Accurate and Energy-efficient Congestion Level Measurement in Ad Hoc Networks

Jaewon Kang
Computer Science
Rutgers University
jwkang@cs.rutgers.edu

Yanyong Zhang
Electrical & Computer Engineering
Rutgers University
yyzhang@winlab.rutgers.edu

Badri Nath
Computer Science
Rutgers University
badri@cs.rutgers.edu

***Abstract*— Congestion in ad hoc networks not only degrades throughput, but also wastes the scarce energy due to a large number of retransmissions and packet drops. For efficient congestion control, an accurate and timely estimation of resource demands by measuring the network congestion level is necessary. Unlike the wired networks, congestion level measurement in ad hoc networks is more difficult due to time-variant channel capacity, contention among neighboring nodes, and non-deterministic node scheduling.**

In this paper, we propose a new congestion detection mechanism that quantifies the congestion level *accurately* and *energy-efficiently* at both a node-level (implemented at the MAC layer) and a flow-level (implemented at the routing layer) in ad hoc networks. For accurate congestion measurement, a set of metrics that decouple the measurement from various MAC protocol characteristics is defined. For energy-efficient congestion measurement, an asynchronous channel loading measurement scheme called *Lazy Measurement,* which emulates synchronous measurement by using virtual channel sampling, is incorporated into the proposed scheme. Simulation results show the proposed mechanism significantly cut down the energy needed to accurately measure congestion while maintaining high level of accuracy needed for timely congestion control.

I. Introduction

Congestion in data networks happens when too many incoming packets are contending for limited shared resources such as queue buffers and outgoing bandwidth. During congestion, packets experience large delay or even get dropped due to queue overflow, resulting in throughput degradation. In ad hoc networks, congestion occurs as well whenever the incoming traffic volume exceeds available resources. Compared to wired network, however, we observe a much higher packet drop rate, which is caused by not only queue overflow, but also collisions at the receiving node(s). In addition, these packet drops have more adverse impact on ad hoc networks because the energy consumed to forward this packet from its source to the current location will be wasted. Energy in each node is the most scarce resource in ad hoc networks, and sometimes it may not be replenished.

Hence, the fact that congestion in ad hoc networks is more likely to lead to packet drops and that these packet drops have a serious impact on energy efficiency, we must take remedial actions as soon as congestion rises before the problem deteriorates. A number of congestion-controlling solutions have been proposed in the literature [1], [2], [3]. In order for these solutions to effectively limit packet and energy losses caused by congestion, however, one of the most critical factors is the ability to accurately and thereby timely detect congestion in the network. A common strategy for congestion detection is to monitor several metrics to identify "symptoms" associated with congestion.

The congestion measurement in ad hoc networks, however, is harder than in wired or other wireless networks due to *collision*, *contention*, and *interference* as explained below.

- During congestion, packets are frequently collided at the receiving node. When a packet is corrupted due to collision, it is not only discarded immediately at the receiving node, but also discarded at the sending node after several unsuccessful retransmissions. Therefore, the reduced queue length due to the dropped packets by collisions does not necessarily mean that the node has more bandwidth or congestion is alleviated. This makes the queue-based congestion measurement schemes, which are frequently used in wired networks, such as RED [4] unfit for ad hoc networks.
- Contention among neighboring nodes to acquire a chance to send a packet makes a node's outgoing channel capacity *time-variant*. The fluctuation of a node's available outgoing channel capacity might be enlarged by the non-deterministic distributed node scheduling. This makes the node's congestion level also fluctuating even with constant incoming traffic rate and unpredictable especially during congestion. This implies the incoming traffic volume alone is not a good indicator of the congestion level.
- A node's outgoing channel capacity is affected by the neighboring nodes' interference. When a node is placed within the radio range of a congested node without sharing the same routing path with the node, the node's congestion level is also high even when its incoming traffic volume is low. This requires each node to measure the channel loading around itself.

In this paper, we propose a new congestion detection mechanism which accurately quantifies the degree of congestion at a node-level (implemented at the MAC layer). We have shown that this mechanism is able to capture a node's congestion level more accurately. Furthermore, it can decouple the congestion measurement from various MAC protocol characteristics such

as link reliability and buffer capacity, so that it can work regardless of the MAC protocols that are employed by the networks. Based on the node-level congestion measurement, a flow-level congestion measurement is implemented at the routing layer, providing the overall resource demand/supply picture of the flow.

A good congestion detection scheme should not only be accurate, but it should be energy-efficient as well, especially because energy is the most constrained resource for a wireless device. Our proposed scheme addresses this concern by incorporating an asynchronous channel loading measuring technique called *Lazy Measurement*. The Lazy Measurement technique samples the channel loading level in an asynchronous fashion, while maintaining the same level of accuracy as synchronous measurements, which can significantly cut down the energy needed to do this. The main idea for this asynchronous technique is to emulate synchronous measurement using *virtual channel sampling*.

The rest of the paper is organized as follows. In Section II, we propose the two-level congestion measurement scheme for accurate congestion measurement. In Section III, Lazy Measurement is described for energy-efficient congestion measurement. Section IV concludes the paper.

II. Accurate Congestion Level Measurement

In this section, the two-level congestion measurement scheme is explained. The node-level congestion measurement serves as a basis for the flow-level congestion measurement.

A. Node-Level Congestion Measurement

Per-node congestion depicts the local congestion level around that node. When its local congestion level is high, the node may not be able to transmit packets successfully. Depending on the MAC protocol, unsuccessful transmission can lead to a high queue buildup, a high packet drop rate, or a high channel loading. The reason for a sudden congestion level increase is due to the increase of the incoming traffic rate, not necessarily only to this particular node, but to other nodes within its radio range as well. As a result, these statistics (e.g., buffer utilization, channel loading, packet drop rate, incoming traffic rate, etc) can be used to measure the node-level congestion.

Several earlier studies have suggested using a subset of these statistics to measure the local congestion level. For example, ESRT [2] uses buffer utilization alone; CODA [1] uses channel loading and buffer utilization. While each metric captures certain aspect(s) of the congestion level, we find that a combination of (at least) the following three metrics: *channel loading*, *packet drop rate*, and *buffer utilization* is necessary to decouple the congestion level measurement from various MAC protocol characteristics such as link reliability and buffer capacity, thereby to provide an accurate congestion level regardless of the deployed MAC protocol.

Before we present our method of measuring the node-level congestion, we first discuss the impact of various MAC protocol characteristics on the relevance of different congestion level measurement metrics. We perform a simulation based study using ns-2. The network configuration is shown in Figure 1 in which node 0 and 1 (sources) try to deliver data to node 6 and 7 (destinations), respectively. We use IEEE 802.11 DCF (Distributed Coordination Function) [5] without RTS/CTS as the MAC protocol. Simulations with RTS/CTS have similar results. During the time period between 4 and 5 (sec), the sources increase their transmission rates, which will create a hot spot around node 4. The resulting channel loading, packet drop rate, and buffer utilization at node 2 when varying the retransmissions counts for the collided packets and buffer capacities are summarized in Figure 2[1]. For the sake of simplicity, we do not differentiate the packet drops due to collision from the packet drops due to buffer overflow.

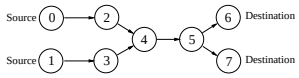


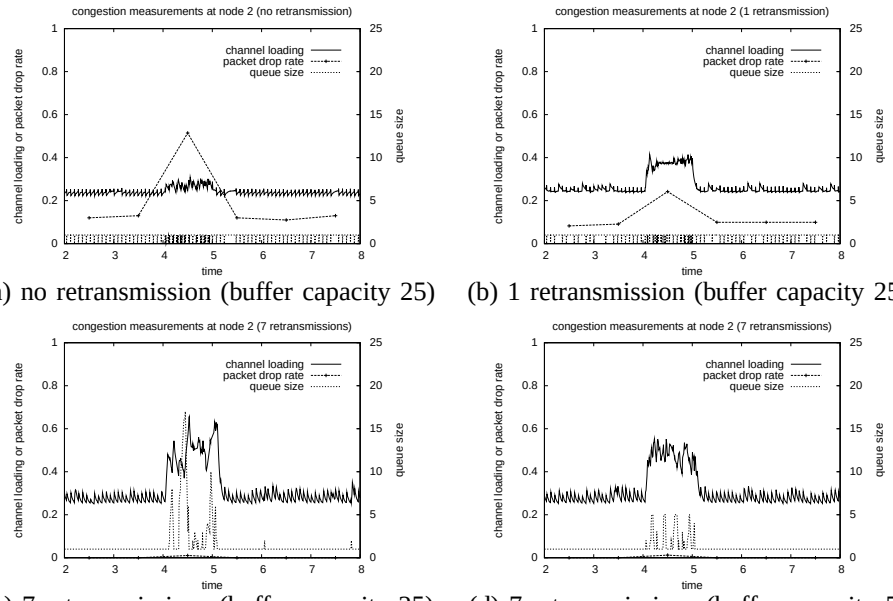
Fig. 1. A simple ad hoc network. Only the nodes connected by an arrow are within each other's radio range.

Figure 2(a) simulates the scenario in which collided packets are simply lost without retransmission. We observe that the packet drop rate alone is sufficient to reflect the congestion level while the other two metrics stay constant even though the congestion has occurred. This observation can be explained as follows: despite the high packet arrival rate, these packets will not increase the channel loading and buffer utilization because they leave the node's queue *quickly* because of collision and no retransmission after the collision.

Figure 2(b) demonstrates that buffer occupancy cannot be used to capture the congestion level in some cases. Specifically, it simulates the scenario where the queue capacity is reasonably large (25) but the retransmission count is low (1 in this case). In such scenarios, channel loading and/or packet drop rate are both good metrics.

While packet drop rate can be quite effective when the retransmission count is small (i.e., low link reliability), it becomes not as useful when the retransmission count increases (i.e., high link reliability). Figure 2(c) simulates the scenario with both large buffer capacity and high retransmission count (7). Due to retransmissions and large buffer capacity, packets are rarely dropped even during congestion period with a high collision rate. As a result, the packet drop rate keeps constant despite the congestion. On the other hand, frequent collisions and retransmission can significantly increase the channel loading. Buffer utilization will as well dramatically increase since a packet will not leave the buffer until its transmission is successful or its retransmission count has expired. Hence, channel loading and buffer occupancy can be used to accurately reflect the congestion level in such scenarios, shown in Figure 2(c). When packets start to be dropped due to buffer overflow, the packet drop rate can also be a congestion metric, but congestion is already worsened

[1]We use *buffer capacity* to denote the total number of packets the buffer can hold, and *buffer utilization*, *buffer occupancy*, or *queue size* to denote how many packets are currently occupying the buffer.



(a) no retransmission (buffer capacity 25) (b) 1 retransmission (buffer capacity 25)

(c) 7 retransmissions (buffer capacity 25) (d) 7 retransmissions (buffer capacity 5)

Fig. 2. Measurements of channel loading, packet drop rate, and buffer occupancy at node 2 when varying the retransmission counts and buffer capacity

due to the large queue size, therefore can be detected before the packet drop rate gets high.

After looking at the impact of link reliability, we next study the impact of buffer capacity on the choice of metrics. In fact, buffer occupancy is usually used for congestion measurement simply because it is easily measured by reading the current queue size. Figure 2(d) simulates the scenario where the buffer can only hold 5 packets and the retransmission count is 7. In this case, a high buffer utilization does not necessarily mean that there is a congestion because its capacity is not large enough. At the same time, a small buffer leads to more packet drops due to buffer overflow than that due to collision (especially when link reliability is high). However, in Figure 2(d), the incoming traffic throughout this simulation is not high enough to create frequent queue overflows, so that we can still notice a low buffer occupancy during time 4-5. For the same reason, the packet drop rate only marginally increases during time 4-5. Please note that packet drop rate will start showing effect only when the buffer overflows. On the other hand, channel loading increases significantly during time 4-5 due to the traffic increase and high retransmission count. Therefore, it shows that channel loading can quantify the level of *incipient* congestion before the buffer builds up or packets start to be dropped. Another good example of requiring channel loading as a congestion metric is when nodes are near the hotspot but not on the routing path. Since no traffic traverse through these nodes, their buffer utilization and packet drop rate is 0. However, their channel loading is significantly high.

Unlike traditional networks, nodes in the ad hoc network exhibit a high degree of heterogeneity in terms of both hardware and software configurations. For example, the radio range, maximum retransmission counts, buffer capacity might be different across the nodes. Even for the same hardware and software configurations, it is non trivial to find the right metrics that can accurately reflect the congestion level across different congestion scenarios.

In order to efficiently deal with the aforementioned heterogeneity and various congestion scenarios, a combination of all three metrics is needed to draw a correct conclusion. Congestion measurement in wired networks can be considered as a special case from the entire design space of our congestion detection policy.

B. Flow-Level Congestion Measurement

The traffic from the originating node is bottlenecked by the most congested node among the nodes along the flow's routing path towards the destination node. For end-to-end congestion control schemes such as TCP, the flow-level congestion is of great importance because the congestion level at a single node does not depict the overall resource demand/supply picture. We build this flow-level congestion measurement based on the node-level congestion measurement discussed in Section II-A.

The challenge that rises in designing the flow-level congestion measurement lies in the tradeoff between accurate measurement and energy efficiency. The basic idea is that every node along the routing path should forward their congestion measurements to the destination so that the application at the destination learns the degree of flow-level congestion. However, if every node sends its measurement periodically, it will incur a considerable amount of traffic and energy consumption. In this paper, we try to balance between energy consumption and accurate congestion detection by the following two optimizations: (1) a node starts sending only when its congestion measurement is above a threshold by embedding its congestion level into the header of a data packet; and (2) as soon as a node receives a data packet with a

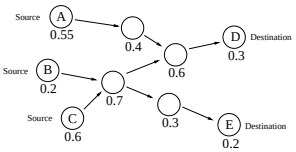


Fig. 3. Forwarding the highest congestion level of a flow to destination

per-flow congestion level, it compares its current congestion measurement with the per-flow congestion level included in the packet header, and updates the per-flow congestion level in the packet header to its own congestion measurement only when its per-node congestion level is greater than the existing per-flow congestion level.

Figure 3 shows a snapshot which includes the congestion levels at each node. In this example, we assume the threshold above which a node will start sending its congestion measurement is set to 0.5. When destination node D receives a data packet with a flow congestion level from source node B, the flow congestion level recorded in the header of the data packet is 0.7 while the flow congestion level from source node A is 0.6.

Using this flow-level congestion information, the destination node can initiate any type of flow-based (or end-to-end) congestion control. This type of congestion notification towards the destination node by marking packet header was discussed in ESRT [2]. In ESRT, however, since the destination node will only get 1-bit congestion information, it cannot perform a fine-grained congestion control according to different congestion levels. In our scheme, however, the destination node as well as intermediate nodes can initiate any type of end-to-end or hop-by-hop *congestion avoidance* scheme before congestion even happens by studying the trend of each flow's congestion level. In CODA, flow-based rate control is performed by the destination node by monitoring the packet arrival rate. Compared to the bookkeeping overhead for each flow in CODA, the flow-level congestion measurement in the packet header gives all the nodes along the path an accurate flow congestion level of the upstream segment of the flow.

III. Energy-efficient Congestion Level Measurement

As mentioned earlier, in ad hoc networks, we cannot afford to measure all three metrics continuously because it will cause a considerable energy consumption, which will reduce the network lifetime. In this section, we discuss our methods of measuring these three metrics which saves energy and computational overheads.

A. Buffer Occupancy Measurement

In order to cut down the overhead involved in measuring the buffer occupancy, we only perform the action when a packet is inserted to or dropped from the buffer. This simple method automatically adapts its frequency according to the incoming traffic volume. When the packet arrival rate is low, the buffer is measured less frequently; as the packet arrival rate increases, the buffer is measured more frequently. It can avoid unnecessary measurements when not needed while making sure an up-to-date congestion level to be reported timely during congestion.

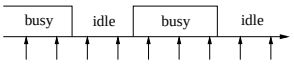


Fig. 4. Channel loading measurement using fixed-rate sampling

B. Packet Drop Rate Measurement

To measure the packet drop rate, a fixed period called *epoch* is defined. During each epoch, the statistics on the number of packet arrivals and packet drops are maintained. To save computational overhead, i.e. energy, the packet drop rate is calculated when a packet is newly enqueued or dropped either by queue overflow or by collisions in each epoch.

C. Channel Loading Measurement

1) Fixed-rate Channel Loading Measurement: Measuring the channel loading is much more challenging than the other two metrics because we have to turn on the radio to do so. As a result, it is critical to consume as little energy as possible to measure the channel loading as well as to report accurate and timely measurements.

A common technique to measure the channel loading is to sample the channel periodically as shown in Figure 4 and then use an exponentially weighted moving average to smooth the measured channel loading. Let C_i^{avg} be the average channel loading after the ith sampling and $C_{i+1}^{sampled}$ be the $(i+1)$th sampled value, where $C_{i+1}^{sampled}$ is either 0 or 1 depending on the channel is idle or busy when sampled. As a result, the average channel loading at i+1 is calculated as follows.

$$C_{i+1}^{avg} = (1-\alpha) * C_i^{avg} + \alpha * C_{i+1}^{sampled} \quad (1)$$

, where α is a weight for the recent sampled value. Ideally, one would like to sample continuously (or frequently) to present the up-to-date average value. However, this is not practical because sampling consumes energy. Determining when to sample thus becomes an important research question.

2) Lazy Measurement: A node usually alternates between two states: active (with radio on) and doze (with radio off). When a node is active, its perceived wireless channel condition goes through alternating idle and busy periods depending on the channel activities of itself and its neighbors.

Earlier work [1], [2], [6] has suggested using fixed-rate sampling to measure the channel loading, which has been illustrated in Figure 4. While sampling at a fixed rate is a viable solution for energy-rich environments, we argue that it is neither efficient nor necessary for ad hoc networks. In Figure 4, to increase the accuracy of channel loading, the sampling period should be shortened while the sampling period should be increased to save energy. Instead, we propose an asynchronous *event-driven* approach, referred to as *Lazy Measurement*, which guarantees both accuracy and energy efficiency.

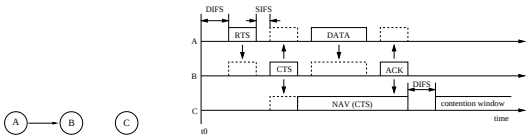


Fig. 5. The channel activity of 802.11 MAC protocol when node A sends a packet to node B. Node A and C are not within each other's radio range.

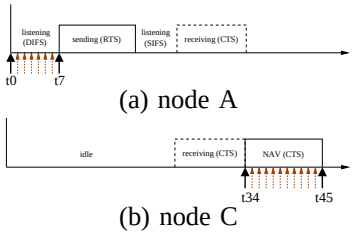


Fig. 6. Lazy channel loading measurement in an event-driven fashion

As mentioned earlier, a node can be in either an active or a doze state. Since our primary interest is in monitoring the channel loading when the congestion is about to happen or when the congestion has already happened (in either case, a node rarely enters into the doze state), we do not make any measurements during the doze state. This can cut down sampling energy significantly because otherwise one needs to turn on the node's radio and perform measurement.

The basic idea of Lazy Measurement is to sample the channel activity at the end of an idle or a busy period. Let's assume that the average channel loading at the beginning of an idle or busy period is C_0^{avg}. At the end of this period, the fixed-rate samplings called *virtual channel sampling* are emulated at the same time. This is illustrated in Figure 6(a) and (b) when node A is sending a single packet to node B while node C is overhearing node B's activity as shown in Figure 5. The virtual channel samplings are represented in dotted arrows. If an active or busy period last for n virtual sampling intervals, then the average channel loading at the end of the period is represented as follows. α is the weight of an exponential weighted moving average.

$$C_n^{avg} = \begin{cases} (1-\alpha)^n C_0^{avg} & (idle\ channel) \\ (1-\alpha)^n (C_0^{avg} - 1) + 1 & (busy\ channel) \end{cases} \quad (2)$$

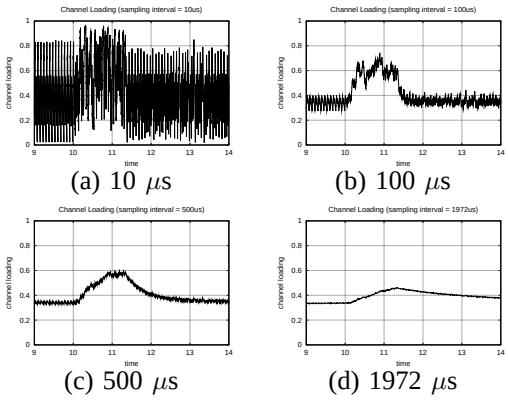
This scheme is sufficient when each of the idle or busy periods is short, in which case it is similar to the fixed-rate sampling. However, one may question that Lazy Measurement may break for a long idle or busy period since the average channel loading is not updated until the end of this relatively long period. We have studied this issue carefully and concluded that *Lazy Measurement is not only sufficient, but also more accurate for large periods in most of the cases*, which is explained below for a long idle or busy period and verified by simulations later in this section.

Even during congestion, an idle period can be long due to the long backoff windows when the channel contention is high. If we frequently sample the channel during this backoff window, the sampled value is 0, so that the calculated channel loading becomes lower. A lower channel loading under high contention might be useful for some scenarios, but it is inaccurate for the sake of congestion detection. On the other hand, if we use use Lazy Measurement, the channel loading value does not change during the backoff period (because the channel loading is updated at the end of the backoff period), which is very helpful to detect the congestion and further apply remedial actions. If the channel is monitored during backoff such as in IEEE 802.11, which stops decrementing the backoff counter when the channel gets busy, the average channel loading is simply updated when the channel gets busy again. In addition, if the packet size is small, the busy period caused by packet transmissions will not be long. Frequently, a large packet is fragmented into smaller frames to increase channel utilization [5].

However, there is a scenario where Lazy Measurement is inaccurate due to its deferred sampling: the transition from the congested period to the dormant period. As soon as a network burst passes, the network traffic becomes low, and the channel enters a long idle period. Please note that the average congestion level before this long idle period is pretty high. In this case, we would like to frequently sample during this idle period so that the average congestion level can decrease quickly to reflect the actual network behavior. Also, we would like to point out that this type of long idle period is much longer than the backoff windows under contention discussed above. In order to tailor our scheme for this case, we set an idle timer, which is longer than the usual backoff window duration. As soon as the timer expires, we sample the channel and update the average congestion level. One enhancement to reduce the overhead of the idle timer is to set the timer duration short when its congestion level is high, so that the dormant state can be quickly found. When the node's congestion level is low, the duration of the idle timer can be extended in an additive manner (or exponentially to aggressively reduce the overhead) with some upperbound.

Another advantage of Lazy Measurement is that when a node knows the duration of busy channel in advance such as NAV in 802.11 MAC protocol, turning off its radio does not affect the channel loading measurement. As shown in Fig. 6(b), node C can turn off its radio during deferring and turn on its radio at the end of the deferring, assuming the channel condition is busy during the deferring.

For the channel loading measurement after the doze mode, right before a node enters into the doze mode, a node updates its channel loading level and saves it with current time for later use. During the doze mode, radio is not turned on again for the purpose of channel loading measurement. When the node wakes up by turning on its radio after the sleep timer expires, the current time and the time of last channel loading measurement are compared. If the difference is less than some threshold, the last recorded average channel loading is used as its initial average channel loading, hoping that the channel loading does not change a lot. If the difference is greater than



(a) 10 μs (b) 100 μs

(c) 500 μs (d) 1972 μs

Fig. 7. Channel loading measurements by Lazy Measurement with respect to various virtual sampling intervals

the threshold, the node waits for some period for its neighbor nodes to notify their channel loading levels.

To obtain the channel loading levels from its neighbors, a node overhears the packets sent by its neighbors. For this, each node periodically embeds its channel loading level into the header of a data packet. If a node overhears several channel loadings during the period, they are averaged to be its initial channel loading assuming the channel loadings in the same area are *correlated*. If no channel loading is overheard during this period, it sets its initial channel loading to 0. The advantage of overhearing the neighbors' channel loading levels is that the nodes in the hotspot are highly likely to overhear the data packets containing their neighbors' channel loading levels since the data traffic is high in the hotspot.

3) Performance of Lazy Channel Loading Measurement: To investigate the accuracy and energy-efficiency of Lazy Measurement, we perform a ns-2 simulation under the same network topology as in Figure 1. IEEE 802.11 DCF with RTS/CTS is used for the underlying MAC protocol. During the time period between 10 and 11 (sec), the sources increase their transmissoin rate, making congestion around node 4. Shorter sampling interval and higher weight value, α in Equation 2 can measure the channel loading level more accurately, but it incurs a large fluctuation. In our simulations, we fix the weight of the sample value (i.e. α in Equation 2) to be 0.001. The observed average real sampling interval of Lazy Measurement at node 2 between 9 and 14 sec is 1.972 ms (i.e. 1972 μs). Given the observed average sampling interval of 1972 μs, we varies the virtual sampling interval to 10μs, 100μs, 500μs, and 1972μs, as shown in Figure 7 (a), (b), (c), and (d), respectively. Figure 7 shows too short sampling interval results in the high fluctuation of the average channel loading level. In this case, the virtual sampling interval of 100μs provides an accurate and smooth channel loading level. As the virtual sampling interval increases, the resulting channel loading measurement does not accurately reflect the congestion level in the network, as shown in Figure 7 (d), in which the observed average sampling interval and the virtual sampling interval are equal.

Our simulations show the channel loading measurements by the fixed-rate sampling scheme are almost identical to the

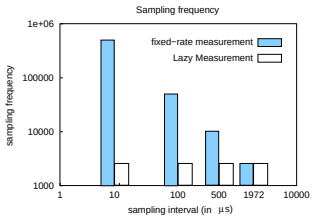


Fig. 8. Sampling frequency with respect to various sampling intervals

measurements by Lazy Measurement. However, to accurately measure the channel loading, the fixed-rate channel sampling requires high sampling rate, thereby consuming more energy. Figure 8 shows the sampling frequencies of sampling intervals 10μs, 100μs, 500μs, and 1972μs. The x and y axes are calibrated in log scales. If we assume the energy in each node is consumed in proportion to the sampling frequency, then Figure 8 also can be an indication of energy consumption. While the sampling frequency of Lazy Measurement is constant since it only changes the virtual sampling interval, the fixed-rate sampling consume a lot of energy to accurately measure the channel loading.

IV. Conclusion and Future Work

In this paper, we propose a two-level congestion detection scheme that provides an accurate node-level and flow-level congestion measurements in an energy-efficient way in ad hoc networks. Simulation results show the node-level congestion measurement, which uses the set of buffer occupancy, packet drop rate, and chanel loading as an indicatoin of congestion, accurately portrays the congestion level by decoupling the measurement from various MAC protocol characteristics. The flow-level congestion measurement based on the node-level congestion measurement provides a fined-grained congestion information in the network. For energy-efficiency, the lazy channel loading measurement saves a lot of energy needed to accurately measure the channel loading while maintaining the same level of accuracy as synchronous measurements.

In ad hoc network, the backoff window size can be another indication of congestion. We plan to further optimize the proposed congestion detection mechanism by investigating various candidate metrics for accurate and energy-efficient congestion measurement in ad hoc networks.

References

[1] C. Wan, S. Eisenman, and A. Campbell, "CODA: Congestion Detection and Avoidance in Sensor Networks," in *ACM SenSys*, 2003.

[2] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz, "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks," in *ACM MobiHoc*, 2003.

[3] Y. Yi and S. Shakkottai, "A Hop-by-hop Congestion Control over a Wireless Multi-hop Network," in *Proceedings of IEEE INFOCOM'04*, March 2004.

[4] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, Aug. 1993.

[5] B. P. Crow, I. Widjaja, J. G. Kim, and P. T. Sakai, "IEEE 802.11 Wireless Local Area Networks," *IEEE Communications Magazine*, Sep. 1997.

[6] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "Infrastructure tradeoffs for sensor networks," *ACM Workshop on Wireless Sensor Networks and Applications (WSNA)*, Sep. 2002.