

Data Transmission over Networks for Estimation and Control

Vijay Gupta, *Student Member, IEEE*, Amir F. Dana, *Student Member, IEEE*, João P. Hespanha, *Senior Member, IEEE*, Richard M. Murray, *Fellow, IEEE* and Babak Hassibi *Senior Member, IEEE*

Abstract

We consider the problem of controlling a linear time invariant process when the controller is located at a location remote from where the sensor measurements are being generated. The communication from the sensor to the estimator is supported by a communication network with arbitrary topology composed of channels that stochastically drop packets. Using a separation principle, we prove that the optimal LQG controller consists of an LQ optimal regulator along with an estimator that estimates the state of the process across the communication network mentioned above. We then determine the optimal information processing strategy that should be followed by each node in the network so that the estimator is able to compute the best possible estimate in the minimum mean squared error sense. The algorithm is optimal for any packet-dropping process and at every time step, even though it is recursive and hence requires a constant amount of memory, processing and transmission at every node in the network per time step. For the case when the packet drop processes are memoryless and independent across links, we analyze the stability properties and the performance of the closed loop system. The algorithm is an attempt to escape the more commonly used viewpoint of treating a network of communication links as a single end-to-end link with the probability of successful transmission determined by some measure of the reliability of the network.

I. INTRODUCTION

Recently a lot of attention has been directed towards networked control systems in which components communicate over wireless links or communication networks that may also be used for transmitting other unrelated data (see, e.g., [1], [3], [14] and the references therein). The estimation and control performance in such systems is severely affected by the properties of the communication channels. Communication links introduce many potentially detrimental phenomena, such as quantization error, random delays, data loss, data corruption to name a few, that lead to performance degradation or even stability loss.

Research supported in part by AFOSR grant F49620-01-1-0460 and by NSF grant CCR-0326554 for the first author and by the National Science Foundation under grant no. CCR-0133818, by David and Lucille Packard Foundation and by Caltech's Lee Center for Advanced Networking for the second author.

V. Gupta is with the Institute of Systems Research, University of Maryland, College Park. vijay@cds.caltech.edu

A. F. Dana is with the Division of Engineering and Applied Science at the California Institute of Technology. amirf@caltech.edu

J. P. Hespanha is with the Department of Electrical Engineering at the University of California, Santa Barbara. hespanha@ece.ucsb.edu

R. M. Murray is with the Division of Engineering and Applied Science at the California Institute of Technology. murray@caltech.edu

B. Hassibi is with the Division of Engineering and Applied Science at the California Institute of Technology. hassibi@caltech.edu

In this work, we are specifically interested in the problem of estimation and control across a *network* of communication links that drop packets. We consider a dynamical process evolving in time that is being observed by a sensor. The sensor needs to transmit the data over a network to a remote node, which can either be an estimator or a controller. However the links in the network stochastically drop packets. Preliminary work in this area has largely concentrated on simple networks consisting of a single link between sensor and remote estimator/controller. Within the one-link framework, both the stability [26], [31] and the performance [19], [26] problems have been considered. Approaches to compensate for the data loss to counteract the degradation in performance have also been proposed. As some representative examples, Nilsson [23] proposed using the previous control input or time-updating the previous estimate in case of data loss, Hadjicostis and Touri [12] proposed applying zero control if sensor data is lost, Ling and Lemmon [19] posed the problem as a filter-design through a non-linear optimization for scalar observations and Smith and Seiler [28] proposed a sub-optimal but computationally efficient estimator for packet drops occurring according to a Markov chain. Also relevant are the works of Azimi-Sadjadi [2], Schenato et al. [25] and Imer et al. [15] who looked at controller structures to minimize quadratic costs for systems in which both sensor-controller and controller-actuator channels are present. The related problem of optimal estimation across a packet-dropping link was considered by Sinopoli et al. in [27] for the case of one sensor and packet drops occurring in an i.i.d. fashion, while Gupta et al. [10] considered multiple sensors and more general packet drop models.

It has often been recognized that typical network / communication data packets have much more space for carrying information than required inside a traditional control loop. For instance, the minimum size of an ethernet data packet is 72 bytes, while a typical data point will only consume 2 bytes [7]. Many other examples are given in Lian et al. [18]. Moreover, many of the devices used in networked control systems have some processing and memory capabilities on account of being equipped to communicate across wireless channels or networks. Thus the question arises if we can use this possibility of pre-processing information prior to transmission and transmission of extra data to combat the effects of packet delays, loss and so on and improve the performance of a networked control system. In Gupta et al. [11] it was shown that pre-processing (or encoding) information before transmission over the communication link can indeed yield significant improvements in terms of stability and performance. Moreover, for a given performance level, it can also lead to a reduced amount of communication [30]. This effect can also be seen in the recent works on receding horizon networked control, in which a few future control inputs are transmitted at every time step by the controller and buffered at the actuator to be used in case subsequent control updates are dropped by the network and do not arrive at the actuator(s), see, e.g., [8], [9], [17], [21], [22].

In this paper, we consider the design of encoders and decoders when the sensor data has to be transmitted over a network of arbitrary topology. Transmission of data over networks for the purpose of estimation and control is largely an open problem. In [29], Tatikonda studied some issues related to the quantization rates required for stability when data is being transmitted over a network of digital noiseless channels. Also relevant is the work of Robinson and Kumar [24] who considered the problem of optimal placement of the controller when the sensor and the actuator are connected via a series of communication links. They ignore the issue of delays over paths of different lengths (consisting of different number of links) and under a *Long Packet Assumption* come up with

the optimal controller structure. There are two main reasons why the problem of encoding data for transmission is much more complicated in the case of transmission over a network:

- 1) Intermediate nodes have memory and processing ability. This memory should be used and therefore one should not view the network as a "passive" memoryless erasure channel.
- 2) Typically there are many paths from the source sensor to the remote estimator/controller. These paths typically exhibit different delays and levels of reliability. This diversity should be explored by the system designer.

We begin by proving a separation principle that separates the LQG control design problem into one of designing a state-feedback LQ optimal regulator and another of transmitting information across unreliable links for the estimation of a linear time invariant process across a network. We show that because the control inputs are applied to the process in an affine fashion, their effect on the estimate can be disregarded by the nodes in the network and only the estimator (assumed to be collocated with the controller) needs to have access to the past control inputs. This allows us to concentrate on the optimal information processing algorithm for the estimation problem.

We then propose a simple recursive algorithm that solves the estimation problem. Even though the algorithm requires a constant amount of memory, transmission and processing at any node, it is optimal for *any realization* of the packet drop sequence and has many additional desirable properties that we illustrate. The analysis of the algorithm identifies a property of the network called the max-cut probability that completely characterizes the network for its ability to be used for stabilizing a control loop. We also provide a framework to analyze the performance of our algorithm.

The main contributions of the paper are as follows:

- 1) We prove a separation principle that decomposes the optimal control problem into an LQ optimal regulator design and the estimation of a process across a network. Moreover, for the estimation problem, the intermediate nodes of the network do not require access to the control inputs.
- 2) We propose a strategy for information processing at the nodes of the network so that the estimator can calculate the optimal estimate at every time step. This algorithm is optimal for *any* packet-dropping process yet requires a constant amount of memory, processing and transmission at any node per time step.
- 3) We analyze the stability of the expected error covariance for this strategy when the packet drops are independent across time steps and across channels. For any other scheme (e.g., transmitting measurements without any processing), these conditions are necessary for stability. For channels with correlated drops, we show how to extend this analysis.
- 4) We calculate the performance for our algorithm for channels that drop packets independently. We provide a mathematical framework for evaluating the performance for a general network and provide expressions for networks containing links in series and parallel. We also provide lower and upper bounds for the performance over general networks. For any other strategy, these provide lower bounds for achievable performance.

Our results can also be used for *synthesis* of networks to improve estimation performance. We consider a simple example in which the optimal number of relay nodes to be placed is identified for estimation performance. We also

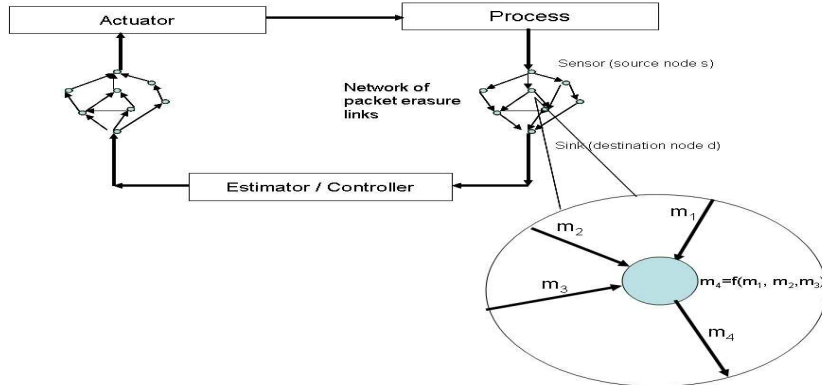


Fig. 1. The set-up of the control across communication networks problem. Every node computes a function of the incoming messages and transmits it. For most of the discussion in the paper, we ignore the network between the controller and the actuator. See, however, Section 3.

consider optimal routing of data in unicast networks. Simulation results are provided to illustrate the results.

The paper is organized as follows. In the next section, we set up the problem and state the various assumptions. Then, we state a separation principle that allows us to focus on the optimal estimation problem. In Section IV we identify a recursive yet optimal processing and transmission algorithm. We then specialize to the case of packet drops occurring in a memoryless fashion and independently across different links. We first do a stability analysis in Section V to obtain conditions on the packet drop probabilities under which the estimate error retains a bounded covariance. In Section VI we analyze the performance of the algorithm. We derive an expression for general networks and evaluate it explicitly for specific classes of networks. We also provide bounds for general networks. We then illustrate the results using some examples. Finally we consider some extensions of the analysis by considering correlated drops and using the results for optimal routing in unicast networks and for network synthesis.

II. PROBLEM SETUP

Consider the problem setup shown in Figure 1. Let a discrete-time linear process evolve according to the equation

$$x(k+1) = Ax(k) + Bu(k) + w(k), \quad (1)$$

where $x(k) \in \mathbf{R}^n$ is the process state, $u(k) \in \mathbf{R}^m$ is the control input and $w(k)$ is the process noise assumed to be white, Gaussian, and zero mean with covariance matrix R_w . The initial condition $x(0)$ is assumed to be independent of $w(k)$ and to have mean zero and covariance matrix $R(0)$. The state of the plant is measured by a sensor that generates measurements according to the equation

$$y(k) = Cx(k) + v(k). \quad (2)$$

The measurement noise $v(k)$ is white, zero-mean, Gaussian (with covariance matrix R_v) and independent of the plant noise $w(k)$. We assume that the pairs (A, B) and $\{A, R_w^{\frac{1}{2}}\}$ are stabilizable and the pair (A, C) is observable. Even though we consider the time-invariant case to simplify the presentation, the results in the paper continue to

hold for time-varying systems. A time-varying model can be useful, e.g., if the discrete-time process (1) results from non-uniform sampling of a continuous-time process.

The sensor communicates with an estimator or a controller across a network of communication links that stochastically drop packets. By an arbitrary network, we mean a network in which communication links are connected in an arbitrary topology. The sensor constitutes the source node and is denoted by s . The estimator / controller is designated as the destination node d . We can model the communication network as a directed graph \mathcal{G} and denote the node set by \mathcal{V} (in particular, \mathcal{V} contains s and d) and the edge set by $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The edges of the graph represent the communication links and are, in general, directed. Specifically, the link $e = (u, v)$ models a communication channel between node u and node v . We assume there are M edges or links present in the network. For any node $i \in \mathcal{V}$, the set of outgoing edges corresponds to the channels along which the node can transmit messages while the set of incoming edges corresponds to the channels along which the node receives messages. We denote the set of in-neighbors of node v by \mathcal{N}_v .

The communication links are modeled using a packet erasure model. The links take in as input a finite vector of real numbers. For every link, at each time-step, a packet is either dropped or received without any error at the output node. We assume sufficient bits per data packet so that the quantization error is negligible. This assumption makes sense if the communication packet provides enough bits for transmitting data (as in most modern communication network protocols) so that the effect of quantization error is dominated by the effect of the process and the measurement noises. We nominally consider the delays introduced by the channel to be less than one time step according to which the discrete-time dynamical process evolves. Most of the results in the paper can however be extended to the case when larger delays are present. In particular, the algorithm in the case of arbitrary delays and packet rearrangements is provided in Remark 2. In this paper, we also assume a global clock so that each node is synchronized. Finally, we assume either wireline communication or a wireless network in which each node can listen to all the messages coming from the incoming links without interference from each other.¹

If the packet dropping process is independent from one time step to the next (or, in other words, memoryless), the probability of dropping a packet on link $e = (u, v) \in \mathcal{E}$ is given by p_e (or equivalently p_{uv}) independent of time. If the process is uncorrelated in space, each such packet drop process is independent of packet drops in other links. While our analysis is largely limited to packet drop processes that are independent in time and uncorrelated in space, the algorithm that we propose is optimal for more sophisticated models of drop such as drops occurring according to a Markov chain etc. We refer to individual realizations of the random process that describes the drops as a *packet drop sequence*. The operation of the different nodes in the network at every time-step k can be described as follows:

- 1) Every node computes a function of all the information it has access to at that time.
- 2) It transmits the function on all the out-going edges. Potentially the node can transmit different functions

¹This property can be easily achieved by using a division multiple access scheme like FDMA, TDMA, CDMA etc. Similarly technologies like Software Radio (SWR) also have these properties.

along different edges. The destination node calculates the control input $u(k)$ or the estimate $\hat{x}(k)$ based on the information it possesses.

- 3) Every node observes the messages from all the incoming links and updates its information set for the next time step. For the source node, this message corresponds to the observation $y(k)$.

The time line that we have proposed ensures a strictly causal operation. Thus at time step k , the function that the source node transmits depends on measurements $y(0), y(1), \dots, y(k-1)$. Similarly, even if there were no packet drops, if the destination node is d hops away from the source node (i.e., the shortest path from the source node to the destination node involves d edges), its control input $u(k)$ or the estimate of the state $\hat{x}(k)$ at time k can only depend on measurements $y(0), y(1), \dots, y(k-d-1)$. Thus unlike the model in [24], every communication edge consumes one hop, or in other words, one time step as data is transmitted over it. We can easily adapt the discussion presented below to the causal case.

In the estimation problem, we are interested only in the estimate $\hat{x}(k)$ that minimizes the mean squared distortion

$$D_k = E \left[(x(k) - \hat{x}(k)) (x(k) - \hat{x}(k))^T \right], \quad (3)$$

where the expectation is taken over the uncorrelated variables $x(0)$, $\{w(k)\}$ and $\{v(k)\}$. In the control problem, the controller at every time step calculates a control input $u(k)$ and transmits it to the actuator. For the time being we ignore any communication channel between the controller and the actuator. We later revisit the presence of a controller - actuator channel and show how simple modifications to our design can take care of them. The controller aims at minimizing a quadratic cost function

$$J_T = E \left[\sum_{k=0}^T (x^T(k)Qx(k) + u^T(k)Ru(k)) + x^T(T+1)P_{T+1}^c x(T+1) \right], \quad (4)$$

where the expectation is again taken over the uncorrelated variables $x(0)$, $\{w(k)\}$ and $\{v(k)\}$. Note that the cost functionals D_k and J_T above also depend on the random packet-drop sequences in each link. However, we do *not* average across packet-drop processes; *the solution we present is optimal for arbitrary realizations of the packet dropping processes*. We assume that the controller has access to all the previous control inputs $u(0), \dots, u(k-1)$ while calculating $u(k)$. We show that the estimation problem and the control problem described above are deeply related through a separation principle. In particular, the solution to the control problem involves solving the estimation problem as well. We, therefore, concentrate on the control problem for now and point out the similarities with the estimation problem as they arise.

Without the communication network, the control problem is the same as the classical LQG control synthesis problem. However, in the presence of the network, it is unclear *a priori*, what the structure of the optimal control algorithm should be, and in what way the links in the network should be used to transmit information. We aim to solve the following problems:

- 1) Identify the optimal processing and transmission algorithm at the nodes that allow the estimator to minimize the cost D_k and the controller to minimize the cost J_T . Clearly, sending measurements alone might not be the optimal thing to do, since in such a scheme, dropping a packet would mean loss of information that cannot

be compensated for in the future. We are particularly interested in strategies that do not entail an increasing amount of transmission and memory at the nodes. Furthermore, we are not interested in strategies based on interleaving bits to transfer infinite amount of data since it is unclear what the effect of having finite (even though large) number of bits would be for such a strategy.

- 2) Identify the conditions on the network that would lead to a stable system.
- 3) Identify the best possible performance of the system in terms of the quadratic cost that can be achieved.

For future reference, we denote this problem set-up as problem \mathcal{P}_1 .

III. A SEPARATION PRINCIPLE

For the node i , denote by $\mathcal{I}^i(k)$ the information set that it can use to generate the message that it transmits at time step k . This set contains the aggregate of the information the node has received on the incoming edges at time steps $t = 0, 1, \dots, k - 1$. As an example, for the source node s , $\mathcal{I}^s(k) = \{y(0), y(1), \dots, y(k - 1)\}$. Without loss of generality, we can restrict our attention to information-set feedback controllers, i.e., controllers of the form $u(k) = u(\mathcal{I}^d(k), k)$. For a given information set at the destination $\mathcal{I}^d(\cdot)$, let us denote the minimal value of J_T by $J_T^*(\mathcal{I}^d)$. Let $\lambda_{pq}(k)$ be the binary random variable describing the packet drop event on link $(p, q) \in \mathcal{E}$ at time k . $\lambda_{pq}(k)$ assumes the value ‘dropped’ if the packet is dropped on link (p, q) at time k and ‘received’ otherwise. For a network with independent and memoryless packet drops, $\lambda_{pq}(k)$ is distributed according to Bernoulli distribution with parameter p_{pq} . We define $\lambda_{pp}(k) = \text{‘received’}$. Given the packet drop sequences in each link, at time step k we can define a time stamp $t^i(k)$ for node i such that the packet drops did not allow any information transmitted by the source after $t^i(k)$ to reach the i -th node in time for it to be a part of $\mathcal{I}^i(k)$.

Now consider an algorithm \mathcal{A}_1 that proceeds as follows. At time step k , every node takes the following actions:

- 1) Calculate the estimate of state $x(k)$ based on the information set at the node.
- 2) Transmit its entire information set on the outgoing edges.
- 3) Receive any data successfully transmitted along the incoming edges.
- 4) Update its information set and affix a time stamp corresponding to the time of the latest measurement in it.

When this algorithm is executed for a particular drop sequence, the information set at node i is of the form $\mathcal{I}^i(k) = \{y(0), y(1), \dots, y(t^i(k))\}$, where $t^i(k) < k$ is the time stamp as defined above. This is the maximal information set $\mathcal{I}^{i, \max}(k)$ that the node i can possibly have access to with any algorithm. For any other algorithm, the information set is smaller since earlier packets, and hence measurements, might have been dropped.

Note that for two information sets $\mathcal{I}^d(k, 1)$ and $\mathcal{I}^d(k, 2)$ related by $\mathcal{I}^d(k, 1) \subseteq \mathcal{I}^d(k, 2)$, we have $J_T^*(\mathcal{I}^d(k, 1)) \leq J_T^*(\mathcal{I}^d(k, 2))$. Thus, in particular, one way to achieve the optimal value of J_T is through the combination of an information processing algorithm that makes the information set $\mathcal{I}^{d, \max}(k)$ available to the controller and a controller that optimally utilizes the information set. Further, one such information processing algorithm is the algorithm \mathcal{A}_1 described above. However, this algorithm requires increasing data transmission as time evolves. Surprisingly, in a lot of cases, we can achieve the same performance using a constant amount of transmission and memory. To

see this, we first state the following separation principle. For any random variable $\alpha(k)$, denote by $\hat{\alpha}(k|\beta(k))$ the minimum mean squared error (mmse) estimate of $\alpha(k)$ given the information $\beta(k)$.

Proposition 1: [Separation Principle] Consider the packet-based optimal control problem \mathcal{P}_1 defined in section II. Suppose that each node transmits all the measurements it has access to at every time step, so that the decoder has access to the maximal information set $\mathcal{I}^{d,\max}(k)$ at every time step k . Then, for an optimizing choice of the control, the control and estimation costs decouple. Specifically, the optimal control input at time k is calculated to be

$$u(k) = \hat{u}_{LQ}(k|\mathcal{I}^{d,\max}(k), \{u(t)\}_{t=0}^{k-1}),$$

where $u_{LQ}(k)$ is the optimal LQ control law and $\hat{u}_{LQ}(k|\mathcal{I}^{d,\max}(k), \{u(t)\}_{t=0}^{k-1})$ denotes its minimum mean squared error (mmse) estimate given the information set $\mathcal{I}^{d,\max}(k)$ and the previous control inputs $u(0), \dots, u(k-1)$.

Proof: The proof is along the lines of the standard separation principle (see, e.g., [13, Chapter9]; see also [11]) and is omitted for space constraints. ■

There are two reasons this principle is useful to us:

- 1) We recognize that the optimal controller does not need to have access to the information set $\mathcal{I}^{d,\max}(k)$ at every time step k . The encoders and the decoder only need to ensure that the controller receives the quantity $\hat{u}_{LQ}(k|\mathcal{I}^{d,\max}(k), \{u(t)\}_{t=0}^{k-1})$, or equivalently, $\hat{x}(k|\mathcal{I}^{d,\max}(k), \{u(t)\}_{t=0}^{k-1})$.
- 2) If we can ensure that the controller has access to this quantity, the controller design part of the problem is solved. The optimal controller is the solution to the LQ control problem.

Thus the controller needs access to the estimate $\hat{x}(k|\mathcal{I}^{d,\max}(k), \{u(t)\}_{t=0}^{k-1})$ based on the information set $\mathcal{I}^{d,\max}(k)$ (that is of the form $\{y(0), y(1), \dots, y(j)\}$ for some $j < k$) and the previous control inputs. We can make another simplification in the problem by separating the dependence of the estimate on measurements from the effect of the control inputs. In the context of our problem, this is useful since the nodes in the network do not then need access to the control inputs and can concentrate solely on the effect of measurements. The effect of the control inputs can be taken care of by the decoder or the controller that has access to all previous control inputs.

Proposition 2: Consider the problem \mathcal{P}_1 defined in section II. The quantity $\hat{x}(k|\mathcal{I}^{d,\max}(k), \{u(t)\}_{t=0}^{k-1})$, where $\mathcal{I}^{d,\max}(k)$ is of the form $\{y(0), y(1), \dots, y(j)\}$ for some $j < k$, can be calculated as the sum of two quantities

$$\hat{x}(k|\mathcal{I}^{d,\max}(k), \{u(t)\}_{t=0}^{k-1}) = \bar{x}(k) + \psi(k),$$

where $\bar{x}(k)$ depends only on $\mathcal{I}^{d,\max}(k)$ and $\psi(k)$ depends only on the control inputs. Further, both the quantities $\bar{x}(k)$ and $\psi(k)$ can be calculated using recursive linear filters.

Proof: The quantity $\hat{x}(k|\{y(t)\}_{t=0}^j, \{u(t)\}_{t=0}^{k-1})$ can be calculated using the Kalman filter in two steps.

- 1) First calculate $\hat{x}(j+1|\{y(t)\}_{t=0}^j, \{u(t)\}_{t=0}^j)$ by processing the measurements and the control inputs from time $m = 0$ to j as follows:

Measurement Update for the Kalman filter:

$$(P(m|m))^{-1} = (P(m|m-1))^{-1} + C^T R_v^{-1} C \quad (5)$$

$$(P(m|m))^{-1} \hat{x}(m|\{y(t)\}_{t=0}^m, \{u(t)\}_{t=0}^{m-1}) = (P(m|m-1))^{-1} \hat{x}(m|\{y(t)\}_{t=0}^{m-1}, \{u(t)\}_{t=0}^{m-1}) + C^T R_v^{-1} y(m).$$

Time Update for the Kalman filter:

$$\begin{aligned} P(m|m-1) &= AP(m-1|m-1)A^T + R_w \\ \hat{x}(m|\{y(t)\}_{t=0}^{m-1}, \{u(t)\}_{t=0}^{m-1}) &= A\hat{x}(m|\{y(t)\}_{t=0}^{m-1}, \{u(t)\}_{t=0}^{m-2}) + Bu(m-1). \end{aligned} \quad (6)$$

The initial conditions for the Kalman filter are given by $\hat{x}(0|y(-1), u(-1)) = 0$ and $P(0|-1) = P(0)$.

2) Construct the estimate $\hat{x}(k|\{y(t)\}_{t=0}^j, \{u(t)\}_{t=0}^{k-1})$ as

$$\hat{x}(k|\{y(t)\}_{t=0}^j, \{u(t)\}_{t=0}^{k-1}) = A^{k-j-1}\hat{x}(j+1|\{y(t)\}_{t=0}^j, \{u(t)\}_{t=0}^j) + \sum_{i=0}^{k-j-2} A^i Bu(k-i-1). \quad (7)$$

However, the effect of the control input appears linearly and can be separated. Let us calculate the quantity $\hat{x}(j+1|j)$ using the measurements from time $m = 0$ to $k-1$ according to the following *modified Kalman filter*.

Measurement Update for the modified Kalman filter:

$$\begin{aligned} (P(k|k))^{-1} &= (P(k|k-1))^{-1} + C^T R_v^{-1} C \\ (P(k|k))^{-1} \hat{x}(k|k) &= (P(k|k-1))^{-1} \hat{x}(k|k-1) + C^T R_v^{-1} y(k), \end{aligned} \quad (8)$$

Time Update for the modified Kalman filter:

$$\begin{aligned} P(k|k-1) &= AP(k-1|k-1)A^T + R_w \\ \hat{x}(k|k-1) &= A\hat{x}(k-1|k-1). \end{aligned} \quad (9)$$

The initial conditions are given by $\hat{x}(0|-1) = 0$ and $P(0|-1) = P(0)$. Note that calculation of the terms $P(k|k)$ and $P(k|k-1)$ does not require knowledge of either the measurements $y(t)$'s or the control inputs $u(t)$'s and these terms can even be calculated offline. The equations for the modified Kalman filter are identical to the ones for the Kalman filter given in (5) and (6) except that the control input $u(m)$ is assumed to be 0. The effect of the control inputs can be taken care of through the term $\tilde{\psi}(j+1)$ that evolves as

$$\begin{aligned} \tilde{\psi}(m) &= Bu(m-1) + \Gamma(m-1)\tilde{\psi}(m-1) \\ \Gamma(m) &= A(P(m-1|m-1))^{-1}P(m-1|m-2) \end{aligned}$$

with the initial condition $\tilde{\psi}(0) = 0$. It can readily be verified that

$$\hat{x}(j+1|\{y(t)\}_{t=0}^j, \{u(t)\}_{t=0}^j) = \hat{x}(j+1|j) + \tilde{\psi}(j+1). \quad (10)$$

The estimate at time step $k+1$ can then once again be calculated using (7). Comparing the two methods, we see that

$$\hat{x}(k|\{y(t)\}_{t=0}^j, \{u(t)\}_{t=0}^{k-1}) = A^{k-j-1}\hat{x}(j+1|j) + A^{k-j-1}\tilde{\psi}(j+1) + \sum_{i=0}^{k-j-2} A^i Bu(k-i-1),$$

where the term $\hat{x}(j+1|j)$ depends only on the measurements and the terms $\tilde{\psi}(j+1)$ and $u(k-i-1)$ depend only on the control inputs. Further, as shown above, these terms can be calculated using linear recursive filters. To

complete the proof, we simply identify

$$\begin{aligned}\bar{x}(k) &= A^{k-j-1}\hat{x}(j+1|j) \\ \psi(k) &= A^{k-j-1}\tilde{\psi}(j+1) + \sum_{i=0}^{k-j-2} A^i B u(k-i-1).\end{aligned}$$

■

As mentioned above, the advantage of separating the effects of measurements and the control inputs is that the nodes in the network can concentrate on delivering $\bar{x}(k)$ to the controller and the controller (which has access to all the control inputs) can then calculate $\psi(k)$ and, in turn, $\hat{x}\left(k|\{y(t)\}_{t=0}^j, \{u(t)\}_{t=0}^{k-1}\right)$. The nodes in the network do not need access to the control inputs. Finally, note that the term $\bar{x}(k)$ that the network needs to deliver is, in fact, the mmse estimate of the state $x(k)$ of a process evolving as

$$x(k+1) = Ax(k) + w(k), \quad (11)$$

given the measurements $y(0), y(1), \dots, y(j)$ that are assumed to originate from a sensor of the form (2). Thus consider an alternative estimation problem \mathcal{P}_2 . A process of the form (11) is observed by a sensor of the form (2). There is an estimator across the network that needs to estimate the state $x(k)$ of the process in the mmse sense at every time step k . The network is modeled in the same manner as in the original problem \mathcal{P}_1 . We can once again define the information set $\mathcal{I}^i(k)$ that the node i has access to at time k and the corresponding maximal information set $\mathcal{I}^{i,\max}(k)$. What is the optimal information processing algorithm to be followed by each node that allows the estimator to calculate the estimate of $x(k)$ based on the information set $\mathcal{I}^{d,\max}(k)$? By the arguments above, the optimal information processing algorithm for the nodes in the network in the problems \mathcal{P}_1 and \mathcal{P}_2 is identical². For the presentation of the algorithm and analysis of its properties, we consider this equivalent problem \mathcal{P}_2 while keeping in mind that, to solve problem \mathcal{P}_1 , the controller can then calculate $\psi(k)$ to include the effect of the previous control inputs and finally the new control input $u(k)$ by utilizing the separation principle.

IV. AN OPTIMAL YET RECURSIVE ENCODING ALGORITHM

We now describe an algorithm \mathcal{A}_2 that achieves the same performance as the algorithm \mathcal{A}_1 at the expense of constant memory, processing and transmission (modulo the transmission of the time stamp). The algorithm proceeds as follows. At each time step k , every node i takes the following actions:

- 1) Calculate its estimate $\hat{x}^i(k)$ of the state $x(k)$ based on any data received at the previous time step $k-1$ and its previous estimate. The estimate can be computed using a switched linear filter, as shown later.
- 2) Affix a time stamp corresponding to the latest measurement used in the calculation of the estimate in step 1 and transmit the estimate on the outgoing edges.
- 3) Receive data on the incoming edges, if any, and store it for the next time step.

To prove that algorithm \mathcal{A}_2 is indeed optimal, we need the following intermediate result.

²Thus, as we noted in Section II, the algorithms for solving the control and the estimation problems are identical.

Lemma 1: Consider any edge (i, j) and any packet drop pattern. At time step k , let the node i transmit the measurement set $S^{ij} = \{y(0), y(1), \dots, y(l)\}$ on the edge (i, j) if algorithm \mathcal{A}_1 is executed. If, instead, algorithm \mathcal{A}_2 is executed, the node i transmits the estimate $\hat{x}(k|S^{ij}) = \hat{x}(k|\{y(0), y(1), \dots, y(l)\})$ along the edge (i, j) at time step k .

Proof: The proof readily follows by induction on the time step k . For time $k = 1$, by definition, the source node s transmits $\{y(0)\}$ along all edges of the form (s, \cdot) while following algorithm \mathcal{A}_1 and the estimate $\hat{x}(1|y(0))$ while executing algorithm \mathcal{A}_2 . If any edge is not of the form (s, \cdot) , there is no information transmitted along that edge in either algorithm. Thus the statement is true for $k = 1$. Now assume that the statement is true for $k = n$. Consider the node i at time $k = n + 1$. If the node i is the source node, the statement is true once again by definition. Let us assume that node i is not the source node. Consider all edges that transmitted data at time step $k = n$ to node i . For $p \in \mathcal{N}_i$, let the edges (p, i) transmit the measurement set $S^{pi} = \{y(0), y(1), \dots, y(t(p))\}$, if algorithm \mathcal{A}_1 is being executed. Also denote the measurement set that the node i has access to from time step $k = n - 1$ as $S^{ii} = \{y(0), y(1), \dots, y(t(i))\}$. Note that at time step $k = n$, the node i transmitted the set S^{ii} along all outgoing edges (i, \cdot) . Let v be the node for which

$$t^v(n) = \max\{t(i) \cup \{t(p) | p \in \mathcal{N}_i\}\}.$$

At time $k = n + 1$, the node i transmits along all outgoing edges the measurement set $\mathcal{S}^1 = \{y(0), y(1), \dots, y(t(v))\}$. Now consider the case when algorithm \mathcal{A}_2 is being executed. By the assumption of the statement being true at time step $k = n$, the edges (p, i) transmit the estimate

$$\hat{x}(n|S^{pi}) = \hat{x}(n|\{y(0), y(1), \dots, y(t(p))\}),$$

for all $p \in \mathcal{N}_i$. Also since at time $k = n$ the node transmitted S^{ii} on any edge (i, \cdot) in algorithm \mathcal{A}_1 , it has access to the estimate $\hat{x}(n|S^{ii})$ when algorithm \mathcal{A}_2 is executed. Clearly, the set S^{vi} is the superset of all sets S^{ii} and S^{ji} where $j \in \mathcal{N}_i$ and v have been defined above. Thus the estimate that the node i calculates at time $k = n + 1$ is $\hat{x}(n + 1|S^{vi})$. But the measurement set S^{vi} is simply the set \mathcal{S}^1 . Hence at time step $k = n + 1$, the node i transmits along all outgoing edges the estimate $\hat{x}(n + 1|\mathcal{S}^1)$. Thus the statement is true at time step $k = n + 1$ along all edges of the form (i, \cdot) . Since the node i was arbitrary, the statement is true for all edges in the graph. Thus we have proven that if the statement is true at time $k = n$, it is true at time $k = n + 1$. But it is true at time $k = 1$. Thus, by the principle of mathematical induction, it is true at all time steps. ■

Note that we have also shown that if at time step k , the node has access to the measurement set S^{ii} from time step $k - 1$ when algorithm \mathcal{A}_1 is executed; it has access to the estimate $\hat{x}(k - 1|S^{ii})$ from time step $k - 1$ when algorithm \mathcal{A}_2 is executed. We can now state the following result.

Proposition 3: The algorithm \mathcal{A}_2 is optimal in the sense that it leads to the minimum possible error covariance at any node at any time step.

Proof: Consider a node i . At time k , let $j \in \{i\} \cup \mathcal{N}_i$ such that $\lambda_{ji}(k - 1) = \text{'received'}$. Denote the measurement set that is transmitted from node j to node i at time step k under algorithm \mathcal{A}_1 by S^{ji} . As in the

proof of Lemma 1 there is a node v , such that S^{vi} is the superset of all the sets S^{ji} . Thus the estimate of node i at time k under algorithm \mathcal{A}_1 is $\hat{x}^{\mathcal{A}_1}(k) = \hat{x}(k|S^{vi})$. From Lemma 1, when algorithm \mathcal{A}_2 is executed, at time step k , the node i has access to the estimates $\hat{x}(k-1|S^{ji})$. Once again, since S^{vi} is the superset of all the sets S^{ji} , the estimate of node i at time step k is simply

$$\hat{x}^{\mathcal{A}_2}(k) = A\hat{x}(k-1|S^{vi}) = \hat{x}(k|S^{vi}).$$

Thus, for any node i , the estimates $\hat{x}^{\mathcal{A}_1}(k)$ and $\hat{x}^{\mathcal{A}_2}(k)$ are identical for any time step k for any packet drop pattern. Since algorithm \mathcal{A}_1 leads to the minimum possible error covariance at any node, algorithm \mathcal{A}_2 is optimal. ■

The step of calculating the estimate at each node in the algorithm \mathcal{A}_2 can be implemented using a switched linear filter as follows. The source node implements a Kalman filter and updates its estimate at every time step with the new measurement received. Every other node i checks the time-stamps on the data coming on the incoming edges. The time-stamps correspond to the latest measurement used in the calculation of the estimate being transmitted. Then node i updates its time-stamp using the relation

$$t^i(k) = \max_{j \in \mathcal{N}_i \cup \{i\}} \lambda_{ji}(k-1)t^j(k-1). \quad (12)$$

Suppose the maximum of (12) is achieved by node $n \in \mathcal{N}_i \cup \{i\}$. If $\hat{x}^n(k)$ denotes the estimate of the state $x(k)$ maintained by the node n , the node i updates its estimate as $\hat{x}^i(k) = A\hat{x}^n(k-1)$.

Remark 1 (Optimality for any Drop Sequence and the ‘Washing Away’ Effect): We have made no assumptions on the packet drop pattern or knowledge of the statistics of the packet drops at any of the nodes. The algorithm provides the optimal estimate for an arbitrary packet drop sequence, irrespective of whether the packet drop can be modeled as an i.i.d. process or a more sophisticated model like a Markov chain. The algorithm results in the optimal estimate at every time step for any instantiation of the packet drop sequence, not merely in the optimal average performance. Also note that any received data vector $\hat{x}(k|j)$ ‘washes away’ the effect of all previous packet drops. It ensures that the estimate at the receiving node is identical to the case when all measurements $y(0), y(1), \dots, y(j)$ were available, irrespective of which previous data packets had been dropped.

Remark 2 (Presence of Delays): If the links introduce random delays, the algorithm remains optimal irrespective of the possibility of packet rearrangements. Each node, at every time step, still calculates the estimate of the state $x(k)$ based on any information received at that time step and the previous estimate from its memory, affixes the correct time stamp and transmits it along out-going edges.

Remark 3 (Channel between the controller and the actuator): If we look at the proof of the separation principle given above, the crucial assumption was that the controller knows what control input is applied at the plant. Thus, if we have a channel between the controller and the plant, the separation principle would still hold, provided there is a provision for acknowledgment from the receiver to the transmitter for any packet successfully received over that channel³. The optimal information processing algorithm presented above carries over to this case. We can also ask the question of the optimal encoder-decoder design for the controller-actuator channel. The optimal decoding

³Note that we do not require acknowledgements for the sensor-controller channel.

at the actuator end depends on the information assumed to be known to the actuator (e.g. the cost matrices Q and R). Design of the decoder for various information sets is an interesting open problem.

The algorithm proposed above is optimal for any node. Thus we do not need to assume only one destination. The algorithm is also optimal for multiple sources if all sources have access to measurements from the same sensor. For multiple sources with each source obtaining measurements from a different sensor, the problem remains open. It should also be noted that a priori we had not made any assumption about a node transmitting the same message along all the out-going edges. It turned out that in this optimal algorithm, the messages are the same along all the edges. This property is especially useful in the context of wireless communication which is inherently broadcast in nature. Finally, the communication requirements of the algorithm can be reduced by adopting an event-based protocol in which a node transmits only if it updated its estimate based on data arriving on an incoming edge. This does not degrade the performance but reduce the number of transmissions, especially if packet drop probabilities are high. In Sections V and VI, we analyze the stability and performance of the above algorithm by assuming that packets are dropped independently from one time step to next and uncorrelated in space. We return to more general packet dropping processes in Section VIII.

V. STABILITY ANALYSIS

We are interested in stability in the bounded second moment or the mean squared sense. Thus for the problem \mathcal{P}_1 , we say that the system is stable if $E[J_\infty]$ is bounded, where $J_\infty = \lim_{T \rightarrow \infty} \frac{J_T}{T}$ and the expectation is taken over the packet dropping processes in the network. For the problem \mathcal{P}_2 , denote the error at time step k as $e^d(k) = x(k) - \hat{x}^d(k)$, where $\hat{x}^d(k)$ is the estimate of the destination node. We can compute the covariance of the error $e(k)$ at time k as $R^d(k) = E[e^d(k)(e^d(k))^T]$, where the expectation is taken over the initial condition $x(0)$, the process noise $w(j)$ and the measurement noise $v(j)$. We can further take the expectation with respect to the packet dropping process in the network and denote $P^d(k) = E[R^d(k)]$. We consider the steady-state error covariance in the limit as k goes to infinity, i.e.,

$$P^d(\infty) = \lim_{k \rightarrow \infty} P^d(k). \quad (13)$$

If the limit exists and is bounded, we say that the estimate error is stable; otherwise it is unstable⁴. Note that because of the separation principle, the cost J_T and the estimation error covariance $R^d(k)$ are related through

$$J_T = E[x^T(0)S(0)x(0)] + \text{trace}(S(0)R^d(0)) + \sum_{k=0}^{T-1} \text{trace}(S(k+1)R_w + (A^T S(k+1)A + Q - S(k))R^d(k)),$$

where $S(k)$ is the Riccati variable that arises because of the LQ optimal control being calculated and evolves as

$$S(k) = A^T S(k+1)A + Q - A^T S(k+1)B (B^T S(k)B + R)^{-1} B^T S(k+1)A.$$

⁴Our definition of stability requires the average estimation error or quadratic control cost to be bounded. Other metrics that require the probability density function of the cost to decay at a fast enough rate are also possible.

Because of the stabilizability assumptions, $S(k)$ would tend to a constant value S as the horizon T becomes longer. Thus in the limit as $T \rightarrow \infty$, we obtain

$$J_\infty = \text{trace}(SR_w) + \text{trace}((A^T SA + Q - S)R^d(\infty)).$$

If we now take the expectation with respect to the packet dropping processes in the network, we obtain

$$E[J_\infty] = \text{trace}(SR_w) + \text{trace}((A^T SA + Q - S)P^d(\infty)). \quad (14)$$

Thus the stability conditions for problems \mathcal{P}_1 and \mathcal{P}_2 are identical. We now proceed to evaluate these conditions.

For node d and time k , let $t^d(k)$ denote the time-stamp of the most recent observation used in estimating $x(k)$ at the destination node d . This time-stamp evolves according to (12). The expected estimation error covariance at time k at node d can thus be written as

$$\begin{aligned} P^d(k) &= E \left[(x(k) - \hat{x}^d(k)) (x(k) - \hat{x}^d(k))^T \right] \\ &= \sum_{l=0}^{k-1} E \left[(x(k) - \hat{x}^d(k|t^d(k)=l)) (x(k) - \hat{x}^d(k|t^d(k)=l))^T \right] \text{Prob}(t^d(k)=l), \end{aligned}$$

where in the second equation we have evaluated the expectation with respect to the packet dropping process and $\hat{x}^d(k|t^d(k)=l)$ denotes the estimate of $x(k)$ at the destination node given all the measurements $\{y(0), y(1), \dots, y(l)\}$. We see that the effect of the packet dropping process shows up in the distribution of the time-stamp of the most recent observation used in estimating $x(k)$. For future use, we denote the *latency* for the the node d at time k as $l^d(k) = k - 1 - t^d(k)$. Also, denote the mmse estimate of $x(k)$ given all the measurements $\{y(0), y(1), \dots, y(k-1)\}$ by $P(k)$. It is well-known that $P(k)$ evolves according to the Riccati recursion

$$P(k+1) = AP(k)A^T + R_w - AP(k)C^T (CP(k)C^T + R_v)^{-1} CP(k)A^T.$$

We can now rewrite the error covariance $P^d(k)$ as

$$P^d(k) = \sum_{l=0}^{k-1} \left[A^l P(k-l) (A^l)^T + \sum_{j=0}^{l-1} A^j Q (A^j)^T \right] \times \text{Prob}(l^d(k)=l). \quad (15)$$

The above equation gives the expected estimation error covariance for a general network with any packet dropping process. The effect of the packet dropping process appears in the distribution of the latency $l^d(k)$. As we can see from (15), the stability of the system depends on how fast the probability distribution of the latency decreases. To analyze the stability, we use the following result from [11] restated here for independent packet drops.

Proposition 4: Consider a process of the form (11) being estimated using measurements from a sensor of the form (2) over a packet-dropping link that drops packets in an i.i.d. fashion with probability q . Suppose that the sensor calculates the mmse estimate of the measurements at every time step and transmits it over the channel. Then the estimate error at the receiver is stable in the bounded second moment sense if and only if $q|\rho(A)|^2 < 1$, where $\rho(A)$ is the spectral radius of the matrix A appearing in (11).

A. Network with Links in Parallel

We begin by considering a network consisting only of links in parallel. Consider the source and the destination node being connected by a network with m links in parallel with the probability of packet drop in the i -th link being p_i . Since the same data is being transmitted over all the links, the distribution of the latency in (15) remains the same if the network is replaced by a single link that drops packets when all the links in the original network drop packets and transmits the information if even one link in the original network allows transmission. Thus the packet drop probability of this equivalent link is $p_1 p_2 \cdots p_m$. The necessary and sufficient condition for the error covariance to diverge thus becomes $p|\rho(A)|^2 < 1$, where $p = p_1 p_2 \cdots p_m$.

B. Necessary Condition for Stability in Arbitrary Networks

Using the result for parallel networks, we can obtain a necessary condition for stability for general networks.

Proposition 5: Consider a process of the form (11) being estimated using measurements from a sensor of the form (2) through an arbitrary network of packet dropping links. Consider every possible division of the nodes of the network into two sets with the source and the destination node being in different sets (also called a cut-set). For any such division, let p_1, p_2, \dots, p_n denote the packet erasure probabilities of the edges that connect the two sets. Define the cut-set erasure probability as $p_{\text{cut set}} = p_1 p_2 \cdots p_n$. Then a necessary condition for the error covariance to converge is $p_{\text{max-cut}} |\rho(A)|^2 < 1$, where $p_{\text{max-cut}}$ is the network erasure probability defined as

$$p_{\text{max-cut}} = \max_{\text{all possible cut-sets}} p_{\text{cut set}}. \quad (16)$$

Proof: Denote the given network by \mathcal{N}_1 . Consider a cut set C of the network \mathcal{N}_1 , with the source s being in set A and the destination node d in set B and the links $1, 2, \dots, n$ joining the sets A and B . Form another network \mathcal{N}_2 by replacing all links within the sets A and B by links that do not drop packets and additionally do not consume one time step to transmit data. Now for any packet drop pattern, denote the information sets that the destination node has access to at any time step k over networks \mathcal{N}_1 and \mathcal{N}_2 by $\mathcal{I}^{d, \mathcal{N}_1}(k)$ and $\mathcal{I}^{d, \mathcal{N}_2}(k)$ respectively. It is obvious that $\mathcal{I}^{d, \mathcal{N}_1}(k) \subseteq \mathcal{I}^{d, \mathcal{N}_2}(k)$. Thus the estimate error covariances at the destination node for the two networks are related by $P^d(k|\mathcal{I}^{d, \mathcal{N}_1}(k)) \geq P^d(k|\mathcal{I}^{d, \mathcal{N}_2}(k))$. Hence, by considering the stability of error covariance over network \mathcal{N}_2 , we can obtain a necessary condition for the stability of error covariance over network \mathcal{N}_1 . Since the edges within the source and the destination sets do not introduce any delay or error, \mathcal{N}_2 consists of the source and the destination joined by edges $1, 2, \dots, n$ in parallel. The condition for the error covariance across \mathcal{N}_2 to converge is thus $p_{\text{cut set}} |\rho(A)|^2 < 1$, where $p_{\text{cut set}} = p_1 p_2 \cdots p_n$. This is thus a necessary condition for error covariance across \mathcal{N}_1 to be stable. One such condition is obtained by considering each cut-set. Thus a necessary condition for the error covariance to converge is $p_{\text{max-cut}} |\rho(A)|^2 < 1$, where $p_{\text{max-cut}}$ is given by (16). ■

C. Network with Links in Series

Consider a network consisting of two links in series with probabilities of packet drop p_1 and p_2 respectively. Denote the nodes as v_1, v_2 and v_3 with v_1 being the source node and v_3 the destination. Also, denote the estimate at

node v_i at time k by $\hat{x}^i(k)$. Let $e^1(k)$ be the error between $x(k)$ and $\hat{x}^2(k)$. Similarly let $e^2(k)$ be the error between $\hat{x}^2(k)$ and $\hat{x}^3(k)$. We are interested in the second moment stability of $e^1(k) + e^2(k)$. Clearly a sufficient condition is that both $e^1(k)$ and $e^2(k)$ individually be second moment stable. Applying Proposition 4, if $p_1 | \rho(A) |^2 < 1$, $e^1(k)$ would be stable. Now for the second link, we can consider the sensor

$$\hat{x}^2(k) = x(k) + e^1(k),$$

generating the measurements. The quantity transmitted by node 2 at any time step in the algorithm \mathcal{A}_2 can be seen to be the mmse estimate of $x(k)$ given all the measurements $\{\hat{x}^2(j)\}_{j=0}^{k-1}$. Consequently, if $p_2 | \rho(A) |^2 < 1$, then the error $e^2(k)$ is stable. If p be the greater of the probabilities p_1 and p_2 , the sufficient condition thus is $p|\rho(A)|^2 < 1$. But this is identical to the necessary condition stated in Proposition 5. Thus the condition above is both necessary and sufficient.⁵ Clearly this argument can be extended to any finite number of links in series. If there are m links in series with the probability of drop of the i -th link being p_i , then a necessary and sufficient condition for the estimate error to converge at the destination node is $p|\rho(A)|^2 < 1$, where $p = \max(p_1, p_2, \dots, p_m)$.

D. Sufficient Condition for Arbitrary Networks

We now proceed to prove that the condition stated in Proposition 5 is sufficient as well for stability.

Proposition 6: Consider the assumptions of Proposition 5 on the process and the network. Then the estimation error covariance under algorithm \mathcal{A}_2 is stable if $p_{\max\text{-cut}}|\rho(A)|^2 < 1$.

Proof: First note that if a packet dropping link between two nodes v and u with probability of drop p_e is replaced by two parallel links with drop probabilities $p_i^{(1)}$ and $p_i^{(2)}$ such that $p_i = p_i^{(1)} p_i^{(2)}$, then the average error covariance of the estimation under algorithm \mathcal{A}_2 does not change at any node. This is true simply because the probability distribution of the latency in (15) does not change with this replacement.

Next consider the set $\Gamma = \{\gamma_1, \gamma_2, \dots, \gamma_m\}$ of all simple directed paths from the source to the destination in the network graph. An edge i may be in more than one of these paths. If the edge i is in path γ_j , we denote that as $i \in \gamma_j$. Consider the following optimization problem

$$\min_{\beta_j} \prod_{j=1}^m \beta_j, \quad (17)$$

subject to the constraints

$$\begin{aligned} \prod_{i \in \gamma_j} \beta_j &\geq p_i \quad \forall \text{ edges } i \\ 0 \leq \beta_j &\leq 1 \quad \forall j = 1, 2, \dots, m. \end{aligned} \quad (18)$$

A simple change of variables

$$\psi_j = -\log \beta_j, \quad (19)$$

⁵Another approach to find the necessary and sufficient condition for stability is to evaluate the cost function of (15) for a series of links. This approach is taken in Section VI-A.

transforms the above optimization problem into the following linear program in the variables ψ_j 's.

$$\max_{\psi_j} \sum_{j=1}^m \psi_j \quad (20)$$

subject to

$$\begin{aligned} \sum_{i \in \gamma_j} \psi_j &\leq -\log p_i \quad \forall \text{edges } i \\ \psi_j &\geq 0 \quad \forall j = 1, 2, \dots, m. \end{aligned}$$

The solutions of the optimization problems (17) and (20), denoted by $\{\beta_j^*\}$ and $\{\psi_j^*\}$, are related through $\psi_j^* = -\log \beta_j^*$. The structure of the linear program (20) is the same as the one used for finding the maximum flow possible in a fluid network [4, Page 59], which has the same topology as our packet dropping network with the capacity of the link i equal to $-\log p_i$. The solution to the problem of finding the maximum flow through a fluid network is well-known to be given by the max-flow min-cut theorem. Using this fact, we see that the solution to the optimization problem (20) is given by

$$\psi_j^* = \min_{\text{all possible cut-sets}} \sum_{i \in \text{cut}} -\log p_i.$$

Thus for the optimization problem (17), the solution is given by

$$\begin{aligned} \beta_j^* &= \max_{\text{all possible cut-sets}} \prod_{i \in \text{cut}} p_i \\ &= \max_{\text{all possible cut-sets}} p_{\text{cut set}} \\ &= p_{\text{max-cut}}, \end{aligned} \quad (21)$$

where $p_{\text{cut set}}$ and $p_{\text{max-cut}}$ have been defined in Proposition 5.

Consider the paths in the set Γ . Form a new set \mathcal{B} of all those paths γ_j 's for which the associated optimal variable β_j^* is strictly less than one. The remaining paths in Γ have equivalent erasure probability as unity and can thus be ignored. Now form a new network \mathcal{N}' as follows. The node set of \mathcal{N}' is the union of those nodes of the original network \mathcal{N} that are present on any path in \mathcal{B} . Each pair of nodes (u, v) in the node set of \mathcal{N}' is connected by (possibly) multiple links. If an edge i between two nodes u and v is present in a path $\gamma_j \in \mathcal{B}$, we add an edge between nodes u and v in \mathcal{N}' and associate with it an erasure probability β_j^* . By considering all the edges in \mathcal{N}' and following this procedure, we construct the edge set of \mathcal{N}' . The following properties of \mathcal{N}' are easily verified.

- By construction, \mathcal{N}' can be presented as union of edge-disjoint paths. Each path in \mathcal{N}' corresponds to one path in \mathcal{B} . Furthermore, for each path, the probabilities of packet drop on all the links of that path are equal.
- By virtue of (21) and the procedure followed to construct \mathcal{N}' , the product of the probabilities of packet drop of the different paths is equal to the equivalent probability of the network, $p_{\text{max-cut}}$, for the network \mathcal{N} .
- For any pair of nodes that were connected by a link in \mathcal{N} , the product of the probabilities of packet dropping of the links in \mathcal{N}' connecting these two nodes is greater than or equal to the drop probability of the link between the same pair of nodes in \mathcal{N} . This can be seen from the first inequality constraint of (18).

Therefore the estimate error covariance at the destination by following algorithm \mathcal{A}_2 in the original network \mathcal{N} is less than or equal to the error covariance by following \mathcal{A}_2 in the new network \mathcal{N}' . Thus to obtain a sufficient condition on stability, we can analyze the performance of \mathcal{A}_2 in the network \mathcal{N}' . For this we consider another algorithm, which we denote as \mathcal{A}_3 . In this algorithm we consider the disjoint paths given in \mathcal{N}' and assume that estimates on different paths are routed separately. Thus if a node lies on many paths, on each path it forwards the packets it received on that path only. Clearly the performance \mathcal{A}_3 cannot be better than \mathcal{A}_2 since in \mathcal{A}_2 we send the most recent estimate received from different paths at any node compared to forwarding the estimates on different paths separately from each other.

Therefore to prove the theorem we only need to show the stability of estimation using protocol \mathcal{A}_3 assuming that the condition of Proposition 5 holds. Since we do not mix the estimates obtained from different paths in \mathcal{A}_3 , the network can be considered as a collection of parallel paths, with each path consisting of links with equal drop probability. We further upper-bound the error covariance through two means:

- 1) Convert the network \mathcal{N}' into a network \mathcal{N}'' by introducing links in the different paths from the source to the destination in \mathcal{N}' such that each path consists of equal number of links, while retaining the property that every link in a path has the same probability of drop. Thus, we can consider nodes present in layers, with layer 0 corresponding to the source, layer 1 consisting of all nodes that are one hop away along the different paths from the source and so on. Let the destination be at level m .
- 2) Let the nodes in level t have estimate error covariances $P_i^t(k)$'s at time k . When selecting messages to be transmitted to the nodes in level $t + 1$, we use the algorithm \mathcal{A}_4 , in which the nodes transmit the estimate that has the maximum $P_i^t(k)$ ⁶.

Let us now prove that the condition in Proposition 5 is sufficient for the stability of the error covariance when algorithm \mathcal{A}_4 is executed over network \mathcal{N}'' . Clearly $p_{\max\text{-cut}} |\rho(A)|^2 < 1$ implies $p_i |\rho(A)|^2 < 1$ for the probability p_i corresponding to any path i . Thus the error covariance at the nodes in the layer 1 is stable. Now we can consider the path that has the maximum error covariance at time k in the layer 1. Repeating an argument similar to the one outline in Section V-C, we can obtain that all the nodes in layer 2, 3, \dots , $m - 1$ have stable estimate error covariance as well. Finally for the estimate at the destination node, all the estimates from the nodes in layer $m - 1$ can be considered to be in parallel with each other. Since for this network

$$p_{\max\text{-cut}} = \prod_{\text{all paths } j} p_j,$$

if $p_{\max\text{-cut}} |\rho(A)|^2 < 1$ the estimate at the destination have bounded error covariance as well. \blacksquare

Remark 4: We have provided a necessary and sufficient condition for the expected error covariance to remain bounded for a network of arbitrary topology. For any other causal data processing algorithm, it provides a necessary condition for stability. Let us, in particular, compare the stability conditions for the algorithm \mathcal{A}_2 to those for a simpler algorithm $\bar{\mathcal{A}}$ in which the intermediate nodes do not have *any* memory. At each time step k , the source

⁶Note that all $P_i^t(k)$'s form an ordered set.

node forwards the measurement $y(k-1)$. The intermediate nodes compare the time stamps of the measurements they received at the previous time step along different incoming edges and forward the most recent one. If they did not receive any measurement on the last time step, they do not transmit anything. It is clear that the probability that the destination node receives any particular measurement $y(k)$ from the source over the network is upper-bounded by the reliability of the network (see, e.g., [6]). Let us consider a simple example of a line network in which n edges each with drop probability p are combined in series. With our optimal algorithm, the necessary and sufficient condition for expected estimate error covariance to be stable is $p|\rho(A)|^2 < 1$. On the other hand, in algorithm \bar{A} , the probability that any measurement is received by the destination node is $q = 1 - (1-p)^n$. By a method similar to the one used in [27], it can be proven that a *necessary* condition for stability is $q|\rho(A)|^2 < 1$. As an example, for $n = 5$ links and drop probability $p = 0.2$, $q = 0.67$. Thus our algorithm improves the stability margin from $\rho(A) \leq 1.22$ for the simple algorithm to $\rho(A) \leq \sqrt{5}$.

VI. PERFORMANCE ANALYSIS

In this section we calculate the performance of the algorithm \mathcal{A}_2 for drops independent in time and uncorrelated in space. Once again, from (14), we realize that the cost function for the problem \mathcal{P}_1 can be calculated easily as long as we are able to calculate the steady-state expected estimate error covariance defined in (13). We now provide a framework to calculate the expected error covariance at *any* node. Let $Z_{uv}(k)$ be the difference between k and the time at which the last successful transmission before time k occurred on link (u, v)

$$Z_{uv}(k) = \min\{j \geq 1 | \lambda_{uv}(k+1-j) = \text{'received'}\}.$$

By convention, we adopt $Z_{uv}(k) = 1$. Thus, the last time that any message is received at node v from link (u, v) is $k - Z_{uv}(k) + 1$ and that message has time-stamp $t^u(k - Z_{uv}(k))$. Then (12) can be rewritten as

$$t^v(k) = \max_{u \in \mathcal{N}_v \cup v} t^u(k - Z_{uv}(k)). \quad (22)$$

Now $Z_{uv}(k)$ is distributed as a truncated geometric random variable with the density function

$$\text{Prob}(Z_{uv}(k) = i) = \begin{cases} (1 - p_{uv})p_{uv}^{i-1}, & 1 \leq i < k \\ 1 - \sum_{i=1}^{k-1} \text{Prob}(Z_{uv}(k) = i), & i = k. \end{cases}$$

We can get rid of the truncation by extending the definition of $t^u(k)$. For all $k < 0$, we define $t^u(k) = 0$. As an example, for the source node s , without extending the definition we have $t^s(k) = k - 1$ for $k \geq 1$. Using the extended definition, $t^s(k) = (k - 1)^+$ for all k , where $x^+ = \max\{0, x\}$. In general, using the extended definition of $t^u(k)$ for all k and for any node u , we can easily verify that (22) continues to hold; however, $Z_{uv}(k)$'s are now independent random variables distributed according to a geometric distribution with parameters p_{uv} 's. Thus

$$\text{Prob}(Z_{uv}(k) = i) = (1 - p_{uv})p_{uv}^{i-1} \quad \forall \quad i \geq 1, \quad \forall \quad k. \quad (23)$$

Since $Z_{uv}(k)$'s do not depend anymore on k , from now on, we omit the argument k . From (22), we can write $t^v(k)$ in terms of the time-stamp at the source node $(k-1)^+$ as

$$t^v(k) = \max_{P:\text{an s-v path}} (k-1 - \sum_{(u,v) \in P} Z_{uv})^+, \quad (24)$$

where the maximum is taken over all paths P in the graph \mathcal{G} from source s to the node v . Therefore the latency at node v can be written as

$$l^v(k) = k-1 - t^v(k) = \min\{k-1, \min_{P:\text{an s-v path}} (\sum_{(u,v) \in P} Z_{uv})\}.$$

Since we are interested in the steady-state expected error covariance, we consider the steady-state behavior of the latency $l^v(k)$. As $k \rightarrow \infty$, the distribution of $l^v(k)$ approaches that of the variable l^v defined as

$$l^v = \min_{P:\text{an s-v path}} (\sum_{(u,v) \in P} Z_{uv}). \quad (25)$$

Let us now concentrate on the destination node⁷. For the destination node d , we refer to l^d as the *steady-state latency* of the network. From (15), the steady-state error covariance can now be rewritten as

$$P(\infty) = \sum_{l=0}^{\infty} \text{Prob}(l^d = l) \left[A^l P^* A^l + \sum_{j=0}^{l-1} A^j Q A^j \right], \quad (26)$$

where P^* is the steady-state estimation error covariance of $x(k)$ based on $\{y(0), y(1), \dots, y(k-1)\}$ and is the solution to the Discrete Algebraic Riccati Equation (DARE)

$$P^* = AP^*A^T + R_w - AP^*C^T(CP^*C^T + R_v)^{-1}CP^*A^T.$$

Since the pair $\{A, R_w^{\frac{1}{2}}\}$ is stabilizable, the rate of convergence of $P(0)$ to P^* is exponential [16] and the substitution of P^* for $P(k-1)$ in (15) does not change the steady-state error covariance.

Let us define the generating function of the complementary density function $G(X)$ and the moment generating function $F(X)$ of the steady state latency l_d

$$G(X) = \sum_{l=0}^{\infty} \text{Prob}(l^d \geq l+1) X^l, \quad F(X) = \sum_{l=0}^{\infty} \text{Prob}(l^d = l) X^l, \quad (27)$$

where X is an arbitrary matrix. Thus

$$F(X) = (X - I)G(X) + I. \quad (28)$$

On vectorizing (26) we obtain

$$\begin{aligned} \text{vec}(P(\infty)) &= F(A \otimes A) \text{vec}(P^*) + G(A \otimes A) \text{vec}(Q) \\ &= ((A \otimes A - I)G(A \otimes A) + I) \text{vec}(P^*) + G(A \otimes A) \text{vec}(Q), \end{aligned} \quad (29)$$

where $A \otimes B$ is the Kronecker product of matrices A and B . Thus the performance of the system depends on the value of $G(X)$ evaluated at $X = A \otimes A$. In particular, the system is stable if and only if $G(X)$ is bounded at

⁷If we are interested in the error covariance at some other node v , simply denote v as the destination node.

$A \otimes A$. Since $G(X)$ is a power series, boundedness of $G(x)$ at $A \otimes A$ is equivalent to the boundedness of $G(x)$ (evaluated for a scalar x) at the square of the spectral radius of A . We thus have the following result.

Proposition 7: Consider a process of the form (11) being observed using a sensor of the form (2) through an arbitrary network of packet dropping links. Let the packet drops be independent from one time step to the next and across links. Then the minimum expected steady-state estimation error covariance at the receiver is given by (29). Furthermore, the error covariance is stable, in the sense of bounded expected steady-state error, iff $|\rho(A)|^2$ lies in the region of convergence of $G(x)$, where $\rho(A)$ is the spectral radius of A .

The above theorem allows us to calculate the steady state expected error covariance for any network as long as we can evaluate the function $G(X)$ for that network. We now consider some special networks and evaluate the performance explicitly. We start with a network consisting of links in series, or a line network.

A. Networks with Links in Series

In this case, the network consists of only one path from the source to the destination. Thus we have

$$F(X) = E \left[X^{l^d} \right] = E \left[X^{\sum_{(u,v)} Z_{uv}} \right],$$

where the summation is taken over all the edges in the path. Since the drops across different links are uncorrelated, the variables Z_{uv} 's are independent. Thus we have

$$F(X) = E \left[X^{\sum_{(u,v)} Z_{uv}} \right] = \prod_{(u,v)} E \left[X^{Z_{uv}} \right],$$

where we have used the independence of Z_{uv} 's. Since Z_{uv} is a geometric random variable (23),

$$E \left[X^{Z_{uv}} \right] = (1 - p_{uv})X (I - p_{uv}X)^{-1},$$

provided that $\rho(X)p_{uv} < 1$, where $\rho(X)$ is the spectral radius of matrix X . Therefore,

$$F(X) = E \left[X^{l^d} \right] = \prod_{(u,v)} \left[(1 - p_{uv})X (I - p_{uv}X)^{-1} \right].$$

Using partial fractions and the relation in (28), we then obtain

$$G(X) = \sum_{i=0}^{n-1} X^i + X^n \sum_{(u,v)} c_{uv} \frac{p_{uv}}{1 - p_{uv}} (I - p_{uv}X)^{-1},$$

where

$$c_{uv} = \left(\prod_{(r,s) \neq (u,v)} \left(1 - \frac{p_{uv}}{p_{rs}} \right) \right)^{-1}.$$

Therefore the cost can be written as

$$\text{vec}(P(\infty)) = \prod_{(u,v)} \left[(A \otimes A) \left(\frac{I - p_{uv}A \otimes A}{1 - p_{uv}} \right)^{-1} \right] \text{vec}(P^*) + G(A \otimes A) \text{vec}(Q). \quad (30)$$

We can also see from the above argument that the system is stable if for every link (u, v) we have $p_{uv}|\rho(A)|^2 < 1$ or equivalently $\max_{(u,v)} p_{uv}|\rho(A)|^2 < 1$. This matches with the condition in section V. Also note that for the case

that some of p_{uv} 's are equal, a different partial fraction expansion applies. In particular for the case when there are n links all with the erasure probability p , we obtain

$$\begin{aligned} \text{vec}(P(\infty)) &= (A \otimes A)^n \left(\frac{I - pA \otimes A}{1 - p} \right)^{-n} \text{vec}(P^*) \\ &\quad + \sum_{i=0}^{n-1} \left[\frac{p}{1-p} (A \otimes A)^n \left(\frac{I - pA \otimes A}{1-p} \right)^{-i-1} \right] \text{vec}(Q) + \sum_{i=0}^{n-1} (A \otimes A)^i \text{vec}(Q). \end{aligned} \quad (31)$$

Finally, when there is only one link between the source and the destination, the cost reduces to a particularly simple form. It is easily seen that, in that case, the steady state error covariance is the solution to the Lyapunov equation

$$P(\infty) = \sqrt{p}AP(\infty)\sqrt{p}A + (Q + (1-p)AP^*A).$$

This expression can alternately be derived using Markov jump linear system theory as in [11].

B. Network of Parallel Links

Consider a network with the sensor connected to a destination node through n links with probabilities of packet drop p_1, \dots, p_n . Since the same data is transmitted over all the links, using (25) the steady state latency is $l_d = \min_{1 \leq i \leq n} (Z_i)$. Since Z_i 's are all independent geometrically distributed variables with parameters p_i 's respectively, their minimum is itself geometrically distributed with parameter $p_{eq} = \prod_i p_i$. Thus $F(X)$ can be evaluated as

$$F(X) = (1 - p_{eq})X(I - p_{eq}X)^{-1},$$

and $G(X)$ can, in turn, be written as

$$G(X) = (I - \prod_i p_i X)^{-1}.$$

Thus the steady-state error can be evaluated using (29). Note that the region of convergence of $G(X)$ enforces for stability $\prod_i p_i |\rho(A)|^2 < 1$, which again matches with the condition in Section V.

C. Arbitrary Network of Parallel and Serial Links

We can similarly obtain the following two rules for the steady-state error covariance of any network derived from the parallel and serial concatenations of sub-networks. Let $l_d(\mathcal{G})$ denote the steady-state latency function of network \mathcal{G} . Also given two subnetworks \mathcal{G}_1 and \mathcal{G}_2 , denote their series combination by $\mathcal{G}_1 \oplus \mathcal{G}_2$ and their parallel combination by $\mathcal{G}_1 \parallel \mathcal{G}_2$.

- 1) Suppose the network \mathcal{G} can be decomposed as a series of two subnetworks \mathcal{G}_1 and \mathcal{G}_2 . Since packet erasures in the two subnetworks are independent of each other, we have

$$l^d(\mathcal{G}_1 \oplus \mathcal{G}_2) = l^d(\mathcal{G}_1) + l^d(\mathcal{G}_2).$$

Thus we obtain

$$\begin{aligned} F_{\mathcal{G}_1 \oplus \mathcal{G}_2}(X) &= E \left[X^{l^d(\mathcal{G}_1 \oplus \mathcal{G}_2)} \right] \\ &= E \left[X^{l^d(\mathcal{G}_1) + l^d(\mathcal{G}_2)} \right] \\ &= E \left[X^{l^d(\mathcal{G}_1)} \right] E \left[X^{l^d(\mathcal{G}_2)} \right] = F_{\mathcal{G}_1}(X) F_{\mathcal{G}_2}(X). \end{aligned}$$

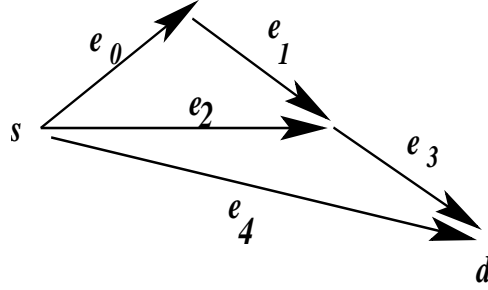


Fig. 2. Example of a network of combination of parallel and serial links

Finally using (28), the complementary density function of the network \mathcal{G} is given by

$$\begin{aligned} G_{\mathcal{G}_1 \oplus \mathcal{G}_2}(X) &= (X - I)^{-1} (F_{\mathcal{G}_1 \oplus \mathcal{G}_2}(X) - I) \\ &= G_{\mathcal{G}_1}(X) (X - I) G_{\mathcal{G}_2}(X) + G_{\mathcal{G}_1}(X) + G_{\mathcal{G}_2}(X) \\ &= (X - I) G_{\mathcal{G}_1}(X) G_{\mathcal{G}_2}(X) + G_{\mathcal{G}_1}(X) + G_{\mathcal{G}_2}(X), \end{aligned}$$

where in the last line we have used the fact that $G(X)(X - I) = (X - I)G(X)$.

2) If the network \mathcal{G} can be decomposed as parallel combination of two sub-networks \mathcal{G}_1 and \mathcal{G}_2 , we have

$$l^d(\mathcal{G}_1 \parallel \mathcal{G}_2) = \min\{l^d(\mathcal{G}_1), l^d(\mathcal{G}_2)\}.$$

Once again, the erasures in the two subnetworks are independent of each other. Thus

$$\text{Prob}(l^d(\mathcal{G}_1 \parallel \mathcal{G}_2) \geq l) = \text{Prob}(l^d(\mathcal{G}_1) \geq l) \text{Prob}(l^d(\mathcal{G}_2) \geq l).$$

Thus we see that if

$$G_{\mathcal{G}_1}(X) = \sum_{i=0}^{\infty} a_i X^i \quad G_{\mathcal{G}_2}(X) = \sum_{i=0}^{\infty} b_i X^i,$$

then

$$G_{\mathcal{G}_1 \parallel \mathcal{G}_2}(X) = \sum_{i=0}^{\infty} a_i b_i X^i.$$

Thus we can use (29) with the above two rules to derive the steady state error of any network consisting of links in series and parallel with each other. As an example consider the network depicted in Figure 2. In this case

$$\mathcal{G} = (((\mathcal{G}_0 \oplus \mathcal{G}_1) \parallel \mathcal{G}_2) \oplus \mathcal{G}_3) \parallel \mathcal{G}_4),$$

where each of the sub-networks \mathcal{G}_i is just a link with probability of packet drop p . The generating function of a link with erasure probability p is given by $G(X) = (I - pX)^{-1}$. Moreover, for a subnetwork with generating function $G(X)$ in parallel with a link with erasure probability p , the generating function of the entire network is given by $\mathcal{L}_p(G)(X)$, where \mathcal{L}_p is an operator such that $\mathcal{L}_p(G)(X) = G(pX)$. Thus for the network,

$$G(X) = \mathcal{L}_p(\mathcal{L}_p(G_0 * G_1) * G_3)(X)$$

where

$$G_i(X) = (I - pX)^{-1}, \quad i = 0, 1, 3$$

is the generating function for the i -th link and $G_i * G_j$ denotes the generating function of the series combination of link i and j . The steady state error covariance can thus be evaluated.

D. Networks with Arbitrary Topology

In this section, we consider the performance of general networks. Finding the distribution of the steady-state latency l_d of a general network is not an easy task because different paths may overlap. This can introduce dependency in the delays incurred along different paths and the calculation of the minimum delay and hence the steady-state latency becomes involved. However, using a method similar to the one used in Section V, we can provide upper and lower bounds on the performance. We first mention the following intuitive lemma without proof.

Lemma 2: Let $P^\infty(\mathcal{G}, \{p_{uv}, (u, v) \in \mathcal{E}\})$ denote the expected steady-state error of a system with communication network represented by graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and probabilities of packet drop $p_{uv}, (u, v) \in \mathcal{E}$. Then the expected steady-state error is non-increasing in p_{uv} 's, i.e., if $p_{uv} \leq q_{uv} \quad \forall (u, v) \in \mathcal{E}$

$$P^\infty(\mathcal{G}, \{p_{uv}, (u, v) \in \mathcal{E}\}) \leq P^\infty(\mathcal{G}, \{q_{uv}, (u, v) \in \mathcal{E}\}),$$

where $A \leq B$ means that $B - A$ is positive semi-definite.

1) *Lower Bound:* Using the above lemma we can lower bound the steady-state error by making a subset of links erasure free. This is similar to the method we used to obtain a necessary condition for stability in section V. Thus once again consider any cut-set of the network. Setting the probability of erasure equal to zero for every link except those crossing the cut (i.e., of the form (u, v) where u is in the source set and v in the destination set) gives a lower bound on the error. Therefore,

$$P^\infty(\mathcal{G}, \{p_{uv}, (u, v) \in \mathcal{E}\}) \geq P^\infty(\mathcal{G}, \{q_{uv}, (u, v) \in \mathcal{E}\}),$$

where

$$q_{uv} = \begin{cases} p_{uv} & (u, v) \text{ crosses the cut} \\ 0 & \text{otherwise.} \end{cases}$$

Now $P^\infty(\mathcal{G}, \{q_{uv}, (u, v) \in \mathcal{E}\})$ can be evaluated using the results given above for a network of parallel links. By considering the maximum along all possible cut-sets, we obtain the closest lower bound.

2) *Upper Bound:* We use a method similar to the one used to obtain the sufficient condition for stability in Section V. In the proof of Proposition 6, it is shown that the performance of the network \mathcal{G} is lower bounded by the performance of another network \mathcal{G}' that has series and parallel links only and has the following properties:

- \mathcal{G} and \mathcal{G}' have the same node set.
- \mathcal{G}' is the combination of edge-disjoint paths from the source to destination.
- The value of the max-cut in \mathcal{G}' is the same as in the original network \mathcal{G} .

The performance of \mathcal{G}' can be computed based on the results given above for arbitrary networks composed of subnetworks in series and parallel. This provides an upper bound on the performance of the original network.

VII. EXAMPLES

We now illustrate the above results using some simple examples. Consider a scalar process evolving as

$$x(k+1) = 0.8x(k) + w(k),$$

that is being observed through a sensor of the form

$$y(k) = x(k) + v(k).$$

The noises $w(k)$ and $v(k)$ are assumed zero-mean, white, independent and Gaussian with unit variances.

Suppose that the source and the destination node are connected using two links in series, each with a probability of packet erasure p . Figure 3 shows the performance of our strategy for the estimation problem \mathcal{P}_2 as the probability p is varied. The simulation results refer to data generated by 20000 random runs, each over 200 time steps while the theoretical values refer to the value predicted by (29). We can see that the two sets of values match quite closely, with the theoretical results lying within the error bars that denote the 90% confidence interval.

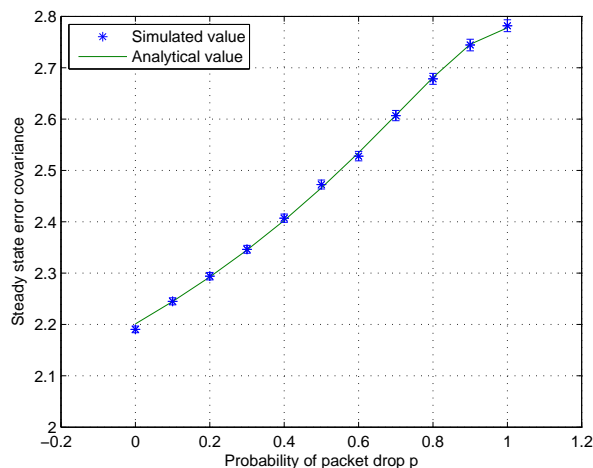


Fig. 3. Simulated and theoretical results for a line network.

We also carried out a similar exercise for the source and destination nodes connected by two links in parallel, each with packet erasure probability p . The results are plotted in Figure 4. We can once again see that the simulated values match quite closely with the theoretical values.

Finally, we consider the source and destination nodes connected by a bridge network shown in Figure 5. We assume all the links in the network to have probability of erasure p . This network cannot be reduced to a series and parallel sub-networks. We can, however, calculate the performance analytically in this particular case and compare it to the upper and lower bounds presented earlier. The networks used for calculating the bounds are also shown in Figure 5. The bounds can be computed from the results for series and parallel networks. Figure 6 shows a comparison of the analytical and simulated values with the bounds. We can see that the bounds are tight.

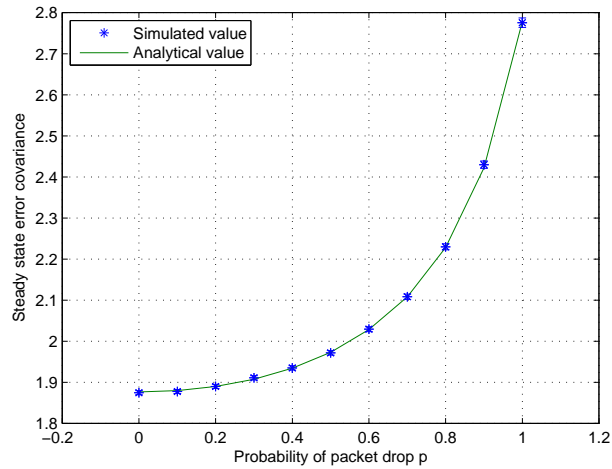


Fig. 4. Simulated and theoretical results for a parallel network.

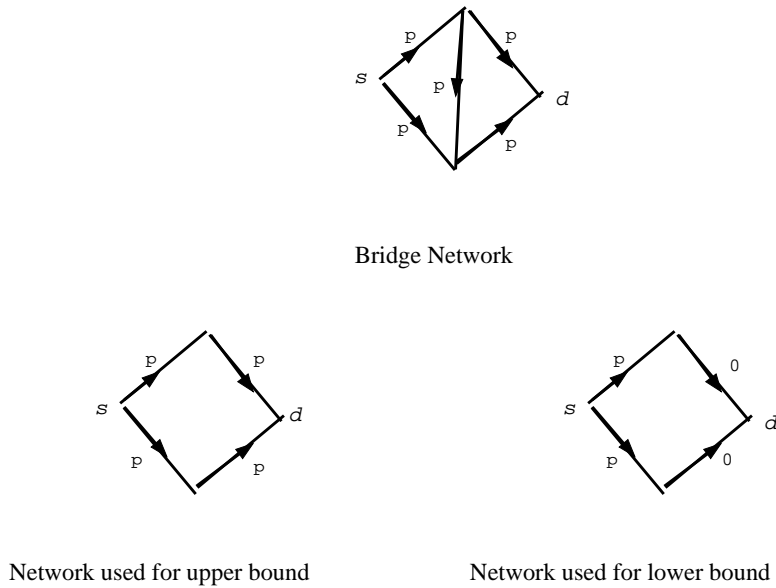


Fig. 5. Bridge network and the networks used for calculating lower and upper bounds.

We can also compare the performance of our algorithm with that obtained if no encoding were done and only measurements were transmitted. Consider the process

$$x(k+1) = \begin{bmatrix} 1.25 & 0 \\ 1 & 1.1 \end{bmatrix} x(k) + w(k)$$

being observed by a sensor of the form

$$y(k) = x(k) + v(k),$$

where $w(k)$ and $v(k)$ are white independent Gaussian noises with means zero and covariance identity. We consider

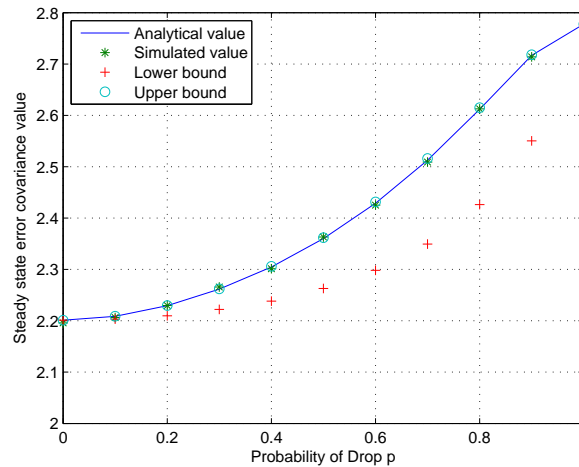


Fig. 6. Simulated values and theoretical bounds for the bridge network.

transmission of data across a series of n channels. Figure 7 shows the difference in simulated performance of the algorithm in which no encoding is done and for our algorithm for various values of n . It can be seen that the error covariance is much more if no encoding is being done, even for a few number of links and moderate values of drop probability. For each point we did 10000 simulations with each simulation being 100 time steps long.

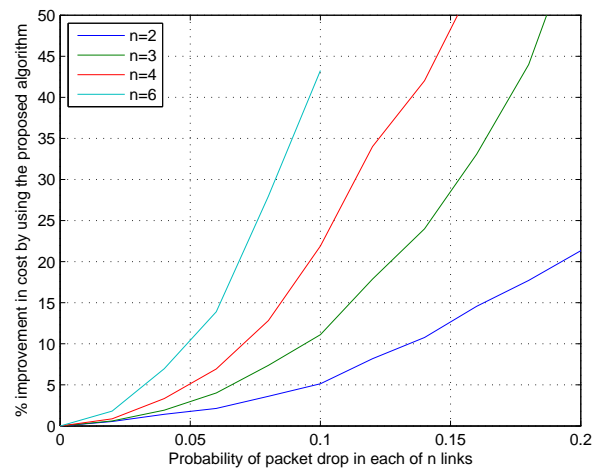


Fig. 7. Simulated difference in performance between algorithm in which no encoding is done and our optimal algorithm for series connection of n links.

As a final example, we consider the process

$$x(k+1) = 1.2x(k) + u(k) + w(k)$$

being observed through a sensor of the form

$$y(k) = x(k) + v(k),$$

which communicates with the controller over a series network of n links each with packet drop probability p . The controller is interested in minimizing the infinite-time quadratic cost J_∞ with the cost matrices Q and R as unity. The noise variances R_w and R_v are also assumed to be unity. Figure 8 shows the variation of the cost with the probability for different number of links n . The performance degrades rapidly with the increase in number of links.

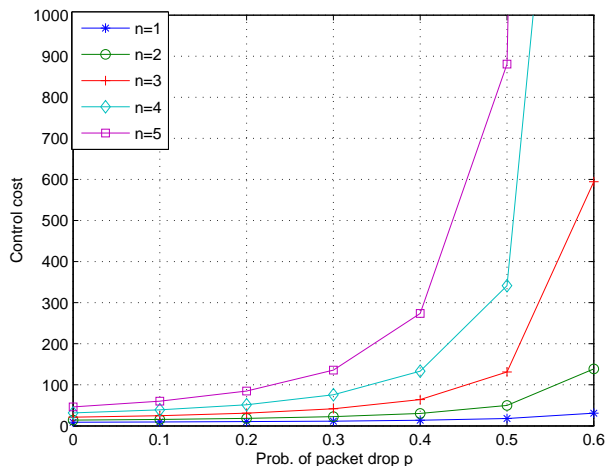


Fig. 8. Loss in performance as a function of packet drop probability for n links in series.

VIII. GENERALIZATIONS

A. Correlated erasure events

Even though the algorithm \mathcal{A}_2 is optimal for any packet dropping pattern, the stability and performance analysis so far assumed that the erasure events are memoryless and independent across different links in the network. We could thus formulate the performance in terms of a generating function of the steady-state latency distribution as defined in (25). We now look at the effect of dropping these assumptions.

1) *Markov events*: If we assume that the drop events on each link are governed by a Markov chain (but are still independent of other links), we can obtain the performance as follows. Suppose that the packet drop event on link (u, v) , denoted by $\lambda_{uv}(k) = \text{'dropped'}$ evolves according to a Markov chain with transition matrix M_{uv} . We further assume that M_{uv} is irreducible and reversible. Let us first consider the case where the initial distribution of packet drop on each link is the stationary distribution of the Markov chain on that link. Then we can rewrite (12)

in a similar fashion as (24) with Z_l being a geometric random variable with distribution

$$\text{Prob}(Z_{uv} = l) = \begin{cases} \alpha_{uv} M_{uv}(1, 2) M_{uv}(1, 1)^{l-2} & \forall l \geq 2 \\ 1 - \alpha_{uv} & l = 1 \end{cases},$$

where α_{uv} is the probability of packet drop based on the stationary distribution of link $e = (u, v)$ and $M_{uv}(i, j)$ as the (i, j) -th element of M_{uv} . Thus, all the previous analysis goes through. In particular, the stability condition is

$$\max_{c: s-d \text{ cut}} \prod_{(u,v) \in c} M_{uv}(1, 1) |\rho(A)|^2 < 1.$$

Now, if the initial distribution is not the stationary distribution, the variables $Z_{uv}(k)$ are not time-independent and the analysis does not go through. However, since for large k the Markov chains approach their stationary distribution, the stability condition remains unchanged.

2) *Spatially correlated events*: Suppose that the packet drop events are correlated across the network but memoryless over time. In other words, at each time step k , the packet drop events occur according to distribution $\text{Prob}(\lambda_{uv}, (u, v) \in \mathcal{E})$. Now $Z_{uv}(k)$'s are not independent across the network and hence finding the steady-state error covariance does not seem to be tractable. However, we can find the condition for stability. For this, we define a generalized notion of equivalent probability of packet drop for correlated events. Consider a cut-set c , and let $\mathcal{B}(c)$ denote the set of edges crossing this cut. Then the equivalent probability of packet drop for this cut is

$$p_{eq}(c) = \text{Prob}(\lambda_{uv} = \text{'dropped'}, \quad \forall (u, v) \in \mathcal{B}(c)).$$

The value of the max-cut for the network is the maximum of $p_{eq}(c)$ over all the cuts,

$$p_{mc}(\mathcal{G}) = \max_{\text{all cut-sets } c} p_{eq}(c).$$

We can then show that the condition for stability of the system is $p_{mc}(\mathcal{G}) |\rho(A)|^2 < 1$. To see this, consider the scenario when only one packet is to be routed from the source to destination starting at time t_0 . For each time-step $t \geq t_0$ let $\mathcal{V}_r(t)$ denote the set of nodes that have received the packet at time t . Clearly $\mathcal{V}_r(t_0) = \{s\}$. We want to bound the probability that at time $t_0 + T$, destination node has not yet received the packet. Note that for every time-step between t_0 and $t_0 + T$, $\mathcal{V}_r(t)$ defines a cut-set since it contains s and not d . Now the size of $\mathcal{V}_r(t+1)$ does not increase with respect to time-step t iff all the links that cross the cut generated by $\mathcal{V}_r(t)$ drop packets. However by the definition of $p_{mc}(\mathcal{G})$ the probability of this event is at most $p_{mc}(\mathcal{G})$. Therefore, we have

$$|\mathcal{V}_r(t+1)| \begin{cases} \geq |\mathcal{V}_r(t)| + 1 & \text{with prob. at most } p_{mc}(\mathcal{G}) \\ = |\mathcal{V}_r(t)| & \text{with prob. at least } 1 - p_{mc}(\mathcal{G}) \end{cases}$$

Thus for large T , the probability that at time $t_0 + T$ the destination node has not received the packet is upper bounded by $n(1 - p_{mc}(\mathcal{G}))^n T^n p_{mc}(\mathcal{G})^{T-n}$, where n is the number of nodes in the network. Although a new packet is generated at the source at each time step, the importance of the packets is increasing with time; hence the error can be upper-bounded by considering that the network is only routing the packet generated at time $k - l$. The probability that the latency is larger than l grows like $f(l) p_{mc}(\mathcal{G})^l$, where $f(l)$ is polynomial in l with bounded degree and the sufficiency of the stability condition follows. The necessity part involves similar ideas and is omitted.

B. Synthesis of a Network

One can use the performance results above to design networks for the purpose of estimation and control. To consider a simple example, consider a scalar system observed by sensor s . Assume that the destination is located at distance d_0 from the sensor. The probability of dropping a packet on a link depends on its physical length. A reasonable model for probability of dropping packets is given by⁸ $p(d) = 1 - \exp(-\beta d^\alpha)$, where β and α are positive constants. α denotes the exponent of power decay in the wireless environment. We are interested in the optimal number n of relay nodes that we should place between the sensor and the destination so as to minimize the expected steady-state error covariance (or in turn to minimize the cost J_∞). Clearly, there is a trade-off involved since more nodes reduce the probability of erasure but at the same time lead to a higher delay before the destination receives a packet. Assuming that n sensors are uniformly placed, there are $n + 1$ links each with drop probability q . Thus from (31), $P(\infty)$ can be written as

$$P(\infty) = \left(\frac{a^2(1-q)}{1-qa^2} \right)^{n+1} \left(P^* + \frac{R_w}{a^2-1} \right) - \frac{R_w}{a^2-1}.$$

Thus the optimal number of relay nodes, (assuming that $a^2 > 1$) is the solution to the problem

$$\min_n \left(\frac{a^2(1-p(\frac{d_0}{n+1}))}{1-p(\frac{d_0}{n+1})a^2} \right)^{n+1}.$$

If $a^2 < 1$ then the minimization in the above problem is replaced with maximization.

C. Unicast Networks

So far, we have assumed that the topology of the network was fixed and that a node could transmit a message on all the out-going edges. We can consider networks that are unicast in the sense that each node should choose one out of a set of possible edges to transmit the message on. There are two parts of the problem:

- 1) Choose the optimal path for data to flow from the source node to the destination node.
- 2) Find the optimal information processing scheme to be followed by each node.

The two parts can clearly be solved separately in the sense that given any path, the optimal processing strategy is the algorithm described in Section IV. To choose the optimal path, we need to define a metric for the cost of a path. We can consider two choices:

- 1) If the metric is the condition for stability, then the problem can be recast as choosing the shortest path in a graph with the length of a path being given by its equivalent probability of packet drop. Thus we need to find the path that has the minimum p_{path} . Since each path is just a line network, this reduces to the problem $\min_{P:s-d \text{ path}} \max_{(u,v) \in P} p_{uv}$. The above problem is well studied in the computer science community and can be solved as a shortest-path problem over a min-max semi-ring in a distributed fashion [20].

⁸This expression can be derived by considering the probability of outage in a Rayleigh fading environment.

- 2) If the metric is performance, the problem is more complicated in general. We can consider the special case of a scalar system and no process noise. In this case, from (30), we have for path q ,

$$\log P_q(\infty) = \sum_{e \in q} \log\left(\frac{(1-p_e)a^2}{1-p_e a^2}\right)$$

Now the problem is equivalent to

$$\min_{q: \text{s-d path}} \sum_{e \in q} \log\left(\frac{(1-p_e)a^2}{1-p_e a^2}\right)$$

This problem can also be solved in a distributed way [5].

IX. CONCLUSIONS

In this paper, we considered the problem of estimation and control of a process across an arbitrary network of communication links. We identified the optimal information processing strategy to be followed by each node in the network that allows the estimator to calculate the best possible estimate in the minimum mean square sense and the controller to minimize a quadratic cost. The recursive algorithm that we propose requires a constant amount of memory, processing and transmission at every node in the network per time step yet is optimal for any packet-dropping process and at every time step. It has numerous other nice properties as well such as being able to take care of delays and packet reordering. For the case when the drop probabilities are memoryless and independent across links, we also carried out the stability and performance analysis for this algorithm. We characterized any arbitrary network of packet erasure links in terms of its ability to support data transmission for stable estimation of a dynamic process. The results can also be used for the problems of routing of data through a network and synthesis of networks for the purpose of estimation and control.

REFERENCES

- [1] P. Antsaklis and J. Baillieul, "Special issue on networked control systems," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, 2004.
- [2] B. Azimi-Sadjadi, "Stability of networked control systems in the presence of packet losses," in *Proceedings of 2003 IEEE Conference on Decision and Control*, Dec 2003.
- [3] L. Bushnell, "Special issue on networks and control," *IEEE Control Systems Magazine*, vol. 21, no. 1, February 2001.
- [4] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver, *Combinatorial Optimization*. John Wiley and Sons, New York, 1998.
- [5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press and McGraw-Hill, 1990.
- [6] K. Dohmen, "Inclusion-exclusion and network reliability," *The Electronic Journal of Combinatorics*, vol. 5, no. 1, p. Research paper 36, 1998.
- [7] D. Georgiev and D. M. Tilbury, "Packet-based Control: The H_2 -optimal Solution", *Automatica*, 42(1), January 2006, pp. 137-144.
- [8] S. Graham, G. Baliga and P. R. Kumar, "Issues in the convergence of control with communication and computing: Proliferation, architecture, design, services, and middleware", *Proceedings of the 43rd IEEE Conference on Decision and Control*, Bahamas, 2004, pp. 1466-1471.
- [9] V. Gupta, S. Adalakha and B. Sinopoli, "Towards Receding Horizon Networked Control", *Automatica*, Submitted. See also V. Gupta, S. Adalakha and B. Sinopoli, "Towards Receding Horizon Networked Control", Allerton conference on communication, control and computing, 2006.
- [10] V. Gupta, T. H. Chung, B. Hassibi, and R. M. Murray, "On a stochastic sensor selection algorithm with applications in sensor scheduling and sensor coverage," *Automatica*, vol. 42, no. 2, pp. 251-260, February 2006.

- [11] V. Gupta, D. Spanos, B. Hassibi, and R. M. Murray, "Optimal LQG control across a packet-dropping link," *Systems and Controls Letters*, 2006, accepted. See also, V. Gupta, D. Spanos, B. Hassibi, and R. M. Murray, "Optimal LQG control across a packet-dropping link," *Proc. of the American Control Conference (ACC)*, 2006.
- [12] C. N. Hadjicostis and R. Touri, "Feedback control utilizing packet dropping network links," in *Proc. of the IEEE Conference on Decision and Control*, 2002.
- [13] B. Hassibi, A. H. Sayed, and T. Kailath, *Indefinite-Quadratic Estimation and Control*. Studies in Applied and Numerical Mathematics, 1999.
- [14] J. P. Hespanha, P. Naghshtabrizi and Y. Xu, "Networked Control Systems: Analysis and Design," To appear in the Proc. of IEEE, Special Issue on Networked Control Systems, 2007.
- [15] O.C. Imer, S. Yüksel and T. Basar, "Optimal Control of LTI Systems over Communication Networks", *Automatica*, 42(9):1429-1440, September 2006.
- [16] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*. Prentice Hall, 2000.
- [17] P. A. Kawka and A. G. Alleyne, "Stability and performance of packet-based feedback control over a Markov channel", Proceedings of the American Control Conference, 2006, pp. 2807-2812.
- [18] F.L. Lian, J.R. Moyne and D.M. Tilbury, "Performance evaluation of control networks", *IEEE Control Systems Magazine* 21 (2001) (1), pp. 6683.
- [19] Q. Ling and M. D. Lemmon, "Power spectral analysis of networked control systems with data dropouts," *IEEE Transactions on Automatic control*, vol. 49, no. 6, pp. 955-960, June 2004.
- [20] M. Mohri, "Semiring frameworks and algorithms for shortest-distance problems," *Journal of Automata, Languages and Combinatorics*, pp. 321-350, 2002.
- [21] L. A. Montestruque and P. Antsaklis, "On the Model-based Control of Networked Systems", *Automatica*, 39(10), October 2003, pp. 1837-1843.
- [22] P. Naghshtabrizi and J. Hespanha, "Anticipative and Non-anticipative Controller Design for Network Control Systems", In Panos J. Antsaklis, Paulo Tabuada, *Networked Embedded Sensing and Control*, volume 331 of *Lect. Notes in Contr. and Inform. Sci.*, pages 203218, 2006.
- [23] J. Nilsson, "Real-time control systems with delays," Ph.D. dissertation, Department of Automatic Control, Lund Institute of Technology, 1998.
- [24] C. Robinson and P. R. Kumar, "Control over networks of unreliable links: Location of controllers, control laws and bounds on performance," in *45th IEEE Conference on Decision and Control*, 2006, submitted.
- [25] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla and S. S. Sastry, "Foundations of Control and Estimation over Lossy Networks", *Proc. of the IEEE*, To Appear 2006.
- [26] P. J. Seiler, "Coordinated control of unmanned aerial vehicles," Ph.D. dissertation, University of California, Berkeley, 2002.
- [27] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry, "Kalman filtering with intermittent observations," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1453-1464, Sep 2004.
- [28] S.C. Smith and P. Seiler, *Estimation with lossy measurements: jump estimators for jump systems*. *IEEE Transaction on Automatic Control*, 48(12):1453-1464, 2003.
- [29] S. Tatikonda, "Some scaling properties of large scale distributed control systems," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, vol. 3, 2003, pp. 3142-3147.
- [30] Y. Xu and J. P. Hespanha, "Optimal Communication Logics for Networked Control Systems," In *Proceedings of the 43rd IEEE Conference on Decision and Control*, 2004.
- [31] W. Zhang, M. S. Branicky, and S. M. Philips, "Stability of networked control systems," *IEEE Control System Magazine*, vol. 21, no. 1, pp. 84-89, Feb 2001.