Compositional Modeling and Analysis of Multi-Hop Control Networks

Rajeev Alur, *Fellow, IEEE*, Alessandro D'Innocenzo, Karl H. Johansson, *Senior Member, IEEE*, George J. Pappas, *Fellow, IEEE*, and Gera Weiss

***Abstract*—We propose a mathematical framework for modeling and analyzing multi-hop control networks designed for systems consisting of multiple control loops closed over a multi-hop (wireless) communication network. We separate control, topology, routing, and scheduling and propose formal syntax and semantics for the dynamics of the composed system, providing an explicit translation of multi-hop control networks to switched systems. We propose formal models for analyzing robustness of multi-hop control networks, where data is exchanged through a multi-hop communication network subject to disruptions. When communication disruptions are long, compared to the speed of the control system, we propose to model them as *permanent link failures*. We show that the complexity of analyzing such failures is NP-hard, and discuss a way to overcome this limitation for practical cases using compositional analysis. For typical packet transmission errors, we propose a *transient error model* where links fail for one time slot independently of the past and of other links. We provide sufficient conditions for almost sure stability in presence of transient link failures, and give efficient decision procedures. We deal with errors that have random time span and show that, under some conditions, the permanent failure model can be used as a reliable abstraction. Our approach is compositional, namely it addresses the problem of designing scalable scheduling and routing policies for multiple control loops closed on the same multi-hop control network. We describe how the translation of multi-hop control networks to switched systems can be automated, and use it to solve control and networking co-design challenges in some representative examples, and to propose a scheduling solution in a mineral floatation control problem that can be implemented on a time triggered communication protocols for wireless networks.**

***Index Terms*—Compositional analysis, multi-hop sensor and actuator networks, networked control.**

Manuscript received April 02, 2010; revised April 16, 2010; accepted April 18, 2011. Date of publication August 08, 2011; date of current version October 05, 2011. This work was supported in part by NSF CPS 0931239 and by the European Commission under Project HYCON2. Recommended by Associate Editor I. Paschalidis.

R. Alur is with the Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: alur@cis.upenn.edu).

A. D'Innocenzo is with the Department of Electrical and Information Engineering, University of L'Aquila, Center of Excellence DEWS, L'Aquila 67100, Italy (e-mail: alessandro.dinnocenzo@univaq.it).

K. H. Johansson is with the ACCESS Linnaeus Centre, School of Electrical Engineering, Royal Institute of Technology, Stockholm 114 28, Sweden (e-mail: kallej@ee.kth.se).

G. J. Pappas is with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: pappasg@ee.upenn.edu).

G. Weiss is with the Department of Computer Science, Ben-Gurion University of the Negev, Ben Gurion University, Be'er Sheva 84105, Israel (e-mail: geraw@bgu.ac.il).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2011.2163873

I. Introduction

THIS paper is part of the research on robustness of networked control systems (e.g., [1]–[3]). While most of the research in this field is on direct networking, we focus on multi-hop networks such as those encountered when analyzing wireless sensor technologies. Wireless networked control systems are spatially distributed control systems where the communication between sensors, actuators, and computational units is supported by a shared wireless communication network. Control with wireless technologies typically involves multiple communication hops for conveying information from sensors to the controller and from the controller to actuators.

The use of wireless networked control systems in industrial automation results in flexible architectures and generally reduces installation, debugging, diagnostic and maintenance costs with respect to wired networks. Wide deployment of wireless industrial automation requires substantial progress in wireless transmission, networking and control, in order to provide formal tools to quantify performance and robustness of a wireless networked control system. The design of the control system has to take into account the presence of the network, as it represents the interconnection between the plant and the controller, and thus affects the dynamic behavior of the closed loop system. Using a wireless communication medium, new issues such as fading and time-varying throughput in communication channels have to be addressed, and communication delays and packet losses may occur. Moreover analysis of stability, performance, and reliability of real implementations of networked control systems requires addressing issues such as scheduling and routing for real communication protocols. The main motivation for studying such systems is the emerging use of wireless technologies in control systems (see, e.g., [4]–[7]). While offering many advantages, the use of multi-hop networks for control is a challenge when it comes to predictability. Motivated by this challenge, we propose a formal modeling and analysis approach for multi-hop control networks.

The challenges in designing and analyzing multi-hop control networks are best explained by considering the recently developed wireless industrial control protocols, such as WirelessHART and ISA 100. These standards allow designers of wireless control networks to distribute a synchronous communication schedule to all communication nodes of a wireless network. More specifically, time is divided into slots of fixed length and a schedule is an assignment of nodes to send data in each slot. The standard specifies a syntax for defining schedules and a mechanism to apply them. However, the issue of designing schedules and routing remains a challenge for the engineers,

and is currently done using heuristics rules. To allow systematic methods for computing and validating schedules, a clear formulation of the effect of schedules on control performance is needed. We summarize the paper's contributions as follows:

1) In Section II we propose a formal model for analyzing the joint dynamics induced by network topology, scheduling, routing, transmission errors and control, and define a switched system that models the dynamics of the composed multi-hop control network. As illustrated in Section II-K, our model incorporates wireless industrial control protocols, such as WirelessHART and ISA 100, and can be used to co-design control and scheduling. For example, in Section IV we develop an experimental tool presented and show that it is possible to resolve design parameters of a controller by representing the dynamics of a multi-hop control network symbolically.

When relating our paper with the current research about the interaction of network and control parameters, most efforts in the literature focuses on scheduling message and sampling time assignment for sensors/actuators and controllers interconnected by wired common-bus networks [8]–[13], while what is needed for modeling and analyzing control protocols is an integrated framework for analysing/co-designing network topology, scheduling, routing, transmission errors and control. To the best of our knowledge, the only formal model of multi-hop wireless sensor and actuator networks is reported in [14]. In this paper, a simulation environment that facilitates simulation of computer nodes and communication networks interacting with the continuous-time dynamics of the real world is presented. The main difference between the work presented in [14] and this work is that here we propose a formal mathematical model that allows more than just simulation. In fact, we show that our approach allows systematic mathematical design techniques such as sensitivity and compositional analysis. This work is also related to the growing research body on switched system (see e.g. [15], [16]). As we show in this paper, a multi-hop control network can be abstracted as a switched system. While generic approaches that ignore the specific structure of the switched system are applicable, we provide a detailed model that identifies the contribution of specific constituents to the dynamics. For example, the elaborated model allows us to apply the approach proposed in [17], [18] for analyzing each control loop separately in a compositional manner.

2) While in Section II perfect communication links are assumed and the focus is on scheduling, in Section III we assume fixed schedules and focus on modeling and analyzing the effects of link failures on the stability of the control loops. We analyze fault tolerance of multi-hop control networks, by proposing a formal model and an analysis tool for verifying stability of the closed loop system in the presence of link failures.

In Section III-A we consider communication errors whose duration is long compared to the speed of the control system, and propose to model them as *permanent link failures*. We show that the complexity of analyzing such failures is NP-hard, and discuss a way to overcome this limitation using compositional analysis. In Section III-B we consider typical packet transmission errors, where links fail for one time slot independently of the past and of other links, and we propose a *transient error model*. We provide a sufficient condition for stability with probability one in presence of transient link failures, and show how it can be used in typical scenarios. In Section III-C we consider failures that have random time span. We identify conditions that allow to reduce the verification of almost sure stability of such systems to the verification of high probability of exponential stability of systems with permanent failures. We focus on practical tools that can scale to large and complex systems. In particular, we analyze the computational complexity of checking the sufficient conditions and propose ways to cope with these complexities be means of over-approximations and compositional analysis.

Relating our approach for addressing link failures, we remark that we model multi-hop control systems as switched systems where link failures induce random switching signals. For this reason the theory of Discrete-Time Markov Jump Linear Systems (see e.g. [19]) applies. In particular any sufficient condition for almost sure stability can be used as a sufficient condition for stability of multi-hop control systems. The difference between this paper and papers that give general such conditions (e.g. [20]–[26]) is that we use the specific structure of the switched systems that arises when multi-hop control networks are modeled. Also, our focus is on conditions that can be efficiently checked under assumptions that are reasonable in relevant applications (wireless sensor/actuator networks). Another line of research that is related to this paper is complexity analysis of control problems (see, e.g., [27], [28]). We establish a new NP-hardness result and discuss ways to walk around computational complexities using compositional analysis and over-approximations.

3) In Section IV we describe how the translation of multi-hop control networks to switched systems can be automated and use it to solve control and networking co-design challenges in some representative examples. In particular, we show that our model allows compositional analysis. We address the problem of designing scalable scheduling and routing policies when closing a considerable number of control loops on the same communication network, and propose an approach based on task-graph abstraction [29]–[31]. The main difference between our work and other studies of task-graph abstractions is that we focus on finding the set of all schedules that satisfy the task-graph constraints as a basis for further analysis, while most of the research is focused on finding individual optimal schedules (see, e.g., [29]). We apply methods developed in [18] in Section IV, and propose a scheduling solution in a mineral floatation control problem that can be implemented on a time triggered communication protocols for wireless networks in Section V.

II. Formal Model for Multi-Hop Control Networks

A Multi-hop control network, consists of a set of plants, a controller, and nodes that communicate sensing and actuation data from plants to controllers and back. The control scheme is illustrated in Fig. 1, where seven wireless nodes are used to measure information from two plants, send the information to a controller and then pass it back and actuate the plants. We assume that each node has radio and memory capabilities, in order to receive, store and transmit data. Each plant is considered as a dynamical system with a finite number of scalar output signals (observable outputs) and scalar input signals (control signals). Nodes in the network interact with the plants through these signals, namely they measure the observable outputs and provide

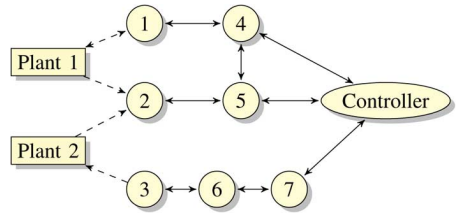


Fig. 1. Example of a multi-hop control network. Circles represent nodes with wireless communication capabilities, solid lines represent radio connectivity and dashed lines represent actuation/sensing.

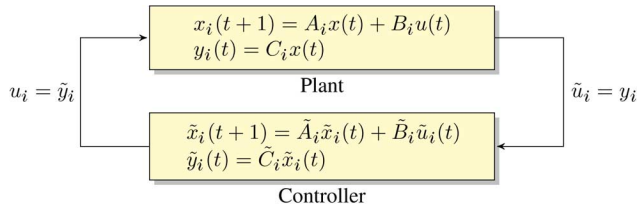


Fig. 2. Model of one control loop.

actuation for the control inputs (dotted arrows). For example, in Fig. 1, node 1 has both sensing and actuation capabilities for Plant 1 (bidirectional dotted arrow); node 2 has only sensing capabilities for both Plant 1 and Plant 2 (unidirectional dotted arrows to node 2 from both Plant 1 and Plant 2); and node 3 has only actuation capabilities for Plant 2 (unidirectional dotted arrow from node 3 to Plant 2). In order to close the control loop, measured data is sent from sensors to the controller through the wireless network. The computation of the control signal is performed in the controller, and control commands are sent from the controller back to the actuators. The solid arrows that connect the nodes model radio connectivity, i.e., a solid arrow is drawn from node v_1 to node v_2 if and only if node v_2 can receive signals transmitted by node v_1.

As detailed in Section II-A, we propose to describe multi-hop control networks by: (1) A mathematical model of the control loops, each consisting of a plant and a dynamic feedback algorithm. (2) The topology of the network, the location of the sensors/actuators, and the routing strategy. Note that the feedback algorithms for all the plants are typically executed by a single computer (controller), but we choose to model them with the plant. In this text, when we focus on one control loop, we will identify the controller with the control algorithm and say that a control loop consists of a plant and a controller, as is done in many control texts.

In Section II-F, we formalize the semantics of multi-hop control networks by defining a switched system. The semantics of the model reflect data flow, as follows. A state of the system is a snapshot of data stored in the nodes. Transitions consist of copying data from nodes to nodes and of transformations of the control algorithms and plants states. Because transitions are governed by schedules, we propose to model the system as a discrete-time switched system where the switching signal is the communication and computation schedules. This model allows analysis of multi-hop control networks using the growing arsenal of techniques from the switched systems theory [16].

A. Syntax

We propose the following formal syntax for describing a multi-hop control network. See the subsections that follow the definition for a more detailed description of each part.

Definition 1: A multi-hop control network is a tuple $\mathcal{N} = \langle \mathcal{D}, \mathcal{G}, \Omega, \mathcal{R} \rangle$, where:

- $\mathcal{D} = \{\langle \langle A_i, B_i, C_i \rangle, \langle \tilde{A}_i, \tilde{B}_i, \tilde{C}_i \rangle \rangle\}_{i=1}^{p}$ models the control loops. Each control loop in D is modeled by a pair of triplets of matrices. The first triplet in each pair defines the dynamics of the plant and the second triplet defines the dynamic of the control algorithm, both in terms of matrices of Linear Time Invariant (LTI) systems. The number of columns in B_i must be the same as the number of rows in $\tilde{C}_i$, which is the number of inputs to the plant. Similarly, the number of rows in C_i must be the same as the number of columns in $\tilde{B}_i$, which is the number of measurable outputs from the plant. Let $\mathbb{I} = \cup_{i=1}^{p} \{u_{i,1} \ldots, u_{i,m_i}\}$ be the set of input signals for the plants, where m_i is the number of columns in B_i (rows in $\tilde{C}_i$). Let $\mathbb{O} = \cup_{i=1}^{p} \{y_{i,1} \ldots, y_{i,l_i}\}$ be the set of output signals from the plants, where l_i is the number of rows in C_i (columns in $\tilde{B}_i$).
- $\mathcal{G} = \langle V, E \rangle$ is a directed graph that models the radio connectivity of the network, where vertices are nodes of the network, and an edge from v_1 to v_2 means that v_2 can receive messages transmitted by v_1. We denote with v_c the special node of V that corresponds to the controller. Let $\mathcal{P}$ be the set of simple paths in $\mathcal{G}$ that start or end with the controller.;
- $\Omega : \mathbb{O} \to V$ assigns to every input and output signal the node that implements, respectively, actuation or sensing;
- $\mathcal{R} : \mathbb{O} \to 2^{\mathcal{P}}$ is a map, which associates to each input (respectively, output) signal a set of allowed simple paths from (respectively, to) the controller. We require that all elements in $\mathcal{R}(y)$ (respectively, $\mathcal{R}(u)$) start (respectively, end) with $\Omega(y)$ (resp.$\Omega(u)$) and end (respectively, start) with the controller, for all $y \in \mathbb{O}$ (respectively, $u \in \mathbb{I}$).

B. Control Loops

The variable $\mathcal{D}$, in the above definition, models the dynamics of the controlled plant and of the feedback algorithm associated with it using the matrices A_i, B_i, C_i, $\tilde{A}_i$, $\tilde{B}_i$ and $\tilde{C}_i$. We omit without loss of generality the feedthrough terms D_i, $\tilde{D}_i$, which can be taken into account with a simple extension of the model. The meaning of this model is illustrated in Fig. 2. Namely, each triplet $\langle A_i, B_i, C_i \rangle$ models an LTI plant and each triplet $\langle \tilde{A}_i, \tilde{B}_i, \tilde{C}_i \rangle$ models an LTI feedback block, interconnected with the plant in the usual way.

Note that the figure depicts a direct interconnection of the plant with the controller while, in reality, the wireless network introduces both measurement and actuation delays. We will model these delays later, based on the topology of the wireless network and the communication and computation schedules.

C. Radio Connectivity Graph

The graph $\mathcal{G}$, in the definition of a multi-hop control network, models the ability of nodes in the wireless network to receive signals sent by others. Formally, the vertices of the graph are the nodes in the network and a directed edge from node v_1 to node v_2 exists if and only if v_2 can receive signals sent by v_1. For

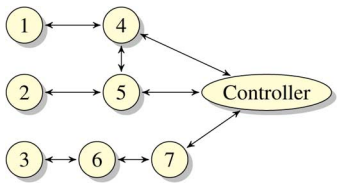


Fig. 3. Radio connectivity graph. Vertices are nodes, an edge from v_1 to v_2 says that v_2 can receive signals from v_1.

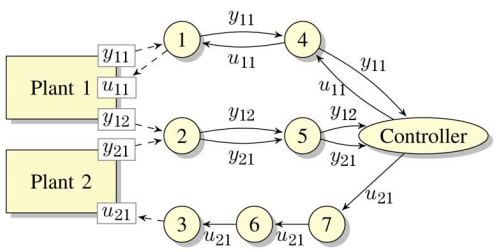


Fig. 4. Static routing, expressed as a set of paths from sensors to the controller and from the controller to actuators.

example, the radio connectivity graph for the multi-hop control networks in Fig. 1 is depicted in Fig. 3.

Plants are not present in the radio connectivity graph because they are not active nodes in the wireless network.

D. Sensors and Actuators

The function $\Omega : \mathbb{O} \to V$ formally defines which nodes of the network are sensors and/or actuators. Moreover, it associates sensors and actuators to the components of the output and input signals of the plant. As an example, in the system illustrated in Fig. 4 the function Ω is depicted with dotted arrows, and is formally defined by $\Omega(u_{11}) = 1$, $\Omega(u_{21}) = 3$, $\Omega(y_{11}) = 1, \Omega(y_{12}) = \Omega(y_{21}) = 2$. where $\mathbb{I} = \{u_{11}, u_{21}\}$ and $\mathbb{O} = \{y_{11}, y_{12}, y_{21}\}$.

The signal $u_{i,j}$ (respectively, $y_{i,j}$) denotes the jth input (respectively, output) of the ith plant. With this naming convention, Ω maps rows (respectively, columns) of the B matrices (respectively, C matrices) to nodes of the wireless network. Specifically, if $\Omega(y_{i,j}) = k$ and $c_{i,j}$ is the jth row of C_i then the data at node k is $c_{i,j}x_i$, where x_i is the state of the ith plant. Similarly, if $\Omega(u_{i,j}) = k$ and $b_{i,j}$ is the jth column of B_i then the scalar at node k is multiplied by $b_{i,j}$ and added to x_i (every time step). These equations formalize the dynamics of the sensors and the actuators.

E. Routing

A (static) routing in a multi-hop control network is a set of acyclic paths from sensors to the controller and from the controller to actuators. For example, in Fig. 4, each sensor is connected to the controller by one path and the controller is connected to each actuator by a path. It is worth to remark that our model allows multiple paths for a specific input/output: since $\mathcal{R}$ is a map, we can associate to each input/output pair a set of routing paths (e.g. to increase reliability by redundancy).

We propose two possible use cases with routing. The first use case is when the designer of the network specifies static routing as a set of allowed paths for each pair sensor-controller and controller-actuator. In this case, data can only flow along the specified paths. The second use case is when no explicit routing is specified, namely the user does not define $\mathcal{R}$. In this case, we assume a default routing $\mathcal{R}$ by considering the set of all acyclic paths from each sensor to the controller, and from the controller to each actuator.

F. Semantics

In an ideal control loop, the input and output signals of plants and controllers are directly interconnected, namely $u(t) = \tilde{y}(t)$, $y(t) = \tilde{u}(t)$, as depicted in Fig. 2 above. However, when a multi-hop network is used to transport measured data from sensors to the controller, and actuation data from the controller to actuators, the semantics of the closed loop system need to incorporate the delays induced by the network. In particular, we need to define (i) how the measured and control data flow through the network (communication schedule), and (ii) how the controller computes the control commands (computation schedule).

G. Memory Slots

As the dynamics of multi-hop control networks are based on modeling information flow from sensors to the controller and from the controller to actuators, the first step towards formal semantics is an identification of the memory slots (registers) that hold that information. Specifically, each node of the network has a memory slot for each input or output signal designated for keeping the information passed to the node regarding the signal. Formally, the vertices of the memory slots graph are pairs $\langle v, \sigma \rangle$ where v is a node and σ is a signal. The edges of the memory slots graph reflect information flow channels. Specifically, there is an edge from $\langle v_1, \sigma \rangle$ to $\langle v_2, \sigma \rangle$ iff $v_1 = v_2$ or if v_1 and v_2 are consecutive nodes on some routing path of the signal σ. Namely, an edge in the graph shows where the information in each memory slot can flow—it can stay in the same memory slot or be moved to a consecutive one.

Definition 2: Given a multi-hop control network with network topology $\mathcal{G} = \langle V, E \rangle$ and routing $\mathcal{R} : \mathbb{O} \to 2^{\mathcal{P}}$, we define the graph $\mathcal{G}_{\mathcal{R}} = \langle V_{\mathcal{R}}, E_{\text{self}} \cup E_{\text{route}} \rangle$ where $V_{\mathcal{R}} = V \times (\mathbb{O})$, $E_{\text{self}} = \{\langle \langle v, \sigma \rangle, \langle v, \sigma \rangle \rangle : v \in V, \sigma \in \mathbb{O}\}$, and $E_{\text{route}} = \{\langle \langle v_1, \sigma \rangle, \langle v_2, \sigma \rangle \rangle : \sigma \in \mathbb{O}$, v_1 and v_2 are consecutive on some $r \in \mathcal{R}(\sigma)\}$.

To avoid handling unneeded memory slots, we consider (without loss of generality) only the sub graph of $\mathcal{G}_{\mathcal{R}}$ reachable from/to the controller.

The function Ω, defined in Section II-E above, which maps each input/output signal to a node, can be automatically extended to the function Ω_{Plant} which maps signals to memory slots (because each memory slot is mapped to a path which maps to a unique signal). Similarly, we will also use the function Ω_{Con} that maps signals of the controller to memory slots. These functions are depicted in Fig. 5.

H. Controllers as Switched Systems

In Section II-B we defined controllers as linear time invariant dynamical systems. Semantically, however, we think of them as linear switched systems. The main reason for this generalization is to allow controllers to collect data before the actual control computation is executed. In particular, according to the dependence between each control signal and the measured data, we want to allow that any element of the control vector can be computed separately, when the relevant subset of measurements

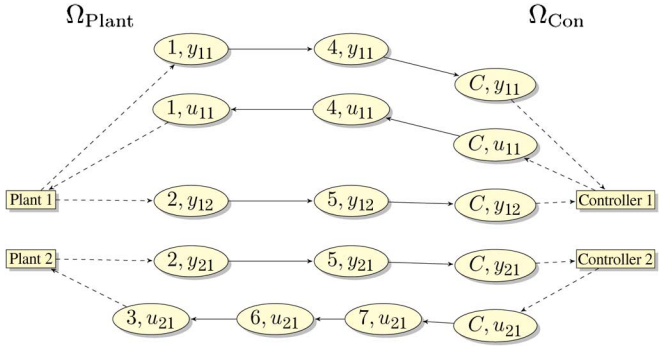


Fig. 5. Graph $\mathcal{G}_{\mathcal{R}}$ obtained by splitting each node to memory slots according to the routing scheme. The self loops are omitted for clearness of the picture.

is ready. This requires coordinating (co-scheduling) computations and communication. Another motivation for modeling the controllers as switched systems is to allow analysis of systems with limited computational resources, where controllers need to operate in "light" mode some of the time, e.g. because other control loops need CPU resources. By defining the dynamics of the controller as switched system, we also allow modeling control techniques such as Kalman filters and Luenberger observers. To accommodate for such generalizations, all the analysis methods that we propose in this paper are independent of the structure of the controller (number of modes, dimensions, etc.).

Similar to what we proposed for routing, we propose two use-cases for handling conversion of controllers to switched systems. The first use-case is when an explicit model of the controller as a switched system is provided, and the second use-case is when only a linear time-invariant model of the controller is specified. The first case requires more modeling efforts, but it allows more general analysis of communication/computation co-scheduling.

For the second case, used in all the examples in this paper, we propose an implicit transformation of the controller from a time invariant system to a switched system as follows. Let $\langle \tilde{A}, \tilde{B}, \tilde{C} \rangle$ be a formulation of a feedback algorithm as a linear time invariant system. We define a switched system with the two modes $M = \{\text{Idle}, \text{Active}\}$. The Idle mode is defined by the matrices $\tilde{A}(\text{Idle}) := \mathbf{1}$ (identity matrix), $\tilde{B}(\text{Idle}) := \mathbf{0}$ (zero matrix) and $\tilde{C}(\text{Idle}) := \tilde{C}$. The Active mode is defined by $\tilde{A}(\text{Active}) = \tilde{A}$, $\tilde{B}(\text{Active}) := \tilde{B}$ and $\tilde{C}(\text{Active}) := \tilde{C}$. This definition models that the computation of the state variables of the controller does not have to be applied at every step, and that the state variables remain constant while the computation is not scheduled.

Mode switches of the controller are coordinated by the computation schedule described in the following section.

I. Scheduling

We propose a formal syntax for describing communication and computation schedule for a multi-hop control network:

Definition 3: Given a multi-hop control network $\mathcal{N}$, let $\mathcal{G}_{\mathcal{R}} = \langle V_{\mathcal{R}}, E_{\mathcal{R}} \rangle$ be the memory slots graph as defined above.

- A communication schedule is a function $\eta : \mathbb{N} \to 2^{E_{\mathcal{R}}}$, that associates to each time t a set of edges of the memory slot graph. The intended meaning of this schedule is that $\langle v_1, v_2 \rangle \in \eta(t)$ iff at time t the content of the memory slot v_1 is copied to the memory slot v_2 (i.e. the physical node that maintains v_1 sends data to the physical node that maintains v_2). We require that if $\langle v_1, v_2 \rangle \in \eta(t)$ then for every $v_3 \neq v_1$, $\langle v_3, v_2 \rangle \notin \eta(t)$. Namely we do not allow assignment of two values to the same memory slot.
- A computation schedule for the ith control loop (corresponding to the ith entry in $\mathcal{D}$ of Definition 1) is a function $\mu_i : \mathbb{N} \to M_i$ where M_i is the set of modes of the switched-system that models the controller of the ith control loop, as described in Section II-H. The meaning of this function is that $\mu_i(t)$ defines the mode of the controller at time t.

It is worth to remark that a constraint in the simultaneous usage of certain communication channels (e.g. because of interference among transmitting nodes) can be given by restricting the set of possible communication schedules as a subset of $2^{E_{\mathcal{R}}}$.

In Section IV, below, we present a compositional analysis based on representing sets of communication schedules as regular languages\ over the alphabet $2^{E_{\mathcal{R}}}$. In this context, one can also represent the set of feasible schedules in the same form. For example, if the transmission of data from node v_1 to node v_2 uses a mutually exclusive resource (e.g. radio frequency) shared with the transmission of data from node v_3 to node v_4 then the set of feasible schedules should be a sub-language of $\{S \subset E_{\mathcal{R}} : \langle v_1, v_2 \rangle \notin S \vee \langle v_3, v_4 \rangle \notin S\}^*$ (where $*$ is the Kleene star [32]).

J. Multi-Hop Control Networks as Switched Systems

Based on the syntactical definition of a multi-hop control network and the schedules, we now define dynamics as switched systems. To allow compositional analysis, we model each control loop separately (plant, controller, and the data flow between them). Let i be the identifier of that control loop (corresponding to its index in the array $\mathcal{D}$ in Definition 1). We use the descriptions of the plant and the control algorithm as LTI systems, modeled by the matrices $\langle A_i, B_i, C_i \rangle$ and $\langle \tilde{A}_i, \tilde{B}_i, \tilde{C}_i \rangle$ respectively. Recall that, in Section II-H, we transformed the control algorithm to a switched linear system with the parameterized matrices $\langle \tilde{A}_i(\cdot), \tilde{B}_i(\cdot), \tilde{C}_i(\cdot) \rangle$. The state of the switched system that models the control loop is a vector $x = \langle x_p, x_{v_1}, \ldots, x_{v_n}, x_c \rangle$ where x_p is the state of the plant, $\langle x_{v_1}, \ldots, x_{v_n} \rangle$ is a vector representing the values of the memory slots (in some fixed order), and x_c is the state of the controller. The evolutions of the different parts of the state are as follows:

- Using the matrices A_i, B_i from the definition of the plant as LTI system, we write $x_p(t+1) = A_i x_p(t) + B_i u(t)$ and $u(t) = \langle x_{\Omega_{\text{Plant}}(u_1)}(t), \ldots, x_{\Omega_{\text{Plant}}(u_m)}(t) \rangle$ where $u_1, \ldots, u_m \in \mathbb{I}$ are the input signals of the plant and Ω_{Plant} is the function that maps signals of the plant to sensor/actuator memory slots, i.e. the inputs to the plant are the values stored in the actuators memory slots. Similarly, $\langle x_{\Omega_{\text{Plant}}(y_1)}(t), \ldots, x_{\Omega_{\text{Plant}}(y_l)}(t) \rangle = C_i x_p(t)$ where $y_1, \ldots, y_l$ are the output signals of the plant, i.e., the outputs from the plant are stored in the memory slots of the sensors.
- The rest of the memory slots are updated according to the communication schedule. Specifically, $x_v(n+1) = \sum_{\langle v', v \rangle \in \mu(t)} x_{v'}(t)$ i.e., the data in each memory slot is updated to be the sum of the values of all the sources of the incoming edges to the node. In this paper, we insist that each node has at most one incoming edge which means

that each destination of an edge is assigned with the value stored in the source of the edge.

- For the controller, we write $x_c(t+1) = \tilde{A}_i(\eta(t))x_c(t) + \tilde{B}_i(\eta(t))\tilde{y}(t)$ and $\tilde{y}(t) = \langle x_{\Omega_{\mathrm{Con}}(y_1)}(t), \ldots, x_{\Omega_{\mathrm{Con}}(y_l)}(t)\rangle$, where $y_1, \ldots, y_l \in \mathbb{O}$ are the output signals and Ω_{Con} is the function that maps signals of the controller to memory slots. Similarly, $\langle x_{\Omega_{\mathrm{Con}}(u_1)}(t), \ldots, x_{\Omega_{\mathrm{Con}}(u_m)}(t)\rangle = \tilde{C}_i(\eta(t))x_c(t)$ where $u_1, \ldots, u_l \in \mathbb{I}$ are the input signals.

The following two definitions formalize these dynamics as a linear switched system. The dynamics of the memory slots are modeled using the adjacency matrix of the graph induced by the communication schedule. The state of the system is $x = \langle x_p, x_{v_1}, \ldots, x_{v_n}, x_c\rangle$.

Definition 4: Given a multi-hop control network $\mathcal{N}$, consider a plant $\langle A_i, B_i, C_i\rangle$, and the corresponding switched linear controller $\langle \tilde{A}_i(\cdot), \tilde{B}_i(\cdot), \tilde{C}_i(\cdot)\rangle$. For any subset $e \subseteq E_{\mathcal{R}}$ representing a sub-graph of the memory slots graph, and for any controller mode $m \in M$, we define

$$T_i(e,m) := \begin{pmatrix} A_i & B_i \cdot O_{\mathrm{Plant}} & 0 \\ I_{\mathrm{Plant}}^T \cdot C_i & \mathrm{Adj}(\langle V_{\mathcal{R}}, e\rangle)^{\mathrm{T}} & O_{\mathrm{Con}}^T \cdot \tilde{C}_i(m) \\ 0 & \tilde{B}_i(m) \cdot I_{\mathrm{Con}} & \tilde{A}_i(m) \end{pmatrix}$$

where $V_{\mathcal{R}} = \{v_1, \ldots, v_{|V_{\mathcal{R}}|}\}$, $\mathbb{I} = \{i_1, \ldots, i_{|\mathbb{I}|}\}$ and $\mathbb{O} = \{o_1, \ldots, o_{|\mathbb{O}|}\}$ are respectively enumerations of memory slots, inputs and outputs, $\mathrm{Adj}(\langle V_{\mathcal{R}}, e\rangle)^{\mathrm{T}}$ is the transposed adjacency matrix of the sub-graph induced by e on $\langle V_{\mathcal{R}}, E_{\mathcal{R}}\rangle$, and I_x (respectively, O_x) is a $\{0, 1\}$ matrix of matching size with the entry $I_x(r,c)$ (respectively, $O_x(r,c)$) being one if and only if $\Omega_x(v_r) = i_c$ (respectively, $\Omega_x(v_r) = o_c$), for $x \in \{\mathrm{Plant}, \mathrm{Con}\}$.

Definition 5: The dynamics of the control loop are modeled by the switched system

$$x_i(t+1) = T_i\left(s(t)\right) x_i(t)$$

where the communication and computation schedule $s(t) = \langle \eta(t), \mu(t)\rangle$ is the switching signal.

It is worth to remark that the switching signal $s(t) = \langle \eta(t), \mu(t)\rangle$ can be both interpreted as a predefined time signal or as a further input of the system. In the former case, the interpretation is that a static schedule has been defined a-priori, which cannot be modified as time flows. In the latter case, the interpretation is that the schedule is an input variable, and a dynamic schedule can be applied to the system.

Note the structure of the matrix $T_i(\cdot,\cdot)$ that explicitly expresses the interplay between the components of a multi-hop control network. Specifically, the dynamics of the plant are at the top left, the dynamics of the controller are at the bottom right and the adjacency matrix of the sub-graph of the memory slots graph induced by the communication schedule is at the center. This model allows to use techniques from the theory of switched systems to analyze multi-hop control networks.

By combining the models of the individual loops, one can obtain a model of the whole multi-hop control network. For example, in Section IV we show how the methods presented in [17], [18], [33] are applied in the context of multi-hop control networks. Specifically, the theory of formal languages is applied for answering competing resource requirements of the loops to achieve stability of the whole system. The ability to analyze systems in a compositional manner is enabled by modeling each loop separately.

In this paper we assume that the schedules are periodic, i.e. there exists Δ such that $\eta(t+\Delta) = \eta(t)$ and $\mu(t+\Delta) = \mu(t)$, for all t. In that case, the schedules can be specified by the sequences $\eta(1), \ldots, \eta(\Delta)$ and $\mu(1), \ldots, \mu(\Delta)$. Let $\hat{T}_i := T_i(\eta(\Delta), \mu(\Delta)) \cdots T_i(\eta(1), \mu(1))$ model the transformation of the state over a period of the schedules. Assuming periodic schedules is reasonable, since most of the time triggered control protocols specifies periodic transmission schedules (see e.g. the WirelessHART specification [34]).

The focus of the paper is on stability, as expressed in the following definition.

Definition 6: The control loop i is called stable iff the matrix $\hat{T}_i$ is stable (all eigenvalues in the unit sphere). The whole control network is called stable if all the control loops are stable.

The communication schedule that occurs in each period might be not deterministically identified, because of local routing choices and/or transmission errors. Let $\mathcal{L}$ be the finite set of all communication schedules that might occur for any period. In this case, the transformation of the state over a period of the schedules is given by $\hat{x}_i(t+1) = \hat{T}_i(s(t))\hat{x}_i(t)$, where $s(t) \in \mathcal{L}$ is a non-deterministic switching signal that models the schedule during the period t.

For simplicity, we assume a central controller in this paper. Note that allowing assignment of different controllers to control loops requires only a minor adjustment to the proposed model (the dynamics of the individual loops remain the same). If we also want to allow multiple controllers to a single loop, we may need to add some more adjustments (change the definition of the matrix T_i).

K. Multi-Hop Control Networks as WirelessHART Compliant Networks

Our model for multi-hop control networks incorporates the WirelessHART wireless industrial control protocol. More precisely, it allows modeling the MAC layer (communication scheduling) and the Network layer (routing) of the WirelessHART specification. About Wireless HART MAC Layer [34] access to the channel is time slotted, where each slot is 10 ms. A series of time slots for a given frequency channel forms a superframe. The superframe is repeated periodically, and can be defined as illustrated in Definition 3. About WirelessHART Network Layer [35] routing can be implemented using *graph routing*, which provides for each pair of nodes (one source and one destination) a set of paths connecting the two nodes as an acyclic directed graph associated to the destination node. Graph routing can be defined as illustrated in Definition 1.

III. Multi-Hop Control Networks With Probabilistic Link Failures

We introduce in this Section link failures to the model. The main question that we want to ask is whether a stable system remains stable in the presence of link failures.

A. Permanent Failures

We consider a model that takes into account the occurrence of a permanent fault in a node. This model is a natural abstraction

of systems where link failures are long compared to the speed of the control system. The property that we would like to guarantee for such systems is that the probability that the system is stable is higher than a prescribed threshold. We begin with a formal definition of the error model discussed in this section:

Definition 7: A permanent link-failure model for the network is a function $F : E \to [0,1]$ where $F(\langle v_1, v_2\rangle)$ models the probability that the communication link from v_1 to v_2 fails. The function $\Phi(S) = \Pi_{e\in S} F(e)\Pi_{e\in E\setminus S}(1-F(e))$ assigns each set $S \subseteq E$ with the probability that the set of failed links is exactly S (i.e. the edges in S fail and the edges not in S do not fail).

For a set $S \subseteq E$ let $\eta_S(t) := \eta(t) \setminus (S \times (\mathbb{O}))$. Namely, $\eta_S(t)$ is obtained from the schedule $\eta(t)$ by removing all messages that use an edge in S. This definition models the effective schedule when the links in S fail. The following definition specifies the notion of stability that we are interested in, when the permanent link-failure model is considered.

Definition 8: The probability that a control loop with index $i \in \{1,\dots,p\}$ is stable is the probability that $T_i(\eta_S(P), \mu(P)) \cdots T_i(\eta_S(1), \mu(1))$ is stable (all eigenvalues are inside the unit sphere) when S is chosen randomly according to the distribution Φ. The probability that the system is stable is the probability that all the control loops are stable.

The usefulness of permanent failure analysis is clearer when considering the design phase of the scheduling and routing. Each link can be characterized by the probability of being subject to a failure: however, e.g. if we introduce redundancy in the scheduling and routing with multiple routing paths between plant and controller, a failure of one node does not necessarily lead to unstability of a control loop. More precisely, the more scheduling and routing introduce redundancy, the more stability of a control loop is robust with respect to permanent link failures. Of course there is a tradeoff, since unnecessary redundancy misspends communication resources. Permanent failures analysis allows addressing the following problem:

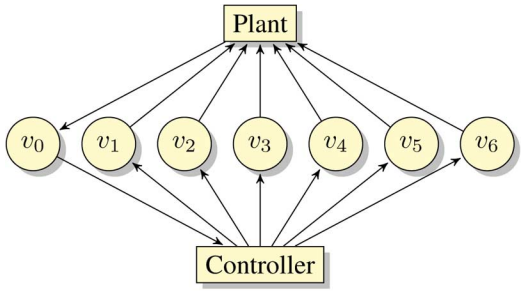
> Design the minimum redundancy in the scheduling and routing, such that the probability that the control loop i is stable is greater than P'',

where $P \in [0,1]$ is for example equal to 0.99, and is the probability that the control loop i preserves stability when permanent failures of links occur.

Definition 8 suggests the following algorithm for computing the probability that the system is stable.

Algorithm 9 (Naive Algorithm): We can compute the probability that the system is stable by enumerating all the subsets of E. Specifically, for each $S \subseteq E$, we can compute the matrix $T_i(\eta_S(P), \mu(P)) \cdots T_i(\eta_S(1), \mu(1))$, check whether it is stable or not, and sum the probabilities.

Clearly, the complexity of the naive algorithm is exponential in the size of the graph. A natural question is whether there exists a polynomial algorithm for that problem. To answer this question, we analyze the following decision problem: Given a description of the multi-hop control network and of a permanent error model, decide whether the probability that the system is unstable is above a specified threshold. In the next proposition we show that this decision problem is NP-hard, suggesting that it may not be possible to improve the time complexity of Algorithm 9 (if P $\neq$ NP).



(a)

$$A_p = 1/2,$$
$$B_p = \left(.37, .37, .37, -.2, -.2, -.2\right),$$
$$C_p = 1,$$
$$A_c = 0, B_c = 1,$$
$$C_c = \left(1, 1, 1, 1, 1, 1\right)^T$$

(b)

$$\eta(1) = \eta(3) = \eta(5) = \emptyset$$
$$\eta(2) = \{\langle\langle v_0, C\rangle, y_1\rangle\}$$
$$\eta(4) = \{\langle\langle C, v_i\rangle, u_i\rangle : i = 1,\dots,6\}$$
$$\mu(1) = \mu(2) = \mu(4) = \mu(5) = \text{Idle}$$
$$\mu(3) = \text{Active}$$

(c)

Fig. 6. Multi-hop control network. (a) Network topology; (b) control loop; and (c) schedule with period $P = 3$.

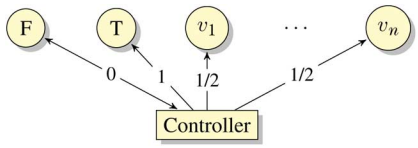


Fig. 7. Topology of a multi-hop control network for formula with variables $x_1,\dots,x_n$. Edge labels represent failure probabilities.

Proposition 10: Given a permanent error model, deciding whether $P_{\text{stable}} > \alpha$, where $\alpha \in [0,1]$ is a constant and P_{stable} is the probability that a multi-hop control network is stable, is NP-hard.

Proof: Consider the multi-hop control network depicted in Fig. 6 where $\Omega(y_1) = v_0$ and $\Omega(u_k) = v_k$ for $k = 1,\dots,6$ and $E = \{\langle C, v_i\rangle : i = 1,\dots,6\} \cup \{\langle v_0, C\rangle\}$ (where C is the controller node). It is easy to verify by computing the eigenvalues of the closed loop system for each edge failure (e.g., using the Mathematica based tool described in [7]) that this system is stable iff one of the edges between the controller and nodes v_1, v_2, v_3 fails or an edge between the controller and one of the nodes v_4, v_5, v_6 does not fail. In other words, using a Boolean random variable x_i to denote the event that the edge from the controller to node v_i fails, we can say that the system is stable iff $x_1 \vee x_2 \vee x_3 \vee \neg x_4 \vee \neg x_5 \vee \neg x_6$.

Let P be a Boolean formula in 3CNF (conjunctive normal form with 3 literals per clause). Let $x_1,\dots,x_n$ be the variables in P. We define a multi-hop control network that is stable with nonzero probability iff the formula is satisfiable, as follows.

The topology of the network is depicted in Fig. 7, namely, $E = \{\langle F, C\rangle, \langle C, F\rangle, \langle C, T\rangle\} \cup \{\langle C, v_i\rangle : i = 1,\dots,n\}$. The permanent link-failure model, illustrated by the edge labels in Fig. 7, is defined by $F(\langle C, T\rangle) = 1, F(\langle C, F\rangle) = F(\langle F, C\rangle) = 0$, and $F(e) = 1/2$ for all $\{\langle C, v_i\rangle : i = 1,\dots,n\}$.

The control loops of the network correspond to the clauses of the Boolean formula (in 3CNF form). For the ith clause $C_i = l_{i,1} \vee l_{i,2} \vee l_{i,3}$ (where each $l_{i,j}$ is either a variable or the negation of a variable), we define the ith control loop with the dynamics shown in Fig. 6(b). The communication schedule is given by

$$\begin{aligned}\eta(1)=\eta(3)=\eta(5)=\emptyset;\ \eta(2)=\{\langle\langle F,C\rangle, y_{i,1}\rangle : i=1,\ldots,p\};\\ \eta(4)=\{\langle\langle C,v_k\rangle, u_{i,j}\rangle : l_{i,j}=x_k\}\cup\{\langle\langle C,F\rangle, u_{i,j+3}\rangle : l_{i,j}=x_k\}\\ \cup\{\langle\langle C,v_k\rangle, u_{i,j+3}\rangle : l_{i,j}=\neg x_k\}\\ \cup\{\langle\langle C,T\rangle, u_{i,j}\rangle : l_{i,j}=\neg x_k\}\end{aligned}$$

and Ω is defined accordingly, i.e. $\Omega(y_{i,1}) = F$ and $\Omega(u_{i,j}) = \{v : \langle\langle C,v\rangle, u_{i,j}\rangle \in \eta(4)\}$. By the claim we made at the beginning of this proof, if we regard $x_1,\ldots,x_n$ as the Boolean random variables where x_k is true iff the edge $\langle C, v_k\rangle \in S$, we get that the ith loop is stable iff the ith clause is satisfied by these Boolean variables, and therefore the system is stable with nonzero probability iff there exists a satisfying assignment to the whole formula.

We established a polynomial reduction from deciding satisfiability of 3CNF formulas to deciding whether a multi-hop control network is stable with positive probability. In particular, since deciding satisfiability of 3CNF formulas is NP-complete, we get that the later decision problem is NP-hard. ∎

The remedy of the above result is that verifying that the probability of stability is above a threshold is, in general, hard. Thus, we should not expect to solve it in a polynomial time (unless P = NP). However, since our modeling approach allows compositional analysis, these news may not be so bad. Specifically, by analyzing each control loop separately, we can focus on the subgraph relevant to the considered loop, thus reducing the number of edges to allow subsets enumeration.

In some cases, such as the one that we are going to explore in Section III-C, stability of an abstract model is not enough to infer correctness of the system. To cope with this difficulty, we propose a parameterized notion of stability where the speed of convergence to the stable equilibrium is explicitly specified, as follows.

Definition 11: The probability that a network control system with a permanent link-failure model is exponentially stable with the parameters $q \in \mathbb{N}$ and $r \in (0,1]$ is the probability that $\|T_i(\eta_S(P),\mu(P))\cdots T_i(\eta_S(1),\mu(1))^q\| < r$ for all $i = 1,\ldots,p$, when S is chosen randomly according to the distribution Φ.

Algorithm 9 extends to exponential stability directly. It is also easy to verify that the probability of stability is higher than the probability of exponential stability for any parameters q and r. A system might be stable but converge to equilibrium slowly, while exponential stability bounds the rate of convergence from above and thus may prove more suitable for applications where fast convergence is required (see [17] for a similar discussion).

B. Transient Errors

In this section, we analyze an error model where links fail for one time slot independently of each other, which is the typical model adopted for packet loss probabilistic characterization of a communication channel. A formal specification of the transient link-failures model follows.

Definition 12: A transient link-failures model for a multi-hop control network is a function $D : E \to [0,1]$. For an edge $\langle v_1, v_2\rangle \in E$, the number $D(\langle v_1, v_2\rangle)$ models the probability that the link from v_1 to v_2 fails, independent of the past and of other links. In other words, the probability that $S \subseteq E$ is the set of links that failed it an arbitrary time slot is $\prod_{e\in S} D(e)\prod_{e\in E\setminus S}(1-D(e))$.

The property that we want to verify for multi-hop control networks with transient link failures is almost sure stability, defined as follows.

Definition 13: A multi-hop control network, with a transient link-failures model D, is said to be almost surely stable if the state converges to zero almost surely.

The above definition requires that the probability that all the control loops are stable, when messages drop independently with probabilities given by D, is one. Let

$$\rho := \sum_{S_1,\ldots,S_P\subseteq E} P(S_1,\ldots,S_P)\left\|\prod_{t=P}^{1} T_i\left(\eta_{s_t}(t),\mu(t)\right)\right\|$$

where $P(S_1,\ldots,S_P) := \prod_{t=1}^{P}(\prod_{e\in S_t} D(e)\prod_{e\in E\setminus S_t}(1-D(e)))$ be the probability that the sets of links that failed at times $t = 1,\ldots,P$ are $S_1,\ldots,S_P$, respectively. In words, ρ is the expected norm $E(\|\hat{T}_i\|)$ of the transformation applied to the state variables over a period of the schedule. In the following proposition we relate this number to almost sure stability.

Proposition 14: If $\rho < 1$ then the multi-hop control network is almost surely stable.

Proof: Let $T_i(j) := \prod_{t=(j+1)P}^{jP} T_i(\eta_{S(t)}(t),\mu(t))$ where $S(t)$ is the set of failed edges at time t chosen randomly by the distribution $P(S(t)=s) = \prod_{e\in s} D(e)\prod_{e\notin s}(1-D(e))$. Namely, $T_i(j)$ is the random transformation of the state variables in the jth cycle that depends on edges that fail between times jP to $(j+1)P$. Since $\|T_i(1)\|, \|T_i(2)\|, \ldots$ is a sequence of i.i.d random variables whose expectation ρ is smaller than one, by the strong law of large numbers, there exists an integer N such that for every $n > N$ the summation $(1/n)\sum_{j=1}^{n}\|T_i(j)\|$ is almost surely smaller than $\alpha := (1+\rho)/2$ which is smaller than one. Since the geometric mean is smaller than the arithmetic mean, $\prod_{j=1}^{n}\|T_i(j)\| \le ((1/n)\sum_{j=1}^{n}\|T_i(j)\|)^n < \alpha^n$ which goes to zero as n goes to infinity (in an exponential rate). In particular, since the norm of a product is smaller than the product of the norms, $\|\prod_{j=1}^{n} T_i(j)\|$ goes to zero almost surely as n goes to infinity. ∎

Proposition 14 suggests an algorithm for verifying almost sure stability by enumerating all P sequences of subsets of E and verifying that ρ is smaller than one. This algorithm is, of course, only applicable for small systems. However, we can derive easier to check conditions in some special cases as we show below.

Proposition 15: $\rho \le (1-\varepsilon)\|\hat{T}_i\| + \varepsilon\delta^P$ where ε is the probability of having a link failure during a single cycle of the schedule and $\delta := \max\{\|T_i(\eta_S(t),\mu(t))\| : S \subseteq E, t = 1,\ldots,P\}$.

Proof: By the law of conditional expectation, $E(\|\hat{T}_i\|) = P[S]\cdot E(\|\hat{T}_i\|\|S) + P[S^C]\cdot E(\|\hat{T}_i\|\|S^C)$ where $E(\cdot)$ denotes expectation, S is the event of not having any link failure along a schedule, and S^C is the complement of this event.

In particular, ρ can be written as the sum of $\varepsilon E(\|\prod_{t=P}^{1} T_i(\eta_{s_t}(t),\mu(t))\| | \forall t.s_t = \emptyset)$ and $(1-\varepsilon)E(\|\prod_{t=P}^{1} T_i(\eta_{s_t}(t),\mu(t))\| | \exists t.s_t \neq \emptyset)$. The first summand reduces to $\varepsilon\|\hat{T}_i\|$ because the expectation is redundant once the sets $s_1,\ldots,s_t$ are fixed to be empty. For the second summand we apply the following over-approximation. By definition, $\|T_i(\eta_{s_t}(t),\mu(t))\| \leq \delta$ for every t and s_t. In particular, since the norm of a product is smaller or equal to the product of the norms, $\|\prod_{t=P}^{1} T_i(\eta_{s_t}(t),\mu(t))\| \leq \delta^P$ for any $s_1,\ldots,s_P \subseteq E$. Together, we get that ρ is smaller or equal to $\varepsilon\|\hat{T}_i\| + (1-\varepsilon)\delta^P$. ∎

Proposition 15 provides an over-approximation of ρ and therefore a stricter sufficient condition for almost sure stability. However, in some cases, this condition is easier to verify. For example: Imagine that we have only one link active at each time slot ($|\eta(t)| = 1$ for all t) and the schedule is 100 steps long. To apply Proposition 14, we need to enumerate all sequences $\langle S_1,\ldots,S_{100}\rangle$ such that $S_t \subseteq \eta(t)$; requiring 2^{100} iterations. With Proposition 15 we get down to 100 iterations because the computation of δ can be done by enumerating the individual transformations $T_i(S_t,\mu(t))$ for $S_t \subset \eta(t)$. We can do with less iterations if parts of the schedule repeat (see Example 16).

An application of Proposition 15, when $\|\hat{T}_i\| < 1$, is computing a threshold $\varepsilon_0 = (\|\hat{T}_i\| - 1)/(\|\hat{T}_i\| - \delta^P)$ such that if the probability of having a link failure during a period of the schedule is smaller than ε_0 the system is almost surely stable. This estimation is practical when the cardinality of $\eta(t)$ is small, namely only few edges transmit simultaneously. In fact, let $|\eta(t)| \leq m$ for all $t = 1,\ldots,P$: the complexity of computing δ by enumerating the subsets of each $\eta(t)$ is given by $O(2^m P)$. The following example illustrates this idea.

Example 16: Consider a plant whose dynamics are given by the following discrete-time single-input-single-output linear time invariant system

$$x_p(t+1)=\begin{pmatrix}2/3 & 0\\ 1/4 & 1\end{pmatrix}x_p(t)+u_p(t)\begin{pmatrix}0\\1\end{pmatrix};\ y_p(t)=\begin{pmatrix}1\\1\end{pmatrix}^T x_p(t).$$

The topology of the multi-hop network connecting the plant with the controller is given by $G = \langle\{1,C\},\{\langle 1,C\rangle,\langle C,1\rangle\}\rangle$ and the sensors/actuators mapping $\Omega_{\text{Plant}}(y_{1,1}) = \Omega_{\text{Plant}}(u_{1,1}) = 1$. In words: there is a single node that measures the output of the plant, sends it to the controller and actuates the input when it gets the command from the controller. Assume the natural communication and computation schedules $\eta = \{\emptyset,\{\langle\langle 1,C\rangle,y_{1,1}\rangle\},\emptyset,\{\langle\langle C,1\rangle,u_{1,1}\rangle\},\emptyset\}$ and $\mu = \{\text{Idle},\text{Idle},\text{Active},\text{Idle},\text{Idle}\}$.

We use eigenvalues assignment to design a controller, as follows. First, we fix the dynamics of the controller to be $x_c(t+1) = \alpha x_c(t) + \beta u_c(t)$; $y_c(t) = \gamma x_c(t)$ where $x_c(t)$, $y_c(t)$, and $u_c(t)$ are scalars describing, respectively, the state, the output and the input to the controller at time t. Next, we set the design parameters α,β, and γ such that the eigenvalues of $\hat{T}_i$ are in the unit sphere (the matrix is stable). For example, it can be readily verified using Equation (4) and the definition of $\hat{T}_i$ that the values $\alpha = -0.7, \beta = -1$, and $\gamma = .72$ achieve this goal.

To analyze robustness of this system, we find a natural number m such that $\|(\hat{T}_i)^m\|$ is smaller than one. In this case, $m = 4$ is the minimal such number. In particular, if we use the schedules $\eta' = \eta^4$ and $\mu' = \mu^4$ (concatenations of four copies of the original schedules) we get that the matrix $\hat{T}_i' = (\hat{T}_i)^4$, modeling the state transformation induced by the long schedule, satisfies $\|\hat{T}_i'\| = 0.43$ which is smaller than one. By listing the matrices $T_i(\eta(t),\mu(t))$ and $T_i(\emptyset,\mu(t))$ for $t = 1,\ldots,5$, we compute that $\delta = 1.84373$.

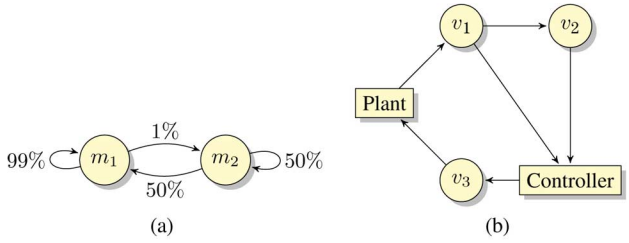


Fig. 8. Example of a system with a Markov link-failures model. (a) Markov chain; (b) network topology; and (c) message drop probabilities.

Using Proposition 15, we can now compute a threshold $\varepsilon_0 = (\|\hat{T}_i'\| - 1)/(\|\hat{T}_i'\| - \delta^{20}) = 2.7 \times 10^{-6}$ such that if the probability of having a link failure during a period of the schedule is smaller than ε_0 the system is almost surely stable. In particular, almost sure stability is guaranteed if the probability of having a link failure in an individual time slot is below $1-(1-\varepsilon_0)^{1/20} = 1.38 \times 10^{-7}$.

C. Error With Random Time Span

In this section, we analyze a detailed failure model for multi-hop control networks where links can recover from failures after some time (random or deterministic). Specifically, we describe the dynamics of failures by a Markov chain.

Definition 17: A Markov link-failures model for a multi-hop control network is a pair (Θ, D_Θ). Θ is a discrete-time Markov chain with state space M, where each $m \in M$ represents a different link-failures model, that switches mode at the beginning of each period of the communication schedule. $D_\Theta : M \times E \to [0,1]$ is a function that assigns to each link-failures model $m \in M$ and to each link $\langle v_1, v_2\rangle \in E$ the probability that the link from v_1 to v_2 fails for the duration of the period of the communication schedule.

Example 18: See Fig. 8. In this example we have a network topology with a short-cut edge from node v_1 to the controller. This edge becomes unreliable when the Markov chain moves to mode m_2. With this formal model we can compare schedules that use the shortcut to schedules that go through v_2.

Note that Markov link-failures model is more general than both the permanent-failures and the transient-failures models, in the sense that both models are special cases of it. The more general model allows more realistic modeling but it is harder to analyze. In the following proposition, we reduce almost sure stability of a system with the Markov link-failures model to high probability of exponential stability of the same system with a permanent link-failures model.

Proposition 19: Consider a multi-hop control network with Markov link-failures model where the Markov chain has a stationary distribution π. Let $F(e) = \sum_{m\in M} \pi(m) D_\Phi(m,e)$ be a permanent link-failures model for the system. Then a sufficient

condition for the system with the Markov link-failures model to be almost surely stable is $\alpha r + (1-\alpha)\delta < 1$ where α is the probability that the system with the permanent link-failures model is exponentially stable with parameters $q = 1$ and $r < 1$ and $\delta = \max_{s,i} \| \prod_{t=P}^{1} T_i(\eta_s(t), \mu(t))\|$.

Proof: Similar to the proof of Proposition 14, we can establish that a sufficient condition for almost sure stability of a system with the Markov link-failures model is $\sum_{S \subseteq E} P(S) \| \prod_{t=P}^{1} T_i(\eta_s(t), \mu(t))\| < 1$ for every i; where $P(S) := \prod_{e \in S} F(e) \prod_{e \in E \setminus S} (1 - F(e))$. Because $\alpha \geq \sum \{P(S) : \| \prod_{t=P}^{1} T_i(\eta_s(t), \mu(t))\| < r\}$ and $\delta \geq \| \prod_{t=P}^{1} T_i(\eta_s(t), \mu(t))\|$ for every S and i, we get that $\sum_{S \subseteq E} P(S) \| \prod_{t=P}^{1} T_i(\eta_s(t), \mu(t))\| \leq \alpha r + (1-\alpha)\delta$ and therefore $\alpha r + (1-\alpha)\delta < 1$ is a sufficient condition for almost sure stability. ∎

The number $\delta = \max_{s,i} \| \prod_{t=P}^{1} T_i(\eta_s(t), \mu(t))\|$, used in the statement of the proposition, is typically easy to compute because in well engineered systems it should be possible to show that the worst case performance is when the network is completely unavailable (otherwise, we can improve performance by not sending some information). Proposition 19 supports our choice to focus on analyzing the probability of stability under the permanent link-failures mode, by showing that the analysis of a more general model can be reduced to it. Furthermore, the proposition identifies that the permanent link-failures model is a good abstraction of the general link-failures model when $\alpha r + (1-\alpha)\delta < 1$, namely, when the probability of exponential stability is high compared to δ which is a parameter quantifying the worst-case divergence speed of the system.

IV. Examples of Applying Our Models to Analysis and Design of Multi-Hop Control Networks

We demonstrate how our approach can be practically used. The main tool is the transformation of a structural description of a multi-hop control network to a switched system that models the combined dynamics of the control loops and the wireless network. This switched system is a time varying (switched) linear system that models the dynamics of the multi-hop control network where the communication and computation schedules act as a switching signal that governs the selection of transformation matrices. The variables that the matrices transform are the state of the controlled plants, the state of the controllers, and the values of the memory slots. The switched system model can be used to analyze schedules, network topologies, and controller designs, as demonstrated below.

First Example: Fixing a Schedule and Designing the Controller Accordingly: As a first instance of how the proposed modeling approach can be used, we show an example that demonstrates how it can be applied to design a controller. Consider the network depicted in Fig. 3 where the first plant is a double integrator, modeled by the equation

$$\dot{x} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

with output $y = x$. When sampling with time-step (sampling interval) h, the discretized system is

$$x^+ = \begin{pmatrix} 0 & h \\ 0 & 1 \end{pmatrix} x + \begin{pmatrix} h^2/2 \\ h \end{pmatrix}.$$

For the sake of the example, we choose $h = 1/20$.

The approach that we propose in this example consists of fixing a schedule for the system and, then, designing a controller that stabilizes the plant while taking into account the delays introduced by the network. To that end, we start with a cyclic schedule whose cycle is the following communication and computation sequences. As a communication schedule (i.e. a sequence of sets of edges of the memory slots graph) we choose:

$$\begin{aligned} &\langle \emptyset, \{\langle 1, y_{1,1}\rangle \to \langle 4, y_{1,1}\rangle\}, \{\langle 2, y_{1,2}\rangle \to \langle 5, y_{1,2}\rangle\}, \\ &\quad \{\langle 4, y_{1,1}\rangle \to \langle C, y_{1,1}\rangle\}, \{\langle 5, y_{1,2}\rangle \to \langle C, y_{1,2}\rangle\}, \emptyset, \\ &\quad \emptyset, \{\langle C, u_{1,1}\rangle \to \langle 4, u_{1,1}\rangle\}, \{\langle 4, u_{1,1}\rangle \to \langle 1, u_{1,1}\rangle\}, \emptyset\rangle \end{aligned}$$

where $\{\langle a, y\rangle \to \langle b, y\rangle\}$ means that y is sent from node a to node b, and $\emptyset$ means that no message is sent. As a computation schedule (i.e., a sequence of modes of the controller) we choose

$$\langle \text{Idle}, \text{Idle}, \text{Idle}, \text{Idle}, \text{Idle}, \text{Active}, \text{Idle}, \text{Idle}, \text{Idle}, \text{Idle}\rangle.$$

This pair of schedules models the process of sending data from the plant to the controller, computing the control signal, and sending the result of the computation back to the actuator. The finite schedules are assumed to repeat periodically.

Towards a controller design, we first fix the matrices of the controller and leave the values of some entries as symbolic variables (design parameters). We will compute the switched system and apply a pole-assignment computation to set values to the design parameters. Specifically, the dynamics of the controller are defined by the equations $A_c = (K_3); B_c = (K_1, K_2); C_c = (1)$ where K_1, K_2 and K_3 are symbols that we will later assign with scalar numerical values.

To assign values to these parameters, we compute the matrix $\hat{T}_i$ defined in the discussion after Definition 5. Note that this matrix contains numerical values and algebraic expressions with the variables K_1, K_2 and K_3.

Using standard tools such as MATLAB or Mathematica, one can compute the parameters K_1, K_2 and K_3 by assigning the poles of $\hat{T}_i$. Because this matrix models the dynamics of the system through a cycle of the schedule, assigning its eigenvalues to be contained in the unit ball (of the complex plane) assures stability.

Second Example: Verifying Stability Under Non-Deterministic Schedules: As discussed in Section II-J, scheduling in wireless control networks may not be deterministic. To demonstrate another feature of the proposed modeling approach, we consider a system where scheduling constraints vary in time. Specifically, consider the network depicted in Fig. 3 but assume that some times it is possible to send data from both nodes 1 and 2 simultaneously (e.g., when two radio frequencies are available) and some times data has to be sent sequentially (e.g,, when only one frequency is available). Thus, we design two alternatives for a period of the schedule. While both alternatives are stable (as can be verified by computing the eigenvalues of the corresponding $\hat{T}_i(1)$ and $\hat{T}_i(2)$ matrices) it does not necessarily means that any switching between them is stable (see, e.g., [16]). To guaranty stability, we apply a sufficient condition for stability of switched systems to verify that switching arbitrarily between the two schedules is safe. Specifically, we verify that $\|T_i(\sigma(7)) \cdots T_i(\sigma(1))\| < 1$ for every $\sigma \in \{1,2\}^7$ where $T_i(1)$ and $T_i(2)$ are matrices modeling

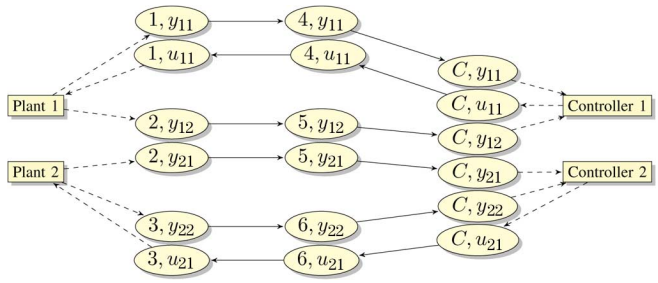


Fig. 9. Memory slots graph of a multi-hop control network with two symmetric double integrators.

the transformation of state variables through the first and the second schedule, respectively. This is, of course, a sufficient condition for stability (even exponential stability) under arbitrary switching, because it implies that every seven steps are contracting. This computation is easily implemented with tools such as MATLAB or Mathematica.

Third Example: Using Compositional Analysis for Schedules Design: One notable advantage of our modeling approach is that, because dynamics are defined for each control loop separately, it allows compositional analysis. As an example, we show how a system comprising of two control loops is analyzed, in a compositional manner, to obtain a joint schedule that renders both loops stable.

Consider the network depicted in Fig. 9. Assume that both plants are double integrators with dynamics and controller as described above (in the first example of this section). Assume also that at most one node can send data at any time slot.

The design approach that we demonstrate in this example is as follows. First, we analyze each control loop separately to obtain a representation of a set of schedules such that the loop is stable. More specifically, we compute an automaton for each control loop whose language (of infinite words) contains only schedules that are "safe" for the control loop, i.e. schedules that will not render the loop unstable. Then, we use formal-languages based algorithms to compute the intersection of the languages and obtain a joint schedule that is "safe" for both control loops.

The following steps can be implemented with numerical computation tools such as MATLAB or Mathematica using a package such as [36] for automata and formal languages manipulation:

1) Compute an automaton for each control loop, as follows. Construct a set of schedules by all possible interleaving of idle steps into a base schedule, and compute an automaton whose language is the interleaved schedules that are stable.
2) Compute an automaton whose language is the intersection of the languages expressed by the two automata. Note that we need to lift the automata to a common alphabet (pairs $\langle \sigma_1, \sigma_2 \rangle$ where σ_1 is a letter from the alphabet of the first automaton and σ_2 is a letter form the alphabet of the second automaton).
3) Choose any word of the intersection automaton (e.g., the first in length-lex order) as a schedule.

The chosen schedule is safe with respect to both control loops, by construction. Note that compositional analysis allows synchronizing node transmissions to send data related to different control loops (plants) simultaneously. This is possible because we lifted the alphabet of the automata to a common alphabet so that a transition is interpreted as sending data related to all the loops at the same time.

Fourth Example: Analysis of Conditions for Almost Sure Stability: We explain how analysis of the type explained in Example 16 can be computed. The first steps are similar to the preceding examples: (1) model the dynamics of the plant, controller, and the schedule; (2) compute the matrix $\hat{T}_i$ leaving some of its entries as (polynomial) expressions in the parameters α, β, and γ; (3) use pole-assignment to find values of the parameters such that $\hat{T}_i$ is stable.

Robustness analysis is done by seeking a natural number m such that $\|(\hat{T}_i)^m\|$ is smaller than one. We can do that. e.g., by applying a small Mathematica script (not shown here) that runs over increasing values of m and, for each m, checks the condition. Note that $\hat{T}_i$ is a fixed matrix, once we assign numerical values to α, β, and γ. The rest of the example can be computed with Mathematica in the same manner.

V. Case Study: A Mineral Floatation Control Problem

In this section, we describe a case study conducted to test how well our approach scales relative to a real system. The case study was to design a wireless control network for an industrial plant that controls mineral processing of ores. A description of the plant and the details of the control parameters is given in [37]. In the context of the present paper, it is enough to say that the part of the plant that we considered contains 17 control loops: 9 *air flow*, 6 *pulp level* and 2 *reagent*. The main "trick" that helped us manage the complexity of the problem was to analyze each control loop in separation and use the compositional approach, outlined in this paper, to design schedules for the integrated system.

Specifically, we abstracted each control loop by a time constraint which specifies the maximum delay between sensing and actuation. For example, this constraint can be chosen to be equal to the sampling interval that the control loop is designed to use. Then, we extended the tools, described in Section IV above, to support a scalable methodology for designing scheduling policies that allow data transmission from the sensors to the controller and from the controller to the actuators for each control loop, within the specified time bounds. One of the advantages of our approach is that we focus on finding the set of all schedules that satisfy the constraints for one control loop (e.g. time bounds) that can later be used for on-line or off-line admission of additional control loops, instead of having to repeat the whole computation whenever a new loop is added.

Let us elaborate the details of the extension to the tool-box that made it possible to complete the case-study. The main feature that we added allows specifications of the form "the data from all sensors has to be sent to the controller, then a computation of the control signals is carried, and after that all the control signals are sent to the actuators". This requirement needs to be satisfied in each period of the schedule, and the designer can also specify an upper time bound for the round trip to complete (e.g. the sampling interval). We found out that we need symbolic representation of automata for answering such queries because explicit representation of automata does not scale well enough. Specifically, to maintain a compact representation of the transition relation we used the NuSMV symbolic model checker [38] which combines Binary Decision Diagrams (BDDs) [39] and

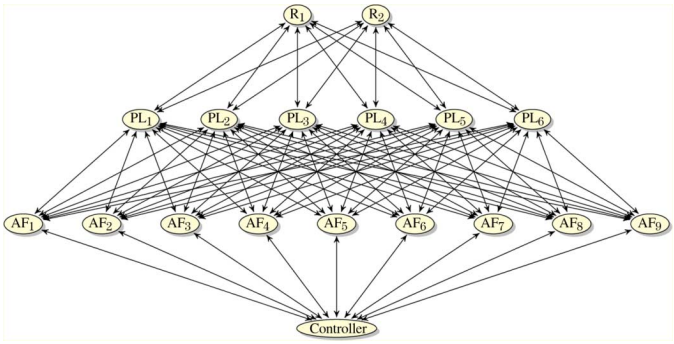


Fig. 10. Topology of wireless control network. R stands for REAGENTS, PL for PULP LEVEL, and AF for AIR FLOW.

SAT based model checking [40]. The tool allows verification of Temporal Logic [41] properties of a transition system expressed in the SMV input language [38]. In practice, we use a Mathematica notebook similar to the approach described in Section IV as a front end. The tool, then, automatically translates the notebook to SMV and invokes NuSMV to analyze the composite transition system resulting from integrating the control loops. NuSMV scans the set of possible schedules and looks for one that meets the needs of all the control loops. The output of this analysis is then translated to a schedule (if one exists) that can be safely applied.

The input to the tool is a textual description of the graph depicted in Fig. 10. As illustrated in the figure, the network has three layers. Each node in each layer is connected to all the nodes in the layer below it. Only the nodes in the third layer can communicate with the controller. All communication links in this case-study are bidirectional. Each wireless node is both a sensor and an actuator of a single-input-single-output plant.

Given a description of the above topology, the tool generates an SMV code. Note that, in principle, we also need to specify a routing path for each signal. In practice however, if a routing path is not explicitly specified, the tool automatically selects a minimal path from the sensor to the controller (for input signals) or from the controller to the actuator (for output signals). The SMV code for the case-study contains seventeen modules, one for each control loop, and a main module.

It takes NuSMV about two minutes (on 2 GHz Intel Core Duo with 1 GB of RAM memory MacBook 1.1) to dispute the claim that there is no schedule that is compatible with all seventeen loops: the NuSMV produces a counter example, from which a valid schedule can be extracted. If we want to look for the schedule with the shortest period, we can add a counter to the main module and add the constraint that the schedule must be smaller than a constant. Starting with the length of the original schedule, we can use binary search to find the minimal such constant. This gives us the shortest admissible schedule, which we do not include in the text since it is a string of length 200 (we recall that the schedule has to allow any allowed routing). A similar procedure can also be used to optimize other properties of the schedule.

VI. Conclusion

The main contributions of this paper can be summarized as follows; (1) we proposed a novel compositional mathematical framework for modeling and analyzing multi-hop control networks that embeds control, network topology, routing, scheduling and transmission errors, and allows co-design of communication parameters and control algorithms for wireless industrial control protocols such as WirelessHART and ISA 100; (2) we addressed and solved robustness of a multi-hop networked control system in the non-deterministic case by worst case analysis of scheduling, routing and packet losses, and in the stochastic case by almost sure stability analysis of node faults probability and packet loss probability.

Future research directions include extending the mathematical results and the tool to support richer specifications (e.g., allow multiple routing) and to incorporate optimization techniques for power minimization of wireless communication and control performance.

Acknowledgment

The authors wish to thank A. Isaksson, P. McLaughlin, and A. Chernoguzov for interesting discussions on WirelessHART and ISA 100 protocols.

References

[1] H. Lin, G. Zhai, and P. Antsaklis, "Robust stability and disturbance attenuation analysis of a class of networked control systems," in *Proc. CDC*, 2003, pp. 1182–1187.

[2] M. Cloosterman, N. van de Wouw, W. Heemels, and H. Nijmeijer, "Robust stability of networked control systems with time-varying network-induced delays," in *Proc. CDC*, 2006, pp. 4980–4985.

[3] L. Shi, M. Epstein, and R. M. Murray, "Towards robust control over a packet dropping network," in *Proc. SMTNS*, 2006, pp. 2130–2138.

[4] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: Research challenges," *Ad Hoc Netw.*, vol. 2, no. 4, pp. 351–367, 2004.

[5] J. Song, S. Han, A. Mok, D. Chen, M. Lucas, M. Nixon, and W. Pratt, "WirelessHART: Applying wireless technology in real-time industrial process control," in *Proc. RTAS*, 2008, pp. 377–386.

[6] J. Song, S. Han, X. Zhu, A. K. Mok, D. Chen, and M. Nixon, "A complete wirelessHART network," in *Proc. ACME*, 2008, pp. 381–382.

[7] R. Alur, A. D'Innocenzo, K. Johansson, G. Pappas, and G. Weiss, "Modeling and analysis of multi-hop control networks," in *Proc. RTAS*, 2009, pp. 223–232.

[8] G. Walsh, H. Ye, and L. Bushnell, "Stability analysis of networked control systems," *IEEE Trans. Control Syst. Technol.*, vol. 10, no. 3, pp. 438–446, May 2002.

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

[10] J. K. Yook, D. M. Tilbury, N. R. Soparkar, E. Syst, and E. S. Raytheon, "Trading computation for bandwidth: Reducing communication indistributed control systems using state estimators," *IEEE Trans. Control Syst. Technol.*, vol. 10, no. 4, pp. 503–518, Jul. 2002.

[11] K. Aström and B. Wittenmark, *Computer-Controlled Systems: Theory and Design*. Englewood Cliffs, NJ: Prentice-Hall, 1997.

[12] M. Donkers, L. Hetel, W. Heemels, N. van de Wouw, and M. Steinbuch, "Stability analysis of networked control systems using a switched linear systems approach," in *Proc. HSCC*, 2009, pp. 150–164.

[13] D. Dacic and D. Nesic, "Quadratic stabilization of linear networked control systems via simultaneous protocol and controller design," *Automatica*, vol. 43, pp. 1145–1155, 2007.

[14] M. Andersson, D. Henriksson, A. Cervin, and K. Arzen, "Simulation of wireless networked control systems," in *Proc. CDC-ECC*, 2005, pp. 476–481.

[15] A. J. van der Schaft and H. Schumacher, *An Introduction to Hybrid Dynamical Systems*, 1st ed. New York: Springer, Dec. 1999.

[16] D. Liberzon, *Switching in Systems and Control*. Boston, MA: Birkhäuser, 2003.

[17] G. Weiss and R. Alur, "Automata based interfaces for control and scheduling," in *Proc. HSCC*, 2007, pp. 601–613.

[18] R. Alur and G. Weiss, "Regular specifications of resource requirements for embedded control software," in *Proc. RTAS*, 2008, pp. 1080–1812.

[19] O. L. V. Costa, R. P. Marques, and M. D. Fragoso, *Discrete-Time Markov Jump Linear Systems*. New York: Springer, 2005.
[20] P. Bolzern, P. Colaneri, and G. D. Nicolao, "On almost sure stability of discrete-time markov jump linear systems," in *Proc. CDC*, 2004, pp. 3204–3208.
[21] Y. Fang, "Stability Analysis of Linear Control Systems With Uncertain Parameters," Ph.D. dissertation, Case Western Reserve U., Cleveland, OH, 1994.
[22] A. L. White, N. L. R. Center, and V. A. Hampton, "Two matrix norm conditions for asymptotic stability in the presenceof controller disturbances," *IEEE Trans. Autom. Control*, vol. 44, no. 1, pp. 169–172, Jan. 1999.
[23] P. Seiler and R. Sengupta, "H infinity approach to networked control," *IEEE Trans. Autom. Control*, vol. 50, no. 3, pp. 356–364, Mar. 2005.
[24] M. Tabbara and D. Nesic, "Input-output stability of networked control systems with stochastic protocols and channels," *IEEE Trans. Autom. Control*, vol. 53, no. 5, pp. 1160–1175, May 2008.
[25] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. Sastry, "Foundations of control and estimation over lossy networks," *Proc. IEEE*, vol. 95, no. 1, pp. 163–187, Jan. 2007.
[26] D. Antunes, J. Hespanha, and C. Silvestre, "Control of impulsive renewal systems: Application to direct design in networked control," in *Proc. CDC*, 2009, pp. 6882–6887.
[27] R. Mercado and K. J. R. Liu, "NP-hardness of the stable matrix in unit interval family problem in discrete time," *Syst. Control Lett.*, vol. 42, pp. 261–265, 2001.
[28] V. Blondel and J. Tsitsiklis, "NP-hardness of some linear control design problems," in *Proc. CDC*, 1995, pp. 2910–2915.
[29] C. V. Ramamoorthy, K. M. Chandy, and M. J. Gonzalez, "Optimal scheduling strategies in a multiprocessor system," *IEEE Trans. Computers*, vol. 21, no. 2, pp. 137–146, Feb. 1972.
[30] Y. K. Kwok and I. Ahmad, "Benchmarking the task graph scheduling algorithms," in *Proc. IPPS*, 1998, pp. 531–537.
[31] A. Bakshi, V. K. Prasanna, J. Reich, and D. Larner, "The abstract task graph: A methodology for architecture-independent programming of networked sensor systems," in *Proc. EESR*, 2005, pp. 19–24.
[32] J. Hopcroft and J. Ullman, *Introduction to Automata Theory, Languages and Computation*. Boston, MA: Addison-Wesley, 1979.
[33] R. Alur and G. Weiss, "RTComposer: A framework for real-time components with scheduling interfaces," in *Proc. EMSOFT*, 2008, pp. 159–168.
[34] TDMA Data-Link Layer Specification HART communication foundation, HCF SPEC 075 Revision 1.0, 2007.
[35] Network Management Specification HART communication foundation, HCF SPEC 085 Revision 1.0, 2007.
[36] K. Sutner, "Automata, a hybrid system for computational automata theory," in *Proc. CIAA*, 2002, pp. 217–222.
[37] A. D'Innocenzo, G. Weiss, R. Alur, A. J. Isaksson, K. H. Johansson, and G. J. Pappas, "Scalable scheduling algorithms for wireless networked control systems," in *Proc. CASE*, 2009, pp. 409–414.
[38] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, "NuSMV version 2: An open-source tool for symbolic model checking," in *Proc. CAV*, 2002, pp. 241–268.
[39] R. E. Bryant, "Symbolic Boolean manipulation with ordered binary-decision diagrams," *ACM Comp. Surveys*, vol. 24, no. 3, pp. 293–318, 1992.
[40] E. Clarke, A. Biere, R. Raimi, and Y. Zhu, "Bounded model checking using satisfiability solving," *Form. Methods Syst. Des.*, vol. 19, no. 1, pp. 7–34, Jul. 2001.
[41] A. Pnueli and Z. Manna, *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Berlin, Germany: Springer-Verlag Gmbh, 1991.

Rajeev Alur (F'08) is Zisman Family Professor of Computer and Information Science at the University of Pennsylvania.

Dr. Alur is an ACM Fellow.

Alessandro D'Innocenzo received the Ph.D. degree and accomplished the International Curriculum Option of Doctoral Studies in Hybrid Control for Complex, Distributed and Heterogeneous Embedded Systems, from the University of L'Aquila, Aquila, Italy, in 2007.

He is Assistant Professor in the Department of Electrical and Information Engineering, University of L'Aquila. He was a Postdoctoral Researcher in the Department of Electrical and Information Engineering, University of L'Aquila, from 2007 to 2009, and in the Department of Electrical and Systems Engineering, University of Pennsylvania, Phildadelphia, in 2008. His research focuses on control theory and in particular, hybrid systems, formal verification and networked control, with applications to air traffic management, automation and communication systems.

Dr. D'Innocenzo received the Fondazione Filauro Award for Ph.D. students in 2005.

Karl H. Johansson (SM'08) received the M.Sc. and Ph.D. degrees in electrical engineering from Lund University, Lund, Sweden, in 1992 and 1997, respectively.

He is Director of the ACCESS Linnaeus Centre and Professor at the School of Electrical Engineering, Royal Institute of Technology, Sweden. He is a Wallenberg Scholar and holds a Senior Researcher Position with the Swedish Research Council. He has held visiting positions at UC Berkeley (1998–2000) and the California Institute of Technology, Pasadena (2006–2007). His research interests are in networked control systems, hybrid and embedded control, and control applications in automotive, automation and communication systems.

Dr. Johansson received an Individual Grant for the Advancement of Research Leaders from the Swedish Foundation for Strategic Research in 2005, the triennial Young Author Prize from IFAC in 1996, the Peccei Award from the International Institute of System Analysis, Austria, in 1993, and the Young Researcher Awards from Scania in 1996 and from Ericsson in 1998 and 1999. He is Chair of the IFAC Technical Committee on Networked Systems. He has served on the Executive Committees of several European research projects in the area of networked embedded systems. He has served on the editorial boards of *Automatica*, the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, and *IET Control Theory & Applications*.

George J. Pappas (F'09) is the Joseph Moore Professor in the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia. He also holds a secondary appointment in the Departments of Computer and Information Sciences, and Mechanical Engineering and Applied Mechanics. He is member of the GRASP Lab and the PRECISE Center. He currently serves as the Deputy Dean for Research in the School of Engineering and Applied Science. His research focuses on control theory and in particular, hybrid systems, embedded systems, hierarchical and distributed control systems, with applications to unmanned aerial vehicles, distributed robotics, green buildings, and biomolecular networks.

Dr. Pappas received various awards such as the Antonio Ruberti Young Researcher Prize, the George S. Axelby Award, and the National Science Foundation PECASE.

Gera Weiss is an Assistant Professor in the Department of Computer Science, Ben Gurion University of the Negev, Israel.