

Sink-to-Sensors Congestion Control Strategy

Ramanuja Vedantham, Raghupathy Sivakumar
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, USA
{ramv,siva}@ece.gatech.edu

Seung-Jong Park
Department of Computer Science and CCT
Louisiana State University, Baton Rouge, USA
sjpark@bit.csc.lsu.edu

Abstract—The problem of congestion in sensor networks is significantly different from conventional ad-hoc networks and has not been studied to any great extent thus far. In this paper, we focus on providing congestion control from the sink to the sensors in a sensor field. We identify the different reasons for congestion from the sink to the sensors and show the uniqueness of the problem in sensor network environments. We propose a scalable, distributed approach that addresses congestion from the sink to the sensors in a sensor network. Through ns2 based simulations, we evaluate the proposed framework, and show that it performs significantly better over a basic approach which does not provide any congestion control.

I. INTRODUCTION

Wireless Sensor Networks (WSNs) have gained tremendous importance in recent years because of its potential use in a wide variety of applications. This, along with the unique characteristics of these networks [1], has spurred a significant amount of research for coming with network protocols specifically tailored for sensor networks. In this paper, we propose an approach for congestion control from sink-to-sensors (*downstream*).

The need for congestion control in sensor network is clearly motivated by the nature of the application that it is used for. In mission critical environments such as military applications, it is necessary that the sink is able to transmit the data to the sensors in the least possible time. As we show in Section II, merely transmitting at a higher data rate will not accomplish this as it results in more number of collisions and packet losses. This translates to poor usage of network and node resources per packet delivered, which are a premium for sensor nets. The congestion in the network is increased in the presence of reverse path traffic, which accounts for the bulk of traffic in WSNs. Thus, congestion control is vital as it allows for fast and reliable message delivery with efficient use of available network bandwidth and energy resource of sensors.

There have been a few works that have addressed congestion control in sensor networks from sensors-to-sink (*upstream*) [2], [3]. However, these works are not applicable in the downstream direction because of the differences in the nature of communication paradigm and the availability of node resources. For example, in downstream communication in WSNs, when all nodes are receivers of a particular message, the buffering of different packets in a message is not an issue as the nodes will anyway store all the packets in a message. Hence, the receiving rate can be much higher than the sending rate without worrying about buffer overflow. This assumption is not desirable in the upstream direction as the intermediate

nodes should not be required to buffer all packets sent by other sensors. Also, while the communication model in the downstream direction is one-to-many, it is the converse in the upstream direction and this consideration will influence the design of the congestion control mechanism. Our work is orthogonal to these upstream approaches and can coexist and compliment them by providing congestion control in the downstream direction.

In this work, we clearly motivate the need for *explicit* downstream congestion control. We then propose an adaptive, explicit rate control approach, called *CONgestion control from Sink to Sensors* (CONWISE), that adjusts the downstream sending rate at each of the sensor nodes to utilize the available network bandwidth depending on the congestion level in the local environment. The proposed approach is scalable and easily implementable and can provide large performance benefits and efficient usage of resources with minimal overheads. The realization of CONWISE's design addresses and leverages the specific characteristics of the sensor network environments, as we elaborate in later sections.

The rest of the paper is organized as follows: Section II defines the scope of the paper, motivates the problem of downstream congestion control and identifies the challenges. Section III enumerates the key design elements in CONWISE that address the research challenges. Section IV presents CONWISE in detail while Section V evaluates the performance of CONWISE with a basic scheme that does not provide any congestion control. Section VI discusses related works and Section VII concludes the paper.

II. PROBLEM DEFINITION

In this paper, we propose a mechanism for performing downstream congestion control in the following context:

- *Network Model*: We consider a multi-hop WSN with one or more sinks coordinating the sensors in the field.
- *Receiver Model*: We assume that all or only a subset of nodes are receivers for a particular message.

Given the above problem definition, our goal is to determine the rate at which each node will forward the packets in a scalable and distributed fashion, irrespective of whether it is a receiver or non-receiver, without *over-utilizing* or *under-utilizing* the available bandwidth.

A. Motivation

We now present the factors causing congestion in the downstream direction in WSNs. The factors that contribute to

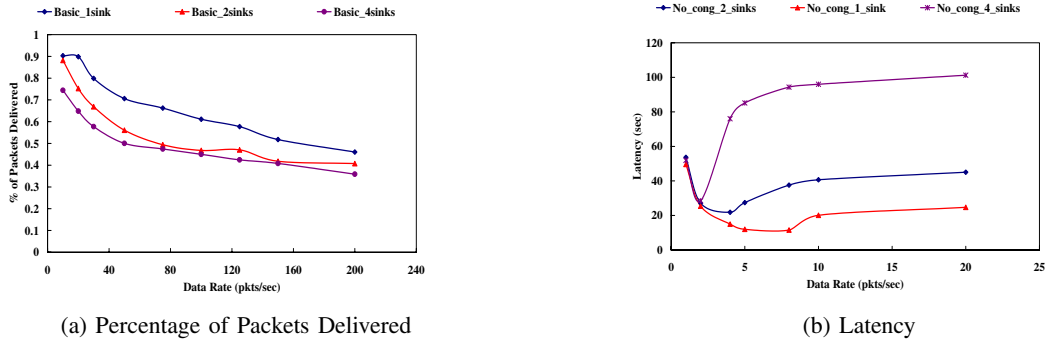


Fig. 1. Impact of Congestion for Multiple Sinks

the congestion include *reverse path traffic* from the sensors to the sink and *broadcast storm problem*, which refers to the higher level of contention and collisions occurring due to a series of local broadcasts. We have observed that in a network of 650x650m grid with 100 nodes, the number of nodes receiving a message of 100 packets decreases from about 97% when the aggregate background traffic is about 25 Kbps to about 76% when the background traffic is increased to 400 Kbps. Also, for a network of 650x650m we have observed that the success rate drops from about 99% for the 100 node scenario to 83% for the 800 node scenario illustrating the effect of broadcast storm.

While the above-mentioned reasons, clearly illustrate the need for congestion control mechanisms in sensor networks, the impact of the lack of such approaches in the context of WSNs needs to be studied. Figure 1 considers the impact of congestion in the presence and absence of reliability. Figure 1 (a) shows the percentage of packets delivered in a message for a network of size 650 x 650m and 400 nodes, with increasing number of sinks in the absence of reliability. We observe that the percentage of packets received decreases with increasing sending rates as well as number of sinks. Figure 1 (b) shows the the latency of reception of all packets in a message for different number of sinks for the same network with 800 nodes with strict reliability semantics. We consider simple flooding with out-of-sequence forwarding as the routing protocol and a NACK based scheme with retransmission timeout for loss recovery [4]. Here again, we notice that the latency increases as the sending rate increases beyond a certain point and that the point at which the minimum latency occurs shifts for different number of sinks.

B. Challenges

In realizing our main goal of determining the sending rate for all the nodes, we now identify the following key challenges in a downstream WSN environment:

1) *Receivers and Non-receivers*: In the target environment, a node can either be a receiver or a non-receiver. In the case of non-receivers, the resources of that node must be utilized to a bare minimum. Also, receivers may not form a contiguous region with the sink and can have multiple intermediate hops between two receivers or between a receiver

and sink. The design of the algorithm should be done such that the resources of the receivers are utilized in an efficient way with the non-receivers participating to a minimal extent.

2) *Lack of Buffering at Non-receivers*: As mentioned earlier, a non-receiver should utilize its resources minimally. So, these nodes will act as a mere forwarders and cannot aid in retransmission of a lost packet.

3) *Differing Congestion Levels*: In a WSN, different regions can have different congestion levels because of: (i) reverse path congestion in a localized region due to increase in sensing activity; (ii) differences in the node density across the network; and (iii) node failures leading to other neighboring nodes having to share the traffic load carried by this node. This difference in node densities and background traffic across the sensor network can cause the local congestion to be different in both size and intensity.

4) *Minimizing Delay*: A receiver would ideally like to receive packets in a message in the fastest possible time. To realize this, it is important that the rate of reception of messages at a receiver is affected solely by the fastest available path upstream towards the sink. It should not be influenced by alternate slower paths to the sink or by the congestion level downstream of the node.

5) *Efficient Data Dissemination*: In conventional flooding based approaches, a node forwards every new packet irrespective of whether it is necessary at the downstream nodes. However, this is clearly inefficient and leads to increased contention and collisions in the network. To decrease the contention in the network, a node should not forward a packet unless it is required by the nodes downstream of it.

6) *Network Dynamics*: The congestion level in the network may not only have variations across different regions but also have variations with time for a given region. This may be because of transients occurring due to node mobility and failures, differences in the sensing and reporting activity and the nature of the reverse path traffic over a period of time. Thus, the congestion control mechanism should be responsive to temporal network dynamics.

III. CONSIDER DESIGN ELEMENTS

While we present the details of the CONSIDER approach in Section IV, we discuss in this section the key design

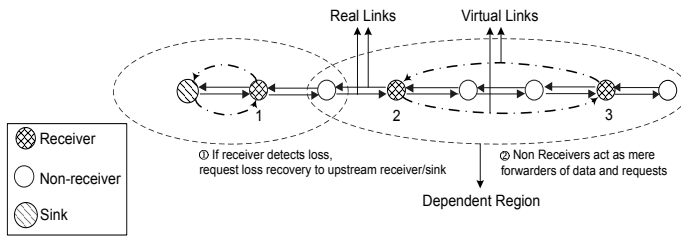


Fig. 2. Dependent Regions and Virtual Links

components and how they address the challenges described in Section II. At a high-level, there are fundamentally three components to the design of CONWISE: (i) determination of the receiving rate for a receiver; (ii) determination of sending rate for a receiver; and (iii) determination of receiving rate for a non-receiver. The sending rate of a non-receiver is set to the receiving rate as we will explain later in this section. These components are also determined on a periodical basis defined by *epoch* duration. This allows the nodes to adapt to the available bandwidth on a periodical basis addressing the challenge pertaining to network dynamics. We now present how the design of CONWISE addresses the challenges.

A. Differentiating Receivers from Non-Receivers

There are fundamentally two challenges associated with the classification of nodes as receivers and non-receivers: (i) Differences in resources available in receivers and non-receivers, and (ii) Handling the lack of buffering at non-receivers.

In CONWISE, every node maintains two parameters: a receiving rate and a sending rate. Here, receiving rate corresponds to the rate of successful reception of packets and sending rate corresponds to the rate at which packets are forwarded from the node. In CONWISE, the sending and receiving rates of a receiver are decoupled while they are maintained the same for a non-receiver. This decoupling of the sending and receiving rates for the receivers is possible as they anyway buffer the packets and this is used to alleviate local congestion. The receiving rate of a receiver is determined based on the path from which it received the maximum number of packets in an epoch while the sending rate is determined based on the receiving rates of the downstream receivers. We will elaborate more on this aspect in Section IV. Since non-receivers ideally would not prefer to buffer packets, the sending and receiving rates are set to the same value. A non-receiver reduces its receiving rate to the sending rate desired for that node. When two receivers are multiple-hops apart, the intermediate non-receivers form a chain of links which essentially acts as a single virtual link to aid the communication between two receivers. Figure 2 also shows the virtual links connecting successive receivers in a linear topology. In this linear topology there are two non-receivers connecting the second and third receiver. If the receiving rate of the third receiver is 10 pkts/sec, the sending and receiving rates of the non-receivers are also set to 10 pkts/sec.

B. Handling Different Congestion Levels

To address the variation in node densities and the background traffic across the sensor network, there is a need for a localized approach based on the contention level in the local environment.

In CONWISE, a dependent-region based localized approach is proposed to tackle this problem. A dependent-region is defined as the union of transmission regions between two successive receivers with any intermediate nodes also contributing to that dependent region. In CONWISE, each node apart from maintaining the *sending rate* also maintains another parameter called the *maximum sending rate*. While the maximum sending rate is determined based on the channel conditions of the local node, the current sending rate is determined based on both the channel conditions of the local node and that of the downstream nodes. The value of the sending rate for the upstream receiver is determined based on the receiving rate reported by the downstream nodes with the local receiving rate being the upper bound. A receiver upon reception of the feedback in terms of the receiving rate sets its sending rate appropriately. The intermediate nodes that are not receivers adjust their sending and receiving rates to the same value. Figure 2 shows the dependent regions in a sample linear topology, where the marked dependent region encompasses five 1-hop real links. In this example, if the receiver 2 has a receiving rate of 12 pkts/sec and if the receiver 3 has a receiving rate of 10 pkts/sec, the sending rate of receiver 2 is set to 10 pkts/sec.

C. Fast Reception for Receivers

One of the challenges is to ensure that every node receives the message with minimum latency. For this, it is necessary to receive all the packets in a message in the fastest possible time irrespective of downstream channel conditions.

In CONWISE, every receiver always determines the fastest possible route from the sink. This is accomplished by the following sequence of operations: (i) Every receiver maintains the number of packets received from each of its upstream receivers; (ii) At the expiry of an epoch, the node from which the maximum number of packets was received is notified as the preferred upstream receiver; (iii) The preferred upstream receiver sets its sending rate based on the receiving rate/s of its downstream receiver/s. The notifications only affect the sending rate and not the receiving rate of the upstream receiver. This ensures that packets are received at the fastest possible rate to the receiver irrespective of the downstream channel conditions. At any given time, the congestion level at a receiver is solely influenced only by the congestion levels along the fastest possible path. In figure 3, node 5 has three upstream receivers. However, once it elects node 3 as its preferred upstream receiver, the reception of message is completely influenced by the congestion in the path indicated.

D. Selective Transmission

In order to minimize the contention in the network, it is necessary that a node, whether it is a receiver or not, forwards

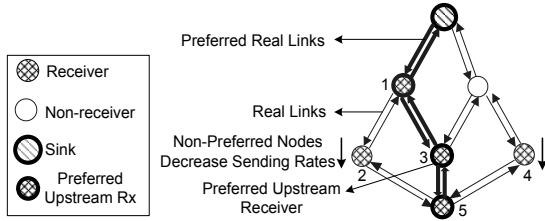


Fig. 3. Fast Reception and Selective Transmission

packets only if it is desired by the downstream receivers.

In CONWISE, if an upstream receiver does not get notification as to whether it is a preferred receiver from any of its downstream receivers over the duration of an epoch, it gradually decreases its sending rate to a minimal value. This procedure reduces the contention in the local neighborhood as receivers along slower paths will be discouraged from forwarding packets. For non-receivers, a similar procedure is adopted with the non-receivers forwarding only if it is along the path from a receiver to the preferred upstream receiver. If there is no notification received at a non-receiver, it reduces its sending and receiving rates similar to a receiver that has not received any notifications. Thus, eventually there is only one preferred path from the sink to a receiver with other receivers and non-receivers decreasing their sending rates to a minimum. Figure 3 shows how the links get pruned in a representative topology. Here, if node 5 chooses node 3 as its preferred upstream receiver, then the sending rates of receivers 2 and 4 is gradually decreased to zero.

IV. THE CONWISE APPROACH

In this section, we describe how the different components described in Section III fit into the CONWISE approach. We present the CONWISE algorithm and describe it works for the following cases: (i) all nodes being receivers, and (ii) a subset of nodes being receivers.

A. Algorithm

There are five basic phases involved at each node in the CONWISE approach. They include the transmit, receive, receive-decision, transmit-decision and notification and update phases. During every epoch, each node upon reception of a packet transmits meta-level information consisting of four-tuple consisting of (i, S, S_{max}, BOT_DEP) , that correspond to the node address of the sender, the current and maximum sending rates and the bottleneck downstream receiver that throttled the sender's rate along with the packet. This is indicated in lines 6-7 in figure 4 and corresponds to the transmit phase. This information is updated to reflect the corresponding values for this node and is piggybacked along with the packet that needs to be forwarded. This information is saved by the downstream nodes corresponding to every upstream receiver during an epoch. Thus, each node maintains a list of upstream neighbors with the above mentioned information. In addition to that, it also stores the number of packets received from that upstream neighbor(k), indicated by Rk .

Nodes may also receive requests, requesting them to be the preferred designated upstream receiver along with the required receiving rate ($reqR$). This information is transmitted to the preferred upstream receiver, m , as a two-tuple $(i, reqR)$, where i is the downstream node id and $reqR$ is the required receiving rate. This information can again be piggybacked along with the data packets. The two-tuples sent by the downstream receivers is used to determine the upstream receiver's sending rate. The preferred upstream receiver checks if the packet was meant for it by checking with m , and stores the required receiving rate ($reqR$) information along with the address of the downstream receiver (i). This corresponds to the receive phase in the CONWISE approach (see lines 8-13 in figure 4). Thus, every node maintains a separate structure to maintain a list of upstream receivers and their sending rates as well as a list of downstream dependents and their required receiving rates.

At the end of an epoch, a node determines its sending rate and provides explicit feedback to the upstream node. The sending rate is determined based on the explicit feedbacks received from downstream nodes (lines 21-29 in figure 4), while the feedback rate is computed based on the meta-level information collected during the epoch (lines 14-20 in the figure 4). We will describe these two phases in detail below. Finally, at the end of all these phases, the required receiving rate information for a node is sent to the preferred upstream receiver as a special packet called the DEP-REQUEST($i, reqR$), after which the structures are initialized to reflect the beginning of next epoch (line 30 in the figure 4). Note that, the transmit and receive phases can overlap, while the receive-decision, transmit-decision and the notification occur strictly in that order and only after the expiry of an epoch. The execution sequence is mentioned in lines 31-39 in figure 4. We will now explain the receive-decision and transmit-decision phases in detail.

1) *Receiving Rate Determination*: Upon the expiry of an epoch, a node enters the receive-decision phase. Based on the number of packets received (Rk values) from all the upstream receivers, the preferred upstream receiver is set to the node address(m) corresponding to the one from which the maximum number of packets where received. By choosing the preferred upstream receiver in this fashion, CONWISE allows for fast reception of all the packets in a message at the receivers. The number of packets received is translated to $reqR$ as $reqR = Rm = Rk(m)/epoch$, where, Rm is the normalized receiving rate of the node from which the maximum number of packets was received. If for that upstream node, the current node was the bottleneck dependent and if all the packets that were transmitted during the epoch were received successfully, the $reqR$ is incremented as given by line 17 in figure 4. This allows for a linear increase in case the current node is the bottleneck for the upstream receiver. If for that upstream node, the current node was not the bottleneck dependent, the $reqR$ is retained as is (line 19 in figure 4). The maximum sending rate ($S_{max}m$) is set to the $reqR$ in both of these cases.

```

Variables(i)
1  i: node id, S: send rate, Rm: maximum receiving rate,
2  Rk: number of packets received,
3  Smax: maximum possible send rate,
4  BOT_DEP: identifier of bottleneck dependent node, and
5  reqR: required rate of reception.
Transmit(i)
6  Transmit every data packet with a 4-tuple
7  (i, S, Smax, BOT_DEP)
Receive(i)
8  For every data packet P received with 4-tuple
9  (k, Sk, Smax - k, BOT - DEP - k)
10  Rk++
11  save Sk, Smax_k, BOT_DEP_k
12  For every DEP-REQUEST received
13  save DEP-REQUEST
At epoch end - Receive decision process(i)
14  Pick neighbor m with the maximum Rm
15  If Rm==Sm and BOT_DEP_k==i,
16  then
17  reqR=min(Smax_m,Rm+1)
18  else
19  reqR = Rm
20  Smax = reqR

At epoch end - Transmit decision process(i)
21  If received any DEP-REQUEST
22  then
23  Pick neighbor b with minimum reqR
24  S = reqR
25  BOT_DEP = b
26  Smax = min(Smax,reqR + 1)
27  else
28  S = max(S - 1,1)
29  BOT_DEP = NOBODY
At epoch end - Notification(i)
30  Send DEP-REQUEST(i, reqR) to neighbor m
Execution sequence:
31  Do
32  For every packet P to be sent
33  Transmit(i)
34  For every packet P received
35  Receive(i)
36  While (!(epoch end))
37  Receive-decision-process(i)
38  Transmit -decision-process(i)
39  Notification(i)

```

Fig. 4. CONSISE pseudo-code

2) *Sending Rate Determination*: The notifications sent by the downstream receivers at the end of the previous epoch, is used to determine the sending rate for the current epoch. At the end of the current epoch, a node has the list of *reqR* information along with the downstream node addresses. The information is used to adjust the sending rate as follows:

- If there was no notification received during the epoch, the sending rate and *BOT_DEP* values are set as given in line 28 and 29 in figure 4. In this case, there are no downstream receivers depending on this node and so the sending rate for this node is gradually decreased to zero.
- If there was one or more notifications received, the node with the minimum *reqR* is chosen from the list of notifications (say *b*). The sending rate, maximum sending rate and the bottleneck dependent fields are updated as shown in lines 24-26 in figure 4. By selecting the minimum *reqR*, CONSISE adjusts the sending rate corresponding to the worst case congestion scenario for the set of downstream receivers.

In order to adapt to network dynamics, CONSISE uses a window based averaging where the computed sender_rate value is weighted with the previous value according to relation, $S(n) = \alpha * S(n) + (1 - \alpha) * S(av)$, where, $S(n)$ is the computed sending rate for this epoch and $S(av)$ is weighted average of the sending rate computed at the end of the previous epoch.

3) *Transmit-Decision Phase for Non-receivers*: We now consider the modifications to the algorithm where only a few nodes are receivers of a particular message. The rate adaptation at the receivers are the same as before. However, the receiving rate for non-receivers is different in this case.

The transmit, receive, receive-decision and notification phases are the same as before for non-receivers. The transmit-decision phase is however modified to adjust the receiving rates of these nodes. The receiving rate is modified by resetting the required receiving rate after the end of the transmit-decision process as $reqR = \min(reqR, S)$. In the absence of any congestion in the local channel of the non-receivers, we notice that the required receiving rate (*reqR*) and the sending rate of all non-receivers are the same and is equal to the downstream receiver's required receiving rate (*reqR*). Thus, it provides this notion of a single virtual link between two successive receivers, where the non-receivers act as mere forwarders.

V. PERFORMANCE EVALUATION

This section evaluates the performance of CONSISE both in the absence and presence of strict reliability semantics. In the latter case, evaluate it for one or more sinks for the following two cases: (i) when all the nodes are receivers of a particular sink from a designated sink; and (ii) when there are a subset of nodes that are receivers.

A. Simulation Environment

The NS2 simulator [5] is used for all evaluations. The environment for the case when there is no strict reliability semantics consists of 400 sensor nodes and is similar to that used in general sensor networks: (a) the first 100 nodes are placed in a grid to ensure connectivity while the remaining 300 nodes are randomly deployed in 600m X 600m square area, and the sink node is located at the center of one of the edges of the square; (b) transmission range of each node is 67m [6]; (c) channel bit rate is 1 Mbps; and (d) each message

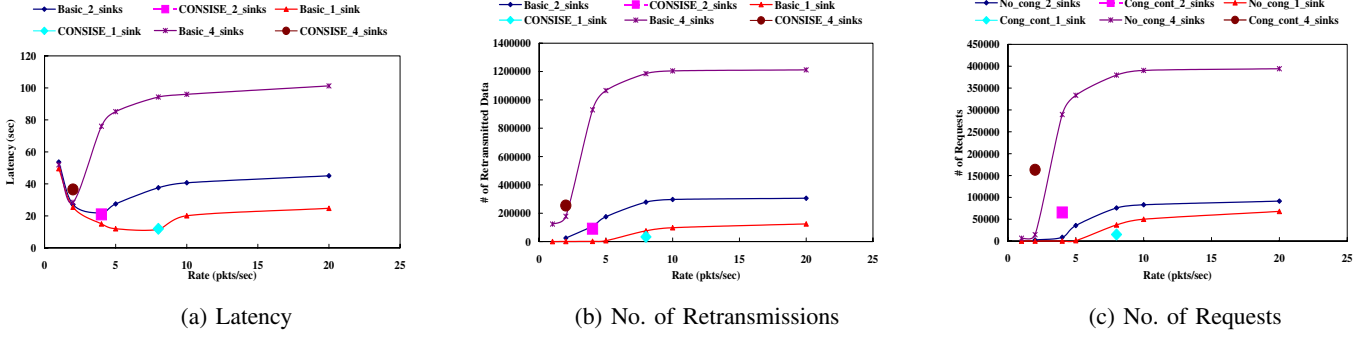


Fig. 5. Comparison of Basic and CONSISE schemes for All Receiver Case

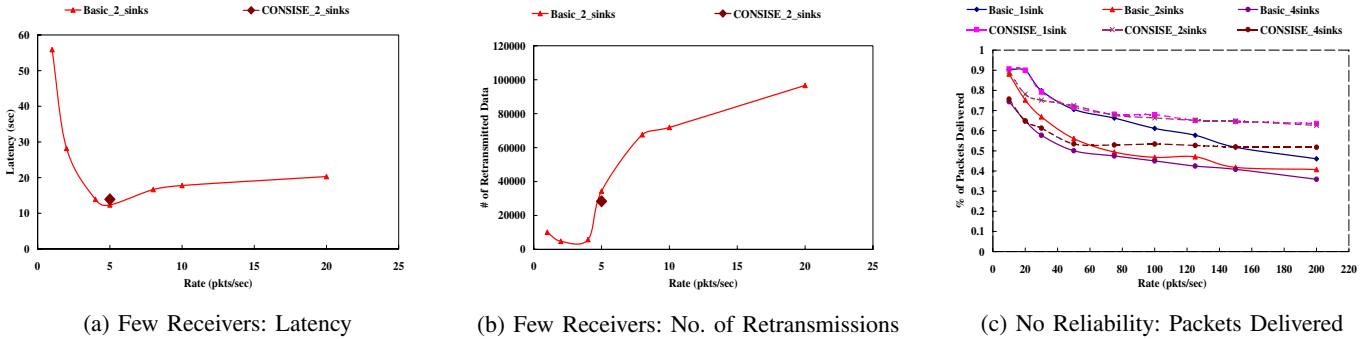


Fig. 6. Results for Few Receiver Case and No Reliability Case

consists of 100 packets of size 1 Kbyte each. The number of sinks is chosen from 1, 2 or 4 sinks.

The environment for the scenarios with strict reliability semantics comprises of 800 nodes. Depending on whether there are 1, 2 or 4 sinks, the number of nodes for each of these sinks are assigned to be 800, 400 and 200 respectively. Among these nodes, the first 100 nodes are assigned in a grid to ensure connectivity. The remaining 700, 300 and 100 nodes for each of the sinks in the 1, 2 and 4 sink case respectively are randomly deployed in 600mx600m square area with the sinks being located on the edge of the squares on opposite ends. The channel rate and the transmission range are the same as before while the message consists of 50 packets and is transmitted by all the sinks simultaneously.

We use CSMA/CA and flooding as the MAC and routing protocols respectively. The flooding scheme is coupled with an unicast loss recovery scheme for scenarios with strict reliability semantics. The value of the parameter α was set to 0.8 to accommodate network dynamics while mitigating the fluctuations in the sending rate. We choose a fixed loss rate of 5% to emulate the random wireless losses in the network.

B. All Receivers with Strict Reliability Semantics

Figures 5 (a), (b), (c) show the latency, total number of retransmissions and the number of requests for CONSISE for different number of sinks. We have compared CONSISE with a simple out-of-sequence with NACK based local loss recovery and no congestion control (henceforth referred to as basic scheme).

1) *Latency*: The latency as a function of the converged sending rate is presented in figure 5 (a) with and without the proposed congestion control implemented for 1, 2 and 4 sinks. CONSISE is able to mitigate the effects of congestion significantly better for all three values of sinks as it is able to adjust the sending rate to the available bandwidth, incurring minimal losses. For all three values of sinks, the sending rate in the scheme with CONSISE is brought down to the rate corresponding to the minimum latency and hence reduces to a single point.

2) *Retransmitted Data Sent*: Figure 5(b) shows the number of retransmitted data sent by CONSISE and the basic scheme for different number of sinks. CONSISE has significantly lesser number of retransmissions than the basic schemes at high sending rates. This is mainly because the proposed congestion control scheme adapts to available network bandwidth without overwhelming the network. The rate adaptation mechanism pro-actively prevents any retransmission unless it is sure that the local channel can support the transmission without a loss.

3) *Number of Requests Sent*: The number of requests sent for both the schemes are shown in figure 5(c). We see that the maximum number of requests for the scheme with CONSISE is significantly smaller than the basic scheme. This is again due to the proactive available bandwidth estimation mitigating congestion related losses. This reduces the number of losses and consequently the number of requests for retransmissions.

C. Few Receivers with Strict Reliability Semantics

Figures 6 (a), (b) show the latency and total number of retransmissions for CONSISE for 2 sinks for the case when there are only 200 receivers among 800.

1) *Latency*: The latency as a function of the converged sending rate is presented in figure 6 (a) with and without CONSISE. CONSISE is able to mitigate the effects of congestion significantly better when the sink sending rate increases. This is because of the contention-region based approach used in CONSISE, where the sending rate is adjusted at the upstream receiver in the event of some congestion at the downstream receiver even though they are not 1-hop neighbors. The intermediate nodes act as just mere forwarders giving the impression of a single *virtual link*.

2) *Retransmitted Data Sent*: Figure 6(b) shows the number of retransmitted data sent by CONSISE scheme and basic scheme. We see that CONSISE has significantly lesser number of retransmissions than the basic schemes. The virtual link concept coupled with the contention-based congestion control between pair of adjacent receivers reduces the number of congestion related losses in the network and consequently the number of retransmissions.

D. No Reliability Semantics

Figure 6 (c) shows the percentage of packets received when there is *no loss recovery* for both the simple flooding scheme and the simple flooding with CONSISE for congestion control as a function of the sending rate for different number of sinks. The total number of packets was calculated as the product of the number of nodes and the number of packets in a message. CONSISE has a more gradual drop when compared with the basic scheme and stabilizes at about 65% with increasing node density. The reason for the steady drop in the absence of any congestion control scheme is the increased contention and collisions at higher sending rate. At very high sending rates, nodes tend to transmit at a rate that exceeds the local capacity of the channel leading to losses. With CONSISE, the effect is mitigated significantly.

VI. RELATED WORKS

To address congestion, researchers have proposed a few related mechanisms in the context of: (i) Sensor networks and (ii) Efficient flooding in ad-hoc networks.

A. Congestion Control in WSNs

Congestion control in sensor networks in the downstream direction has not been studied to any reasonable extent thus far. In fact, the need for congestion avoidance in sensor networks was identified only recently in the context of sensor networks [7]. Most of the works addressing congestion in sensor networks are in the upstream direction, with the exception of [8]. PSFQ [8] is a transport layer protocol that addresses reliability in sensor networks. The key idea in PSFQ is to distribute the data from a source node by transmitting data at a relatively slow speed ("Pump Slowly"), but allowing nodes that experience losses to recover the missing packets

from immediate neighbors aggressively ("Fetch Quickly"). However, PSFQ was designed for providing reliability and requires fine tuning of the parameters in order to prevent congestion. ESRT [2] improves the upstream communication reliability in the network by doing upstream congestion control and is not suitable for downstream congestion control. CODA [3] addresses congestion due to transport of event data. If the scheme is adopted as is for downstream traffic, when the sending rate is increased beyond a certain value, ACK feedbacks will be requested from all the sensors by the sink. This could potentially lead to generation of a hotspot near the sink.

B. Efficient Flooding

Several works have been proposed to address to perform efficient flooding in multi-hop wireless networks [9], [10]. [10] classifies some of these approaches as probability-based, area-based, neighbor knowledge based schemes. While such approaches improve the successful delivery rate of messages [10] and address global congestion, they still do not provide any explicit mechanisms to address local congestion.

VII. CONCLUSIONS

In this paper, we have proposed a new congestion control approach in the downstream direction for sensor networks, called CONSISE. We have identified the different reasons for congestion, the challenges in doing downstream congestion control and addressed them in the design of the CONSISE. We have shown through ns2 based simulations that CONSISE performs significantly better than a basic scheme, which does not provide any congestion control.

ACKNOWLEDGMENT

This work was supported in part by NSF grants ANI-0117840, ITR-0313005, ECS-0225497, ECS-0428329, and the Georgia Tech Broadband Institute.

REFERENCES

- [1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," in *IEEE Communications Magazine*, Aug. 2002.
- [2] Y. Sankarasubramaniam, O.B. Akan, and I.F. Akyildiz, "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks," in *ACM MOBIHOC*, 2003.
- [3] C.Y. Wan, S.B. Eisenman, and A.T. Campbell, "CODA: Congestion Detection and Avoidance in Sensor Networks," in *SenSys*, Nov 2003.
- [4] Seung-Jong Park, Ramanuja Vedantham, Raghupathy Sivakumar, and Ian Akyildiz, "A Scalable Approach for Reliable Downstream Data Delivery in Wireless Sensor Networks," in *ACM MOBIHOC*, May 2004.
- [5] "The network simulator - ns2," Available from the following website: <http://www.isi.edu/nsnam/ns>.
- [6] Andreas Savvides and Mani B. Srivastava, "A Distributed Computation Platform for Wireless Embedded Sensing," in *Proc. of ICCD*, 2002.
- [7] S. Tilak, N.B. Abu-Ghazaleh, and W. Heinzelman, "Infrastructure Tradeoffs for Sensor Networks," in *WSNA*, Sep 2002, pp. 49–58.
- [8] C.-Y. Wan, A. Campbell, and L. Krishnamurthy, "PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks," in *WSNA*, Sept. 2002.
- [9] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu, "The Broadcast Storm Problem in a Mobile ad hoc Network," in *ACM MOBICOM*, 1999, pp. 151–162.
- [10] Brad Williams and Tracy Camp, "Comparison of Broadcasting Techniques for Mobile Adhoc Networks," in *ACM MOBIHOC*, June 2002.