ARTICLE IN PRESS

Ad Hoc Networks xxx (2008) xxx–xxx

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Locally scheduled packet bursting for data collection in wireless sensor networks

Ren-Shiou Liu *, Kai-Wei Fan, Prasun Sinha

Department of Computer Science and Engineering, The Ohio State University, 395 Dreese Laboratories, 2015 Neil Avenue, Columbus, OH 43210-1277, United States

A R T I C L E I N F O

Article history:
Received 13 March 2008
Received in revised form 9 August 2008
Accepted 15 August 2008
Available online xxxx

Keywords:
MAC Protocol
Wireless sensor network

A B S T R A C T

In wireless sensor networks, the many-to-one data communication pattern induces high collision losses as multiple transmissions cause contention and interference along the paths from sources to the sink. This paper proposes a low-overhead MAC layer solution to address the high contention problem to improve system throughput and reduce energy consumption. Periods of burst transmissions with reduced contention from neighboring nodes are exploited to efficiently clear up backlogged queues and improve the performance of CSMA. Through analytical modeling we characterize the expected performance improvement. Using extensive simulations on *ns-2* and experiments on the 49-node sensor network testbed (*Kansei*) running TinyOS, we show that the proposed scheme can increase the throughput by up to a factor of four.

1. Introduction

Energy and channel capacity are two critical resources in wireless sensor networks. When a large number of nodes start reporting data, sensor networks easily get overwhelmed by high contention and interference along adjacent multihop routing paths and in the neighborhoods of data collection points such as the sink. This leads to inefficient use of these resources. Various approaches have been proposed to mitigate this problem, such as improved MAC layer designs [1,2] and back-pressure techniques at the link layer [3–6]. In [1], a hybrid TDMA/CSMA approach is proposed to address congestion near the sink. However, it requires specific capabilities only available at the sink. ZMAC [2] is another hybrid TDMA/CSMA based solution, but it requires global time synchronization and distributed slot assignment using the DRAND [7] protocol, which significantly increases the complexity and overhead of the protocol. In addition, the computation of the TDMA schedules is expensive in dynamic environments where the traffic sources change with time. Back-pressure based mechanisms for congestion control [3–6] operate over the MAC layer to maintain the queue size at acceptable levels to avoid queue drops. As these mechanisms are not integrated into the MAC layer where congestion is first observed, their impact on performance improvement is limited.

The convergecast traffic pattern of wireless sensor networks leads to high load at nodes that are aggregating data. We demonstrate this problem by using simulations in a $1000 \times 1000\ \text{m}^2$ network of 1000 nodes. An event with radius 100 m is randomly generated in the network and data is forwarded over the Steiner tree joining the source nodes to the sink. We increase the offered load by increasing the event radius. Fig. 1 shows that increase in the amount of data generated leads to sharp decrease in throughput. This trend is observed for data collection with or without aggregation.

In this paper we seek to design a low-overhead MAC layer solution to address the overload problem in wireless sensor networks. Our solution is based on the observation that throttling sensors' reports to prevent simultaneous transmissions can reduce contention and increase throughput [13,14]. We propose a burst scheduling approach at the MAC layer specifically designed to mitigate the overload

---

* Corresponding author. Tel.: +1 614 688 4617; fax: +1 614 292 2911.
E-mail addresses: rsliu@cse.ohio-state.edu (R.-S. Liu), fank@cse.ohio-state.edu (K.-W. Fan), prasun@cse.ohio-state.edu (P. Sinha).

1570-8705/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2008.08.002

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002

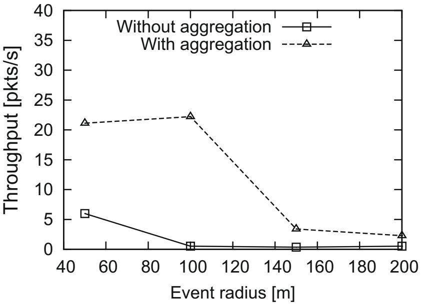


Fig. 1. Convergecast traffic pattern in wireless sensor networks degrades performance of CSMA MAC.

problem. The scheduling overhead is reduced as a burst of packets as opposed to a single packet is scheduled for transmission. If a node observes an increase in its queue backlog, it performs a low-overhead coordination with neighboring nodes to reserve a period for transmitting a packet burst. By alleviating contention during the burst periods, throughput is boosted for transmissions from sources to the sink. In addition, by explicitly addressing backlogged queues, overall queue drop rate decreases and network performance is improved. We make the following contributions in this paper.

- We propose a low overhead packet bursting based approach called ClearBurst, for mitigating overload at any node in the network.
- We analytically model the expected performance gains for representative network scenarios by extending the analysis techniques used in Bianchi's work [8,9].
- We perform extensive evaluation using *ns-2* and show that Clearburst can improve throughput by two folds under high load and it is up to four times more energy efficient than CSMA.
- We present results from experiments on a large-scale indoor testbed based on implementation on TinyOS, and show that ClearBurst can have four times higher throughput and at least 20% more energy efficient than CSMA.

The organization of the rest of the paper is as follows. We contrast our work with related work in Section 2. Sections 3 and 4 present our proposed approach and analytical modeling of the proposed solution. Simulations and experimental results are presented in Sections 5 and 6. Applying ClearBurst in moving event scenarios is explored in Section 7. Section 8 concludes the paper.

2. Related work

2.1. Congestion mitigation at MAC layer

In order to alleviate interference and contention, various TDMA based MAC protocols [19–22] and hybrid CSMA/TDMA techniques [1,2,23] have been proposed. TDMA based approaches [19–22] suffer from high global time-synchronization and slot assignment overheads, which make them unsuitable for scenarios with changing traffic patterns. For example, in dynamic or moving event scenarios, not all the sensors need to report data to the sink. Thus, slots assigned to those sensors which do not have data to report are wasted. To improve channel utilization of traditional TDMA protocols in such scenarios, hybrid schemes have been proposed. ZMAC uses prioritized CSMA based transmissions in TDMA slots. In other words, if an unused slot is detected, other nodes can compete for that empty slot. Though this improves channel utilization, it has been shown in [1] that the performance of ZMAC degrades with time and falls back to that of CSMA due to clock drift. Furthermore, it is difficult for ZMAC to adapt to changing topology as it requires recomputation of slot assignments using the DRAND [7] protocol, which incurs an extra overhead of $O(\Delta)$ per node, where Δ is the maximum number of contending nodes in a neighborhood. Harvest [23] is also a hybrid MAC aiming for minimizing the slot assignment overhead. Harvest only uses four slots in a frame, meaning that only four nodes in a two-hop neighborhood can be assigned a slot. Harvest's slot assignment is randomized. Whoever chooses a slot first wins that slot. Once a node has finished transmitting all its data, it releases its slot and its one hop neighbors who did not get a slot previously can compete for that vacated slot. Although Harvest can complete slot assignment in $O(1)$ time, it still requires global time synchronization. FMAC [1] is a sink-oriented hybrid MAC protocol. It only mitigates congestion near the sink as the authors observe in the experiments that contention is more intensive around the neighborhood of the sink. For this reason, the neighborhood around the sink is named the intensity region in the paper. The sink constantly monitors the traffic from the sensors and dynamically tunes the depth of the intensity region. Nodes in the intensity region use TDMA, while nodes outside the intensity region use CSMA. Scheduling information is periodically computed and broadcast by the sink at a high power to all the nodes in the intensity region. As FMAC relies on the sink to compute TDMA schedules, it does not scale well and cannot quickly react to the onset of the congestion which arises deep in the network. In addition to the aforementioned global time synchronization and slot assignment overheads, none of the above TDMA based protocols addresses the special requirement of convergecast traffic pattern in sensor networks. Nodes performing data aggregation or those that are closer to the sink usually have to forward more traffic. As a result, they should have more bandwidth share of the channel. However, to the best of our knowledge, this issue is not addressed by existing TDMA protocols.

2.2. Congestion mitigation at higher layers

CODA [4], Fusion [5], IFRC [6], and RAIN [3] are hop-by-hop control based strategies for mitigating congestion in sensor networks. In CODA [4], a weighted moving average of channel measurements combined with queue length assessment is used to determine the onset of congestion. Once congestion is detected, a node broadcasts back-pressure messages towards upstream nodes. Nodes that receive back-pressure messages throttle their transmission rates and determine whether to propagate the back-pres-

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002

sure further according to their local congestion assessment. In addition to this open loop hop-by-hop back-pressure mechanism, CODA also adopts a close loop rate regulation scheme. Based on the application requirements and network conditions, the sink controls the rate of the sensors by sending or dropping ACKs. Sensors then adjust their rates using a function, such as AIMD, of the arrival rate of ACK packets. In contrast to CODA, Fusion [5] uses queue length as the only congestion indicator because the authors find, through experimental measurements, queue length is at least as good as channel sampling. Furthermore, Fusion combines hop-by-hop flow control with the ideas of rate limiting and prioritized MAC to control congestion in sensor networks. A node adds a token to its bucket every time it hears N transmissions from its parent, where N is the total number of unique sources routing though the parent. A node is allowed to transmit a packet if and only if its token count is not zero. In order to allow congested nodes to propagate back-pressure quickly, Fusion makes CSMA's contention window size a function of a node's congestion state. If a sensor is congested, its window size is one fourth of that of a non-congested node. This gives a congested node more opportunities to clear packets accumulated in the queue and propagate the back-pressure. Following Fusion, IFRC [6] also uses queue length as the congestion indicator. The difference is that IFRC adopts a variation of the AIMD scheme to control the rate of each node. Multiple queue length thresholds are employed by IFRC. If the queue length exceeds any of the thresholds, a node reduces its rate by half. The distance between kth and $(k+1)$th threshold gets smaller as k becomes larger. In other words, IFRC reduces the rate more aggressively if congestion remains and queue length continues to grow after a rate deduction. On the other hand, if there is no congestion, a node additively increments its rate. In addition to rate control, IFRC shares a node's congestion with its two-hop neighbors by including the rate and queue length information of a node itself and its most congested child in the packet header. Each node sets its rate to the smaller of its own rate and the rates of its neighbors and their children. Unlike IFRC which uses multiple queue length thresholds, RAIN [3] uses a single small queue length threshold (queue length threshold of size 1 is used in the paper) to detect congestion and create a back-pressure. Using a small queue length threshold allows congestion to be detected at an early stage. It also ensures that the back-pressure can be quickly propagated toward the source. The concept of AIMD is also adopted in RAIN, but it is implemented as a light-weight transport protocol called ReTP. ReTP can be light weight because packet losses due to contention or congestion are rare, and it only has to deal with routing failures. In contrast to these higher-layer approaches, ClearBurst addresses congestion at the MAC layer. These approaches can be used to further improve the performance of ClearBurst.

Certain protocols [24,25] designed primarily for reliability also include mechanisms for congestion control. PSFQ [24] is a reliable data dissemination protocol that also attempts to address congestion by controlling the rate of pumping packets into the network. PSFQ uses two timers, $T_{\min}$ and $T_{\max}$, to control the rate of pumping and relaying packets respectively. The source node broadcasts a packet every $T_{\min}$ interval until all the data fragments are sent out. $T_{\min}$ is long enough for neighboring nodes to recover lost fragments while not overwhelming the network. Relay nodes randomly back off for an interval between $T_{\min}$ and $T_{\max}$ before relaying data fragments to prevent collisions. Furthermore, to avoid congestion, the relay of a data fragment will be suppressed if it has been broadcast in the same neighborhood for at least four times. In ESRT [25], sensor nodes monitor their buffer levels and notify the sink of the congestion condition by setting a congestion notification bit in the packet header. The sink, in turn, informs the source nodes to appropriately adjust their reporting rates with the goal of increasing reliability while conserving energy. These higher layer mechanisms are orthogonal to our work and they can be used along with ClearBurst to provide support for reliability and congestion control.

2.3. Architecture for congestion mitigation

In Siphon [26], a few nodes with a high-bandwidth second tier radio interface, called virtual-sinks, are randomly deployed in the network. Whenever congestion is detected, packets are redirected to a nearby virtual sink and then sent to the next virtual sink which is closer to the real sink using the more powerful second-tier radio. The need for a powerful second-tier radio interface makes it a more expensive solution. Moreover, virtual sinks can still suffer from congestion.

3. Design of ClearBurst

To mitigate the overload problem at data collection nodes, we propose *ClearBurst* in the MAC layer to coordinate media access for sensor nodes. ClearBurst has three phases. They are C-node election, congestion detection, and burst scheduling. When an event is detected, the sensor in the event region and closest to the sink is elected as the C-node. A steiner tree connecting all the source nodes and the sink is then constructed. Once the tree is constructed, all the source nodes send data toward the C-node for aggregation. If the application requires high reporting rate, the C-node and its immediate one-hop upstream nodes will soon be overloaded. ClearBurst uses queue length as an overload indicator. When any one of C-node's immediate one-hop upstream node detects overload, ClearBurst uses a small time window to reserve the channel in the two-hop region of the C-node and the node that detects overload. Three dedicated slots are reserved for the node that detects overload, the C-node, and C-node's immediate downstream node, respectively. During these three reserved slots, these three nodes initiate burst transmission in order. This reduces contention and packets can be forwarded out of the intensive contention area toward the sink quickly. As a result, overall throughput is increased and energy wastage due to collision and contention is minimized. After a burst duration ends, all the nodes fall back to the CSMA operation and the next burst transmission is not triggered until overload is detected again. The detailed descriptions of these three phases is provided in the remainder of this section.

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002

3.1. C-node election

First a C-node is elected for a set of sources to act as a data collection point as well as a schedule coordinator. Although a TDMA-based approach can reduce contention, it will also incur high overhead for time synchronization and slot assignment. Using a C-node as the coordinator not only eliminates these overheads but also makes the schedule adaptive to dynamic traffic and unpredictable topological changes. In addition, the C-node can serve as an aggregation point that aggregates raw data packets and reduces the amount of information transmitted in the network.

Many cluster-head election and tree construction algorithms have been proposed and can be adopted. For example, the cluster-head election, tree construction and migration approach described in [10] can be used to elect the C-node and to construct and maintain the tree. However, it is crucial that the C-node is elected quickly and efficiently at the routing layer so that we can respond to emerging congestion instantaneously while minimizing overheads.

The election protocol selects one of the sensors in the event region that is closest to the sink as the C-node and builds a spanning tree rooted at the C-node. This approach has several advantages. First, instead of using the shortest path, forwarding packets to the C-node increases the opportunity of early aggregation. Second, once data is aggregated, they can be forwarded toward the sink by burst transmissions to avoid delay and collision caused by intensive contention around the event region. Third, if the subgraph containing all of the source nodes are connected, this heuristic can compute an optimal data collection tree [11]. Fourth, building a spanning tree rooted at the C-node can be achieved by simple extensions to existing routing protocols such as MintRoute [28].

When an event is detected, all nodes that can detect this event start generating reports at a low rate and forward the packets to the sink using the shortest path. Besides raw data, these low data rate reports include the ID of the node generating the packet, its hop distance to the sink and to its chosen C-node, and the C-node's hop distance to the sink. Initially, every source node chooses itself as the C-node. If a node learns that any of its neighbor's chosen C-node is closer to the sink or is equally close to the sink but with a smaller ID, it chooses that C-node as its C-node, and sets that neighbor as its parent. If more than one downstream nodes choose the same best C-node, the one with stronger signal is picked as its parent. When a node observes that all its neighbors have reached a consensus on the C-node's location, it can start reporting data at the rate specified by the application and the C-node election and tree construction phases end. If the diameter of the subgraph connecting all the source nodes is d, then the time complexity of this C-node election and tree construction process is $O(d)$.

3.2. Congestion detection

When the traffic load is low, the nodes use a CSMA-based protocol to transmit their packets since the CSMA-based approaches perform well in low traffic scenarios. However, when the traffic load is high, its performance drops significantly due to the congestion caused by high contention and collisions. Various congestion indicators have been proposed, such as the number of packets in the queue [3,5,6], the channel utilization [4], and queue length information of neighbor nodes [12]. To minimize the overhead, we adopt queue length as the congestion indicator in ClearBurst as accessing queue length does not require any energy-consuming signalling or overhearing. Furthermore, it has been shown in [5,6] that using queue length as the congestion indicator is as good as channel sampling for convergecast traffic in sensor networks. When the number of packets in the transmission queue exceeds a predefined threshold, ClearBurst steps in and starts coordinating the transmissions. As the bottleneck is likely to happen around the C-node, ClearBurst coordinates the transmissions only for nodes near the C-node to minimize the control overhead. Multiple C-nodes for coordination is possible if these C-nodes are not interfering with each other and we leave it for future work.

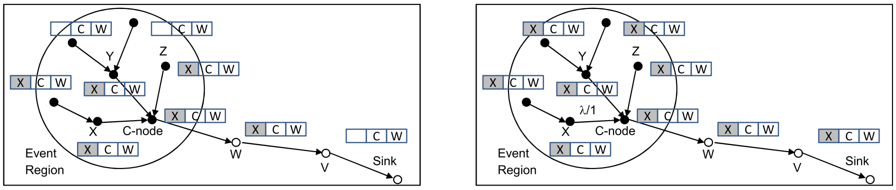
3.3. Burst scheduling

To start the coordination, children nodes of the C-node signal the need for burst transmission by setting the request bit in the data packet header. When the C-node receives a packet with the request bit turned on and if it is not serving any burst transmission, it grants the request by piggybacking the acknowledgement in outgoing data packets. The child node can overhear the data packets and learn that its request has been granted. The child node whose request is granted is called an active node. The request and acknowledgement handshake serves the purpose of reserving the channel for the burst transmission. Because the interference range is usually larger than communication range, this scheduling information needs to be propagated to the nodes that may interfere with the burst transmission as demonstrated in Fig. 2(a).

To make sure that these potential contending nodes and interfering nodes are shut off during burst periods, ClearBurst uses a small time window after the handshake to propagate the scheduling information before starting the burst period. During this small time window, nodes can still access the channel using CSMA, but all the nodes, who have learned the scheduling information by overhearing, help propagate the information by piggybacking it in every outgoing data packet. The number of hops to propagate the information can be controlled by the TTL field in the packet header (TTL of 2 was used in simulations and experiments). When the burst schedule propagation time ends, the node requesting for the burst transmission can start its burst transmission.

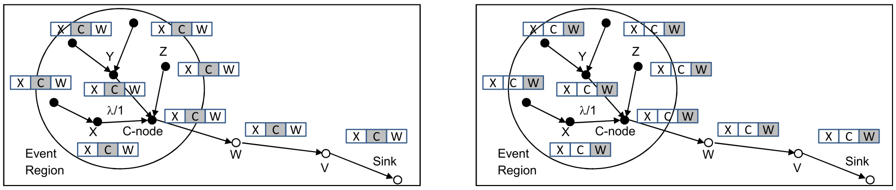
A burst period consists of three slots. As demonstrated in Fig. 2(b)–(d), the first slot of length λ is used by the active node to transmit packets to the C-node. The second and the third slots are used by the C-node and its parent node to forward packets to the sink. Assuming that the C-node aggregates packets with aggregation ratio ρ as shown in Fig. 2(e), the time required to forward the aggregated packets is $\mu = \rho\lambda$. If we do not reserve slots

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002



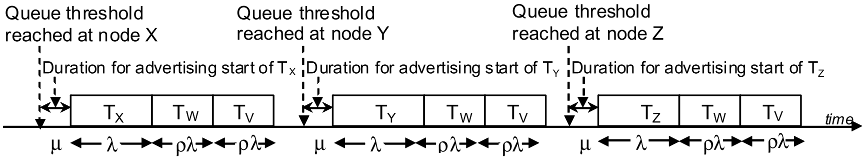
(a) **Step. 1:** Schedule information of burst transmission is propagated in the two-hop range of X and the C-node.

(b) **Step. 2:** When burst schedule starts, X starts sending packets in a burst to the C-node.



(c) **Step. 3:** After X's burst transmission, C-node forwards packets from X to its parent W in a burst.

(d) **Step. 4:** W continues to forward packets to its parent V. After that, packets originally accumulated in X's queue will be in V's queue and they are free of the intensive contention around the event region.



(e) Demonstration of three consecutive burst transmissions from three different nodes X, Y, and Z. Before each burst transmission starts, a small time window μ is used to reserve the channel for the burst transmission. If the burst duration of X, Y and Z is λ and aggregation ratio is ρ, the burst duration of transmission from the C-node will be $\rho\lambda$.

Fig. 2. Demonstration of ClearBurst operation. The time slot assignments are shown along with every node. The gray ox represents the current slot.

for the C-node and its downstream node to forward their packets, when a burst period ends and all the nodes resume their transmissions, they will have very little chance to forward these packets because it has to compete for the channel with other nodes in the interference range. Reserving dedicated slots for the C-node and its parent to transmit their packets in a burst can avoid queue buildup at these nodes.

During a burst period, both the active node and the C-node keep announcing the progress of the burst operation by including the remaining time of the burst period in data packet headers. Any neighboring node who missed the schedule information during the schedule advertisement time window and overhears the scheduling information freezes its transmission immediately. This further minimizes the chance of interference during burst periods.

Due to unpredictable channel conditions and unsynchronized clocks and duty cycles, some nodes may miss the scheduling information of a burst period and still try to access the channel. In order to let the active node dominate its use of the channel under potential interference from its neighbors, the active node uses smaller initial backoff and contention window sizes. This helps suppress unexpected transmissions originating from the neighbors of the active node and the C-node during burst periods. The smaller initial window size also helps to minimize the overhead of initial backoff and improve channel utilization during burst periods.

When the burst period ends, all the nodes go back to the CSMA mode to contend for the channel, and other children nodes of the C-node whose queue lengths exceed the threshold can start requesting another burst transmission.

4. Performance analysis

In this section we analytically derive the throughput and energy efficiency of the ClearBurst protocol.

4.1. Throughput

We adapt Bianchi's work [9] for our analysis. However, in the simulation we found that the "capture" phenomenon has a huge impact on overall throughput when the contention is high. The capture effect is not modeled in [9]. In this section we consider the capture effect and derive the corresponding throughput. First, we need the probability that a node transmits in an idle slot, called the transmission probability τ. The backoff mechanism in MAC layer determines the transmission probability. The default MAC layer in sensor nodes, e.g. Mica2, uses fixed initial and random backoff window sizes. The initial backoff window size is 16. If the channel is busy during the first transmission attempt, nodes randomly backoff with window size 32. To simplify the analysis, we assume that the backoff window size is always W.

Using a discrete time Markov chain with W states as in [9] we can easily show that in the steady state, the transmission probability, τ, is $\frac{2}{W+1}$. With τ, we can derive the probability that a transmission is a success or a collision. Second, we need to know what is the probability that a "capture" will happen if two nodes transmit at the same time. Though it is possible that capture can still happen if three or more nodes transmit at the same time, the probability is relatively small. Therefore, we ignore this case in the analysis.

A "capture" happens if a packet with a stronger signal can be correctly decoded at the receiver in the presence of interference from a weaker signal. In the simulation we use the two-ray ground propagation model; therefore the signal strength is inversely proportional to the square of the distance. Suppose the capture threshold is C_{t}. If nodes a and b transmit a packet to s at the same time with the same transmission power, the packet from node a can be decoded if the distance between s and b is at least $\sqrt{C_{\mathrm{t}}}$ times longer than the distance between s and a, as shown in Fig. 3.

We assume that N nodes are uniformly distributed within a circular region with radius R. All nodes are within interference range of each other and the receiver s is at the center of the circle. If we observe a tagged node whose distance to s is r. If $r < \frac{R}{\sqrt{C_{\mathrm{t}}}}$, only nodes within radius $r \times \sqrt{C_{\mathrm{t}}}$ can collide with its transmission. In a uniformly distributed deployment, there are $N \times \frac{(r\times\sqrt{C_{\mathrm{t}}})^2}{R^2} - 1$ such nodes other than the tagged node. If $r >= \frac{R}{\sqrt{C_{\mathrm{t}}}}$, all the other nodes can collide with the tagged node. Therefore, the probability for a node to successfully transmit a packet is

$$p = \begin{cases} (1-\tau)^{N\times\frac{(r\times\sqrt{C_{\mathrm{t}}})^2}{R^2}-1} & \text{if } r < \frac{R}{\sqrt{C_{\mathrm{t}}}} \\ (1-\tau)^{N-1} & \text{if } r >= \frac{R}{\sqrt{C_{\mathrm{t}}}} \end{cases} \tag{1}$$

Eq. (1) is the conditional probability of a successful transmission given that a sender is at a distance r to its receiver. To derive the marginal success probability, we can integrate p defined in Eq. (1) from $r = 0$ to R, and we get $p_s =$

$$\frac{\int_0^{\frac{R}{\sqrt{C_{\mathrm{t}}}}} 2r \times (1-\tau)^{N\times\frac{(r\times\sqrt{C_{\mathrm{t}}})^2}{R^2}-1} \mathrm{d}r + (R^2 - \frac{R^2}{C_{\mathrm{t}}}) \times (1-\tau)^{N-1}}{R^2} \tag{2}$$

Note that p_s is the conditional probability given that at least one node transmits.

To compute throughput for an individual node, we first compute the expected time it spends on a successful transmission, a collision, and the time the channel is sensed as busy and idle. These values can be approximated as the probability of occurrence of each condition, multiplied by the time spent on that condition. We then compute the expected amount of transmitted bytes of a successful transmission, which can be approximated as the probability of successful transmission multiplied by the data packet size, divided by the sum of the times spent on each condition to compute the throughput.

Therefore we need the probability and duration of each of the following conditions:

1. A node transmits and the transmission is a successful transmission. The probability of this condition is $P_s = \tau \times p_s$, and the duration is T_s, where T_s is the time to transmit a data packet, ack packet, plus DIFS, SIFS time, and two propagation delays.
2. A node transmits but the transmission collides with others. The probability of this condition is $P_c = \tau \times (1 - p_s)$, and the duration is T_d, where T_d is the time to transmit a data packet plus DIFS and one propagation delay.
3. A node backs off due to a busy channel. There are two possibilities. First, the channel is busy because of a successful transmission. The probability is $P_{bs} = (1-\tau) \times (N-1) \times \tau \times (1-\tau)^{N-2}$, and the time is T_s. Second, the channel is busy because of a collision. The probability is $P_{bc} = (1-\tau) \times [1-(1-\tau)^{N-1} - P_{bs}]$, and the time is T_d.
4. The channel is idle. In such a case, the probability is $P_i = (1-\tau)^N$, and the time is a time slot ρ.

Therefore the throughput of a node operating a CSMA MAC is

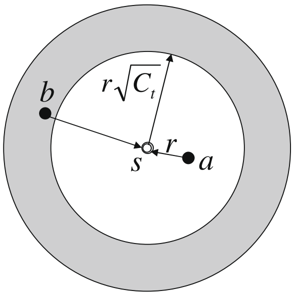


Fig. 3. If a and b transmit their packets to s at the same time, a's packet can still be correctly decoded at s due to the capture effect.

$$\frac{P_s D}{P_s T_s + P_c T_d + P_{bs} T_s + P_{bc} T_d + P_i \rho}, \tag{3}$$

where D is the packet size.

ClearBurst reserves a few slots of the channel for sources, the C-node, and downstream nodes of the C-node, to forward the packets to the sink in a burst, as shown in Fig. 2(e). Assume that there is no other nodes transmitting during these reserved slots and the C-node does not aggregate packets, i.e. $\rho = 1$, and the length of schedule advertisement time window is μ. We can compute the throughput of the C-node as there is only one node transmitting, i.e. $N = 1$, multiplied by the ratio of its slot to the entire epoch of a burst period, which is $\frac{\lambda}{3\lambda+\mu}$. However, for those intermediate nodes on the path from the C-node to the sink, they have to contend for the channel with their two-hop upstream nodes and two-hop downstream nodes. Therefore, the throughput of intermediate nodes can be computed by considering $N = 5$. Note that this computed throughput will be smaller then the throughput of the C-node. Thus, the system throughput of ClearBurst is equal to the throughput at intermediate nodes.

In simulations, the data packet size is 40 bytes, the ack packet size is 12 bytes, and the bandwidth of the radio is 19.2 kbps. By plugging these numbers into Eq. (3), we can get the analytic throughput of CSMA. For ClearBurst, since the system throughput equals the throughput of intermediate nodes, we use $N = 5$ and Eq. (3) to compute its analytic throughput.

We run simulations on CSMA and ClearBurst and compare the results with our analysis. The simulation methodology is described in Section 5. For $100 \times 100\,\mathrm{m}^2$ event size, there are approximately 15–30 source nodes when the network is deployed with 500–1000 sensors, all are within interference range. The results are shown in Fig. 4. We can see that the throughput of CSMA drops as the network density increases, while ClearBurst remains similar across different network densities and performs much better than CSMA. This confirms our claim and demonstrates the benefit of ClearBurst.

4.2. Energy consumption

In this section, we analyze the energy consumption of CSMA and ClearBurst protocols. When contention is intensive, the probability of collision is high. If a collision happens, a node has to retransmit the packet. Therefore, we use the expected number of transmissions to successfully transmit one packet as the metric to compare the energy efficiency of ClearBurst and CSMA. The metric called normalized number of transmissions is defined as:

$$E_{tx} = \frac{TX_{\text{success}} + TX_{\text{collision}} + TX_{\text{ack}}}{TX_{\text{success}}} \tag{4}$$

where TX_{success} is the number of successfully transmitted packets, $TX_{\text{collision}}$ is the number of collisions, and TX_{ack} is the number of ACK packets. Assume that there is no collision for ACK packets, $TX_{\text{ack}} = TX_{\text{success}}$. Therefore Eq. (4) becomes

$$E_{tx} = 2 + \frac{TX_{\text{collision}}}{TX_{\text{success}}} \tag{5}$$

In Eq. (2), we have computed p_s, which represents the conditional probability of a successful transmission given that a node transmits. The conditional probability of a collision is therefore $p_c = 1 - p_s$. Accordingly, Eq. (5) becomes

$$E_{tx} = 2 + \frac{p_c}{p_s} \tag{6}$$

The expected number of transmissions for the entire network can be approximated by $N \times E_{tx}$. For ClearBurst, when nodes are in the burst transmission mode, only the C-node can transmit, and there is no collision. Therefore, the expected number of transmissions for a successful transmission is two. When nodes are in the CSMA mode, the expected number of transmissions is simply E_{tx}. In simulations, since burst transmissions occupy 90% of the simulation time and CSMA accounts for 10% of the simulation time, the expected number of transmissions for ClearBurst is $2 \times 0.9 + (2 + p_c/p_s) \times 0.1$.

Fig. 5 shows the analytical and simulation results. In analysis, we do not consider the transmissions of intermediate nodes that are on the path from the C-node to the sink because it depends on the distance between the C-node and the sink. However, in simulations, all transmissions in the network are considered for computing E_{tx}. Therefore there are small gaps between the analysis and simulation results. However the trends are similar. When node density increases, CSMA consumes more energy for each successfully received packet.

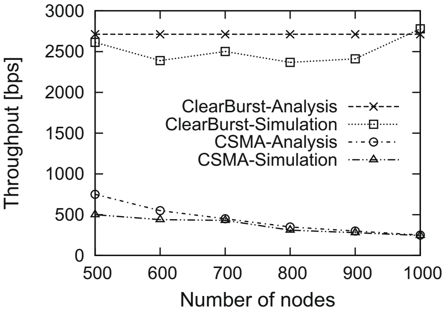


Fig. 4. Throughput analysis and simulation results for CSMA and ClearBurst.

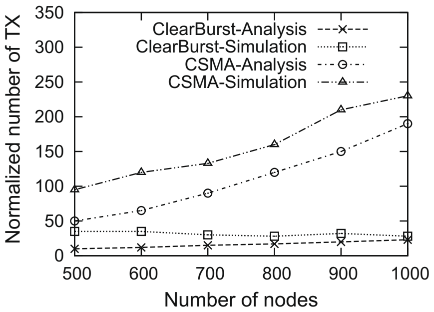


Fig. 5. Energy efficiency analysis and simulation results for CSMA and ClearBurst.

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002

5. Simulations

To study the performance of ClearBurst, we use *ns-2* to conduct extensive simulations using random topologies with various node densities, event sizes and source rates. We also studied the impact of event mobility, aggregation ratio and different queue length thresholds that are used to trigger Clearburst. Performance metrics include throughput, energy tax, latency, and fairness. Energy tax is defined as $(TX_{\text{data}} + TX_{\text{ack}})/R$, where R is the number of packets received at the sink, TX_{data} is the total number of transmissions for data packets and TX_{ack} is the total number of transmissions for ACK packets. Energy tax represents the average number of transmissions required to forward a packet to the sink. Fairness is evaluated by Jain's fairness index [27].

Table 1 shows the parameters used in the simulations. We compare the performance of ClearBurst with Carrier Sense Multiple Access (CSMA) and CSMA with Shortest Path Routing (CSMA + SP). In CSMA, all data packets are forwarded to the same C-node as in ClearBurst before they are forwarded to the sink. Whereas, in CSMA + SP, data packets follow shortest-path routes to the sink. All results are generated based on the ER model (all source nodes are within a certain radius of the event center), and packets with the same time stamp can always be aggregated.

5.1. Network density

We randomly generated six sensor networks of area $1000 \times 1000\ \text{m}^2$ to evaluate performance for different network densities, with 500–1000 nodes uniformly distributed in the network. The sink is at the bottom left corner. Sensors within the event region generate traffic at a constant rate of 2 pkts/s. For each simulation, 30 static events with 100 m radius are randomly generated. The average throughput, energy tax, latency, and fairness index with 95% confidence intervals are plotted in Fig. 6(a), (b), (c) and (d) respectively.

As network density increases, more sensors are in the event region. This increases the channel contention and results in more collisions which leads to packet drops in CSMA. Even worse, the C-node has little chance to forward packets accumulated in its queue. Therefore, even though packets are successfully delivered to the C-node, only a few of them can be forwarded toward the sink, which results in low throughput and energy efficiency in CSMA as shown in Fig. 6(a) and (b). By observing the queue length at the C-node, we found that the problem is due to contention induced queue overflow. Similar results can also be observed in CSMA + SP. However, it is important to note that using shortest path routes reduces the chance of early aggregation. Thus, CSMA + SP has the lowest throughput. Furthermore, in CSMA + SP, packets along interfering paths have to compete for the channel, which results in 50% higher energy consumption and latency than ClearBurst. In contrast to CSMA and CSMA + SP, ClearBurst has the highest throughput, lowest energy tax and a moderate latency between CSMA and CSMA + SP. This shows that ClearBurst successfully resolves the contention around the C-node. However, it should be noted that the design of ClearBurst favors the C-node and its children nodes because burst transmission is only applied at these nodes. Thus, we restrict these node such that they cannot generate a packet until a total number of n packets are received, where n is the number of immediate upstream nodes. After applying this policy, ClearBurst shows the same fairness level as the other two protocols as shown in Fig. 6(d).

Table 1
Parameters used in *ns-2* simulation

Parameter	Value
Communication range	100 m
Carrier sensing range	220 m
Channel bit-rate	19.2 kbps
Initial backoff window size	15
Congestion window size	32
Capture threshold	10
Queue size	50 pkts
Congestion threshold	20 pkts
Burst duration	1.9902 s
CSMA duration	0.438575 s
Event radius	100 m

5.2. Source rates

In this set of simulation, we fixed the number of nodes at 1000, and event radius at 100 m and varied the source rate from 0.5 pkt/s to 6 pkt/s. In Fig. 7(a) and (b), we can observe similar results as those for different network densities in Fig. 6(a) and (b). As the source rate increases, contention becomes more intensive causing the throughput of CSMA and CSMA + SP decline. However, when the source rate is as low as 0.5 - 1 pkts/s, CSMA outperforms ClearBurst. This is due to the fact that there are only 31 sensors in the event region and they generate traffic at a low rate (less than 1 pkt/s). CSMA can efficiently arbitrate transmissions among sensor nodes without incurring signalling overhead. However, as the source rate increases beyond 1 pkts/s, CSMA's throughput decreases and latency and energy tax increase dramatically. Note that when the source rate is low, CSMA + SP still has the lowest throughput. This is, again, due to the fact that shortest-path prohibits aggregation and long interfering routing paths increase the transmission delay and the collision probability (see Fig. 8).

5.3. Event radius

To evaluate the impact of the event size, we fix the network density at 1000 nodes and vary event radius from 50 m to 200 m. In this set of simulations, we can see a clear trend of performance degradation of CSMA. When the event radius is small, CSMA can arbitrate the channel access efficiently. Thus, it has similar performance as ClearBurst. As the event radius grows, CSMA's throughput drops and energy tax and latency increase dramatically.

In contrast, ClearBurst's throughput and energy tax remain steady until the event radius becomes larger than 100 m. When the event radius goes beyond 100 m, the performance of ClearBurst starts falling as well. By examining the packet reception and transmission of C-node, we found that ClearBurst successfully clears up all packets accumulated in C-node's queue. But the reception rate during the

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002

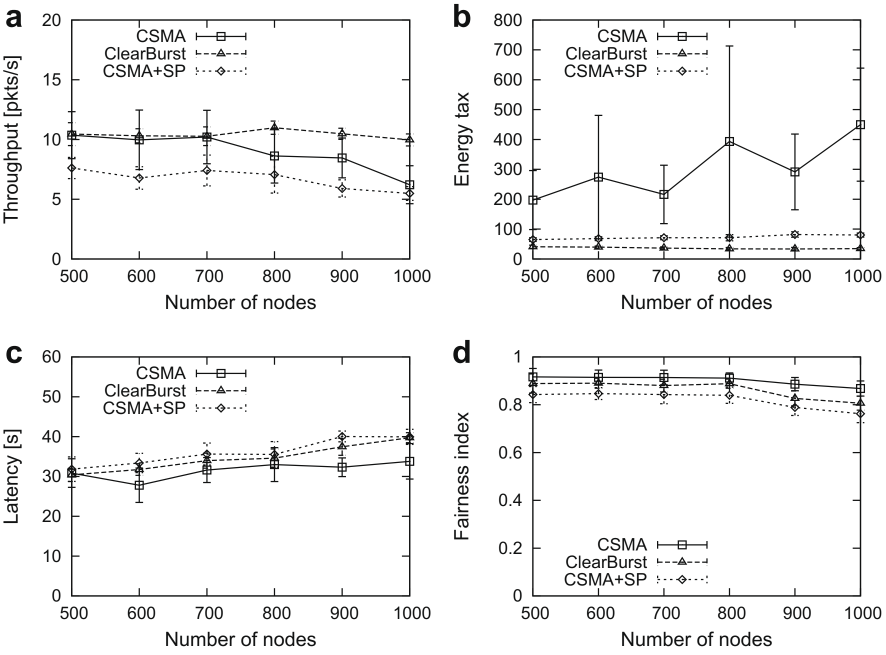


Fig. 6. Throughput, energy tax, latency and fairness index for various node densities.

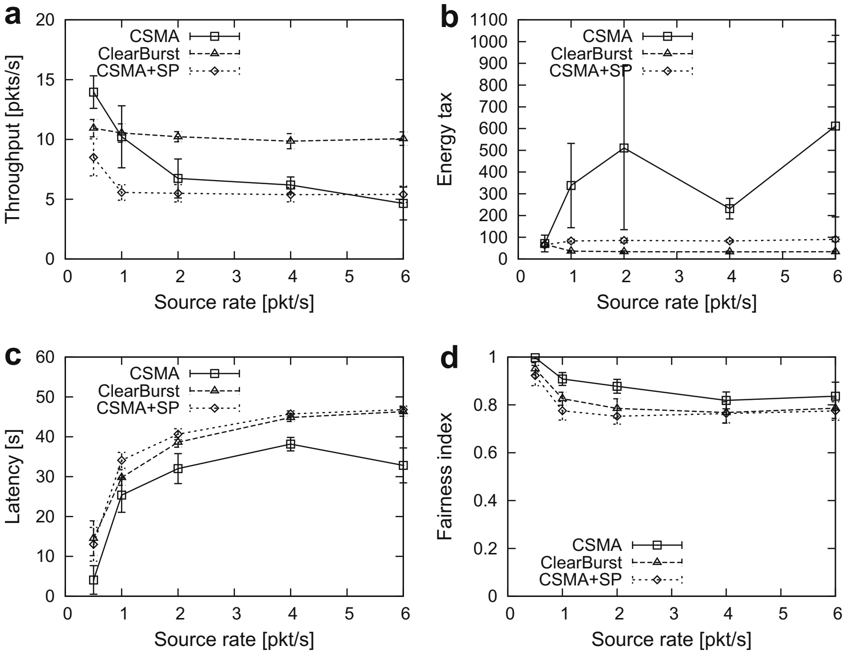


Fig. 7. Throughput, energy tax, latency and fairness index for various source rates.

CSMA duration is low. Therefore, the overall performance degrades.

5.4. Aggregation ratio and queue length thresholds

In sensor networks, data aggregation is an effective technique for reducing energy consumption and traffic load in the network. Here, we study the impact of aggregation ratio on CSMA, CSMA + SP and ClearBurst. The aggregation ratio is defined as the number of raw packets that can be aggregated to a single packet. To find out the impact of different burst sizes on the performance of ClearBurst, different queue length thresholds are used to trigger burst transmissions.

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002

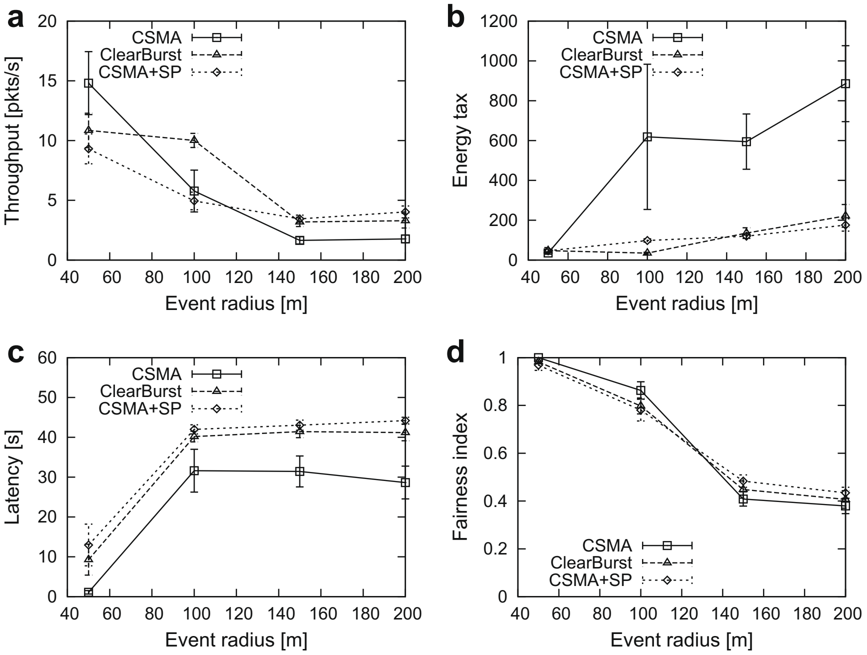


Fig. 8. Throughput, energy tax, latency and fairness index for various event radius.

Figs. 9(a) and (b) present the impact of aggregation ratio on throughput and energy efficiency. We observe that, the higher the aggregation ratio, the higher the throughput and energy efficiency. This is true for all three protocols. However, it should be noted that when the aggregation ratio is high, CSMA outperforms CSMA + SP in terms of throughput. This performance gain comes from aggregation. ClearBurst increases this gain further by reducing contention around the C-node. Furthermore, it should also be noted that even though CSMA's throughput improves as the aggregation ratio increases, its energy efficiency is still at least three times higher than the other two protocols (Fig. 9(b)). In contrast, ClearBurst provides highest throughput with lowest energy tax.

Fig. 10(a) shows the throughput of ClearBurst with different burst sizes when aggregation is turned off. It can be seen that, if aggregation is turned off, the larger the burst size, the higher the throughput. However, a larger burst size means packets must wait in the queue for a longer time before a burst transmission can be initiated. The longer queueing delay is reflected on latency which can be observed in Fig. 10(b). In contrast, as shown in Fig. 11(a), if aggregation is always possible, smaller burst sizes are better. This is because aggregation greatly reduces the amount of traffic in the network. If a larger burst size is used, packets must wait in the queue for an even longer time. This longer delay not only increases the overall packet delivery latency (Fig. 11(b)) but also lowers the throughput as fewer burst transmissions are triggered. These results suggest that when aggregation is always possible, smaller burst sizes are favored.

6. Experiments

We implement ClearBurst on TinyOS [15] to evaluate its performance in a real environment. Fig. 12 shows the

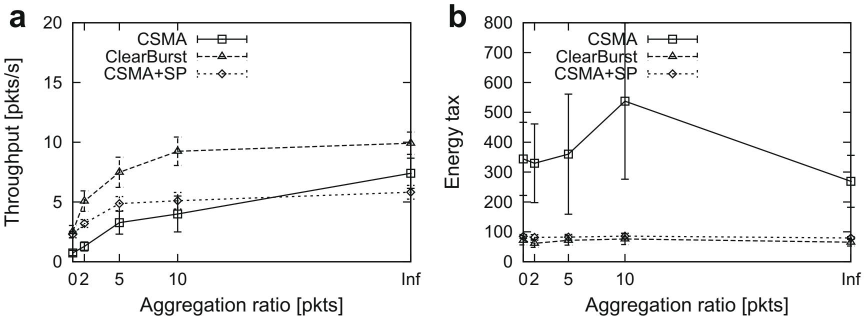


Fig. 9. Impact of aggregation ratio.

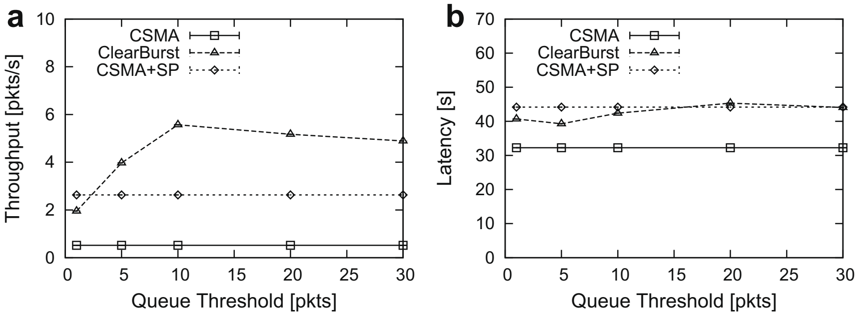


Fig. 10. Impact of queue threshold without aggregation.

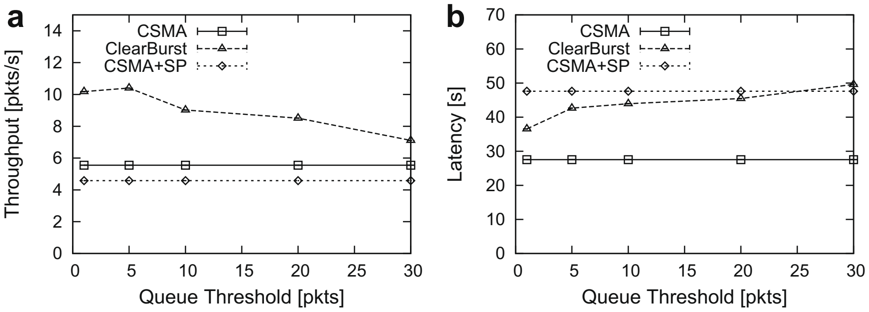


Fig. 11. Impact of queue thresholds with aggregation.

architecture of ClearBurst TinyOS implementation [15]. To support the functions required by ClearBurst, we extended the MacControl interface to include the following commands and events.

1. *SetPriority*: Specify the priority. When MAC is in the high priority mode, it uses smaller initial backoff and contention window size.
2. *SetSlotLength*: Set the length of the burst period needed to flush *BURST_THRESHOLD* packets in the queue.
3. *SlotExpired*: An event used by the MAC layer to inform ClearBurst module that the specified burst period or CSMA period has ended.

A circular queue of size 32 is implemented for ClearBurst. The ClearBurst module interacts with the circular queue through the CirQueueControl interface. CirQueueControl interface provides three commands which include Enqueue, Dequeue, and Length. This circular queue serves as the TX queue which is shared by all the applications. Messages generated locally and packets being routed through a node are sent to the queue first. The ClearBurst module then pops one packet from the queue each time and transmits it to the next hop.

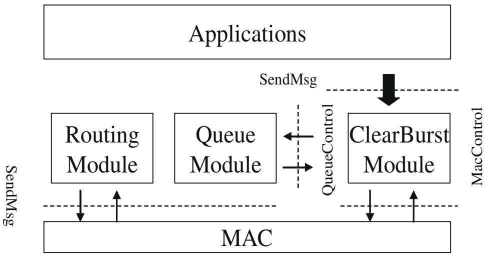


Fig. 12. TinyOS implementation.

The ClearBurst module continuously monitors the queue length. If it exceeds the congestion threshold, ClearBurst intervenes packet transmission. Before sending a packet to the MAC layer, the ClearBurst module calls *MacControl.SetSlotLength()* command to pass the remaining time of the burst period to the MAC layer. The MAC layer stamps the remaining time in the header right before the packet is to be transmitted. Every time the *SpiByteFifo.dataReady()* interrupt handler in the MAC is executed, the remaining time of current burst period is decremented. When the time reaches zero, the MAC layer informs ClearBurst module by signaling a *MacControl.SlotExpired* event.

Experiments are conducted on the Kansei testbed [16,17] with 49 nodes in grid topology. The sink is located at the bottom right corner. Eight nodes at the top left corner periodically send a packet to the C-node. In the experiments, C-nodes are manually selected.

As shown in Fig. 13(a) and (b), ClearBurst achieves four times higher throughput than TinyOS's CSMA MAC and yet is more energy-efficient. The results are similar to the results presented in the previous section. However, it clearly demonstrates that, even with a small network, the many-to-one traffic pattern in sensor networks has a severe impact on data delivery. Thus, transmission in the neighbor-

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002

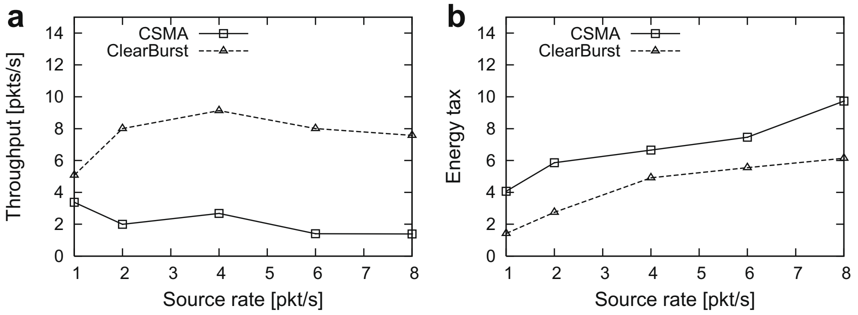


Fig. 13. Performance evaluation of various source rates on Kansei testbed.

hood of C-node must be coordinated, which validates the design of our protocol.

7. Discussion

Though we focused on static event in previous sections, we explore the possibility of applying ClearBurst in the scenarios where the event is mobile. In contrast to static events, when an event moves, the data collection tree should be reconfigured. New edges must be established to connect new source nodes to the tree, and old edges connecting to nodes that falls out of the event region should be pruned. Event tracking in sensor networks is a challenging problem [18]. Although protocols such as DCTC [10] can be used to construct and reconfigure the data collection tree to track the event, its signalling overhead is high. Furthermore, DCTC chooses the node closest to the event center as the root. In high data rate applications, contention around the tree will be extremely high. Thus, it is not applicable under high data rate applications. Therefore, instead of applying DCTC directly, we extend our C-node election protocol and design a distributed C-node's migration protocol for the purpose of event tracking and tree reconfiguration.

In order to track the event, the C-node periodically monitors the movement of the event. If the event has moved farther then a predefined distance, it releases its role and designates one of the neighbors, which is in the event region and has the smallest hop distance to the sink, as the new C-node. If the C-node is migrated too often, signalling overhead will increase and ClearBurst is applied only for a short period of time before the next migration. On the contrary, if the C-node does not migrate until the event is far away, congestion arising at the event's new location cannot be resolved. Furthermore, if the C-node's location becomes deep in the event region after event's movement, it will experience more intensive contention and the number of slots needed for burst transmission will have to be increased. As a compromise, we choose to migrate the C-node when the event has moved farther than half of the communication range. After the migration is completed, both the new C-node and the old C-node announce the new C-node's location in their neighborhoods. This information is then propagated in the event region for tree reconfiguration. When a node receives the new C-node's location information, it follows the tree construction procedure described in Section 3 to select its new parent. To reduce the tree reconfiguration overhead, new C-node's location information is only propagated in the region where reconfiguration is needed. In other words, if a node finds that its parent does not change after the C-node migration, it does not forward this information to downstream nodes. If the event does not move further, the new C-node can reduce or stop announcing its location information to minimize overhead.

We simulate moving events with 100 m event radius and random motion. The model used for events' moving pattern is random waypoint mobility model. All the sen-

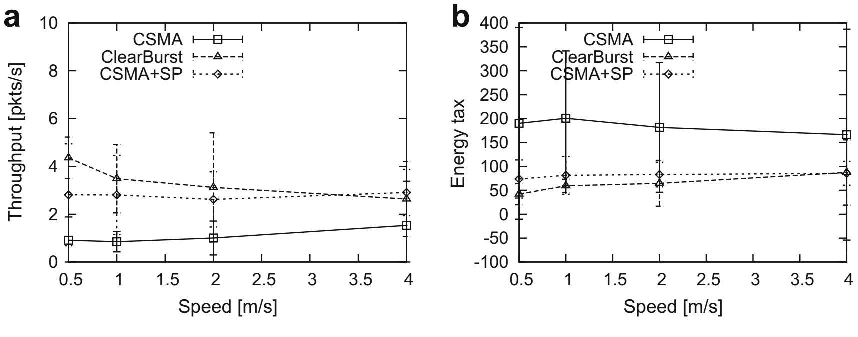


Fig. 14. Performance evaluation of moving events.

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002

sors that can detect the event generate event status update at the rate of 2 pkts/s. Fig. 14 shows the throughput and energy tax under various event moving speeds. When event moves slowly, ClearBurst outperforms CSMA and CSMA + SP. Therefore, ClearBurst does have the potential to increase network throughput if a more efficient event tracking and tree reconfiguration protocol can be applied.

8. Conclusion

This paper addresses the overload problem around nodes that are aggregating data, and provides a solution to improve system throughput and reduce energy consumption. The proposed MAC layer solution makes use of burst transmissions with low-overhead local advertisements to avoid contention during the bust-periods. Using extensive simulations, we observe that our proposed approach can achieve up to two times higher throughput and four times higher energy efficiency than CSMA in static event scenarios, with an increasing performance gap as the network gets overloaded (higher node densities and/or larger event sizes). These observations are also supported by the experiments on the *Kansei* testbed on different data rates. To apply ClearBurst in moving event scenarios, the location of C-node must migrate as the event moves. Through extensive simulations, we showed that ClearBurst with the C-node migration protocol can have four times higher throughput and two times more energy efficiency than CSMA. However, when the event moves fast, ClearBurst does not show significant performance gain in comparison to CSMA + SP. The reason for this performance degradation is due to the overhead of C-node migration and the reconfiguration of the data collection tree. The latter is a time consuming operation which limits how fast the protocol can react to the congestion arising at the new location of the event. In summary, our proposed approach is highly suited for data collection applications in sensor networks, especially for static and slow moving events. As to fast moving events, a more efficient data collection protocol is needed for event tracking and fast congestion resolution. These requirements impose many challenges in different network layers and will be further studied in our future works.

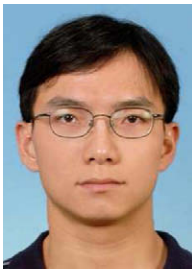
Acknowledgements

This material is based upon work supported by the National Science Foundation under Grants CNS-0546630 (CAREER Award), CNS-0721434, and CNS-0403342. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Gahng-Seop Ahn, Se Gi Hong, Emiliano Miluzzo, Andrew T. Campbell, Francesca Cuomo, Funneling-MAC: a localized, sink-oriented MAC for boosting fidelity in sensor networks, in: Proc. of SENSYS, 2006, pp. 293–306.
[2] Injong Rhee, Ajit Warrier, Mahesh Aia, Jeongki Min, Z-MAC: a hybrid MAC for wireless sensor networks, in: Proc. of SENSYS, 2005, pp. 90–101.
[3] Chaegwon Lim, Haiyun Luo, Chong-Ho Choi, RAIN: a reliable wireless network architecture, in: Proc. of ICNP, 2006, pp. 228–237.
[4] Chieh-Yih Wan, Shane B. Eisenman, Andrew T. Campbell, CODA: congestion detection and avoidance in sensor networks, in: Proc. of SENSYS, 2003, pp. 266–279.
[5] Bret Hull, Kyle Jamieson, Hari Balakrishnan, Mitigating congestion in wireless sensor networks, in: Proc. of SENSYS, 2004, pp. 134–147.
[6] Sumit Rangwala, Ramakrishna Gummadi, Ramesh Govindan, Konstantinos Psounis, Interference-aware fair rate control in wireless sensor networks, in: Proc. of SIGCOMM, 2006, pp. 63–74.
[7] Injong Rhee, Ajit Warrier, Jeongki Min, Lisong Xu, DRAND: distributed randomized TDMA scheduling for wireless ad-hoc networks, in: Proc. of MOBIHOC, 2006, pp. 190–201.
[8] Giuseppe Bianchi, Luigi Fratta, Matteo Oliveri, Performance evaluation and enhancement of the CSMA/CA MAC protocol for 802.11 wireless LANs, in: Proc. of 7th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 1996, pp. 392–396.
[9] Giuseppe Bianchi, Performance analysis of the IEEE 802.11 distributed coordination function, IEEE Journal on Selected Areas in Communications 2 (2000) 535–547.
[10] Wensheng Zhang, Guohong Cao, DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks, IEEE Transactions on Wireless Communication 3 (2004) 1689–1701.
[11] B. Krishnamachari, D. Estrin, S. Wicker, Impact of data aggregation in wireless sensor networks, in: 22nd International Conference on Distributed Computing Systems Workshops, 2002, pp. 575–578.
[12] Kaixin Xu, Mario Gerla, Lantao Qi, Yantai Shu, Enhancing TCP fairness in ad hoc wireless networks using neighborhood RED, in: Proc. of MOBICOM, 2003, pp. 16–28.
[13] Ren-Shiou Liu, Kai-Wei Fan, Prasun Sinha, ClearBurst: burst scheduling for contention-free transmissions in sensor networks, Technical report OSU-CISRC-9/07-TR68, 2007.
[14] Ren-Shiou Liu, Kai-Wei Fan, Prasun Sinha, ClearBurst: burst scheduling for contention-free transmissions in sensor networks, in: Proc. of IEEE WCNC, 2007, pp. 1899–1904.
[15] TinyOS Homepage: http://www.tinyos.net/.
[16] Emre Ertin, Anish Arora, Rajiv Ramnath, Mikhail Nesterenko, Vinayak Naik, Sandip Bapat, Vinod Kulathumani, Mukundan Sridharan, Hongwei Zhang, Hui Cao, Kansei: a testbed for sensing at scale, in: Proc. of IPSN/SPOTS Track, 2006, pp. 399–406.
[17] Anish Arora, Emre Ertin, Rajiv Ramnath, William Leal, Mikhail Nesterenko, Kansei: a high-fidelity sensing testbed, in: Proc. of IPSN, 2006, pp. 35–47.
[18] Kai-Wei Fan, Sha Liu, Prasun Sinha, Scalable data aggregation for dynamic events in sensor networks, in: Proc. of SENSYS, 2006, pp. 181–194.
[19] M. Younis, M. Youssef, K. Arisha, Energy-aware TDMA-based MAC for sensor networks, in: Proc. of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems, 2002, pp. 129–136.
[20] Venkatesh Rajendran, Katia Obraczka,J.J. Garcia-Luna-Aceves, Energy-efficient collision-free medium access control for wireless sensor networks, in: Proc. of SENSYS, 2003, pp. 181–192.
[21] Shashidhar Gandham, Milind Dawande, Ravi Prakash, Link scheduling in sensor networks: distributed edge coloring revisited, in: Proc. of INFOCOM, 2005, pp. 2492–2501.
[22] Jing Li, Georgios Y. Lazarou, A bit-map-assisted energy-efficient MAC scheme for wireless sensor networks, in: Proc. of IPSN, 2004, pp. 55–60.
[23] Vinayak Naik, Anish Arora, Harvest: a reliable and energy efficient bulk data collection service for large scale wireless sensor networks, Technical report OSU-CISRC-4/06-TR37, 2006.
[24] Chieh-Yih Wan, Andrew T. Campbell, Lakshman Krishnamurthy, PSFQ: a reliable transport protocol for wireless sensor networks, in: Proc. of WSNA, 2002, pp. 1–11.
[25] Yogesh Sankarasubramaniam, Ozgur B. Akan, Ian F. Akyildiz, ESRT: event-to-sink reliable transport in wireless sensor networks, in: Proc. of MobiHoc, 2003, pp. 177–188.
[26] Chieh-Yih Wan, Shane B. Eisenman, Andrew T. Campbell, Jon Crowcroft, Siphon: overload traffic management using multi-radio virtual sinks in sensor networks, in: Proc. of SENSYS, 2005, pp. 116–129.
[27] R. Jain, D.M. Chiu, W. Hawe, A quantitative measure of fairness and discrimination for resource allocation in shared systems, DEC Research Report TR-301, 1984.
[28] Alec Woo, Terence Tong, David Culler, The MintRoute protocol, tinyos-1.x/tos/lib/MintRoute.

Ren-Shiou Liu received his BS and MS degree in Computer Science and Information Engineering from National Chiao Tung University, Taiwan in 1998 and 2000. He worked at BenQ Mobile System as a software engineer and researcher from 2000–2002 and 2003–2005, respectively. He is pursuing his Ph.D. in Department of Computer Science and Engineering at Ohio State University. His research focuses on design and analysis of network protocols for sensor networks and mesh networks.



Kai-Wei Fan received his BS and MS degree in Computer Science and Information Engineering from National Chiao Tung University, Taiwan in 1997 and 1999, respectively. He was a senior engineer and project manager in network security division in a start-up company from 1999 to 2004. He is currently pursuing his PhD degree in Department of Computer Science and Engineering at The Ohio State University. His research interests include wireless sensor networks and mesh networks where he focuses on design and implementation of energy efficient protocols.

Prasun Sinha received his PhD from University of Illinois, Urbana-Champaign in 2001, MS from Michigan State University in 1997, and B. Tech. from IIT Delhi in 1995. He worked at Bell Labs, Lucent Technologies as a Member of Technical Staff from 2001 to 2003. Since 2003 he is an Assistant Professor in Department of Computer Science and Engineering at Ohio State University. His research focuses on design of network protocols for sensor networks and mesh networks. He has served on the program committees of various conferences including INFOCOM, MOBICOM and MOBIHOC. He has won several awards including Ray Ozzie Fellowship (UIUC, 2000), Mavis Memorial Scholarship (UIUC, 1999), and Distinguished Academic Achievement Award (MSU, 1997). He received the prestigious NSF CAREER award in 2006.

Please cite this article in press as: R.-S. Liu et al., Locally scheduled packet bursting for data collection ..., Ad Hoc Netw. (2008), doi:10.1016/j.adhoc.2008.08.002