Journal of Network and Systems Management (2024) 32:65
https://doi.org/10.1007/s10922-024-09837-5

End-to-End No-wait Scheduling for Time-Triggered Streams in Mixed Wired-Wireless Networks

Gourav Prateek Sharma[1] · Wouter Tavernier[2] · Didier Colle[2] · Mario Pickavet[2] · Jetmir Haxhibeqiri[2] · Jeroen Hoebeke[2] · Ingrid Moerman[2]

Received: 29 October 2023 / Revised: 27 May 2024 / Accepted: 4 June 2024

Abstract
Proprietary communication technologies for time-critical communication in industrial environments are being gradually replaced by Time-sensitive Networking (TSN)-enabled Ethernet. Furthermore, attempts have been made to bring TSN features into wireless networks so that the flexibility of wireless networks can be utilized, and the end-to-end timings for Time-Triggered (TT) streams can be guaranteed. Given a mixed wired-wireless network, the scheduling problem should be solved for a set of TT stream requests. In this paper, we formulate the no-wait scheduling problem for mixed wired-wireless networks as a Mixed Integer Linear Programming (MILP) model with the objective of minimizing the flowspan. We also propose a relaxation of the original MILP in the form of a 2-stage MILP formulation. Next, a scalable approach based on the greedy heuristic is proposed to solve the problem for realistic-size networks. Evaluation results show that the greedy heuristic is suitable for realistic problem sizes where the MILP-based approach is found to be practically infeasible. Furthermore, the impact of wireless requests on the performance of the greedy heuristic is reported.

Keywords TSN · Ethernet · TAS · WiFi · ILP · MILP · Scheduling

1 Introduction

Due to its cost-effectiveness and maturity, Ethernet has been the de-facto link layer technology in standard communication networks. However, certain applications such as industrial automation, aviation systems and audio-video bridging (AVB)

✉ Gourav Prateek Sharma
gpsharma@kth.se

[1] EECS, KTH Royal Institute of Technology, 114 28 Stockholm, Sweden

[2] IDLab, Department of Information Technology, Ghent University - IMEC, 9052 Ghent, Belgium

Published online: 08 July 2024

require stringent guarantees on the timings of data exchange between endpoints (e.g., sensors, controller and actuators), which are difficult to promise in Ethernet [1]. For instance, the closed-loop control used for motion control of an industrial robot typically requires sub-10 ms latency [2]. Traditionally, networking in such domains is dominated by proprietary communication technologies such as Time-Triggered Ethernet (TTEthernet) and EtherCAT which are inflexible and costly but guarantee timely delivery of Time Sensitive (TS) traffic. To bring time sensitivity to Ethernet, the IEEE 802.1 Time-Sensitive Networking (TSN) Task Group (TG) released a suite of standards in 2012 [3]. This suite of standards specifies various mechanisms (e.g., time-synchronization, time-aware shaping) aimed at enabling the co-existence of time-triggered (TT) traffic that requires end-to-end timing guarantees along with standard Best-Effort (BE) traffic streams that do not need any timing assurances.

Time-Aware Shaping (TAS) or IEEE 802.1Qbv is one of the TSN components that emulates Time Division Multiple Access (TDMA) on each egress port [4]. The transmission time on links is divided into time slices that are assigned to each TT stream, thus ensuring no interference from other TT or BE streams. Link-layer technologies (e.g., Ethernet) based on wired communications provide stable and consistent communication as the frame transmission in the wired medium is more reliable than in wireless medium that suffers from inference issues. Therefore, the majority of industrial communication networks are based on wired links, which offer performance guarantees and high reliability needed to automate production processes [5]. However, wired networks, being inflexible, are not well suited in dynamic environments (e.g., with moving machine parts and robots). For such scenarios, wireless networks are quite practical.

A wireless network that connects several devices can be quickly deployed without the need for complex cable installation, resulting in cost savings. It also enables the connectivity of endpoints that are not accessible by cables, especially in dynamic industry environments. Other advantages include easier reconfigurability of factory settings and scalability. Owing to the evolution of high-speed wireless networking and the above advantages, networks consisting of both wired and wireless segments are prevalent in all kinds of industry environments [6]. End-to-end guarantees in such *mixed wired-wireless networks* can be provided by implementing TSN mechanisms in wireless, in addition to wired links. Implementation of such TSN mechanisms in wireless has been gaining some attention, recently, given the importance of time-sensitivity in several application domains. In [7], a Proof of Concept (PoC) has been demonstrated for wireless TSN, which is based on WiFi-based Software-Defined Radio (SDR) called OpenWifi [8]. The PoC implements time-synchronization and time-aware shaping in the wireless domain. Considering the prospects of wireless TSN, the problem of scheduling TT streams on a given mixed wired-wireless network still remains a challenge [9]. The end-to-end scheduling problem has been addressed for wired TSN widely [10–12]. The scheduling algorithms proposed in these works do not consider the specific characteristics of mixed wired-wireless networks. However, modeling of the end-to-end scheduling problem for a mixed wired-wireless network is not straightforward because of

these reasons: (i) the difference in transmission speeds of wired and wireless links, (ii) half-duplex communication in wireless links as opposed to that of wired links and (iii) sharing of wireless links among Wireless Access Point (WAP) and multiple endpoints. To this end, this paper addresses the scheduling problem for mixed wired-wireless networks. More concretely, the contributions of this paper are as follows:

1. We propose the transformation of a wireless link to a graph consisting of only simplex links. This transformation simplifies the modeling of wireless links in the mixed network and simplifies the scheduling algorithm.
2. A Mixed Integer Linear Program (MILP) model representing the no-wait scheduling problem for a mixed wired-wireless network is presented along with a relaxed formulation consisting of 2-stage MILPs. A greedy heuristic method is also proposed, as the MILP approach is not feasible beyond small-size problem instances.

For both the MILP model and the greedy heuristic, we assume no-wait scheduling and the aim is to minimize the flowspan time among all requests. The no-wait scheduling constraint in TSN ensures that packets are transmitted immediately after processing, avoiding queueing in the queue of the egress port [10]. This results in packets traveling through the network "non-stop" with minimal possible end-to-end delay.

The flowspan time is an essential metric in the scheduling problem, as it determines the frame queueing time at the source node and consequently, the end-to-end delay for TT streams [10].

The remainder of the article is organized as follows: Sect. 2 provides a survey of key related works. The system model along with the problem statement is presented in sect. 3. The MILP formulation for the mixed network wired-wireless scheduling problem is discussed in 4. Next, a greedy heuristic algorithm is described in sect. 6. The MILP approach and heuristic are evaluated for scalability and performance in sect. 7. The paper is concluded in sect. 8.

2 Related Works

Efforts have been made to introduce the sensitivity of time to Ethernet for several years [13, 14]. The TSN Task Group (TSN TG), a task group of IEEE 802.1 Working Group (WG), has published a suite of standards targeting extensions to IEEE 802 networks that enable end-to-end deterministic connectivity, i.e, bounded delays and jitter along with high availability. The most relevant standards in the suite are listed in Table 1.

The IEEE 802.1Qbv, also known as Time-aware Shaping (TAS), specifies the mechanisms to implement TDMA in IEEE 802 networks [4]. The use of the TAS entails the computation of routes and schedules based on the given network and

Table 1 Overview of various TSN standards

Standard	Description
IEEE 802.1AS-Rev	Timing and synchronization for time-sensitive Applications [15]
IEEE 802.1Qav	Prioritization of time-sensitive streams over best-effort using Credit Based Shaper (CBS) [16]
IEEE 802.1Qbv	Scheduling of time-sensitive traffic using time-aware shaping [4]
IEEE 802.1Qca	Path control and reservation [17]
IEEE 802.1Qci	Per-stream Filtering and Policing [18]
IEEE Std 802.1CB-2017	Frame Replication and Elimination for Reliability [19]

TT stream specifications. Earlier work on the TSN scheduling problem was mostly concerned with computing schedules for proprietary networks (e.g., TTEthernet, PROFINET), while recent work has focused on computing schedules for TAS-enabled bridged networks. A Satisfiability Modulo Theory (SMT) model was proposed by Tămaş-Selicean et al. to solve the scheduling problem in TTEthernet [13]. Schweissguth et al. in [12] have proposed an Integer Linear Program (ILP) to jointly solve the Routing and Scheduling problem for TTEthernet networks. Hanzaelk et al. have formulated the scheduling problem for Profinet in terms of Resource-Constrained Project Scheduling with Temporal Constraints and present an algorithm to solve it [20].

The No-wait Packet Scheduling Problem (NW-PSP) for TAS-enabled bridged network has been mapped to the No-wait Job-shop Scheduling Problem (NW-JSP) in [10] and is formulated as an ILP. In the ILP model for NW-PSP, the objective is to minimize the flowspan time which is equivalent to the makespan time in NW-JSP. The key constraints in NW-PSP enforce that no two frames of TT stream requests can temporally overlap on the same link. This constraint enforces the no-wait requirement for the frames of TT stream requests. Hellmanns et al. proposed and evaluated various optimizations for ILP-based TSN scheduling in [21]. However, despite these optimizations, the ILP-based approach to solving the TSN scheduling problem does not scale well concerning the network topology size and the total number of TT streams. This stems from the fact that the TSN scheduling problem, in general, is an NP-hard problem similar to the bin-packing problem [22]. Therefore, exact approaches such as ILP and SMT become infeasible for problem instances with large sizes.To address this various heuristics and metaheuristics have been proposed to provide suboptimal solutions faster than exact approaches. By compromising on the solution quality, the problem can still be solved for large problem instances. Dürr et al. also presented a tabu-search-based heuristic, as a faster alternative to the ILP approach for solving the NW-PSP problem [10]. Pahlevan et al. used a Genetic Algorithm (GA) to solve the joint routing and scheduling problem and demonstrated significant improvements in schedulability and flowspan compared to the List Scheduling (LS) approach [23]. Yang et al. proposed an improved Ant-Colony Optimization (ACO) algorithm to ensure deterministic end-to-end delay and jitter for TT traffic [24].

The aforementioned works are mainly focused on scheduling TT streams in a TSN-capable wired network despite the fact that various wireless technologies such as 5 G and WiFi have been extended to incorporate TSN features. Haxhibeqiri et al. and Cavalcanti et al. extensively discussed the prerequisites, such as time synchronization and traffic shaping, for extending TSN capabilities in wireless networks [7, 25]. They also demonstrated a WiFi-based TSN solution that can be synchronized within a few microseconds along with an end-to-end latency of 3 ms. The authors of [26] have discussed how key features of WiFi7 (IEEE802.11be) can be used to implement TSN functionality to support low-latency communication. TSN-capable wired-wireless networks are feasible, although the scheduling problem for such mixed networks has not received serious attention. Ginthör et al. solved the scheduling problem for a mixed network consisting of TSN and 5 G bridges using constraint programming while optimizing resource utilization [27]. To the best of our knowledge, the scheduling problem for mixed wired-wireless networks has not been modeled or addressed yet. Existing SMT/ILP formulations and heuristics to represent the scheduling problem in wired TSN are not readily applicable to wired-WiFi mixed networks for the reasons explained in sect. 3. Hence, this paper presents a MILP formulation and a greedy heuristic to address the problem.

3 Problem Statement

To achieve end-to-end timing guarantees for TT streams with TAS in a mixed wired-wireless network, a gating mechanism for all classes of traffic is required at every node, as shown in Fig. 1. Each egress port of the TAS-capable switch has multiple queues, each corresponding to a traffic class. A frame is assigned to a queue based on the Priority Code Point (PCP) in the VLAN tag of the frame and a configurable mapping of the PCP values to the hardware queues. The logic for queue selection for transmission at an egress port is based on the Gate Control List (GCL) for that

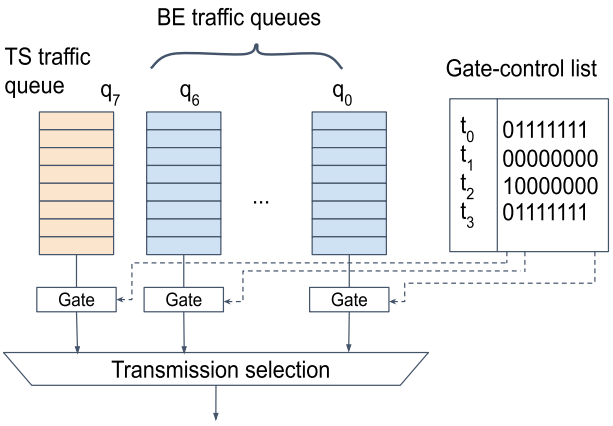

Fig. 1 Typical architecture of an egress port supporting TAS (IEEE 802.1Qbv)

egress port. A GCL contains a sequence of entries, each containing the duration of the entry and a bit-mask indicating the queues that are allowed to transmit until the start of the next entry. Strict priority queueing is used for transmission selection in case two queues have frames to transmit at the same time. The total duration of GCL entries is called the cycle time, denoted by T^{cyc}, i.e., the GCL sequence repeats itself after T^{cyc} interval.

We assume one queue is reserved at each egress port for TT traffic and its frames are assigned to this queue after selecting the egress port. We also apply the no-wait constraint to TT streams. In other words, a TT traffic frame traverses the routed path from the source to the destination without queuing at any intermediate node. With the no-wait constraint, the network offers the minimum possible delay on the routed path, as the queueing delay in each node is zero. It can be safely assumed that all nodes (switches, WAPs and endpoints) of the network are synchronized (for example, using the Precision Time Protocol (PTP)) with an accuracy μs and are capable of adhering to the schedules with which they are configured. This entails that the wireless nodes (WAP and endpoints) sharing the wireless medium can only transmit during their allocated time slots thus avoiding frame collision even within the shared medium. Given such guarantees, it is thus possible to determine the frame arrival time at any point on the routed path in a mixed wired-wireless network thus guaranteeing end-to-end delay. Next, we describe the system model and introduce the scheduling problem for mixed wired-wireless networks.

3.1 Model Description

The set of requests corresponding to the TT streams that need to be scheduled is denoted by R. For each request, $r \in R$, a tuple $(n_r^{src}, n_r^{dst}, F_r, fsize_r, T_r, D_r^{e2e})$ is used to represent the attributes of each stream request. Here, n_r^{src} and n_r^{dst} denote the requested stream's source and destination nodes, respectively. The stream request r produces a total of F_r frames, each equal to $fsize_r$ bytes every T_r seconds. A request source, e.g., camera, might produce multiple $fsize_r$ sized frames in a cycle duration, thus $F_r \geq 1$. T_r denotes the time period corresponding only to request r. The period of all requests in R need not be equal. We denote the common period (hyperperiod) of all requests as T^{cyc}, that is, the minimum time that is a period of all requests; $T^{cyc} = LCM(\{T_r : \forall r \in R\})$. This is equal to the cycle time of the GCL sequence in each node of the network. At the start of every request cycle (T_r), F_r frames are produced by n_r^{src} that need to be scheduled in next the T_r seconds. Therefore, the number of frames that need to be scheduled in cycle time T^{cyc} is $F_r^{cyc} = (T^{cyc}/T_r)F_r$. The maximum end-to-end delay allowed for request $r \in R$ is indicated by D_r^{e2e}. The end-to-end delay requirement D_r^{e2e} for each request encompasses the entire path between the source n_r^{src} and the destination n_r^{dst} nodes covering both wired and wireless links.

The topology of the mixed wired-wireless network is represented by a directed graph $G = (N, E)$. The endpoints and switching nodes (TSN switches and WiFi APs) form the set of nodes N; while E denotes the set of physical links (wired and wireless) between the nodes; $N_{sw} \in N$ and N_{AP} are the set of TSN switches and

wireless APs, respectively. The forwarding nodes are capable of implementing TAS for TT flows, i.e., they comply with IEEE 802.1Qbv. The different delay components on a link (n_i, n_j) are illustrated in Fig. 2. In our model, we consider two types of delays, transmission delay and processing delay. The contribution of propagation delay ($< 1\%$ of T_r) is ignored as it is negligible in a LAN environment (e.g., industrial network). The processing delay of the node is denoted by d_n^{proc} and the transmission delay for frame f of r' s on the link (n_i, n_j) is denoted by d_{r,n_i,n_j}^{trans} $(= fsize_r / B_{n_i,n_j})$, where B_{n_i,n_j} is the link speed in Bytes per second. It is worth mentioning that for a wireless link (n_i, n_j), B_{n_i,n_j} corresponds to the transmission speed of the Modulation and Coding Scheme (MCS) with maximum reliability. For instance, in WiFi 6 (IEEE 802.11ax) $B = 8.6$ Mbps for the lowest MCS index. Therefore, d_{r,n_i,n_j}^{trans} is equal to the maximum duration for which the gate for the TT queue on node n_i (on output port corresponding to (n_i, n_j)) should be opened to let the frame to be transmitted.

$\forall r \in R$, a route must be determined from n_r^{src} to n_r^{dst}. We assume that a maximum of K shortest paths $(P_{r,K})$ between n_r^{src} and n_r^{dst} are predetermined using off-the-shelf routing algorithms and passed as an input to the model to compute transmission schedules on one of these routes. The K shortest path algorithm is proposed by Yen et al. in [28]. This pre-processing vastly reduces the number of variables in the MILP model and thus reduces the execution time significantly [21].

For a request r, p denotes a path among $P_{r,K}$ in the network consisting of an ordered list of nodes. The ordered list of edges on the path p is denoted by ϵ_p. D_{p,n_i} is the delay along the path p from the source node (n_p^{first}) to node n_i; this includes the transmission delay on the last link (n_i). D_p is the delay along path p until the last node (n_p^{last}) of p. It is worth noting that two different requests can have different D_{p,n_i} (and D_p) for the same list of edges as their transmission delays can vary depending on their frame size.

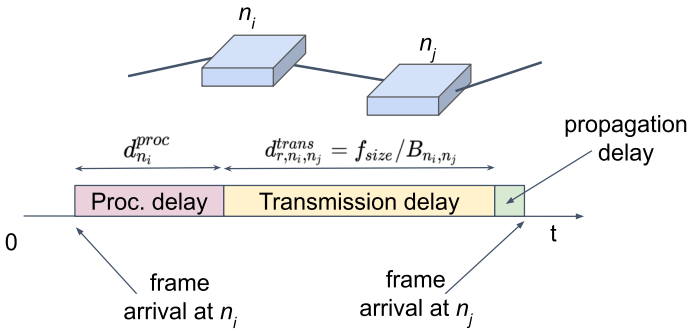

Fig. 2 An illustration for different delay components on a given link

3.2 Wireless Link Modeling

Calculating end-to-end TT flows' schedules in mixed wired-wireless networks is generally more complex than in wired TSN networks. In our model, we assumed that the frames of TT flows are successfully transmitted in their first attempt. It is reasonable to assume this is the case in a TAS-enabled private network in a closed environment with all wireless transmission sources known beforehand, e.g., an industrial network connecting devices on a factory shop floor. As all nodes (WAP and endpoints) sharing a wireless link are TAS-enabled, they can only transmit during their allocated time slots thus avoiding interference from other nodes. Furthermore, the impact of external interference sources is minimized by choosing the MCS scheme with the highest reliability. However, this model could be further extended for the case where MCS schemes with lower reliabilities (and higher transmission speeds) are allowed thus resulting in the possibility of multiple re-transmissions in order to achieve a given reliability threshold. As explained next, an additional challenge is to model the given mixed wired-wireless topology such that a given scheduling algorithm can take it as input.

The half-duplex wireless links are shared by multiple endpoints and the WAP whereas a wired link is always point-to-point and full-duplex. Therefore, computing the transmission schedule on the wireless link becomes complicated. To simplify the scheduling problem, the original wireless link, as shown in Fig. 3a, can be modeled as a graph (Fig. 3b) consisting of the WAP, additional dummy nodes

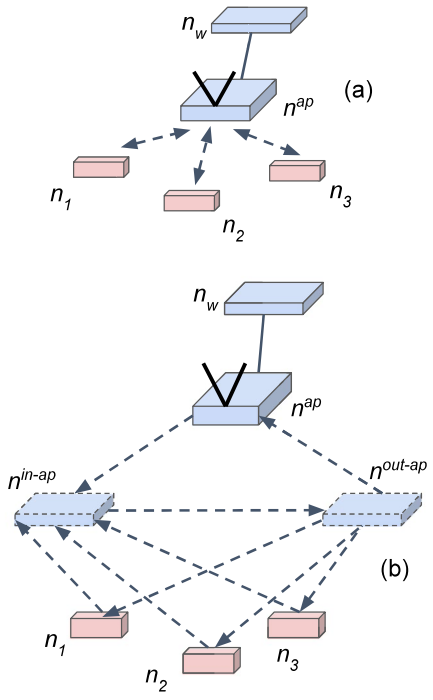

Fig. 3 **a** Half-duplex wireless link modeled as **b** a graph consisting of dummy nodes and simplex links. The half-duplex property of the wireless link is retained by ensuring all communication traverses the dummy link (n^{in-ap}, n^{out-ap})

and dummy links. n^{ap} is connected to two dummy nodes– n^{in-ap} and n^{out-ap} via simplex links and the two dummy nodes are connected to each via another simplex link (n^{in-ap}, n^{out-ap}). The dummy nodes n^{in-ap} and n^{out-ap} have 0 processing delay, whereas the processing delay of n^{ap} is the same as that of the original AP. The links (n_i, n^{in-ap}), (n^{out-ap}, n_i), (n^{ap}, n^{in-ap}) and (n^{out-ap}, n^{ap}) all have 0 transmission delay (or ∞ bandwidth). The link (n^{in-ap}, n^{out-ap}) represents the medium shared between the WAP and its connected endpoints and having a bandwidth equal to the (minimum) wireless bandwidth. It can be seen that the half-duplex property of the wireless link is maintained in Fig. 3b, as any communication path between the WAP and an endpoint or between endpoints themselves includes the link (n^{in-ap}, n^{out-ap}). Therefore, the no-overlap constraint ensures that only one node is transmitting at a particular time. As the transmission delay between the WAP and a wireless node n_i is included in the transmission delay on the link (n^{in-ap}, n^{out-ap}), the delays on other dummy links are set to zero.

It is important to highlight that the adaptation and scheduling of wireless links for Quality of Service (QoS) purposes may lead to varying delays between an endpoint and the WAP [29, 30]. In our system model, we conceptualize these variations as represented within the processing and transmission delays of the wireless link. The internal mechanisms of wireless networks for adaptation and scheduling could potentially be optimized together with the scheduling of flows in the wired segments. Nevertheless, such joint optimization falls beyond the scope of this paper. Using the above transformation, each half-duplex wireless link can be broken into a subgraph consisting only of simplex links. Therefore, the given mixed wired-wireless network denoted by $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ can be transformed into a network, denoted by $G = (N, E)$, consisting only of simplex or full-duplex links. Table 2 lists the notations used for various parameters and sets along with their short description.

3.3 Problem Definition

In IEEE 802.1Qcc, the centralized configuration model for TSN is described, where a central entity, known as the centralized network controller (CNC), has a global view of the complete network and manages all nodes; this includes wireless APs and wireless endpoints. The architectural model for a centrally controlled mixed wired-wireless TSN is depicted in Fig. 4.

The CNC is mainly responsible for (i) discovering the network topology that includes capturing the capabilities of various nodes and (ii) configuring nodes with appropriate GCLs. GCLs are computed by the scheduler, a component of the CNC, based on the specifications of TT traffic (R) and the given network topology G (with delay parameters). As topology discovery for mixed wired-wireless networks is not the subject of this article, we will focus only on the scheduling aspect. We assume that the scheduler has a global view of the network along with the delay parameters and the request set.

The end-to-end scheduling problem can be defined as the problem of computing the end-to-end optimized schedules for all TT flow requests given the topology graph of a mixed wired-wireless network and requests' specifications. The proposed

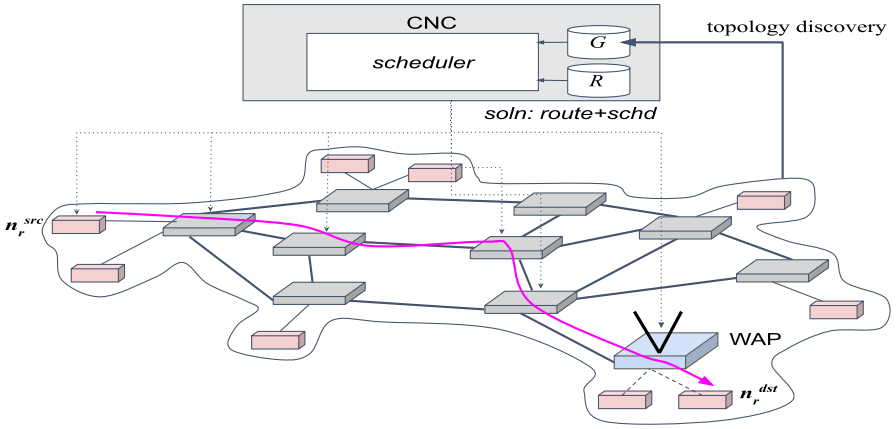

Fig. 4 Overview of the end-to-end scheduling problem in a mixed wired-wireless network. The red, blue and grey boxes represent the endpoints, WAP and switching nodes in the network, respectively (Color figure online)

approaches for solving the scheduling problem in wired networks (mentioned in sect. 2) do not apply to the scheduling problem in mixed wired wireless networks. It is important to note that the approaches suggested for solving the scheduling problem in wired networks are not applicable, even after converting semi-duplex links to simplex links. The reason for this is these approaches assume uniform transmission speeds across all links [10]. The relevant constraints need to be generalized to account for the variability in mixed-wireless networks. In our model, we avoid making this assumption, allowing us to present a more comprehensive approach to solving the scheduling problem in mixed wired-wireless networks.

The optimization criterion considered in our model is the flowspan time. The flowspan time is defined as the maximum injection time among all TT streams with respect to the start of the request cycle (T_r). Figure 5 shows the frame injection times (length of arrows) of a request r; where $F_r = 2$ and $T^{cyc} = 4T_r$ thus $F_r^{cyc} = 8$. Here, the flowspan of the request corresponds to the relative injection time of the frame 7 since it has the longest relative injection time.

By choosing flowspan as our optimization, we make sure the injection time of TT stream frames is minimized at the source endpoints such that these frames are not queued at the source node for too long, thereby reducing the end-to-end delay. Furthermore, by reducing the flowspan time the schedules for TT stream requests are compact, i.e., time slots for TT stream requests are aligned one after another thereby avoiding the requirement to have minimum guard interval between consecutive time slots as described in [10]. Typically, a fixed time interval called the guard interval is required between a BE time slot and a TT stream slot. During this interval, no transmissions are allowed in order to allow the completion of transmission of BE frames and thereby preventing interference from BE frames that are scheduled just before the start of the time slot of a TT stream request.

With the no-wait constraint, the scheduling problem boils down to the calculation of the solution consisting of: (i) the routed path and (ii) the injection times for all TT

Table 2 Description of the notations used for the parameters and variables involved in the system model

Notation	Description
R	The set of requests that need to be routed and scheduled on a given Time-Sensitive Network (TSN). Each request $r \in R$ is associated with a tuple $\mathcal{T} = (n_r^{src}, n_r^{dst}, F_r, T_r, D_r^{e2e})$, where n_r^{src}, n_r^{dst} and T_r are request's source node, destination node, total frames and time period, respectively. The set of all frames that needs to be transmitted by request r during request period T_r is denoted by $\mathcal{F}_r = \{f : 0 \leq f \leq F_r - 1\}$. D_r^{e2e} is the maximum end-to-end permissible delay for the request
T^{cyc}	The lowest time interval that is a period of all the requests, i.e., $T^{cyc} = LCM(\{T_r : \forall r \in R\})$. D_r^{e2e} is the maximum allowable end-to-end delay for all frames of request $r \in R$; this is relative to the start of the last common cycle (T^{cyc})
F_r^{cyc}	The total number frames of request $r \in R$ during T^{cyc}, i.e., $F_r^{cyc} = (T^{cyc}/T^r)F$. The set of frames during T^{cyc} is denoted by $\mathcal{F}_r^{cyc} = \{f : 0 \leq f \leq F_r^{cyc} - 1\}$
$G = (N, E)$	Directed graph representation of the network, where N is the set of nodes (switches, APs and endpoints) and E is the set of physical links between the nodes. $N_{sw} \in N$ and N_{AP} are the set of TSN switches and WAPs, respectively
d_{n_i,n_j}^{prop}	The propagation delay on physical link $(n_i, n_j) \in E$
d_{n_i,n_j}^{trans}	The transmission delay on physical link $(n_i, n_j) \in E$
d_n^{proc}	The frame processing time of node $n \in N$
$P_{r,K}$	K shortest paths between nodes n_r^{src} and n_r^{dst} for request $r \in R$; $p \in P_{r,K}$ is an ordered list of nodes from n_r^{src} to n_r^{dst}
ϵ_p	Ordered list of edges in path p
D_{p,n_i}	The delay along path p starting from the first node n_p^{first} up to node $n_i \in p$
D_p	The delay along path p starting from the first node (n_p^{first}) until the last node (n_p^{last})
$\gamma_{r,p}$	The decision variable indicates if $r \in R$ is mapped to a physical path p
$\rho_{r_1,p_1,f_1,r_2,p_2,f_2}^{n_i,n_j}$	The decision variable indicates that frame $f_1 \in \mathcal{F}_{r_1}^{cyc}$ of request $r_1 \in R$'s frame is scheduled before frame $f_2 \in \mathcal{F}_{r_2}^{cyc}$ of request $r_2 \in R(\neq r_1)$'s on the common link (n_i, n_j) of the routed paths p_1 and p_2 of r_1 and r_2, respectively
$t_{r,f}^{in}$	The continuous decision variable is the injection time of frame $f \in \mathcal{F}_r^{cyc}$ of request $r \in R$ at the source node; $t_{r,f}^{in}$ is the time offset relative to the start of the last common cycle (T^{cyc})
FS	The continuous decision variable denotes the flowspan time among all the requests

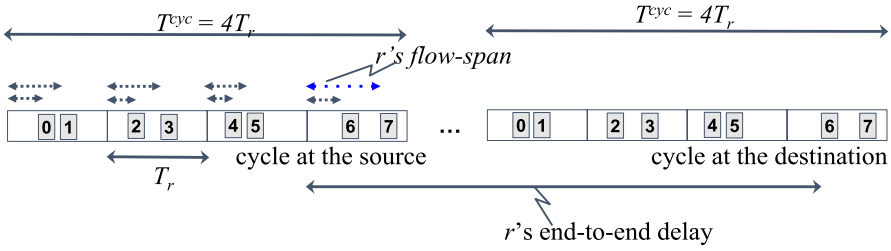

Fig. 5 Illustration for the flowspan and end-to-end delay of request r. The length of each dotted arrow indicates its injection time relative to the start of the request's cycle. The flowspan of this request corresponds to the relative injection time (in blue) of frame 7. The end-to-end delay for frame 7 illustrated is the sum of flowspan time and the path delay

stream requests. Using this solution and the delay along the routed path, the opening and closing times of the gate are determined at each intermediate node, along with the guard intervals, to prevent BE traffic from interfering with scheduled TT traffic [10]. The gate opening and closing times are then used for the generation of the GCL for network nodes. The CNC configures each node in the network with the GCL so that all TT stream requests are supported.

4 MILP Formulation

In this section, we present the MILP formulation for the end-to-end scheduling problem for mixed wired-wireless networks. Here, we describe the MILP objective function along with various constraints that ensure that end-to-end guarantees are given to TT streams. Table 2 lists the notations used for all variables involved in the MILP formulation, together with their short description.

4.1 Constraints

Each request should be assigned exactly to one physical path in G. The constraint in (1) ensures that only one path is selected for routing request $r \in R$ out of K shortest paths between n_r^{src} and n_r^{dst}.

$$\sum_{p \in P_{r,K}} \gamma_{r,p} = 1, \quad \forall r \in R. \tag{1}$$

The no-wait scheduling disallows frames to be queued on any node in the routed path. Therefore, frames from two different requests that are routed on a common physical link cannot be transmitted such that they overlap in time. To avoid the overlap of two frames: $f_1 \in \mathcal{F}_{r_1}^{cyc}$ and $f_2 \in \mathcal{F}_{r_2}^{cyc}$ on the physical link (n_i, n_j), either the transmission of f_1 should end before the start of the transmission of f_2' s or the transmission of f_2 should end before the start of the transmission of f_1's.

The pair of constraints in (2 and 3) ensures this condition.

$$\begin{aligned} & t_{r_1,f_1}^{in} - t_{r_2,f_2}^{in} + M_1(\rho_{r_1,p_1,f_1,r_2,p_2,f_2}^{n_i,n_j}) + M_2(2 - \gamma_{r_1,p_1} - \gamma_{r_2,p_2}) \\ & \quad \geq D_{p_2,n_i} - D_{p_1,n_i-1} - d_{n_i}^{proc}, \\ & \quad \forall r_1, r_2 \in R(r_1 \neq r_2), \forall f_1 \in \mathcal{F}_{r_1}^{cyc}, \forall f_2 \in \mathcal{F}_{r_2}^{cyc}, \\ & \quad \forall p_1 \in P_{r_1,K}, p_2 \in P_{r_2,K}, \forall (n_i, n_j) \in \epsilon_{p_1} \cap \epsilon_{p_2}. \end{aligned} \tag{2}$$

$$\begin{aligned} & t_{r_2,f_2}^{in} - t_{r_1,f_1}^{in} + M_1(1 - \rho_{r_1,p_1,f_1,r_2,p_2,f_2}^{n_i,n_j}) \\ & \quad + M_2(2 - \gamma_{r_1,p_1} - \gamma_{r_2,p_2}) \geq D_{p_1,n_i} - D_{p_2,n_i-1} - d_{n_i}^{proc}, \\ & \quad \forall r_1, r_2 \in R(r_1 \neq r_2), \forall f_1 \in \mathcal{F}_{r_1}^{cyc}, \forall f_2 \in \mathcal{F}_{r_2}^{cyc}, \\ & \quad \forall p_1 \in P_{r_1,K}, p_2 \in P_{r_2,K}, \forall (n_i, n_j) \in \epsilon_{p_1} \cap \epsilon_{p_2}. \end{aligned} \tag{3}$$

Here, $D(p_1, n_i)$ is the delay up to node n_i on path p_1 starting from the source node (n_r^{src}). It is worth noting that $D(p_1, n_i)$ includes delay contributions of all links (wired and wireless) between n_r^{src} and n_i on path p_1. M_1 and M_2 are arbitrary constants greater than the maximum value of the right-hand side in (2) and (3).

The indicator variable $\rho^{n_i,n_j}_{r_1,p_1,f_1,r_2,p_2,f_2}$ indicates the order of transmission of frames f_1 and f_2 on (n_i, n_j), when f_1 and f_2 are routed on same physical link (n_i, n_j). If paths p_1 and p_2 are selected for routing requests r_1 and r_2, respectively, then $\gamma_{r_1,p_1} = \gamma_{r_2,p_2} = 1$. Moreover, if $\rho^{n_i,n_j}_{r_1,p_1,f_1,r_2,p_2,f_2} = 0$, the constraint in (3) becomes inactive, whereas the constraint in (2) is simplified to $t^{in}_{r_1,f_1} + D_{p_1,n_i-1} + d^{proc}_{n_i} \geq t^{in}_{r_2,f_2} + D_{p_2,n_i}$. Thus f_2's transmission on link (n_i, n_j) precedes f_1's transmission. Conversely, if $\rho^{n_i,n_j}_{r_1,p_1,f_1,r_2,p_2,f_2} = 1$, the constraint in (2) becomes inactive, whereas the constraint in (3) is simplified to $t^{in}_{r_2,f_2} + D_{p_2,n_i-1} + d^{proc}_{n_i} \geq t^{in}_{r_1,f_1} + D_{p_1,n_i}$ thus f_1's transmission on link (n_i, n_j) precedes f_2's transmission. The first constraint in the pair is depicted in Fig. 6.

The transmission of two frames $f_1, f_2 \in \mathcal{F}^{cyc}_r, f_1 \neq f_2$ of the same request $r \in R$ should not overlap each other in the time domain. This condition is enforced by the constraint in (4); $\sum_{p \in P_{r,K}} \max_{(n_i,n_j)\in\epsilon_p}$ ensures that temporal overlap does not happen on any physical link of the routed path.

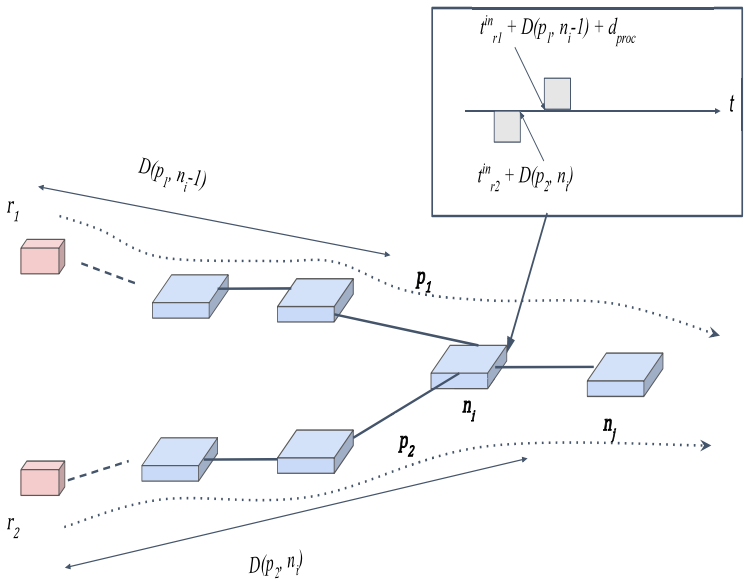

Fig. 6 Overview of the frame overlap constraint. Here $\rho = 0$, thus f_2 of r_2 is transmitted before f_1 of r_1 on common link (n_i, n_j)

$$t^{in}_{r,f_1} + \sum_{p \in P_{r,K}} \max_{(n_i,n_j) \in \epsilon_p} \gamma_{r,p} d^{trans}_{r,n_i,n_j} \leq t_{r,f_2}, \quad \forall r \in R, f_1 \in \mathcal{F}^{cyc}_r, f_2 = f_1 + 1. \tag{4}$$

As $r \in R$ can have different time periods, we schedule all requests for a common time period or cycle time T^{cyc}. However, for all requests, F_r frames should be scheduled during *their* own period T_r, that is, the injection time of f should occur in the $\lfloor f/F_r \rfloor$'th time period. The constraint in (5) ensures this.

$$\lfloor f/F_r \rfloor T_r \leq t^{in}_{r,f} \leq (\lfloor f/F_r \rfloor + 1)T_r - \sum_{p \in P_{r,K}} \max_{(n_i,n_j) \in p} \gamma_{r,p} d^{trans}_{r,n_i,n_j}, \quad \forall r \in R, f \in \mathcal{F}^{cyc}_r. \tag{5}$$

The summation in the right-hand side of (5) ensures that the frame transmission finishes before the end of the period.

It is worth noting that the constraints in (4) and (5) are not linear because of the *max* function. However, these constraints are linearized by an ILP solver (e.g., CPLEX) by using auxiliary variables [31].

The end-to-end delay of a request cannot exceed the corresponding application requirement D^{e2e}_r. The constraint below (6) ensures that the end-to-end delays are bounded by the given limit. The first term in the left-hand side of (6) represents the flowspan of frame f of request r while the summation represents the delay contributions of both wired and wireless links on path p.

$$(t^{in}_{r,f} - \lfloor f/F_r \rfloor T_r) + \sum_{p \in P_{r,K}} \gamma_{r,p} D_p \leq D^{e2e}_r, \ \forall r \in R, \forall f \in \mathcal{F}^{cyc}_r. \tag{6}$$

For a given frame $f \in F^{cyc}_r$ of request $r \in R$, the flowspan is calculated with respect to the start of the period of f, i.e., $\lfloor f/F_r \rfloor T_r$).

$$(t^{in}_{r,f} - \lfloor f/F_r \rfloor T_r)/T_r \leq FS, \quad \forall r \in R, \forall f \in \mathcal{F}^{cyc}_r. \tag{7}$$

The constraint in (8) ensures that the binary decision variables can only take values in $\{0, 1\}$ and injection time ($t^{in}_{r,f}$) is non-negative.

$$\gamma_{r,p}, \rho^{n_i,n_j}_{r_1,p_1,f_1,r_2,p_2,f_2} \in \{0,1\}, t^{in}_{r,f} \geq 0, FS \geq 0; \quad \forall r, r_1, r_2 (\neq r_1) \in R, \forall p, p_1, p_2 \in P_{r,K}, \quad \forall f, f_1, f_2 (\neq f_1) \in \mathcal{F}^{cyc}_r \tag{8}$$

4.2 Objective Function

The objective function of the MILP minimizes the flowspan among all frames of all requests as shown in (9).

$$obj : \min FS \tag{9}$$

By minimizing flowspan time, compact packet schedules are achieved, where gate openings for time-triggered flows are concentrated at the start of the cycle instead of being spread throughout the entire cycle. As identified by [10], this maximizes the bandwidth available for best-effort traffic.

5 2-stage MILP

As mentioned above, the MILP model for the scheduling problem is an NP-hard problem. The 2-stage procedure shown below relaxes the original MILP model by decomposing the original MILP problem into two MILP subproblems as proposed in [32]. The first MILP subproblem for the scheduling problem is as follows:

$$\begin{aligned} obj &: \min FS \\ s.t. \quad & (1); \\ & \sum_{p \in P_{r,K}} \max_{(n_i,n_j) \in p} \gamma_{r,p} d^{trans}_{r,n_i,n_j} \leq FS, \\ & \forall r \in R, \forall f \in \mathcal{F}^{cyc}_r, FS \geq 0, \quad \gamma \in \{0,1\}. \end{aligned} \tag{10}$$

The solution to the first MILP subproblem determines the routing path for each request. The constraint in (10) ensures that for any link (n_i, n_j) on path p, the maximum transmission delay is less than or equal to the overall flowspan. From the solution of the above MILP, the paths for the given TT stream requests are determined, i.e., the values for $\gamma^0_{r,p}$ are obtained from the solution for $\forall r \in R$. These obtained paths for the stream requests are substituted in the formulation of the second MILP as shown below:

$$\begin{aligned} obj &: \min FS \\ s.t. \quad & \gamma_{r,p} = \gamma^0_{r,p}; \\ & (2) - (7); \\ & \rho^{n_i,n_j}_{r_1,p_1,f_1,r_2,p_2,f_2} \in \{0,1\}, t^{in}_{r,f} \geq 0; \\ & \forall r, r_1, r_2 (\neq r_1) \in R, \forall p, p_1, p_2 \in P_{r,K}, \\ & \forall f, f_1, f_2 (\neq f_1) \in \mathcal{F}^{cyc}_r, FS \geq 0. \end{aligned} \tag{11}$$

Although the 2-stage model is a more efficient method to solve the scheduling problem, it is worth noting that both the original MILP and the 2-stage MILP approaches are still combinatorial optimization problems and thus have exponential worst-case time complexity in terms of their integer variables similar to NW-PSP.

6 Heuristics

In this section, we describe the greedy heuristic, which is a scalable alternative to the two MILP approaches to solving the mixed wired-wireless network scheduling problem. The greedy heuristic iteratively considers each TT stream request and attempts to schedule it on the given network. While scheduling a frame of a given request, the smallest injection time is chosen, for which there is no interference (i.e., the no-wait constraint is satisfied) with respect to the frames of the already scheduled requests. The pseudocode of the proposed greedy algorithm is discussed in detail next. The procedure in Alg. 1 is the main procedure for the greedy heuristic. *Paths* and *Times*, are first initialized with empty dictionaries; they are used by the heuristic to store the solution path and injection times of the scheduled requests. Next, the given set of requests is simply ordered based on the *crit* criterion and assigned to R_{ord}. For example, a given set of requests can be sorted in ascending order of the request period ($crit = T_r$). After ordering the requests, the request with the smallest T_r is scheduled first and the one with the longest T_r is scheduled last. Next, scheduling of each request in R_{ord} is attempted through `SchReq`, which is explained in Alg. 2. If request r is scheduled ($p \neq \phi$), the solution path and the frame injection times are stored in dictionaries *Paths* and *Times*, respectively, both keyed by request r.

Algorithm 1 The pseudo-code for the main procedure responsible for scheduling a given set of TT stream requests.

```
Procedure SchReqsGrd(R, crit):
    Paths, Times ← {}, {} ; // Initialize solution paths and
     injectimes as empty collections
    R_ord ← ReqsOrder(R, crit) ; // sort requests in R with crit
     criterion
    for r ∈ R_ord do
        p, t_r^in ← SchReq(r, Paths, Times) ; // get path and injection
         times for the request
        if p ≠ ϕ then
            Paths[r] ← p ; // store routed path
            Times[r] ← t_r^in ; // store injection time of all frames
        end
        else
            return ϕ, ϕ ; // terminate the algorithm if a schedule is
             not found for a request
        end
    end
    return Paths, Times ; // return greedy solution
end
```

In Alg. 2, the frame injection times of the considered request are first initialized with $\lceil f/F_r \rceil T_r$ (the minimum possible injection time). The pair of `for` loops iteratively goes through the paths in $\forall p \in P_{r,K}$ and terminates, if $\forall f \in \mathcal{F}_r^{cyc}$ can be scheduled on it and the injection time meets the end-to-end delay requirements. The procedure `SchFrame`, explained in Alg. 3, is called to determine the injection time of frame f on the path p. If a path can schedule all frames, the path (p) and the corresponding injection times (t_r^{in}) are returned; otherwise, ϕ and ϕ are returned.

Algorithm 2 The pseudo-code for the procedure responsible for determining the routing path and frame injection times for a request.

Procedure SchReq(r, $Paths$, $Times$)**:**
  $t_r^{in} \leftarrow \{f : \lfloor f/F_r \rfloor T_r | \forall f \in \mathcal{F}_r^{cyc}\}$; // Initilize the injection times to the start of the cycle
  for $p \in P_{r,K}$ **do**
    for $f \in \mathcal{F}_r^{cyc}$ **do**
      $t^{in} \leftarrow$ SchFrame(r, p, $Paths$, $Times$, $t_r^{in}[f]$) ; // get f's injection
      if $t^{in} \neq \phi$ **and** $t^{in} + D_p \leq D_r^{e2e}$ **then** $t_r^{in}[f] \leftarrow t_{in}$;
      else break;
      ; // break if a feasible injection time is found
    end
    if $t^{in} \neq \phi$ **then return** (p, t_r^{in});
  end
  return ϕ, ϕ ; // no feasible schedule found
end

To satisfy the no-wait constraint, no two frames should overlap in time at any of the common edges of their routed paths. To this end, we identify the time windows at the source of r that are forbidden, i.e., the injection times that cause time overlap with other scheduled requests. For instance in Fig. 6, assume that request r_1 is routed on path p_1 and scheduled with the injection time $t_{r_1}^{in}$. If p_2 is selected for request r_2, the forbidden injection time interval for r_2 starts from $t_{r_1}^{in} + D(p_1, n_i - 1) + d_{proc}$ with the length equal to the d_{r,n_i,n_j}^{trans}. The procedure SchFrame (in Alg. 3) is responsible for computing the injection time ($\geq t^{in}$) of frame f on the path p, given the injection times of the frame (*Times*) of other scheduled requests. t^{frb}, the list of forbidden injection time windows, is first initialized with an empty list [ ]. For each frame (f^o) of the already scheduled request (r^o), the arrival time (t_{r^o,f^o}^{arr}) on an overlapping edge ($(n_i, n_j) \in$ OvrlpEdges(p, p^o)) is calculated. Using t_{r^o,f^o}^{arr}, forbidden time window(s) are computed by subtracting the path delay on p until the overlapping edge and added to list t^{frb}. If the forbidden window continues beyond T_{cyc}, two windows per cycle are added to t^{frb}. A feasible injection time ($\geq t^{in}$, $< t^{in} + T_r - d^{trans}$) that is not forbidden in t^{frb} is returned by tinFrmTwnds.

Table 3 Default values/range of various parameters involved in the evaluation

Parameter	Value or range	Units
Topology	RING, MESH, ORION	–
Request time period (T_r)	{2048, 4096, 8192}	μs
Request frame size (*fsize*)	{50, 100, 150}	B
Request frames per period (F_r)	1	–
Request e2e delay (D_r^{e2e})	$2T_r$	μs
Wired transmission speed (B^{wrd})	1000	Mbps
Wireless transmission speed (B^{wl})	10	Mbps
Node processing time (d_n^{proc})	50	μs

Algorithm 3 The pseudo-code for the procedure responsible for determining an injection time for a frame.

Procedure SchFrame(r, p, $Paths$, $Times$, t^{in}):
  $t^{frb} \leftarrow [\,]$; // sorted list of forbidden time windows at the source
  for $r^o \in Paths$ **do**
    $p^o \leftarrow Paths[r^o]$; // get the routed path for the scheduled request
    for $(n_i, n_j) \in$ OvrlpEdges(p, p^o) **do**
      $d^{trans}_{r^o,n_i,n_j} \leftarrow (r^o.fsize)/B_{n_i,n_j}$; // transmission delay for the scheduled request
      for $f^o \in [0, |F^{cyc}_{r^o}| - 1]$ **do**
        $t^{arr}_{r^o,f^o} \leftarrow Times[r^o, f^o] + D_{r^o,p^o,n_i-1} + d^{proc}_{n_i}$; // arrival time of f_o at node n_i
        $t^{src} \equiv t^{arr}_{r^o,f^o} - D_{r,p,n_i-1} - d^{proc}_{n_i}(\mod T^{cyc})$; //
        if $t^{src} + d^{trans}_{r^o,n_i,n_j} \leq T^{cyc}$ **then**
          $t^{frb}.append((t^{src}, t^{src} + d^{trans}_{r^o,n_i,n_j}))$; // forbidden time window is inside the cycle
        end
        else
          $t^{frb}.append((t^{src}, T^{cyc}))$;
          $t^{frb}.append((0, d^{trans}_{r^o,n_i,n_j} - (T^{cyc} - t^{src})))$; // forbidden time window ovelaps with the edge of the cycle
        end
      end
    end
  end
  return tinFrmTwnds(t^{in}, $t^{in} + T_r$, t^{frb}, D_r^{e2e}) ; // return smallest feasible injection time between t^{in} and $t^{in} + T_r - d^{trans}$
end

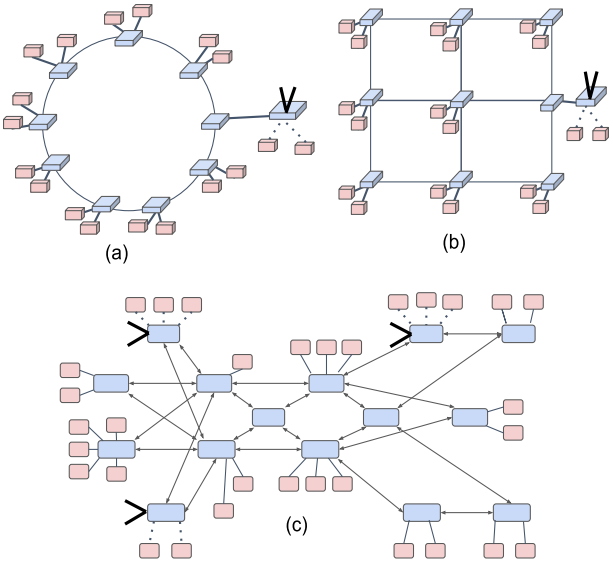

Fig. 7 Topologies used for the evaluation: **a** `RING`, **b** `MESH` and **c** `ORION` [36]

It can be observed from Alg. 3 and 2 that the time complexity of `SchFrame` and `SchReq` is $O(|R||F|)$ and $O(|R|K|F|^2)$, respectively. Therefore, the complete heuristic in Alg. 1 has polynomial time complexity in $|R|$, $|F|$ and K, i.e, $\mathcal{O}(|R|^2|K||F|^2)$.

7 Evaluations

Proposed approaches, MILP, 2-stage MILP and greedy, return a feasible solution to the mixed network wired-wireless scheduling problem ensuring that all TT stream requests meet their timing requirement. However, an extensive evaluation of these approaches is needed to compare their performance and determine the scenarios in which one approach is more advantageous than the other.

In this section, we discuss first the evaluation setup and then present the results of our evaluations for the proposed approaches in terms of scalability, scheduling performance and resource utilization.

7.1 Evaluation Setup

The greedy heuristic was written in Python; whereas the MILP formulations are done in DOcplex, a native Python modeling library for mathematical modeling, and solved using IBM's ILOG CPLEX solver [33]. For both approaches, NetworkX's Python API is used for the (pre)computation of $K = 5$ shortest paths [34]. We conducted the simulations on a PC with an Intel Core i5-8265U processor running @ 2.40GHz with 16GB of RAM running the Ubuntu−18.04 operating system.

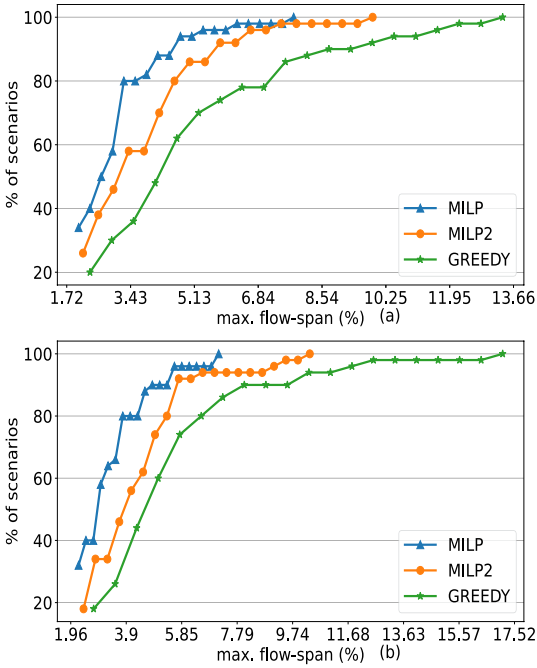

Fig. 8 The percentage of the total number of scenarios having flowspan less than a given value (x-axis) for MILP, MILP2 and GREEDY with $|R| = 20$ on **a** RING and **b** MESH topologies

Three types of topologies were used for the evaluation: (i) the ring topology (RING), (ii) the mesh topology (MESH) and (iii) the Orion Crew Exploration Vehicle (CEV) topology as shown in Fig. 7. Both RING and MESH consists of nine switch nodes, one WAP and twenty endpoint hosts. While the number of nodes in the ring and the mesh is the same, the difference in the average degree of the topologies impacts the performance of the scheduling approach. For the evaluation of the greedy heuristic on a realistic network topology we have chosen a network based on the Orion CEV network. The network consists of switches, 3 APs and 31 end stations. This has been previously considered in related studies [35, 36].

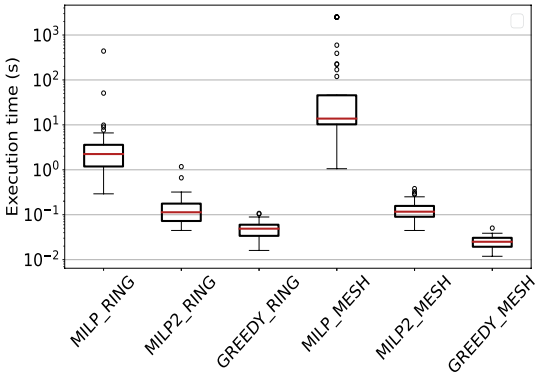

Fig. 9 Box plot of the execution time (in log scale) for MILP, MILP2 and GREEDY with $|R| = 20$ on RING and MESH

To generate a request (r) in the set of TT streams (R), we choose two randomly selected (distinct) nodes in the given topology as the endpoints (n_r^{src} and n_r^{dst}). For both topologies, we have assumed a maximum of five (K) shortest paths between the request endpoints. The request's period (T_r) can be chosen randomly from set $\{2048, 4096, 8192\}(\mu s)$; thus the overall cycle time (T^{cyc}) is $8192\mu s$. The number of frames per period (F_r) is one and the frame size is randomly chosen from the set $\{50, 100, 150\}$B. Since T_r is randomly chosen, the total number of frames per cycle (T_{cyc}) is generally greater than or equal to one, producing a similar effect as selecting a different number of frames per cycle (F_r) for each request. The end-to-end delay requirement for the request is chosen as $2Tr$ for all requests. The values/ranges of relevant evaluation parameters are listed in Table 3.

In what follows, the labels– `MILP`, `MILP2` and `GREEDY` refer to the original MILP, the 2-stage MILP approach and the greedy approach, respectively. Unless stated otherwise, the default criteria for the ordering is by default in ascending order based on their period of request.

7.2 Results

Different types of evaluations were carried out to evaluate the proposed approaches. We describe each approach and present the obtained results separately. For each evaluation result, a hundred scenarios were generated and the observations were recorded. In each scenario, a total of $|R|$ TT stream requests were generated between $|R|$ pairs of randomly selected endpoints using the parameters listed in Table 3.

7.2.1 `MILP`, `MILP2` and `GREEDY` Comparison

First, we observe the flowspan (time) percentage among all requests. Here, we define a request's flowspan time as the maximum delay in the injection time of the frame relative to the start of the cycle as a percentage of the request's cycle duration (T_r). We focus on the flowspan time as indicates the schedulability, i.e., the likelihood of scheduling more TT stream requests on the network. A higher value of flowspan time indicates that links are over-utilized thus reducing the schedulability of the

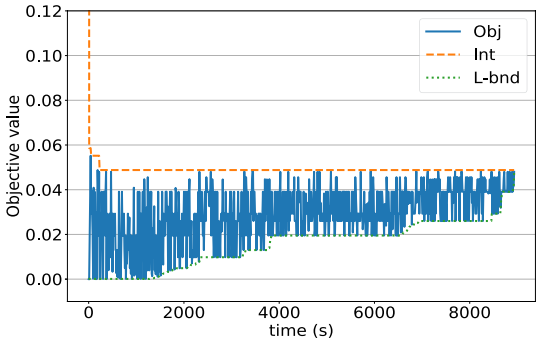

Fig. 10 Plot showing the evolution of (i) the current objective value (Obj), (ii) the best integer solution value (Int) and (iii) integer lower bound (L-bnd) with time with $|R| = 30$ on `MESH`

Table 4 Overview of various greedy criteria

Criterion	Description
RAND	Random ordering of requests
ENDPOINT_BW	Ordering of requests in the ascending order of endpoint transmission speed
BW	Ordering of requests in the descending order of requests normalized bandwidth ($fsize_r/T_r$)
PERIOD_FSIZE	Ordering of requests in the ascending order of request period (T_r; the ties are broken based on the request frame-size ($fsize_r$)

network. On the other hand, a lower value of flowspan implies that the network has the resources to schedule more requests. In Fig. 8a and b, the y-axis is the percentage of total scenarios with the flowspan time less than or equal to the corresponding x-axis value with $|R| = 20$ on RING and MESH, respectively. The flowspan time of MILP and MILP2 are very close to each other. Further, it can be observed that the flowspan time gap (absolute difference) between MILP and GREEDY is less than 5% at most of the ordinate values. This results in GREEDY having marginally smaller schedulability and higher end-to-end delays for TT stream requests compared to MILP. As discussed next, this comes at the cost of an extremely long execution time of MILP-based approaches. This cost is insignificant for realistic problem instances where MILP is impractical.

Next, we compare the execution time of the proposed approaches for the same problem instances. Here, the execution time is defined as the time required by an algorithm to find a solution. Execution time (in logarithmic scale) for MILP, MILP2 and GREEDY when scheduling $|R| = 20$ requests on RING and MESH are shown in Fig. 9. The execution time is highest for MILP_MESH as compared to MILP_RING because $|P_{r,K}| \leq 2, \forall r \in R$ in RING whereas $|P_{r,K}| \geq 2$ for many requests in MESH. The greater number of paths between two endpoints in MESH compared to RING results in a huge solution space due to the explosion in the number of decision variables (γ and ρ). The execution times for MILP2_RING are similar to that of MILP2_MESH. In contrast to MILP and MILP2, the execution time of the greedy heuristic for MESH is lower than for RING. The greedy heuristic returns the first

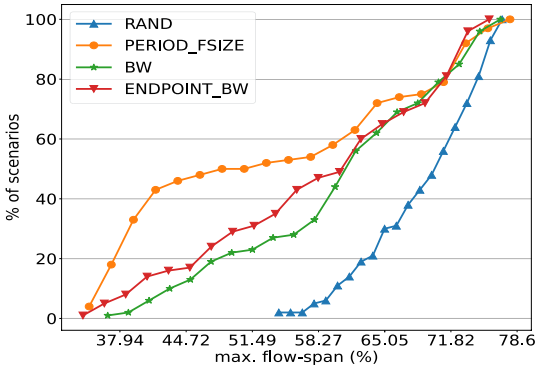

Fig. 11 The percentage of the total number of scenarios having flowspan less than a given value (x-axis) for various greedy criteria on ORION with $|R| = 100$

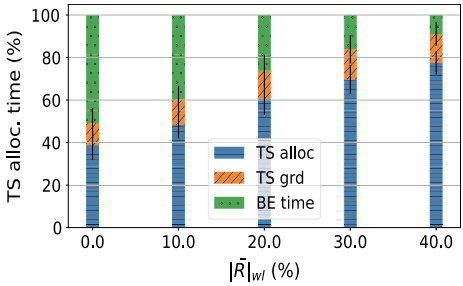

Fig. 12 The impact of the percentage of wireless requests ($|\bar{R}|_{wl}$) on the percentage of time allocation for TT stream requests (blue), guard interval (orange) and BE traffic (green) on ORION with $|R| = 100$ (Color figure online)

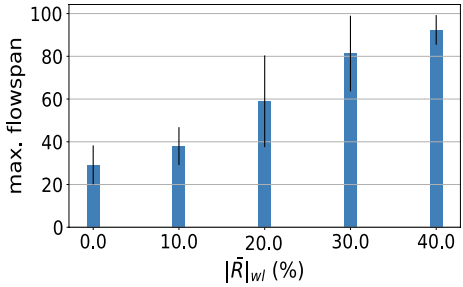

Fig. 13 The impact of the percentage of wireless requests ($|\bar{R}|_{wl}$) on the flowspan time (% of cycle time) on ORION with $|R| = 100$

feasible (not necessarily the optimum) solution. As MESH's average $|P_{r,K}|$ is higher than RING's, the average time to find a schedule with GREEDY in MESH is shorter than in RING.

Also, there is an order of speed difference between the MILP-based approaches (i.e., MILP and MILP2) and the greedy heuristic GREEDY. This can be attributed to the fact that MILP and MILP2 look for an optimum solution, whereas GREEDY returns a feasible greedy solution. It appears that MILP and MILP2 become impractical to solve for $|R| \geq 30$ on MESH or a bigger network (e.g., ORION) because of extremely large solution space arising from the combination of a large number of decision integer variables. As a result, the CPLEX solver takes several hours to return the optimal solution. Fig. 10 plots the evolution of three main CPLEX solver's parameters with time: (i) the objective value (solution to the instantaneous relaxed MILP), (ii) the integer solution and (iii) the lower bound (best achievable integer solution) with time. Evidently, the CPLEX solver has to traverse through the huge solution space of the problem though the returned optimum (integer) solution was found quite early. Therefore, it is clear that the MILP approach quickly becomes intractable, i.e., the execution time exceeding several hours (for $|R| \geq 30$).

7.2.2 GREEDY Criteria

Next, we evaluate the performance of GREEDY comprehensively. To this end, we have chosen ORION as a mixed wired-wireless network. The total number of TT stream requests (|*R*|) is 100.

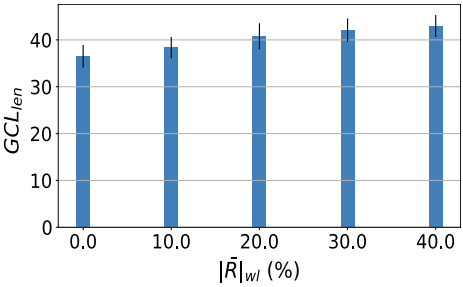

Fig. 14 The impact of the percentage of wireless requests ($|\bar{R}|_{wl}$) on the average number of GCL entries (per node) on ORION with $|R| = 100$

We evaluate the impact of four request sorting criteria (*crit*)– RAND, ENDPOINT_BW, BW and PERIOD_FSIZE on the GREEDY performance. The criteria are summarized in Table 4. RAND serves as a baseline to show GREEDY's performance without any prioritization. The other criteria rely on the heuristic that requests which are more challenging to schedule should be prioritized over those with less stringent requirements (e.g., number of frames per cycle) or lower resource demands (e.g., bandwidth). In ENDPOINT_BW, requests are sorted in ascending order of their endpoint transmission speeds. This approach prioritizes requests with potentially slower transmission rates, usually due to wireless endpoints, thus aiming to address bottlenecks at slower nodes. In BW, requests are sorted in descending order based on their bandwidth requirement ($fsize_r/T_r$), favoring requests that demand more bandwidth over a given period. This prioritization aims to ensure high-bandwidth requests are accommodated early. Finally, in PERIOD_FSIZE requests are sorted in ascending order by their period and in case of ties, those with larger frame sizes (T_r) are prioritized. Figure 11 shows that the PERIOD_FSIZE criterion has the best performance while ENDPOINT_BW has the worst performance after RAND among all criteria. Scheduling requests with smaller periods (T_r) is challenging as the frame injection times (Alg. 3) for such requests are more constrained than for requests with larger T_r.

7.2.3 Wireless Requests Scheduling

For mixed wired-wireless networks, it is essential to evaluate the performance of the scheduling algorithm for wireless TT stream requests. For a given number of total requests ($|R|$), we vary the fraction of requests that have wireless endpoints and observe the impact on the time allocation for TT stream requests, the flowspan time and the average GCL length. The mixed network considered for the evaluation is the same as the one considered in Sect. 7.2.2.

The impact of the percentage of wireless TT stream requests ($|\bar{R}|_{wl}$) on the average time-allocation per link for TT stream requests is shown in Fig. 12 for $|R| = 100$. The height of the blue bar indicates the average time spent in the transmission of TT stream requests and the average guard interval for these streams is indicated by the height of the orange bars. The sum of these heights indicates the time allocated by GREEDY to TT stream requests. The requests are ordered in ascending order based

on the period of requests. The rest of the time in a cycle (height of green bars) is the time in which BE streams can be transmitted. Clearly, as $|\bar{R}|_{wl}$ increases, the average time allocated for TT stream requests increases and the BE stream time is reduced. This is because of the longer wireless transmission delays as compared to the wired transmission delays. Therefore, for a given value of $|R|$, a higher number of $|\bar{R}|_{wl}$ results in more frames traversing wireless links, thus increasing the average TT allocation time. Furthermore, the percentage of guard interval (overhead in allocation) increases slightly with $|\bar{R}|_{wl}$.

Figure 13 illustrates the impact of ($|\bar{R}|_{wl}$) on the flowspan time for $|R| = 100$. The increase in the percentage of wireless requests increases is accompanied by an increase in the flowspan time. This stems from the fact the large value of $|\bar{R}|_{wl}$ results in longer allocation time for TT stream requests thus increasing the flowspan time (cfr. Fig. 12). Further, around $|\bar{R}|_{wl} = 40\%$, the flowspan reaches 90% thus indicating high contention for the network bandwidth, especially in wireless links.

The shared memory available on a switch is a scarce resource and for some chip architectures, it limits the total number of GCL entries per switch (or port) [37]. The average number of GCL entries (GCL_{len}) resulting from the generated schedule `GREEDY` is plotted in Fig. 14. Although, the number of requests ($|R|$) remains constant, by varying $|\bar{R}|_{wl}$, GCL_{len} increases. This is because the number of requests whose schedule cannot be aligned with the already scheduled requests increases with $|\bar{R}|_{wl}$. This observation also corresponds to the increase in guard interval with $|\bar{R}|_{wl}$ (Fig. 12).

8 Conclusion

With the advent of smart factories and Industry 4.0, the need for efficient support of time-critical applications in hybrid wired-wireless networks has become increasingly vital. In this paper, we presented the problem of no-wait end-to-end scheduling in mixed wired-wireless networks. The problem is first defined and then three approaches to solve it were discussed. The MILP-based approaches result in an optimal solution but do not scale beyond a small-sized problem instance. The execution time of the MILP-based approaches was found to be orders of magnitude longer as compared to the greedy heuristic, making the MILP approaches impractical for realistic problem instances. Furthermore, the comparison of the approaches in terms of flowspan showed that the performance gap between the greedy heuristic and MILP-based approaches is reasonably small ($< 5\%$). The results also showed that ordering requests in the increasing order of their time period resulted in the best performance for the greedy heuristic.

In our future work, we would like to address the online scheduling problem in TSN, where TT stream requests arrive in an online fashion and are scheduled one by one with the possibility of re-scheduling the already scheduled TT stream requests. Furthermore, we plan to investigate the integration of communication and cloud scheduling for end-to-end control applications that require time sensitivity in compute along with communications.

Acknowledgements This research was funded by the Flemish FWO SBO S003921N VERI-END.com (Verifiable and elastic end-to-end communication infrastructures for private professional environments) project, by the FWO project under grant agreement #G055619N and the Flemish Government under the "Onderzoeksprogramma Artificiele Intelligentie (AI) Vlaanderen".

Funding Open access funding provided by Royal Institute of Technology. This research was funded by the Flemish FWO SBO S003921N VERI-END.com (Verifiable and elastic end-to-end communication infrastructures for private professional environments) project, by the FWO project under grant agreement #G055619N and the Flemish Government under the "Onderzoeksprogramma Artificiele Intelligentie (AI) Vlaanderen".

Availability of Data and Materials Not applicable.

Declarations

Conflict of interest The authors declare that they have no Conflict of interest.

Ethical Approval Not Applicable.

References

1. Finn, N.: Introduction to time-sensitive networking. IEEE Commun. Stand. Mag. **2**(2), 22–28 (2018)
2. Bartolin-Arnau, L.M., Vera-Perez, J., Sempere-Paya, V.M., Silvestre-Blanes, J.: Private 5G networks for cyber-physical control applications in vertical domains. In: 2023 IEEE 19th international conference on factory communication systems (WFCS), pp. 1–4 (2023). IEEE
3. IEEE Standard for Local and Metropolitan Area Networks--Bridges and Bridged Networks. In: IEEE Std 802.1Q-2022 (Revision of IEEE Std 802.1Q-2018), vol. no., pp.1–2163 (2022). https://doi.org/10.1109/IEEESTD.2022.10004498
4. IEEE Standard for Local and Metropolitan Area Networks -- Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic. In: IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, and IEEE Std 802.1Q-2014/Cor 1-2015), vol. no., pp. 1–57, (2016). https://doi.org/10.1109/IEEESTD.2016.8613095
5. Chandramouli, D., Liebhart, R., Pirskanen, J.: 5G for the Connected World, 1st edn. Wiley, New Jersey (2019)
6. Zhang, L., Zeadally, S.: Enabling end-to-end QoS over hybrid wired-wireless networks. Wirel. Pers. Commun. **38**(2), 167–185 (2006)
7. Haxhibeqiri, J., Jiao, X., Municio, E., Marquez-Barja, J.M., Moerman, I., Hoebeke, J.: Bringing time-sensitive networking to wireless professional private networks. Wirel. Pers. Commun. **121**(2), 1255–1271 (2021)
8. Jiao, X., Liu, W., Mehari, M., Aslam, M., Moerman, I.: Openwifi: a free and open-source IEEE802. 11 SDR implementation on SoC. In: 2020 IEEE 91st vehicular technology conference (VTC2020-Spring), pp. 1–2 (2020). IEEE

9. Sharma, G.P., Patel, D., Sachs, J., De Andrade, M., Farkas, J., Harmatos, J., Varga, B., Bernhard, H.-P., Muzaffar, R., Ahmed, M., Dürr, F., Bruckner, D., De Oca, E.M., Houatra, D., Zhang, H., Gross, J.: Toward deterministic communications in 6G networks: state of the art, open challenges and the way forward. IEEE Access **11**, 106898–106923 (2023). https://doi.org/10.1109/ACCESS.2023.3316605
10. Dürr, F., Nayak, N.G.: No-wait packet scheduling for IEEE time-sensitive networks (TSN). In: Proceedings of the 24th international conference on real-time networks and systems, pp. 203–212 (2016)
11. Craciunas, S.S., Oliver, R.S., Steiner, W.: Formal scheduling constraints for time-sensitive networks. arXiv preprint arXiv:1712.02246 (2017)
12. Schweissguth, E., Danielis, P., Timmermann, D., Parzyjegla, H., Mühl, G.: ILP-based joint routing and scheduling for time-triggered networks. In: Proceedings of the 25th international conference on real-time networks and systems, pp. 8–17 (2017)
13. Tămaş-Selicean, D., Pop, P., Steiner, W.: Design optimization of TTEthernet-based distributed real-time systems. Real-Time Syst. **51**(1), 1–35 (2015)
14. Tamas-Selicean, D., Pop, P., Steiner, W.: Synthesis of communication schedules for TTEthernet-based mixed-criticality systems. In: Proceedings of the eighth IEEE/ACM/IFIP international conference on hardware/software codesign and system synthesis, pp. 473–482 (2012)
15. IEEE Approved Draft Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications. In: IEEE P802.1AS-Rev/D8.3), vol. no., pp.1-516, (2020).
16. IEEE Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams. In: IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005) , vol. no., pp.C1–72, 5 Jan. 2010, doi: 10.1109/IEEESTD.2009.5375704.
17. IEEE Standard for Local and Metropolitan Area networks—Bridges and Bridged Networks—Amendment 24: Path Control and Reservation. In: IEEE Std 802.1Qca-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qcd-2015 and IEEE Std 802.1Q-2014/Cor 1-2015) , vol. no., pp.1–120 (2016). https://doi.org/10.1109/IEEESTD.2016.7434544
18. IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—Amendment 28: Per-Stream Filtering and Policing. In: IEEE Std 802.1Qci-2017 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, IEEE Std 802.1Q-2014/Cor 1-2015, IEEE Std 802.1Qbv-2015, IEEE Std 802.1Qbu-2016, and IEEE Std 802.1Qbz-2016) , vol. no., pp.1–65, (2017). https://doi.org/10.1109/IEEESTD.2017.8064221
19. IEEE Standard for Local and Metropolitan area Networks—Frame Replication and Elimination for Reliability. In: IEEE Std 802.1CB-2017, vol. no., pp.1–102 (2017), https://doi.org/10.1109/IEEESTD.2017.8091139
20. Hanzálek, Z., Burget, P., Šucha, P.: Profinet IO IRT message scheduling. In: 2009 21st Euromicro conference on real-time systems, pp. 57–65 (2009). IEEE
21. Hellmanns, D., Haug, L., Hildebrand, M., Dürr, F., Kehrer, S., Hummen, R.: How to optimize joint routing and scheduling models for TSN using integer linear programming. In: Proc. ACM Int. Conf. Real Time Netw. Syst., Nantes, France (2021)
22. Steiner, W.: An evaluation of SMT-based schedule synthesis for time-triggered multi-hop networks. In: 2010 31st IEEE real-time systems symposium, pp. 375–384 (2010). IEEE
23. Pahlevan, M., Obermaisser, R.: Genetic algorithm for scheduling time-triggered traffic in time-sensitive networks. In: 2018 IEEE 23rd international conference on emerging technologies and factory automation (ETFA), vol. 1, pp. 337–344 (2018). IEEE
24. Wang, Y., Chen, J., Ning, W., Yu, H., Lin, S., Wang, Z., Pang, G., Chen, C.: A time-sensitive network scheduling algorithm based on improved ant colony optimization. Alex. Eng. J. **60**(1), 107–114 (2021)
25. Cavalcanti, D., Perez-Ramirez, J., Rashid, M.M., Fang, J., Galeev, M., Stanton, K.B.: Extending accurate time distribution and timeliness capabilities over the air to enable future wireless industrial automation systems. Proc. IEEE **107**(6), 1132–1152 (2019). https://doi.org/10.1109/JPROC.2019.2903414

26. Adame, T., Carrascosa-Zamacois, M., Bellalta, B.: Time-sensitive networking in IEEE 802.11 be: on the way to low-latency WiFi 7. Sensors **21**(15), 4954 (2021)
27. Ginthör, D., Guillaume, R., Hoyningen-Huene, J., Schüngel, M., Schotten, H.D.: End-to-end optimized joint scheduling of converged wireless and wired time-sensitive networks. In: 2020 25th IEEE international conference on emerging technologies and factory automation (ETFA), vol. 1, pp. 222–229 (2020). IEEE
28. Yen, J.Y.: Finding the K shortest loopless paths in a network. Manage. Sci. **17**(11), 712–716 (1971)
29. Pocovi, G., Pedersen, K.I., Mogensen, P.: Joint link adaptation and scheduling for 5G ultra-reliable low-latency communications. IEEE Access **6**, 28912–28922 (2018)
30. Tramarin, F., Mok, A.K., Han, S.: Real-time and reliable industrial control over wireless lans: algorithms, protocols, and future directions. Proc. IEEE **107**(6), 1027–1052 (2019). https://doi.org/10.1109/JPROC.2019.2913450
31. Asghari, M., Fathollahi-Fard, A.M., Mirzapour Al-e-hashem, S., Dulebenets, M.A.: Transformation and linearization techniques in optimization: a state-of-the-art survey. Mathematics **10**(2), 283 (2022)
32. Pitts, R.A.J.: A mathematical programming approach for routing and scheduling flexible manufacturing cells. Dissertation, Pennsylvania State University (2006). https://etda.libraries.psu.edu/catalog/7230
33. IBM, "IBM ILOG CPLEX optimization studio," Software. [Online]. Available: https://www.ibm.com/products/ilog-cplex-optimization-studio. Accessed: 01 Mar .2024
34. Hagberg, A.A., Schult, D.A., Swart, P.J.: Exploring network structure, dynamics, and function using NetworkX. In: Varoquaux, G., Vaught, T., Millman, J. (eds.) Proceedings of the 7th python in science conference, Pasadena, CA pp. 11–15 (2008)
35. Obermaisser, R.: Time-triggered communication. In: Embedded systems handbook, pp. 14–1. CRC Press, Florida (2017)
36. Zhao, L., Pop, P., Li, Q., Chen, J., Xiong, H.: Timing analysis of rate-constrained traffic in TTEthernet using network calculus. Real-Time Syst. **53**, 254–287 (2017)
37. Hellmanns, D., Glavackij, A., Falk, J., Hummen, R., Kehrer, S., Dürr, F.: Scaling TSN scheduling for factory automation networks. In: 2020 16th IEEE international conference on factory communication systems (WFCS), pp. 1–8 (2020). IEEE

Gourav Prateek Sharma is a postdoctoral researcher in the ISE division (School of EECS), KTH Royal Institute of Technology Sweden since October 2022. Prior to joining KTH, he received his Ph.D. from IDLab, Ghent University in June 2022. His Ph.D. dissertation work involved developing and analyzing optimization algorithms for efficient resource allocation in telecom and media broadcast networks.

Wouter Tavernier received his BS and MS degree in Computer Science in 2002 from Ghent University (Belgium). He joined the Internet-Based Communications Networks group (which became part of IDLab in October 2016) of Ghent University in 2006 as researcher on Carrier Ethernet. In 2012 he obtained a Ph.D degree from the same university on reliable routing and switching. Currently, he is employed as Professor at Ghent University, where he teaches courses on computer networks. His current research interests focus on performance and resource optimization aspects of deterministic and high-performance computing networks. This work is performed in the context of European projects such as 5GPPP NGPAAS, SONATANFV, 5G TANGO, and Horizon Europe projects such as HEXA-X-II and OASEES. This research has been published in more than 120 scientific publications

Didier Colle is senior full professor at Ghent University since 2022. He was associated professor since 2011 and full professor since 2014 at the same university and received a PhD degree in 2002 and a M. Sc. degree in electrotechnical engineering in 1997 from the same university. He is co-responsible for the research cluster on network modelling, design and evaluation (NetMoDeL) inside the IMEC IDlab research group. This research cluster deals with fixed internet architectures and optical networks, green-ict, design of network algorithms and technoeconomic studies. His research is mainly conducted inside international (mainly European), national and bilateral research projects together with the industry. This research has been published in more than 500 international journals and conference articles and has resulted in more than 20 PhD degrees.

Mario Pickavet received a M.Sc. and Ph.D. degree in electrical engineering, specialized in telecommunications in 1996 and 1999, respectively. Since 2000, he is professor at Ghent University where he is teaching courses on discrete mathematics and network modeling. He is co-leading the research cluster on Network Modeling, Design and Evaluation (NetMoDeL). His main research interests are Fixed internet architectures and optical networks, green ICT and design of network algorithms. He has published about 500 international publications, both in journals (IEEE JSAC, IEEE Comm. Mag., Journal of Lightwave Technology, Proceedings of the IEEE, ...) and in proceedings of conferences. He is co-author of the book 'Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS'. He is holder of a bronze medal at the International Mathematical Olympiad (Sweden, 1991).

Jetmir Haxhibeqiri (Member, IEEE) received the master's degree in engineering (information technology and computer engineering) from RWTH Aachen University, Germany, in 2013, and the Ph.D. degree in computer science engineering from Ghent University, in 2019, with his research on flexible and scalable wireless communication solutions for industrial warehouses and logistics applications. Currently, he is a Senior Researcher with the Internet Technology and Data Science Laboratory (IDLab), Ghent University—imec. His current research interests include wireless communications technologies (IEEE 802.11, IEEE 802.15.4e, and LoRa) and their application, the IoT, wireless time sensitive networking, in-band network monitoring, and wireless network management.

Jeroen Hoebeke is an associate professor in the Internet Technology and Data Science Lab of Ghent University and imec. He is conducting and coordinating research on wireless (IoT) connectivity, embedded communication stacks, deterministic wireless communication and wireless network management. This expertise has been applied in a variety of application domains such as logistics, Industry 4.0, building automation, healthcare and animal monitoring. He is particularly active in national funded projects as well as in defining, executing and managing such projects. He has also been involved in several EU research funded projects and is author or co-author of more than 200 publications in international journals or conference proceedings.

Ingrid Moerman received a degree in electrical engineering in 1987, and a Ph.D. degree from Ghent University in 1992, where she became a part-time Professor in 2000. She is a Staff Member with IDLab, Core Research Group, imec with research activities embedded with Ghent University and the University of Antwerp. Her main research interests include cooperative and intelligent radio networks, real- time software-defined radio, time-sensitive networks, dynamic spectrum sharing, coexistence across heterogeneous wireless networks, vehicular networks, open-source prototyping platforms, software tools for programmable networks, next-generation wireless networks (5G/6G), and experimentally supported the research.