The Wireless Control Network: Synthesis and Robustness

Miroslav Pajic  
*University of Pennsylvania*, pajic@seas.upenn.edu

Shreyas Sundaram  
*University of Waterloo*, ssundara@uwaterloo.ca

Jerome Le Ny  
*University of Pennsylvania*, jerome.le-ny@polymtl.ca

George J. Pappas  
*University of Pennsylvania*, pappasg@seas.upenn.edu

Rahul Mangharam  
*University of Pennsylvania*, rahulm@seas.upenn.edu

Follow this and additional works at: [https://repository.upenn.edu/mlab_papers](https://repository.upenn.edu/mlab_papers)

Part of the [Controls and Control Theory Commons](https://repository.upenn.edu/mlab_papers), [Digital Communications and Networking Commons](https://repository.upenn.edu/mlab_papers), and the [Systems and Communications Commons](https://repository.upenn.edu/mlab_papers)

**Recommended Citation**


**Suggested Citation:**


©2010 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

This paper is posted at ScholarlyCommons. [https://repository.upenn.edu/mlab_papers/19](https://repository.upenn.edu/mlab_papers/19)  
For more information, please contact repository@pobox.upenn.edu.
The Wireless Control Network: Synthesis and Robustness

Abstract
We consider the problem of stabilizing a plant with a network of resource constrained wireless nodes. Traditional networked control schemes are designed with one of the nodes in the network acting as a dedicated controller, while the other nodes simply route information to and from the controller and the plant. We introduce the concept of a Wireless Control Network (WCN) where the entire network itself acts as the controller. Specifically, at each time-step, each node updates its internal state to be a linear combination of the states of the nodes in its neighborhood. We show that this causes the entire network to behave as a linear dynamical system, with sparsity constraints imposed by the network topology. We then provide a numerical design procedure to determine the appropriate linear combinations to be applied by each node so that the transmissions of the nodes closest to the actuators will stabilize the plant. We also show how our design procedure can be modified to maintain mean square stability under packet drops in the network.

Keywords
Wireless Network Control, Control Systems, Network Control Systems, Cyber-Physical Systems

Disciplines
Controls and Control Theory | Digital Communications and Networking | Systems and Communications

Comments
Suggested Citation:

©2010 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.
The Wireless Control Network: Synthesis and Robustness

Miroslav Pajic, Shreyas Sundaram, Jerome Le Ny, George J. Pappas and Rahul Mangharam

Abstract—We consider the problem of stabilizing a plant with a network of resource constrained wireless nodes. Traditional networked control schemes are designed with one of the nodes in the network acting as a dedicated controller, while the other nodes simply route information to and from the controller and the plant. We introduce the concept of a Wireless Control Network (WCN) where the entire network itself acts as the controller. Specifically, at each time-step, each node updates its internal state to be a linear combination of the states of the nodes in its neighborhood. We show that this causes the entire network to behave as a linear dynamical system, with sparsity constraints imposed by the network topology. We then provide a numerical design procedure to determine the appropriate linear combinations to be applied by each node so that the transmissions of the nodes closest to the actuators will stabilize the plant. We also show how our design procedure can be modified to maintain mean square stability under packet drops in the network.

I. INTRODUCTION

The advent of low-cost and reliable wireless networks holds great promise for large, spatially distributed industrial control systems. In contrast to the traditional wired interconnections that exist in such systems, wireless networks allow sensor measurements of plant variables to be transmitted to controllers, data centers and plant operators without the need for excessive wiring, thereby yielding gains in efficiency and profitability for the operator.

The topic of control over networks has been intensively studied by researchers over the past decade, leading to design procedures for controllers that are tolerant to network issues such as packet dropouts and transmission delays [1], [2], [3], [4], [5], [6]. These works typically adopt the convention of having a dedicated controller/estimator located somewhere in the network, and study the stability of the closed loop system assuming that the sensor-estimator and/or controller-actuator communication channels are unreliable (dropping packets with a certain probability, for example).

In this paper, we introduce the concept of a Wireless Control Network (WCN), which is a paradigm change for control over a wireless network. In a WCN the entire network itself acts as a controller, as the computation of the control input is spread over the whole network. We consider a setup where several resource constrained wireless nodes are deployed in the proximity of a plant, with some nodes having access to the sensor measurements (outputs) of the plant, and some nodes placed within the listening range of the plant’s actuators. Each node in the WCN is capable of maintaining only a limited internal state. We present a linear iterative strategy for each node to follow, where each node periodically updates its state to be a linear combination of the states of the nodes in its immediate neighborhood. The actuators of the plant also apply linear combinations of the states of the nodes in their neighborhood. Given a linear plant model and the topology of the wireless network, we devise a numerical design procedure that produces the coefficients of the linear combinations for each node and actuator to apply in order to stabilize the plant.

A. Notation

We use $e_i$ to denote the $i^{th}$ unit column vector (of appropriate dimension) and the symbol $1$ denotes the column vector (of appropriate size) consisting of all $1$’s. The symbol $I_N$ denotes the $N \times N$ identity matrix. The notation $\text{diag} \{ \cdot \}$ indicates a square matrix with the quantities inside the brackets on the diagonal, and zeros elsewhere. The notation $\text{tr} \{ \cdot \}$ indicates the trace of a square matrix. We will denote the cardinality of a set $S$ by $|S|$. The set of nonnegative integers is denoted by $\mathbb{N}$. The notation $A \succeq 0 (\succeq 0)$ indicates that matrix $A$ is positive (semi)definite. The set of all $n \times n$ positive definite matrices is denoted by $\mathbb{S}^n_{++}$. A graph $G$ is denoted by an ordered pair $G = \{ \mathcal{V}, \mathcal{E} \}$, where $\mathcal{V} = \{ v_1, v_2, \ldots, v_N \}$ is a set of vertices (or nodes), and $\mathcal{E}$ is a set of ordered pairs of different vertices, called directed edges. The vertices in the set $N_{v_i} = \{ v_j \mid (v_j, v_i) \in \mathcal{E} \}$ are said to be neighbors of vertex $v_i$.

II. THE WIRELESS CONTROL NETWORK

Consider the system presented in Fig. 1, where the plant is controlled using a multi-hop, fully synchronized wireless network. In this paper we focus on plants of the form

\begin{equation}
\begin{aligned}
x[k+1] &= Ax[k] + Bu[k] \\
y[k] &= Cx[k],
\end{aligned}
\end{equation}

with $A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{p \times n}$. The output vector $y[k] = [y_1[k] \ y_2[k] \ \ldots \ y_p[k]]^T$ contains measurements of the plant state vector $x[k]$ provided by the sensors $s_1, \ldots, s_p$. The input vector $u[k] = [u_1[k] \ u_2[k] \ \ldots \ u_m[k]]^T$ corresponds to the signals applied to the plant by actuators $a_1, \ldots, a_m$.

The plant model can be generalized to include update and measurement noise; if the noise is taken to be independent and identically distributed with a bounded variance, all of our analysis and results will still ensure that the system state is bounded in a mean square sense. For the purposes of clarity, we will therefore omit the noise terms in our discussion.
The WCN consists of a set of nodes that communicate with each other and with the sensors and actuators installed on the plant. Each node in the network is equipped with a radio transceiver along with (limited) memory and computational capabilities. Similarly, each sensor and actuator on the plant contains a radio transceiver, allowing them to communicate with neighboring nodes. The wireless network is described by a graph $\mathcal{G} = \{V, \mathcal{E}\}$, where $V = \{v_1, v_2, \ldots, v_N\}$ is the set of $N$ nodes and $\mathcal{E} \subseteq V \times V$ represents the radio connectivity (communication topology) in the network (i.e., edge $(v_i, v_j) \in \mathcal{E}$ if node $v_i$ can receive information directly from node $v_j$). We also define $\mathcal{V}_s \subset V$ as the set of nodes that can receive information directly from at least one sensor, and $\mathcal{V}_a \subset V$ as the set of nodes whose transmissions can be heard by at least one actuator.

To facilitate our development, we consider a new graph $\hat{\mathcal{G}}$ that includes the plant’s sensors and actuators. This graph is obtained by taking the graph $\mathcal{G}$ and adding $p + m$ new vertices $\mathcal{S} \cup \mathcal{A}$, where $\mathcal{S} = \{s_1, s_2, \ldots, s_p\}$ corresponds to the plant’s sensors, while $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$ corresponds to the plant’s actuators. Define the edge sets:

$$\mathcal{E}_O = \left\{ (s_i, v_j) \mid s_i \in \mathcal{S}, v_j \in \mathcal{V}_s, \text{ and } v_j \text{ can receive values from sensor } s_i \right\},$$

$$\mathcal{E}_I = \left\{ (v_i, a_l) \mid a_l \in \mathcal{A}, v_i \in \mathcal{V}_a, \text{ and } a_l \text{ can receive values from } v_i \right\}.$$

We then obtain $\hat{\mathcal{G}} = \{V \cup \mathcal{S} \cup \mathcal{A}, \mathcal{E}_O \cup \mathcal{E}_I \cup \mathcal{E}_O\}$. Let $\mathcal{N}_I$ denote the number of links in $\hat{\mathcal{G}}$ ($\mathcal{N}_I = |\mathcal{E}_O \cup \mathcal{E}_I \cup \mathcal{E}_O|$). We can also define an injective mapping $\Omega : \mathcal{E}_O \cup \mathcal{E}_I \cup \mathcal{E}_O \to \{1, \ldots, \mathcal{N}_I\}$ to enumerate all links in the network. In the rest of the paper, we will sometimes denote a link $(a, b) \in \mathcal{E}_O \cup \mathcal{E}_I \cup \mathcal{E}_O$ by its label $\Omega(a, b)$ for convenience.

Unlike traditional networked control schemes where a particular node $v_i \in V$ is designated as the controller (and all other nodes are used to route information between $v_i$ and the plant), the WCN employs a fully distributed control scheme where the entire network itself acts as a controller. At each time-step, every node in the WCN updates its value to be a linear combination of its previous value and the values of its neighbors. In addition, the update procedure of each node from the set $\mathcal{V}_s$ includes a linear combination of the sensor measurements (i.e. plant outputs) from all sensors in its neighborhood. If we let $z_i[k]$ denote node $v_i$’s (scalar) state at time step $k$, we obtain the update procedure:

$$z_i[k+1] = w_{ii}z_i[k] + \sum_{v_j \in \mathcal{N}_i} w_{ij}z_j[k] + \sum_{s_j \in \mathcal{N}_s} h_{ij}s_j[k].$$

Each plant input $u_i[k], i \in \{1, \ldots, m\}$ is taken to be a linear combination of values from the nodes in the neighborhood of the actuator $a_i$:

$$u_i[k] = \sum_{j \in \mathcal{N}_a} g_{ij}z_j[k].$$

The scalars $w_{ij}, h_{ij}$ and $g_{ij}$ specify the linear combinations that are computed by each node and actuator in the network. If we aggregate the values of all nodes at time step $k$ into the value vector $z[k] = [z_1[k], z_2[k], \ldots, z_N[k]]^T$, the linear iterative procedure for the entire system can be described as:

$$z[k+1] = Wz[k] + Hy[k],$$

$$u[k] = Gz[k]$$

for all $k \in \mathbb{N}$ ($W \in \mathbb{R}^{N \times N}$, $H \in \mathbb{R}^{N \times p}$, $G \in \mathbb{R}^{m \times N}$). In the above equation, for all $i \in \{1, \ldots, N\}$, $u_{ij} = 0$ if $v_j \notin \mathcal{N}_i \cup \{v_i\}$, $h_{ij} = 0$ if $s_j \notin \mathcal{N}_i$, and $g_{ij} = 0$ if $v_j \notin \mathcal{N}_a$. Thus the matrices $W, H$ and $G$ are structured, meaning that they have sparsity constraints determined by the topology of the WCN. Throughout the rest of the paper, we will define $\Psi$ to be the set of all tuples $(W, H, G) \in \mathbb{R}^{N \times N} \times \mathbb{R}^{N \times p} \times \mathbb{R}^{m \times N}$ satisfying the aforementioned sparsity constraints. If we denote the overall system state by $\hat{x}[k] = [x[k]^T \ z[k]^T]^T$, the closed-loop system evolves as:

$$\hat{x}[k+1] = \begin{bmatrix} A & BG & HC & W \end{bmatrix} \begin{bmatrix} x[k] \ z[k] \end{bmatrix} \triangleq \hat{A} \hat{x}[k].$$

In this paper we describe an algorithm that can be used to find an element of $\Psi$ that causes the matrix $A$ to be Schur (provided such an element exists). We first consider the case with reliable communication links, and then extend our approach to accommodate independent Bernoulli link failures in the network. In the companion paper [8], we also show how to design an “Intrusion Detection System” for this control scheme, which observes the transmissions of certain nodes in order to identify any abnormal behavior.

### A. Advantages of Wireless Control Networks

In the context of multi-hop embedded wireless networks used for control, the WCN has the following advantages.

1. **Low overhead:** The proposed scheme is computationally inexpensive since a node only needs to compute a linear combination of its value and values of its neighbors. Thus, the WCN can be easily implemented even on resource constrained, low-power wireless nodes (e.g., FireFly, tMote, micaZ) using very simple, periodic tasks executed on a real-time operating system (e.g., nano-RK [9] or TinyOS [10]). Also, as each node is only required to transmit its state once per frame, the proposed scheme can be easily “piggy-backed” into existing wireless networks that assign a transmission.

2. **We will model these resource constraints by limiting the size of the state vector maintained by each node. To present our results, we will focus on the case where each node’s state is represented as a scalar. Our results can be readily extended to the more general case where each node can maintain a vector state with possibly different dimensions. For details see [7].

3. **The neighborhood $\mathcal{N}_v$ of a vertex $v$ is with respect to the graph $\hat{\mathcal{G}}$.**

4. **A matrix is Schur if all of its eigenvalues are inside the unit circle.**
slot for each node to maintain network related information (e.g., wireless systems for factory automation based on the ISA100.11a standard [11] or wirelessHART [12]). This also allows the possibility of using the proposed scheme as a backup mechanism in traditional networked control systems. Specifically, if the primary control mechanism (i.e., dedicated controller) in the existing networked control infrastructure fails, the wireless network itself can take over the role of stabilizing the plant until the primary controller is restored.

2. Compositionality: The WCN allows compositionality, meaning that an existing design can be easily extended to control new subsystems that are added to the plant. In subsequent sections we describe how the WCN can be used to stabilize a given plant (or set of interconnected plants). However, suppose that some new subsystems (or plants) are added in the proximity of the WCN. Rather than recalculating a stabilizing set of linear combinations that would work for all plants simultaneously, one can instead calculate a separate stabilizing set of linear combinations for each of the new plants, with corresponding separate states maintained by each node. If $P$ is the total number of different plants, each node would calculate $P$ linear combinations of the values received from its neighbors and group the $P$ corresponding states in one transmission per frame. This enables a completely decoupled computation of the matrices $(\{W_i, H_i, G_i\})_{i=1}^P$, each guaranteeing MSS for each of the $P$ plants, although physically realized by the same WCN.

3. Simple scheduling: The presented scheme does not require complex communication scheduling, since each node needs to transmit exactly once in a frame and the WCN does not impose end-to-end delay constraints (i.e., nodes close to the actuators do not need to wait for information to propagate all the way from the sensors). The only requirement of the communication schedule is to be conflict-free (i.e., two transmission scheduled at the same slot should not affect each other). Thus, if $d_i$ is the maximal degree of the interference graph, a static conflict-free schedule can be derived using graph coloring, with at most $d_i$ slots in a frame. Since the duration of a frame is equal to the plant’s sampling period, the minimal sampling period of the plant is equal to $d_i T_{slot}$, where $T_{slot}$ is the duration of a communication slot. In contrast, some techniques used for traditional networked control systems impose a requirement that the sampling period be greater than the end-to-end delay, causing the minimal sampling period to directly depend on the network diameter.

4. Multiple sensing/actuation points: The WCN can handle plants with multiple geographically distributed sensors and actuators, a case that is not easily handled by the “sensor → channel → controller/estimator → channel → actuator” setup that is commonly adopted in networked control design.

III. STABILIZING THE CLOSED-LOOP SYSTEM

From Eq. (4), the closed-loop system is stable if the matrices $A, W, H$ are chosen so that $\hat{A}$ is Schur. The traditional approach to achieving this would be to find a positive definite matrix $X$ satisfying the Lyapunov inequality $X - \hat{A}^T X \hat{A} > 0$, or equivalently, $[X \hat{A}^T \hat{A}] > 0$. The condition is not linear in the design parameters $X, W, H, G$: this is of no consequence in standard controller design (without structural constraints on the design matrices), because this condition can be converted to a LMI via an appropriate transformation of the system matrices (e.g., as done in [13]). However, the fact that the matrices are structured in our framework prevents us from directly applying these standard procedures. Still, the following alternative characterization of stability of structured systems from [14] offers a solution.

Theorem 1: ([14]) A matrix $\hat{A}$ is Schur iff there exist symmetric, positive-definite matrices $X$ and $Y$ such that $[X \hat{A}^T \hat{A}] > 0, X = Y^{-1}$.

The theorem provides a matrix inequality that is linear in the design variables $W, G$ and $H$, but suffers from the fact that the constraint $X = Y^{-1}$ is nonconvex. However, as pointed out in [14], constraints of this form commonly occur in the design of static output feedback controllers, and there are various numerical methods to address this issue. One particularly appealing approach, suggested in [15], [16], is to approximate the constraint $X = Y^{-1}$ with a linear optimization problem using the following lemma (the proof can be obtained in [7]).

Lemma 1: Positive-definite matrices $X, Y$ satisfy the constraint $X = Y^{-1}$ iff they are optimal points for the problem

$$P: \quad \min tr(X Y), \quad s.t. \ X \succeq Y^{-1}, \ X, Y \in S^+_n$$

and the optimal cost of the problem is $n$.

Using the Schur complement, the constraint $X \succeq Y^{-1}$ in the above lemma can be readily transformed to the form $[X 1 \ 1] \succeq 0$. From Eq. (4), Theorem 1 and Lemma 1 the following corollary can be obtained.

Corollary 1: The WCN can stabilize the system if and only if the following optimization problem

$$\min tr(X Y),$$

$$\begin{bmatrix} X & \hat{A}^T \\ \hat{A} & Y \end{bmatrix} \succeq 0, \quad \begin{bmatrix} X & I \\ I & Y \end{bmatrix} \succeq 0,$$

$$\hat{A} = \begin{bmatrix} A & B G \\ H C & W \end{bmatrix},$$

$$(W, H, G) \in \Psi, \ X, Y \in S^+_n \in S^+_n$$

is feasible with optimal cost $n + N$.

Note that with the exception of the objective function (5), all of the constraints in the above corollary are linear in the unknown parameters, and can readily be solved using LMI tools. In [16], El Ghaoui et al. showed that the nonconvex function $tr(X Y)$ can be replaced with a linear approximation

$$\phi_{lin}(X, Y) = constant + tr(Y_0 X + X_0 Y),$$

This is usually not a limitation even for low-bandwidth 802.15.4 networks (where each transmission packet can contain up to 1024 bits). For instance, if a node maintains a scalar 32 bit state for each plant, then up to 32 plants can be controlled in parallel.

The interference graph is defined as $\mathcal{G}_{int} = \{Y \cup S \cup A, E_{int}\}$, where a link between two nodes (or a node and a sensor/actuator) indicates that they can interfere with each other (i.e., cannot transmit simultaneously).
for any given matrices $X_0$ and $Y_0$. With this insight, [15], [16] showed that an iterative algorithm can be used to minimize $\text{tr}(XY)$, while ensuring satisfaction of LMI constraints. For our application, the iterative approach proposed in those papers can be formulated as Algorithm 1.

**Algorithm 1 Stabilizing closed-loop system with the WCN**

1. Find feasible points $X_0, Y_0, W_0, H_0, G_0$ that satisfy the constraints (6)-(8). If a feasible point does not exist, then it is not possible to stabilize the system with this network topology.

2. At iteration $k$ ($k \geq 0$), from $X_k, Y_k$ obtain the matrices $X_{k+1}, Y_{k+1}, W_{k+1}, H_{k+1}, G_{k+1}$ by solving the following LMI problem

   $$\min \text{tr}(Y_k X_{k+1} + X_k Y_{k+1})$$

   $$\begin{bmatrix} X_{k+1} & I \\ A_{k+1} & Y_{k+1} \end{bmatrix} \succ 0,$$

   $$\hat{A}_{k+1} = \begin{bmatrix} A & BG_{k+1} \\ H_{k+1}C & W_{k+1} \end{bmatrix},$$

   $$\text{if} \ (W_{k+1}, H_{k+1}, G_{k+1}) \in \Psi, \ X_{k+1}, Y_{k+1} \in S^{n+N}.$$  

3. If the matrix

   $$\hat{A}_{k+1}$$

   is Schur, stop the algorithm. Otherwise, set $k = k + 1$ and go to the step 2.

In [16] the authors showed that the sequence $t_k = \text{tr}(Y_k X_{k+1} + X_k Y_{k+1})$ always converges. In addition, if it converges to $2(n + N)$ the condition $Y = X^{-1}$ can be satisfied under the given LMI constraints. A similar proof can be constructed in this case, leading to the following theorem.

**Theorem 2:** Algorithm 1 determines a tuple $(W, H, G) \in \Psi$ that causes the matrix $A(W, H, G)$ to be Schur if the sequence $t_k$ converges to $2(n + N)$.

While each iteration of the above algorithm is a convex optimization problem (which can be efficiently solved using standard LMI toolboxes), we do not have a characterization of the number of iterations required for the algorithm to converge.

**IV. STABILIZATION DESPITE UNRELIABLE COMMUNICATION LINKS**

In this section we focus on more “realistic” system models, where potential message drops are taken into consideration. In this case the system’s evolution can be described as

$$\tilde{x}[k + 1] = \begin{bmatrix} H_{wk}C & W_{wk} \end{bmatrix} \hat{x}[k],$$

where $\hat{x}[k] \in \mathbb{R}^{n+N}$ is the overall system’s state and the subscript $\theta(k)$ describes time-variations in the matrices $(W, H, G)$ caused by (probabilistic) drops of communication packets. The focus of this section is a design procedure that can guarantee mean-square stability (MSS) of the closed loop system, defined as:

**Definition 1:** ([17]) The system is mean-square stable if for any initial state $\tilde{x}[0]$, $\lim_{k \to \infty} \mathbb{E}\left[\|\tilde{x}[k]\|^2\right] = 0$, where the expectation is with respect to the probability distribution of the packet drop sequence $\theta(k)$.

There are relatively few results that explicitly consider packet drops in networked control systems with general topologies. The paper [5] considered the problem of the optimal location for a controller in a network, and the papers [6], [18] considered the issue of allowing intermediate nodes to encode information that they are routing to the controller. All of these papers assume a single sensor and actuation point on the plant, consider the existence of a designated controller within the network, and focus on the issue of transmitting the sensor measurements (or some function of them) to that controller.

The topic of modeling networks with unreliable channels was also considered in [17], where it was shown that such networks can be cast in a robust control framework. In the framework of [17], a communication link is modeled over time as a memoryless, discrete, independent and identically distributed (IID) random process $\xi_t^7$ which maps each transmitted value $t_x[k]$ into a received value $r_x[k] = \xi_t^7 t_x[k].^8$

For arbitrary nodes $v_i$ and $v_j$ consider a communication link $(v_i, v_j) \in \mathcal{E}$ with weight $w_{ij}$ (as shown in Fig. 2(a)). In the rest of the paper we will also denote this link as $t = \Omega(v_i, v_j)$ and its weight as $w_t, h_t$ or $g_t$. In addition, all variables related to the link will be denoted with index $t$ (e.g. $\xi_t^7$, instead of $\xi_{ij}^7[k]$). The contribution of the node $v_t$ to the linear combination calculated by node $v_j$ at time $k$ can be represented as $w_t \xi_t^7 t_j^2 [v_j] [v_i]$, where $\xi_t^7$ has mean $\mu_t = \mathbb{E}[\xi_t^7[k]]$ and a finite variance $\sigma_t^2 = \mathbb{E}[(\xi_t^7[k] - \mu_t)^2]$.

Following the approach in [17], we consider the link transformation shown in Fig. 2(b). By writing $\xi_t^7[k] = \mu_t + \Delta_t^k$, where $\Delta_t^k$ is a zero-mean random variable with variance $\sigma_t^2$, the original unreliable link is modeled as a combination of the deterministic link (without message drops) with gain $\mu_t$ and the random link described with gain $\Delta_t^k$. Let $r_t[k]$ denote the signal that is injected into the $t^{th}$ link, scaled by the weight on that link:

$$r_t[k] = \begin{cases} h_t y_t[k] & \text{if } t = \Omega(v_i, v_j), \\
 w_t z_t[k] & \text{if } t = \Omega(v_i, v_j), \\
 g_t z_t[k] & \text{if } t = \Omega(v_i, a_j). \end{cases}$$

Stacking all of the $r_t[k]$’s in a vector $\mathbf{r}[k]$ of length $N_t$, we can write

$$\mathbf{r}[k] = J^\text{or} \begin{bmatrix} y[k] \\
 z[k] \end{bmatrix} = J^\text{or} \begin{bmatrix} 0 & 0 & 1_N \end{bmatrix} \hat{x}[k], \quad (9)$$

7Here IID implies that the random variables $[\xi_t^7[k]]_{k \geq 0}$ are IID.
8Note that a Bernoulli packet drop channel can be modeled by setting $\xi_t^7 = 0$ with probability $p$ and 1 with probability $1 - p$. 

Fig. 2. Remote control over fading channel; (a) A link between nodes $v_i$ and $v_j$; (b) Link transformation into a robust control form.
where each row of the matrix $J^{or} \in \mathbb{R}^{N_x \times (N_p+1)}$ contains a single nonzero element, equal to a gain $w_j, h_j$ or $g_j$.

Based on the link transformation shown in Fig. 2(b), and using (2), the update equation for each node $v_j$ is

\[
z_j[k+1] = w_j z_j[k] + \sum_{t=0}^{\Omega(v_j,v_j)} \mu_{j,t} z_j[k] + \sum_{t=\Omega(v_j,v_j)+1}^{\Omega(v_j,v_j)} \mu_{h,t} y_j[k] \\
+ \sum_{t=\Omega(v_j,v_j)+1}^{\Omega(v_j,v_j)} \Delta_i[k] r_i[k] + \sum_{t=\Omega(v_j,v_j)+1}^{\Omega(v_j,v_j)} \Delta_j[k] r_j[k].
\]

Also, from (3), the input value applied by each actuator at time step $k$ is

\[
w_j[k] = \sum_{t=\Omega(v_j,v_j)}^{\Omega(v_j,v_j)} \mu_{j,t} g_j z_j[k] + \sum_{t=\Omega(v_j,v_j)}^{\Omega(v_j,v_j)} \Delta_i[k] r_i[k].
\]

Let $\Delta[k] = \text{diag}((\Delta_i[k])_{i=1}^{N_i})$, so that the above expressions can be written in vector form as

\[
z[k+1] = \mathbf{W}_u z[k] + \mathbf{H}_u y[k] + J^{dist}(\Delta[k] r[k]), \\
\mathbf{u}[k] = \mathbf{G}_u z[k] + J^{dist}(\Delta[k] r[k]),
\]

where each nonzero entry of matrices $\mathbf{W}_u, \mathbf{H}_u$ and $\mathbf{G}_u$ (except the diagonal entries of $\mathbf{W}_u$) is of the form $\mu_{j,t} w_j, \mu_{h,t}$ and $\mu_{j,t} g_j$, respectively. Each entry in the matrices $J^{dist}$ and $J^{dist}$ is either 0 or 1. Specifically, each row of those matrices simply selects which elements of the vector $\Delta[k] r[k]$ are added to the linear combinations calculated by the actuators and the wireless nodes. From this, the overall system (with potential message drops) can be represented as:

\[
\mathbf{x}[k+1] = \begin{bmatrix}
\mathbf{A} & \mathbf{B} \\
\mathbf{G} & \mathbf{H}
\end{bmatrix} \mathbf{x}[k] + \begin{bmatrix}
\mathbf{0} \\
\mathbf{0}
\end{bmatrix} \mathbf{r}[k],
\]

with $J^{dist} = \begin{bmatrix} J^{dist} \\ J^{dist} \end{bmatrix} \in \mathbb{R}^{(m+N \times N_i)}$. As in Section III, Algorithm 2 can be constructed to solve the inequalities presented in the above theorem (since the matrix $J^{dist}$ and $\sigma_i$ are constants).

Theorem 4: Algorithm 2 will determine the tuple $(\mathbf{W}, \mathbf{G}, \mathbf{H}) \in \Psi$ that guarantees MSS of the system under the given links’ failure distribution if the sequence $t_k = tr(\mathbf{X}, \mathbf{X}, \mathbf{X}, \mathbf{X})$ converges to $2(n+1)$. \qed

Remark 1: If we consider all links to have the same $\sigma_i = \sigma$, the largest value of packet loss for which the system is MSS can be found by allowing $\sigma \in \mathbb{R}$ to be a variable. This causes the last matrix inequality in step 2 of Algorithm 2 to be a bilinear constraint, but this can be handled by using bisection on the parameter $\sigma \in \mathbb{R}$ (e.g., as done in [13]). \qed

Algorithm 2: Stabilizing the closed-loop system with unreliable communication links using the WCN

1. Find feasible points $X_0, Y_0, W_0, H_0, G_0$ that satisfy the constraints (11), where $X_0, Y_0 \in \mathbb{S}^{n+N}_{++}$ and:

\[
\begin{bmatrix}
X_0 & I \\
I & Y_0
\end{bmatrix} \succeq 0, \quad (W_0, H_0, G_0) \in \Psi
\]

If there is no feasible point, it is not possible to obtain MSS with this network topology and distribution on the communication links.

2. At iteration $k$, $(k \geq 0)$ from $X_k, Y_k$ obtain the matrices $X_{k+1}, Y_{k+1}, W_{\mu,k+1}, H_{\mu,k+1}, G_{\mu,k+1}$ and a vector $\alpha_{k+1} \in \mathbb{R}^{N_i}$ by solving the following LMI problem

\[
\min tr(Y_{k+1}X_{k+1} + X_{k+1}Y_{k+1})
\]

\[
X_{k+1} - (\hat{J}^{dist} \text{diag}(\alpha_{k+1})(\hat{J}^{dist})^T \hat{A}_{\mu,k+1} Y_{k+1}) \geq 0,
\]

\[
A_{\mu,k+1} = \begin{bmatrix}
\mathbf{A} & \mathbf{B} \\
\mathbf{G} & \mathbf{H}
\end{bmatrix} \mathbf{W}_{\mu,k+1} \mathbf{C} \mathbf{W}_{\mu,k+1}^T,
\]

\[
\sigma_i(j^{dist}_{i,k+1} \mathbf{J}_{\mu,k+1}^T Y_{k+1}) \geq 0, \quad 1 \leq i \leq N_i,
\]

\[
(\mathbf{W}_{\mu,k+1}, \mathbf{H}_{\mu,k+1}, \mathbf{G}_{\mu,k+1}) \in \Psi, \quad X_{k+1}, Y_{k+1} \in \mathbb{S}^{n+N}_{++}
\]

3. Stop the algorithm if the following conditions are true

\[
X_{k+1} \succeq \hat{A}_{\mu,k+1} X_{k+1} \hat{A}_{\mu,k+1}^T + J^{dist} \text{diag}(\alpha_{k+1})(\hat{J}^{dist})^T
\]

\[
\sigma_i(j^{dist}_{i,k+1} \mathbf{J}_{\mu,k+1}^T + \hat{J}^{dist} \text{diag}(\alpha_{k+1})(\hat{J}^{dist})^T)
\]

\[
\alpha_{i,k+1} \geq \sigma_i(j^{dist}_{i,k+1} \mathbf{J}_{\mu,k+1}^T \mathbf{J}_{\mu,k+1}^T)^{1/2}, \quad 1 \leq i \leq N_i
\]

Otherwise, set $k = k+1$ and go to step 2.

V. DISCUSSION AND EXAMPLE

At first glance, the control scheme that we have presented in this paper might seem to introduce some delay into the feedback loop (since the sensor nodes and actuator nodes might be separated by multiple intermediate nodes, each taking one time-step to propagate information), which might limit the class of plants that can be stabilized with this method. However, the relationship between the WCN and the traditional notions of delay introduced by the feedback loop is not as obvious as it might appear at first glance. Specifically, note that we allow each node in the network to maintain a value that is a function of its previous value and the values of all its neighbors, rather than simply routing values to a controller. This simple modification causes the network to essentially act as a linear dynamical system with sparsity constraints in the system matrices; in other words, this control scheme should be viewed as a dynamic compensator, rather than a static feedback gain at the end of a chain of delay elements. The following example shows that this fact allows our scheme to stabilize plants that cannot be stabilized with delayed static feedback.

Consider the single-state plant shown in Fig. 3 (with $\alpha >$
The graph $G = (V, E)$ is complete ($V = \{v_1, \ldots, v_N\}$). In these cases maximal probabilities are $p_{\text{max}} = 0.74\%$ and $p_{\text{max}} = 0.77\%$ respectively. As can be seen, adding more nodes in the network increases the robustness of the system to packet drops in the wireless network.

REFERENCES


