CSR–EHS: Integrating Decentralized Control and Real-Time Scheduling for Networked Dynamical Systems

Michael Lemmon and Xiaobo Sharon Hu, University of Notre Dame

1.0 Introduction:

A networked dynamical system consists of numerous loosely coupled systems. These networked systems are found throughout our national infrastructure in the guise of our electrical power grid and transportation networks. Interruptions in the smooth operation of these networks can disrupt the lives of millions of users. Increased demand due to demographic shifts and greater regulatory burdens have made it more difficult to operate these networks in a cost effective manner. Tighter coupling between the constituent subsystems in the network can make it easier for pointwise disturbances to blossom into system wide outages. Good examples of such failures appear frequently in our electrical power grid. There is, therefore, a compelling national need to devise more **robust** and **cost effective** methods for controlling such networked systems.

Many factors effect the behavior of a networked dynamical system. These include location dependent factors, uncertain external disturbances, timely completion of tasks at each computational core, timely transmission of required messages through the network, to name a few. Traditionally, these factors are dealt with by domain specific experts. That is, control engineers derive sophisticated control algorithms to handle disturbances based on the periodically sampled systems, while computer scientists design scheduling algorithms in an effort to satisfy the timing requirements. Unfortunately, due to the unpredictable nature of disturbances suffered by many physical systems and the inherently dynamic states of networks (particularly wireless ones), these approaches invariably sacrifice either robustness or cost effectiveness.

This project aims to improve significantly the robustness and cost effectiveness of controllers for networked dynamical systems through investigating several innovative approaches. First, based on recent advances in **self-triggered** feedback control systems, we propose a method to adaptively determine the sampling periods of a distributed control system. This represents a radical departure from the traditional use of fixed sampling periods. We then study distributed scheduling approaches that are suitable for such self-triggered control systems. Centralized scheduling is avoided to achieve scalability. Finally, we examine the interplay of control and real-time scheduling. We believe that our framework provides a seamless integration of control and real-time scheduling and thus can achieve robust systems in a cost effective manner.

2.0 Mathematical Preliminaries:

Let's first consider a set of *decoupled* dynamical systems. The state of the ith subsystem is the function, $x_i : \Re \to \Re^{n_i}$ that satisfies the ordinary differential equation,

$$\dot{x}_i(t) = f_i(x_i(t), w_i(t)) + g_i(x_i(t))u_i(t)$$

where $i = 1, \ldots, N$. The functions $u_i : \Re \to \Re^{m_i}$ and $w_i : \Re \to \Re^{p_i}$ are the *control* input and *disturbance* input, respectively. We assume that $f_i(0,0) = 0$ so the unforced decoupled system's equilibrium point lies at the origin. For simplicity we assume the system state is observable so we can use a *centralized* state feedback controller, $k_i : \Re^{n_i} \to \Re^{m_i}$ of the form, $u_i(t) = k_i(x_i(t))$. This controller is chosen to *reject* the input disturbance, w_i, at the system's state, x_i. The amount of disturbance rejection is measured by the system *gain*. In particular, let $\mathcal{S}_i$ denote a functional mapping the ith disturbance input, w_i, onto the state function, x_i, under the state feedback control. $\mathcal{S}_i$ therefore represents the *closed-loop* dynamical system. The *gain* of this system is denoted as $\|\mathcal{S}_i\|$ where

$$\|\mathcal{S}_i\| = \sup_{w_i} \frac{\|\mathcal{S}_i[w_i]\|_2}{\|w_i\|_2}$$

and $\|\cdot\|_2$ is the usual $\mathcal{L}_2$ function norm corresponding to signal energy. The controller, k_i, is chosen to ensure that $\|\mathcal{S}_i\| \leq \gamma$ where γ is a constant specified by the designer. In this case γ represents the specified performance level of the system. The smaller this value is, the greater the disturbance is rejected at the output.

The framework in the preceding paragraph assumes that all subsystems are decoupled. We extend these equations to include weak coupling between subsystems. This is done by requiring that the state function, x_i, satisfy the following differential equations,

$$\dot{x}_i(t) = f_i(x_i(t), w_i(t)) + g_i(x_i(t))u_i(t) + \sum_{j \in \mathcal{N}_i} h_{ij}(x_j(t)) \tag{1}$$

In this equation, the coupling between subsystems is modeled by the summation. This sum is taken over all neighbors of the ith subsystem. This neighborhood set is denoted as $\mathcal{N}_i \subset \{1, \cdots, N\}$. The "size" of the function $h_{ij} : \Re^{n_j} \to \Re^{n_i}$ characterizes the degree of coupling between subsystems.

Decentralized controllers for the system in equation (1) use feedback to limit the coupling between subsystems to the point where the "local" controller, k_i, is able to do its job. These controllers only use information from their immediate neighbors to achieve their objective so that no single controller observes the state of the entire network. The decentralized controller for the ith subsystem might have the form,

$$u_i(t) = k_i(x_i(t)) + \sum_{j \in \mathcal{N}_i} \ell_{ij}(x_j(t))$$

where k_i is the local controller and $\ell_{ij} : \Re^{n_j} \to \Re^{n_i}$ is a *decoupling* controller that essentially tries to cancel out the coupling between the subsystems.

This ad hoc approach to controlling loosely coupled dynamical systems has been in use for a long time. Early work in decentralized control dates back to the 1970's where researchers enforced diagonal dominance conditions on the system matrices for networks of linear systems [46]. A review of this work will be found in [43]. This control was rarely optimized and recent work in [12] and [22] has tackled that issue by recasting the networked system as a linear fractional transformation. This enables the use of well known optimal and robust controller synthesis methods, again for linear systems. Extensions of these ideas to nonlinear systems are possible using Lyapunov analysis methods. Very recent efforts in this direction developed a "distributed" receding horizon control strategy [13] to decentralized control.

All of the aforecited work, however, presumes that the neighboring state information can be accessed at any time. This is, of course, not possible in practice. The subsystems in real-life networked systems (such as the electric power grid) are separated by large physical distances. This means that state information must be *communicated* between subsystems and usually this communication takes place over packet-switched networks. Packet switched networks only transmit messages at discrete instants in time. Let's associate the following monotone increasing sequence $\{r_i[k]\}_{k=1}^{\infty}$ of time instants with the ith subsystem. The time $r_i[k]$ denotes the time when the neighbors of subsystem i sampled their states for the kth time. We also identify a sequence of *time delays* $\{\tau_i[k]\}_{k=1}^{\infty}$. The sum $r_i[k] + \tau_i[k]$ represents the time when the ith controller first outputs the control signal based on the kth sample. These sampled states are then used by the ith subsystem's controller so that the actual control for this system is

$$u_i(t) = k_i(x_i(t)) + \sum_{j \in \mathcal{N}_i} \ell_{ij}(x_j(r_i[k])) \tag{2}$$

where t is constrained to lie between two consecutive sampling times, $r_i[k] + \tau_i[k]$ and $r_i[k+1] + \tau_i[k+1]$.

The system formed by equations (1) and (2) constitutes this project's object of interest. It is a **hybrid dynamical system** for the control is a function of both continuous time processes and discrete-time processes. What we are interested in doing is characterizing the sample time sequences, $\{r_i[k]\}_{k=1}^{\infty}$, for each subsystem such that the *gain* of the entire networked system is less than the specified level of γ. We will characterize those sampling times using an extension of the analysis that [23] used in developing self-triggered controllers for single processor systems. This project will use that characterization to identify a set of quality-of-service (QoS) constraints on a sporadic firm/soft real-time environment such that satisfaction of these constraints guarantees the safe operation of the networked system. We intend to develop networking protocols for wireless multi-hop communication networks that satisfy these constraints. In short, this project's agenda is to find a path by which we can discard the hard periodic real-time task models that have dominated computer controlled systems. We believe the self-triggered controllers may provide this path.

3.0 Self-triggered Control Systems

This project's approach is based on a generalization of the *self-triggered* control analysis found in [23]. That earlier work characterized sampling times for a single processor system executing multiple control tasks. This section generalizes that analysis to networked control systems with multiple processors communicating over a packet switched network. The main result is a threshold condition (equation 7) that provides a basis for self-triggering the exchange of information between processors within the networked systems.

3.1 System Model

We can assume without great loss of generality that the time between successive sampling instants is small. It therefore makes sense to use a linearization of equation 1 around the network's equilibrium point to determine the sampling times. These linearized state equations are

$$\dot{x}_i(t) = A_i x_i(t) + B_{1i} u_i(t) + B_{2i} w_i(t) + \sum_{j \in \mathcal{N}_i} H_{ij} x_j(t)$$

where A_i, B_{1i}, B_{2i}, and H_{ij} are matrices of suitable dimension. We now form the network state vector, $x = \text{vect}(x_i)$ by stacking the state vectors of all subsystems. In a similar way we let $u = \text{vect}(u_i)$ and $w = \text{vect}(w_i)$ denote the control input and disturbance input vectors to the entire system. The ith subsystem receives "samples" of its neighbors' states.

To simplify the following narrative, we assume that the delay $\tau_i[k] = 0$ and simply focus on the sampling times $r_i[k]$. Under this simplification the control generated by the ith subsystem can be written as $u(t) = Lx(r_i[k])$. The function $x : \Re \to \Re^n$ (where $n = \sum_i n_i$) is therefore the state function of the entire networked system and it satisfies the linear differential equation,

$$\dot{x}(t) = Ax(t) + B_1 Lx(r_i[k]) + B_2 w(t) + Hx(t)$$

for $t \in [r_i[k], r_i[k+1])$ where $A = \text{diag}(A_i)$, $B_1 = \text{diag}(B_{1i})$, $B_2 = \text{diag}(B_{2i})$ are block diagonal matrices and H is a block matrix whose components are the matrices H_{ij}. The gain matrix L is a block matrix whose

ijth block , L_{ij} is a gain that **decouples** subsystems i and j. Note that $L_{ij} = 0$ for $i \notin \mathcal{N}_i$, so the actual control $u_i(t)$ only depends on the states of those subsystems that are in $\mathcal{N}_i$.

3.2 Sample Time Selection

We will now impose three *assumptions* on the system under consideration to help identify the sampling instants for which the closed-loop gain is less than γ. These assumptions are itemized below.

Assumption 1: *The decoupled systems have an induced gain less than* γ.
A sufficient condition for this assumption [14] is that there exist a symmetric positive definite matrix, P_i, such that

$$A_i^T P_i + P_i A_i + \frac{1}{\gamma^2} P_i B_{2i} B_{2i}^T P_i + I \leq 0 \tag{3}$$

for $i = 1, \ldots, N$. This is an algebraic Riccati inequality in which γ is the desired performance level for the decoupled system. This inequality always has a solution for some positive γ.

Assumption 2: *The interconnection strength between subsystems is "weak".*
A sufficient condition for this assumption is that there exist symmetric matrices Q_i such that

$$PH + H^T P \leq Q = \text{diag}(Q_i) \tag{4}$$

where $P = \text{diag}(P_i)$. This condition will not hold for all interconnected system. It essentially represents a limit on the interconnection strength between the subsystems and is identical to related assumptions used by [13] in formulating distributed receding horizon control strategies.

Assumption 3: *There exists a suitable set of "stabilizing" decoupling gains,* L.
This assumption can be enforced by requiring that there exist decoupling gain matrices, L, such that

$$PH + H^T P + PB_2 L + L^T B_2^T P \leq \alpha I \tag{5}$$

for some $\alpha \in (0, 1)$. Given assumption 2, this condition is relatively easy to enforce. This assumption can be more difficult to enforce if the subsystem only has indirect access to its neighbors' state vectors.

By using Schur complements, the inequalities in equations 3-5 can be recast as linear matrix inequalities (LMI). We can then use well known LMI numerical algorithms to determine matrices P_i, Q_i, and L.

With these three assumptions we can readily find a condition characterizing sampling times, $\{r_i[k]\}$, that ensure the networked system's gain is less than $\gamma/\sqrt{1-\alpha}$. In particular, let's define the new variable $z_i = \sum_{j \in \mathcal{N}_i} L_{ij} x_j(r_i[k])$ where $r_i[k]$ is the kth sampling instant for subsystem i's neighbors. We may then rewrite the network state equations as $\dot{x} = Ax + B_1 w + B_2 z + Hx$ where $z = \text{vect}(z_i)$. Consider the function, $V : \Re^n \to \Re$ such that $V(x) = x^T P x$ where $P = \text{diag}(P_i)$. We treat V as a candidate Lyapunov function for the networked system. V becomes a Lyapunov function if its directional derivative is negative definite. Using the preceding assumptions and a completing the square argument, we can readily show that the directional derivative for V satisfies the inequality

$$\dot{V}(x) \leq -\|x\|^2 + \gamma^2 \|w\|^2 + x^T(PH + H^T P)x + x^T P B_2 z + z^T B_2^T P x \leq \overline{x}^T X \overline{x} + \gamma^2 \|w\|^2$$

where $\overline{x}^T = \begin{bmatrix} x^T & z^T \end{bmatrix}$ and $X = \begin{bmatrix} -I + Q & PB_2 \\ B_2^T P & 0 \end{bmatrix}$. So if we can ensure that for all $t \in [r_i[k], r_i[k+1])$ that

$$\overline{x}^T X \overline{x} \leq -(1-\alpha)\|x\|^2 \tag{6}$$

for some $0 < \alpha < 1$, then we can conclude that $\dot{V}(x) \leq -(1-\alpha)\|x\|^2 + \gamma^2\|w\|^2$, which from passivity arguments imply that the networked system's induced gain is less than $\gamma/\sqrt{1-\alpha}$.

Note that the matrix X consists of four block diagonal matrices. This means that the condition in equation 6 can be enforced by requiring the following inequality

$$\begin{bmatrix} x_i^T(t) & z_i^T \end{bmatrix} \begin{bmatrix} -I_i + Q_i & P_i B_{2i} \\ B_{2i}^T P_i & 0 \end{bmatrix} \begin{bmatrix} x_i(t) \\ z_i \end{bmatrix} \leq -(1-\alpha)\|x_i(t)\|^2 \tag{7}$$

hold for all $i = 1, \ldots, N$. (7) is a function of the sampled neighboring state vectors, z_i, as well as the subsystem state, x_i. Over the time interval $[r_i[k], r_i[k+1])$, the sampled states, z_i, are held constant while x_i is a function of time. The inequality in (7) can therefore be checked "locally" by the ith subsystem without any additional information from the neighboring subsystems.

The analysis leading to (7) can be seen as generalizing the analysis in Lemmon et al [23] to real-time networked systems. This analysis method makes use of candidate Lyapunov functions to characterize a system's local behavior. This is a common technique that has been used to approximate sampling intervals in nonlinear sampled-data systems. Zheng et al [54], for example, used it to estimate sampling times for a class of nonlinear sampled data systems. Nesic et al. [38] used this approach within the context of input-to-state stability (ISS) to bound intersample behavior. Tabuada et al [49] built upon the ISS analysis method to estimate sampling periods for a class of nonlinear event-triggered control systems.

3.3 Benefits of Self-triggered Control

The most significant consequence of (7) is that the ith subsystem can use it as a local threshold test for deciding when to request updates of its neighbors' state vectors. This threshold test can be viewed as the basis for either an *event-triggered* or a *self-triggered* feedback controller. Event-triggered controllers were proposed by [2]. Variations on the approach have appeared under a variety of similar names such as interrupt-based feedback [16], Lebesgue sampling [3], asynchronous sampling [52], or state-triggered sampling [49]. All of these approaches apply a threshold test on some signal within the control loop to trigger sampling of either the plant output or state. Our inequality in (7) can simply be treated as another possible threshold test for event-triggered feedback.

Except for relay systems [50] or pulse width modulated feedback [39], event-triggered feedback can be impractical since it requires integrating an analog event detector into the physical plant. A more pragmatic approach is found in the **self-triggered** task model of [51]. In self-triggered systems, the control task determines its next sampling time based on the most recent sampled state. Self-triggered task models, therefore, can be implemented in existing computer controlled system without the need for any special analog event-detectors.

In *self-triggered feedback*, the next "sampling" instant is determined using our prior knowledge of the system dynamics. For our application, the ith subsystem knows its own dynamical models and assumption 2 represents an upper bound on the rate of growth in the neighboring systems' states. We can therefore use this information to numerically integrate forward in time to determine when the threshold test (equation 7) is first violated. Once this "time" is known, the ith subsystem can broadcast this time to its neighbors. The neighbors would then start interval timers that expire at this predicted sampling time. Upon expiration of this timer, all of the neighbors then sample and transmit their state information back to the ith subsystem. Self-triggered feedback has a number of benefits that we discuss below.

Note that this particular approach to trigger sampling does not require "global" time synchronization over the entire network. The sampling "requests" are issued in an asynchronous manner. If these requests are passed over multi-hop wireless (radio) channels, like those found in embedded sensor networks, the broadcast request can be used to "locally" synchronize the clocks of all neighbors so that samples are gathered at the same time. This is an important benefit of our method, since the message overhead required to maintain precise global clock synchronization can be prohibitive in any large-scale communication network.

Prior work by [23] suggests that self-triggered controllers have robust performance to timing delays. This prior work showed that self-triggering was able to maintain high-levels of application performance even in the face of significant processor overloading. The system's robust performance is a direct consequence of the inherent feedback nature of the self-triggered control. Predicted sampling periods are strongly dependent on the state of the system when it was last sampled. This means that if there were some timing delay, we can compensate for that disturbance by shortening the time until the next sample is taken. As we'll discuss below, such timing delays occur often in best effort communication networks such as the Internet or wireless multi-hop networks. This observation suggests that by using self-triggered decentralized controllers, we can still guarantee the performance of the overall system even when messages are passed over such networks.

Self-triggered control systems usually generate *sporadic* streams of messages between neighboring systems. This occurs because the time between the samples is determined by how quickly the state vector violates the condition in equation 7. That rate, of course, is a function of the system state at the prior sampling time, so we can expect these sampling times to vary over a wide range. This fact was experimentally verified by [23] in which histograms of the sampling times clearly showed a probabilistic distribution with a wide variance. Note that earlier sample time approximations [54] and [49] can be seen as lower bounds on the distribution of sampling times satisfying equation 7. So our approach can often reduce the average sampling rate by using longer sampling periods.

We can further reduce the sampling rate by adaptively adjusting the performance level γ required by the system. Recall that control system performance is measured by the "gain" from the disturbance to the state output. This gain is really a measure of the maximum amount of disturbance rejection for the "worst-case" disturbance. In real-life, such worst-case disturbances may not always be present. So if our system can detect when the disturbance level is below this "worst-case", it would make sense to "relax" the performance requirement by raising γ. Relaxing the performance requirement will change the sample time distribution in a way that increases the average sampling period, thereby further reducing the sampling rate. In other words, if our subsystems can monitor the average "disturbance" level, then they can adaptively adjust their performance objectives in response to observed changes in the disturbance environment, thereby providing greater control over the flow of information across the communication network.

The fact that self-triggered controllers require lower bit rates than traditional open-loop time-triggered controllers is important in developing scalable strategies for control over real-time communication networks. It is already well known that multi-hop wireless networks have inherent throughput limitations [15]. These are fundamental limitations that can only be circumvented if we reduce the amount of "work" that the network has to do in transporting data. This is done by either reducing the bit rate, reducing the distance over which messages are transmitted, or doing both [44]. Our application does both. Self-triggered control strategies appear to reduce the bit rate required to assure a specified level of control system performance. But in addition to this, the decentralized nature of the control guarantees that messages are only transported over a single hop, thereby reducing the distance over which network traffic must travel.

3.4 Proposed Work - Decentralized Controllers

The self-triggered approach in this section generalizes results in a preliminary paper [23] that is just about to appear. The generalization in Section 3.3 has not been previously presented by our group. There are a number of open issues that we plan to investigate in this project. In particular, we will study the proposed self-triggered control architecture in detail.

We need to do additional work to determine the impact that delays and dropped feedback might have on control system performance. It may be the case that "certain" neighboring measurements should be assigned a higher priority. This assignment of priority should be based on a solid understanding of the system dynamics. How one should assign such priorities must be determined if we are to be successful in developing the real-time scheduling approaches (to be discussed in Section 4).

We will aso explore the extent to which the three assumptions given above can be relaxed. Of particular interest is how best to relax the decoupling assumption in equation (4). The analysis given above relied on a linearization of the underlying dynamical systems. There is good reason to believe, however, that the $\mathcal{L}_2$ analysis methods we're using can also be extended to certain classes of nonlinear dynamical systems as well. Finally, we need to extend our framework to deal with output feedback as opposed to pure state feedback.

4.0 Real-time Scheduling over Networks of Self-Triggered Controllers

Implementing self-triggered control systems via a packet-switched communication network faces a number of challenges. For example, what is the best way of "scheduling" predictable transmissions in order to prevent message collisions while satisfying some real-time guarantees on message delivery, and how do we deal with packet losses due to network congestion? Furthermore, we recognize that deriving the next sampling periods is computationally non-trivial and it can be unrealistic to expect the sampling periods to be re-evaluated during every sampling period. A natural question is how often the sample periods should be updated and what to do in between these updates. Analyzing the performance of the resulting networked dynamical system is also of critical importance in quantifying the robustness of such a system. This part of the project will address these questions by investigating scheduling and admission control strategies in a real-time network environment.

4.1 Real-time Networking Model

Let's first describe the messaging environment for decentralized control over ad hoc wireless networks. Such communication networks occur frequently in a variety of embedded sensor network applications. The networked dynamical system consists of N systems that communicate over a wireless network. Connectivity within the communication network is modeled as a graph, $(\mathcal{N}, \mathcal{A})$ where $\mathcal{N} = \{1, \cdots, N\}$ is a discrete set of nodes (systems) and $\mathcal{A} \subset \mathcal{N} \times \mathcal{N}$ is a set of arcs (communication channels) between subsystems. The pair (i, j) is an arc in $\mathcal{A}$ if subsystems i and j can exchange messages. We associate a **message stream** (analogous to *tasks* used in traditional textbooks on real-time systems [37, 21]) with each channel (arc) in the communication graph. A message stream consists of **messages** (analogous to jobs of a real-time task) that are **released** for transmission at a release time. We may therefore associate a sequence of **release times**, $\{r_{ij}[k]\}_{k=1}^{\infty}$ to the (i, j)th message stream. If the release times satisfy $r_{ij}[k] = r_{ij}[k-1] + T_i$ for some constant T_i and for all $k > 0$, then we say that the system is **periodic**. Otherwise, we say that the system is **sporadic**.

A released message may be **transmitted** at some time after its release. In multi-hop radio networks, we can assume that a transmitted message is **caught** by the recipient's radio if the radio can successfully recover all of the bits in the message packet. Transmitted messages cannot always be caught and even if they are caught, they may not be **delivered** to the application. If the radio detects an error in the packet or if the "time-stamp" is too old, then the radio may elect to "drop" the packet. In particular, let D_{ij} denote the **relative deadline** on message stream (i, j) and let $f_{ij}[k]$ denote the time when the kth message of this stream is *caught*. We'll assume that the radio **delivers** the message to the application if $D_{ij} \geq f_{ij}[k] - r_{ij}[k]$. If we can guarantee that this occurs for all k, then we say the messages are delivered with a **hard** real-time guarantee. In practice, however, we know messages may be corrupted or delayed in transmission, so it is more reasonable to assume that $\Pr\left((f_{ij}[k] - r_{ij}[k]) \leq D_{ij}\right) < 1$ so that there is a finite probability of a message missing its deadline. We say message delivery is **soft** if such late messages are still delivered to the application. We say message delivery is **firm** if these late messages are dropped.

For the decentralized self-triggered control systems discussed in Section 3, we argue that a natural real-time abstraction presumes that messages are transmitted in a **sporadic** manner with **firm** or **soft** real-time guarantees on their delivery. Provided that this environment satisfies certain quality-of-service (QoS) constraints, the decentralized controller should be able to guarantee specified levels of application performance. So the chief question addressed in this part of the project concerns the precise form of these QoS constraints and how to schedule message transmission to ensure these constraints are satisfied.

The reliable transmission of released messages requires a number of "services" that are usually encapsulated into layers that form a *network stack*. We consider, for the purposes of this proposal, a simplified version of the traditional OSI stack that consists of only three layers: (i) the top layer called the *application layer* consisting of services that control the application's (i.e. controller) access to the network, (ii) the middle layer called the *network layer* appending routing headers to the application layer's data packet, and (iii) the lowest layer, i.e., the *radio layer*, which is responsible for the transmission of packets over the physical channel.

4.2 Related Work and Unique Challenges

Radio channels must be accessed in a mutually exclusive manner. This means that node i's message is caught by the jth node if no neighbor of node i or j (other than i) is also transmitting at the same time. If this condition is not satisfied, messages from two transmitters collide and cannot be successfully decoded by the radio layer. The radio layer can reduce the likelihood of packet collisions through carrier-sense media access (CSMA) protocols. CSMA schemes detect if the channel is "busy" and then back off the transmission of a packet by a random amount. CSMA is very important in helping reduce the frequency of message collisions, but it is not fool proof. Collisions are still possible if two transmitters that are "hidden" from each other decide to transmit at the same time to a receiver that lies within the range of both transmitters. This is sometimes referred to as the *hidden node* problem and its solution requires a more elaborate handshaking mechanism between the transmitter and receiver to help prevent such collisions [1]. These handshaking mechanisms are usually implemented in mid-level layers of the network stack. The MACAW protocol is a well-known example of such a handshaking mechanism [5].

CSMA/MACAW can help reduce the frequency of message collisions in multi-hop radio networks. These access protocols were designed for networks in which few assumptions were placed on users' requests for media access. In self-triggered control systems, however, user requests can be predicted based on our knowledge of the control application. This fact can be exploited to **schedule** transmissions in a way that further reduces channel collisions and hence maximizes channel bandwidth.

Scheduling message transmission and delivery to meet real-time constraints is a non-trivial task particularly for wireless networks. Prior work usually relied on fully synchronized nodes coupled with time-division multiple access (TDMA) schemes or presume a priori knowledge of packet arrival patterns [10] [27] [40]. Such approaches lack the flexibility to respond to dynamic changes in the networking environment and are not suitable for our self-triggered control system.

A number of papers have examined the problem of distributed scheduling and resource allocation. These papers can be categorized in two groups. One group exploits the broadcast nature of the wireless medium and piggybacks scheduling related information (such as priority) so as to obtain global information about the network state [6] [19]. The disadvantage of this approach is that extra resource overhead makes it less scalable. The other group [53] totally forgoes global information and simply monitors packet delay information to adjust the contention window, thereby adjusting the priority of the packets.

Our proposed approach adopts the distributed scheduling concept. The unique properties of self-triggered dynamical systems make the basic scheduling procedure easier to implement but in the meantime introduce some interesting problems. We believe that if done properly, the combination of self-triggered model together with distributed scheduling can significantly improve system robustness in a cost effective manner.

Let us examine the precise way in which control messages are passed in our system. Self-triggered controllers "predict" the next time they must receive sampled updates of their neighbors' states. This means that a node can "request" a future update from its neighbors. This request would be transmitted as a short control message. Upon receipt of that message, the neighboring nodes would start an interval timer. Upon expiration of that timer the neighbors would sample and transmit their states back to the requesting node. There are two scenarios we need to consider and they are shown in figure 1.

Scenario 1: In the first scenario shown on the lefthand side of figure 1, we have a single node requesting future samples from its neighbors. The neighbors would all want to release their messages at the same time. So the transmission of such messages must be scheduled in a way to avoid collisions (as shown in the figure). We refer to this scenario as **receiver scheduling**.

Scenario 2: The second scenario is shown on the righthand side of the figure 1. In this case we have a single neighbor that receives two requests. The two requests may or may not have the same sampling time and deadline. Scheduling may be needed in order to satisfy the deadline constraints. We refer to this scenario as **sender scheduling**.

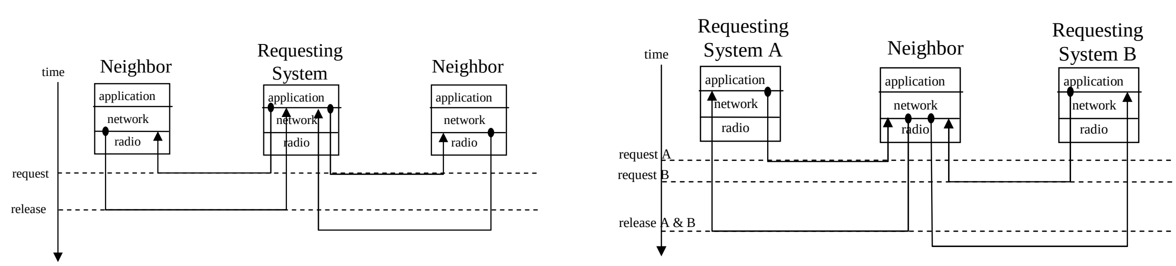

Figure 1: LEFT: neighbors need to schedule message transmission to prevent collision at the requesting system. RIGHT: neighbor needs to schedule message transmissions to multiple requesting subsystems

Traditional real-time scheduling employs priorities to order task execution. The most often used priority is task deadline, which leads to the well known earliest-deadline first (EDF) scheduling discipline [34]. With the self-triggered networked control systems, the message deadlines can be selected by the requesting node and readily made known to the neighbors when sampling requests are sent out. As discussed in Section 3.3, since each node only requires information from its neighboring nodes, no global synchronization is needed. Therefore, our self-triggered controllers adopt the EDF scheduling discipline locally at each node such that messages with the earliest deadlines are transmitted first. How to select message deadlines to ensure performance is one of the proposed task as discussed in Section 3.4.

To achieve robustness in the self-triggered dynamical system, we face additional unique problems. First, in the receiver scheduling scenario, we often have messages with the same deadline (left hand picture in figure 1). Should we break the ties arbitrarily as done in most real-time scheduling? Second, if a sender must transmit multiple messages with different deadlines, what is to be done if not all deadlines can be satisfied? Third, due to channel conditions, some messages may be lost. What is the best way to maintain overall system performance? We will discuss below the approaches that we will explore for solving these problems.

4.3 Proposed Work – Real-time Network Scheduling

In handling the receiver scheduling scenario, we believe that arbitrary tie breaking is not desirable since the underlying physical system under control may respond to the lateness of different neighbors differently. We propose that the receiver (i.e., the requesting node) assigns priorities to the neighbors' replies. Priority assignment will be based on how "important" that neighbors' reply is in enforcing the application's performance specification.

Recall that the decoupling feedback gain, ℓ_{ij}, in equation (2) is chosen to help "decouple" neighboring subsystems. Clearly, if the neighboring state is small, its contribution to decoupling will be small and so it should have a lower priority. The requesting node can use its a priori knowledge of its neighbors' dynamics to predict which states should be "large" at the next release time. So the natural thing to do is have the requesting nodes broadcast these "priority" levels when it first broadcasts the requested next release time. The priority may be in the form of "deadlines" such that higher priority ones get a shorter deadline. The challenge here is how to "decrease" or "increase" the deadlines such that they are meanful to both the receiver and sender nodes. We will investigate the deadline choices in detail in the project.

When attempting to address the problem of missing deadlines by the senders, we note that the sender node basically schedules periodic messages between sampling period updates. Well known schedulability conditions can be used to determine whether all deadlines (which are often much smaller than periods) can be satisfied [4]. If the messages cannot be feasibly scheduled, EDF scheduling can lead to unpredictable performance degradation. To overcome this problem, we can either selectively drop some messages or allow all messages to be sent but alter the periods of certain messages. The goal of both of these approaches is to achieve predictable performance degradation.

Suppose we could somehow determine the relative importance of the messages to be sent. We may simply drop those messages of less importance. It is not difficult to see that continuously dropping messages intended for the same receiver would cause the physical system at that receiver to suffer severe performance degradation. Therefore, some mechanism is needed to dynamically adjust message importance. The impact of dropping samples on control performance needs to be studied.

An alternative to dropping messages is to modify message periods to reduce bandwidth requirement. This approach is attractive as it somewhat "resembles" the self-triggered control that we intend to adopt, but could be made much less computationally demanding. We propose to leverage our experience on *elastic scheduling* to examine this approach in greater detail. The elastic scheduling mechanism was first proposed by Buttazzo [7] for single processor real-time systems. Buttazzo's elastic task model uses a mechanical analogy to develop an algorithm for adjusting task periods. The task model was extended by Caccamo et al. [9] to handle uncertainties in computation time. A later paper [8] showed how to modify the original algorithm to handle additional resource constraints.

A critical factor in elastic scheduling is the task's elastic coefficient which models a task's relative importance. In essence, the algorithm tries to extend the periods of less important tasks in order to satisfy the deadline requirement of the resulting tasks. In doing so, some optimization objective is met. Our recent work [17] revealed that Buttazzo's elastic scheduling algorithm can be viewed as minimizing a task set's summed squared utilization subject to the EDF schedulability condition [34]. Furthermore, we have developed a elastic scheduling algorithm for task deadlines less than periods [11]. Since the elastic scheduling algorithm (with $O(n^2)$ complexity) in fact solves a specific convex programming problem, we feel that it can be used in our proposed work to help achieve quantifiable performance degradation. We will investigate different objective functions to find the proper one for the networked control system.

From the above discussions, both message dropping and elastic scheduling have a common concern regarding the way in which relative importance between messages might be chosen. The relative importance (or elastic coefficients) reflects the "relative" priority of the two requesting nodes in the sender scheduling scenario. One way of selecting these weights might be through a "learning" process in which we use the time rate of change in requested message release times as a measure of how "dissatisfied" the requesting agent is with its given priority level. The scheduling agent would then use this "dissatisfaction" to set a "price" for earlier transmission in much the same way that Kelly et al. [20] proposed using shadow price formalisms for rate control in multi-hop networks. We propose using the game theoretic ideas inherent in the shadow price formalism to develop "learning" algorithms that schedulers use to automatically adjust the elastic coefficients in Buttazzo type scheduling algorithms.

Addressing the message loss problem may also leverage the elastic scheduling approach. For example, when a receiver does not receive its desired messages, it could "raise" the importance of its message stream. The elastic scheduling algorithm would then re-adjust the message periods by allowing this message stream to be transmitted more often (with shorter periods) and thus increases the probability for successful message delivery. Note that the message loss problem could also be handled through the self-triggering mechanism by re-evaluating the sampling periods. In our project, we will study and compare these two different approaches in terms of tradeoff between the impact on system performance and the computational demands.

Precisely how well control systems work under self-triggering together with the above scheduling methods remains to be seen. The scheduling methods can only provide firm real-time guarantees. One goal of this project is to study the behavior of self-triggered control under the firm and soft real-time environments and determine suitable constraints whose satisfaction assures some level of application performance. The precise issue is how quickly control system performance degrades under dropped or delayed messages?

We have answered that question, in part, for firm real-time messaging environments. Our prior work studied a Markov chain generalization of the (m, k)-firm guarantee model [41]. In [30] we related this Markov chain model for dropped messages to the application's performance using jump linear systems. We then identified a Markov chain dropout process that "optimized" the application's performance for a given average dropout rate. That “optimal” dropout process was used as a quality-of-service (QoS) constraint (called an MC-constraint) on the firm real-time environment. In [36, 35, 18], we showed how to develop real-time schedulers enforcing this QoS constraint in a way that provided guarantees on the control system’s performance.

Prior work [25] examined how this approach might be applied to networked dynamical systems, so our work with MC-constraints provides a natural starting point for studying the impact that firm real-time environments might have on the performance of self-triggered decentralized controllers. This is, however, just one initial direction. We believe that similar methods might be used to investigate soft real-time messaging as well. The expectation is that these analyses will enable us to identify a suitable set of soft or firm real-time constraints that can assure some probabilistic measure of application performance (similar to what we did in [36]. We believe this is a realistic expectation, based on the apparent robustness of self-triggered control to delays [23].

5.0 Technology Transfer

We believe that it is important that this project have an impact that extends well beyond our respective research communities. One way of broadening this project’s impact is through technology transfer. This section discusses our technology transfer plans.

Dr. Lemmon is currently working with a local company named EmNet LLC. This company was created by a former Ph.D. student at the University of Notre Dame (Dr. Luis Montestruque). Its start up was capitalized by a grant that Dr. Lemmon had from the state of Indiana to develop an embedded sensor network for reducing the frequency of combined sewer overflow (CSO) events. The company’s main product line is the Chasqui node. This is an embedded sensor node based on U.C. Berkeley’s MICA2 sensor node that has been ruggedized for outdoor use. The Chasqui node uses a more powerful radio than the MICA2 node, so it can form ad hoc networks in which average link ranges are 2-4 kilometers. EmNet LLC provides researchers at the University of Notre Dame was a unique local ability to build custom sensor networks for a variety of real-life applications including CSO networks [42] and monitoring of lake environments [45]. This project will work with EmNet LLC to assess the effectiveness of self-triggered decentralized controllers in EmNet’s CSO network (CSOnet) infrastructure project.

EmNet LLC’s CSO network project was originally funded under a grant from the State of Indiana. The project consists of a partnership between the University of Notre Dame, EmNet LLC, and the City of South Bend that is building an embedded sensor-actuator network to monitor and control CSO events in the South Bend’s sewer system. CSO events represent a major environmental health hazard. These events occur in older sewer systems where sanitary flows are mixed with storm water flows. These older systems become overloaded during storm events and the excess water is dumped into the nearest river. Since this excess water contains raw sewage it is highly impacted with chemical and biological contaminants. Under the 1974 clean water act, the environmental protection agency (EPA) has begun fining local municipalities for each CSO event. These fines can be on the order of tens of millions of dollars and cities are scrambling for methods to reduce the frequency of these events. The most direct way to solve this problem is to rebuild the physical sewer infrastructure. This approach however is extremely expensive (100’s of millions of dollars) and highly disruptive of the community. The CSO network being built by EmNet LLC is using embedded sensor-actuator networks to provide an alternative lower cost solution to the problem. The embedded sensor-actuator network is used to monitor and control storm water flows in a way that takes advantage of excess unused capacity in the city’s existing sewer system.

EmNet LLC demonstrated a small part of CSOnet in the summer of 2005. That network controlled the storm water flows generated by a 3.2 mile by 1500 foot corridor in the city of South Bend. The system has been in operation for over a year and has prevented numerous CSO events from being generated by this corridor. Future plans are to expand this demonstration network to the entire city's interceptor sewer line, thereby providing a metropolitan-scale demonstration of the technology. This project's PI will be working with EmNet LLC under sponsorship from the State of Indiana to complete this demonstration. The working relationship between the PI and EmNet LLC provides an excellent opportunity for technology transfer.

The CSOnet technology provides an excellent platform for testing the use of self-triggered controllers in the real world. CSOnet is an embedded sensor-actuator network. Actuator nodes are located at CSO diversion points along the entire length of the city's interceptor sewer. These actuator nodes control the rate at which water enters the interceptor structure. Sensor nodes are placed in other parts of the sewer system feeding into the interceptor. These sensor nodes monitor current flow rates. All nodes communicate over a wireless multi-hop network. CSOnet, therefore, is a specific example of a networked dynamical system being studied by this project.

The PIs will work with EmNet LLC to help evaluate the suitability of this project's work to CSOnet. We think this might be done by first building a scale hardware model of the city's interceptor sewer in our lab, and then using this model to experimentally evaluate how best to transfer the project's algorithms and network protocols to CSOnet.

6.0 Curriculum Development

For the most part, the undergraduate control curriculum has changed little over the past fifty years. This is regrettable, for it has meant that undergraduate control classes have not always kept up with advances in computer and communication technology. The consequence of this neglect is that enrollment in undergraduate control classes has steadily dwindled over the past 20 years. But more importantly, it means that many graduating seniors are ill prepared to work with the control systems that they will inevitably encounter during their careers. If this trend is to be reversed, we must find ways of better engaging undergraduates so that our future engineering workforce has a good understanding of control theory as well as a working knowledge of the impact that modern communication and computer technologies have on control.

The basic research work being done under this project is directly concerned with the impact that networking and real-time systems technologies have on control. We believe it is time to develop a new curriculum for control that is integrated with real-time system concepts. As part of this project, both PI's intend to develop undergraduate courses on *real-time control systems*. This course will be targeted at junior/senior level students across engineering including both computer, electrical, and mechanical engineers. The course will cover the mathematical system models usually treated in traditional signal/systems and control courses. But we intend to integrate the systems theory material with a laboratory to provide students with a hands-on understanding of these abstract concepts. The laboratory will familiarize students with real-time computing systems and help them better understand the current directions that the technology is taking. We will make the class lecture notes and lab notes available to all instructors to broaden its impact beyond the University of Notre Dame.

7.0 Statement of Work

We see this project as consisting of four primary tasks based on the ideas discussed in Sections 3-6. These tasks are summarized below.

Task 1 – Self-Triggered Decentralized Controllers:
This task will study self-triggered decentralized controllers for networked dynamical systems. The analysis outlined in Section 3 represents this task's starting point. This analysis generalizes our earlier work [23] to networked dynamical systems. We intend to broaden our earlier work to output feedback control systems. A major objective of this analysis is to understand the impact that firm/soft real-time message delivery might have on the achievable performance level.

Task 2 – Real-Time Scheduling over Networks of Self-Triggered Controllers:
This task will study means to schedule messages for self-triggered controllers communicating over a packet-switched network in order to meet real-time constraints. The work will be built on our experiences on elastic scheduling, (m,k) and Markov Chain firm scheduling. We will explore game theory based learning mechanisms to dynamically tune the relative importance of message streams. The ultimate goal is to achieve robust control performance in a cost effective manner.

Task 3 – Technology Transfer:
This task will investigate the feasibility of integrating the self-triggered controllers into the CSOnet project described in Section 5. We will work with our technology partner, EmNet LLC, to build a hardware model demonstrating the benefits of the project's work. The expectation is that this hardware demonstration will lead to the transfer of the project's technology to EmNet LLC's CSOnet project.

Task 4 – Curriculum Development:
This task will develop a new curriculum for control engineering at the junior and senior undergraduate level. The task will develop lecture and lab modules that integrate real-time system concepts into traditional control theory curriculum to provide graduating seniors with a working knowledge of how real-time control systems are built in the real world.

8.0 Prior NSF Research

Dr. Lemmon received prior support under NSF grants

- **Scalable Decentralized Control over Ad Hoc Sensor Actuator Networks**
  ECS04-00479 - $210,000 - (2004-2007).

This grant studied networked control systems and the relationship between control system performance in the presence of dropouts and quantized feedback. It also studied coordination in multi-agent networked control systems. This work generated several papers. In [28] and [29] we developed a power spectral analysis for linear system performance in the presence of randomly dropped feedback packets. That work was later summarized in [31]. [30] and [25] used a Markov-chain model of the dropout process and jump linear systems to characterize the performance achievable under dropped data. [24] and [33] studied the stability of dynamically quantized feedback systems. Optimal bit assignment algorithms for noise-free quantized feedback [32] and noisy quantized feedback [26] were developed. We also studied the use of periodic communication logics for multi-agent systems [47]. We studied the interconnection of swarming dynamics and consensus filtering in [26] and [48].

Dr. Lemmon and Dr. Hu received NSF funding under grant

- **Flexible Scheduling in Real-Time Control Systems with Uncertainty**
  CNS-0410771 - $240,000 - (2004-2007).

This grant focused on the relationship between control systems and real-time computing systems. This second grant is most closely related to the current proposal. That earlier work generated a number of publications. This work developed the Markov chain QoS constraint for firm real-time systems. Initial ideas on this appeared in [18]. That work was later extended in [35] and a journal publication for the approach appeared in [36]. This sponsored work has also studied Buttazzo's elastic scheduling algorithm [7]. It was initially suggested in [17] that elastic scheduling could be viewed as an optimization problem. That idea was used in [11] to develop elastic scheduling algorithms for periodic task sets whose deadlines were less than their periods. The self-triggered controller approach was introduced in [23].

9.0 Intellectual Impacts

Computer controlled systems have been dominated by periodic hard real-time systems. This domination has made it difficult to develop decentralized controllers over packet-switched networks such as the Internet or ad hoc wireless networks. This project will develop self-triggered decentralized controllers (Section 3) and the associated real-time networking environments (Section 4) that will finally allow us to build control large-scale physical processes over such best-effort networks. In simple words, this project's chief intellectual impact lies in developing firm and soft real-time control systems that have performance levels comparable to traditional hard real-time systems.

10.0 Broader Impacts

The project will broaden the impact of the basic scientific research goals through technology transfer and curriculum development.

Technology transfer will be realized through the collaborative work described in Section 5 concerning CSOnet. If successful, this collaboration will lead to the deployment of self-triggered decentralized controllers in embedded sensor-actuator networks used to control the frequency of CSO events. The potential impact of this technology transfer is large. CSO problems effect over 800 cities across this nation. It is a problem that also effects many metropolitan areas across the world. Should the CSOnet collaboration succeed, there is the strong possibility that the technology will be adopted on a national level, thereby significantly extending the influence of this project on effectively managing our national infrastructure.

Curriculum development will be realized by developing a new course at the University of Notre Dame that integrates real-time system concepts into the traditional control theory curriculum. A more detailed description will be found in Section 6. This curriculum will be made available to students in computer, electrical, and mechanical engineering. It will consist of both lecture and lab components to provide students with a hands-on introduction to real-time control. The educational materials developed under the project will be made available to other educators in an attempt to broaden its reach beyond the University of Notre Dame.

References

[1] IEEE 802.11g-2003. In *http://www.iee802.org/11/*.

[2] K.E. Arzen. A simple event-based pid controller. In *Proceedings of the 14th IFAC World Congress*, 1999.

[3] K.J. Astrom and B.M. Bernhardsson. Comparison of riemann and lebesgue sampling for first order stochastic systems. In *Proceedings of the IEEE Conference on Decision and Control*, 1999.

[4] S.K. Baruah, L.I. Rosier, and R.R. Howell. Algorithms and complexity concerning the preemptive scheduling of periodic , real-time tasks on one processor. *Journal of Real-Time Systems*, 2:301–324, 1990.

[5] Vaduvur Bharghavan, Alan Demers, Scott Shenker, and Lixia Zhang. Macaw: a media access protocol for wireless lan's. In *SIGCOMM '94: Proceedings of the conference on Communications architectures, protocols and applications*, pages 212–225, 1994.

[6] R. Boder and C.G. Lee. Real-time guarantee of aperiodic packets in single-hope ad hoc wireless networks. In *IEEE Int'l Conf. On Embedded and Real-Time computing Systems and Applications*, pages 254–259, 2005.

[7] G. Buttazzo, G. Lipari, and L. Abeni. Elastic task model for adaptive rate control. In *IEEE Real-Time Systems Symposium (RTSS)*, 1998.

[8] G. Buttazzo, G. Lipari, M. Caccamo, and L. Abeni. Elastic scheduling for flexible workload management. *IEEE Transactions on Computers*, 51(3):289–302, 2002.

[9] M. Caccamo, G. Buttazzo, and L. Sha. Elastic feedback control. In *IEEE Euromicro Conference on Real-Time Systems (ECRTS)*, 2000.

[10] M. Caccamo, L.Y. Zhang, and G. Buttazzo. An implicit prioritized access protocol for wireless sensor networks. In *IEEE Real-Time Systems Symposium*, pages 39–48, 2002.

[11] T. Chantem, X. Hu, and M.D. Lemmon. Generalized elastic scheduling. In *IEEE Real Time Systems Symposium*, 2006.

[12] R. D'Andrea and G.E. Dullerud. Distributed control design for spatially interconnected systems. *IEEE Transactions on Automatic Control*, 48(9):1478–1495, 2003.

[13] W.B. Dunbar. A distributed receding horizon control algorithm for dynamically coupled nonlinear systems. In *IEEE Conference on Decision and Control*, 2005.

[14] M. Green and D.J.N. Limebeer. *Linear Robust Control*. Prentice-Hall, 1995.

[15] P. Gupta and P.R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, 2000.

[16] D. Hristu-Varsakelis and P.R. Kumar. Interrupt-based feedback control over a shared communication medium. In *Proceedings of the IEEE Conference on Decision and Control*, 2002.

[17] X. Hu, T. Chantem, and M.D. Lemmon. Optimal elastic scheduling. In *IEEE Real-time and Embedded Technology and Applications Symposium (works in progress track)*, 2006.

[18] X. Hu, D. Liu, M.D. Lemmon, and Q. Ling. Firm real-time system scheduling based on a novel qos constraint. In *Real Time Systems Symposium*, 2003.

[19] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly. Distributed multi-hop scheduling and medium access with delay and trhoughput constraints. In *ACM Mobile Computing and Networking*, pages 200–209, 2001.

[20] F.P. Kelly, A.K. Maulloo, and D.K.H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3):237–252, 1998.

[21] C.M. Krishna and K. Shin. *Real-Time Systems*. McGraw-Hill, 1997.

[22] C. Langbort, R.S. Chandra, and R. D'Andrea. Distributed control design for systems interconnected over an arbitrary graph. *IEEE Transactions on Automatic Control*, 49(9):1502–1519, 2004.

[23] M.D. Lemmon, T. Chantem, X. Hu, and M. Zyskowski. On self-triggered full information h-infinity controllers. In *Hybrid Systems: computation and control*, 2007.

[24] M.D. Lemmon and Q. Ling. Control system perforamnce under dynamic quantization: the scalar case. In *IEEE Conference on Decision and Control*, 2004.

[25] M.D. Lemmon, Q. Ling, and Y. Sun. Overload management in sensor-actuator networks used for spatially-distributed control systems. In *1st ACM Conferenced on Embedded Network Sensor Systems (SenSys03)*, 2003.

[26] M.D. Lemmon and R. Sun. Performance rate curves for dynamically quantized feedback systems. In *IEEE Conference on Decision and Control*, 2006.

[27] C.R. Lin and C.C. Liu. An on-demand QoS routing protocol for mobile ad hoc networks. In *IEEE Global Telecommunications Conf. (GLOBECOM)*, 2000.

[28] Q. Ling and M.D. Lemmon. Robust performance of soft real-time networked control systems with data dropouts. In *IEEE Conference on Decision and Control*, 2002.

[29] Q. Ling and M.D. Lemmon. Optimal dropout compensation in networked control systems. In *IEEE Conference on Decision and Control*, 2003.

[30] Q. Ling and M.D. Lemmon. Soft real-time scheduling of networked control systems with dropouts governed by a markov chain. In *American Control Conference*, 2003.

[31] Q. Ling and M.D. Lemmon. Power spectral analysis of networked control systems with data dropouts. *IEEE Transactions on Automatic Control*, 49(6):955–959, 2004.

[32] Q. Ling and M.D. Lemmon. Optimal dynamic bit assignment in noise-free quantized linear control systems. In *IEEE Conference on Decision and Control*, 2005.

[33] Q. Ling and M.D. Lemmon. Stability of quantized control systems under dynamic bit assignment. *IEEE Transactions on Automatic Control*, 50(5):734–740, 2005.

[34] C.L. Liu and J.W. Layland. Scheduling for multiprogramming in a hard-real-time environment. *Journal of the Association for Computing Machinery*, 20(1):46–61, 1973.

[35] D. Liu, X. Hu, M.D. Lemmon, and Q. Ling. Scheduling tasks with markov-chain constraints. In *17th Euromicro Conference on Real-time Systems*, 2005.

[36] D. Liu, X. Hu, M.D. Lemmon, and Q. Ling. Firm real-time system scheduling based on a novel qos constraint. *IEEE Transactions on Computers*, 55(3):320–333, 2006.

[37] J.W.S. Liu. *Real-Time Systems*. Prentice Hall, NJ, 2000.

[38] D. Nesic, A.R. Teel, and E.D. Sontag. Formulas relating kl stability estimates of discrete-time and sampled-data nonlinear systems. *Systems and Control Letters*, 38:49–60, 1999.

[39] E. Polak. Stability and graphical analysis of first order of pulse-width modulated sampled data regulator systems. *IRE Trans. Automatic Control*, AC-6(3):276–282, 1963.

[40] D. Pong and T. Moors. Call admission control for IEEE 802.11 contention access mechanism. In *IEEE Global Telecommunications Conf. (GLOBECOM)*, 2003.

[41] P. Ramanathan. Overload management in real-time applications using (m, k)-firm guarantee. *IEEE. Transactions on Parallel and Distributed Systems*, 10(6):549–599, 1999.

[42] T.P. Ruggaber and J.W. Talley. Detection and control of combined sewer overflow events using embedded sensor network technology. In *Proceedings of the World Water and Environmental Resources Congress*, 2005.

[43] N.R. Sandell, P. Varaiya, M. Athans, and M.G. Safonov. Survey of decentralized control methods of lrage scale systems. *IEEE Transactions on Automatic Control*, 23(2):108–128, 1978.

[44] A. Scaglione and S. Servetto. On the interdependence of routing and data compression in multi-hop sensor networks. *Wireless Networks*, 11:149–160, 2005.

[45] L.A. Seders, C.A. Shea, M.D. Lemmon, P.A. Maurice, and J.W. Talley. Lakenet: an integrated sensor network for environmental sensing in lakes. to appear in Environmental Engineering Science, 2007.

[46] D.D. Siljak. *Large Scale Dynamical Systems: stability and structure*. North-Holland, New York, 1978.

[47] Y. Sun and M.D. Lemmon. Periodic communication logics for the decentralized control of multi-agent systems. In *IEEE Conference on Control Applications*, 2005.

[48] Y. Sun and M.D. Lemmon. Swarming under perfect consensus using integral action. In *American Control Conference*, 2007.

[49] P. Tabuada and X. Wang. Preliminary results on state-triggered scheduling of stabilizing control tasks. In *IEEE Conference on Decision and Control*, 2006.

[50] Y.Z. Tsypkin. *Relay Control Systems*. Cambridge University Press, 1984.

[51] M. Velasco, P. Marti, and J.M. Fuertes. The self triggered task model for real-time control systems. In *Work-in-Progress Session of the 24th IEEE Real-Time Systems Symposium (RTSS03)*, 2003.

[52] P. Voulgaris. Control of asynchronous sampled data systems. *IEEE Transactions on Automatic Control*, 39(7):1451–1455, 1994.

[53] Y. Yang and R. Kravets. Distributed qos guarantees for realtime traffic in ad hoc networks. In *IEEE Conference on Sensor and Ad Hoc Communications and Networks*, pages 118–127, 2004.

[54] Y. Zheng, D.H. Owens, and S.A. Billings. Fast sampling and stability of nonlinear sampled-data systems: Part 2. sampling rate estimations. *IMA Journal of Mathematical Control and Information*, 7:13–33, 1990.