i
HP	The hyperperiod of $\mathcal{F}$
$\mathcal{G}_i$	The schedule of f_i
g_{ij}	j_{th} configuration of f_i
c_{ij}	The corresponding control message schedule of configuration g_{ij}
J_i	The number of data packets of f_i for the given HP
$\alpha_{g_{ij}}$	The RB allocation pattern of g_{ij}
$x_{g_{ij}}$	The index of the first allocated time slot of g_{ij}
$l_{g_{ij}}$	The number of consecutive time slots assigned for a single data transmission of g_{ij}
$p_{g_{ij}}$	The period between two consecutive data transmissions of g_{ij}
$r_{g_{ij}}$	The number of data transmissions involved in g_{ij}
$v_{g_{ij}}$	The valid indicator of g_{ij}
$\alpha_{c_{ij}}$	The RB allocation pattern of c_{ij}
$x_{c_{ij}}$	The index of the first allocated time slot of c_{ij}
$v_{c_{ij}}$	The valid indicator of c_{ij}
RU_i^d	The number of needed RUs for one single data transmission of f_i
RU_i^c	The number of needed RUs for one single control message of f_i
$T_{\mathcal{G}_i}$	The assigned time windows of $\mathcal{G}_i$
$T_{g_{ij}}$	The assigned time windows of g_{ij}
$\mathcal{A}_i$	The arrival time points for all data packets of f_i
$\mathcal{D}_i$	The deadlines for all data packets of f_i
$\mathbb{N}_0$	The natural numbers
$\mathbb{N}^+$	The positive natural numbers
β_{ixy}	The resource allocation indicator for the current resource grid
RB_{max}	The maximum NRBs
x	Offset for the first data transmission (Algorithm Co1)
w	The number of time slots allocated for each data transmission (Algorithm Co1)
h	The number of frequency RBs allocated for each data transmission (Algorithm Co1)
p	The time interval between every two adjacent data transmissions (Algorithm Co1)
b	The start index of the frequency RB (Algorithm Co1)

0. Take the second data transmission of TF A in Figure 5(a) as an example. The packet arrival time is 4, which means that the earliest transmission can start at the beginning of time slot 4. The corresponding deadline is five, requiring that the transmission must be finished before time slot 5. To sum up, only resources in time slot 4 can be allocated in this case. We generate schedules for one hyperperiod HP, which is the least common multiple of all TF's transmission periods. The schedules repeat periodically each hyperperiod.

5.3 Network Model

The wireless resources are represented by a two-dimensional resource grid as shown in Figure 4. Since we only need to perform the scheduling within a hyperperiod, we denote the number of

Table 3. A Lookup-table Example for MCS Selection [26]

SNR (dB)	MCS index	Effective transmitted bits
≥ -0.4167	0	16
≥ 1.0417	1	24
≥ 1.6667	2	32
≥ 2.9167	3	40
≥ 3.5147	4	56
≥ 5.0000	5	72
...	...	...

time slots of the resource grid as *HP*. The channel quality experienced by each user is given as the **signal-to-noise ratio** (**SNR**) impacting that user. We assume that the channel quality of each user is sufficiently stable within each hyperperiod. The number of needed RUs for each data transmission depends on three parameters: the channel quality, the block error rate required by the application (e.g., 10^{-5} for URLLC traffic), and the payload. In order to achieve the required block error rate, an appropriate MCS is needed. Different MCS selections correspond to different error correction capabilities. An MCS with higher error correction capability can be utilized in a low SNR scenario to achieve a certain block error rate. This higher error correction capability assumes a certain amount of overhead in terms of transmitted bits. This means that the effective (useful) bits transmitted per RU decrease with a lower SNR. Thus, for a given payload, a lower SNR results in the need of more RUs to be allocated.

In this work, we select the appropriate MCS depending on the given SNR and the required block error rate. The selection of the MCS can be performed using the lookup table specified in [3, 26]. A lookup-table example is given in Table 3, corresponding to a block error rate of 10^{-5}. For different block error rates, different tables are provided. Hence, the number of RUs required for each data transmission can be obtained by utilizing the payload divided by the effective transmitted bits for a single RU. This guarantees the required error rate and, thus, reliability. For example, given a packet with a payload of 40 bytes and an SNR of 2 dB, we first select the MCS index of 2 since it can at least provide the required error rate while not wasting too many resources compared with the MCS index of 0 and 1. The current MCS index means that one RU would transmit 32 bits. Hence, 10 RUs (i.e., $\lceil 40 \times 8 \div 32 \rceil$) are needed. Regarding the control message, since it is usually small, we assume that one RU is needed for the delivery of one single control message.

5.4 Problem Formulation

The aim of the scheduling problem in this study is to allocate enough resources within the suitable time windows so as to fulfill the TFs' requirements while minimizing the total number of needed RBs. The inputs include the initial offsets, the transmission periods, the payloads, the latency requirements, the required block error rate (by default, 10^{-5}), and the experienced SNR of each TF for all TFs, while the outputs are the configurations for data transmissions and the corresponding control messages for all TFs in a hyperperiod.

6 SMT-Based Solution

In this section, we propose an SMT-based[4] technique to establish an exact solution of the scheduling problem. First, the concept of schedule/configurations is defined. Then, all related constraints

[4]In general, SMT can determine whether a mathematical formula is satisfiable, which can be thought of as a formalized approach to constraint programming [23].

are established such as the number of RUs required, and the deadlines for all data transmissions among others. Finally, the optimization goal is stated. An SMT solver can be applied to solve the defined constraints-based problems and derive the optimal solution [23].

6.1 Schedule/Configurations

A schedule for a TF f_i is a sequence $\mathcal{G}_i = \{\{c_{i1}, g_{i1}\}, \ldots, \{c_{ij}, g_{ij}\}, \ldots, \{c_{iJ_i}, g_{iJ_i}\}\}$, in which g_{ij} represents one configuration and c_{ij} means the corresponding control message schedule of configuration g_{ij}. For the schedule of TF f_i, there are at most $J_i = HP/T_i$ configurations and control messages. Each configuration g_{ij} is characterized by a six-tuple $\langle \alpha_{g_{ij}}, x_{g_{ij}}, l_{g_{ij}}, p_{g_{ij}}, r_{g_{ij}}, v_{g_{ij}} \rangle$, where $\alpha_{g_{ij}}$ denotes the RB allocation pattern (a vector with each element showing the allocation status of each RB), $x_{g_{ij}}$ is the index of the first allocated time slot, $l_{g_{ij}}$ is the number of consecutive time slots assigned for one data transmission, $p_{g_{ij}}$ is the period between two consecutive data transmissions, $r_{g_{ij}}$ represents the number of data transmissions (i.e., the number of packets covered by the current configuration), and $v_{g_{ij}}$ is the valid indicator. Since the number of needed configurations is unknown until the optimized scheduling decision has been made, the valid indicator $v_{g_{ij}}$ is introduced to represent the valid status of each candidate configuration. The valid indicator can be set to 1 by the SMT solver, which indicates that the corresponding configuration is enabled and will take effect. Otherwise, the configuration is of no use to the current schedule. Similarly, the schedule of the control messages c_{ij} is represented by a three-tuple $\langle \alpha_{c_{ij}}, x_{c_{ij}}, v_{c_{ij}} \rangle$ consisting of the RB allocation pattern $\alpha_{c_{ij}}$, the index of the first allocated time slot $x_{c_{ij}}$, and the valid indicator $v_{c_{ij}}$.

Taking the schedule of TF B in Figure 5(c) as an example, the second configuration can be represented by $\langle [1, 1, 1], 10, 1, 5, 2, 1 \rangle$, which is denoted as "*configurationTFB*". The semantics of the parameters of the configuration is as follows: (1) "α = [1, 1, 1]": the RBs 0, 1, and 2 of the resource grid are to be allocated; (2) "x = 10": the index of the first allocated time slot for the first data transmission indicated by the current configuration is 10; (3) "l = 1": the number of allocated time slot is 1; (4) "p = 5": the period between two consecutive data transmissions is 5, hence the second data transmission in this configuration is in time slot 15; (5) "r = 2": in this configuration, 2 data transmissions are involved; and (6) "v = 1": the current configuration is valid. To sum up, the configuration implies that the third data packet of TF B is transmitted in time slot 10 using three RBs while the fourth data packet is transmitted in time slot 15 spanning also three RBs.

As aforementioned, a schedule consists of a sequence of configurations. Each configuration has to be introduced by a control message. These control messages themselves have to be scheduled, except the first one, which is assumed to be sent to the user before the first hyperperiod starts, and stored as the default configuration that is used every time a new hyperperiod starts. Taking the control messages of TF B in Figure 5(c) as an example, the first one is issued before time slot 0. The resource allocation of the second control message corresponding to *configurationTFB* can be represented by $\langle [1, 0, 0], 9, 1 \rangle$ meaning that the RU for control message in time slot 9 and RB 0 would be allocated.

6.2 Constraints

The constraints for the scheduling problem are defined as follows:

6.2.1 Number of Resource Units Required. The allocated RUs have to support the transmission of: (1) data transmission for the payload; and (2) control message. Considering a TF f_i and a configuration g_{ij}, the number of RUs for transmission is the product between the number of assigned time slots ($l_{g_{ij}}$) and the NRBs allocated ($\sum \alpha_{g_{ij}}$). The same is true for the corresponding control message c_{ij}, where the number of the assigned time slots is 1 and the NRBs is $\sum \alpha_{c_{ij}}$. The numbers of

needed RUs for data transmissions and control messages are denoted by RU^d and RU^c, respectively (for the related calculation, see Section 5.3). Thus, we can formulate the following two constraints:

$$\sum \alpha_{g_{ij}} \times l_{g_{ij}} \geq RU_i^d, \forall g_{ij} \in \mathcal{G}_i, \forall f_i \in \mathcal{F},$$
$$\sum \alpha_{c_{ij}} \geq RU_i^c, \forall c_{ij} \in \mathcal{G}_i, \forall f_i \in \mathcal{F}.$$

In *configurationTFB*, α is [1, 1, 1] and l is 1. Therefore, $\sum \alpha$ is 1+1+1 = 3 and then we have $\sum \alpha \times l \geq RU^d = 3$.

6.2.2 Deadlines. First, we build the mapping between all configurations $\forall g_{ij} \in \mathcal{G}_i$ and the allocated time resources. We use $T_{\mathcal{G}_i}$ to represent all assigned time windows according to the configurations of $\mathcal{G}_i$. Then we have $T_{\mathcal{G}_i} = \{T_{g_{i1}}\} \cup \cdots \cup \{T_{g_{ij}}\} \cup \cdots \cup \{T_{g_{ij_i}}\}$, which is an ordered set sorted by the start time of each time window, and,

$$T_{g_{ij}} = \begin{cases} \emptyset & \text{if } v_{g_{ij}} = 0, \\ \{[t, t + l_{g_{ij}} - 1] | t = x_{g_{ij}} + (r' - 1)p_{g_{ij}}, \\ \quad \forall r' \in \{1, \ldots, r_{g_{ij}}\}\} & \text{if } v_{g_{ij}} = 1. \end{cases} \tag{1}$$

The format of each time window is [the start time slot index, the end time slot index].

The arrival times and deadlines for all data packets of TF f_i are $\mathcal{A}_i = \{t_a | t_a = \lceil \phi_i + a \times T_i \rceil, \forall a \in \{0, 1, \ldots, J_i - 1\}\}$ and $\mathcal{D}_i = \{t_d | t_d = \lfloor t_a + D_i \rfloor, \forall t_a \in \mathcal{A}_i\}$, respectively. To fulfill the latency requirement, we have,

$$s\left(T_{\mathcal{G}_i}^j\right) \geq t_a^j \wedge e\left(T_{\mathcal{G}_i}^j\right) < t_d^j,$$
$$\forall T_{\mathcal{G}_i}^j \in T_{\mathcal{G}_i}, t_a^j \in \mathcal{A}, t_d^j \in \mathcal{D}, j \in [0, J_i - 1], \tag{2}$$

where $s(\cdot)$ and $e(\cdot)$ are the start time slot index and the end time slot index of each time window in $T_{\mathcal{G}_i}^j$, respectively.

Taking the two configurations of TF B in Figure 5(c) as an example, we have $T_{\mathcal{G}_B} = \{[1, 1], [5, 5], [10, 10], [15, 15]\}$, where [1, 1] and [5, 5] are derived from the first configuration $\langle [1, 1, 1], 1, 1, 4, 2, 1 \rangle$ and [10, 10] and [15, 15] from the second configuration $\langle [1, 1, 1], 10, 1, 5, 2, 1 \rangle$. Taking [15, 15] as an example, the start time slot index 15 is calculated by $x + (2 - 1) \times p$ (i.e., $10+(2-1)\times 5$), while the end time slot index 15 is from $x+(2-1)\times p+l-1$ (i.e., $10+(2-1)\times 5+1-1$). The arrival times and deadlines for all packets are $\mathcal{A}_B = \{0, 5, 10, 15\}$ and $\mathcal{D}_B = \{1, 6, 11, 16\}$ according to the initial offset ϕ of 0, the transmission period T of 5 and latency requirement D of 1. In essence, the constraints given in Equation (2) are employed to coordinate the timing information of the data packets (i.e., the arrival times and deadlines) and the schedule information indicated by $\mathcal{G}$.

6.2.3 Control Overhead. The number of valid configurations for data transmission determines the number of control messages. In addition, a control message should be issued before its corresponding new configuration except that the control message for the first configuration of each TF is assumed to be issued beforehand:

$$(v_{c_{ij}} == v_{g_{ij}}) \wedge (x_{c_{ij}} < x_{g_{ij}}), \forall \mathcal{G}_i \setminus \{g_{i0} \cup c_{i0}\}, \forall f_i \in \mathcal{F},$$
$$v_{g_{i0}} == 1 \wedge v_{c_{i0}} == 0, \forall f_i \in \mathcal{F}.$$

In Figure 5(c), the first control message is issued before the hyperperiod starts, and the second control message is transmitted in time slot 9, which is ahead of the start time 10 of *configurationTFB*. Since the first control message does not need to allocate resources, we let $v_{g_{i0}} == 1 \wedge v_{c_{i0}} == 0$ meaning that the first configuration is valid while the first control message is invalid. And for other

configurations and control messages, the numbers of the valid indicators should be the same to assign each configuration a corresponding control message. And finally, the constraint of $x_{c_{ij}} < x_{g_{ij}}$ is applied to guarantee that the control message is issued before the corresponding configuration.[5]

6.2.4 *Other Complementary Constraints.*

— *Range constraint:* The ranges of variables used in $\mathcal{G}$ are as follows:

$$\begin{aligned} &\forall f_i \in \mathcal{F}, \forall g_{ij} \in \mathcal{G}_i, \forall c_{ij} \in \mathcal{G}_i, \\ &x_{g_{ij}} \in \mathbb{N}_0, l_{g_{ij}} \in \mathbb{N}^+, p_{g_{ij}} \in \mathbb{N}^+, r_{g_{ij}} \in \mathbb{N}^+, v_{g_{ij}} \in \{0,1\}, \\ &x_{c_{ij}} \in \mathbb{N}_0, v_{c_{ij}} \in \{0,1\}, \\ &\alpha'_{g_{ij}} \in \{0,1\}, \forall \alpha'_{g_{ij}} \in \alpha_{g_{ij}}, \alpha'_{c_{ij}} \in \{0,1\}, \forall \alpha'_{c_{ij}} \in \alpha_{c_{ij}}. \end{aligned}$$

— *Data transmissions constraint:* Each data packet should be only allocated resources once:

$$\sum_{\forall g_{ij} \in \mathcal{G}_i} r_{g_{ij}} \times v_{g_{ij}} = J_i, \forall f_i \in \mathcal{F}.$$

— *Resource unit collision constraint:* In order to facilitate the description of the status of all RUs, we define a resource allocation indicator β_{ixy} as follows:

$$\beta_{ixy} = \begin{cases} 1 & \text{the RU located in time slot } x \\ & \text{and RB } y \text{ is assigned to } f_i, \\ 0 & \text{otherwise,} \end{cases}$$

which can be inferred from $\mathcal{G}$ with the resource allocation pattern $\alpha_{g_{ij}}$ and Equation (1) for data transmissions, as well as the resource allocation pattern $\alpha_{c_{ij}}$ and the offset of assigned time slot $x_{c_{ij}}$ for the corresponding control messages with $v_{c_{ij}} == 1$. Since each RU can only be used once, we have:

$$\sum_{f_i \in \mathcal{F}} \beta_{ixy} \leq 1, \forall x \in [0, HP-1], \forall y \in [0, RB_{max}-1],$$

where RB_{max} represents the maximum **number of RBs** (**NRBs**) of the overall resource grid.

6.3 Optimization Goal

Our goal is to schedule multiple traffic flows such that all the deadlines are satisfied and the number of needed RBs is minimized. The optimal solution is:

$$\underset{\mathcal{G}}{\operatorname{argmin}}\ \mathcal{R}(\mathcal{G}),$$

where $\mathcal{R}(\mathcal{G})$ is the total NRBs assigned over $\mathcal{G}$.

The above optimization problem is NP-hard. We provide the proof based on the **Strip-Packing Problem** (**SPP**) [22]. The task is to pack all rectangle items into the strip to minimize the used height, given a finite width but infinite height and a set of rectangle items. We consider a set of traffic flows, which have the same transmission period. Their latency requirements are equal to the transmission period. This means that each traffic flow only has a single data packet to be transmitted and the data transmissions can only occur within the current hyperperiod. The needed time slots and RUs for each data transmission are given and fixed, although for each transmission multiple candidates exist. Now, we can construct the mapping between SPP and the CG scheduling problem as follows:

[5]Since every configuration has the parameter x indicating the earliest time to take effect, there is no need to issue all control messages in order.

(1) Rectangles in SPP/data transmissions in CG: the width of the rectangle corresponds to the needed time slots of the data transmission; the height of the rectangle corresponds to the needed RBs of the data transmission.
(2) Finite width in SPP/hyperperiod in CG: in SPP, all rectangles are packed in a strip with a finite width. In CG, since all traffic flows have the same transmission period and latency requirement equalling the transmission period, all data transmissions can only be scheduled within the hyperperiod. Hence, the finite width in SPP corresponds to the hyperperiod in CG.
(3) Minimizing the height in SPP/minimizing the number of needed RBs in CG: the goal of the CG scheduling problem is to minimize the number of needed RBs, which corresponds to the goal of SPP minimizing the height.

The reduction from SPP to the CG scheduling problem can be done in polynomial time. Based on the mapping between SPP and the CG scheduling problem, minimizing the needed RBs in the above-mentioned CG problem is equivalent to a feasible solution to SPP. When there is a schedule to the CG scheduling problem, the needed RBs are minimized. Since the generated schedule contains the resource allocation pattern for all data transmissions, the schedule can be translated into a feasible solution (i.e., the locations of all rectangles in the strip) to SPP directly. When there is a feasible solution of SPP, the location and shape information of all rectangles can be utilized to construct a schedule, which can be regarded as a solution to the CG scheduling problem. Hence the conclusion.

Considering the above constraints, we formulated our problem with **satisfiability modulo theories (SMT)**. In order to solve the SMT model, we used the off-the-shelf Z3 solver[23]. While producing optimal solutions, the SMT-based technique suffers from severe scaling problems and can only be applied to small examples (experimental results are shown in Section 8). Therefore, we developed the heuristic approaches as presented in the next section.

7 Heuristic Techniques

In this section, two heuristic approaches are proposed. The first technique as the baseline, Co1, follows the basic idea of CG scheduling that all packets are scheduled by one configuration per TF, thus avoiding any control signaling overhead. Then, we propose the second technique, CoU, to explore the trade-off between scheduling flexibility and control overhead.

7.1 Algorithm: Co1 (1 Configuration)

In algorithm Co1, only one configuration is allowed per TF, meaning that all data transmissions for a given TF have the same resource allocation pattern (i.e., the same amount of time slots and the same RBs) and a dedicated fixed period between every two consecutive data transmissions. As shown in Figure 7, five scheduling parameters should be determined to represent the scheduling information in a single configuration for all data transmissions of one traffic flow. The needed scheduling parameters are as follows:

— Offset x (offset for the first data transmission);
— The number of time slots allocated for each data transmission w;
— The number of frequency RBs allocated for each data transmission h;
— Period p (time interval between every two adjacent data transmissions);
— The start index of the frequency RB b.

Taking Figure 7(a) for an example, the schedule for the current target TF can be represented by a tuple of the abovementioned five parameters, i.e., $\{x = 3, w = 2, h = 2, p = 6, b = 2\}$.[6] The

[6]For simplicity, we omit the units for these scheduling parameters. For parameters x, w, and p, the unit is time slot(s). For parameters h and b, the unit is RB(s).

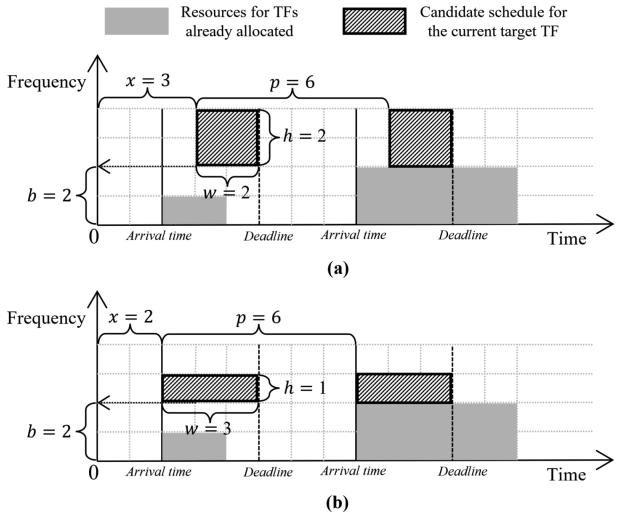


Fig. 7. Illustration of two candidate schedules for a TF with initial offset = slot 2, transmission period = 6 time slots, payload = 3 RUs, and latency requirement = 3 time slots.

allocated resources for the two data transmissions of the current target TF can be derived directly. For the first data transmission, the RUs in time slots [3, 4] (i.e., $[x, x+w-1]$) and RBs [2, 3] (i.e., $[b, b+h-1]$) are allocated. For the second data transmission, the allocated RUs use the same amount of time slots (i.e., 2 time slots) and the same RBs (i.e., RB 2 and RB 3) as the first data transmission. Since the parameter p is 6, the RUs for the second data transmission are in slots [9, 10] (i.e., $[x+p, x+w-1+p]$) and RBs [2, 3] (i.e., $[b, b+h-1]$). The goal of Co1 is to determine the abovementioned scheduling parameters for each TF such that all TFs' requirements are satisfied and the amount of needed RBs is minimized.

The main idea of the algorithm Co1 is as follows:

(1) Sort all TFs according to the "scheduling emergency" (see Section 7.1.1);
(2) For each TF, determine the corresponding scheduling parameters (see Section 7.1.2).
   — Calculate the feasible ranges of the scheduling parameters $\{x, w, h, p, b\}$;
   — Enumerate all combinations of the scheduling parameters $\{x, w, h, p, b\}$ according to the calculated feasible ranges and perform the collision checking operation to avoid resource reallocation; find all feasible candidate schedules;
   — Among all feasible candidate schedules, select the one with the minimal impact on the increase of RBs to achieve the local optimum for minimizing the usage of RBs.

As we stated, there are two main steps in Co1. We first sort all TFs and then generate the schedules for TFs one by one. For each TF, we aim to find the optimal scheduling parameters locally, which implies the minimization of needed RBs. The pseudo-code is given in Algorithm 1. Specifically, the inputs are the TFs' information $\mathcal{F}$, the RB_{max}, and the SNR values for all TFs, and the outputs are the aforementioned scheduling parameters $\{x, w, h, p, b\}$ for each TF. For more explanations, please refer to Section 7.1.1 and Section 7.1.2.

7.1.1 Preparation and Initialization ***(Line 1–5)****.* In this step, we need to process the inputs to the algorithm, initialize the scheduling parameters, and sort all TFs according to the scheduling emergency. The detailed operations are as follows:

ALGORITHM 1: Co1: 1 Configuration

Input: $\mathcal{F}$, RB_{max}, SNR values for all TFs
/* Stage 1: preparation and initialization */
1 Calculate the number of RUs needed, RU_i^d, for each TF f_i according to B_i, the corresponding SNR value, and the required block error rate
2 Calculate the hyperperiod HP and initialize the resource grid RG according to HP and RB_{max}
3 Calculate the number of data transmissions J_i according to HP and the transmission period T_i for each TF f_i
4 Declare $\{x_i, w_i, h_i, p_i, b_i\}$ for each TF f_i
5 $\mathcal{F} \leftarrow sort(\mathcal{F})$ ▷ Sort all TFs according to the scheduling emergency
/* Stage 2: for each TF, generate schedules */
6 **for** $i = 1$ *to* $|\mathcal{F}|$ **do**
7 $min_{b+h} \leftarrow RB_{max}$ ▷ Initialize min_{b+h} indicating the minimum $b + h$ of all explored candidate schedules
8 $w_{i_{max}} \leftarrow min(RU_i^d, D_i)$ ▷ Calculate maximum w_i according to RU_i^d and D_i
9 **for** $w_i' = 1$ *to* $w_{i_{max}}$ **do**
10 $h_i' \leftarrow \lceil \frac{RU_i^d}{w_i'} \rceil$ ▷ Calculate h_i' according to RU_i^d and w_i'
11 $x_{i_{max}} \leftarrow \phi_i + D_i - w_i'$ ▷ Calculate maximum x_i according to ϕ_i, D_i, and w_i'
12 $x_{i_{min}} \leftarrow \phi_i$ ▷ Calculate minimum x_i according to ϕ_i
13 **for** $x_i' = x_{i_{max}}$ *to* $x_{i_{min}}$ **do**
14 $p_{i_{min}} = T_i - \lfloor (x_i' - x_{i_{min}})/J_i \rfloor$ ▷ Calculate minimum p_i according to T_i, x_i', $x_{i_{min}}$ and J_i
15 $p_{i_{max}} = T_i + \lfloor (x_{i_{max}} - (x_i' + w_i' - 1))/J_i \rfloor$ ▷ Calculate maximum p_i according to T_i, $x_{i_{max}}$, x_i', w_i', and J_i
16 **for** $p_i' = p_{i_{min}}$ *to* $p_{i_{max}}$ **do**
17 **for** $b_i' = 0$ *to* $RB_{max} - h_i'$ **do**
18 **if** $b_i + h_i \geq RB_{max}$ **then**
19 **return** *no solution*
20 **end**
/* Calculate the indicated RUs for all data transmissions of TF f_i according to $\{x_i', w_i', h_i', p_i', b_i'\}$, J_i */
21 $RUs \leftarrow calculateRUs(x_i', w_i', h_i', p_i', b_i', J_i)$
/* Perform collision checking regarding the current RG */
22 **if** $collisionChecking(RG, RUs) == False$ **then**
23 **Continue**
24 **end**
/* For a schedule without collisions to the already allocated RUs, compare $b_i' + h_i'$ and min_{b+h} */
25 **if** $b_i' + h_i' < min_{b+h}$ **then**
26 $min_{b+h} \leftarrow b_i' + h_i'$ ▷ Update min_{b+h}
27 $x_i \leftarrow x_i'; w_i \leftarrow w_i'; h_i \leftarrow h_i'; p_i \leftarrow p_i'; b_i \leftarrow b_i'$ ▷ Update $\{x_i, w_i, h_i, p_i, b_i\}$
28 **end**
29 **endfor**
30 **endfor**
31 **endfor**
32 **endfor**
33 **else**
34 $updateResourceGrid(RG, x_i, w_i, h_i, p_i, b_i, J_i)$ ▷ Update the resource grid according to $\{x_i, w_i, h_i, p_i, b_i\}$, J_i
35 **end**
36 **endfor**
Output: $\{x_i, w_i, h_i, p_i, b_i\}$ for each TF f_i

— The number RU_i^d of needed RUs is calculated according to the payload, the SNR value, and the required block error rate for each TF **(Line 1)**.
— The hyperperiod HP needs to be calculated in order to initialize the resource grid RG. RG is a two-dimensional matrix indicating the allocation status of each RU **(Line 2)**.
— The number of data packets of each TF needs to be determined **(Line 3)**.
— The scheduling parameters $\{x, w, h, p, b\}$ need to be declared **(Line 4)**.
— Sort all TFs according to the scheduling emergency **(Line 5)**. The scheduling emergency is the ratio between the payload and the latency requirement. The TFs with a higher ratio would be scheduled earlier since those TFs are relatively hard to schedule and have a higher influence on other TFs.

7.1.2 Generate Schedules for Each TF ***(Line 6–36)***. In this step, we aim to generate schedules for all TFs one by one. Specifically, three sub-steps are performed.

Sub-step1: calculate the feasible ranges of the scheduling parameters $\{x, w, h, p, b\}$ according to TF's requirements **(Line 8–17)**.

— *Parameter w:*
For parameter w, the upper bound relates to the number RU^d of needed RUs and the latency requirement D at the same time **(Line 8)**. In any case, the data transmission can at most last for D time slots in order to satisfy the latency requirement. However, if RU^d is smaller than D, w is constrained by RU^d instead. In Figure 7(a), the payload requires 3 RUs and the latency requirement is 3, which results in $w_{max} = 3$. Suppose that the payload needs 4 RUs, w_{max} is 3 as well since w is constrained by the latency requirement, otherwise, a larger w would violate the deadlines.

— *Parameter h:*
Parameter h can be induced directly according to RU^d and w **(Line 10)**.

— *Parameter x:*
The upper bound of offset x can be calculated according to the initial offset ϕ, the latency requirement D, and the value of w **(Line 11)**. Different w corresponds to different x_{max}. The lower bound of the offset x is the same as the initial offset of the TF ϕ **(Line 12)**. In Figure 7(a), $w = 2$ results in $x = 2$ or $x = 3$, while in Figure 7(b), $w = 3$ leads to $x = 2$.

— *Parameter p:*
The aforementioned calculation of w and x can guarantee that the first data transmission will not violate the timing constraint (i.e., arrival time and deadline constraint). The parameter p would impact the other data transmissions. In general, p can simply be set up to the transmission period T of the TF, which naturally guarantees the correct timing information for all consequent data transmissions. In order to explore the range of p further, we calculate the lower bound and upper bound of p according to the formula given in **Line 14** and **Line 15**, respectively. If p is not the same as T, time shifting appears, which means that the relative timing relationship between the allocated resources and the corresponding arrival time and deadline would be different for different data transmissions. This results in the fact that even though the timing constraint is satisfied for the first data transmission, without an appropriate p, other data transmissions would violate the timing constraint. For $p \neq T$, we denote the difference of p and T as Δ. If $\Delta < 0$, data transmissions would get closer to the corresponding arrival times. Similarly, for $\Delta > 0$, the deadlines may be violated. Hence, we calculate p_{min} considering the time interval between the current value of x and the minimum offset x_{min} and the number of involved data transmissions J. For p_{max}, the difference between the finish time of the data transmission and the maximum offset x_{max} is calculated. For the case in Figure 7(a) where two data transmissions are involved with $x = 3$, and $w = 2$, p_{min} would be 5 and p_{max} would be 6 so that the timing constraints of the two data transmissions are not violated. If three data transmissions are considered, p can only be 6. This is because, if p is 5, the violation of the arrival time arises for the third data transmission since the arrival time of the third data packet is $\phi + (3-1) \times T = 14$ (i.e., $2 + (3-1) \times 6$), while the offset of the third data transmission is $x + (3-1) \times p = 13$ (i.e., $3 + (3-1) \times 5$). In Figure 7(b), $w = 3$ results in $p_{min} = p_{max} = 6$, no matter how many data transmissions would be involved.

— *Parameter b:*
Parameter b is simply from 0 to $RB_{max} - h$ (**Line 17**). When $b + h == RB_{max}$, the scheduling problem has no solution (**Line 18–20**).

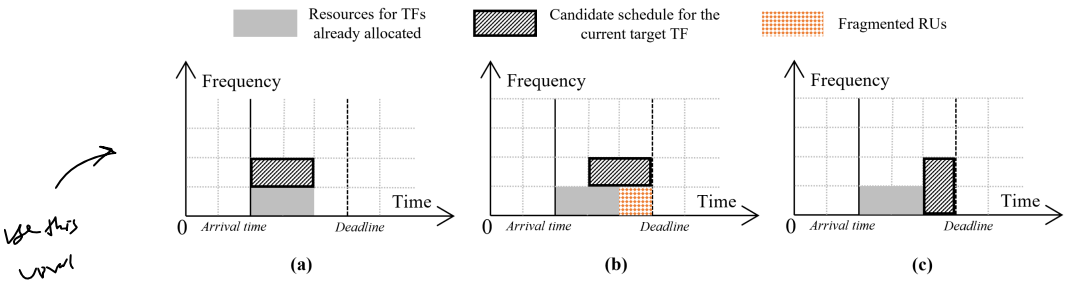


Fig. 8. Illustration of the intuition behind algorithm Co1 Line 13 and Line 9.

Sub-step2: enumerate all combinations of the scheduling parameters $\{x, w, h, p, b\}$, and perform the collision checking operations **(Line 21–24)**. The involved RUs can be derived via the scheduling parameters **(Line 21)**. The collision checking operation needs to be done to guarantee that the RUs of the current schedule are not being allocated to other TFs that are scheduled in the previous iterations **(Line 22)**. Taking the current candidate schedule in Figure 7(a) as an example, there is no collision between the current resource allocations with the already allocated resources. However, if we assume $b = 1$, collisions occur regarding the second data transmission for RUs in time slots 9 - 10, and RB 1. The schedules without any collisions can be regarded as candidate schedules.

Sub-step3: given the optimization goal of minimizing the usage of frequency RBs, among all feasible candidate schedules, we select the one with the minimal impact on the increase of RBs, which attempts to achieve the local optimum for the current TF **(Line 25–27)**.

In our model, this could be achieved by selecting the one with the minimal $b + h$ **(Line 25)**. For the current candidate schedule in Figure 7(a), four RBs are needed in total. For another candidate schedule in Figure 7(b) with $x = 2, w = 3, h = 1, p = 6$, and $b = 2$, three RBs are needed. Hence, schedule (b) is better compared to schedule (a). For the case in which two candidate schedules have the same $b+h$, we prefer the one with "larger x" and "smaller w", which is achieved by **Line 13** and **Line 9**, respectively. Compared to "larger x," "smaller w" is preferred. We illustrate the intuition behind the above preference via Figure 8, which consists of three candidate schedules for one data transmission of a TF with the payload of 2 RUs. Among Figure 8(a), (b), and (c), all $b + h$ are the same. Comparing (a) and (b), we prefer to select (b) because it can fulfill the deadline constraint thus providing more scheduling flexibility for other TFs. For a given w indicated by Line 9, a "larger x" means the finish time of the data transmission is closer to the deadline. This corresponds to the aforementioned statement that we prefer to select the candidate schedule with "larger x". When comparing (b) and (c), we prefer to select (c) with a "smaller w" since it implies better resource efficiency, which means that less **fragmented RUs** (**FRUs**) would arise. A schedule can create FRUs. An example is the RU below the candidate schedule (b), which is in time slot 4 and RB 0. That RU is relatively hard to allocate in the future for other TFs. The difficulty of utilizing the fragmented RU in this example is that it would be hard to find a candidate schedule under the constraint of h of 1 and x of 4 exactly at the same time for other TFs. However, with candidate schedule (c), that fragmented RU would be avoided.

7.1.3 Time Complexity. The calculation time is needed mainly by the iterations over the TFs (Line 6) including the enumeration of scheduling parameters (Line 8–17) and the access to matrix RG (i.e., collision checking operations) (Line 22). For the iterations over TFs, the complexity relies on the number of TFs $|\mathcal{F}|$. For scheduling parameters, w and x, both of them are restricted by D of each TF resulting in the complexity of $O(\overline{D})$, where $\overline{D}$ is the mean value of D for all TFs. Parameter

h can be induced by w directly, the complexity is simply $O(1)$. Parameter p leads to the complexity of $O(\overline{T})$ (Line 14–15), where $\overline{T}$ is the mean value of T for all TFs. The iteration of parameter b has the complexity of $O(RB_{max})$ (Line 17). Since RG is a two-dimensional matrix ($HP \times RB_{max}$), the access to RG takes $O(HP \times RB_{max})$ times (Line 19). In conclusion, the time complexity of algorithm Co1 is $O(|\mathcal{F}|\overline{T}(\overline{D} \times RB_{max})^2 HP)$.

7.2 Algorithm: CoU (Unlimited Configurations)

In contrast to Co1, in our CoU heuristic, we propose to utilize multiple configurations to enhance scheduling flexibility, while at the same time, we must consider the involved control messages (i.e., the scheduling overhead) carefully. In CoU, an unlimited number of configurations can be used. However, it is not trivial to determine the optimal number of configurations and the involved data transmissions. Note if a configuration is used for multiple consecutive data transmissions, the control overhead is reduced. This motivates us to find solutions, which minimize the number of used RBs without using an excessive number of configurations. As demonstrated in Figure 5, fewer configurations may lead to more frequency resource consumption, while more configurations introduce more control messages thus reducing the resource efficiency as well. The point is to find the most appropriate number of configurations and the corresponding data transmissions so that enough scheduling flexibility can be exploited with a reasonable number of control messages.

To obtain the schedule with multiple configurations, we aim to explore multiple configuration settings (i.e., different schedules) with different numbers of configurations for each TF and then select the best one as the scheduling decision. In this process, two questions need to be answered:

(1) *Question1:* Among different configuration settings, how to evaluate and compare their quality?
(2) *Question2:* How to explore the configuration settings efficiently?

Regarding *question1*, we develop the approach that utilizes the concept of "FRUs" and use the "schedule score" as the metric to evaluate the quality of the configuration setting. For detailed explanations, please refer to the illustrative example depicted in Figure 9 in Section 7.2.1.

Regarding *question2*, we develop a strategy based on iteratively applying the "combination process". The basic idea is as follows: for the current target TF, we initially generate the configuration setting with as many configurations as possible. Then, based on "combination processes", we reduce step by step the number of configurations to be explored in newly generated configuration settings. In this process, we use the "schedule score" to guide the exploration direction. Finally, among all explored configuration settings with different numbers of configurations, we select the one with the best accumulated "schedule score" as the locally optimal schedule. For more detailed explanations, please refer to Section 7.2.2.

7.2.1 Evaluate the Configuration Settings. Figure 9 illustrates the intuition behind the quality evaluation of different configuration settings in CoU. In Figure 9, two TFs, TF A and TF B, need to be scheduled. The parameters of TF A are transmission period = 2 time slots, payload = 2 RUs, and latency requirement = 2 time slots, while for TF B, the parameters are transmission period = 6 time slots, payload = 3 RUs, and latency requirement = 5 time slots. Figure 9(a)-(c) illustrates three configuration settings for TF A, while Figure 9(d)-(f) shows the schedules of TF B that are based on the corresponding configuration settings of TF A in Figure 9(a)-(c), respectively. As shown in Figure 9(a) and (b), the configuration settings both comprise two configurations. In the case of Figure 9(a), the first data transmission belongs to a configuration and the last two data transmissions need another configuration. Regarding the case in Figure 9(b), the first two data transmissions need a configuration, while the last data transmission requires another configuration. Therefore,

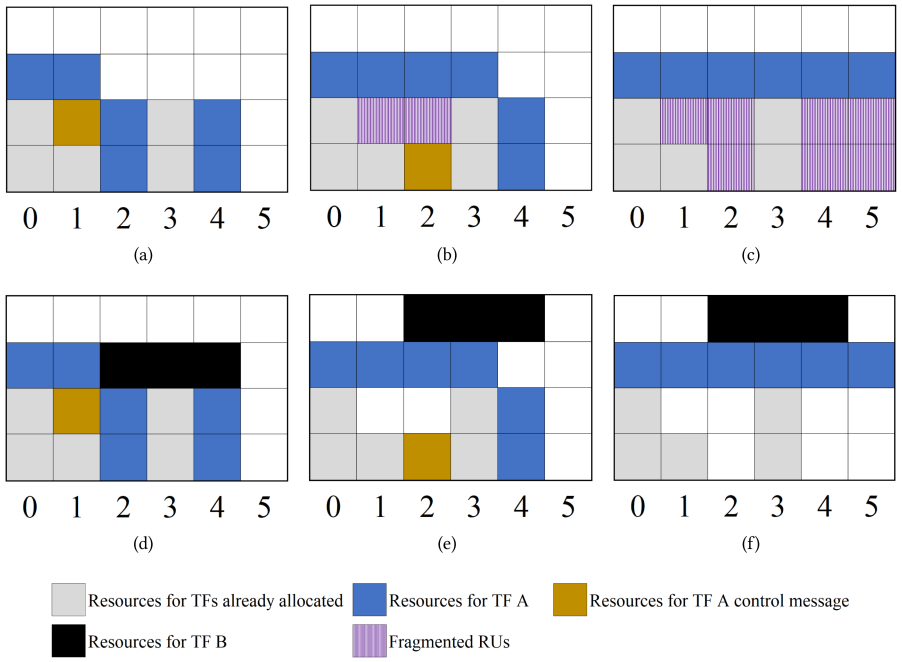


Fig. 9. The motivational example of the intuition of algorithm CoU.

for both of them, one control message needs to be scheduled.[7] Though the configuration settings in Figure 9(a) and (b) have the same cost regarding the control message, they impact the schedule of TF B, which is done after TF A has been scheduled, differently, which is shown in Figure 9(d) and (e). For the case in Figure 9(d), after scheduling TF B, three RBs are occupied, while in Figure 9(e), one more RBs are needed. Therefore, we know that even for the schedules with the same number of configurations, their impact on the overall resource efficiency can be different. If we consider further reducing the number of configurations (i.e., only a single configuration for TF A in this case), due to the lack of enough scheduling flexibility, the overall resource efficiency can also be reduced, as shown in Figure 9(c) and (f).

This example demonstrates that, in order to assess the quality of different configuration settings, we need to consider their potential impact on future TFs. Since it would introduce tremendous complexity if scheduling the current TF would consider all future TFs directly, in this work, we assess the impact on future TFs by the concept of FRUs as already mentioned in Section 7.1. In Figures 9(b) and (c), the RUs in purple are regarded as FRUs since these RUs are generally hard to be used for future TFs so that the overall resource efficiency would be reduced and more RBs are needed. This difficulty comes from the fact that only if a future TF has the data transmissions satisfying both the timing constraints (i.e., considerations on initial offset, transmission period, and latency requirement) and the payload constraints, those FRUs can be utilized. For example, as shown in Figure 9(e) and (f), all FRUs are not utilized by TF B due to the above-mentioned constraints. Formally, the unutilized RUs that are in the same slots as the data transmissions of the current target TF while with smaller RB indices are regarded as FRUs. In the following, we use the number of FRUs to assign a score (i.e., a penalty) called "schedule score" for each candidate

[7]The first control message is ignored since it is issued before the current resource grid starts.

schedule during the explorations of different configuration settings. For example, the scores for the schedules in Figure 9(a), (b), and (c) are 0, -2, and -7, respectively.

7.2.2 Overall Workflow of CoU. In Section 7.2.1, we demonstrate the impact of different configuration settings on resource efficiency and the calculation of the schedule score. The detailed description of CoU for the overall workflow is given in Algorithm 2. Like Co1, CoU consists of two steps as well. In **Step 1**, essential preparation and initialization are performed (**Line 1–4**). In **Step 2**, TFs are scheduled one by one according to the given order sorted by the scheduling emergency (**Line 5**). For the current target TF, initially, all data transmissions are scheduled individually (**Line 8–10**), therefore, each data transmission may correspond to a single configuration. Excessive control messages may exist so that RUs allocated to the control messages may negatively affect resource optimization. By iteratively applying the "combination process", the number of configurations for the candidate schedules reduces. In this process, many different configuration settings are explored. Then, candidate schedules with different numbers of configurations are obtained (**Line 13–29**). Finally, the schedule with the best schedule score is regarded as the most appropriate schedule of the current TF (**Line 31**), and the schedule of the corresponding control messages is generated according to the scheduling policy of Co1 (**Line 32**). More details of **Step 2** are given as follows.

- For the current TF, allocate resources for each data packet individually through *subco1*.[8] Each data transmission is regarded as a single **scheduling group** (**SG**). All SGs are the elements of the **scheduling group vector** (**SGV**), which indicates one candidate schedule (**Line 8–10**). The collection of the current candidate schedule and the corresponding schedule score is added to the **hyper scheduling group vector** (**HSGV**) (**Line 11–12**).
- For every two adjacent SGs in the current SGV, perform the pseudo-combination process (**Line 14–25**) including calculating the corresponding schedule score (**Line 16–19**) and recording the information of the best schedule (i.e., the one with the largest schedule score) among all combination iterations (**Line 20–24**).
- After obtaining the schedule with the largest schedule score, SGV will be updated (**Line 26–29**). Two old SGs will be removed and the newly generated SG will be added to SGV, which results in a decrease in the number of configurations. Then, the SGV as a new candidate schedule and the corresponding schedule score will be added to HSGV (**Line 29**). When all hyper iterations are finished (**Line 13–30**), all candidate schedules with different numbers of configurations are generated and stored at HSGV.
- Among all candidate schedules, the one with the best schedule score is regarded as the best schedule (**Line 31**). And the corresponding control messages of the best schedule can be determined (**Line 32**).

The exploration of different configuration settings is performed by the combination process as described in Algorithm 2 Line 13–30. Figure 10 depicts the exploration of different configuration settings for a single TF with five data packets. Initially, there are five data transmissions, which form five SGs as shown in Figure 10 SGV(a). This also means that five configurations exist and four control messages are needed to be scheduled. After that, four pseudo-combination processes (i.e., combinations between SGs ① and ②, ② and ③, ③ and ④, and ④ and ⑤) are to be performed. The combination process means that two adjacent SGs are scheduled by a single configuration. Therefore, the number of configurations can be reduced by one. In this process, for each pseudo-combination process, the corresponding schedule score is calculated. Among the four

[8]Since only part of the data packets needs to be scheduled, we apply subCo1, which is based on the same scheduling policy as Co1 but not scheduling all data packets for a TF.

ALGORITHM 2: CoU: Unlimited Configurations

Input: $\mathcal{F}$, RB_{max}, SNR values for all TFs
/* Stage 1: preparation and initialization */
1 Calculate the number of RUs needed RU_i^d for each TF f_i according to B_i, the corresponding SNR value, and the required block error rate
2 Calculate the hyperperiod HP and initialize the resource grid RG according to HP and RB_{max}
3 Calculate the number of data transmissions J_i according to HP and the transmission period T_i for each TF f_i
4 $\mathcal{F} \leftarrow sort(\mathcal{F})$
/* Stage 2: for each TF, generate schedules */
5 **for** $i = 1$ *to* $|\mathcal{F}|$ **do**
6 $SGV \leftarrow \emptyset$ ▷ Initialize Scheduling Group Vector SGV to empty
7 $HSGV \leftarrow \emptyset$ ▷ Initialize Hyper Scheduling Group Vector $HSGV$ to empty
8 **for** $j = 1$ *to* J_i **do**
9 $SGV.add(\{\{DT_{ij}\}, subCo1(\{DT_{ij}\})\})$ ▷ Each Data Transmission (DT) is scheduled individually and is regarded as a single Scheduling Group (SG)
10 **endfor**
11 $scheduleScore \leftarrow calculateScheduleScore(SGV)$
12 $HSGV.add(\{SGV, scheduleScore\})$
13 **for** $hyperIteration = J_i - 1$ *to* 1 **do**
/* For the current candidate schedule, perform pseudo-combination processes */
14 $maxScheduleScore \leftarrow -(HP \times RB_{max})$; $maxSG \leftarrow \emptyset$; $maxCombinationIndex \leftarrow 0$
15 **for** $combinationIteration = 0$ *to* $hyperIteration - 1$ **do**
16 $firstSG \leftarrow SGV[combinationIteration]$
17 $secondSG \leftarrow SGV[combinationIteration + 1]$
18 $combinedSG \leftarrow subCo1(\{firstSG, secondSG\})$
19 $currentScheduleScore \leftarrow calculateScheduleScore(firstSG, secondSG, combinedSG, SGV)$
20 **if** $currentScheduleScore > maxScheduleScore$ **then**
21 $maxSG \leftarrow combinedSG$
22 $maxScheduleScore \leftarrow currentScheduleScore$
23 $maxCombinationIndex \leftarrow combinationIteration$
24 **end**
25 **endfor**
/* Reduce the number of configurations and generate a new candidate schedule */
26 $SGV.remove(maxCombinationIndex + 1)$
27 $SGV.remove(maxCombinationIndex)$
28 $SGV.insert(maxCombinationIndex, maxSG)$
29 $HSGV.add(\{SGV, maxScheduleScore\})$
30 **endfor**
31 $schedule_i^d \leftarrow max(HSGV.scheduleScore)$ ▷ $schedule_i^d$: the schedule for data transmissions
32 $schedule_i^c \leftarrow subCo1(schedule_i^d)$ ▷ $schedule_i^c$: the schedule for control messages
33 **endfor**
Output: $schedule_i^d$, $schedule_i^c$ for each TF f_i

pseudo-combination processes, we select the one with the largest schedule score and perform the actual combination for those two involved SGs (i.e., SGs ③ and ④ in this example). Then, SGV(b) is formed as a new candidate schedule, which consists of four SGs meaning that at most four configurations are needed. We repeat the above operations to explore cases with different numbers of configurations further until only one SG is left. At the same time, we record each SGV (i.e., the SGVs (a), (b), (c), and so on) and the corresponding schedule score. Finally, we select the schedule with the largest schedule score as the most appropriate schedule for the current TF.

7.2.3 Time Complexity. The calculation time is needed mainly by the iterations over all hyper iterations (Line 13) and combination iterations (Line 15). The complexity is determined by the number of data packets of each TF, which is $O(\overline{J})^2$, where $\overline{J}$ is the average number of data packets of all TFs. In each combination process, *subCo1* is utilized, which has the same level of complexity as algorithm Co1 when considering a single TF. The calculation of the schedule score needs to

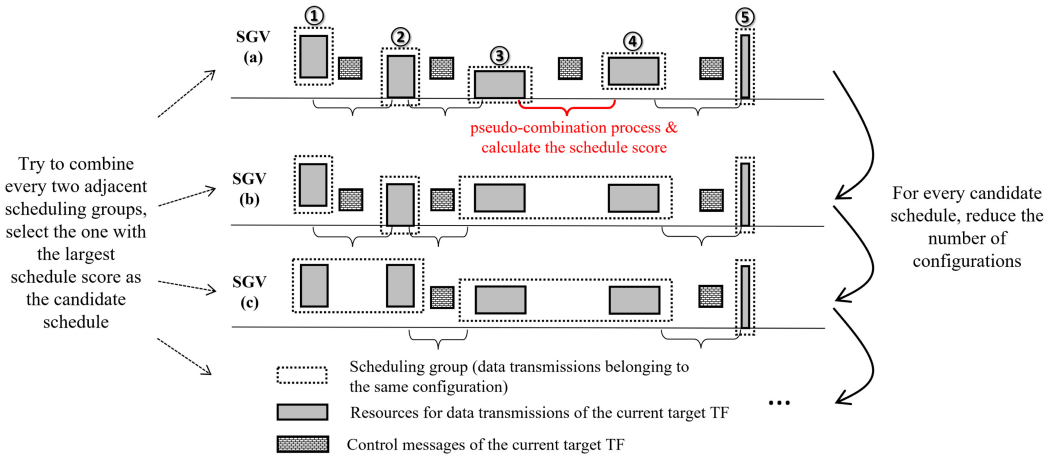


Fig. 10. Illustration of the exploration of different configuration settings.

check for the whole resource grid, which has the complexity of $O(HP \times RB_{max})$. In conclusion, the time complexity of algorithm CoU is $O(|\mathcal{F}|\overline{T}(\overline{D} \times RB_{max} \times \overline{J})^2 HP)$.

8 Experiments

In this section, we evaluate the performance and scalability of our proposed techniques as well as the superiority of algorithm CoU compared to the baseline Co1. When generating the input scenarios, we considered the following parameters:

- Number of TFs;
- Latency requirement;
- Transmission period;
- Payload size.

We considered three metrics to assess the performance of the algorithms, as follows:

- *NRBs* represents the consumption of frequency resources, which is the optimization goal;
- *Schedulable ratio* represents the ratio of the systems that are schedulable for a given NRBs (for the experiments producing this ratio, we introduced as an input the **maximum allowed NRBs (MANRB)**);
- *Optimization/execution time* reflects the scalability and time efficiency of the proposed algorithms.

Since the SMT-based solution suffers from the scalability issue, two scenarios are tested with different problem sizes: small-scale systems and large-scale systems. All algorithms are written in C++ and run on a Windows Desktop with Intel Xeon W-1250 CPU and 32 GB main memory. The open-sourced implementation is available online.[9]

8.1 Small-Scale Systems

In order to make the SMT models solvable within several hours, each test case only consists of three TFs, which are randomly selected from the following four candidate TFs: $\langle[0, 1], 1, 20, 1\rangle$, $\langle[0, 1], 2, 30, 1\rangle$, $\langle[0, 2], 3, 50, 1\rangle$, and $\langle[0, 2], 5, 80, 3\rangle$ in the format of $\langle$[minimum initial offset (ms), maximum initial offset (ms)], transmission period (ms), payload (bytes), latency requirement (ms)$\rangle$

[9]For more details: https://github.com/dlzrmr99/CGScheduling

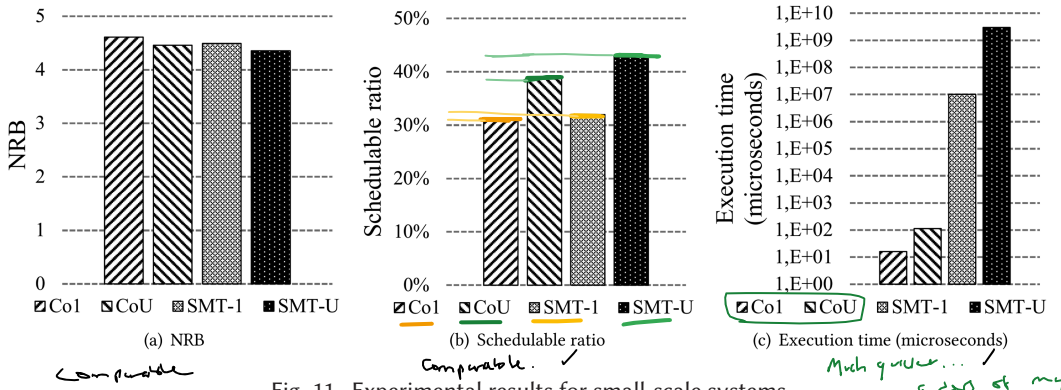


Fig. 11. Experimental results for small-scale systems.

and MARB of 5. Among 200 test cases, the initial offset is uniformly selected between the minimum initial offset and the maximum initial offset. In addition, a randomly selected SNR value in the range [2, 20] dB is applied to every TF. Each control message occupies one RU. The duration of each time slot is 1 ms. Four approaches are tested including SMT-U (SMT-based solution with an unlimited number of configurations per TF), SMT-1 (SMT-based solution with one configuration per TF), CoU, and Co1. The experimental results for small-scale systems are shown in Figure 11. They represent the average values of the metrics of all test cases. Note that only the statistics of successfully scheduled test cases among all four approaches are included for calculating NRB.

As shown in Figure 11(a) and Figure 11(b), SMT-U performs the best among all approaches and achieves the lowest NRB = 4.35 and the highest schedulable ratio = 43%. This is no surprise, since SMT-U produces the optimal result and, as opposed to SMT-1, it is not limited to one single configuration per TF. However, the good performance of SMT-U comes with a dramatically increased solving time of up to 3,000 seconds on average for such small systems, which is unacceptable. Regarding SMT-1, due to the restriction to a single configuration per TF (i.e., the lack of scheduling flexibility), it performs worse than SMT-U in terms of both NRB = 4.49 and schedulable ratio = 32%.

Compared with Co1 and even SMT-1, CoU performs better with regard to the schedulable ratio and NRB. The advantage of CoU comes from the exploitation of the scheduling flexibility without excessive control overhead so that more test cases are scheduled, hence a higher schedulable ratio = 39%. Because when calculating NRB, only the test cases having solutions for all approaches are taken into account, the superiority of CoU in terms of NRB compared with Co1 and SMT-1 is insignificant. Though compared with SMT-U, there is a performance gap for CoU in terms of NRB and schedulable ratio, the execution times (i.e., 110 microseconds on average) are 3×10^7 times shorter. This makes CoU appropriate to handle large-scale problems. Regarding Co1, due to the lack of scheduling flexibility, fewer test cases can be scheduled compared with CoU and SMT-U. The experiments demonstrate the quality and scalability of the proposed heuristics compared to the optimal solutions.

8.2 Large-Scale Systems

In order to evaluate the scalability and performance of our heuristic approaches, extensive test cases are generated and tested for large-scale systems. We first establish a *default* scenario with the parameters listed in Table 4. The parameters are selected according to the requirements of typical industrial applications (Table 1). We first start conducting experiments with the default settings (shown in Table 4) to study the impact of the number of TFs. After that, in order to evaluate the individual impact of latency requirement, transmission period, and payload size, we conduct experiments for each parameter specifically and let other parameters keep constant.

Table 4. Parameters for Large-scale Systems (Default)

Parameter	Value
SNR	[2, 20] (dB)
Number of TFs	{10, 20, 30, 40, 50, 60, 70, 80}
The ratio between the latency and the corresponding transmission period	[0.2, 0.6]
Transmission period	[2, 3, 4, 5, 6] (ms)
Payload size	[40, 250] (bytes)
Control message size	1 RU
Duration of the time slot	0.25 ms

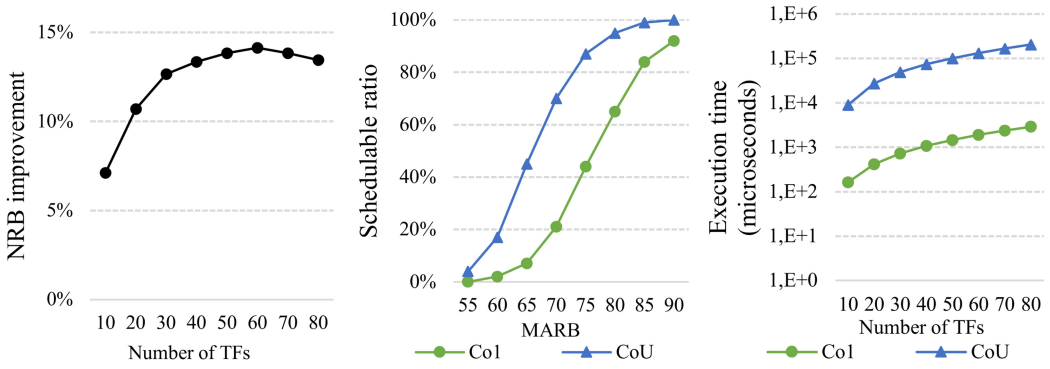
8.2.1 Default Scenario. For the default scenario, we mainly focus on the *number of TFs*. As per Table 4, 8 groups of test cases with different numbers of TFs ranging from 10 to 80 are considered. The experimental results are the average of 1,000 cases for each group. Note that the NRB obtained here is not the same as in the case of small-scale systems. We assume that an unlimited NRBs are available. Thus, while trying to minimize the NRBs, eventually a schedulable solution will be found in each case. For the particular experiments targeted toward evaluating the schedulable ratio, we set up the MARB threshold. If the minimum NRBs needed in order to find a feasible solution is larger than the threshold, the corresponding test case is considered not schedulable.

Experimental results are given in Figure 12, which demonstrates the superiority of CoU. Figure 12(a) demonstrates the NRB improvement (i.e., the decrease in the NRBs) of CoU compared to Co1. The NRB improvement is in the interval of 7%–14% for different numbers of TFs. This means that even though more configurations introduce additional control messages, the profit of having much higher scheduling flexibility still helps to utilize resources more efficiently.

Regarding the schedulable ratio, for simplicity, we only select cases with 80 TFs, and the MARB ranges from 55 to 90. The results given in Figure 12(b) show that CoU obtains better schedulable ratio compared to Co1. The gap can be up to 50% in the case of 70 MARBs. This shows that for a harsher scheduling task with the MARB constraint, CoU outperforms Co1 significantly. Figure 12(c) shows the execution time (in microseconds) on a logarithmic scale with the increase in the number of TFs. We achieve execution time for CoU below 200 ms for problems with 80 TFs and around 270,000 data packets in each test case.

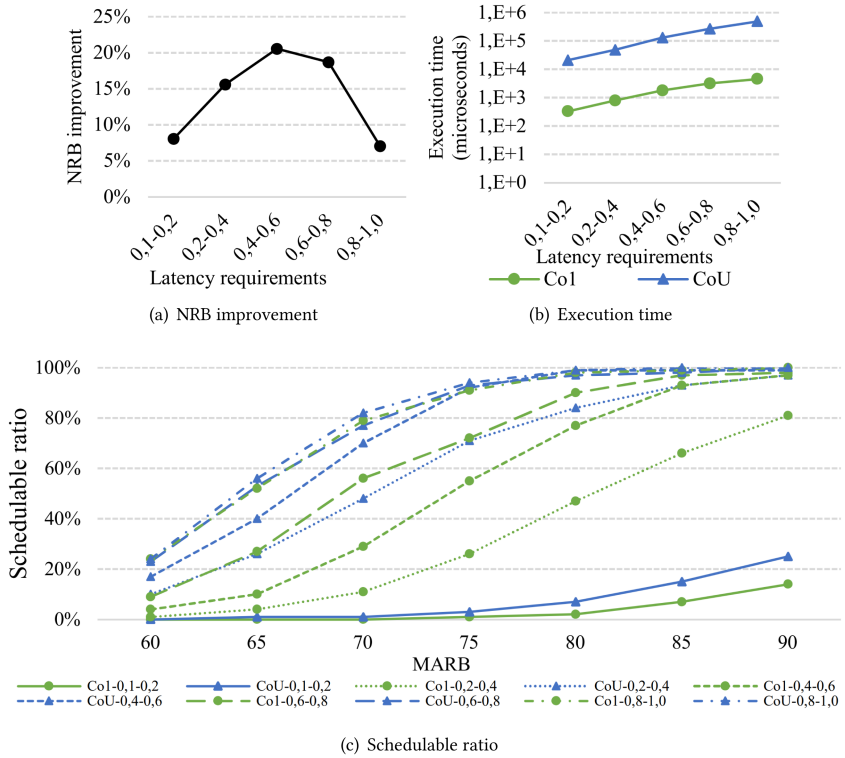
8.2.2 Latency Requirements. In the following, we focus on the parameter *latency requirement*. As shown in Table 4, we define the latency requirement as an interval $[a, b]$, in which a and b are the minimal and the maximal ratios between the actual latency and the corresponding transmission period of a certain TF. We set up five intervals including $[0.1, 0.2]$, $[0.2, 0.4]$, $[0.4, 0.6]$, $[0.6, 0.8]$, and $[0.8, 1.0]$, which represent various latency requirements from the most stringent to the most relaxed. The actual latency requirements are generated randomly with a uniform distribution inside this interval. The experimental results are given in Figure 13. The data regarding NRB improvement and execution time is obtained by averaging the results of cases with different numbers of TFs ranging from 10 to 80. The schedulable ratio is only for cases of 80 TFs.

Figure 13(a) and Figure 13(c) demonstrate that the performance gap between CoU and Co1 is relatively small (i.e., around 7%) in the cases of extremely tight $[0.1, 0.2]$ and extremely loose $[0.8, 1.0]$ latency requirements in terms of NRB improvement and schedulable ratio. For all other cases, CoU outperforms Co1 by a large gap of up to 20%. Under very tight latency constraints, there is very little room to exploit the potential flexibility of CoU. Under very loose latency constraints, there is very little need for such a flexibility and Co1 can do well likewise. As shown in Figure 13(b), the execution time of CoU is larger compared to Co1.



(a) NRB improvement (b) Schedulable ratio (c) Execution time (microseconds)

Fig. 12. Experimental results for the default scenario.



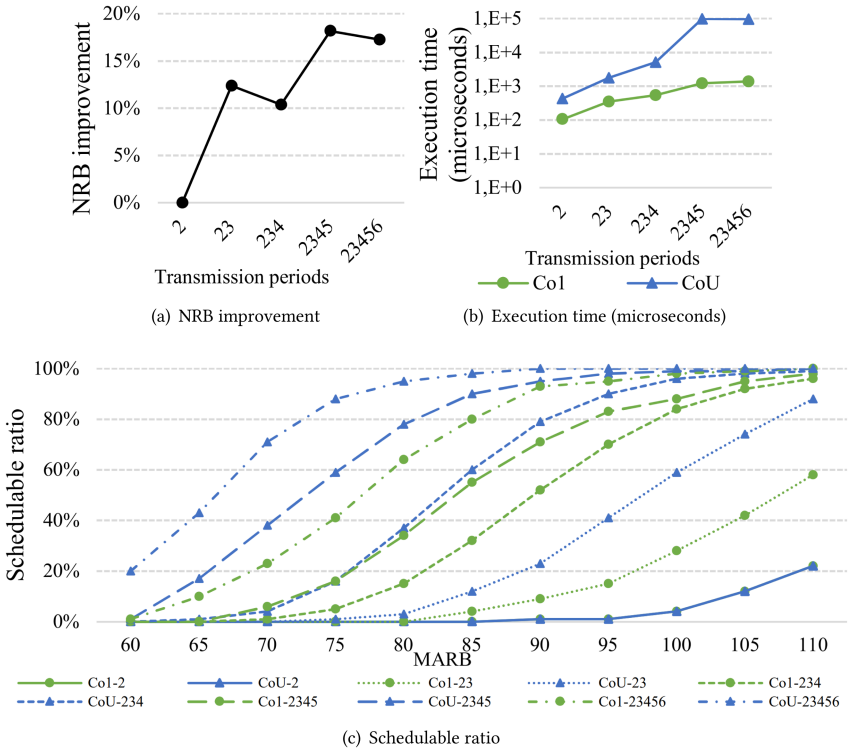
(a) NRB improvement (b) Execution time

(c) Schedulable ratio

Fig. 13. Experimental results with various latency requirements.

In conclusion, CoU performs much better in the case of moderate latency requirements. For other extreme cases, Co1 and CoU perform closely.

8.2.3 Diversity of Transmission Periods. In the following experiments, we focus on the parameter *transmission period*. Specifically, we aim to study the impact of the diversity of transmission periods. We set up five different transmission period groups, including $\{2\}$, $\{2, 3\}$, $\{2, 3, 4\}$, $\{2, 3, 4, 5\}$, $\{2, 3, 4, 5, 6\}$. In a certain transmission period group, all test cases have the same transmission



(a) NRB improvement (b) Execution time (microseconds)

(c) Schedulable ratio

Fig. 14. Experimental results for the tests of the diversity of transmission periods.

period distribution. The five transmission period groups generally show an increased diversity/complexity. The results are shown in Figure 14. As shown in Figure 14(a) and Figure 14(c), the performance gap between CoU and Co1 in terms of NRB improvement and schedulable ratio is becoming larger with the increase of the diversity of transmission periods. In general, with higher diversity, the interleaving behavior among TFs becomes more complex. CoU outperforms Co1 in such a complex scenario. The cost of handling such complex scenarios is the increase in execution time, especially when the hyperperiod increases dramatically with higher diversity of transmission periods. The increase in hyperperiod generally corresponds to the increase in the number of data packets J to be scheduled, which is an important factor impacting the time efficiency of CoU.

In conclusion, CoU is more appropriate for handling complex cases with higher diversity of transmission periods than Co1.

8.2.4 Payload Sizes. In this section, we focus on the parameter *payload size*. First, we study the impact of the diversity of payload sizes. Second, we aim to explore the impact of the relationship between the payload size and the control message size.

In order to study the impact of the diversity of payload sizes, we generate test cases with the same average payload size (i.e., 150 bytes), but from different ranges including [150, 150], [120, 180], [90, 210], [60, 240], and [30, 270]. The experiments show that there is no significant variation (within 5%) when comparing the results of CoU and Co1 among the above-mentioned five test scenarios. The reason is that both CoU and Co1 explore time-frequency resource allocations efficiently with various numbers of RBs and time slots. Hence, even regarding the case [30, 270] with a huge diversity of payload sizes, good performances for both Co1 and CoU are still expected.

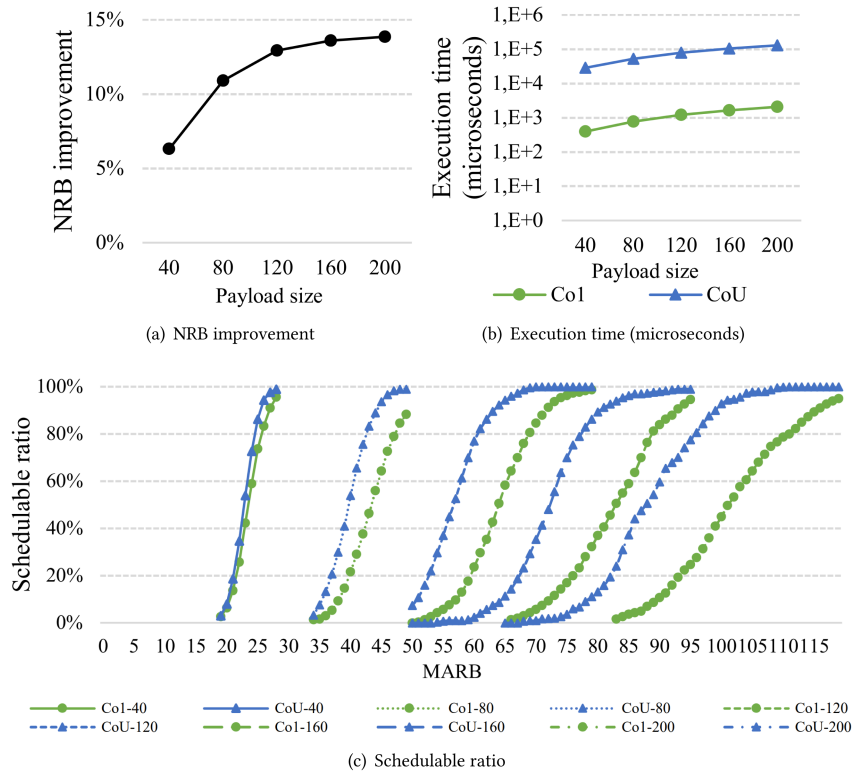


Fig. 15. Experimental results for the tests of different payload sizes.

Indeed, a higher level of diversity does impact the performance, though only slightly. The reason is that extremely large or small payload sizes may induce very harsh scheduling conditions.

In order to study the impact of the relationship between the sizes of the payload and the control message, we conduct experiments with test cases of different payload sizes. The control message size is 1 RU in our model as stated in Section 5.1. Five payload size groups are tested including 40 bytes, 80 bytes, 120 bytes, 160 bytes, and 200 bytes. In each group, all test cases and corresponding TFs have the same payload size. The experimental results regarding NRB improvement, execution time, and schedulable ratio are shown in Figure 15, where the results are the average value for different numbers of TFs ranging from 10 to 80.

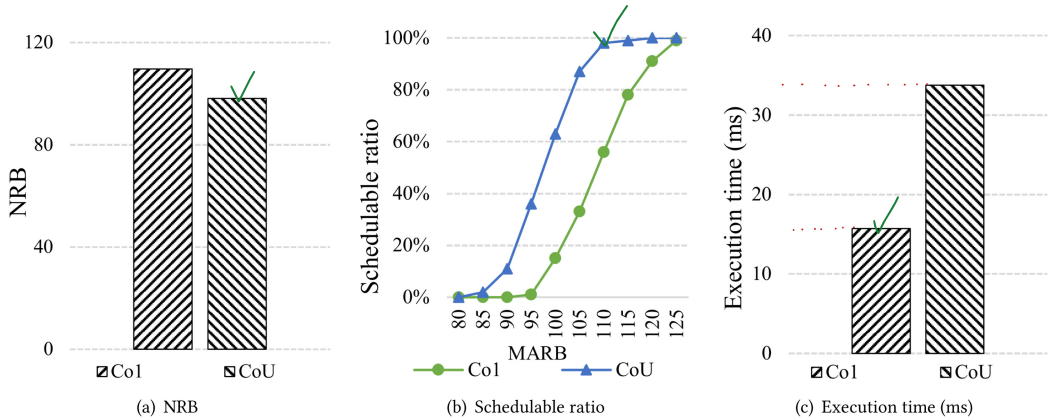
Figure 15(a) and Figure 15(c) demonstrate that with the increase of payload size, CoU performs much better, which means that for a smaller control overhead relative to the payload, the increased scheduling flexibility provided by CoU should be utilized to improve the scheduling performance. Naturally, with larger payload sizes, the execution time gets longer due to a larger searching space as shown in Figure 15(b), due to more iterations on the parameter w in Algorithm 1 and Algorithm 2. In conclusion, with larger payload sizes (i.e., the ratio between the payload size and the control message size is larger), the advantages of CoU become more prominent.

8.3 Realistic Case Study: Factories of the Future

In order to evaluate the performance of our proposed solutions in a realistic scenario (i.e., factories of the future), we established the experimental conditions in accordance with the descriptions and requirements in the relevant reference [4, 16]. The selected periodic TFs are characterized by

Table 5. Experimental Settings of Selected Industrial Applications [4, 16]

Applications	Transmission period	Payload	Latency requirement	# Nodes
Robotic motion control	2 ms	80 bytes	1 ms	40
Machine control	5 ms	256 bytes	3 ms	50
Control-to-control communication	10 ms	1 Kbytes	6 ms	10



(a) NRB (b) Schedulable ratio (c) Execution time (ms)

Fig. 16. Experimental results of the realistic case study.

transmission periods of 2 ms, 5 ms, and 10 ms, while the corresponding latency requirements are 1 ms, 3 ms, and 6 ms, respectively. In total, 100 TFs are scheduled by Co1 and CoU. The detailed settings of selected industrial applications are shown in Table 5. The offsets and the corresponding SNR values ranging from 2 to 20 dB are randomly selected for each TF. The time duration of each time slot is 0.25 ms. The experimental results shown in Figure 16 are obtained by averaging 1,000 test cases with different seed values.

CoU outperforms Co1 in terms of NRB and schedulable ratio as shown in Figure 16(a) and Figure 16(b). CoU achieves about 11% NRB improvement compared to Co1. The superiority of CoU regarding the schedulable ratio is significant. The gap in the schedulable ratio between CoU and Co1 can be up to 54% when MARB is 105. The advantage of CoU comes at a cost. A 2 times longer execution time of 33.75 ms is needed compared to the execution time of 15.7 ms of Co1.

9 Conclusion

This article addresses the configured grant scheduling problem while considering multiple real-time applications and resource optimization. The SMT-based exact solution and two heuristic solutions are proposed. The extensive experiments demonstrate the performance and superiority of the proposed CoU heuristic approach.

Appendix

A Acronyms

5G	the fifth generation technology standard
CG	configured grant
URLLC	ultra-reliable low latency communication
SMT	satisfiability modulo theories
QoS	quality-of-service
BS	base station

SPS	semi-persistent scheduling
VoIP	voice over Internet protocol
MCS	modulation and coding scheme
RG	resource grid
RB	resource block
RU	resource unit
TF	traffic flow
HP	hyperperiod
SNR	signal-to-noise ratio
FRU	fragmented resource unit
CRU	resource unit for control message
DRU	resource unit for data transmission
SG	scheduling group
SGV	scheduling group vector
HSGV	hyper scheduling group vector
NRB	number of resource blocks
MARB	maximum allowed number of resource blocks

References

[1] Dahlman, Erik, Parkvall, Stefan, Skold, Johan. 2020. 5G NR: The next generation wireless access technology.

[2] 2020. Service requirements for cyber-physical control applications in vertical domains, document 3GPP. TS 22.104 V16.5.0, Sep. 2020.

[3] 2021. NR; Physical layer procedures for data, document 3GPP. TS 38.214 V16.8.0, Dec. 2021.

[4] 2020. Study on communication for automation in vertical domains, document 3GPP. TR 22.804 V16.3.0, Jul. 2020.

[5] 2021. NR; NR and NG-RAN overall description, document 3GPP. TS 38.300 V16.8.0, Dec. 2021.

[6] 2021. Radio Resource Control (RRC) protocol specification, document 3GPP. TS 38.331 V16.7.0, Dec. 2021.

[7] Pan, Yungang, Mahfouzi, Rouhollah, Samii, Soheil, Eles, Petru, Peng, Zebo. 2023. Resource optimization with 5G configured grant scheduling for real-time applications (extended abstract). In *Proceedings of the 2023 Design, Automation & Test in Europe Conference & Exhibition*. 1–2.

[8] Zhang, Tianyu, Hu, Xiaobo Sharon, Han, Song. 2023. Contention-free configured grant scheduling for 5g urllc traffic. In *Proceedings of the 2023 60th ACM/IEEE Design Automation Conference*. 1–6.

[9] Larrañaga, Ana and Lucas-Estañ, M Carmen and Martinez, Imanol and Gozalvez, Javier. 2023. 5G configured grant scheduling for 5G-TSN integration for the support of industry 4.0. In *Proceedings of the 2023 18th Wireless On-Demand Network Systems and Services Conference*. 72–79.

[10] Abu-Ali, Najah, Taha, Abd-Elhamid M, Salah, Mohamed, Hassanein, Hossam. 2013. Uplink scheduling in LTE and LTE-advanced: Tutorial, survey and evaluation framework. *IEEE Communications Surveys & Tutorials*. 16, 3 (2013), 1239–1265.

[11] Afrin, Nusrat, Brown, Jason, Khan, Jamil Y. 2014. An adaptive buffer based semi-persistent scheduling scheme for machine-to-machine communications over LTE. In *Proceedings of the 2014 8th International Conference on Next Generation Mobile Apps, Services and Technologies*. 260–265.

[12] Afrin, Nusrat, Brown, Jason, Khan, Jamil Y. 2015. Design of a buffer and channel adaptive LTE semi-persistent scheduler for M2M communications, In *Proceedings of the 2015 IEEE International Conference on Communications*. 5821–5826.

[13] Afrin, Nusrat, Brown, Jason, Khan, Jamil Y. 2022. A multi-service adaptive semi-persistent LTE uplink scheduler for low power M2M devices, Future Internet, 107.

[14] Brown, Jason, Khan, Jamil Y. 2013. Key performance aspects of an LTE FDD based smart grid communications network. *Computer Communications*. 36, 5 (2013), 551–561.

[15] Feng, Ye, Nirmalathas, Ampalavanapillai, Wong, Elaine. 2019. A predictive semi-persistent scheduling scheme for low-latency applications in LTE and NR networks. In *Proceedings of the ICC 2019-2019 IEEE International Conference on Communications*. 1–6.

[16] García-Morales, Jan and Lucas-Estañ, M Carmen and Gozalvez, Javier. 2019. Latency-sensitive 5G RAN slicing for industry 4.0. *IEEE Access*. 7 (2019), 143139–143159.

[17] Jiang, Dajie, Wang, Haiming, Malkamaki, Esa, Tuomaala, Esa. 2007. Principle and performance of semi-persistent scheduling for VoIP in LTE system. In *Proceedings of the 2007 International Conference on Wireless Communications, Networking and Mobile Computing*. 2861–2864.

[18] Jiang, Nan, Aijaz, Adnan, Jin, Yichao. 2021. Recursive periodicity shifting for semi-persistent scheduling of time-sensitive communication in 5G. In *Proceedings of the 2021 IEEE Global Communications Conference*. 01–06.
[19] Karadag, Goksu, Gul, Recep, Sadi, Yalcin, Ergen, Sinem Coleri. 2019. QoS-constrained semi-persistent scheduling of machine-type communications in cellular networks. *IEEE Transactions on Wireless Communications*. 18, 5 (2019), 2737–2750.
[20] Li, Yu-Ngok Ruyue, Chen, Mengzhu, Xu, Jun, Tian, Li, Huang, Kaibin. 2020. Power saving techniques for 5G and beyond. *IEEE Access* 18 (2020), 108675–108690.
[21] Liu, Yan, Deng, Yansha, Elkashlan, Maged, Nallanathan, Arumugam, Karagiannidis, George K . 2020. Analyzing grant-free access for URLLC service, *IEEE Journal on Selected Areas in Communications*. 39, 3 (2020), 741–755.
[22] Lodi, Andrea, Martello, Silvano, Monaci, Michele. 2002. Two-dimensional packing problems: A survey. *European Journal of Operational Research*. 141, 2 (2002), 241–252.
[23] Moura, Leonardo de and Bjørner, Nikolaj. 2008. Z3: An efficient SMT solver. In *Proceedings of the International conference on Tools and Algorithms for the Construction and Analysis of Systems*. 337–340.
[24] Rao, Sriganesh K, Prasad, Ramjee. 2018. Impact of 5G technologies on industry 4.0, Wireless personal communications. *Wireless Personal* 100 (2018), 145–159.
[25] Sousa, Ivo and Queluz, Maria Paula and Rodrigues, António. 2020. A survey on QoE-oriented wireless resources scheduling. *Journal of Network and Computer Applications* 158 (2020), 102594.
[26] Yang, Wen-Bin, Souryal, Michael. 2014. LTE physical layer performance analysis. Vol. NISTIR 7986. US Department of Commerce, National Institute of Standards and Technology.
[27] You, Chunhua, Zhang, Yuan. 2014. A radio resource scheduling scheme for periodic M2M communications in cellular networks. In *Proceedings of the 2014 6th International Conference on Wireless Communications and Signal Processing*. 1–5.

Received 2 June 2023; revised 1 April 2024; accepted 10 April 2024