Optimal Routing and Scheduling of Complemental Flows in Converged Networks

Jonathan Falk
jonathan.falk@ipvs.uni-stuttgart.de
University of Stuttgart
Stuttgart

Frank Dürr
frank.duerr@ipvs.uni-stuttgart.de
University of Stuttgart
Stuttgart

Steffen Linsenmayer
steffen.linsenmayer@ist.uni-stuttgart.de
University of Stuttgart
Stuttgart

Stefan Wildhagen
stefan.wildhagen@ist.uni-stuttgart.de
University of Stuttgart
Stuttgart

Ben Carabelli
ben.carabelli@ipvs.uni-stuttgart.de
University of Stuttgart
Stuttgart

Kurt Rothermel
kurt.rothermel@ipvs.uni-stuttgart.de
University of Stuttgart
Stuttgart

ABSTRACT

Converged networks support applications with completely different (real-time) requirements. The communication paradigms offered in converged networks are predominantly treated as separate entities from the perspective of traffic engineering, e.g., time-triggered traffic for closed-loop control systems, shaped traffic for multimedia-streaming applications, and best-effort traffic for non-time-critical IT applications. However, there are scenarios where applications benefit from considering time-triggered messages and non-time-triggered messages as complemental components of a single traffic flow. These applications have the property that time-triggered transmissions guarantee basic functionality (e.g., stability of a control system), and additional non-time-triggered transmissions improve the application's performance.

We present how to model these so-called complemental traffic flows for this type of application using a traffic metric for the description of the non-time-triggered traffic part. Furthermore, we show that complemental flows are suitable for traffic engineering by presenting two different approaches for the problem of optimized joint routing and scheduling in converged networks with mixed integer linear programming.

In our evaluations, we use an exemplary min-max objective for the joint routing and scheduling problem which yields an average reduction of the peak value of the traffic metric by 20-30% over constraint-based approaches.

CCS CONCEPTS

• **Networks** → *Network algorithms*; • **Computer systems organization** → *Real-time systems.*

ACM Reference Format:
Jonathan Falk, Frank Dürr, Steffen Linsenmayer, Stefan Wildhagen, Ben Carabelli, and Kurt Rothermel. 2019. Optimal Routing and Scheduling of Complemental Flows in Converged Networks. In *Proceedings of the 27th International Conference on Real-Time Networks and Systems (RTNS 2019), November 6–8, 2019, Toulouse, France.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3356401.3356415

1 INTRODUCTION

Real-time communication is a fundamental requirement for implementing distributed real-time systems. In particular, so-called Networked Control Systems (NCS), where physical processes are controlled through distributed networked sensors, actuators, and controllers, depend on real-time communication. NCS can be found in many areas, such as the Industrial Internet of Things (Industry 4.0) for controlling robots or machines, or on-board systems of cars, planes, etc.

Beyond mere real-time communication, there is a strong trend towards so-called *converged networks*. To avoid the costly installation of dedicated networks for different traffic classes, converged networks support a gamut of communication paradigms ranging from time-triggered real-time communication to simple best-effort communication with class-based priority scheduling in one shared network. In particular, real-time Ethernet technologies are promising candidates to implement converged networks. The relevance of converged networks is also highlighted by the standardization efforts of the IEEE, who have recently ratified a set of standards, commonly referred to Time-Sensitive Networking (TSN) [10], enabling standard Ethernet to implement converged networks. In particular, TSN includes a scheduling discipline called Time-aware Shaper (formerly IEEE 802.1Qbv, now part of IEEE 802.1Q) implementing a time-division multiplexing (TDMA) scheme to enforce temporal isolation of traffic from different traffic classes. The Time-aware Shaper enables real-time communication with deterministic latency bounds and extremely low jitter bounds (in the order of microseconds), alongside other traffic classes with relaxed requirements.

Although the TSN standard describes the scheduling mechanism (Time-aware Shaper) executed by each network element (switch), the algorithmic part to calculate schedules for configuring the network is not covered. To this end, several algorithms for calculating time schedules for the Time-aware Shaper have been proposed in the literature, partially also considering the complementary problem of calculating routes together with schedules (joint routing and scheduling [5]) for time-triggered traffic. These algorithms isolate time-triggered traffic in time (scheduling) and space (routing) from other traffic and aim at different goals such as compactness of schedules [4], latency optimization [21], or repairability [19]. Some approaches also consider non-time-triggered traffic during the calculation of schedules for real-time traffic [18, 24]. However, common

RTNS 2019, November 6–8, 2019, Toulouse, France
2019. ACM ISBN 978-1-4503-7223-7/19/11…$15.00
https://doi.org/10.1145/3356401.3356415

to all of these approaches is the principle of strictly distinguishing between time-triggered flows and non-time-triggered flows and treating them as strictly separate entities.

In contrast to this binary classification and separate treatment of time-triggered and non-time-triggered flows, we argue that some applications would benefit from a third type of flows, which we call *complemental flows*. Complemental flows consist of both, a deterministic real-time part, which has to be delivered deterministically within bounded latency and is generated deterministically according to a time schedule, and a complemental non-time-triggered part transported with relaxed or no latency guarantees (called opportunistic part in the following). The basic idea is that the deterministic part of the flow provides the strictly mandatory guarantees to *safely* operate the system, whereas the opportunistic part just improves the system performance beyond the mandatory minimum. In other words, the deterministic part ensures that nothing "bad" will ever happen, whereas the opportunistic part is "nice to have" and used to further optimize the performance.

NCS are a prime example of a class of applications that would benefit from such complemental flows as we have already outlined in [14]. Here, the deterministic part is used to guarantee the stability of the control system by enforcing latency bounds for a base rate of periodically transmitted sensor values or actuator commands. The opportunistic part transports additional sensor values and actuator commands that allow the controller to improve the control system performance, e.g., by staying closer to the set-point, whenever there is extra bandwidth available.

It is important to see that the value of the opportunistic part is not completely independent from the deterministic part. For instance, an opportunistically transmitted sensor value transmitted immediately after a deterministically transmitted sensor value might carry only little additional information since both values are almost the same. In contrast, an opportunistic value with a greater temporal distance to the previous deterministic value might provide really new(er) information. This example shows that complemental flows should be considered when it comes to planning schedules and routes if we strive for maximum performance.

Since existing TSN routing and scheduling approaches for real-time networks in general and specifically TSN do not have a notion of complemental flows, they miss the opportunity of optimizing opportunistic parts in relation to deterministic parts during scheduling and routing, ultimately resulting in lower application performance. Therefore, we propose specific joint routing and scheduling algorithms for TSN networks explicitly supporting complemental flows in this paper. In detail, the contributions of this paper are the following. We first introduce the concept of complemental flows consisting of time-triggered, periodic transmissions with deterministic real-time requirements and interspersed opportunistic transmissions. We propose different generic models to model opportunistic transmissions, which serve as basis for defining optimization objectives. We then formalize the optimal routing and scheduling problem for complemental flows using mixed integer-linear programs (MILP) that can be solved by standard MILP solvers. Finally, we evaluate the performance of our approach with exemplary min-max objective with respect to the solution quality and solution runtime, showing that our models reduce "traffic concentration" on average by 20-30%.

The rest of this paper is structured as follows. We first discuss related work in Section 2, before we introduce concept of complemental flows in detail in Section 3. Next, we formalize in Section 4 the system model, and subsequently the joint routing and scheduling problem for complemental flows in Section 5 with MILP, and present the evaluation results in Section 6. Finally, we conclude the paper in Section 7.

2 RELATED WORK

In this paper, we use a traffic model where each application generates a complemental flow consisting of both time-triggered and non-time-triggered messages. NCS have been shown to be a type of applications which are able to exploit complemental flows, e.g., in [14] which focuses on the control theoretic aspects, or in [15] which targets a specific network technology (CAN). In this paper, we have a network-centric perspective and show how to compute routes and transmission schedules for applications with complemental flows in converged networks.

The problem of scheduling time-triggered messages in converged networks is a sub-problem of the joint-routing and scheduling problem for complemental flows. To synthesize transmission schedules for time-triggered traffic, it is common to express the scheduling constraints in a constraint-based programming framework such as Linear Programming or Satisfiability Modulo Theories, e.g., scheduling of time-triggered traffic with given routes [3, 4, 16, 23, 25], or joint routing and scheduling of time-triggered traffic [5, 21, 22]. The aforementioned work limits its scope to the time-triggered traffic, addressing questions such as feasibility of the solution and runtime characteristics of the solving process.

In the literature, some integration effects of different traffic types (time-triggered, non-time-triggered, etc.) in converged networks are addressed, usually with the intention of reducing the jitter and delay for non-time-triggered traffic. For example, [24] proposes to explicitly incorporate "gaps" into the transmission schedule for time-triggered messages, and [13, 18] computes optimized routes for AVB traffic, taking into account the existing routing and scheduling of the time-triggered traffic. Apart from converged networks, [12] proposes a centralized scheduling heuristic targeted at wireless sensor-actuator networks (WASN) for event-triggered (i.e., non-time triggered) flows and time-triggered flows with precalculated routes. Besides the system model of WASNs differing from converged networks with TSN (e.g., message reordering, medium access, message sizes), in [12] all messages are subject to real-time constraints.

However, in the aforementioned approaches, time-triggered and non-time-triggered traffic are treated as separate entities, whereas we explicitly consider the interdependency of the two traffic parts with different criticality that compose a complemental flow.

3 COMPLEMENTAL FLOWS AND APPLICATION MODEL

In this section, we describe the concept of complemental flows in detail, and put it into the context of real-time applications.

Deterministic and Opportunistic Messages. We consider real-time applications that, on the one hand, require the *time-triggered, cyclic transmission of messages* that need to be delivered within *deterministic time bounds and limited jitter* via a switched (bridged) real-time

Ethernet network, for instance, following the TSN standard (IEEE 802.1Qbv). We call these messages in the following *deterministic messages*. Such deterministic messages can be found in many real-time applications where timely delivery is a safety requirement. Consider, for example, the diverse applications found in modern cars where controllers, networked sensors, and actuators (servo motors) form several control loops. Here, sensor values need to be delivered from the sensors to the controller, and actuator commands from the controller to the actuators within time bounds, or the car might be damaged or passenger's lives endangered. Depending on the specifications of the car (engine power, speed, accuracy of sensors, etc.), some minimum update (message) rate and cycle time can be defined a priori guaranteeing that the car will be operational with (at least) a minimum required quality (safety condition).

On the other hand, additional updates might increase the accuracy and efficiency (quality) of the operation of the car beyond the safe minimum. Here, quality is a performance metric that should be optimized, and quality is related to transmitted messages. The problem is that network bandwidth is a shared resource since different applications might use the same onboard-network at the same time. Reserving more bandwidth beyond the minimum required bandwidth for one application makes it unavailable for other applications. Moreover, not all applications might require additional bandwidth at all time. For instance, a car cruising with high speed on the autobahn might need fewer updates (messages) for the steering application but benefits from higher updates on the lane-keeping assistance application. While maneuvering in an urban environment at low velocity, higher update rates for steering can improve the driving experience. Therefore, reserving excessive bandwidth in a time-triggered fashion at all time seems to be unjustified. Instead, we propose to use residual network resources *opportunistically*, i.e., when available, to optimize performance. We call messages transmitted outside the a priori defined time-triggered schedule for deterministic traffic *opportunistic messages*.

Our model distinguishing between deterministic and opportunistic messages has relevance beyond this concrete example. For instance, it can be used to implement the well-known class of *weakly-hard real-time (WHRT) systems*. WHRT systems [1, 2] can tolerate a certain number of messages missing their deadline within a time window. We can easily ensure this using deterministic messages at a minimum rate, such that enough deterministic messages are part of each time window, and all other messages within the window are sent as opportunistic messages.

Optimizing Application Performance: Traffic Metric. Complemental flows consist of the aforementioned deterministic and opportunistic messages that guarantee safety (e.g., stability) on the one hand, and optimize performance on the other hand. Considering the optimization of performance through opportunistic messages, a traffic *metric* is required that relates opportunistic messages to their utility. Based on this metric, we would like to schedule and route complemental flows such that performance is maximized (optimization objective). Note that we strive for an *offline* scheduling and routing approach, thus, we need to define traffic metrics that also can be estimated offline. Obviously, these metrics are application-specific, therefore, we cannot provide an exhaustive list of concrete metrics here, but focus on the generic properties of metrics.

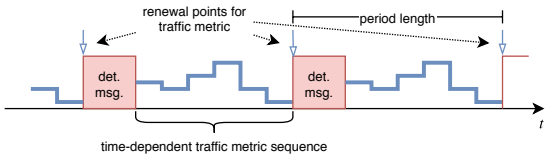

Figure 1: Time-dependent traffic metrics assign a value to time intervals with respect to the position within transmission period.

In general, we distinguish between time-independent and time-dependent metrics. For *time-independent metrics*, the utility of an opportunistic message is not related to the schedule of the deterministic traffic, i.e., the time distance between sending an opportunistic message and the previous deterministic message does not influence the utility of the opportunistic message. For instance, the average bandwidth (rate) of opportunistic messages is such a time-independent traffic metric following the reasonable assumption: the more messages (e.g., sensor values), the higher the performance.

However, this simplifying assumption is obviously neglecting the fact that not all messages might have equal utility, but the utility of a message also depends on its value and time. For instance, a sensor value deviating significantly from the previous deterministically reported value might be more useful than sending the same value again. The problem is that for an offline approach, it is impossible to predict the concrete sensor values transmitted at runtime, which obviously depend on the situation (otherwise, one would not need to send predicted messages at all, as done by the orthogonal model predictive control paradigm).

Therefore, we can make reasonable assumptions, e.g., that opportunistic messages sent at a larger time distance to the previous or next message might provide more useful information than messages send immediately after or before deterministic messages. For instance, sensor readings taken immediately after each other might have the almost same value. This motivates the introduction of *time-dependent traffic metrics* that incorporate the relation of opportunistic messages to the time-schedule of deterministic messages. In other words, a time-dependent metric can be interpreted as an a priori given function over time defining for each point in time the traffic metric value of an opportunistic message sent at that time. Since we assume that utility is defined in relation to the previous and/or next deterministic message, this function only needs to be defined over the time period of one cycle of the deterministic traffic and then repeats for every cycle (cf. Figure 1)

4 SYSTEM MODEL

In this section, we formalize the so far abstract concepts of flows, traffic metrics and network graphs. Moreover, we state our assumptions about the system.

We use calligraphic letters (e.g., $\mathcal{V},\mathcal{E},\mathcal{F}$) for sets. Bold letters denote an indexable data structure $\mathbf{d}$ where $\mathbf{d}[i,k]$ is the component associated with indices i and k, and indexable data structures may be nested. The wildcard character $*$ indicates that element $i \in \mathcal{I}$ may have any possible value from its domain.

We represent a packet-switched communication network by a directed graph $G(\mathcal{V},\mathcal{E})$. Edge $e \in \mathcal{E}$ denotes a directional link (thus, a full-duplex Ethernet link is represented by two edges), and vertex $v \in \mathcal{V}$ denotes a switch (bridge). For each switch in the network,

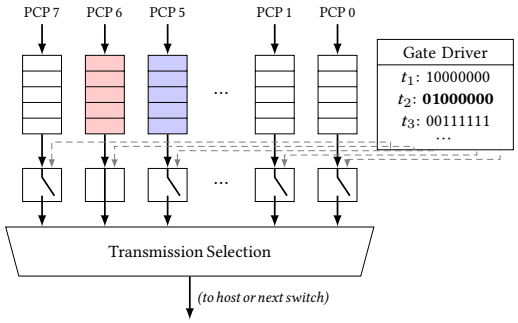

Figure 2: Output port with TAS of IEEE Std 802.1Q-compliant switch. Deterministic and opportunistic messages are enqueued in queue 6 and queue 5, respectively.

there exists a unique vertex v in the graph G. $\mathcal{F}$ denotes the set of all complemental flows f.

Network and Switches. We make some assumptions regarding how flows are handled, which are compatible with the TSN standards: We assume there exists a mechanism to establish routes for flows in the network, such that an incoming message will be transmitted on the correct outgoing link according to the calculated route of that flow. Furthermore, we assume (for the sake of simplicity) that the processing delay is constant and the same for all switches, and all links have the same propagation delay and data rate. Switches can distinguish between different traffic classes, for instance using the Priority Code Point (PCP) field of Ethernet. Each switch uses FIFO-queuing for the aggregate of deterministic traffic, as well as for the aggregate of opportunistic traffic, i.e., switches are not required to distinguish between the individual traffic flows.

Switches have the ability to precisely control the point in time, when messages are scheduled for transmission for each traffic class, e.g., using the Time-aware Shaper (TAS) from IEEE Std. 802.1Q (cf. Figure 2). TAS uses gates for each FIFO queue of each output port to control when messages from a queue are eligible for transmission over that outgoing port. The opening and closing of gates is controlled by a Gate Driver according to a time schedule. The clocks of all switches are synchronized, for example using the Precision Time Protocol (PTP).

These switch capabilities are also required for the *zero-queuing* abstraction for time-triggered (deterministic) messages. As the name indicates, with zero-queuing time-triggered messages traverse the network without queuing. Put differently, with zero-queuing at every switch a time-triggered message will enter an empty queue and is immediately selected for forwarding and sent out. This has several advantages: zero-queuing eliminates the queuing delay, thus, allows for lower jitter, and the message delay depends directly on the path length. Additionally, zero-queuing trivially satisfies ordering constraints imposed by the FIFO property of the switches' queues, and simplifies the validation of the timing behavior of deterministic messages in the design phase.

Complemental Flows. For each complemental flow f of an application, we define the vertex indices of the origination vertex $\mathbf{v}_\mathrm{o}[f] \in \mathcal{V}$ and destination vertex $\mathbf{v}_\mathrm{d}[f] \in \mathcal{V}$, which map to those switches in the network where source node and destination node are attached,

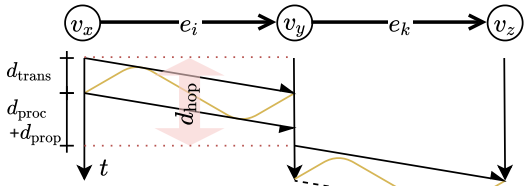

Figure 3: Per-hop delay: Time between start of transmission on two subsequent edges.

respectively. All messages of a specific complemental flow are forwarded through the network according to a static route which is part of the solution of the joint routing and scheduling problem. Application nodes are aware of the distinction between deterministic and opportunistic messages and "annotate" messages in a way that allows switches to map the messages to traffic classes, e.g., by setting different PCP values for deterministic and opportunistic transmissions to achieve queue isolation (cf. Figure 2).

The transmission period of flow f is denoted by $\mathbf{p}[f] \in \mathbb{N}$, and the end-to-end delay bound for an individual deterministic message of flows f is denoted by $\mathbf{t}_\mathrm{ddln}[f] \in \mathbb{N}$. Due to the cyclic behavior, we define that each period of flow f starts with the transmission of a deterministic message of flow f, followed by a time-interval during which opportunistic messages can be transmitted. Thus, we use the deterministic messages as *reference point* for the computation of the transmission schedule for deterministic messages, in the sense that we have to compute suitable phases for the transmission periods for each flow, cf. Figure 4. The delay between the transmission of deterministic messages of f at two neighboring switches on the route of f under zero-queuing (cf. Figure 3) is referred to as *per-hop delay* $\mathbf{d}_\mathrm{hop}[f]$of flow f. It consists of a constant processing delay and a constant propagation delay, and a transmission delay which depends on the size of the deterministic messages of flow f.

For a given problem instance, we expect that for all flows, the traffic metric describes the same property of the opportunistic traffic part. For instance, the traffic metric could be equal to the average bandwidth for the opportunistic traffic part of each flow. Additionally, we require that the traffic metric does not change significantly from hop to hop and is additive in nature. This implies, that the traffic metric is actually a property of the application node, or, to be more exact, a property of the process generating opportunistic messages. This has the practical advantage, that the traffic metrics could be obtained for example by analyzing the application's sending behavior. These requirements for the traffic metric facilitate solving the problem with MILPs, which require that we can model our problem only with linear inequalities and linear objective function. Furthermore, additive traffic metrics are established in the context of quality-of-service, e.g., in admission control schemes [11].

To model the *time-independent traffic metric* of complemental flows, we use single scalar value which contains the aggregated information about the opportunistic transmissions for flow f. Correspondingly, we use a discretized representation which splits each transmission period into a sequence of small intervals (so-called *cells*) for the *time-dependent traffic metric* of flow f, and we assign the respective traffic metric value to each cell.

Note, that depending on the actual implementation and the actual meaning of the traffic metric, the queuing of opportunistic messages

has the potential to influence the temporal relation of the traffic metric. In this paper, we work with the assumption that this influence is negligible in relation to the overall uncertainty regarding the generation of opportunistic messages. We would like to remark, that there are approaches to limit message queuing for opportunistic transmissions, e.g. Controlled Load Services [26] or bufferless statistical multiplexing [20].

Time. In the MILPs, we discretize time in general (not only for the time-dependent traffic metric) at the granularity of a cell (e.g., the respective transmission duration of deterministic messages is given in multiples of cell lengths). A cell denotes a time-interval with length τ. To find a suitable mapping of cell length τ to seconds, we can for example use the greatest common divisor of all transmission durations and delay parameters.

5 ROUTING AND SCHEDULING OF COMPLEMENTAL FLOWS

In this section, we present two approaches for joint routing and scheduling of complemental flows with mixed integer linear programming. Additional definitions required for the formulation of the MILPs are successively introduced together with the constraints in Section 5.1 and Section 5.2, and underlined symbols $\underline{x}$ denote decision variables.

The joint routing and scheduling problem takes as input a graph representing the topology of a converged network and a set of complemental flows representing a set of applications. The solution consists of a route for the messages of each flow and a transmission schedule for the deterministic messages of each flow. In this context a route is a set of edges forming a sequence from the application's source node to application's destination node. The transmission schedule must guarantee deterministic traversal of all deterministic messages subject to the flow requirements. Traffic metrics are used to select from the set of feasible routes and schedules those that optimize the objective. In our MILP formulation, we use a min-max objective function that minimizes the highest accumulated value of the traffic metric in the network.

We first present an MILP with edge-granularity routing and time-independent traffic metric in Section 5.1, and next a MILP with path-granularity routing and time-dependent traffic metric in Section 5.2.

5.1 MILP: Edge-Granularity Routing and Time-Independent Traffic Metric

The MILP in this section is based on the ILP from [5], but extended with the time-independent traffic metric and optimization objective for complemental flows.

This MILP contains most notably the two variables $\underline{\mathbf{u}}[f,e]$ and $\underline{\mathbf{t}}[f,e]$ from which routes and schedules for the flows can be recovered. Variable $\underline{\mathbf{u}}[f,e] \in \{0,1\}$ indicates whether messages of flow f are routed along edge e. Variable $\underline{\mathbf{t}}[f,e] \in \mathbb{N}$ is an integer variable denoting the phase of the transmission periods of flow f on edge e in multiples of τ (cf. Figure 4). We can therefore get the path for f by appending all edges with $\underline{\mathbf{u}}[f,*] = 1$, starting at the origin vertex of f. Similarly, we can derive the transmission schedule for the deterministic messages at each edge e from $\underline{\mathbf{t}}[*,e]$.

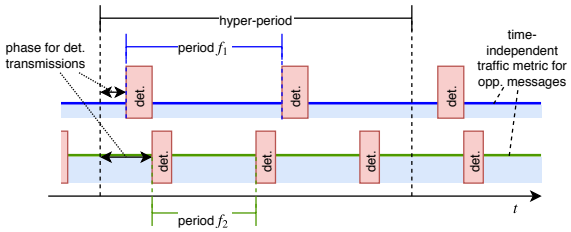

Figure 4: Scheduling by computing feasible phases for the periodic, deterministic transmissions.

Routing Constraints. Constraints (1)-(4) express the properties of a valid route for each flow. For the formulation of the constraints, we define the sets $\mathcal{E}_{\text{in},v} = \{e \in \mathcal{E} | (1 = \mathbf{B}_{\text{VE}}[v,e])\}$ and $\mathcal{E}_{\text{out},v} = \{e \in \mathcal{E} | (-1 = \mathbf{B}_{\text{VE}}[v,e])\}$ which denote the set of all incoming, respectively departing, edges at vertex v using the vertex-edge incidence matrix $\mathbf{B}_{\text{VE}}$ of G. $\mathbf{B}_{\text{VE}}$ is defined as follows: if e departs from v, then $\mathbf{B}_{\text{VE}}[v,e] = -1$, if e enters v $\mathbf{B}_{\text{VE}}[v,e] = 1$, else $\mathbf{B}_{\text{VE}}[v,e] = 0$. The route of each flow f starts at the origination vertex $\mathbf{v}_{\text{o}}[f]$, i.e.,

$$\forall f \in \mathcal{F}: \sum_{e \in \mathcal{E}_{\text{out},\mathbf{v}_{\text{o}}[f]}} \underline{\mathbf{u}}[f,e] = 1. \tag{1}$$

ensures that the route of each flow f contains one edge that departs from the origin vertex of f. Analogously, the constraint

$$\forall f \in \mathcal{F}: \sum_{e \in \mathcal{E}_{\text{in},\mathbf{v}_{\text{d}}[f]}} \underline{\mathbf{u}}[f,e] = 1 \tag{2}$$

ensures that for each flow f, one incoming edge at the destination vertex of f is part of the route of f. Finally, Constraints (3)-(4)

$$\forall f \in \mathcal{F}: \forall v \in \mathcal{V} \setminus \{\mathbf{v}_{\text{o}}[f], \mathbf{v}_{\text{d}}[f]\}: \sum_{e \in \mathcal{E}_{\text{in},v}} \underline{\mathbf{u}}[f,e] = \sum_{e \in \mathcal{E}_{\text{out},v}} \underline{\mathbf{u}}[f,e] \tag{3}$$

$$\forall f \in \mathcal{F}: \forall v \in \mathcal{V} \setminus \{\mathbf{v}_{\text{o}}[f], \mathbf{v}_{\text{d}}[f]\}: \sum_{e \in \mathcal{E}_{\text{in},v}} \underline{\mathbf{u}}[f,e] \leq 1 \tag{4}$$

express that messages of each flow f are routed along a loop-free sequence of connected edges.

Scheduling constraints. We limit the phase of the deterministic transmissions for each flow f to ensure one complete deterministic transmission in every interval with the length of one transmission period of that flow with the constraint

$$\forall f,e \in \mathcal{F} \times \mathcal{E}: \underline{\mathbf{t}}[f,e] + \mathbf{t}_{\text{resv}}[f] \leq \mathbf{p}[f], \tag{5}$$

where $\mathbf{t}_{\text{resv}}[f] \in \mathbb{N}$ denotes the duration of the transmission of a deterministic message of flow f in multiples of τ.

Constraints (6)-(10) are responsible for *interflow scheduling*. Interflow scheduling ensures temporal isolation of deterministic transmissions. For the interflow scheduling, we introduce hyper-period $h = \text{lcm}_{f \in \mathcal{F}}(\mathbf{p}[f]) \in \mathbb{N}$, and the auxiliary variable $\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e] \in \{0,1\}$. Constraints (6)-(8) set $\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e]$ to 1, if messages of f_1 and f_2 are both traversing e, else $\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e] = 0$.

$$\forall f_1,f_2,e \in \mathcal{F} \times \mathcal{F} \times \mathcal{E} \text{ with } f_1 \neq f_2:$$

$$\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e] \leq \underline{\mathbf{u}}[f_1,e] \tag{6}$$

$$\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e] \leq \underline{\mathbf{u}}[f_2,e] \tag{7}$$

$$\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e] \geq \underline{\mathbf{u}}[f_1,e] + \underline{\mathbf{u}}[f_2,e] - 1 \tag{8}$$

We use $\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e]$ in Constraints (9)-(10) to constrain the phases of all flows whose messages traverse an edge e such that all transmissions of deterministic messages of these flows in a hyper-period are non-overlapping, i.e, we "reserve" the edge for mutually exclusive time-intervals with

$$\begin{aligned}
&\forall f_1,f_2,e,a,b \in \mathcal{F}\times\mathcal{F}\times\mathcal{E}\times\mathcal{N}_{\text{inst}}\times\mathcal{N}_{\text{inst}} \\
&\text{with } f_1 \neq f_2,\ a < \frac{h}{\mathbf{p}[f_1]},\ b < \frac{h}{\mathbf{p}[f_2]}: \\
&(\underline{\mathbf{t}}[f_1,e]+a\cdot\mathbf{p}[f_1]+\mathbf{t}_{\text{resv}}[f_1]-(\underline{\mathbf{t}}[f_2,e]+b\cdot\mathbf{p}[f_2])) \\
&-\Big(\underline{\mathbf{o}}_{\text{joint}}[f_1,f_2,e,a,b]+\Big(1-\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e]\Big)\Big)\cdot M_{\text{aux}} \leq 0 \qquad (9)\\
&(\underline{\mathbf{t}}[f_2,e]+b\cdot\mathbf{p}[f_2]+\mathbf{t}_{\text{resv}}[f_2]-(\underline{\mathbf{t}}[f_1,e]+a\cdot\mathbf{p}[f_1])) \\
&-\Big(\Big(1-\underline{\mathbf{o}}_{\text{joint}}[f_1,f_2,e,a,b]\Big)+\Big(1-\underline{\mathbf{u}}_{\text{joint}}[f_1,f_2,e]\Big)\Big)\cdot M_{\text{aux}} \leq 0. \qquad (10)
\end{aligned}$$

In the formulation of Constraints (9)-(10), we have to consider that neither do we know the routes of the flows a priori, nor do we know the order of the transmissions a priori. This is reflected in an implication (if messages of f_1 and f_2 are routed over e, ensure temporal isolation of deterministic transmissions of f_1 and f_2), and a disjunction (ensure transmission of deterministic message of f_1 before transmission of deterministic message of f_2 or vice versa). For the disjunction, we define the set of the number of possible periods per hyper-period for all flows $\mathcal{N}_{\text{inst}} = \{0,\ldots,\max_f\left(\frac{h}{\mathbf{p}[f]}\right)\} \subset \mathbb{N}$, and a sufficiently large (e.g., $2\cdot h$) constant M_{aux} (aka bigM), and we introduce the auxiliary variable $\underline{\mathbf{o}}_{\text{joint}}[f_1,f_2,e,a,b] \in \{0,1\}$ with $f_1,f_2 \in \mathcal{F}, e \in \mathcal{E}, a,b \in \mathcal{N}_{\text{inst}}$.

Constraints (11)-(17)) are the intraflow scheduling constraints. The intraflow scheduling constraints enforce zero-queuing for deterministic messages. Again, we first introduce an auxiliary variable $\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2] \in \{0,1\}$. If e_1 is followed by e_2 on the route of messages of flow f, then $\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2] = 1$, else $\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2] = 0$. To this purpose, we use Constraints (11)-(13):

$$\begin{aligned}
&\forall f \in \mathcal{F}: \forall v \in \mathcal{V}: \forall e_1 \in \mathcal{E}_{\text{in},v}, e_2 \in \mathcal{E}_{\text{out},v}: \\
&\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2] \leq \underline{\mathbf{u}}[f,e_1] \qquad (11)\\
&\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2] \leq \underline{\mathbf{u}}[f,e_2] \qquad (12)\\
&\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2] \geq \underline{\mathbf{u}}[f,e_1]+\underline{\mathbf{u}}[f,e_2]-1 \qquad (13)
\end{aligned}$$

In Constraints (14)-(17), we constrain the start times for the reservations on subsequent edges on the route of each flow f to enforce zero-queuing. Zero-queuing applies only for edges on the route (implication) of the messages, and due to the cyclic behavior, the phase of the transmission period can "wrap around" (i.e., $\underline{\mathbf{t}}[f,e_1]+\mathbf{d}_{\text{hop}}[f] \bmod \mathbf{p}[f] = \underline{\mathbf{t}}[f,e_2]$) [5]. Since, modulus is not a native MILP construct, we model this with an auxiliary variable $\underline{\mathbf{o}}_{\text{seq}}[f,e_{\text{in}},e_{\text{out}}] \in \{0,1\}$, a large constant M_{aux}, and the disjunction $\underline{\mathbf{t}}[f,e_1]+\mathbf{d}_{\text{hop}}[f] = \underline{\mathbf{t}}[f,e_2]$ *or* $\underline{\mathbf{t}}[f,e_1]+\mathbf{d}_{\text{hop}}[f] = \underline{\mathbf{t}}[f,e_2]+\mathbf{p}[f]$. To express implication, disjunction and equality (reservation on subsequent edges are shifted by $\mathbf{d}_{\text{hop}}[f]$), we end up with the four linear inequality constraints for zero-queuing

$$\begin{aligned}
&\forall f \in \mathcal{F}: \forall v \in \mathcal{V}: \forall e_1 \in \mathcal{E}_{\text{in},v}, e_2 \in \mathcal{E}_{\text{out},v}: \\
&\Big(\underline{\mathbf{t}}[f,e_2]-\underline{\mathbf{t}}[f,e_1]-\mathbf{d}_{\text{hop}}[f]\Big) \\
&-\Big(\underline{\mathbf{o}}_{\text{seq}}[f,e_1,e_2]+\Big(1-\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2]\Big)\Big)\cdot M_{\text{aux}} \leq 0 \qquad (14)\\
&\Big(\underline{\mathbf{t}}[f,e_2]-\underline{\mathbf{t}}[f,e_1]-\mathbf{d}_{\text{hop}}[f]\Big) \\
&+\Big(\underline{\mathbf{o}}_{\text{seq}}[f,e_1,e_2]+\Big(1-\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2]\Big)\Big)\cdot M_{\text{aux}} \geq 0 \qquad (15)\\
&\Big(\underline{\mathbf{t}}[f,e_2]+\mathbf{p}[f]-\underline{\mathbf{t}}[f,e_1]-\mathbf{d}_{\text{hop}}[f]\Big) \\
&-\Big(\Big(1-\underline{\mathbf{o}}_{\text{seq}}[f,e_1,e_2]\Big)+\Big(1-\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2]\Big)\Big)\cdot M_{\text{aux}} \leq 0 \qquad (16)\\
&\Big(\underline{\mathbf{t}}[f,e_2]+\mathbf{p}[f]-\underline{\mathbf{t}}[f,e_1]-\mathbf{d}_{\text{hop}}[f]\Big) \\
&+\Big(\Big(1-\underline{\mathbf{o}}_{\text{seq}}[f,e_1,e_2]\Big)+\Big(1-\underline{\mathbf{u}}_{\text{seq}}[f,e_1,e_2]\Big)\Big)\cdot M_{\text{aux}} \geq 0. \qquad (17)
\end{aligned}$$

To keep the end-to-end delay bounds for messages of flow f, Constraint (18) bounds the delay accumulated along each flow's route.

$$\forall f \in \mathcal{F}: \sum_{e\in\mathcal{E}} \mathbf{d}_{\text{hop}}[f]\cdot\underline{\mathbf{u}}[f,e] \leq \mathbf{t}_{\text{ddln}}[f] \qquad (18)$$

Traffic Metric. We introduce variable $\underline{\mathbf{m}}_{\text{edge}}[e] \in \mathbb{R}_+$. With

$$\forall e \in \mathcal{E}: \sum_{f\in\mathcal{F}} \mathbf{m}[f]\cdot\underline{\mathbf{u}}[f,e] = \underline{\mathbf{m}}_{\text{edge}}[e], \qquad (19)$$

$\underline{\mathbf{m}}_{\text{edge}}[e]$ is set to the accumulated value of the traffic metric on a particular edge e. We use min-max optimization to avoid an extremely uneven distribution of the opportunistic traffic (metric). To this end,

$$\forall e \in \mathcal{E}: \underline{\mathbf{m}}_{\text{edge}}[e] \leq \underline{m}_{\text{mm,edge}} \qquad (20)$$

sets the auxiliary variable $\underline{m}_{\text{mm,edge}} \in \mathbb{R}_+$ to the globally highest value of the accumulated traffic metric. Our objective

$$\text{minimize } \underline{m}_{\text{mm,edge}} \qquad (21)$$

thus minimizes this value. This combination of constraints for the traffic metric reduces the highest value of the accumulated values of the traffic metric in the whole network. Due to flow conservation, this can distribute the traffic (subject to the timing constraints of deterministic messages) more evenly. However, using $\underline{\mathbf{m}}_{\text{edge}}[e]$ and replacing Constraints (20)-(21) we can easily adapt the MILP. For example, if our traffic metric is the average bandwidth, we can restrict $\underline{\mathbf{m}}_{\text{edge}}[e]$ to not exceed the link bandwidth on edge e by adding an additional constraint ($\underline{\mathbf{m}}_{\text{edge}}[e] \leq \text{threshold}$).

5.2 MILP: Path-Granularity Routing and Time-Dependent Traffic Metric

The MILP we present next trades in the spatial granularity regarding routing for higher temporal granularity with respect to the traffic metric (time-independent vs. time-dependent). Instead of deciding edge-by-edge whether an edge is part of the route of a flow, path-granularity routing selects one route from a set of predefined paths for each flow. We allow different numbers of (candidate) paths per flow and keep a list of available path indices for flow f denoted by $\mathbf{n}_{\text{path}}[f] = [0,1,2,\ldots]$. We use a binary variable $\underline{\mathbf{u}}_{\text{path}}[f,r] \in \{0,1\}$

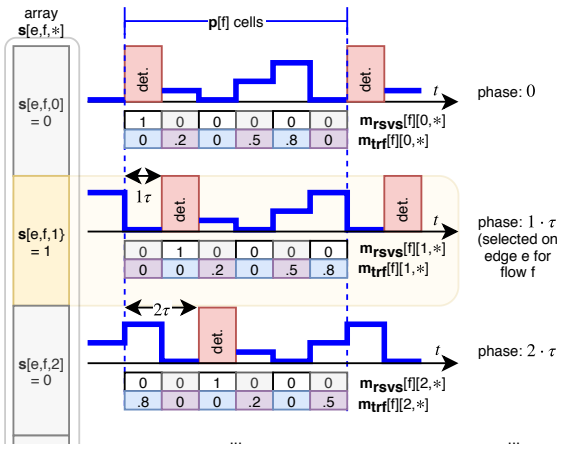

Figure 5: Rows of the cell matrices ($\mathbf{m}_{\text{rsvs}}[f]$, $\mathbf{m}_{\text{trf}}[f]$) represent the complemental traffic parts for different phases. Variable $\underline{\mathbf{s}}[e,f,c]$ indicates whether phase $c \cdot \tau$ is selected for transmission period for flow f on edge e.

to indicate whether path $r \in \mathbf{n}_{\text{path}}[f]$ is selected as route for flow f. $\mathbf{R}[f,r] = [e_a, e_b, e_c, \ldots]$ denotes the path r for flow f, and is a sequence of edges $e \in \mathcal{E}$ from origination vertex to destination vertex of f. Different from the previous MILP, the length of precomputed paths has to be restricted accordingly (cf. Constraint (18)) to incorporate the deadline parameter for deterministic messages.

The time-dependent properties of the complemental flows in this MILP are represented in a discretized fashion with the use of cell matrices, cf. Figure 5. The list of cell indices in a transmission period of flow f is denoted by $\mathbf{n}_{\text{prd}}[f] = [0, 1, \ldots, \mathbf{p}[f] - 1]$. For each flow f, we use the cell matrix $\mathbf{m}_{\text{rsvs}}[f] \in \{0,1\}^{\mathbf{p}[f] \times \mathbf{p}[f]}$ to indicate cells reserved for the transmission of deterministic messages, and similarly a cell matrix $\mathbf{m}_{\text{trf}}[f] \in \mathbb{R}_+^{\mathbf{p}[f] \times \mathbf{p}[f]}$ to store the traffic metric sequences. For both matrices, each *row* is associated with a specific phase for the transmission period of flow f, and each *column* index is associated with the corresponding cell in a time-interval with the same length as the flow period. If flow f is scheduled with phase c_{phase}, then $\mathbf{m}_{\text{rsvs}}[f][c_{\text{phase}}, c_{\text{index}}] = 1$ if the transmission of the deterministic message is in progress in cell c_{index}, else $\mathbf{m}_{\text{rsvs}}[f][c_{\text{phase}}, c_{\text{index}}] = 0$, cf. Figure 5. Similarly, if flow f is scheduled with phase c_{phase}, the value of the traffic metric in cell c_{index} is $\mathbf{m}_{\text{trf}}[f][c_{\text{phase}}, c_{\text{index}}]$. Note, that due to the periodical nature of deterministic transmissions and their property as a reset point for the traffic metric sequence, it is sufficient to consider $c_{\text{phase}}, c_{\text{index}} \in \mathbf{n}_{\text{prd}}[f]$. We can construct both matrices ($\mathbf{m}_{\text{trf}}[f]$, $\mathbf{m}_{\text{rsvs}}[f]$) iteratively from the flow parameters with the roll-operator. The operator $\text{roll}(\mathbf{x}, i)$ rotates the components of $\mathbf{x}$ by i positions to the right, overflowing components being appended on the left. Thus, we have

$$\mathbf{m}_{\text{rsvs}}[f][c_{\text{phase}}, *] = \text{roll}\Big(\mathbf{m}_{\text{rsvs}}[f][0,*], c_{\text{phase}}\Big) \tag{22}$$

for the rows of $\mathbf{m}_{\text{rsvs}}[f]$ where the first $\mathbf{t}_{\text{resv}}[f]$ cells in $\mathbf{m}_{\text{rsvs}}[f][0,*]$ have value 1 (indicating a deterministic transmission) and the remaining cells have value 0. Similarly, we have

$$\mathbf{m}_{\text{trf}}[f][c_{\text{phase}}, *] = \text{roll}\Big(\mathbf{m}_{\text{trf}}[f][0,*], c_{\text{phase}}\Big) \tag{23}$$

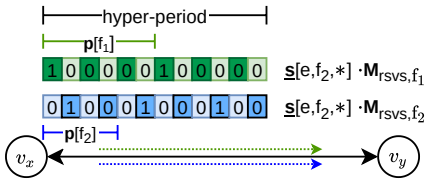

Figure 6: Each cell in the hyper-period can only be assigned to deterministic messages of at most one flow.

for the rows of $\mathbf{m}_{\text{trf}}[f]$ where $\mathbf{m}_{\text{trf}}[f][0,*]$ contains the values of the traffic metric sequence for phase 0.

With this in mind, we can rephrase the scheduling problem as: how much do we have to rotate the cell arrays? The rotation (=phase) is encoded by help of a set of indicator variables $\underline{\mathbf{s}}[e,f,c] \in \{0,1\}$ which encodes the phase c of flow f on edge e. If we consider

$$\underline{\mathbf{s}}[e,f,*] = \begin{bmatrix} \underline{\mathbf{s}}[e,f,0] & \underline{\mathbf{s}}[e,f,1] & \underline{\mathbf{s}}[e,f,2] & \cdots \end{bmatrix}$$

as another row vector, then the phase for the deterministic transmissions of flow f on edge e in the period is equal to the (zero-based) index c of that variable $\underline{\mathbf{s}}[e,f,c]$ which has value 1 (cf. Figure 5). Therefore, $\underline{\mathbf{s}}[e,f,*]$ contains at most one component with $\underline{\mathbf{s}}[e,f,c] = 1$.

Routing Constraints. In this MILP, the routing constraint

$$\forall f \in \mathcal{F}: \sum_{r \in \mathbf{n}_{\text{path}}[f]} \underline{\mathbf{u}}_{\text{path}}[f,r] = 1 \tag{24}$$

chooses one path out of the set of possible paths for each flow, i.e., retrieving the route of f from the MILP solution is trivial. It has to be considered in the scheduling constraints that different paths for one flow can contain identical sub-paths. We define the set $\mathcal{E}_{\text{r}}[f]$ which contains all edges on any path $\mathbf{R}[f,*]$ of a particular flow f, and the set $\mathcal{R}_{e,f} = \{r \in \mathbf{n}_{\text{path}}[f] | e \in \mathbf{R}[f,r]\}$ of all indices r of paths of flow f which contain edge e. Therefore, to link the scheduling with the routing decision, the constraint

$$\forall f \in \mathcal{F}: \forall e_f \in \mathcal{E}_{\text{r}}[f]: \underline{\mathbf{u}}_{\text{edge}}[e_f, f] = \sum_{r_e \in \mathcal{R}_{e_f,f}} \underline{\mathbf{u}}_{\text{path}}[f, r_e] \tag{25}$$

sets auxiliary variable $\underline{\mathbf{u}}_{\text{edge}}[e,f] \in \{0,1\}$ to 1, if edge e is on the chosen path of flow f, else $\underline{\mathbf{u}}_{\text{edge}}[e,f] = 0$.

Scheduling Constraints. Analogously to Constraint (5) we restrict the phase of the transmission periods with

$$\forall f \in \mathcal{F}: \forall e_f \in \mathcal{E}_{\text{r}}[f]: \\ \forall c \in \big\{c \in \mathbf{n}_{\text{prd}}[f] \,|\, c > (\mathbf{p}[f] - \mathbf{t}_{\text{resv}}[f])\big\}: \underline{\mathbf{s}}[e_f, f, c] = 0, \tag{26}$$

i.e., for each flow f only those phases can be selected for which there is one complete transmission of a deterministic message per interval with the length of a transmission period of flow f.

To force all entries in $\underline{\mathbf{s}}[e,f,*]$ to zero on those edges which are not part of the selected route, we use the constraint

$$\forall f \in \mathcal{F}: \forall e_f \in \mathcal{E}_{\text{r}}[f]: \sum_{c_{\text{shift}} \in \mathbf{n}_{\text{prd}}[f]} \underline{\mathbf{s}}[e_f, f, c_{\text{shift}}] = \underline{\mathbf{u}}_{\text{edge}}[e_f, f]. \tag{27}$$

Next, we have the interflow-scheduling constraint in matrix-notation

$$\forall e \in \mathcal{E}: \begin{bmatrix} 0 \\ \vdots \end{bmatrix} \leq \sum_{f_e \in \mathcal{F}_e} \Big(\mathbf{M}^{\intercal}_{\text{rsvs},f_e} \cdot \underline{\mathbf{s}}[e, f_e, *]^{\intercal} \Big) \leq \begin{bmatrix} 1 \\ \vdots \end{bmatrix} \tag{28}$$

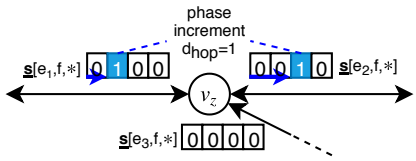

Figure 7: The phase is adjusted by $d_{\mathbf{hop}}[f]$ from hop to hop on the route used by f. On unused edges, all components of $\underline{\mathbf{s}}[e_1,f,*]$ have value 0.

with $\mathcal{F}_e = \{f \in \mathcal{F} | e \in \mathcal{E}_r[f]\}$ denoting the set of flows whose paths contain e. The matrix $\mathbf{M}_{\text{rsvs},f_e}$ is constructed as follows

$$\mathbf{M}_{\text{rsvs},f_e} = \underbrace{\left[\mathbf{m}_{\text{rsvs}}[f_e] \quad \cdots \quad \mathbf{m}_{\text{rsvs}}[f_e]\right]}_{\frac{h}{\mathbf{p}[f_e]} \text{ times}},$$

i.e., we concatenate matrix $\mathbf{m}_{\text{rsvs}}[f_e]$ such that each cell in a row of $\mathbf{M}_{\text{rsvs},f_e}$ is associated with one interval of length τ in the *hyper-period*. The term $\underline{\mathbf{s}}[e,f_e,*] \cdot \mathbf{M}_{\text{rsvs},f_e}$ yields a row vector which indicates which cells in the *hyper-period* are used for the transmissions of deterministic messages of flow f on edge e for the selected phase (encoded in $\underline{\mathbf{s}}[e,f_e,*]$), cf. Figure 6. Since element-wise inequalities are more common for column-vectors, we use the relation $(AB)^\intercal = B^\intercal A^\intercal$ and sum over all flows on edge e. $\sum_{f_e}\left(\mathbf{M}^\intercal_{\text{rsvs},f_e} \cdot \underline{\mathbf{s}}[e,f_e,*]^\intercal\right)$ gives a column vector which indicates how many flows use a specific cell for transmitting a deterministic message. By enforcing that for each edge e each cell in the hyper-period is assigned to the transmission of *at most* one deterministic message, we ensure temporal isolation of deterministic messages. Note that the time schedule for deterministic transmissions at edge e can be recovered from the MILP solution by evaluating the sum term in Constraint (28).

For the corresponding intraflow-scheduling (cf. Figure 7), we define the matrix $\mathbf{m}_{\text{shift}}[f] = \text{roll}\left(\mathbf{I}_{\mathbf{p}[f]\times\mathbf{p}[f]}, \mathbf{d}_{\text{hop}}[f]\right)$ where $\text{roll}(\mathbf{I},x)$ rotates the columns of the identity matrix $\mathbf{I}$ by x positions to the right. The constraints

$$\begin{aligned}
&\forall f \in \mathcal{F} : \forall r \in \mathbf{n}_{\text{path}}[f] : (e_1,e_2) \subseteq \mathbf{R}[f,r] : \\
&\mathbf{m}_{\text{shift}}[f]^\intercal * \underline{\mathbf{s}}[e_1,f,*]^\intercal - \left(\mathbf{1} - \mathbf{1} \cdot \underline{\mathbf{u}}_{\text{edge}}[e_1,f]\right) \\
&\qquad \leq \underline{\mathbf{s}}[e_2,f,*]^\intercal - \left(\mathbf{1} - \mathbf{1} \cdot \underline{\mathbf{u}}_{\text{edge}}[e_2,f]\right) \qquad (29) \\
&\mathbf{m}_{\text{shift}}[f]^\intercal * \underline{\mathbf{s}}[e_1,f,*]^\intercal + \left(\mathbf{1} - \mathbf{1} \cdot \underline{\mathbf{u}}_{\text{edge}}[e_1,f]\right) \\
&\qquad \geq \underline{\mathbf{s}}[e_2,f,*]^\intercal + \left(\mathbf{1} - \mathbf{1} \cdot \underline{\mathbf{u}}_{\text{edge}}[e_2,f]\right) \qquad (30)
\end{aligned}$$

enforce zero-queuing. In Constraints (29)-(30), the tuples (e_1,e_2) are the pairs of subsequent edges of $\mathbf{R}[f,r]$, and $\mathbf{1}$ is a column vector with $\mathbf{p}[f]$ ones. Here, we need no disjunction for the "wrap-around" of the phase (cf. Constraints (9)-(10)), since it is accounted for by the roll-operation. If we ignore the routing aspect, then the scheduling part of these constraints can be expressed with the matrix equation

$$\forall f \in \mathcal{F} : \forall \text{ subsequent edges } e_1,e_2 \text{ on the route of } f : \\ \underline{\mathbf{s}}[e_1,f,*] \cdot \mathbf{m}_{\text{shift}}[f] = \underline{\mathbf{s}}[e_2,f,*]$$

or equivalently, $\mathbf{m}_{\text{shift}}[f]^\intercal * \underline{\mathbf{s}}[e_1,f,*]^\intercal = \underline{\mathbf{s}}[e_2,f,*]^\intercal$. This means, the phase (encoded) in $\underline{\mathbf{s}}[e,f,*]$ is advanced along the edges on the chosen path for flow f according to the respective per-hop delay, cf.

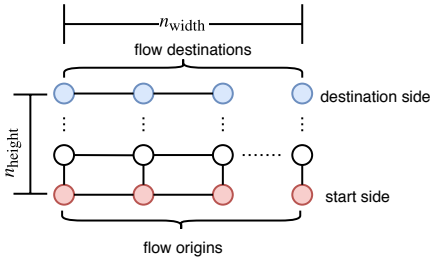

Figure 8: Scenario used for the evaluations.

Figure 7. However, since we are solving the joint routing and scheduling problem, Constraints (29)-(30) contain the additional auxiliary terms $\left(\mathbf{1} - \mathbf{1} \cdot \underline{\mathbf{u}}_{\text{edge}}[e_*,f]\right)$ which trivially satisfy the constraints if not both edges (e_1 and e_2) are part of the chosen path.

Traffic Metric for Non-Time-Triggered Traffic. The constraint

$$\forall e \in \mathcal{E} : \sum_{f_e \in \mathcal{F}_e} \left(\mathbf{M}^\intercal_{\text{trf},f_e} \cdot \underline{\mathbf{s}}[e,f_e,*]^\intercal\right) = \underline{\mathbf{m}}_{\text{cell}}[e,*]^\intercal \qquad (31)$$

computes the value of the accumulated amount of the traffic metric $\underline{\mathbf{m}}_{\text{cell}}[e,c_{\text{hyper}}] \in \mathbb{R}_+$ on edge e, but for each cell c_{hyper} in the set of cell indices C_{hyper} for the hyper-period. Structurally, Constraint (31) is similar to Constraint (28), only that we consider the traffic metric instead of reservations here. Matrix $\mathbf{M}_{\text{trf},f_e}$ is constructed by

$$\mathbf{M}_{\text{trf},f_e} = \underbrace{\left[\mathbf{m}_{\text{trf}}[f_e] \quad \cdots \quad \mathbf{m}_{\text{trf}}[f_e]\right]}_{\frac{h}{\mathbf{p}[f_e]} \text{ times}}. \qquad (32)$$

As it was the case in Section 5.1, we give the constraints and the objective function for the min-max-optimization, but with one difference: the objective is to minimize the maximum value $\underline{m}_{\text{mm,cell}} \in \mathbb{R}_+$ *in any cell* on any edge:

$$\forall e \in \mathcal{E} : \forall c_{\text{hyper}} \in C_{\text{hyper}} : \underline{\mathbf{m}}_{\text{cell}}[e,c_{\text{hyper}}] \leq \underline{m}_{\text{mm,cell}} \qquad (33)$$

$$\text{minimize } \underline{m}_{\text{mm,cell}} \qquad (34)$$

Variable $\underline{\mathbf{m}}_{\text{cell}}[e,c_{\text{hyper}}]$ exposes the accumulated values of the traffic metric, facilitating the adaption of the the objective function of the MILP with path-granularity and time-dependent traffic metric.

6 EVALUATION

In this section, we first give an overview of the evaluation setup, before we present the evaluation results.

Evaluation Setup. We implemented the MILPs with Pyomo [8, 9], and used Gurobi 8.1.0 [7] to solve the MILPs for our evaluation scenarios in a containerized environment on a computing node (4× Intel Xeon E7-4850, 2.1 GHz, 1 TB RAM) running Linux 4.19.4.

Each evaluation scenario consists of a network (graph) and a set of flows. The evaluation scenarios use a regular 4-by-4 grid (cf. Figure 8) as underlying network topology to allow flow placement with a well-known spatial distribution. The values of propagation delay and processing delay are 1 cell length for all edges, respectively, vertices. All flows start at vertices on one side (start side) and the flow destinations are located on the opposite side (destination side) of the grid, thus there exist multiple paths for each pair of origin vertex and destination vertex. To achieve an approximately similar number of a flows per origination vertex and destination vertex, we assign

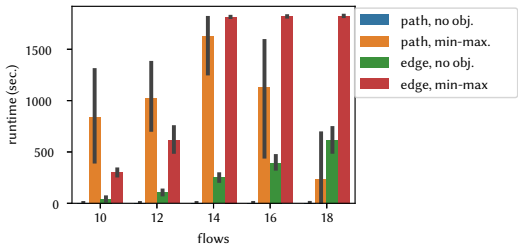

Figure 9: Average solver runtime, varying number of flows.

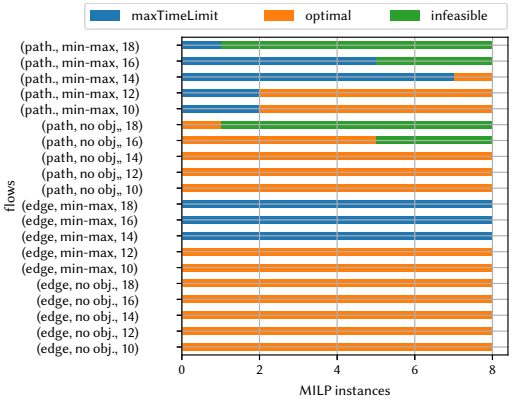

Figure 10: Exit status of solver, varying number of flows.

flows in a round-robin fashion to the vertices on the start side and the destination side. For each flow, the number of cells per period is randomly drawn from the set $\{10, 20, 40\}$, transmissions of deterministic messages require a reservation of 1 cell length, and the deadline (length of precomputed paths, respectively) is such that up to two thirds of all vertices may be traversed by each flow.

Due to the different granularity of routing and the traffic metric description in the MILPs from Section 5.1 and Section 5.2, there are two "pseudo-equivalent" variants of the flow parameters for each evaluation scenario, i.e., for each flow f in the set of flow parameters for the edge-granularity approach, there is a flow f in the set of flow parameters for the path-granularity approach which differs only with respect to the routing and the value of the traffic metric: The flow parameters for MILPs with path-granularity routing contain for each flow f a set of paths which we precomputed with the graph-tool library 2.27 [17]. For the time-dependent traffic metric, we create a random sequences by assigning the absolute values of normally distributed random numbers to the cells of $\mathbf{m}_{\text{trf}}[f][0,*]$. The time-independent traffic metric value ($\mathbf{m}[f]$) is set to the average of $\mathbf{m}_{\text{trf}}[f][0,*]$ of the pseudo-equivalent flow f.

In addition to the MILP with edge-granularity and time-independent traffic metric from Section 5.1 (abbr.: *edge, min-max*) and the MILP with path-granularity and time-independent traffic metric from Section 5.2 (abbr.: *path, min-max*), we additionally use a version of both MILPs without objective. The MILP with edge-granularity routing and time-independent traffic metric and no objective function (abbr.: *edge, no-obj*) lacks Constraint (20). Similarly, the MILP with path-granularity routing and time-dependent traffic and no objective (abbr.: *path., no-obj.*) metrics lacks Constraint (33). Note, that the MILPs without objective solve the "classic" problem of joint routing and scheduling of time-triggered traffic with zero-queuing (ignoring the opportunistic traffic parts of complemental flows).

In our evaluations, we limit the solver runtime to 30 min. For the evaluated scenarios, this did not affect the schedulability, only the optimality of the routes and schedules.

Evaluation Results. Next, we present the evaluation results. Figure 9 and Figure 10 show the results for evaluation scenarios with the number of flows ranging from 10 to 18 flows, and 4 precomputed routes for the MILPs with path-granularity. In Figure 9, we plot the average runtime in seconds for evaluation scenarios with different number of flows. For 10 and 12 flows the MILPs with edge-gran. routing and time-independent traffic metrics are solved faster on average, even though the size of the MILPs with edge-granularity routing are in general a magnitude larger compared the MILPs with path-gran. routing for our evaluation scenarios (edge-gran: from $\sim 1.1{\times}10^5$ constraints and $\sim 1.3{\times}10^5$ variables for 10 flows up to $\sim 3.4{\times}10^5$ constraints and $\sim 3.5{\times}10^5$ variables for 18 flows; path-gran: from $\sim 2{\times}10^4$ constraints and $\sim 8{\times}10^3$ variables for 10 flows up to $\sim 2.7{\times}10^4$ constraints and $\sim 1.1{\times}10^4$ variables for 18 flows). Interestingly, this is not reflected in terms of memory usage of the solver , where we observed similar values ($\leq$ 12.3 GB) during runtime for both approaches. However, the solver hits the run-time limit of 30 min for two instances of evaluation scenarios with only 10 flows for the path-gran., min-max MILP, whereas all instances of evaluation scenarios with 10 and 12 flows for the edge-gran., min-max MILPs are solved optimally in less than 30 min.

In Figure 9, we also we observe an at a first glance counter-intuitive decrease of the average runtime of the path-gran, min-max MILPs for 16 and 18 flows. The reason, why the solver appears to require less runtime to solve the harder problems with more flows is indicated in Figure 10 which shows the solver status for the different instances of the evaluation scenarios. Starting at 16 flows, there are evaluations scenarios which result in infeasible MILPs with path-granularity routing (irregardless whether or not we have an objective function). Since infeasibility is detected relatively quickly by state-of-the-art solvers, these MILP solutions, which actually indicate the absence of a solution for our joint routing and scheduling problem, skew the runtime observations.

This interpretation is supported by the runtime results in Figure 11 where we fix the number of flows to 16 flows, and vary the number of precomputed paths. The corresponding solver statuses for these MILP with path-gran. routing are given in Figure 12. The required runtime for the path-gran., min-max MILPs for different numbers of precomputed paths correlates with the number of infeasible evaluation scenarios, i.e., the number of infeasible evaluation instances decreases monotonically from 5 infeasible scenarios with only 2 precomputed paths to 0, i.e., 10 out of 10 evaluation scenarios are schedulable for evaluation scenarios with 20 or more paths. Or in other words, the schedulability of our evaluation scenarios increases (as expected) with the number of precomputed paths, if we use the approach with path-gran. routing. As before (Figure 9 and Figure 10), all evaluation scenarios with the edge-granularity variant of the flow-parameters are schedulable with the MILPs with edge-gran. routing, i.e., all edge-gran, no obj. MILPs terminate optimally for these evaluation scenarios, and all edge-gran, min-max MILPs yield a (possibly) non-optimal feasible solution within 30 min.

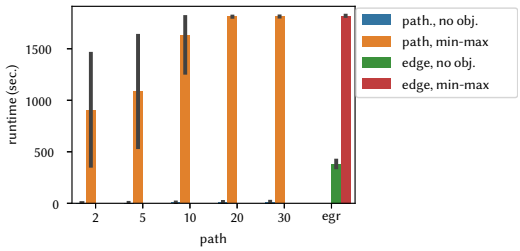

Figure 11: Average solver runtime, varying number of precomputed paths (egr. denoting edge-gran. routing).

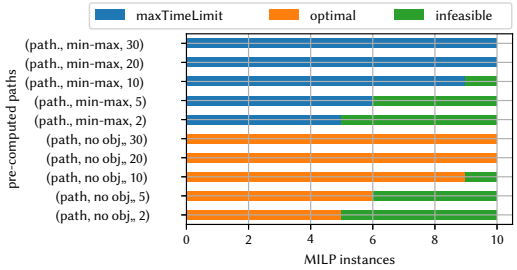

Figure 12: Solver status for MILP with path-gran. routing for varying number of precomputed paths.

If we consider only the MILPs without objective, we can conclude that for our evaluation scenarios the path-granularity approach results not only in smaller MILPs, but is also much faster. In both, Figure 9 and Figure 11 the MILP with path-gran. and time-dependent traffic metric without objective is solved in a few seconds (Figure 9: $\leq$ 4.1 s; Figure 11: $\leq$ 17.1 s for 30 paths), which is much faster compared to the average runtime of the MILP with edge-based routing and time-independent traffic metric (Figure 9: $\leq$ 99 s for 10 flows, $\leq$ 14.7 min for 18 flows; Figure 11: $\leq$ 12 min). In turn, the additional degrees of freedom for routing in the edge-based routing approaches massively improve the schedulability.[1]

For both (edg-gran. and path-gran.) no-obj. MILPs, we retain the constraints to compute the accumulated values of the traffic metric per edge (Constraint (19)) or per cell per edge (Constraint (31)). Despite our method for generating the traffic metrics, simply comparing the traffic metric values of the approach with edge-gran. routing to those of the approach with path-gran. routing is unfair, since they have different objectives (accumulated traffic metric value *per-edge* vs. accumulated value *per-edge per-cell*). Instead, we calculate the respective reduction of the objective value (highest value of the accumulated traffic metric values per edge/per cell per edge)

$$\text{reduction edge-gran., t.i.: } \left(\max_{e\in\mathcal{E}} \underline{\mathbf{m}}_{\text{edge}}[e]\right)/\underline{m}_{\text{mm,edge}} - 1$$

$$\text{reduction path-gran., t.d.: } \left(\max_{c\in C_{\text{hyper}}, e\in\mathcal{E}} \underline{\mathbf{m}}_{\text{cell}}[e,c]\right)/\underline{m}_{\text{mm,cell}} - 1$$

yielded by the min-max MILPs over the no-obj MILPs.

In Figure 13, the reduction is depicted for the evaluations scenarios with varying number of paths from Figure 11 and Figure 12. In other words, Figure 13 shows the improvement over the case where

[1]Note, that this holds for the problem of joint routing and scheduling of time-triggered traffic with zero-queuing, too, since the MILPs without objective solve this problem.

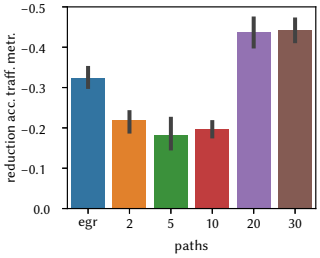

Figure 13: Average reduction of highest value of aggregated traffic metric for varying number of precomputed paths.

a classic time-triggered routing and scheduling approach (which ignores the opportunistic messages) is used for complemental flows. The approach with edg-gran. routing reduces the highest accumulated value of the traffic metric on any edge on average by 32.4%, with the extrema ranging from a reduction of as less as 15.5% up to a reduction of 50.8%. The approach with path-gran. yields an average reduction of the highest accumulated value of the traffic metric in any cell on any edge in the range of 18.3% (for 5 paths) to 44.2% (30 paths). The (feasible) evaluation scenarios from Figure 9 and Figure 10 yield similar results with an average reduction of 23.9% for the approach with path-gran. routing and an average reduction of 30.8% for the approach with edge-gran. routing.

For our evaluation scenarios, we observed that there is often a noticeable improvement shortly after the solver has found the first feasible solution. Thus, depending on the relative value of computing resources, time, and the required "optimality" of the solution, even if we abort the solving process prematurely, the joint routing and scheduling for complemental flows quickly yields substantial improvements over the pure-constraint based solution which ignore the opportunistic traffic part.

7 CONCLUSION AND OUTLOOK

In this paper, we explain the concept of complemental flows for applications in converged networks. We provide two MILPs to jointly compute optimized routes and schedules for applications with complemental flows with different granularity. Our evaluations with a min-max objective show that both approaches can be used, e.g., for better distribution of the complemental flows compared to approaches for pure time-triggered flows, leading to an average reduction of 20% to 30% of the accumulated traffic metric for the non-time-triggered traffic even with limited solver runtime.

Our approach uses offline estimates of the traffic metrics for the opportunistic traffic parts. Therefore, an integrated simulation of the converged network (cf. using [6]) and the applications generating the complemental flows might provide further insights into the relation of the traffic metrics to the actual performance improvements of the application during runtime. Working towards a more integrated approach which explicitly considers the queuing behavior for non-time-triggered messages is another challenge in extension to the work presented in this paper.

ACKNOWLEDGMENT

This work was supported by the German Research Foundation (DFG) under the research grant "Integrated Controller Design Methods and Communication Services for Networked Control Systems (NCS)" (RO 1086/20-1, AL 316/13-1).

REFERENCES

[1] M. E. M. Ben Gaïd, D. Simon, and O. Sename. 2008. A Design Methodology for Weakly-Hard Real-Time Control. In *17th IFAC World Congress (IFAC WC '08)*. IFAC, Seoul, South Korea, 7. https://hal.inria.fr/inria-00269209

[2] G. Bernat, A. Burns, and A. Liamosi. 2001. Weakly Hard Real-Time Systems. *IEEE Trans. Comput.* 50, 4 (April 2001), 308–321. https://doi.org/10.1109/12.919277

[3] S. S. Craciunas, R. S. Oliver, M. Chmelík, and W. Steiner. 2016. Scheduling Real-Time Communication in IEEE 802.1Qbv Time Sensitive Networks. In *Proceedings of the 24th International Conference on Real-Time Networks and Systems (RTNS '16)*. 183–192. https://doi.org/10.1145/2997465.2997470

[4] F. Dürr and N. G. Nayak. 2016. No-Wait Packet Scheduling for IEEE Time-Sensitive Networks (TSN). In *Proceedings of the 24th International Conference on Real-Time Networks and Systems (RTNS '16)*. 203–212. https://doi.org/10.1145/2997465.2997494

[5] J. Falk, F. Dürr, and K. Rothermel. 2018. Exploring Practical Limitations of Joint Routing and Scheduling for TSN with ILP. In *2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications*. Hakodate, Japan, 136–146. https://doi.org/DOI%002010.1109/RTCSA.2018.00025

[6] J. Falk, D. Hellmanns, B. Carabelli, N. G. Nayak, F. Dürr, S. Kehrer, and K. Rothermel. 2019. NeSTiNg: Simulating IEEE Time-sensitive Networking (TSN) in OMNeT++. In *Proceedings of the 2019 International Conference on Networked Systems (NetSys)*. Garching b. München, Germany.

[7] LLC Gurobi Optimization. 2018. Gurobi Optimizer Reference Manual. http://www.gurobi.com

[8] W. E. Hart, C. D. Laird, J.-P. Watson, D. L. Woodruff, G. A. Hackebeil, B. L. Nicholson, and J. D. Siirola. 2017. *Pyomo–optimization modeling in python* (second ed.). Vol. 67. Springer Science & Business Media.

[9] W. E. Hart, J.-P. Watson, and D. L. Woodruff. 2011. Pyomo: modeling and solving mathematical programs in Python. *Mathematical Programming Computation* 3, 3 (2011), 219–260.

[10] IEEE Computer Society. 2018. IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks. *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)* (July 2018), 1–1993. https://doi.org/10.1109/IEEESTD.2018.8403927

[11] S. Jamin, S. J. Shenker, and P. B. Danzig. 1997. Comparison of Measurement-Based Admission Control Algorithms for Controlled-Load Service. In *, Proceedings IEEE INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution*, Vol. 3. 973–980 vol.3. https://doi.org/10.1109/INFCOM.1997.631035

[12] X. Jin, A. Saifullah, C. Lu, and P. Zeng. 2019. Real-Time Scheduling for Event-Triggered and Time-Triggered Flows in Industrial Wireless Sensor-Actuator Networks. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. 1684–1692. https://doi.org/10.1109/INFOCOM.2019.8737373

[13] S. M. Laursen, P. Pop, and W. Steiner. 2016. Routing Optimization of AVB Streams in TSN Networks. *SIGBED Rev.* 13, 4 (Nov. 2016), 43–48. https://doi.org/10.1145/3015037.3015044

[14] S. Linsenmayer, B. W. Carabelli, F. Dürr, J. Falk, F. Allgöwer, and K. Rothermel. 2019. Integration of Communication Networks and Control Systems Using a Slotted Transmission Classification Model. In *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*. 1–6. https://doi.org/10.1109/CCNC.2019.8651811

[15] P. Marti, A. Camacho, M. Velasco, and M. E. M. Ben Gaid. 2010. Runtime Allocation of Optional Control Jobs to a Set of CAN-Based Networked Control Systems. *IEEE Transactions on Industrial Informatics* 6, 4 (Nov. 2010), 503–520. https://doi.org/10.1109/TII.2010.2072961

[16] R. S. Oliver, S. S. Craciunas, and W. Steiner. 2018. IEEE 802.1Qbv Gate Control List Synthesis Using Array Theory Encoding. In *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. 13–24. https://doi.org/10.1109/RTAS.2018.00008

[17] T. P. Peixoto. 2014. The graph-tool python library. *figshare* (2014). https://doi.org/10.6084/m9.figshare.1164194

[18] P. Pop, M. L. Raagaard, S. S. Craciunas, and W. Steiner. 2016. Design Optimisation of Cyber-Physical Distributed Systems Using IEEE Time-Sensitive Networks. *IET Cyber-Physical Systems: Theory Applications* 1, 1 (2016), 86–94. https://doi.org/10.1049/iet-cps.2016.0021

[19] F. Pozo, G. Rodriguez-Navas, and H. Hansson. 2018. Schedule Reparability: Enhancing Time-Triggered Network Recovery Upon Link Failures. In *2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. 147–156. https://doi.org/10.1109/RTCSA.2018.00026

[20] M. Reisslein, K. W. Ross, and S. Rajagopal. 2002. A Framework for Guaranteeing Statistical QoS. *IEEE/ACM Transactions on Networking* 10, 1 (Feb. 2002), 27–42. https://doi.org/10.1109/90.986511

[21] E. Schweissguth, P. Danielis, D. Timmermann, H. Parzyjegla, and G. Mühl. 2017. ILP-Based Joint Routing and Scheduling for Time-Triggered Networks. In *Proceedings of the 25th International Conference on Real-Time Networks and Systems (RTNS '17)*. 8–17. https://doi.org/10.1145/3139258.3139289

[22] F. Smirnov, M. Glaß, F. Reimann, and J. Teich. 2017. Optimizing Message Routing and Scheduling in Automotive Mixed-Criticality Time-Triggered Networks. In *Proceedings of the 54th Annual Design Automation Conference 2017 (DAC '17)*. ACM, Austin, TX, USA, 48:1–48:6. https://doi.org/10.1145/3061639.3062298

[23] W. Steiner. 2010. An Evaluation of SMT-Based Schedule Synthesis for Time-Triggered Multi-Hop Networks. In *2010 31st IEEE Real-Time Systems Symposium*. 375–384. https://doi.org/10.1109/RTSS.2010.25

[24] W. Steiner. 2011. Synthesis of Static Communication Schedules for Mixed-Criticality Systems. In *2011 14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*. 11–18. https://doi.org/10.1109/ISORCW.2011.12

[25] W. Steiner, S. S. Craciunas, and R. S. Oliver. 2018. Traffic Planning for Time-Sensitive Communication. *IEEE Communications Standards Magazine* 2, 2 (June 2018), 42–47. https://doi.org/10.1109/MCOMSTD.2018.1700055

[26] J. Wroclawski <jtw@lcs.mit.edu>. 1997. RFC 2211: Specification of the Controlled-Load Network Element Service. https://tools.ietf.org/html/rfc2211