TDMA Service for Sensor Networks[1]

Sandeep S. Kulkarni Mahesh (Umamaheswaran) Arumugam
Software Engineering and Network Systems Laboratory
Department of Computer Science and Engineering
Michigan State University
East Lansing MI 48824 USA

Abstract

Sensors networks are often constrained by limited power and limited communication range. If a sensor receives two messages simultaneously then they collide and both messages become incomprehensible. In this paper, we present a simple time division multiple access (TDMA) algorithm for assigning time slots to sensors and show that it provides a significant reduction in the number of collisions incurred during communication. We present TDMA algorithms customized for different communication patterns, namely, broadcast, convergecast and local gossip, that occur commonly in sensor networks. Our algorithms are self-stabilizing, i.e., TDMA is restored even if the system reaches an arbitrary state where the sensors are corrupted or improperly initialized.

Keywords: Sensor Networks, Time Division Multiple Access (TDMA), Broadcast, Convergecast, Local Gossip

1 Introduction

Sensor networks have become popular recently due to their application in unattended tracking and detection of undesirable objects, hazard detection, data gathering, environment monitoring, and so on. Further, due to their low cost and small size, it is easy to deploy them in large numbers. However, these sensors are resource constrained. Specifically, sensors are constrained by limited power and limited communication distance. Hence, they need to collaborate with each other to perform the task at hand.

One of the important problems in sensor networks is message collision. More specifically, if a sensor receives two messages simultaneously then they collide and both messages become incomprehensible. Also, it is difficult for a sensor to know whether a given message reached all its neighbors as a message sent by a sensor may collide at one sensor and be correctly received at another sensor.

To deal with the problem of message collision, approaches like collision-avoidance and collision-freedom protocols are proposed. Collision-avoidance protocols like carrier sense multiple access (CSMA) [1, 2] try to avoid collisions by sensing the medium before transmitting a message. Another example of collision-avoidance protocol is carrier sense multiple access and collision detection (CSMA/CD). CSMA/CD [2] is difficult to use in the context of sensor networks as the collisions are often detected at some receivers whereas other receivers and sender(s) may not detect the collision.

Collision-freedom protocols like frequency division multiple access (FDMA), code division multiple access (CDMA), and time division multiple access (TDMA) ensure that collisions do not occur while the sensors communicate. FDMA (e.g., [2]) is not applicable in the context of sensor networks since the sensors (e.g., University of California at Berkeley's MICA motes [3,4]) are often restricted to transmit only on one frequency. CDMA (e.g., [5]) requires expensive operations for encoding/decoding a message. Therefore, CDMA is not preferred for sensor networks that lack the special hardware required for CDMA and that have limited computing power.

In this paper, we present our TDMA service for sensor networks. This service is based on the algorithm in [6] and new TDMA algorithms proposed in this paper. Our service ensures collision-freedom and fair bandwidth allocation among different sensors. Moreover, since it does not require any expensive operations, it is more applicable to sensor networks

Our TDMA service lets one customize the assignment of time slots to different sensors by considering the common communication patterns that occur in the application. Specifically, we consider three commonly occurring communication patterns: *broadcast*, *convergecast*, and *local gossip*. In broadcast, a message is sent to all the sensors in the network. Broadcast is useful when a base station wants to transmit some information (e.g., program capsules for reprogramming the sensors [7]) to all the sensors in the network. We also consider two other communication patterns, convergecast and local gossip. These algorithms are based on our experience with *Line in the Sand* demonstration [8]. In this demonstration, the sensors are arranged in a thick line (grid). When an intruder crosses this line, the sensors detect it. Now, to classify the intruder, the sensors that observed the intruder communicate with each other. We consider two ways of classification, internal and external. In an internal classification, the sensors that detect the intruder communicate with each other locally. Based on this motivation, we consider the communication pattern, local gossip, where a sensor sends a message to its neighboring sensors within some distance. In an external classification, the sensors send their data to the base station that exfiltrates the data outside the sensor network. Based on this motivation, we consider the communication pattern, convergecast, where a group of sensors send a message to a particular sensor.

Another important concern for a communication protocol is the errors in sensor location. Errors are introduced in sensor location due to misplacement of sensors, or external factors like wind, vehicle movement, etc. Communication protocols that depend on sensor location should be able to tolerate this kind of error.

Contributions of the paper. In this paper, we focus on the TDMA service for sensor networks for different communication patterns. The main contributions of this paper are as follows.

- We propose new TDMA algorithms for convergecast and local gossip. And, we show that our TDMA service is collision-free for broadcast, convergecast, and local gossip. Further, we show that the collision-avoidance protocols like CSMA suffer significant number of collisions.

[1]Email: {sandeep,arumugam}@cse.msu.edu, Web: http://www.cse.msu.edu/~{sandeep,arumugam}, Tel: +1-517-355-2387, Fax: +1-517-432-1061.

This work was partially sponsored by NSF CAREER CCR-0092724, DARPA Grant OSURS01-C-1901, ONR Grant N00014-01-1-0744, NSF equipment grant EIA-0130724, and a grant from Michigan State University

- We propose TDMA algorithms for different grid topologies. Specifically, we present TDMA algorithms for rectangular and hexagonal grids.
- We show that the delay in delivering a message using our TDMA algorithm in all three communication patterns is within acceptable limits of the application requirements. We note that the collision-avoidance protocols may provide lesser delay for some messages, but the percentage of messages delivered is often significantly less than 100%.

Organization of the paper. The rest of the paper is organized as follows. In Section 2, we discuss the work related to TDMA protocols for sensor networks. Then, in Section 3, we present new TDMA algorithms for different communication patterns and topologies. In Section 4, we introduce the simulation model and present the simulation results. Finally, we make concluding remarks in Section 5.

2 Related Work

In this section, we discuss the related work on TDMA-based MAC protocols in sensor networks. TDMA protocols can be classified as randomized (e.g., [9–11]) and and deterministic protocols (e.g, [6, 12]).

In [9], whenever a collision occurs during startup (synchronization phase), exponential backoff is used for determining the time to transmit next. The complexity of this algorithm is $O(N)$, where N is the number of system nodes. One of the important assumption in [9] is that each node has a unique message length. In [10, 11], initially, nodes are in random-access mode and TDMA slots are assigned to the nodes during the process of network organization. By contrast, in our work, the length of messages from different nodes can be equal (upto a maximum), the complexity is $O(D)$ where D is the diameter of the network, and deterministic startup algorithm is used to assign time slots to different sensors.

In [12], Arisha et al propose a clustering scheme to allot time slots to different sensors. Each cluster has a gateway node. The gateway node informs each sensor in its cluster about the time slots in which the sensors can transmit messages and also, the time slots in which the sensors should listen. In this algorithm, slot assignment is performed by the gateway and communicated to different sensors.

Collision-free communication algorithm proposed for rectangular grids by Kulkarni and Arumugam in [6] uses the notion of communication and interference range. Specifically, the algorithm assumes that the communication range is 1, i.e., a sensor can communicate with certainty with nodes that are its neighbors. And, the interference range is assumed to be y, i.e., if j receives messages from k and l such that k, l are within (rectangular grid) distance y of j then they collide.

In this program, initial slots are determined using the diffusion initiated by the sensor at the left-top position in the grid. Whenever a sensor receives the diffusion message from its left (respectively, top) neighbor, it forwards the message after 1 (respectively, $y+1$) slot(s). Once the initial slots are determined, the sensors can transmit messages once in every $P=(y+1)^2+1$ subsequent slots.

The TDMA algorithm in [6] is designed for broadcast, where the base station sends some information to all sensors on a rectangular grid. By contrast, the algorithm introduces significant delay for other communication patterns. Further, in [6], the effect of errors in sensor location is not addressed. In this paper, we propose new TDMA algorithms for two other commonly occurring communication patterns, *convergecast* and *local gossip*. We present simulation results that compare CSMA based algorithm with these algorithms and the algorithm in [6]. Also, we present new TDMA algorithms for hexagonal grids.

3 Algorithms for TDMA Service

In this section, we present TDMA algorithms for different communication patterns on a rectangular grid and a hexagonal grid. Towards this end, in Section 3.1, we first state the system model and identify the assumptions made in this paper. Next, in Section 3.2, we discuss the TDMA algorithm for convergecast, and in Section 3.3, we discuss the TDMA algorithm for local gossip; these algorithms assume that the topology is a rectangular grid. In Section 3.4, we present the algorithms for hexagonal grid. For reasons of space, we relegate the proofs of these algorithms as well as extension of these algorithms for triangular grids to [13]. Also, we refer the reader to [13] for the extension of these algorithms to deal with failed/sleeping sensors.

3.1 Model and Assumptions

We assume that the sensors are arranged in a rectangular or hexagonal grid and that each sensor knows its location in this grid. Further, we assume that each sensor has a communication range and an interference range. Communication range is the distance upto which a sensor can communicate with certainty/high probability. Interference range is the distance upto which a sensor can communicate, although the probability of such a communication may be low. However, if k receives another message while j is sending a message, it is possible that collision between these two messages can prevent k from receiving either of those messages. Based on the definition of the interference range, it follows that it is at least equal to the communication range. Also, we assume that the sensors are aware of their communication range and interference range.

3.2 TDMA Service for Convergecast

We recall that the TDMA algorithm from [6] (cf. Section 2) is suitable for broadcast. Hence, we use this algorithm for the case where TDMA service is customized for broadcast. However, the algorithm in [6] introduces a significant delay for convergecast, where a group of sensors send data (for example, information about the activities of an intruder in the field [8]) to the base station.

To reduce the delay for convergecast, we change the slot assignment as follows: If j receives a message from its left neighbor then it chooses to transmit the diffusion in $(-1)^{th}$ slot (in circular sense). In other words, j transmits in the $(P-1)^{th}$ slot, where P (=$(y+1)^2+1$) is the interval between slots assigned to a sensor and y is the interference range of the sensors. If j receives a message from its top neighbor then it transmits in the $(-(y+1))^{th}$ slot. (For example, see Figure 1 for slot assignment for the case where $y=2$.) After the first slot is determined, the sensors can then transmit once in every P slots.

As we can see from Figure 1, when a sensor transmits a message that is to be relayed by sensors closer to the base station (left-top sensor), such a relay introduces only a small (respectively, no) delay. Thus, the TDMA algorithm customized for convergecast is as follows:

```
const $P = (y+1)^2 + 1$;
// Initial slot assignment for convergecast
When sensor $j$ receives a diffusion message from $k$
        if ($k$ is west neighbor at distance $1$)
          transmit in the $P + (-1)^{th}$ slot.
        else if ($k$ is north neighbor at distance $1$)
          transmit in the $P + (-(y+1))^{th}$ slot.
        else // duplicate message
          ignore

// TDMA algorithm for convergecast
If sensor $j$ transmits a diffusion message at time slot $t_j$,
        $j$ can transmit at time slots, $\forall c : c \geq 0 : t_j + c * P$.
```

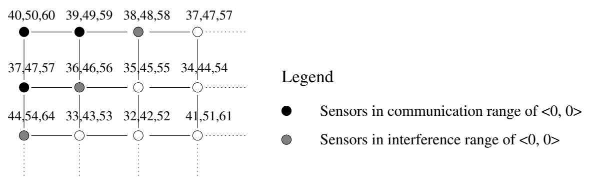

Figure 1. TDMA slot assignment for convergecast where communication range=1, interference range=2. The number associated with each sensor denotes the time at which it can send a message. Some initial slots are not shown.

3.3 TDMA Service for Local Gossip

For local gossip, we increase the value of the period (P) to $2((y+1)^2+1)$, twice the previous value. With this increased value, each sensor gets two slots (even and odd) in this period. Let the slots assigned to the initiator be 0 and $P-1$. To simplify the presentation, let us assume that the initiator starts a diffusion in its even or the 0^{th} slot. When j receives the diffusion from its left neighbor, it chooses the slot that is 2 higher than that used by the left neighbor. Likewise, when j receives the diffusion from its top neighbor, it chooses the slot that is $2(y+1)$ higher than that used by the top neighbor. (For example, see Figure 2 for slot assignment for the case where $y=2$.)

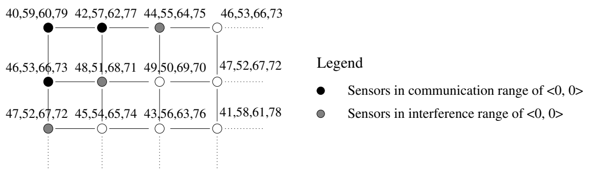

Figure 2. TDMA slot assignment for gossip where communication range=1, interference range=2. The number associated with each sensor denotes the time at which it can send a message. Some initial slots are not shown.

In our solution for gossip, whenever sensor k transmits in the even slot, say t_k, it can also transmit in $((P-1)-t_k)\ mod\ P$, the odd slot. Thus, the TDMA algorithm customized for local gossip is as follows:

```
const $P = 2((y+1)^2 + 1)$;
// Initial slot assignment for local gossip
When sensor $j$ receives a diffusion message from $k$
        if ($k$ is west neighbor at distance $1$)
          transmit after $2$ slots.
        else if ($k$ is north neighbor at distance $1$)
          transmit after $2(y+1)$ slots.
        else // duplicate message
          ignore

// TDMA algorithm for local gossip
If sensor $j$ transmits a diffusion message at time slot $t_j$,
        $j$ can transmit at time slots,
          $\forall c : c \geq 0 : t_j + c * P$,
              $(((P-1)-t_j)\ mod\ P) + c * P$.
```

Observe that if the algorithm in 2 is used for broadcast (respectively, convergecast), the delay is larger than the case where TDMA is optimized for broadcast (respectively, convergecast). In spite of this deficiency, the TDMA service provides substantial benefits for broadcast and convergecast even if it is customized for local gossip. To see this, observe that if a sensor wants to transmit a message in any given direction (east, west, north, south, southeast, southwest, northeast, or northwest) then sensors that receive that message can forward it with a small delay. Thus, even if the communication pattern is unknown or varies with time, customizing the TDMA service for local gossip provides a significant benefit for other communication patterns.

3.4 Hexagonal Grids

Consider a hexagonal grid network where a sensor can communicate with its distance 1 neighbors and interfere with its distance 2 neighbors (cf. Figure 3). We assume that the initiator of the diffusion, which assigns the initial slots to the sensors, is located at the left-most corner on the left-top hexagon in the network (cf. Figure 3).

From Figure 3, we observe that whenever the initiator transmits, sensors located at the top (say, j) and bottom (say, k) of the initiator at geometric distance 1 from the initiator can transmit next. However, if both these sensors transmit simultaneously then collision occurs at the initiator. Hence, we proceed as follows: whenever j receives the diffusion message from the initiator, it retransmits the message after 1 slot. Likewise, whenever k receives the diffusion message from the initiator, it retransmits the message after $2y$ slots, where y is the interference range of the sensors. Further, whenever a sensor receives a message from its neighbor on the straight edge (cf. Figure 3), it forwards the message after 1 slot.

Once the initial slots are assigned, each sensor can determine future slots based on the time it forwards the diffusion message. For a hexagonal grid, the period between successive slots, $P = 2y(y+1) + \lfloor \frac{y}{2} \rfloor + 1$ suffices. Thus, the TDMA algorithm for hexagonal grids is as follows:

```
const $P = 2y(y+1) + \lfloor \frac{y}{2} \rfloor + 1$;
// Initial slot assignment for hexagonal grids
when sensor $j$ receives a diffusion message from $k$
        if ($k$ is at distance $1$ in the same level
        (i.e., $j - k$ is a straight edge))
          transmit after $1$ slot.
        else if ($k$ is at distance $1$ in the lower level)
          transmit after $1$ slot.
        else if ($k$ is at distance $1$ in the upper level)
          transmit after $2y$ slots.
        else // duplicate message
          ignore

// TDMA algorithm for hexagonal grids
If sensor $j$ transmits a diffusion message at time slot $t_j$,
        $j$ can transmit at time slots, $\forall c : c \geq 0 : t_j + c * P$.
```

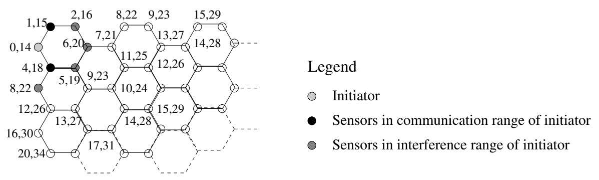

Figure 3. TDMA slot assignment in hexagonal-grid network where communication range=1 and interference range=2. The number associated with each sensor denotes the time at which it can send a message. Slots for some sensors are not shown.

Remark. We note that the above algorithm is customized for broadcast. Similar to the algorithms in Section 3.2 and 3.3, we can also customize the TDMA service on a hexagonal grid for convergecast and local gossip.

4 Simulation of TDMA Service

In this section, we present simulation results for our TDMA algorithms using *prowler* [14], which allows one to simulate arbitrarily large number of sensors (especially MICA motes). Specifically, in Section 4.1, we present the simulation model and in Section 4.2, we present the results.

We note that we have also implemented the TDMA service in the University of California at Berkeley's MICA motes [3, 4]. For reasons of space, we refer the interested reader to `http://www.cse.msu.edu/~sandeep/software` for the middleware architecture of our implementation in MICA motes and for the corresponding downloadable code.

4.1 Simulation Model

In this section, we discuss the simulation model of the experiments. We use a probabilistic wireless network simulator, *prowler* [14].

Using prowler, one can prototype different sensor network applications, radio/communication models, propagation models and topology. For our TDMA simulations, we use the radio/communication model that is based on the algorithms in Section 3. To compare our algorithms with the existing implementation, we use the default radio models (CSMA and 'no mac layer') provided by prowler. Finally, the underlying topology is a rectangular grid where the base station is in one corner of the grid.

Now, we discuss the simulations we performed in the context of these communication patterns. Then, we discuss the simulations we performed to study the effect of location errors.

Broadcast. The base station (sensor at left-top corner) initiates a broadcast. It sends the broadcast message to its neighbors in the communication range. Whenever a sensor receives the broadcast message for the first time, it relays it (for sensors farther from the base station). We conduct the broadcast simulations for different network sizes. In these simulations, we consider the following metrics: maximum delay incurred in receiving the broadcast message, number of sensors that receive the broadcast message, and number of collisions. Since CSMA (respectively, no MAC layer) does not guarantee reception by all sensors, we also consider the delay when a certain percentage of sensors receive the broadcast message. Regarding collisions, we compute the ratio of the number of collisions to the number of messages. Note that this ratio can be greater than 1 as one message can potentially collide at several sensors.

Convergecast. For convergecast, a set of sensors send a message to the base station (approximately) at the same time. In our experiments, we keep the network size fixed at 10x10. We choose a subgrid of varying size; sensors in this subgrid transmit the data to the base station. We assume that the subgrid that sends the data to the base station is in the opposite corner from the base station. For these simulations, we compute maximum delay incurred for receiving messages at the base station, the percentage of sensors whose messages are received by the base station and the number of collisions.

Local gossip. In local gossip, a subgrid of nodes send the data. The goal is to transmit the data from these sensors to the sensors in the subgrid and the neighbors of the sensors in the subgrid. Thus, local gossip is applicable in *locally* determining the set of sensors that observed a particular event. In our experiments, we keep the network size fixed at 10x10. And, we choose different sizes of subgrids; sensors in this subgrid transmit the gossip messages. For these simulations, we compute the average delay incurred for receiving messages at the nodes that are expected to receive the local gossip and number of collisions.

Location errors. We introduce location errors in the sensors as follows. Let e_d be the distance a sensor is perturbed from its ideal location, and θ_d be the angle of perturbation. The error distance e_d is determined using the normal distribution $N(\mu, \sigma)$, where μ is the mean error distance and σ is the standard deviation of e_d. Thus, the error in location on 95% of the sensors is in the range $(-\mu - 2\sigma, \mu + 2\sigma)$. Hence, to determine the topology, we increase the *physical communication range* by $\mu + 2\sigma$. However, the communication and interference range used by the algorithm is 1. For small perturbations (i.e., $\mu \leq 0.2$), increasing the physical communication range is sufficient to ensure that the network is connected. However, for larger perturbations (i.e, $\mu > 0.2$), if the communication and interference range used by the algorithm is 1, number of collisions increase significantly. Hence, we need to increase the interference range that the algorithm uses, say, to 2.

4.2 Simulation Results

In this section, we present the simulation results that compare the TDMA algorithms with the case where CSMA is used and with the case where no MAC layer is used. In these simulations, the communication and interference range is 1. We first present our results for broadcast. Then, we consider convergecast and local gossip. Finally, we consider the issue location errors. Based on the values used in [8], in convergecast and local gossip, the TDMA service groups upto 4 messages in the queue into a single message before transmitting. For reasons of space, the results for different interference ranges and different grouping factors, are relegated to [13].

4.2.1 Broadcast

In Figure 4, we present our simulation results for broadcast. Figure 4(a) identifies the number of collisions that occur in different algorithms. As expected, TDMA is collision free for all network sizes. By contrast, in CSMA, about 10% of messages suffer from collisions. However, if no MAC layer is used then the number of collisions is more than the number of messages sent. This is due to the fact that one transmitted message often collides at more than one sensor.

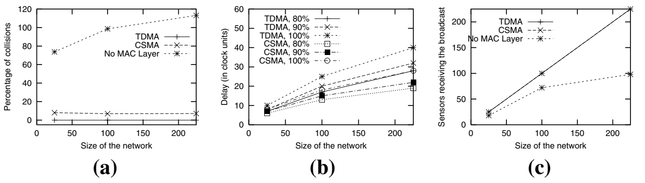

Figure 4. Results for broadcast with communication range=1, interference range=1

Figure 4(b) identifies the maximum delay incurred in receiving broadcast messages. Since all sensors may not receive the broadcast message when CSMA is used, we consider the delay when a certain percentage, 80-100%, of sensors receive the broadcast message. As we can see, the delay in TDMA based schemes is only slightly higher.

Figure 4(c) identifies the number of sensors that receive the broadcast message. We find that with CSMA/TDMA, all sensors receive the message. However, without the MAC layer, the number of sensors that receive the message is less than 50%.

4.2.2 Convergecast

In Figure 5, we present our simulation results for convergecast. Figure 5(a) identifies the number of collisions that occur in different algorithms. As we can see from Figure 5(a), although the TDMA based solution is collision free, there are a significant number of collisions with CSMA. Regarding delay, as we can see from Figures 5(b) and 5(c), the delay incurred by TDMA is reasonable and that the base station receives all the messages sent by the sensors. By contrast, with CSMA, approximately 50% of the messages are received when the number of sensors sending the data to the base station increases.

We note that the number of collisions without the MAC layer is significantly more than that for CSMA/TDMA. The number of collisions decrease as the size of the field sending data increases. This anomaly occurs due to the fact that when the number of sensors that start the convergecast is more, many of their messages fail on the first few links. Effectively, this reduces the number of collisions as collisions occur only on initial links. In fact, as we can see from Figure 5(c), without MAC layer, no data reaches the base station.

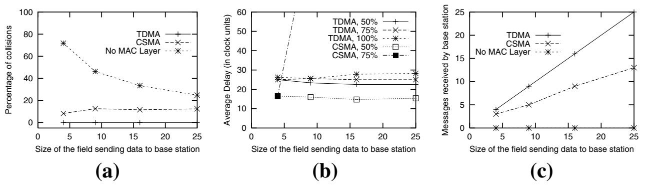

Figure 5. Results for convergecast with communication range=1, interference range=1

4.2.3 Local gossip

In Figure 6, we present our simulation results for local gossip. Figure 6(a) identifies the number of collisions as the size of the group performing local gossip increases. As we can see, CSMA based solution suffers significant collisions whereas TDMA based solution is collision free. Note that the percentage of collisions decrease as the size of the group performing local gossip increases. As discussed in the case of convergecast, this is due to the fact that with increased communication, many messages create collisions in the initial part of the gossip communication and then they are lost. Also, as seen from Figure 6(b), the delay in TDMA is slightly more than that in CSMA. However, unlike TDMA where all expected sensors receive the gossip messages, in CSMA, such sensors receive approximately 50% of messages.

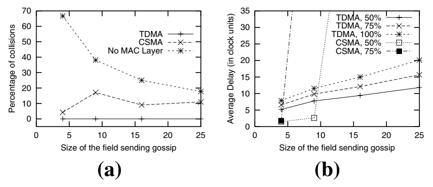

Figure 6. Results for local gossip with communication range=1, interference range=1

4.2.4 Effect of Location Errors

In our location error experiments, we find that even if the sensors are perturbed from their ideal position, the results are close to those presented earlier if the perturbation is small and the communication range is increased so that the network remains connected. As mentioned in Section 4.1, the geometric distance of a sensor from its ideal location is chosen by the uniform distribution $N(\mu, \sigma)$. For $\mu \leq 0.2$, we let interference range to be equal to 1 and for $\mu > 0.2$, we let the interference range to be equal to 2. In our simulations, μ takes the following values: $0.0 - 0.4$ and σ takes the following values: $0.0 - 0.2$.

Broadcast. In Figure 7, we present the simulations results for broadcast with location errors. Figure 7(a) identifies the percentage of collisions during broadcast. As we can see, when μ increases, the number of collisions increases. However, the collisions are within 2%. Figure 7(b) identifies the maximum delay involved in delivering the broadcast message to all the sensors. We can note that the delay is within 10% when compared to the case where no location errors are introduced. Finally, Figure 7(c) identifies the number of sensors receiving the broadcast message. As we can observe, all the sensors receive the message except for the case where $\mu = 0.2$ and $\sigma = 0.1$. Even in this case, more than 96% of the sensors receive the broadcast message. Table 1 shows the percentage of collision for the case where $\mu = 0.4$ and interference range=2. As we can observe, the percentage of collisions is small with increased interference range.

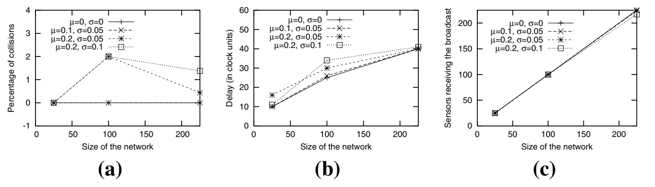

Figure 7. Results for broadcast with location errors

Convergecast. In Figures 8(a) and 8(b), we present the simulation results for convergecast with location errors. Figure 8(a) identifies the number of collisions during the message communication. We note that, as the error in sensor location increases, collisions increase. Further, as observed earlier, the collisions are within acceptable limits, i.e., within 8%. Figure 8(b) identifies the average delay involved in delivering the convergecast messages to the base station; the average delay is within 2% when compared to the case where were no location errors are introduced. Further, similar to the case where no location errors are present, the base station receives all the convergecast messages. Moreover, if the mean error distance increases, we can keep the percentage of collisions small by increasing the interference range (cf. Table 1).

Local gossip. In Figures 8(c) and 8(d), we present the simulation results for local gossip with location errors. Similar to the observations made earlier in this section, from Figure 8(c), we observe that the number of collisions during message communication is small (i.e., within 4%). Further, the delay involved in delivering the local gossip messages is within 15% when compared to the case where no location errors are introduced. Finally, all the local gossip messages are delivered to the group that expects such messages. Moreover, if the mean error distance increases, we can keep the percentage of collisions small by increasing the interference range (cf. Table 1). From these simulations, we conclude that the location errors do not significantly affect the performance of our TDMA service.

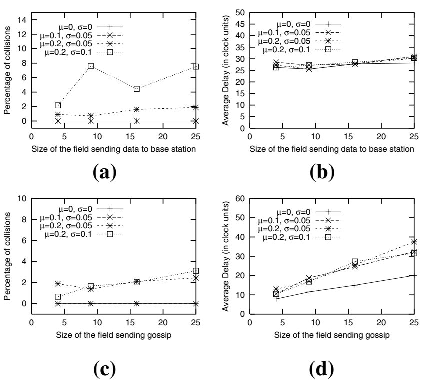

Figure 8. Results for convergecast and local gossip with location errors

Table 1. Results for $\mu = 0.4$, $\sigma = 0.2$, and interference range$= 2$

	Broadcast		Converge-cast	Local Gossip
Network Size	% of Collisions	Field Size	% of Collisions	% of Collisions
		4	9.6	4.9
25	0	9	8.8	7.1
100	5	16	11.7	7.5
225	6.8	25	11.5	6.6

5 Conclusion

In this paper, we presented a TDMA service for sensor networks. As shown in Section 4.2, the service ensures that communication among sensors is collision free. By contrast, previous CSMA based techniques suffer significant collisions.

We considered three types of communication patterns, broadcast, convergecast and local gossip, which occur frequently in sensor networks. We showed how the TDMA service could be customized for each of the communication patterns. As discussed in Section 3, we recommend that if the application requirements are unknown, then the TDMA service for the local gossip be used as the solution for local gossip provides substantial benefit to broadcast and convergecast.

Our solution assumes that the underlying topology is a grid and that each sensor is aware of its location in the grid. We showed that even if the sensor location is perturbed, the TDMA service ensures that the number of collisions is small. Further, we note that it is possible to extend our TDMA service for the case where the underlying topology is not a grid or where some sensors have failed or are sleeping. For reasons of space, we refer the reader to [13] for these extensions.

We have combined the TDMA algorithms proposed in this paper with previous algorithms on clock synchronization (e.g., [15]). TDMA and clock synchronization complement each other. TDMA is useful in ensuring that messages sent for clock synchronization do not collide. And, clock synchronization helps to reduce the clock drift and to ensure that the clock drift does not cause TDMA slots of nearby sensors to overlap.

The TDMA algorithms proposed in this paper use a simple routine such as diffusing computation to (re) validate the slots assigned to a sensor. Further, our algorithms are stabilizing fault-tolerant [16] and, hence, as long as the initiator of the diffusing computation does not fail, the algorithm can recover from states where sensors are improperly initialized or slots assigned to sensors are corrupted or sensors have arbitrary clock drifts. This is achieved by requiring that a sensor *freeze* (i.e., stop transmitting messages) temporarily if it does not receive the required revalidation message. When the revalidation message is received, it can then continue to function correctly. This ensures (cf. [13]) that eventually the revalidation messages will reach all sensors without collisions.

References

[1] A. Woo and D. Culler. A transmission control scheme for media access in sensor networks. *In Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking*, pages 221–235, 2001.

[2] R. Rom and M. Sidi. *Multiple Access Protocols: Performance and Analysis*. Springer-Verlag, 1989. Also available at: `http://www.comnet.technion.ac.il/rom/PDF/MAP.pdf`.

[3] D. E. Culler, J. Hill, P. Buonadonna, R. Szewczyk, and A. Woo. A network-centric approach to embedded software for tiny devices. In *EMSOFT*, volume 2211 of *Lecture Notes in Computer Science*, pages 97–113. Springer, 2001.

[4] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. Pister. System architecture directions for network sensors. *In Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, November 2000.

[5] A. J. Viterbi. *CDMA: Principles of Spread Spectrum Communication*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, 1995.

[6] S. S. Kulkarni and U. Arumugam. Collision-free communication in sensor networks. *In Proceedings of the Sixth Symposium on Self-stabilizing Systems (SSS), Springer*, LNCS:2704:17–31, June 2003.

[7] Crossbow Techonology, Inc. *Mote In-Network Programming User Reference Version 20030315*, 2003. Available at: `http://www.xbow.com/Support/Support_pdf_files/Xnp.pdf`.

[8] The Ohio State University NEST Team. A Line in the Sand (LITeS), A DARPA-NEST Field Experiment. DARPA/US-SOCOM Project Demonstration, Tampa, Florida, August 2003. Experiment report available online at: `http://www.cis.ohio-state.edu/siefast/nest/nest_webpage/ALineInTheSand.html`.

[9] V. Claesson, H. Lonn, and N. Suri. Efficient TDMA synchronization for distributed embedded systems. *In Proceedings of the 20th IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 198–201, October 2001.

[10] K. Sohrabi and G. J. Pottie. Performance of a novel self-organization protocol for wireless ad-hoc sensor networks. *In Proceedings of the IEEE Vehicular Technology Conference*, pages 1222–1226, 1999.

[11] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1(4):660–670, October 2002.

[12] K. Arisha, M. Youssef, and M. Younis. Energy-aware TDMA-based MAC for sensor networks. *In Proceedings of the IEEE Workshop on Integrated Management of Power Aware Communications, Computing and Networking (IMPACCT)*, May 2002.

[13] U. Arumugam. Collision-free communication in sensor networks. Master's thesis, Michigan State University, Draft available at: `http://www.cse.msu.edu/~arumugam/research/MastersThesis/main.ps`, September 2003.

[14] G. Simon, P. Volgyesi, M. Maroti, and A. Ledeczi. Simulation-based optimization of communication protocols for large-scale wireless sensors networks. *In Proceedings of the IEEE Aerospace Conference*, March 2003. Simulator (*Prowler*) available at: `http://www.isis.vanderbilt.edu/projects/nest/prowler`.

[15] T. Herman. NestArch: Prototype time synchronization service. NEST Challenge Architecture. Available at: `http://www.ai.mit.edu/people/sombrero/nestwiki/index/ComponentTimeSync`, January 2003.

[16] E. W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Communications of the ACM*, 17(11), 1974.