CODA: Congestion Detection and Avoidance in Sensor Networks

Chieh-Yih Wan
Dept. of Electrical Engineering
Columbia University
New York, NY 10027
wan@ee.columbia.edu

Shane B. Eisenman
Dept. of Electrical Engineering
Columbia University
New York, NY 10027
shane@ee.columbia.edu

Andrew T. Campbell
Dept. of Electrical Engineering
Columbia University
New York, NY 10027
campbell@ee.columbia.edu

ABSTRACT

Event-driven sensor networks operate under an idle or light load and then suddenly become active in response to a detected or monitored event. The transport of event impulses is likely to lead to varying degrees of congestion in the network depending on the sensing application. It is during these periods of event impulses that the likelihood of congestion is greatest and the information in transit of most importance to users. To address this challenge we propose an energy efficient congestion control scheme for sensor networks called *CODA (COngestion Detection and Avoidance)* that comprises three mechanisms: (i) receiver-based congestion detection; (ii) open-loop hop-by-hop backpressure; and (iii) closed-loop multi-source regulation. We present the detailed design, implementation, and evaluation of CODA using simulation and experimentation. We define two important performance metrics (i.e., energy tax and fidelity penalty) to evaluate the impact of CODA on the performance of sensing applications. We discuss the performance benefits and practical engineering challenges of implementing CODA in an experimental sensor network testbed based on Berkeley motes using CSMA. Simulation results indicate that CODA significantly improves the performance of data dissemination applications such as directed diffusion by mitigating hotspots, and reducing the energy tax with low fidelity penalty on sensing applications. We also demonstrate that CODA is capable of responding to a number of congestion scenarios that we believe will be prevalent as the deployment of these networks accelerates.

Categories and Subject Descriptors

C.2.1 [**Computer-Communications Networks**]: Network Protocols, Wireless Communications.

General Terms

Algorithms, Design, Performance.

Keywords

Energy efficient congestion control, wireless sensor networks.

1. INTRODUCTION

Sensor networks come in a wide variety of forms, covering different geographical areas, being sparsely or densely deployed, using devices with a variety of energy constraints, and implementing an assortment of sensing applications. One application driving the development of sensor networks is the reporting of conditions within a region where the environment abruptly changes due to an observed event, such as in habitat monitoring, target detection, earthquakes, floods, or fires. Sensor networks typically operate under light load and then suddenly become active in response to a detected or monitored event. Depending on the application this can result in the generation of large, sudden, and correlated impulses of data that must be delivered to a small number of sinks without significantly disrupting the performance (i.e., fidelity) of the sensing application. Although a sensor network may spend only a small fraction of time dealing with impulses, it is during this time that the information it delivers is of greatest importance.

The transport of event impulses is likely to lead to varying degrees of congestion in sensor networks. In order to illustrate the congestion problem consider the following simple but realistic simulation scenario. Figure 1 shows the impact of congestion on data dissemination in a sensor network for a moderate number of active sources with varying reporting rates. The ns-2 simulation results are for the well-known directed diffusion scheme [6] operating in a moderately sized 30-node sensor network using a 2 Mbps IEEE 802.11 MAC with 6 active sources and 3 sinks. The 6 sources are randomly selected among the 30 nodes in the network and the 3 sinks are uniformly scattered across the sensor field. Each source generates data event packets at a common fixed rate while the sinks subscribe (i.e., broadcast corresponding interest) to different sources at random times within the first 20 seconds of the 50-second simulation scenario. Event and interest packets are 64 and 36 bytes in size, respectively. The plot illustrates that as the source rate increases beyond a certain network capacity threshold (10 events/s in this network), congestion occurs more frequently and the total number of packets dropped per received data packet at the sink increases rapidly. The plot shows that even with low to moderate source event rates there is a large drop rate observed across the sensor network. For example, with a source event rate of 20 events/s in the network one packet is

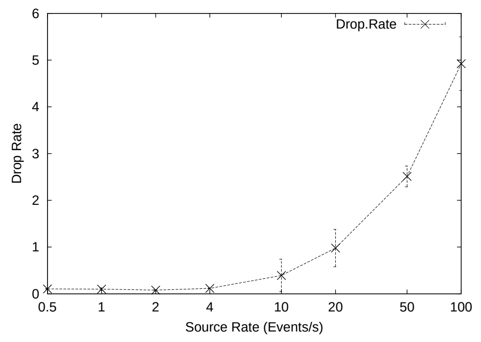


Figure 1: Total number of packets dropped by the sensor network per data event packet delivered at the sink (Drop Rate) as a function of the source rate. The x axis is plotted in log scale to highlight data points with low reporting rates. All packets that are dropped during the 50 second simulation session are counted as part of the drop rate including the MAC signaling (e.g., RTS/CTS/ACK and ARP), data event, and diffusion messaging packets.

dropped across the sensor field for every data event packet received at the sink. Dropped packets can include MAC signaling, data event packets themselves, and the diffusion messaging packets. The drop rates shown in Figure 1 represent not only significant packet losses in the sensor network, but more importantly, the energy wasted by the sensing application.

Depending on the type of sensing application the rate of event impulses may be occasional or more frequent. Some applications may only generate light traffic from small regions of the sensor network (e.g., target detection) while others (e.g., fires, earthquakes) may generate large waves of impulses potentially across the whole sensing area (which causes high loss, as shown in Figure 1).

In response to this, future congestion control mechanisms for sensor networks must be capable of balancing the offered load, while attempting to maintain acceptable fidelity (i.e., rate of events) of the delivered signal at the sink during periods of transient and more persistent congestion. A number of distinct congestion scenarios are likely to arise. First, densely deployed sensors generating impulse data events will create persistent hotspots proportional to the impulse rate beginning at a location very close to the sources (e.g., within one or two hops). In this scenario, localized, fast time scale mechanisms capable of providing backpressure from the points of congestion back to the sources would be effective. Second, sparsely deployed sensors generating low data rate events will create transient hotspots potentially anywhere in the sensor field but likely farther from the sources, toward the sink. In this case, fast time scale resolution of localized hotspots using a combination of localized backpressure (between nodes identified in a hotspot region) and packet dropping techniques would be more effective. Because of the transient nature of congestion, source nodes may not be involved in the backpressure. Third, sparsely deployed sensors generating high data-rate events will create both transient and persistent hotspots distributed throughout the sensor field. In this final scenario, a combination of fast time scale actions to resolve localized transient hotspots, and closed loop rate regulation of all sources that contribute toward creating persistent hotspots would be needed.

In this paper, we propose an energy efficient congestion control scheme for sensor networks called *CODA (COngestion Detection and Avoidance)* that comprises three mechanisms:

- *Congestion detection.* Accurate and efficient congestion detection plays an important role in congestion control of wireless networks. CODA uses a combination of the present and past channel loading conditions, and the current buffer occupancy, to infer accurate detection of congestion at each receiver with low cost. Sensor networks must know the state of the channel, since the transmission medium is shared and may be congested with traffic between other devices in the neighborhood. Listening to the channel to measure local loading incurs high energy costs if performed all the time. Therefore, CODA uses a sampling scheme that activates local channel monitoring at the appropriate time to minimize cost while forming an accurate estimate. Once congestion is detected, nodes signal their upstream neighbors via a backpressure mechanism.

- *Open-loop, hop-by-hop backpressure.* In CODA a node broadcasts backpressure messages as long as it detects congestion. Backpressure signals are propagated upstream toward the source. In the case of impulse data events in dense networks it is very likely that backpressure will propagate directly to the sources. Nodes that receive backpressure signals can throttle their sending rates or drop packets based on the local congestion policy (e.g., packet drop, AIMD, etc.). When an upstream node (toward the source) receives a backpressure message it decides whether or not to further propagate the backpressure upstream, based on its own local network conditions.

- *Closed-loop, multi-source regulation.* In CODA, closed-loop regulation operates over a slower time scale and is capable of asserting congestion control over multiple sources from a single sink in the event of persistent congestion. When the source event rate is less than some fraction of the maximum theoretical throughput of the channel, the source regulates itself. When this value is exceeded, however, a source is more likely to contribute to congestion and therefore closed-loop congestion control is triggered. The source only enters sink regulation if this threshold is exceeded. At this point a source requires constant, slow time-scale feedback (e.g., ACK) from the sink to maintain its rate. The reception of ACKs at sources serve as a self-clocking mechanism allowing sources to maintain their current event rates. In contrast, failure to receive ACKs forces a source to reduce its own rate.

The paper is organized as follows. Section 2 presents the related work. Following this, Section 3 discusses a number of important design considerations for mitigating hotspots in sensor networks including MAC and congestion detection issues. Section 4 details CODA's backpressure and rate regulation mechanisms. Following this, an implementation of CODA is evaluated in an experimental sensor testbed in Section 5. We define two performance metrics (i.e., energy tax and fidelity penalty) to evaluate the impact of CODA on the performance of sensing applications. Because CODA is designed to interwork with existing data dissemination schemes, we also evaluate it using one well-known dissemination mechanism. Section 6 presents a preliminary performance evaluation of CODA working with directed diffusion [6] using the ns-2 simulator. Finally, some concluding remarks and future work are presented in Section 7.

2. RELATED WORK

Congestion control in sensor networks has not been discussed to any great extent in the literature. The need for congestion avoidance techniques is identified in [12] while discussing the infrastructure tradeoffs for sensor networks. Tilak, Abu-Ghazaleh, and Heinzelman [12] show the impact of increasing the density and reporting rate on the performance of the network. While the authors do not propose any congestion avoidance mechanisms, they do note that any such mechanism must converge on a reporting rate that is just sufficient to meet the performance or fidelity of the sensing application. This is an important observation in the context of sensor networks.

Some existing data dissemination schemes [6] [13] can be configured or modified to be responsive to congestion. For example, directed diffusion [6] can use in-network data reduction techniques such as aggressive aggregation when congestion is detected. Other protocols, such as PSFQ (Pump Slowly Fetch Quickly [13], a reliable transport protocol for sensor networks) can adapt the protocol (i.e., modulate its pump/fetch ratio) to avoid congestion. However, such approaches involve highly specialized parameter tuning, accurate timing configuration, and in-depth understanding of the protocol's internal operations. There is a need for a comprehensive set of congestion control mechanisms specifically designed to best fit the unique constraints and requirements of sensor networks and their emerging applications. These mechanisms should provide a general set of components that can be plugged into applications or the MAC in support of energy efficient congestion control.

In [14] a comprehensive study of carrier sensing mechanisms for sensor networks is reported. The authors propose an adaptive rate control mechanism that supports fair bandwidth allocation for all nodes in the network. Implicit loss (i.e., failed attempts to inject a packet into the network) is used as a collision signal to adjust the transmission rate of nodes. The paper focuses on fairness issues in access control but not congestion control.

In [9] an event-to-sink reliable transport protocol (ESRT) provides support for congestion control. ESRT regulates the reporting rate of sensors in response to congestion detected in the network. This paper represents the most thorough work on congestion control in sensor networks to date and is inspired, as our work is, by the observations of Tilak, Abu-Ghazaleh, and Heinzelman [12] discussed above. ESRT monitors the local buffer level of sensor nodes and sets a congestion notification bit in the packets it forwards to sinks if the buffer overflows. If a sink receives a packet with the congestion notification bit set it infers congestion and broadcasts a control signal informing all source nodes to reduce their common reporting frequency according to some function. As discussed in [9] the sink must broadcast this control signal at high energy so that all sources can hear it. Such a signal has a number of potential drawbacks, however, particularly in large sensor networks. Any on-going event transmission would be disrupted by such a high powered congestion signal to sources. In addition, rate regulating all sources in the manner proposed in [9] is fine for homogeneous applications where all sensors in the network have the same reporting rate but not for heterogeneous sources. Even with homogeneous sources, ESRT always regulates all sources regardless of where the hotspot occurs in the sensor field or whether the observed hotspot impacts a path between a source and sink. We believe there is a need to support heterogeneous sources and only regulate those sources that are responsible for, or impacted by, transient or persistent congestion conditions. Furthermore, we believe that closed-loop regulation of sources should not use high energy but instead hop-by-hop signaling that does not interfere with on-going data dissemination.

A number of other groups have looked at the issue of congestion control in wireless networks other than sensor networks. For example, WTCP [10] monitors the ratio of inter-packet separation for senders and receivers to detect and react to congestion in wireless LANs. SWAN [17] forces sources to re-negotiate end-to-end flows if congestion is detected in wireless ad hoc networks. RALM [11] employs TCP-like congestion and error control mechanisms for multicast support in wireless ad hoc networks. While multicast congestion control and congestion control in wireless networks are of interest they do not address the same problem space as energy efficient congestion detection and avoidance for sensor networks.

3. DESIGN CONSIDERATIONS

In what follows, we discuss the technical considerations that underpin the design of CODA while the detailed design is presented in Section 4.

The media access control plays a significant role in the performance of managing impulses of data in a wireless shared medium. There is a growing effort to design suitable TDMA schemes for sensor networks where energy can be conserved by turning off nodes periodically. Because TDMA can strictly control and schedule traffic flows in the network, the need for congestion control is essentially alleviated. However, many practical problems need to be solved before TDMA can be widely used in sensor networks, including synchronization and scheduling overhead.

A growing number of sensor networks use CSMA or variants for medium access. For example, the widely used Berkeley motes [5] use a simple CSMA MAC as part of the TinyOS [19] platform. In [16] the authors proposed a modified version of CSMA called S-MAC, which combines TDMA scheduling with CSMA's contention-based medium access, without a strict requirement for time synchronization. S-MAC uses virtual carrier sense to avoid hidden-terminal problems, allowing nodes other than the sender and receiver to go into a sleep mode (during the NAV after the RTS/CTS exchange), thus saving energy. S-MAC works well in a homogeneous network where the network is mostly used for supporting a single application. Congestion can still occur when using S-MAC or other contention-based schemes when the incoming traffic exceeds the node capacity and the queue overflows.

A number of considerations shape the design of CODA. In what follows, we discuss the MAC and congestion detection considerations.

3.1 CSMA Considerations

3.1.1 Throughput Issues

CODA takes a practical approach that assumes CSMA. The theoretical maximum throughput (channel utilization) for the CSMA scheme is approximately [1]:

$$S_{max} \approx \frac{1}{(1+2\sqrt{\beta})}(for\ \beta \ll 1), \tag{1}$$

where,

$$\beta = \frac{\tau C}{L}. \tag{2}$$

The performance of CSMA is highly dependent on the value of β, which is a measure of radio propagation delay and channel idle detection delay. τ is the delay in seconds, C is the raw channel bit rate and L is the expected number of bits in a data packet. If nodes can detect idle periods quickly, in other words have a very small β value, then CSMA can offer very good channel utilization regardless of the offered load.

Equation (1) gives the channel capacity of CSMA within one hop. In [7] the authors show that an ideal ad hoc multihop forwarding chain should be able to achieve 25% of the throughput that a single-hop transmission can achieve. This observation has important implications in the design of our congestion detection and closed-loop regulation mechanisms, as discussed in Section 3.2 and Section 4.2, respectively.

3.1.2 Hidden Terminals

CSMA suffers from the well-known hidden terminal problem in multihop environments. IEEE 802.11 utilizes virtual carrier sense (VC), namely an RTS/CTS exchange, to eliminate hidden terminals. In order to reduce the signaling overhead incurred by adding VC, IEEE 802.11 does not exchange RTS/CTS for small packets. In sensor networks, packets are usually small in nature (i.e., on the order of few tens of bytes) because of the low duty cycle requirement and traffic characteristics [8]. Therefore, the signaling cost is high if the RTS/CTS exchange is used for every message. Furthermore, sensor nodes have a very limited energy budget making the energy cost of doing this prohibitively high.

Usually, nodes other than event source nodes and the forwarding nodes will be silent most of the time when the workload of the network is low. Therefore, loss due to hidden terminals rarely occurs. In [14], the authors show that in general, when nodes are nicely randomized in sending/forwarding packets, the probability of having any hidden terminal is low even in dense networks. In S-MAC [16], an RTS/CTS exchange is used in an aggregated manner, not for every single packet, to reduce the energy cost.

In summary, in the context of sensor networks, the VC scheme is costly and mostly unnecessary during normal operations. There is a need to devise a scheme that can work satisfactory with or without the VC for collision avoidance, that incurs low cost or no cost during normal operations, and yet is responsive enough to quickly dissolve congestion[1]. In Section 3.2, we will discuss such a scheme.

[1]Depending on the sensing applications and the radio technologies, a user might choose to omit the VC for data packets but retain it for critical signaling message (e.g., control packets for routing protocol) in order to reduce overhead.

3.1.3 Link-layer ARQ

In the IEEE 802.11 MAC, a packet will be kept in the sending buffer until an ACK is received or the number of retransmissions exceeds a certain threshold. This mechanism increases the link reliability at the expense of energy and buffer space. However, both of these resources are scarce in sensor nodes where support for reliability may not always be necessary under normal operations (i.e., due to the application-specific nature of sensor networks not all data packets require strict reliability).

We believe there is a need for separation between reliability and congestion control in the design of sensor networks protocols. VC and link-layer ARQ as a reliable means of communication are essential for critical information exchange (e.g., routing signaling), but they are not necessarily relevant during congestion. In sensor networks, energy expenditure is more important than occasional data loss because of the natural redundancy inherent in disseminated sensor data. The main objective function is therefore to minimize energy expenditure. This is in contrast to TCP where lost data has to be always recovered. In our design, congestion control elements do not explicitly look at loss (unlike TCP), allowing CODA to decouple reliability from other control mechanisms. CODA is therefore capable of working with or without reliability elements, depending on the application.

3.2 Congestion Detection

Accurate and efficient congestion detection plays an important role in congestion control of sensor networks. There is a need for new congestion detection techniques that incur low cost in terms of energy and computation complexity. Several techniques are possible.

3.2.1 Buffer Queue Length

Queue management is often used in traditional data networks for congestion detection. However, without link-layer acknowledgments (some applications might not require this and hence would omit it to save the overhead, as discussed above), buffer occupancy or queue length cannot be used as an indication of congestion. To illustrate this, we perform an ns-2 simulation of the simple IEEE 802.11 wireless 5-node network shown in Figure 2. In the simulation, nodes 1 and 4 each start sending (1 second apart in simulation time) CBR traffic that consumes 50% of the channel capacity through node 2 to node 3 and 5, respectively. One of the sources stops sending data after 10 seconds. We ran two simulation runs, one with the VC enabled (including link ARQ), the other with it disabled and no link ARQ.

Figure 3 shows the time series traces for both channel loading and buffer occupancy as well as the packet delivery ratio measured at intermediate node 2. It is clear from the plot that the channel loading almost immediately rises to 90% during the time both sources are on. Congestion occurs and the packet delivery ratio drops from 100% to around 20% during this period. However, the buffer occupancy grows at a slower rate during this congestion period, particularly in the trace corresponding to the simulation where the VC is disabled. The buffer occupancy even drops at around 5 seconds into the simulation time, which provides false information about the congestion state. This is because without link ARQ, the clearing of the queue does not mean that congestion is alleviated since packets that

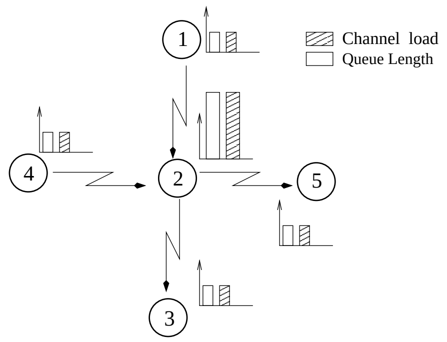


Figure 2: A simple IEEE 802.11 wireless network of 5 nodes to illustrate receiver-based congestion detection.

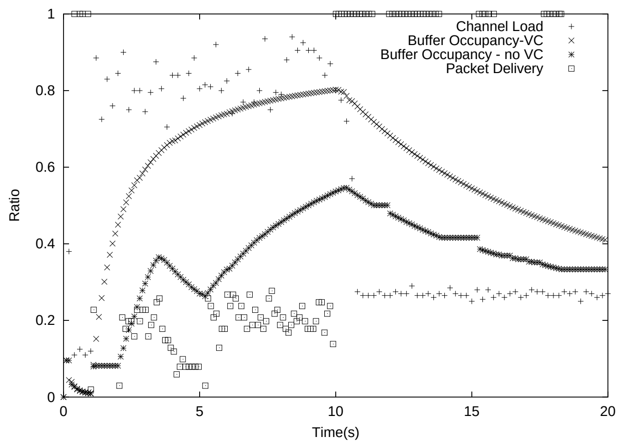


Figure 3: Channel load and buffer occupancy time series traces with and without virtual carrier sense (VC), and packet delivery trace with VC.

leave the queue might fail to reach the next hop as a result of collisions. Note that CSMA does not guarantee collision-free transmissions among neighboring nodes because of the detection delay [1].

This simple simulation shows that the buffer occupancy does not provide an accurate indication of congestion even when the link ARQ is enabled (as shown in the figure) except in the extreme case when the queue is empty or about to overflow. The first case indicates good traffic conditions and the latter one signals serious congestion. In other words, it is difficult to quantify a level of congestion or infer congestion solely based on buffer occupancy. This bimodal effect is not responsive enough and too coarse to provide smooth and efficient congestion control.

3.2.2 Channel Loading

In CSMA networks, it is straightforward for sensors to listen to the channel, trace the channel busy time and calculate the local channel loading conditions. Since S_{max} in Equation (1) gives the optimal utilization of the channel, if one senses that the channel loading reaches a certain fraction of the channel capacity, this would indicate a very high probability of collision [7].

Channel loading gives accurate information about how busy the surrounding network is but it is inherently a local mitigation mechanism. It has limited effect, for example, in detecting large-scale congestion caused by data impulses from sparsely located sources that generate high-rate traffic. Listening to the channel consumes a significant portion of energy [15] in a node. Therefore performing this operation all of the time is not practical in sensor networks. In Section 4.1, we propose a sampling scheme that activates local channel monitoring only at the appropriate time to minimize the energy cost while forming an accurate estimate of conditions.

3.2.3 Report Rate/Fidelity Measurement

For typical applications in sensor networks [6], the sinks expect a certain sampling rate or reporting rate coming from the sources. This rate is highly application-specific, and can be seen as an indication of event fidelity [12]; that is, the reporting rate from the source with respect to certain phenomenon should be high enough to satisfy the applications' desired accuracy. When a sink consistently receives a less than desired reporting rate, it can be inferred that packets are being dropped along the path, most probably due to congestion. In contrast, when ESRT [9] receives less than the desired reporting rate it signals the sources to increase the source reporting rate unless a packet with the congestion notification bit set is received at the sink.

Such fidelity measurement schemes need to operate on a much longer time scale compared to the packet transmission time scale, and consider:

- End-to-end delay between sources and sink nodes since only the sink recognizes its own requirements on the sampling rate.
- Processing delay - a sink typically collects data from multiple sources regarding the same phenomena (e.g., data aggregation/fusion). To cope with the different delays of packets, which possibly travel along different paths from different sources, the sink needs to wait a minimum period of time to collect reports before concluding anything.
- Stability - to avoid unnecessary reaction to transient phenomena that could cause oscillation, a sink should not respond too quickly to events and therefore should define an appropriate "observation period" over a longer time scale.

In short, detection based on the reporting rate is inherently slow and end-to-end in nature, and therefore, cannot cope well with transient hotspots in sensor networks.

4. CODA DESIGN

Hotspots can occur in different regions of a sensor field due to different congestion scenarios that arise. This motivates the need for CODA's open-loop hop-by-hop backpressure and closed-loop multi-source regulation mechanisms. These two control mechanisms, while insufficient in isolation, complement each other nicely. Different rate control functions are required at different nodes in the sensor network depending on whether they are sources, sinks, or intermediate nodes. Sources know the properties of the traffic while intermediate nodes do not. Sinks are best placed to understand the fidelity rate of the received signal, and in some applications, sinks are powerful nodes that are capable of performing complicated heuristics. The goal of CODA is to maintain low or no cost operations during normal conditions, but be responsive enough to quickly mitigate congestion around hotspots once it is detected. In what follows, we discuss CODA's backpressure and multi-source regulation mechanisms.

4.1 Open-Loop Hop-by-Hop Backpressure

Backpressure is the primary fast time scale control mechanism when congestion occurs. The main idea is to use the components mentioned in Section 3.2 to do local congestion detection at each node with low cost. Once congestion is detected, the receiver will broadcast a suppression message to its neighbors and at the same time make local adjustments to prevent propagating the congestion downstream.

A node broadcasts backpressure messages as long as it detects congestion. Backpressure signals are propagated upstream toward the source. In the case of impulse data events in dense networks it is very likely that the backpressure may propagate directly to the sources. Nodes that receive backpressure signals could throttle their sending rates or drop packets based on some local congestion policy (e.g., packet drop, AIMD, etc.).

When an upstream node (toward the source) receives a backpressure message, based on its own local network conditions it determines whether or not to further propagate the backpressure signal upstream. For example, depending on the local congestion policy a node may simply start to drop its incoming data packets upon receiving a backpressure message, preventing its queue from building up, rather than propagating the backpressure signal further upstream because of an overflowing queue. However, closed-loop congestion control would be required to deal with any persistent congestion in this case because of the node's policy to deal locally with the congestion indication, without propagating the backpressure signal.

We use the term *depth of congestion* to indicate the number of hops that the backpressure message has traversed before a non-congested node is encountered. The depth of congestion can be used by the routing protocol and local packet drop policies to help balance the energy consumed during congestion across different paths. Two simple schemes can be used:

- Consider the instantaneous depth of congestion as an indicator to the routing protocol to select better paths, thereby reducing traffic over the paths suffering deep congestion.
- Alternatively, rather than coupling congestion control and routing, the nodes can silently suppress or drop important signaling messages associated with routing or data dissemination protocols (e.g., interests [6], data advertisements [3], etc.). Such actions would help to *push* event flows out of congested regions and away from hotspots in a more transparent way.

Further investigation of using depth of congestion to assist routing is out of the scope of this paper. The rest of this section will describe the main elements and detailed operations of CODA's open-loop control.

4.1.1 Receiver-based Detection

As mentioned in Section 3.2, there are multiple good indications of congestion:

- A nearly overflowing queue.
- A measured channel load higher than a fraction of the optimum utilization. This provides a probabilistic indication of congestion by observing how closely the channel load approaches the upper bound.

Monitoring queue size comes almost for free except for a little processing overhead, but it provides only a bimodal indication. Listening to the channel either to measure the channel loading or to acquire signaling information for collision detection provides a good indication but incurs high energy cost, if performed all the time. Therefore, it is crucial to activate the latter component only at the appropriate time in order to minimize cost.

Consider the typical packet forwarding behavior of a sensor network node and its normal radio operational modes. The radio stays in the listening mode except when it is turned off or transmitting. When a carrier is detected on the channel, the radio switches into the receiving mode to look for a transmission preamble and continues to receive the packet bit stream. Before forwarding this packet to the next hop, CSMA requires the radio to detect an idle channel which implies listening for a certain amount of time. If the channel is clear during this period, then the radio switches into the transmission mode and sends out a packet. There is no extra cost to listen and measure channel loading when a node wants to transmit a packet since carrier sense is required anyway before a packet transmission. Based on this observation, we conclude that the proper time to activate the detection mechanism is when a node's send buffer is not empty. In other words, a node's radio might be turned off most of the time according to some coordination schemes (e.g., GAF [15], SPAN [2], S-MAC [16], etc.), but, whenever it wants to receive or transmit packets, the radio has to be at least in the listening mode.

Figure 2 illustrates a typical scenario in sensor networks in which hotspots or congestion areas could be created. In this example, nodes 1 and 4 each send CBR traffic that consumes 50% of the channel capacity through node 2 to node 3 and 5, respectively. Packets that are received by node 2 stay in its queue because of the very busy channel and are eventually dropped. This simple example shows that in a congested neighborhood, a receiver's (e.g., node 2, the forwarding node) buffer occupancy is high or at least non-empty. A node that activates the channel loading measurement during the moment when the buffer is not empty is highly responsive and at almost no cost. The channel loading measurement will stop naturally when the buffer is cleared, which indicates with high probability that any congestion is mitigated and data flows smoothly around the neighborhood. Based on this observation, there is little extra cost to measure channel loading if a node activates channel monitoring only when it is "receiving" a packet and needs to forward it later on. The only time CODA needs to do this is when a node has something to send, and it has to do carrier sense anyway for those situations.

4.1.2 Minimum Cost Sampling

A sensing epoch is defined as a multiple of the packet time. When a node starts sensing the channel (i.e., when it has something to send in its buffer), we require the MAC to listen for at least 1 epoch time to measure the channel load. During an epoch period, instead of continuously listening during the backoff time, a node performs periodic sampling to save energy so that the radio can be turned off during the interval. We use a simple sampling scheme where the channel load is measured for N consecutive sensing epochs of length E, with a predefined sampling rate to obtain channel state information; that is, the number of times that the channel state is busy or idle within a single sensing epoch. We then calculate the sensed channel load $\overline{\Phi}$ as the exponential average of Φ_n (the measured channel load during epoch n) with parameter α over the previous N consecutive sensing epochs, as shown in Equation (3).

$$\overline{\Phi}_{n+1} = \alpha\overline{\Phi}_n + (1-\alpha)\Phi_n, (n \in \{1, 2, ..., N\}, \overline{\Phi}_1 = \Phi_1). \quad (3)$$

If the send buffer is cleared before n counts to N, then the average value is ignored and n is reset to 1. The tuple (N, E, α) offers a way to tune the sampling scheme to accurately measure the channel load for specific radio and system architectures. In Section 5.2, we describe and demonstrate the tuning of these three parameters in an experimental sensor network testbed comprised of Berkeley Rene2 motes.

4.1.3 Suppression Message

When the sensed channel load exceeds a threshold (this can simply be S_{max}, as shown in later evaluation sections), congestion is implied. A node broadcasts a suppression message as a backpressure signal and at the same time exercises the local congestion policy. Although there is no guarantee that all neighboring nodes will get this message, at least some nodes will get it probabilistically. A node broadcasts a suppression message when it detects congestion based on channel loading and buffer occupancy. A node will continue broadcasting this message up to certain maximum number of times with minimum separation as long as congestion persists.

The suppression message provides the basis for the open loop backpressure mechanism and can also serve as an on-demand "Clear To Send" signal, so that all other neighbors except one sender (which could be picked randomly, or a node can assign more chances to more desirable senders) can be silenced at least for a single packet transmission time. This supports an implicit priority scheme in CODA in which the "chosen node" embedded in the suppression message can be selected based on data type or other metrics that essentially assign the chosen sender a higher priority to use the bandwidth. All nodes share a priority list of data types, and a certain data type has higher priority than others.

4.2 Closed-Loop Multi-Source Regulation

In sensor networks there is a need to assert congestion control over multiple sources from a single sink in the event of persistent congestion, where the sink plays an important role as a 1 to N controller over multiple sources. The cost of closed-loop flow control is typically high in comparison to simple open-loop control because of the required feedback signaling. We propose an approach that would dynamically regulate all sources associated with a particular data event. Under normal operation sources would regulate themselves at predefined rates (e.g., based on the data dissemination protocol [6] [3]) without the intervention of closed loop sink regulation.

When the source event rate (r) is less than some fraction η of the maximum theoretical throughput (S_{max}) of the channel the source regulates itself. When this value is exceeded $(r \geq \eta S_{max})$ [7], a source is more likely to contribute to congestion and therefore closed-loop control is triggered. The threshold η here is not the same as the threshold that used in local congestion detection, in fact η should be much smaller because of the result suggested in [7]. The source only enters sink regulation if this threshold is exceeded. At this point a source requires constant feedback (e.g., ACK) from the sink to maintain its rate (r). A source triggers sink regulation when it detects $(r \geq \eta S_{max})$ by setting a regulate bit in the event packets it forwards toward the sink. Reception of packets with the regulate bit set force the sink to send ACKs (e.g., 1 ACK per 100 events received at the sink) to regulate *all sources* associated with a particular data event. ACKs could be sent in an application specific manner. For example, the sink could send the ACK only along paths it wants to reinforce in the case of a directed diffusion [6] application. The reception of ACKs at sources would serve as a self-clocking mechanism allowing the sources to maintain the current event rate (r).

When a source sets its regulate bit it expects to receive an ACK from the sink at some predefined rate, or better, a certain number of ACKs over a predefined period allowing for the occasional loss of ACKs due to transient congestion. If a source receives a prescribed number of ACKs during this interval it maintains its rate (r). When congestion builds up, ACKs can be lost forcing sources to drop their event rate (r) according to some rate decrease function (e.g., multiplicative decrease, etc.). The sink can stop sending ACKs based on its view of network conditions. The sink is capable of measuring its own local channel loading (ρ) conditions and if this is excessive $(\rho \geq \gamma S_{\max})$ it can stop sending ACKs to sources.

Because the sink expects a certain reporting rate it can also take application-specific actions when this rate is consistently less than the desired reporting rate (i.e., the fidelity of the signal [12]). In this case the sink infers that packets are being dropped along the path due to persistent congestion and stops sending ACKs to sources. When congestion clears the sink can start to transmit ACKs again, and as a result, the event rate of the source nodes will increase according to some rate increase function (e.g., additive increase).

Because in some applications the sink is powerful in comparison to sensors and a point of data collection, it could maintain state information associated with specific data types. By observing packet streams from sources, if congestion is inferred it could send explicit control signals to those sources to lower their threshold value η to force them to trigger sink regulation even at a lower rate than others, (i.e. more important observers). This provides an implicit priority mechanism as part of closed-loop congestion control.

When the event rate at the sources is reset (e.g., via reinforcement [6]) to a value (r) that is less than some η of the maximum theoretical throughput (S_{max}) of the channel then the sources begin again to regulate themselves without the need of ACKs from the sink. Such a multimodal congestion control scheme provides the foundation for designing efficient and low cost control that can be practically implemented in sensor networks based on the Berkeley motes series [5] [4], as discussed in Section 5. Overall, closed loop multi-source regulation works closer to the application layer than its open-loop counterpart.

ESRT [9] always regulates all sources with a single high powered regulation message to the sources using a common rate for all sources. CODA only regulates the sources associated with a data event that have contributed to congestion or are impeded by hotspots between the sources and sink. In addition, CODA does not use a single high powered control message but hop-by-hop signaling between the sink and sources only when sources request such regulation.

5. SENSOR NETWORK TESTBED

In this section, we discuss some initial experiences implementing CODA on a real sensor system using the TinyOS platform [19] on Rene2 motes [5]. We report evaluation results, including measuring the β value, tuning the parameters for accurate channel load measurement, and finally, the evaluation of CODA with a generic data dissemination application.

The sensor device has an ATMEL 4MHz, low power, 8-bit microcontroller with 16K bytes of program memory, 1K byte of data memory, and a 32KB EEPROM serves as secondary storage. The radio is a single channel RF transceiver operating at 916MHz and capable of transmitting at 10kbps using on-off-keying encoding, the radio performs transmission and bit sampling in software (TinyOS). For all our testing we use a Non-Persistent CSMA MAC on top of the Rene2 motes.

5.1 Measuring the β Value

An important decision that must be made when using CODA's open-loop control mechanism described in Section 4.1 is the congestion threshold at which we should start applying backpressure. A first step in making this decision is to determine the maximum channel utilization achievable with the radio and MAC protocol being used.

As noted in Equation (1) in Section 3.1, with the CSMA MAC protocol, channel utilization in a wireless network depends on the propagation delay between the nodes with maximum physical separation that can still interfere with each other's communications, and the channel idle detection delay. In sensor networks, the maximum physical separation is typically tens of meters or less and as such the propagation delay is negligible for most purposes. Thus, if the channel idle detection delay is also negligible, CSMA should provide almost 100% utilization of the offered load of the channel. However, in practice, the utilization is much less due to the latency in the idle channel detection at the MAC layer. We can use the parameter β as defined in Equation (2) to predict how much this latency degrades the maximum channel utilization.

We measure the β value for the Rene2 mote using a simple experimental setup involving two motes both running TinyOS [19]. Stopwatches inserted in the MAC provide the basis for the measurement of β. Figure 4 shows the placement of the stopwatches within the receive and transmit flows of the MAC layer. Mote A starts its watch when the MAC receives a packet to be sent from the upper layers of the network stack and stops its watch when it detects the start-symbol of an incoming packet from mote B. The locations of the stopwatch trigger points in the mote B MAC are the same as in mote A, but the operations are reversed. It starts the watch when it receives a packet and stops it when it transmits.

A single iteration of the measurement consists of mote A sending a packet to mote B and mote B immediately reflecting the packet back to mote A. Due to the symmetry inherent in the placement of the stopwatch trigger points, β is proportional to half the difference between Stopwatch A and Stopwatch B:

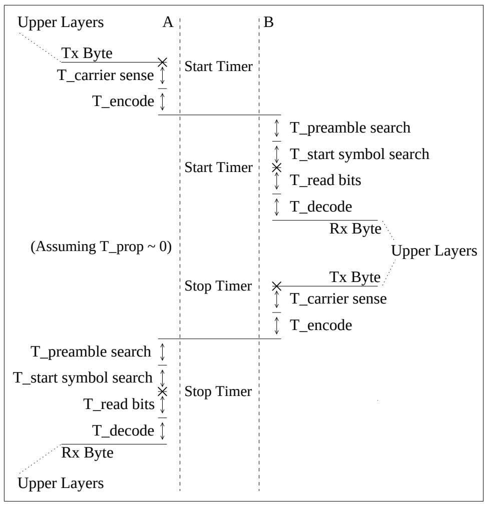


Figure 4: MAC layer stopwatch placement for β measurement. Diagram of receive and transmit state flows in the TinyOS MAC component code. Placement of the stopwatch start/stop trigger points are marked with an X.

$$\beta = \frac{(StopwatchA - StopwatchB)}{(2 * (Packet\ transmission\ time))}. \qquad (4)$$

Over 50 iterations, we measured an average β of 0.030 $\pm$ 0.003 (with confidence level of 95%) for the Rene2 motes. Substituting β into Equation (1), the standard expression for CSMA throughput (S_{max}), we predict a maximum channel utilization of approximately 71%.

5.2 Channel Loading Measurement and Utilization

The channel loading threshold that will trigger the backpressure mechanism must consider the tradeoff between energy savings and fidelity. Conserving energy implies a strategy that senses the channel load sparsely in time. However, the channel load measurement is most accurate when sensing densely in time. As a compromise between dense and sparse sampling, we use a scheme where the channel load is measured for N consecutive epochs of length E (with some fixed channel state sampling rate within this epoch) and an exponential average, with parameter α, is calculated to represent the sensed channel load. The problem then becomes to manipulate these three parameters (N, E, α) so that the node's sensed channel load is as close as possible to the actual channel load.

To do this optimization experimentally, we use two motes running TinyOS with a CSMA MAC. Mote S is a randomized CBR source sending at 4 packets per second. Mote R is the receiver that senses the channel load using the scheme mentioned in the previous paragraph. The channel is sampled once per millisecond for each epoch E for a total of N epochs. Using this setup we tested all combinations of $N \in \{2, 3, 4, 5\}$; $E \in \{100ms, 200ms, 300ms\}$ and $\alpha \in \{0.75, 0.80, 0.85, 0.90\}$. A time series average, of the exponential averages, is taken over 256 seconds for each

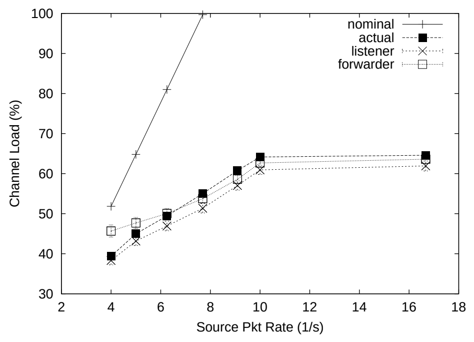


Figure 5: A limit on measured channel load is imposed by β. Nominal load curve increases with constant slope as the source packet rate increases, while the measured load saturates at far lower values.

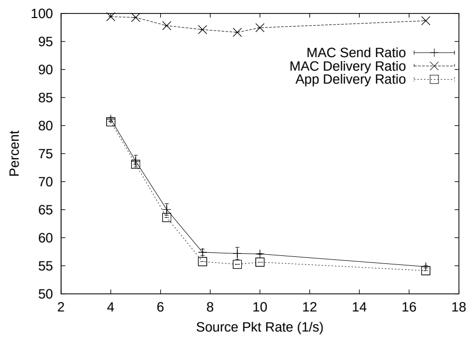


Figure 6: Packet delivery ratio at the MAC and application layers. Channel load saturation is not caused by packet collisions since the MAC delivery ratio is high across all tested source packet rates.

combination (1024 packets are sent). Using this method we found that the combination $(4, 100ms, 0.85)$ yielded the average sensed channel load at mote R closest to the actual average channel load (in %) calculated by mote S with an accuracy of 0.16±0.07. In general, we observe that the detection accuracy is not very sensitive (the difference is within 5%) to the three parameters. Therefore, manual calibration for each new CSMA-based radio might not always be necessary. As part of our future work, we intend to use the new generation of Mica2 mote which uses a different radio than Rene2 to investigate this issue.

In order to address the more realistic case of a node that both listens to and forwards packets, a third mote F is added to the previous experimental setup with all motes well within the transmission range of each other. Mote F forwards packets sent from mote S in a random interval between 30 and 130 milliseconds after it receives them, and also senses the channel load using the same scheme with the same (N, E, α) parameters that mote R uses. There is now contention for the channel since there are two packet sources (motes S and F). Mote R remains as a reference node to check the channel load sensed by mote F and also to keep track of the number of packets sent by motes S and F to calculate the delivery ratio.

With mote S sending 1024 packets, we measure the packet delivery ratio and channel load sensing accuracy using different source packet rates $(viz. 4, 5, 6.25, 7.69, 9.09, 10, 16.67)$. The average sensed channel load at R and F, along with the nominal channel load (calculated based strictly on offered load) and the actual channel load (calculated based on the actual number of packets sent by the MAC layer and the time taken), are plotted against the source packet rate in Figure 5.

Figure 5 shows the β-dependency of the CSMA MAC on the Rene2 mote. We can see from the plot of the nominal channel load that the offered load is more than enough to saturate the channel at points above 7.69 packets per second (source packet rate). However, we can also observe that regardless of the source packet rate, the measured channel load saturates below 70%. This is in agreement with the limitation predicted by β (as shown in Section 5.1), if we can assume that packet collision and buffer limitation do not contribute significantly to the observed reduced channel load. To verify this assumption, we analyze the packet delivery ratio at both the MAC and application layer.

We define the MAC packet delivery ratio as the percentage of packets sent by the MAC layer at motes S and F that are actually received by mote R. The application delivery ratio is the percentage of packets sent by the application layer at motes S and F that are actually received by mote R. Also, we define the MAC send ratio as the percentage of packets passed down by the application layer at both motes S and F that are actually sent out by the MAC layer of each respective mote. The buffer limitation of a mote could cause the dropping of packets from the application layer. In our Rene2 mote, we use the default buffer size of one packet.

Figure 6 shows that the MAC layer is able to send to the radio at most 81% of the packets it receives from the application layer. In fact, although the application delivery ratio degrades quickly as the source packet rate increases, nearly 100% of packets that are actually sent by the MAC are received by mote R. Thus, with this experimental setup, the packet loss is indeed due almost exclusively to the limit imposed by β, and not by collision at the receiver.

Also, from Figure 5, we see that the sensed channel load at motes F and R and the actual offered load are all in good agreement. This further reinforces our choices of the parameters (N, E, α) that we chose based on the data collected by the passive listener.

5.3 Energy Tax and Fidelity Penalty

We have created a simple generic data dissemination application to evaluate our congestion control scheme in a real wireless sensor network. The simple application implements the open-loop fast time scale component of our scheme using TinyOS and runs on our Rene2 mote testbed. When an intermediate (non source/sink) node receives a packet to forward, it enables channel load sensing. It disables sensing when its packet queue is emptied. If the channel load exceeds a given threshold value (70% as discussed in Section 5.1) during the sensing period, it transmits a suppress packet. Consistent with the simulation discussed in the next section, the sources use a multiplicative rate reduction policy. When a source receives a suppression message, it reduces its rate by half. A minimum rate of 2 packets per second is imposed such that a source sending at this rate will ignore subsequent suppression messages. An intermediate node stops transmitting for 400ms when it receives a

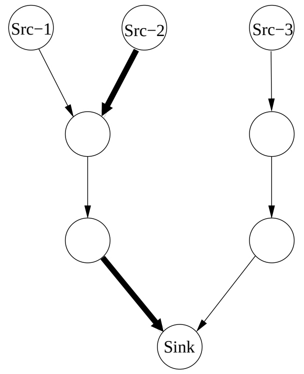


Figure 7: Experimental sensor network testbed topology. Nodes are well connected. Packets are unicast.

suppression message except if it is the "chosen node", as discussed in Section 4.1.3.

We define two metrics to analyze the performance of CODA on sensing applications:

- Average Energy Tax - this metric calculates the ratio between the total number of packets dropped in the sensor network and total number of packets received at the sinks over a simulation session. Since packet transmission/reception consumes the main portion of the energy of a node, the number of wasted packets per received packet directly indicates the energy saving aspect of CODA when compared to the case without CODA.
- Average Fidelity Penalty - we define the data fidelity as the delivery of the required number of data event packets within a certain time limit. This metric measures the difference between the average number of data packets received at a sink when using CODA and when using the ideal scheme discussed in the Appendix. Since CODA's control policy is to rate control the sources during periods of congestion then fidelity is necessarily degraded on the average. This fidelity difference, while normalized to the ideal fidelity obtained at the sink, indicates the fidelity penalty for using CODA. A lower fidelity penalty is desired by CODA to efficiently alleviate congestion while attempting not to impact the system performance seen by sensing applications.

The experimental sensor network testbed topology is shown in Figure 7. Packets are unicast with the arrows in Figure 7 indicating the unicast paths. The topology represents a dense deployment of motes so that the radio range of many of the motes in the graph overlap. The local congestion policy of the intermediate nodes can include the designation of a "chosen parent" (i.e., the chosen node, as discussed in Section 4.1.3) or set of parents, such that a suppression message sent by this node will invoke the suppression method at its neighbors except for the chosen parent(s). This allows for traffic priority. In Figure 7, the thick arrows show the "chosen paths". Paths funnel events toward the sink node. The three source nodes provide a high traffic load to the network, representing a data impulse. The source rates are: Src-1: 8pps (packets per second), Src-2: 4pps, Src-3: 7pps.

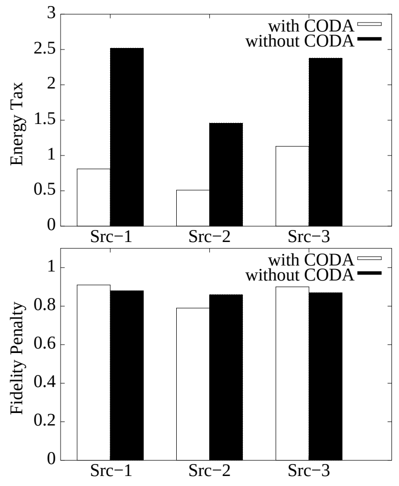


Figure 8: Improvement in energy tax with minimal fidelity penalty using CODA. Priority of Src-2 evident from the fidelity penalty results.

The sink node counts the number of packets it receives from each respective source. Each source node counts the number of packets it actually sends over the air and the number of packets the application tries to send. The difference between these last two counters measures the number of packets a source's MAC layer drops.

Using ten 120-second trials, we obtain average values for the packets received, sent, and attempted to send but failed (e.g., because of a busy channel, buffer overflow, etc.) corresponding to each of the three sources. From this measured data, we calculate the energy tax and fidelity penalty for each of the three sources. Figure 8 shows the result of experiments with and without CODA enabled. We can see from the figure that with a small fidelity penalty compared with non-CODA system we can achieve a 3x reduction in energy tax on average. We observe that without CODA the fidelity penalty is the same for all three sources. With CODA the penalty for Src-2 is much less than the other two sources. In contrast with the other sources, the fidelity penalty for Src-2 is less with CODA than without CODA. The reason is because the data type of Src-2 has the highest priority. When CODA is used in the presence of congestion, the suppression mechanism favors Src-2 packets over the others.

6. SIMULATION RESULTS

We use packet-level simulation to obtain preliminary performance evaluation results for CODA. We also discuss the implications of our results on the design choices that shape CODA.

6.1 Simulation Environment

We implemented both open-loop backpressure and closed-loop regulation in the ns-2 [18] simulator in their simplest instantiation; that is, a simple multiplicative decrease function is implemented at each sensor source by an application agent. The reception of suppression messages at the source, or, in the case of closed-loop control, not receiving a sufficient number of ACKs from the sink over a predefined period of time, will cause a source to cut its rate by half. For intermediate nodes (non source/sink), local congestion policy is such that suppression messages will halt a node's transmission for a small random number of packet times (i.e., packet transmission times) unless a node is the chosen node specified in the suppression message, as discussed in Section 4.1.3.

In all our experiments, we use random topologies with different network sizes. We generate sensor networks of different sizes by placing nodes randomly in a square area. Different sizes are generated by scaling the square size and keeping the nominal radio range (40 meters in our simulations) constant in order to approximately keep the average density of sensor nodes constant. In most of our simulations, we study five different sensor fields with size ranging from 30 to 120 nodes in increments of 20 nodes. For each network size, our results are averaged over five different generated topologies and each value is reported with its corresponding 95% confidence interval.

Our simulations use a 2 Mbps IEEE 802.11 MAC provided in ns-2 simulator with some modifications. First, we disabled the use of RTS/CTS exchanges and link-layer ARQ for data packets for the reasons discussed in Section 3.1 because we want to capture the realistic cases where reliable delivery of data is not needed and the fidelity can be compromised to save energy. Although we use IEEE 802.11 in the simulation, most sensor platforms use simpler link technologies where the ARQ is not enabled by default, (e.g. Berkeley motes). Next, we added code to the MAC to measure the channel loading using the epoch parameters ($N = 3, E = 200ms, \alpha = 0.5$), as defined in Section 4.1.2. The choice of the parameters is not crucial because the ns-2 simulator does not model the details of the IEEE 802.11 physical layer. The MAC broadcasts suppression messages when the measured channel loading exceeds a threshold of 80%. We have added code to model the channel idle detection delay with a β of 0.01, which yields a S_{max} around 80%. Closed-loop multi-source regulation is implemented as an application agent attached to source-sink pairs. A sink receiving a data packet with the regulate bit set sends an application ACK to the source once every 100 data packets that it receives. A source on the other hand halve its rate if no ACK is received within a period $T_{regulate} = 200 \times Pkt\ Interval + \delta$ when closed-loop control is triggered. The choice of $T_{regulate}$ is such that the source can tolerate an ACK loss before slowing down while δ is chosen to accommodate the maximum round-trip delay depending on the network size.

Finally, we use directed diffusion [6] as the routing core in the simulations since our congestion control fits nicely into the diffusion paradigm, and since doing so allows insight into CODA's interaction with a realistic data routing model where congestion can occur.

In most of our simulations, we use a fixed workload that consists of 6 sources and 3 sinks, all sources are randomly selected from nodes in the network. Sinks are uniformly scattered across the sensor field. A sink subscribes to 2 data types corresponding to two different sources. This models the typical case in which there are fewer sinks than sources in a sensor field. Each source generates packets at a different rate. An event packet is 64 bytes and an interest packet is 36 bytes in size [6].

6.2 Results and Discussion

We evaluate CODA under the three distinct congestion scenarios discussed in Section 1 to best understand its behavior and dynamics in responding to the different types of congestion that could be found in sensor networks. First we look at a densely deployed sensor field that generates impulse data events. Next, we examine the behavior of our scheme dealing with transient hotspots in sparsely deployed sensor networks of different sizes. Last, we examine the case where both transient and persistent hotspots occur in a sparsely deployed sensor field generating data at a high rate.

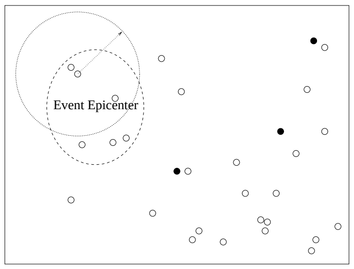


Figure 9: Network of 30 nodes. Sensors within the range of the event epicentre, which is enclosed by the dotted ellipse, generate impulse data when an event occurs. The circle represents the radio range (40m) of the sensor.

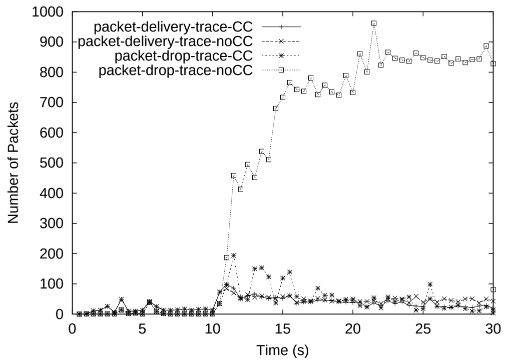


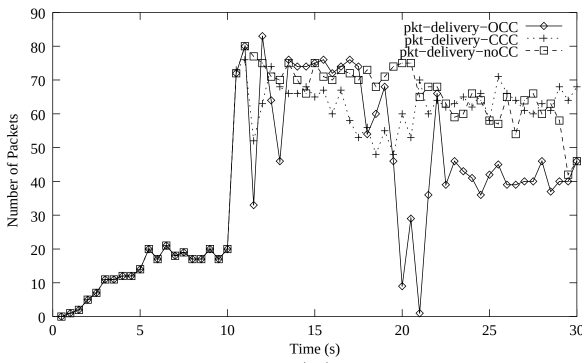
Figure 10: Time series traces for densely deployed sources that generate high rate data.

6.2.1 *Congestion Scenario - Dense Sources, High Rate*

We simulate a network with 30 nodes, as shown in Figure 9, emulating a disaster-related event (e.g., fire, earthquake) that occurs 10 seconds into the simulation. Each node within the epicenter region, which is enclosed by the dotted ellipse, generates at least 100 packets per second sent toward the sinks, shown as filled black dots in the figure.

Figure 10 shows both the number of packets delivered and the packets dropped as time series traces. For the packet delivery trace, we count the number of data packets a sink receives every fixed interval of 500ms, which indicates the fidelity of the data samples. For the packet dropped trace, we count the number of data packets dropped within the whole network every 500ms.

From the traces, it is clear that the difference in data delivery (fidelity) with and without CODA is small, while the number of packets dropped is an order of magnitude smaller (hence the energy savings) when congestion control is ap-



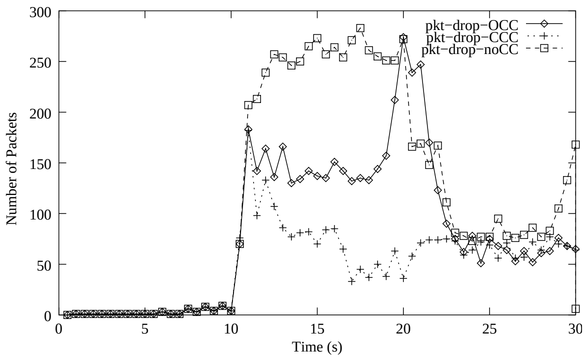


Figure 11: (a) Packet delivery and (b) Packet dropped time series traces for a 15-node network with low rate traffic. The plots show the traces for three cases: when only open-loop control (OCC) is used, both open-loop and closed-loop control (CCC) are enabled and when congestion control is disabled (noCC).

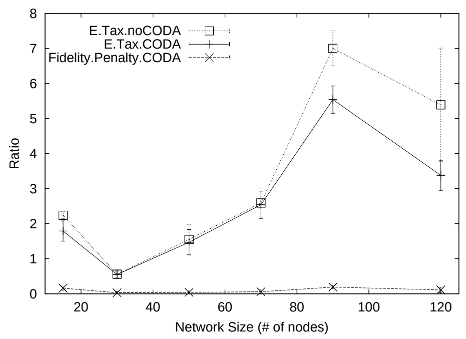


Figure 12: Average energy tax and fidelity penalty as a function of the network size when only CODA's open loop control is used.

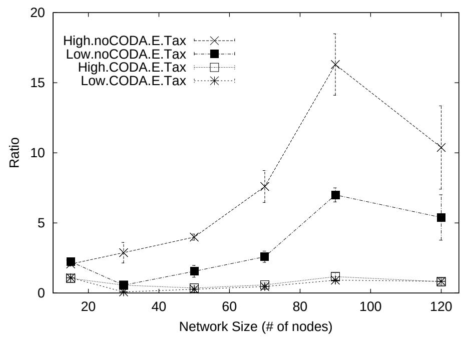


Figure 13: Energy tax as a function of network size for high and low rate data traffic. The difference between the data points with and without CODA indicates the energy saving achieved by CODA.

plied. We can also observe from the plot that the congestion is effectively relieved within 2 to 3 seconds. This shows the adaptive property of CODA. The delivery plot reflects the real system goodput, which is highly dependent on the system capacity, indicating the maximum channel utilization. When impulses happen, the channel is saturated so it can deliver only a limited amount of data. CODA's open-loop backpressure (even with a very simple policy) adapts well to operate close enough to the channel saturation, as shown in Figure 10, while efficiently alleviating congestion. This greatly reduces the number of packets dropped thereby saving energy, which is the key objective function for CODA. The same simulation scenario is repeated 5 times using different topologies of the same size. Overall, using CODA obtains packet (energy) saving up to $88 \pm 2\%$ while the fidelity penalty paid is only $3 \pm 11\%$.

6.2.2 *Congestion Scenario - Sparse Sources, Low Rate*

To examine the ability to deal with transient hotspots, in these simulations all six sources send at low data rates, at most 20 packets per seconds. Four of the sources are randomly selected so that they are turned on and off at a random time between 10 and 20 seconds into the simulation.

Figure 11 shows the packet delivery and packet drop traces for one of the simulation sessions in a network of 15 nodes. Observe in Figure 11(a), the difference in fidelity between the three cases is small, except for around 20 seconds in to the trace, where only open-loop control is used. Figure 11(b) shows a large improvement in energy savings (i.e., packet drop reduction) especially when closed-loop control is also enabled together with open-loop control. Again, the figure shows that at around 20 seconds into the trace, open-loop control cannot resolve congestion since there is no reduction in the number of dropped packets and there is low delivery during this period. This is because transient hotspots turn into persistent congestion at around 18 seconds into the trace until four of the sources turn off after 20 seconds. Open-loop control cannot deal with persistent congestion unless the hotspots are close to the sources, as discussed in Section 4.1. On the other hand, the trace corresponding to closed-loop regulation also shows that the fidelity is maintained while effectively alleviating congestion with only a small amount of extra signaling overhead. The signaling cost of CODA is less than 1% with respect to the number of data packets delivered to the sink.

The same behavior can be observed in Figure 12, where the two metrics (i.e., energy tax and fidelity penalty) are plotted as a function of the network size. Note that when using only open-loop control, the energy savings has a large variation, indicated by the error bars that represent 95% confidence intervals. This indicates that congestion is not always resolved, especially for larger-sized networks. This is

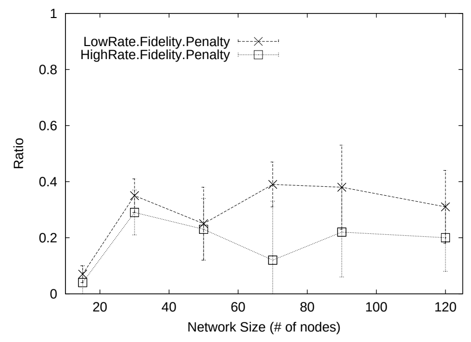


Figure 14: Fidelity penalty as a function of the network size for high and low rate data traffic.

because in larger networks, persistent hotspots, which localized open-loop control is unable to resolve, are more likely to occur given the long routes between source-sink pairs. When closed-loop control is also enabled, the energy savings is large, up to 500% with a small variation, and increases with the growing network size, as shown in Figure 13.

Overall, the gain from using open-loop control in larger networks is limited. Hotspots are likely to persist when the sources are generating data at a low rate because of possible long routes. Enabling closed-loop control even at low source rates can improve the performance significantly, with the addition of a small overhead for the control packets from sinks. Note, that the amount of overhead is only a small fraction, i.e. 1%, of the number of data packets that the sink receives. This result suggests that except for small networks, always enabling closed-loop control is beneficial, regardless of the source rate.

6.2.3 Congestion Scenario - Sparse Sources, High Rate

We examine the performance of our scheme in resolving both transient and persistent hotspots where sparsely located sources generate high data traffic. In the simulations, all sources generate 50 packets per second data traffic over the 30 second simulation time. Both open-loop and closed-loop control are used throughout the simulations. Figure 13 shows that CODA can obtain up to 15 times or 1500% energy savings. Figure 14 shows that CODA can maintain a relatively low fidelity penalty of less than 40% as compared to the ideal scheme. Note that we have yet to implement the additive rate restoration scheme in the closed-loop control; therefore, the fidelity performance of CODA is not fully explored in these initial results.

7. CONCLUSION

In this paper, we have presented an energy efficient congestion control scheme for sensor networks called CODA. The framework is targeted to CSMA-based sensors and comprises three key mechanisms: (i) receiver-based congestion detection, (ii) open-loop hop-by-hop backpressure, and (iii) closed-loop multi-source regulation. We have presented some preliminary experimental results from a small sensor network testbed based on TinyOS running on Berkeley Rene2 motes. We defined two performance metrics, average energy tax and average fidelity penalty, which capture the impact of CODA on sensing applications' performance. A number of important results came out of this first implementation. It was feasible to measure β, channel loading at the receiver, and to evaluate CODA with a generic data dissemination scheme. We have also demonstrated through simulation that CODA can be integrated to support data dissemination schemes and be responsive to a number of different congestion control scenarios that we believe will be prevalent in future sensor network deployments. Simulation results indicated that CODA can improve the performance of directed diffusion by significantly reducing the average energy tax with minimal fidelity penalty to the sensing application. These initial results look very promising and provide a basis for further larger scale experimentation on which we hope to report in the future. Our intention is to release the source code for CODA after these experiments have been conducted. We are also studying as part of our future work the performance benefits of using CODA with reliable transport mechanisms such as PSFQ [13].

8. ACKNOWLEDGEMENT

We would like to thank Professor Hari Balakrishnan (who helped shepherd this paper) and the anonymous reviewers for their constructive comments.

9. REFERENCES

[1] D. Bertsekas and R. Gallager. *DATA NETWORKS, second edition*. Prentice Hall, Upper Saddle River, New Jersey, 1992.

[2] B. Chen, K. Jamieson, and H. Balakrishnan. An energy efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *Proc. of the 7th Annual International Conference on Mobile Computing and Networking (Mobicom 2001)*, pages 85–96, July 2001.

[3] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proc. of the 5th Annual International Conference on Mobile Computing and Networking (Mobicom 1999)*, pages 174–185, 1999.

[4] J. Hill and D. Culler. Mica: A wireless platform for deeply embedded networks. *IEEE Micro.*, 22(6):12–24, November/December 2002.

[5] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System arthitecture directions for network sensors. In *Proc. of the 9th International Conf. on Arch. Support for Programming Languages and Operating Systems*, pages 93–104, November 2000.

[6] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proc. of the 6th Annual International Conference on Mobile Computing and Networking (Mobicom 2000)*, pages 56–67, August 2000.

[7] J. Li, C. Blake, D. D. Couto, H. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *Proc. of the Seventh Annual International Conference on Mobile Computing and Networking*, pages 61–69, July 2001.

[8] G. Pottie and W. Kaiser. Wireless integrated network sensors. *Communications of the ACM*, 43(5):51–58, May 2000.

[9] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz. Event-to-sink reliable transport in wireless sensor networks. In *Proc. of the 4th ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2003)*, pages 177–188. Annapolis, Maryland, June 2003.

[10] P. Sinha, N. Venkitaraman, R. Sivakumar, and V. Bharghavan. Wtcp: A reliable transport protocol for wireless wide-area networks. In *Proc. of the 5th Annual International Conference on Mobile Computing and Networking (Mobicom 1999)*. Seattle, August 1999.

[11] K. Tang and M. Gerla. Reliable on-demand multicast routing with congestion control in wireless ad hoc networks. In *Proceedings of SPIE 2001*. Denver, August 2001.

[12] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. Infrastructure tradeoffs for sensor networks. In *Proc. of First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, pages 49–58. Atlanta, September 2002.

[13] C.-Y. Wan, A. T. Campbell, and L. Krishnamurthy. Psfq:a reliable transport protocol for wireless sensor networks. In *Proc. of First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, pages 1–11. Atlanta, September 2002.

[14] A. Woo and D. Culler. A transmission control scheme for media access in sensor networks. In *Proc. of the 7th Annual International Conference on Mobile Computing and Networking (Mobicom 2001)*, pages 221–235, July 2001.

[15] Y. Xu, J. Heideman, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proc. of the 7th Annual International Conference on Mobile Computing and Networking (Mobicom 2001)*, pages 70–84, July 2001.

[16] W. Ye, J. Heidemann, and D. Estrin. An energy efficient mac protocol for wireless sensor networks. In *Proc. of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom 2002)*, pages 1567–1576. New York, June 2002.

[17] G.-S. Ahn, A. T. Campbell, A. Veres, and L.-H. Sun. Supporting service differentiation for real-time and best effort traffic in stateless wireless ad hoc networks (swan). *IEEE Transactions on Mobile Computing*, 1(3):192–207, July-September 2002.

[18] The network simulator - ns2. Available from http://www.isi.edu/nsnam/ns/.

[19] Tinyos homepage. Available from http://webs.cs.berkeley.edu/tos/.

APPENDIX

Experimentally determining the ideal fidelity of the network

Assume that there exists an ideal congestion control scheme that is capable of rate-controlling each source to share the network capacity equally without dropping each other's packets. The problem then becomes finding out the network capacity or at least the upper bound of the network capacity. The actual capacity of the network is application-specific depending on several factors including the radio bandwidth, the MAC operations, the routing/data dissemination schemes, and the traffic pattern. Assume that the network is homogeneous in the sense that all wireless links are symmetrical and equal. We can determine the upper bound of the network capacity in a simple and practical manner through experimentation. The idea is as follows:

Def: $C_{max,i}$ *= Maximum data delivery rate of a path i associated with source i, in which the packet drop rate is minimum.*

Consider that multiple distinct sources send data toward a common sink travelling along different paths. Assume these dissemination paths from the sources to the sink coincide with each other and share at least one common link. This is a reasonable assumption considering the funneling effect toward the sink that these transmissions have to share at least the air around the sink. Therefore, the data dissemination capacity for a sink is limited by Max$\{C_{max,i}\}$. Thus we can find the upper bound and calculate ideal fidelity by measuring Max$\{C_{max,i}\}$ experimentally.