Scheduling and Robust Invariance in Networked Control Systems

Masoud Bahraini, Mario Zanon, Paolo Falcone, and Alessandro Colombo

***Abstract*— In Networked Control Systems (NCS), impairments of the communication channel can be disruptive to stability and performance. In this paper, we consider the problem of scheduling the access to limited communication resources for a number of decoupled systems subject to state and input constraints, whose loops need to be closed over the network. The schedule must be designed to robustly preserve the invariance property of each system, which in turn guarantees constraint satisfaction. To that end, we first focus on systematic offline scheduling design to preserve robust invariance, and afterwards on online scheduling design with the aim of improving performance and compensating for packet losses while guaranteeing recursive schedulability.**

***Index Terms*— Networked control systems, Robust invariance, Packet loss, Multi-channel scheduling**

I. Introduction

Recent progress in wireless communication technologies has provided new opportunities but also new challenges to control theorists. On the one hand, the use of communication in the control loop has several benefits, such as reduced system wiring, ease of maintenance and diagnosis, and ease of implementation [1]. On the other hand, wireless links are subject to noise, time varying delay, packet loss, jitter, limited bandwidth, and quantization errors, which are not negligible in the stability and performance analysis. Feedback control systems that are closed through a network are called networked control systems (NCS). An important open issue for NCS is the design of communication and control strategies which guarantee that the state remains in an invariant set—a compact subset of its state space—over an infinite time horizon.

The problem of keeping the state in an invariant set in the presence of uncertainties was introduced more than four decades ago [2], [3]. Since then, reachability analysis has been exploited in different applications, e.g., in model checking and safety verification to ensure the system does not enter an unsafe region [4]–[6]. Reachability analysis has several applications in model predictive control, such as terminal set design, recursive feasibility [7], and robust invariant set computation [8]. However, the available results do not directly apply to the case in which the control loop is closed over a non-ideal communication network. There exist techniques to deal with some network imperfections. An $\mathcal{H}_\infty$ filter based on hidden Markov models is designed in [9] for a class of discrete Markov jump linear systems with additive noise when the communication channel is subject to packet losses. A resilient estimation problem is discussed in [10], where redundant communication channels with packet losses described by a Bernoulli probability distribution are used in parallel to improve reliability. A review of recent advances in the filed is given in [11]. To the best of our knowledge, no result covers the case in which a central decision maker is in charge of suitably deciding which node can communicate measurement data or control inputs through the common communication network, see Fig. 1, in order to keep the states of all nodes within given admissible sets. This is, e.g., a model of remotely sensed and actuated robotic systems based on CAN communication, or of remote multi-agent control setups for intelligent transportation systems field testing. A connection between this setting and a classical scheduling problem, called pinwheel problem (PP) [12], [13], has been shown in [14], where the invariance problem is translated into a scheduling problem for single-channel networks. This work was followed up in [15] with an online version of the scheduling algorithm, to improve performance and provide robustness against packet losses and in [16] by jointly designing control and schedule. In this paper, we tackle a more general framework which includes more general couplings of system dynamics and network communications, as well as multi-channel networks.

This work was partially supported by the Wallenberg Artificial Intelligence, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation. Masoud Bahraini is with Chalmers University of Technology, Göteborg, Sweden (e-mail: masoudb@chalmers.se). Mario Zanon is with IMT School for Advanced Studies Lucca, Italy. Paolo Falcone is with Chalmers University of Technology and DIEF, Università di Modena e Reggio Emilia, Italy . Alessandro Colombo is with Politecnico di Milano, Italy.

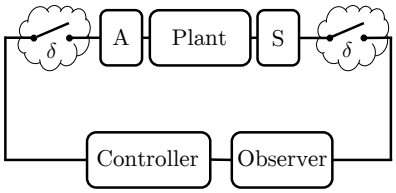

Fig. 1: Feedback loop for each node, which is closed through shared communication channels. The communication traffic is regulated by a central decision maker and specified by decision variable δ.

We study scheduling design to guarantee invariance for NCS consisting of uncertain constrained systems and multiple communication channels with packet loss. The main contributions of this study, which complements the results in [14]–[16], include (i) allowing one to model communication constraints on both measurements and controls; (ii) modeling an arbitrary number of channels; (iii) accounting for the fact that different nodes may have access to a different number of channels; (iv) accounting for stochastic packet loss; (v) tackling feedback loops with a dynamic controller and a state observer. Moreover, we prove that the invariance / scheduling problem is decidable in finite time and we propose improved scheduling algorithms to solve it.

The rest of the paper is organized as follows. In Section II, the problem is formulated and relevant mathematical background is stated. Our main contributions are divided into an offline and an online strategy, presented in Section III and Section IV, respectively. Examples and numerical simulations are provided in Section V.

II. Problem Statement and Background

In this section, we introduce the framework on which this paper's results will be based. Our objective is to elucidate the relations between communication and control for NCS where measurements and/or control signals may travel through the communication medium, with static or dynamic control laws, and at the same time describe systems where the scheduler has direct access to the full state information of the nodes, or where the state is only inferred through an observer. To cover all these cases under as compact a notation as possible, we proceed as follows.

With reference to Fig. 1, we define a binary variable δ_i for each node i, which defines the fact that the node's sensor and/or actuator be connected (when $\delta_i = 1$) or disconnected (when $\delta_i = 0$). We consider both cases where both sensors and actuators are simultaneously connected and disconnected (as in Fig. 1), and cases where only sensors or actuators can be disconnected (Fig. 1, by removing one of the two switches).

For each node, x_i, $\hat{x}_i$ and u_i denote the state of the plant, the state of the observer and the state of the controller, respectively. Note that in NCS the controller typically has memory to cope with packet loss. The discrete-time dynamics of the nodes are described by

$$z_i^+ = \begin{cases} F_i\big(z_i, v_i\big), & \text{if } \delta_i = 1 \\ \hat{F}_i\big(z_i, v_i\big), & \text{if } \delta_i = 0 \end{cases} \tag{1}$$

with

$$F_i := \begin{pmatrix} f_i\big(z_i, v_i\big) \\ g_i\big(z_i\big) \\ c_i\big(z_i\big) \end{pmatrix}, \quad \hat{F}_i := \begin{pmatrix} \hat{f}_i\big(z_i, v_i\big) \\ \hat{g}_i\big(z_i\big) \\ \hat{c}_i\big(z_i\big) \end{pmatrix}, \quad z_i := \begin{pmatrix} x_i \\ \hat{x}_i \\ u_i \end{pmatrix}, \tag{2}$$

where $z_i \in \mathcal{X}_i \times \mathcal{X}_i \times \mathcal{U}_i \subseteq \mathbb{R}^{n_i} \times \mathbb{R}^{n_i} \times \mathbb{R}^{r_i}$ denotes the system states and $v_i \in \mathcal{V}_i \subset \mathbb{R}^p$ denotes an external disturbance. The two dynamics correspond to the *connected mode*, i.e., $\delta_i = 1$ and the *disconnected mode*, i.e., $\delta_i = 0$.

In order to represent a wide variety of network topologies, including single and multi-channel networks, and networks where different number of channels are available at different nodes, we introduce the concept of *connection pattern*. In mathematical terms, a connection pattern $C \in \boldsymbol{\mathcal{C}}$ is an ordered tuple of the indices of the nodes which can be simultaneously connected, i.e.,

$$C := (c_1, \ldots, c_m), \qquad m < q, \qquad c_i \in \mathbb{I}_1^q, \forall\ i \in \mathbb{I}_1^m. \tag{3}$$

Throughout the paper, we denote the connection pattern selected at time t as $C(t)$, and the schedule of selected connection patterns as $\mathbf{C} = C(0), C(1), \ldots$, where $C(t)$ might have different cardinalities at different t. Consequently,

$$i \in C(t) \Leftrightarrow \delta_i(t) = 1. \tag{4}$$

For example, $C(t) = (1, 5)$ means that only nodes 1 and 5 are connected at time t, i.e., $\delta_1(t) = 1$, $\delta_5(t) = 1$, $\delta_j(t) = 0$, $j \notin \{1, 5\}$.

We can now formulate the control task as follows.

Problem 1 (P1)**.** Design a communication allocation such that the node state z_i remains inside an *admissible* set $\mathcal{A}_i \subset \mathcal{X}_i \times \mathcal{X}_i \times \mathcal{U}_i$ at all time, for all $i \in \mathbb{I}_1^q$.

In order to translate P1 into a scheduling problem, we define the concept of *safe time interval*.

Set $\mathcal{S}_i \subset \mathcal{X}_i \times \mathcal{X}_i \times \mathcal{U}_i$ is *robust invariant* for (1) in connected mode, i.e., $\delta_i = 1$, if

$$F_i(z_i, v_i) \in \mathcal{S}_i, \qquad \forall\ z_i \in \mathcal{S}_i,\ v_i \in \mathcal{V}_i. \tag{5}$$

Any robust invariant set $\mathcal{S}_i$ contains all forward-time trajectories of the node (1) in the connected mode, provided $z_i(0) \in \mathcal{S}_i$, regardless of the disturbance v_i. Let $\{\mathcal{S}_i\}$ be the set of all robust invariant sets of (1) that are contained in the admissible set $\mathcal{A}_i$. We call $\mathcal{S}_{i,\infty} \in \{\mathcal{S}_i\}$ the *maximal robust invariant* set:

$$\mathcal{S}_i \subseteq \mathcal{S}_{i,\infty}, \qquad \forall \mathcal{S}_i \in \{\mathcal{S}_i\}. \tag{6}$$

We define the 1-*step reachable set* as the set of states z_i that can be reached in one step from a set of initial states $\mathcal{O}_i$ with dynamics $H_i \in \{F_i, \hat{F}_i\}$:

$$\mathrm{Reach}_1^{H_i}(\mathcal{O}_i) := \{H_i(z_i, v_i) :\ z_i \in \mathcal{O}_i,\ v_i \in \mathcal{V}_i\}. \tag{7}$$

The t-step reachable set, $t = 2, \ldots$ is defined recursively as

$$\mathrm{Reach}_t^{H_i}(\mathcal{O}_i) := \mathrm{Reach}_1^{H_i}(\mathrm{Reach}_{t-1}^{H_i}(\mathcal{O}_i)). \tag{8}$$

Numerical tools for the calculation of $\mathcal{S}_{i,\infty}$ and $\mathrm{Reach}_t^{H_i}(\mathcal{O}_i)$ can be found in [17], for linear H_i.

Definition 1 (Safe time interval)**.** We define the safe time interval for node i as

$$\alpha_i := 1 + \max_\tau \Big\{\tau : \mathrm{Reach}_\tau^{\hat{F}_i}(\mathrm{Reach}_1^{F_i}(\mathcal{S}_{i,\infty})) \subseteq \mathcal{S}_{i,\infty}\Big\}. \tag{9}$$

Essentially, α_i counts the amount of time steps during which node i can be disconnected while maintaining its state in $\mathcal{S}_{i,\infty}$, provided that its initial state is in $\mathcal{S}_{i,\infty}$. Note that, by definition of $\mathcal{S}_{i,\infty}$, node i remains in $\mathcal{S}_{i,\infty}$ for all future times when connected.

The task of keeping the state of each node in its admissible set can now be formulated as follows.

Problem 2 (P2)**.** Given the set of q nodes, each described by (1), an admissible set $\mathcal{A} := \mathcal{A}_1 \times \ldots \times \mathcal{A}_q$, and the set $\boldsymbol{\mathcal{C}}$ of connection patterns (3), determine if there exists an infinite sequence over the elements of $\boldsymbol{\mathcal{C}}$ such that

$$z_i(t) \in \mathcal{S}_{i,\infty},\ \forall z_i(0) \in \mathcal{S}_{i,\infty},\ v_i(t) \in \mathcal{V}_i, i \in \mathbb{I}_1^q,\ t > 0.$$

In other words, we seek an infinite sequence of connection patterns

$$\mathbf{C} := C(0),\ C(1), \ldots, \tag{10}$$

with $C(t) \in \boldsymbol{\mathcal{C}}$ that keeps $(x_i, \hat{x}_i, u_i)$ of all q nodes within $\mathcal{S}_{i,\infty}$ despite the fact that, due to the structure of set $\boldsymbol{\mathcal{C}}$, at each time step some nodes are disconnected. Note that the set $\boldsymbol{\mathcal{C}}$ is assumed to be fixed and given *a priori*, based on the network structure.

A schedule solving P2 is any sequence of C such that every node i is connected at least once every α_i steps. More in general, given a scheduling problem P, a schedule $\mathbf{C}$ that solves P is called a *feasible schedule* for that problem. We write that instance I is accepted by P, denoted

$$I \in \mathrm{P}, \tag{11}$$

if and only if a feasible schedule $\mathbf{C}$ exists for the problem.

We now introduce two classic scheduling problems whose solution we will use, in Theorem 2 and 3, to construct schedules solving P2.

Pinwheel Problem (PP) (From [18])**.** Given a set of integers $\{\alpha_i\}$ with $\alpha_i \geq 1$, determine the existence of an infinite sequence of the symbols $1, \ldots, q$ such that there is at least one symbol i within any subsequence of α_i consecutive symbols.

With $C(t) \in \mathbb{I}_1^q$, a schedule solving PP can be defined as

$$\mathbf{C} := C(1), C(2), \ldots\ . \tag{12}$$

Conditions for schedulability, i.e., existence of a feasible schedule for PP, have been formulated in terms of the *density* of a problem instance I, defined as

$$\rho(I) := \sum_i \frac{1}{\alpha_i}. \tag{13}$$

Proposition 1 (From [19], [20])**.** Given an instance $I := \{\alpha_i\}$ of PP, if $\rho(I) > 1$ then $I \notin$ PP, if $\rho(I) \leq 0.75$ then $I \in$ PP, if $\rho(I) \leq \frac{5}{6}$ and there exists $i : \alpha_i = 2$ then $I \in$ PP, if $\rho(I) \leq \frac{5}{6}$ and I has only three symbols then $I \in$ PP, if $\rho(I) \leq 1$ and I has only two symbols then $I \in$ PP.

It has been conjectured that any instance of PP with $\rho(I) \leq \frac{5}{6}$ is schedulable; however, the correctness of this conjecture has not been proved yet, see [12]. Determining whether a general instance of PP with $\frac{5}{6} < \rho(I) \leq 1$ is schedulable, is not possible just based on the

density $\rho(I)$ (e.g., $\rho(\{2,2\}) = 1$ is schedulable and $\rho(\{2,3,12\}) = \frac{11}{12}$ is not schedulable). Furthermore, determining the schedulability of dense instances, i.e., when $\rho(I) = 1$, is NP-hard in general [13].

Since a schedule for PP is an infinite sequence of symbols, the scheduling search space is also infinite dimensional. Fortunately, the following proposition alleviates this issue.

Proposition 2 (Theorem 2.1 in [13])**.** All instances of PP that admit a schedule admit a *cyclic schedule*, i.e., a schedule whose symbols repeat periodically.

Proposition 2 is paramount for the derivation of our results, since it allows us to restrict our attention to cyclic schedules without any loss of generality.

WSP is a more general version of PP, where multiple symbols can be scheduled at the same time. We call *channels* the multiple strings of symbols that constitute a Windows Schedule.

Windows Scheduling Problem (WSP) (From [21])**.** Given the set of integers $\{\alpha_i\}$ with $\alpha_i \geq 1$, determine the existence of an infinite sequence of ordered tuples with $m_\mathrm{c} \geq 1$ elements of the set $\{1,\dots,q\}$ such that there is at least one tuple that contains the symbol i within any subsequence of α_i consecutive tuples.

An instance $\{m_\mathrm{c}, \{\alpha_i\}\}$ of WSP is accepted, denoted

$$I = \{m_\mathrm{c}, \{\alpha_i\}\} \in \text{WSP}, \tag{14}$$

if and only if there exists a feasible schedule

$$\mathbf{C} = C(1), C(2), \dots, \quad \text{with} \quad C(t) = (c_1(t), \dots, c_{m_\mathrm{c}}(t)). \tag{15}$$

WSP is equivalent to PP when $m_\mathrm{c} = 1$. Similarly to PP, if a schedule for the WSP exists, then a cyclic schedule exists as well. Furthermore, the following schedulability conditions are known.

Proposition 3 (From [21])**.** Given an instance $I = \{m_\mathrm{c}, \{\alpha_i\}\}$ of WSP, if $\rho(I) > m_\mathrm{c}$ then $I \notin$ WSP , if $\rho(I) \leq 0.5 m_\mathrm{c}$ then $I \in$ WSP.

The results on WSP used next rely on special schedules of a particular form, defined as follows.

Definition 2 (Migration and perfect schedule, from [21], [22])**.** A *migrating* symbol is a symbol that is assigned to different channels at different time instants of a schedule. A schedule with no migrating symbols is called a *perfect schedule*.

An instance $I := \{m_\mathrm{c}, \{\alpha_i\}\}$ of WSP is accepted with a perfect schedule if and only if there exists a feasible schedule $\mathbf{C}$ such that

$$c_i(t_1) = c_k(t_2) \implies i = k, \tag{16}$$

for any $i, k \in \mathbb{I}_1^{m_\mathrm{c}}$ and $t_1, t_2 \in \mathbb{N}$; we denote this as:

$$I \in \text{WSP-perfect}. \tag{17}$$

By (16), nodes do not appear on different channels of the schedule.

In the following sections we construct tools for offline and online scheduling of systems of the form (1), (2), based on the results introduced above. Specifically, in Algorithm 1, using Theorems 1 and 2 we provide an offline solution for P2 in its most general form. Theorem 1 also proves, without loss of generality, that the schedule is periodic. Then, in Theorem 3, we link P2, in the special case where all nodes have access to the same number of communication channels, to WSP. This allows us to define a faster algorithm (Algorithm 2) to compute an offline schedule for P2 in these cases, based on Theorem 4. Finally, Algorithm 3 computes an online schedule which is resilient to stochastic packet loss, using Theorem 5 and the offline schedules defined before.

III. Offline Scheduling

In this section, we first prove that P2 is decidable, i.e., there is an algorithm that determines whether an instance is accepted by the problem [23], and we provide an exact solution algorithm and a heuristic to find a feasible schedule. Then, we show that in case that all the nodes share a fixed number of communication channels, the scheduling problem is equivalent to the WSP, we propose a new algorithm to solve this scheduling problem, and we refute a standing conjecture regarding perfect schedules in WSP [22].

A. Solution of P2

Consider sequence $\mathbf{C}$ as the schedule for P2, and define sequence

$$\mathbf{D} := D(1), D(2), \dots \tag{18}$$

where the ordered tuple $D(t)$ defined as

$$D(t) := (d_1(t), d_2(t), \dots, d_q(t)), \tag{19}$$

with $d_i(t) := t - \tau_i^{\mathbf{C}}(t)$. The *latest connection time* is defined as

$$\tau_i^{\mathbf{C}}(t) := \max\{t' \leq t : \ i \in C(t')\}, \tag{20}$$

where $t' := 0$ when the above set is empty.

Lemma 1. The schedule $\mathbf{C}$ is feasible for P2 if and only if $0 \leq d_i(t) \leq \alpha_i - 1$, $\forall i \in \mathbb{I}_1^q$, $\forall t > 0$.

Proof. If $\mathbf{C}$ is a feasible schedule, then $0 \leq d_i(t) \leq \alpha_i - 1$ for $\forall i \in \mathbb{I}_1^q$, $\forall t > 0$ by construction. The converse implies that node i is connected at least once every $(1 + \max_t d_i(t)) \leq \alpha_i$ time instants, which implies $\mathbf{C}$ is a feasible schedule. □

Theorem 1. *Consider the set of integers $\{\alpha_i\}$ defined in* (9)*. If P2 has a feasible schedule $\mathbf{C}$, then it also has a cyclic schedule whose period is no greater than $m = \prod_{i=1}^q \alpha_i$.*

Proof. We define $\mathbf{D}$ as in (19), so that $0 \leq d_i(t) \leq \alpha_i - 1$ holds by Lemma 1. Hence, each $d_i(t)$ can have no more than α_i different values. This implies $D(t)$ can have at most $m := \prod_{i=1}^q \alpha_i$ different values. Hence,

$$\exists\ t_1, t_2 : \ D(t_1) = D(t_2), \ m \leq t_1 < t_2 < 2m. \tag{21}$$

Now, consider the sequence $\mathbf{C}_\mathrm{r} := C(t_1), C(t_1+1), \dots, C(t_2-1)$ as the cyclic part of the cyclic schedule $\mathbf{C}_\mathrm{c}$ for P2, defined as $\mathbf{C}_c := \mathbf{C}_\mathrm{r}, \mathbf{C}_\mathrm{r}, \dots$. Define $\mathbf{D}_\mathrm{c}$ as in (18) for the new schedule $\mathbf{C}_\mathrm{c}$. One can conclude that

$$D_\mathrm{c}(\tau) \leq D(\tau + t_1 - 1), \qquad \forall \tau \in \mathbb{I}_1^{t_2 - t_1}, \tag{22}$$

since for any $i \in \mathbb{I}_1^q$ we have

$$d_{\mathrm{c}i}(\tau) = \tau - \tau_i^{\mathbf{C}_\mathrm{c}} = (\tau + t_1 - 1) - (\tau_i^{\mathbf{C}_\mathrm{c}} + t_1 - 1) \leq d_i(\tau + t_1 - 1).$$

Furthermore, $D(t_1) = D(t_2)$ implies $i \in \mathbf{C}_\mathrm{r}$ for $\forall i \in \mathbb{I}_1^q$. As a result, $d_{\mathrm{c}i}(t_2 - t_1) = d_i(t_2 - 1)$. This implies

$$d_{\mathrm{c}i}(k(t_2 - t_1) + \tau) = d_i(t_1 - 1 + \tau), \qquad k \in \mathbb{N}. \tag{23}$$

Since $d_i(t) \leq \alpha_i - 1$ holds for any $t > 0$, then $d_{\mathrm{c}i}(t) \leq \alpha_i - 1$ also holds for any $t > 0$. Inequality $d_{\mathrm{c}i}(t) \leq \alpha_i - 1$ implies that $\mathbf{C}_c$ is a feasible schedule by Lemma 1. □

Theorem 1 implies that a feasible schedule can always be searched for within the finite set of cyclic schedules of a length no greater than m. An important consequence of this theorem is the following.

Corollary 1. Schedulability of P2 can be decided in a finite time.

Proof. Using Theorem 1, the search space can be limited to a finite set and schedules can be finitely enumerated. □

Theorem 1 allows us to solve P2 by solving the following optimization problem, which searches for a feasible periodic schedule among all schedules of period T_r.

$$\min_{C(1),\dots,C(T_\mathrm{r}),T_\mathrm{r}} \quad T_\mathrm{r} \tag{24a}$$

$$\text{s.t.} \quad C(1),\dots,C(T_\mathrm{r}) \in \boldsymbol{\mathcal{C}}, \tag{24b}$$

$$T_\mathrm{r} \le \prod_{i=1}^{q} \alpha_i,\ T_\mathrm{r} \in \mathbb{N}, \tag{24c}$$

$$\sum_{k=t}^{t+\alpha_j-1} \eta_j(k) \ge 1, \qquad \forall j \in \mathbb{I}_1^q,\ \forall t \in \mathbb{I}_1^{T_\mathrm{r}}, \tag{24d}$$

$$\eta_j(k) = \begin{cases} 1 & \text{if } j \in C(k \bmod T_\mathrm{r}), \\ 0 & \text{otherwise.} \end{cases} \tag{24e}$$

Note that we define $C(0) := C(T_\mathrm{r})$ in (24e). Equation (24b) enforces the schedule elements to be chosen from the set of connection patterns $\boldsymbol{\mathcal{C}}$; (24c) limits the search space by giving an upper bound for the length of the periodic part, i.e., T_r; and (24d) ensures that label i appears at least once in each α_i successive elements of the schedule.

Note that the main challenge in Problem (24) is finding a feasible solution; minimization of T_r is a secondary goal since any solution of (24) provides a feasible schedule for P2. Unfortunately, (24e) is combinatorial in the number of nodes and connection patterns. In order to tackle this issue, we propose next a strategy to simplify the computation of a feasible schedule: a heuristic to solve P2 based on the assumption that the satisfaction of the constraints for a node i is a duty assigned to a single connection pattern C_j. To give some intuition on the assignment of connection patterns, we propose the following example.

Example 1. Consider a network with connection patterns: $C_1 := (1,2)$, $C_2 := (2,4)$, $C_3 := (3,4)$, $C_4 := (5)$ and safe time intervals $\alpha_1 = 10$, $\alpha_2 = 2$, $\alpha_3 = 10$, $\alpha_4 = 2$, $\alpha_5 = 100$.

Let us attempt a schedule using only C_1, C_3, C_4. In this case, one can see that the sequence $C_1, C_3, C_1, C_3, \dots$ is the only possible schedule satisfying the requirements of systems 2 and 4. There is however no space to connect system 5 within this schedule. Alternatively, one can utilize C_1, C_2, C_3, C_4 and design $C_2, C_1, C_2, C_3, C_2, C_4$ as the cyclic part of a feasible schedule.

In the second case, the duty of satisfying the constraints for systems 2 and 4 is assigned to C_2, while systems 1 and 3 are assigned to C_1 and C_3, respectively. As a consequence, C_2 must be scheduled every 2 steps, but C_1 and C_3 can be scheduled once every 10 steps. This allows one to make space for C_4. Borrowing the terminology of the PP, with the first choice C_1 and C_3 are symbols of density 0.5 and C_4 has density 0.01.

Example 1 shows how we assign duties to the connection patterns, and also how it can affect the schedulability. Let us represent the assignment of node i to the connection pattern C_j with a binary variable $\eta_{i,j}$ and—with a slight abuse of notation—the density of symbol C_j with $\hat{\rho}_j$. The proposed strategy is to decide the set of $\eta_{i,j}$ such that $\sum_j \hat{\rho}_j$ is minimized. This is performed by solving

$$\min_{\hat{\rho}_j,\ \eta_{i,j}} \quad \sum_{j=1}^{l} \hat{\rho}_j \tag{25a}$$

$$\text{s.t.} \quad \hat{\rho}_j \ge \frac{1}{\alpha_i}\eta_{i,j}, \qquad \forall j \in \mathbb{I}_1^l,\ \forall i \in C_j, \tag{25b}$$

$$\sum_{j:i\in C_j} \eta_{i,j} \ge 1, \qquad \forall i \in \mathbb{I}_1^q, \tag{25c}$$

$$\eta_{i,j} \in \{0,1\}, \qquad \forall i \in \mathbb{I}_1^q,\ \forall j \in \mathbb{I}_1^l. \tag{25d}$$

Algorithm 1 A heuristic scheduling for P2

1: Define $\hat{\alpha}_i$ as in (26) by solving the optimization problem (25)
2: find a schedule $\mathbf{C}_\mathrm{P}$ for instance $\{\hat{\alpha}_i\}$ of PP using (24) or any other suitable scheduling technique
3: define $C(t) := C_j$ given $C_\mathrm{P}(t) = j$
4: **return** $\mathbf{C} := C(1),\ C(2), \dots$

Constraint (25c) guarantees that every node i is connected by at least one connection pattern. Variables $\hat{\rho}_j$ bound the density of the resulting scheduling problem, where $1/\hat{\rho}_j$ is the maximum number of steps between two occurrences of connection pattern C_j in $\mathbf{C}$ that is sufficient to enforce $(x_i, \hat{x}_i, u_i) \in \mathcal{S}_{i,\infty}$. If $\hat{\rho}_j = 0$, then connection pattern j is not used. Without loss of generality, assume that the solution to (25) returns l distinct connection patterns with $\hat{\rho} > 0$, i.e., $\hat{\rho}_1, \dots, \hat{\rho}_l > 0$ and define

$$\hat{\alpha}_i := \frac{1}{\hat{\rho}_i}, \qquad \forall i \in \mathbb{I}_1^l. \tag{26}$$

Theorem 2. *$\{\hat{\alpha}_i\} \in PP \implies \{\boldsymbol{\mathcal{C}}, \{\alpha_i\}\} \in P2$.*

Proof. Consider schedule $\mathbf{C}_\mathrm{P}$ which is feasible for the instance $\{\hat{\alpha}_i\}$ of PP. Define the schedule $\mathbf{C}$ by $C(t) := C_j$ given $C_\mathrm{P}(t) = j$ for any j. By the statement of PP, $C_\mathrm{P}(t) = j$ once at least in every $\hat{\alpha}_j$ successive time instants. By (25), for all i there exists C_j such that $i \in C_j$ and $\alpha_i \ge \hat{\alpha}_i$. Hence, $\mathbf{C}$ is a feasible schedule for P2. □

Using Theorem 2, we summarize in Algorithm 1 how to find a feasible schedule for P2. As shown by the following example, the converse of Theorem 2 does not hold in general, i.e., if Algorithm 1 does not find a schedule a feasible schedule may still exist for P2.

Example 2. Consider five nodes with $\alpha_1 = \alpha_3 = 3$, $\alpha_2 = \alpha_4 = \alpha_5 = 5$ and $\boldsymbol{\mathcal{C}} := \{C_1, C_2, C_3, C_4\}$ where

$$C_1 = (1,2),\ C_2 = 3,\ C_3 = 4,\ C_4 = (1,5). \tag{27}$$

Using (25), one obtains $\hat{\alpha}_1 = \hat{\alpha}_3 = 5$, and $\hat{\alpha}_2 = \hat{\alpha}_4 = 3$. There is no feasible schedule for this problem considering the assigned density function $\hat{\rho}(\{3,3,5,5\}) = \frac{16}{15}$, see Proposition 1. However, one can verify that the following schedule is feasible

$$\mathbf{C}_c := \mathbf{C}_\mathrm{r}, \mathbf{C}_\mathrm{r}, \dots, \qquad \mathbf{C}_\mathrm{r} := C_1, C_2, C_4, C_2, C_3. \tag{28}$$

B. Solution of P2 in the m_c-channel case

In the previous subsection, $\boldsymbol{\mathcal{C}}$ was an arbitrary set of connection patterns. Assume now that the set $\boldsymbol{\mathcal{C}}$ is

$$\boldsymbol{\mathcal{C}} := \{C :\ C \subseteq \mathbb{I}_1^q,\ |C| = m_\mathrm{c}\}, \tag{29}$$

i.e., the set of all subsets of $\mathbb{I}_1^q$ with cardinality m_c. This is a special case of P2 where any combination of m_c nodes can be connected at the same time. One application of such case is for instance when the connection patterns model a multi-channel star communication topology between a set of nodes and a central controller. This class of problems is easily mapped to the class of WSP:

Theorem 3. *When $\boldsymbol{\mathcal{C}}$ is as in* (29)*, then*

$$\{\boldsymbol{\mathcal{C}}, \{\alpha_i\}\} \in P2 \iff \{m_c, \{\alpha_i\}\} \in WSP. \tag{30}$$

Proof. By definition, any schedule solving P2 must satisfy $i \in C(t) \Rightarrow i \in C(t+\tau)$ with $\tau \le \alpha_i$ for all i, t. Provided that $|C(t)| = m_\mathrm{c}$ for all t, this also defines a schedule solving WSP. □

We exploit this result to solve P2 indirectly by solving WSP. To that end, we propose a heuristic which replaces WSP with a PP relying on modified safe time intervals defined as

$$\tilde{\alpha}_i := m_\mathrm{c}\alpha_i, \qquad \forall i \in \mathbb{I}_1^q. \tag{31}$$

Theorem 4. $\{\tilde{\alpha}_i\} \in PP \implies \{m_c, \{\alpha_i\}\} \in WSP$.

Proof. Given a feasible schedule $\mathbf{C}_P$ for PP, $C_P(t) = i$ at least once every $\tilde{\alpha}_i = m_c\alpha_i$ successive time instants. Define schedule $\mathbf{C}$ using

$$C(t) := (C_P(m_c(t-1)+1), \ldots, C_P(m_c t)). \tag{32}$$

In this schedule, $i \in C(t)$ at least once every α_i successive time instants. This implies that $\mathbf{C}$ is a feasible schedule for WSP. □

Theorem 4 can be used to find a feasible schedule for WSP using a feasible schedule for PP. However, the converse does not hold: if this method does not find a feasible schedule, a feasible schedule for WSP may still exist. Nevertheless, Lemma 2 provides a sufficient condition to determine that there exists no feasible schedule for WSP. Without loss of generality, assume $\alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_q$ and define

$$\zeta_i := \begin{cases} m_c\alpha_i & i \leq m_c \\ m_c\alpha_i + (m_c - 1) & i > m_c \end{cases}. \tag{33}$$

Lemma 2. $\{\zeta_i\} \notin$ PP $\implies \{m_c, \{\alpha_i\}\} \notin$ WSP.

Proof. We proceed by contradiction. Assume $\{m_c, \{\alpha_i\}\} \in$ WSP with a corresponding feasible schedule $\mathbf{C}$, while $\{\zeta_i\} \notin$ PP. Without loss of generality, assume that the labels $i \in \mathbb{I}_1^{m_c}$ are arranged in $C(t)$ so as to satisfy

$$i \in C(t) \implies c_i(t) = i, \tag{34}$$

while labels $i \in \mathbb{I}_{m_c+1}^{q}$ are arranged in an arbitrary order. Construct a schedule $\mathbf{C}_P$ as

$$\mathbf{C}_P = c_1(1), \ldots, c_{m_c}(1), \ldots, c_1(t), \ldots, c_{m_c}(t), \ldots. \tag{35}$$

If $\{\zeta_i\} \notin$ PP, then there exists a $t_0 > 0$ and an $i \in \mathbb{I}_1^q$ such that the sequence $(C_P(t_0), \ldots, C_P(t_0 + \zeta_i - 1))$ does not contain label i, where $C_P(t_0)$ is the entry $c_k(t)$ of $\mathbf{C}$. A pair of integers (t, k) can be found such that $t \geq 0$, $k \in \mathbb{I}_1^{m_c}$, and $t_0 = m_c(t-1) + k$.
Consider the case $i \in \mathbb{I}_1^{m_c}$. By (33) we have $\zeta_i = m_c\alpha_i$, such that the sequence $C_P(t_0), \ldots, C_P(t_0 + \zeta_i - 1)$ contains exactly α_i vectors $C(t), \ldots, C(t + \alpha_i - 1)$ if $k = 1$, or spans $\alpha_i + 1$ vectors $C(t), \ldots, C(t + \alpha_i)$ if $k > 1$. Hence, if $k \leq i$, by (34) one can conclude $i \notin C(t), \ldots, C(t + \alpha_i - 1)$, while if $k > i$, by (34), $i \notin C(t+1), \ldots, C(t + \alpha_i)$. In both cases $\mathbf{C}$ is not feasible, which contradicts our assumption.
Consider the case $i \in \mathbb{I}_{m_c}^{q}$. By (33) we have $\zeta_i = m_c(\alpha_i + 1) - 1$, such that the sequence $C_P(t_0), \ldots, C_P(t_0 + \zeta_i - 1)$ contains α_i subsequent vectors of the schedule $\mathbf{C}$ that do not contain label i. This implies that $\mathbf{C}$ is not a feasible schedule, which is a contradiction. □

By Theorem 4 one can find a schedule for an instance of WSP using a schedule for an instance of PP. A common approach proposed in the literature consists in restricting the search to perfect schedules. We prove next that our heuristic returns a feasible schedule if a perfect schedule exists; in addition, it can also return non-perfect schedules. As we will prove, cases exist when the WSP does not admit a perfect schedule while it does admit a non-perfect one. We will provide such example and show that our heuristic is able to solve it.

The following lemma provides a sufficient condition to exclude existence of a perfect schedule. An immediate corollary of this lemma and of Theorem 4 is that the heuristic in Theorem 4 can schedule all WSP instances that admit a perfect schedule.

Lemma 3. $\{\tilde{\alpha}_i\} \notin$ PP $\implies \{m_c, \{\alpha_i\}\} \notin$ WSP-perfect.

Proof. Assume $\mathbf{C}$ is a perfect schedule for WSP. Then, $c_i(t) = c_j(t') = k$ implies $i = j$ for all $i, j \in \mathbb{I}_1^{m_c}$ and $k \in \mathbb{I}_1^q$. Consider the sequence $\mathbf{C}_P$ as in (35) where $t \geq 1$. Since $\mathbf{C}$ is a perfect schedule, $c_i(t) = c_i(t') = k$ for every $k \in \mathbb{I}_1^q$. Furthermore, $|t - t'| \leq \alpha_k$ implies $C_P(t_1) = C_P(t_2) = k$ with $t_1 = m_c(t-1) + i$, $t_2 = m_c(t'-1) + i$. Hence, $|t_1 - t_2| = m_c|t - t'| \leq m_c\alpha_k$. Consequently, $\mathbf{C}_P$ is a feasible schedule for PP. □

Algorithm 2 A heuristic scheduling for WSP

1: Define $\tilde{\alpha}_i$ as in (31)
2: find the feasible schedule $\mathbf{C}_P$ for instance $\{\tilde{\alpha}_i\}$ of PP
3: define $C(t) := (C_P(m_c(t-1)+1), \ldots, C_P(m_c t))$
4: **return** $\mathbf{C} := C(1),\ C(2), \ldots$

The following example shows that the converse of Theorem 4 does not hold in general, i.e., $\exists\ \{m_c, \{\alpha_i\}\} \in$ WSP while $\{\tilde{\alpha}_i\} \notin$ PP. This also indicates the importance of non-perfect schedules.

Example 3 (Converse of Theorem 4)**.** Consider problem instance

$$\{m_c, \{\alpha_i\}\} = \{2, \{2, 3, 3, 4, 5, 5, 10\}\}. \tag{36}$$

While $\{\tilde{\alpha}_i\} \notin$ PP, a schedule with the cyclic part

$$\begin{aligned} \mathbf{C}_r = &C_1, C_2, C_3, C_4, C_5, C_6, C_1, C_7, C_8, C_9, \\ &C_5, C_9, C_3, C_{10}, C_1, C_6, C_5, C_{11}, C_8, C_2\ , \end{aligned} \tag{37}$$

is feasible for WSP where

$$\begin{aligned} &C_1 = (1,2), \quad C_2 = (3,4), \ C_3 = (1,6), \quad C_4 = (2,5), \\ &C_5 = (1,3), \quad C_6 = (4,7), \ C_7 = (3,6), \quad C_8 = (1,5), \\ &C_9 = (2,4), \quad C_{10} = (3,5), \ C_{11} = (2,6). \end{aligned} \tag{38}$$

Remark 1. Since $\{\tilde{\alpha}_i\} \notin$ PP in Example 3, $\{m_c, \{\alpha_i\}\} \notin$ WSP-perfect by Lemma 3. However, $\{m_c, \{\alpha_i\}\} \in$ WSP which provides a negative answer to an open problem in the scheduling community, i.e., whether all feasible instances of WSP admit perfect schedules too, see [22].

The next example provides a case in which $\{\tilde{\alpha}_i\} \in$ PP while $\{m_c, \{\alpha_i\}\} \notin$ WSP-perfect. This implies that the proposed heuristic for WSP can return feasible schedules for instances in which there is no perfect schedule.

Example 4. Consider the problem instance

$$\{m_c, \{\alpha_i\}\} = \{2, \{2, 3, 4, 5, 5, 5, 7, 14\}\}. \tag{39}$$

In order to find a perfect schedule, one can first compute all possible allocations of nodes to two channels and verify that $\{m_c, \{\alpha_i\}\} \notin$ WSP-perfect. However, a schedule with the following cyclic part is feasible for instance $\{\tilde{\alpha}_i\}$ of PP

$$2, 3, 4, 1, 7, 6, 2, 1, 5, 3, 2, 1, 4, 3, 6, 1, 2, 5, 7, 1, 4, 3, 2, 1, 6, 8, 5, 1.$$

This schedule can be transformed into a feasible schedule for WSP with the cyclic part

$$\begin{aligned} \mathbf{C}_r = &(2,3), (4,1), (7,6), (2,1), (5,3), (2,1), (4,3), \\ &(6,1), (2,5), (7,1), (4,3), (2,1), (6,8), (5,1). \end{aligned} \tag{40}$$

Algorithm 2 computes (possibly non-perfect) schedules for WSP, by checking whether $\{\tilde{\alpha}_i\}$ is accepted by PP or not. Since:

- $\{\tilde{\alpha}_i\} \in$ PP $\implies \{m_c, \{\alpha_i\}\} \in$ WSP,
- $\{\tilde{\alpha}_i\} \notin$ PP $\implies \{m_c, \{\alpha_i\}\} \notin$ WSP-perfect,
- $\exists\ \{m_c, \{\alpha_i\}\}:\ \{\tilde{\alpha}_i\} \in$ PP, $\{m_c, \{\alpha_i\}\} \notin$ WSP-perfect,

Algorithm 2 outperforms the current heuristics in the literature in the sense that it accepts more instances of WSP.

While $\rho(I) \leq 0.5m_c$ is a sufficient condition for schedulability of WSP [21], we provide alternative, less restrictive sufficient conditions in the following theorem.

Proposition 4. Given an instance $I = \{m_c, \{\alpha_i\}\}$ of WSP,

1) if $\rho(I) \leq 0.75 m_{\mathrm{c}}$ then $I \in \mathrm{WSP}$,
2) if $\rho(I) \leq \frac{5}{6} m_{\mathrm{c}}$ and I has only three symbols then $I \in \mathrm{WSP}$.

Proof. We rely on (31) and Theorem 4 to convert WSP into PP, so that Proposition 1 delivers the desired result. □

IV. Online Scheduling

The schedules proposed in Section III are computed offline, i.e., solely based on the information available *a priori*. Note that any offline schedule is conservative, since it is designed to cope with all admissible disturbances and it cannot exploit available knowledge of current state. Moreover, packet losses require at least a basic adaptation of the offline schedule. In an online schedule, available knowledge of past and current states can be exploited to refine the schedule based on the available information. In the following, we design a technique to adapt online a schedule computed offline to exploit the available state information. We first consider the case of no packet losses; we then show how the extension to the case of packet losses can be done with minimal modifications.

A. Online Scheduling without Packet Loss

In this subsection we show, under the assumption of no packet loss, how the schedule can be optimized online. Our strategy is to start with a feasible offline schedule, which we call the *baseline schedule*. Such schedule is then shifted based on estimates of the safe time intervals, which are built upon the available knowledge of past and current states. In fact, while in equation (9) the safe time interval is defined as the solution of a reachability problem with $\mathcal{S}_{i,\infty}$ as the initial set, the scheduler may have a better set-valued estimate of the current state of each node than the whole $\mathcal{S}_{i,\infty}$. This estimate, which we call $\mathcal{O}_i$, can in general be any set with the following properties, for all $t \geq 0$:

$$(x_i(\tau), \hat{x}_i(\tau), u_i(\tau)) \in \mathcal{O}_i(\tau), \qquad \forall \tau \in \mathbb{I}_0^t, \tag{41a}$$

$$\mathcal{O}_i(t) \subseteq \mathcal{S}_{i,\infty}, \qquad \text{if } \delta_i = 1, \tag{41b}$$

$$\mathcal{O}_i(t) \subseteq \mathrm{Reach}_1^{\hat{F}_i}(\mathcal{O}_i(t-1)). \tag{41c}$$

Based on set $\mathcal{O}_i(t)$, we can compute an estimate of the safe time interval. Let us define this estimate, function of t, as follows:

$$\gamma_i^x(t) := \max \left\{ t' : \mathrm{Reach}_{t'}^{\hat{F}_i}(\mathcal{O}_i(t)) \subseteq \mathcal{S}_{i,\infty} \right\}. \tag{42}$$

Equations (9) and (42) imply that, for any feasible schedule $\mathbf{C}$,

$$\gamma_i^x(t) \geq \alpha_i - (t - \tau_i^{\mathbf{C}}(t)), \qquad \forall i \in \mathbb{I}_1^q, \ \forall t > 0. \tag{43}$$

For any arbitrarily defined schedule $\mathbf{C}_{\mathrm{o}}$, we define

$$\gamma_i^{\mathbf{C}_{\mathrm{o}}}(t) := \min\{t' \geq t : \ i \in C_{\mathrm{o}}(t')\} - t, \tag{44}$$

which measures how long node i will have to wait, at time t, before being connected. Using (42) and (44), we specify a condition under which the schedule $\mathbf{C}_{\mathrm{o}}$ is feasible.

Definition 3 (Online feasible schedule)**.** A schedule $\mathbf{C}_{\mathrm{o}}$ is online feasible if the *safety residuals* $\mathbf{r}(\mathbf{C}_{\mathrm{o}}, t)$ defined as

$$r_i(\mathbf{C}_{\mathrm{o}}, t) := \gamma_i^x(t) - \gamma_i^{\mathbf{C}_{\mathrm{o}}}(t) \geq 0, \qquad \forall i \in \mathbb{I}_1^q, \tag{45}$$

with $\gamma_i^x(t)$ defined in (42) and $\gamma_i^{\mathbf{C}_{\mathrm{o}}}(t)$ defined in (44).

In the job scheduling literature (e.g., [24]), the quantities $\gamma_i^{\mathbf{C}}$ correspond to the *completion times* of job i, the quantities γ_i^x are the *deadlines*, and the quantity $\gamma_i^{\mathbf{C}} - \gamma_i^x = -r_i$ is the *job lateness*. A schedule for q jobs with deadlines is feasible provided that the maximum lateness (safety residual) is non-positive.

In the following, we formulate an optimization problem to find a recursively feasible online schedule using safety residuals and shifts of the baseline schedule. Given a cycle $\mathbf{C}_{\mathrm{r}} := C(1), \ldots, C(T_{\mathrm{r}})$ of the baseline schedule, let

$$R(\mathbf{C}_{\mathrm{r}}, j) := C(j), \ldots, C(T_{\mathrm{r}}), C(1), \ldots, C(j-1) \tag{46}$$

be a rotation of the sequence $\mathbf{C}_{\mathrm{r}}$ with $j \in \mathbb{I}_1^{T_{\mathrm{r}}}$.

We define the online schedule

$$C^*(t) := C(j_t^*), \tag{47}$$

as the one maximizing the minimum safety residual by solving

$$j_t^* := \arg\max_j \ \min_i \ r_i(R(\mathbf{C}_{\mathrm{r}}, j), t). \tag{48}$$

Proposition 5. Assume that the cyclic baseline schedule $\mathbf{C}$ is feasible for P2. Then, the online schedule $\mathbf{C}^*$ is feasible for P2.

Proof. At time $t = 1$, the baseline schedule is feasible, i.e., $\min_i r_i(R(\mathbf{C}_{\mathrm{r}}, 1), 1) \geq 0$. As a result, $\min_i r_i(R(\mathbf{C}_{\mathrm{r}}, j_1^*), 1) \geq 0$ by construction and schedule $\tilde{\mathbf{C}}$, defined as

$$\tilde{\mathbf{C}} := C(j_1^*), \ldots, C(T_{\mathrm{r}}), \mathbf{C}_{\mathrm{r}}, \mathbf{C}_{\mathrm{r}}, \ldots \ , \tag{49}$$

is a feasible schedule. Since $C^*(1) = C(j_1^*)$, $\min_i r_i(R(\mathbf{C}_{\mathrm{r}}, j_1^* + 1), 2) \geq 0$, which implies $\min_i r_i(R(\mathbf{C}_{\mathrm{r}}, j_2^*), 2) \geq 0$. Consequently,

$$\tilde{\mathbf{C}} := C(j_1^*), C(j_2^*), C(j_2^*+1), \ldots, C(T_{\mathrm{r}}), \mathbf{C}_{\mathrm{r}}, \ldots \ , \tag{50}$$

is a feasible schedule. This argument can be used recursively which implies $\mathbf{C}^*$ is a feasible schedule. □

Remark 2. The schedule (47) maximizes the minimum residual, as shown in (48). That is, the communication is scheduled for the system which is closest to exit $\mathcal{S}_{i,\infty}$. Clearly, any function of the residuals could be used. For example, the residuals could be weighted, thus reflecting the priority given to the constraints to be satisfied.

B. Robustness Against Packet Loss

In this subsection, we drop the assumption of no packet loss in the communication link and we consider a communication protocol which has packet delivery acknowledgment. We provide the necessary and sufficient conditions for the existence of robust schedules in the presence of packet losses. Then, using these and the results in Section IV-A, we provide an algorithm to compute an online schedule that is robust against packet losses. Let us consider the stochastic binary variable $\nu(t) \in \{0, 1\}$, with $\nu(t) = 1$ indicating that the packet sent at time t was lost. This binary variable is known to the scheduler, since we assume an acknowledgment-based protocol.

Assumption 1. No more than $n_{l,i}$ packets are lost in any α_i consecutive steps, i.e.,

$$\sum_{j=t}^{t+\alpha_i-1} \nu(j) \leq n_{\mathrm{l,i}}, \qquad \forall \ i \in \mathbb{I}_1^q, \ \forall t \geq 0. \tag{51}$$

We observe that (51) defines q different inequalities which must be satisfied at the same time. Additionally, we assume that when a packet is lost, the whole information exchanged at time t is lost. Note that Assumption 1 defines a stochastic packet loss model with bounded support. In case Assumption 1 does not hold, robust invariance cannot be guaranteed. This is a fundamental issue which cannot be circumvented. However, if (51) holds with probability less than 1, one should be able to prove a weaker form of robust invariance, which only holds in probabilistic terms.

Problem 3 (P3)**.** Given the set of q nodes, each described by (1), an admissible set $\mathcal{A} := \mathcal{A}_1 \times \ldots \times \mathcal{A}_q$, and the set $\boldsymbol{\mathcal{C}}$ of connection

patterns (3), determine if there exists an infinite sequence over the elements of $\boldsymbol{\mathcal{C}}$ such that, if Assumption 1 holds, for $t > 0$ we have

$$z_i(t) \in \mathcal{S}_{i,\infty}, \qquad \forall z_i(0) \in \mathcal{S}_{i,\infty}, \qquad v_i(t) \in \mathcal{V}_i, \qquad i \in \mathbb{I}_1^q.$$

A schedule solving P3 is any sequence of C such that every node i is connected at least once every α_i steps in the presence of packet losses satisfying Assumption 1. Instance $I := \{\boldsymbol{\mathcal{C}}, \{\alpha_i\}, \{n_{\mathrm{l,i}}\}\}$ is accepted, i.e., $I \in$ P3, if and only if a schedule $\mathbf{C}$ exists that satisfies the scheduling requirements. Given a feasible baseline schedule $\mathbf{C}$, to compensate the effects of packet losses, we define a *shifted schedule* $\bar{\mathbf{C}}$ as

$$\bar{C}(t) := C\Big(t - \sum_{j=0}^{t-1} \nu(j)\Big). \tag{52}$$

We define the maximum time between two successive connections of node i, under schedule $\mathbf{C}$ as

$$T_i := \big(1 + \max_t \; t - \tau_i^{\mathbf{C}}(t)\big), \tag{53}$$

where the latest connection time $\tau_i^{\mathbf{C}}(t)$ is defined in (20). Feasibility of the baseline schedule implies $T_i \leq \alpha_i$, for all i.

Assumption 1 can be used to provide a sufficient condition for the shifted schedule $\bar{\mathbf{C}}$ to be feasible under packet losses.

Proposition 6 (From [15])**.** Let Assumption 1 be verified. Schedule $\bar{\mathbf{C}}$ defined in (52) is feasible for P3 if and only if

$$\alpha_i - T_i \geq n_{l,i}, \qquad \forall i \in \mathbb{I}_1^q. \tag{54}$$

Proof. In the error-free schedule $\mathbf{C}$, two consecutive appearances of a symbol i are at most T_i steps apart. In the schedule $\bar{\mathbf{C}}$, during α_i steps at most $n_{l,i}$ retransmissions take place. Hence, if $\alpha_i - T_i \geq n_{l,i}$, two consecutive occurrences of symbol i are never spaced more than α_i steps, ensuring feasibility of the schedule. □

In the sequel, we provide necessary and sufficient conditions for the existence of a baseline schedule which is robust against packet losses. To that end, we define a new set of safe time intervals as

$$\big\{\beta_i : \beta_i = \alpha_i - n_{l,i}, \; \forall i \in \mathbb{I}_1^q\big\}. \tag{55}$$

Theorem 5. *Assume that the communication network satisfies Assumption 1. Then, β_i defined in* (55) *yields*

$$\{\boldsymbol{\mathcal{C}}, \{\alpha_i\}, \{n_{\mathrm{l,i}}\}\} \in \text{P3} \qquad \Leftrightarrow \qquad \{\boldsymbol{\mathcal{C}}, \{\beta_i\}\} \in \text{P2}.$$

Proof. We first prove $\{\boldsymbol{\mathcal{C}}, \{\alpha_i\}, \{n_{\mathrm{l,i}}\}\} \in$ P3 $\Rightarrow$ $\{\boldsymbol{\mathcal{C}}, \{\beta_i\}\} \in$ P2. Assume that there exists a feasible schedule for instance $\{\boldsymbol{\mathcal{C}}, \{\alpha_i\}, \{n_{\mathrm{l,i}}\}\}$ of P3 while it is not feasible for instance $\{\boldsymbol{\mathcal{C}}, \{\beta_i\}\}$ of P2. This implies

$$\exists\, i, t > 0 : t - \tau_i^{\mathbf{C}}(t) \geq \beta_i + 1 = \alpha_i - n_{l,i} + 1, \tag{56}$$

where the latest connection time $\tau_i^{\mathbf{C}}(t)$ is defined in (20). Assume $n_{l,i}$ consecutive packets are lost starting from time $t+1$, such that $\tau_i^{\mathbf{C}}(t + n_{l,i}) = \tau_i^{\mathbf{C}}(t)$. This implies $\big(t + n_{l,i}\big) - \tau_i^{\mathbf{C}}(t + n_{l,i}) \geq \alpha_i + 1$, such that node i did not receive any packet for α_i consecutive steps, i.e., $\{\boldsymbol{\mathcal{C}}, \{\alpha_i\}, \{n_{\mathrm{l,i}}\}\} \notin$ P3.
In order to prove $\{\boldsymbol{\mathcal{C}}, \{\beta_i\}\} \in \text{P2} \implies \{\boldsymbol{\mathcal{C}}, \{\alpha_i\}, \{n_{\mathrm{l,i}}\}\} \in \text{P3}$, consider feasible schedule $\mathbf{C}$ for instance $\{\boldsymbol{\mathcal{C}}, \{\beta_i\}\}$ of P2. Each packet loss causes one time step delay in data communication for node i, see (52), and since these packet losses can at most cause $n_{l,i}$ time step delays between two connection times, node i would be connected at least once during each $\beta_i + n_{l,i} = \alpha_i$ time steps. This implies $\bar{\mathbf{C}}$ defined in (52) is a feasible schedule for instance $\{\boldsymbol{\mathcal{C}}, \{\alpha_i\}, \{n_{\mathrm{l,i}}\}\}$ of P3. □

Algorithm 3 Robust online scheduling for P2

1: Define β_i using (55)
2: find a schedule $\mathbf{C}_{\mathrm{P}}$ for instance $\{\beta_i\}$ of PP
3: define $C(t) := C_j$ when $C_{\mathrm{P}}(t) = j, \; \forall j$
4: **for all** t **do**
5: find $\bar{C}^*(t)$ by solving (59)
6: **end for**

Theorem 5 implies that P3 can be cast in the framework of P2 by using equation (55) to define $\{\beta_i\}$ based on $\{\alpha_i\}$ and $\{n_{\mathrm{l,i}}\}$. Algorithm 3, returns an online robust feasible schedule for P3.

Since the shifted schedule $\bar{\mathbf{C}}$ provides a feasible robust schedule against packet losses, one can use the online scheduling method proposed in the previous subsection to improve safety of this robust schedule. To that end, we define the number of packet losses that can occur before node i receives a measurement from time t as

$$n_i^{\mathbf{C}}(t) := \min_{t',n} n \;:\; t' \geq t, \; i \in C(t'), \; \sum_{j=t}^{t'} \nu(j) \leq n. \tag{57}$$

Definition 4. A schedule $\mathbf{C}$ is robust online feasible if the *robust safety residuals* $\bar{\mathbf{r}}(\mathbf{C}, t)$ defined as

$$\bar{r}_i(\mathbf{C}, t) := \gamma_i^x(t) - \gamma_i^{\mathbf{C}}(t) - n_i^{\mathbf{C}}(t), \; \forall i \in \mathbb{I}_1^q, \tag{58}$$

are non-negative for all t, with $\gamma_i^x(t)$ defined in (42), $\gamma_i^{\mathbf{C}}(t)$ defined in (44), and $n_i^{\mathbf{C}}(t)$ defined in (57).

Similarly to the case with no packet loss, we compute the online schedule $\bar{C}^*(t) := C(\bar{j}_t^*)$ by solving

$$\bar{j}_t^* := \arg\max_j \; \min_i \; \bar{r}_i(R(\mathbf{C}_{\mathrm{r}}, j), t). \tag{59}$$

Proposition 7. Assume that the baseline schedule $\bar{\mathbf{C}}$ is feasible for P3. Then, the online schedule $\bar{C}^*(t)$ is feasible for all t.

Proof. Follows mutatis mutandis from Propositions 5 and 6. □

V. Numerical results

In this section, we evaluate the performance of the proposed algorithms 1 and 2, respectively.

Consider 1000 networks with a random number of nodes (2 to 11), random safe time intervals (2 to 21), and random connection patterns (1 to 11). These random instances are used for evaluating the three following implementations:

- M_1: solve optimization problem (24);
- M_2^{A1}: use Algorithm 1 in combination with optimization problem (24) to solve PP;
- M_3^{A1}: use Algorithm 1 in combination with the double-integer method proposed in [12] to solve PP.

We have used Gurobi to solve the integer problems and provided a summary of the results in Table I. M_1 is exact and returns false positives nor false negatives. Although M_2^{A1} and M_3^{A1} do not return any false positive, they might return false negatives. Furthermore, a false negative answer in M_2^{A1} implies the same for M_3^{A1} since the latter uses a heuristic to solve PP while the former finds a schedule for PP whenever it exists. Note also that M_2^{A1} and M_3^{A1} do not necessarily return a solution with the minimum period length. To limit the computation-time, we had to halt the execution of M_1 and M_2^{A1} when no schedule of period ≤ 70 was found. We labeled *undecided* the instances for which these two methods were halted.

TABLE I: Comparison of M_1, M_2^{A1}, and M_3^{A1}

	M_1	M_2^{A1}	M_3^{A1}
accepted instances	887	872	853
average time (sec)	1.7915	1.3119	0.5143
undecided instances	102	32	0
average time (sec)	61.0570	48.9156	0
rejected instances	11	96	147
average time (sec)	53.7730	1.5442	0.5333

Although M_3^{A1} might result in a few false negatives, Table I indicates that its average computation time is significantly lower than the corresponding average computation times of M_1 and M_2^{A1}.

Next, we evaluate the proposed heuristic in Algorithm 2 to find a feasible schedule for P2 when $\boldsymbol{\mathcal{C}}$ is defined as in (29). We have generated 1000 networks with a random number of nodes (from2 to 11), random safe time intervals (from 2 to 21), and the necessary number of channels needed for schedulability, i.e., $m_{\mathrm{c}} = \lceil \sum_i \frac{1}{\alpha_i} \rceil$. These random instances are used for evaluating the three following implementations:

- M_1: solve optimization problem (24);
- M_2^{A2}: use Algorithm 2 in combination with optimization problem (24) to solve PP;
- M_3^{A2}: use Algorithm 2 in combination with optimization problem (24) to solve PP;

Once again, we have halted the solver in M_1 and M_2^{A2} cases if no schedule of length ≤ 70 was found. The results are reported in Table II.

TABLE II: Comparison of M_1, M_2^{A2}, and M_3^{A2}

	M_1	M_2^{A2}	M_3^{A2}
accepted instances	984	980	909
average time (sec)	5.0353	5.5204	1.3043e-04
undecided instances	16	20	0
average time (sec)	267.1501	139.8464	0
rejected instances	0	0	91
average time (sec)	0	0	1.0384e-04

Although M_3^{A2} might result in a few false negatives, Table II indicates that its average computation time is drastically lower than the corresponding average computation times of M_1 and M_2^{A2}.

VI. Conclusions

In this paper we proposed scheduling techniques to guarantee invariance for NCS consisting of uncertain constrained systems and multiple communication channels. These techniques were used to design offline schedules and online ones. The online schedules are recursively feasible in presence of system uncertainties and packet losses.

Future work will consider extending our framework to cover NCS with coupled dynamics and the case of nodes modeled by the period dwell-time switching signal, as in [25].

References

[1] W. Zhang, M. S. Branicky, and S. M. Phillips, "Stability of networked control systems," IEEE Control Systems, vol. 21, no. 1, pp. 84–99, 2001.

[2] D. P. Bertsekas and I. B. Rhodes, "On the minimax reachability of target sets and target tubes," Automatica, vol. 7, no. 2, pp. 233–247, 1971.

[3] D. P. Bertsekas, "Infinite-time reachability of state-space regions by using feedback control," IEEE Trans. Autom. Control, vol. 17, pp. 604–613, 1972.

[4] S. Prajna and A. Jadbabaie, "Safety verification of hybrid systems using barrier certificates," in International Workshop on Hybrid Systems: Computation and Control. Springer, 2004, pp. 477–492.

[5] A. Bouajjani, J. Esparza, and O. Maler, "Reachability analysis of pushdown automata: Application to model-checking," in International Conference on Concurrency Theory. Springer, 1997, pp. 135–150.

[6] J. H. Gillula, H. Huang, M. P. Vitus, and C. J. Tomlin, "Design of guaranteed safe maneuvers using reachable sets: Autonomous quadrotor aerobatics in theory and practice," in Robotics and Automation (ICRA), 2010 IEEE international conference on. IEEE, 2010, pp. 1649–1654.

[7] J. B. Rawlings, D. Bonné, J. B. Jorgensen, A. N. Venkat, and S. B. Jorgensen, "Unreachable setpoints in model predictive control," IEEE Transactions on Automatic Control, vol. 53, no. 9, pp. 2209–2215, 2008.

[8] A. Gupta and P. Falcone, "Full-complexity characterization of control-invariant domains for systems with uncertain parameter dependence," IEEE Control Systems Letters, vol. 3, no. 1, pp. 19–24, 2019.

[9] Y. Zhu, Z. Zhong, W. X. Zheng, and D. Zhou, "Hmm-based $\mathcal{H}_\infty$ filtering for discrete-time markov jump lpv systems over unreliable communication channels," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 48, no. 12, pp. 2035–2046, 2017.

[10] L. Zhang, Y. Zhu, Z. Ning, and X. Yin, "Resilient estimation for networked systems with variable communication capability," IEEE Transactions on Automatic Control, vol. 61, no. 12, pp. 4150–4156, 2016.

[11] X.-M. Zhang, Q.-L. Han, X. Ge, D. Ding, L. Ding, D. Yue, and C. Peng, "Networked control systems: a survey of trends and techniques," IEEE/CAA Journal of Automatica Sinica, vol. 7, no. 1, pp. 1–17, 2019.

[12] M. Chan and F. Chin, "General schedulers for the pinwheel problem based on double-integer reduction," IEEE Transactions on Computers, vol. 41, no. 6, pp. 755–768, 1992. [Online]. Available: http://dx.doi.org/10.1109/12.144627

[13] R. Holte, A. Mok, L. Rosier, I. Tulchinsky, and D. Varvel, "The pinwheel: A real-time scheduling problem," in Proceedings of the Hawaii International Conference on System Science, 1989, pp. 693–702.

[14] A. Colombo, M. Bahraini, and P. Falcone, "Measurement scheduling for control invariance in networked control systems," in 2018 IEEE Conference on Decision and Control (CDC). IEEE, 2018, pp. 3361–3366.

[15] M. Bahraini, M. Zanon, A. Colombo, and P. Falcone, "Receding-horizon robust online communication scheduling for constrained networked control systems," in 2019 18th European Control Conference (ECC). IEEE, 2019, pp. 2969–2974.

[16] ——, "Optimal control design for perturbed constrained networked control systems," IEEE Control Systems Letters, vol. 5, no. 2, pp. 553–558, 2020.

[17] F. Borrelli, A. Bemporad, and M. Morari, Predictive control for linear and hybrid systems. Cambridge University Press, 2017.

[18] C.-C. Han, K.-J. Lin, and C.-J. Hou, "Distance-constrained scheduling and its applications to real-time systems," Transactions on Computers, vol. 45, pp. 814–826, 1996.

[19] P. C. Fishburn and J. C. Lagarias, "Pinwheel scheduling: Achievable densities," Algorithmica, vol. 34, pp. 14–38, 2002.

[20] D. Chen and A. Mok, "The pinwheel: A real-time scheduling problem," in Handbook of Scheduling: Algorithms, Models, and Performance Analysis. Champan & Hall, 2004, ch. 27.

[21] A. Bar-Noy and R. E. Ladner, "Windows scheduling problems for broadcast systems," SIAM Journal on Computing, vol. 32, no. 4, pp. 1091–1113, 2003.

[22] A. Bar-Noy, R. E. Ladner, and T. Tamir, "Windows scheduling as a restricted version of bin packing," ACM Transactions on Algorithms (TALG), vol. 3, no. 3, p. 28, 2007.

[23] M. Margenstern, "Frontier between decidability and undecidability: a survey," Theoretical Computer Science, vol. 231, no. 2, p. 217, 2000.

[24] M. L. Pinedo, Scheduling: Theory, Algorithms, and Systems, 2008.

[25] Y. Zhu and W. X. Zheng, "Observer-based control for cyber-physical systems with periodic dos attacks via a cyclic switching strategy," IEEE Transactions on Automatic Control, 2019.