Radial Coordination for Convergecast in Wireless Sensor Networks

Qingfeng Huang and Ying Zhang
Palo Alto Research Center (PARC) Inc.
3333 Coyote Hill Road
Palo Alto, CA 94304, USA
Emails:{qhuang, yzhang}@parc.com

Abstract

Convergecast is frequently used for supporting various tasks in sensor network applications. In this paper, we investigate the problem of packet loss in the convergecast process in wireless sensor networks due to congestion and collisions near the sink, and propose a simple yet powerful coordinated convergecast protocol for achieving high convergecast reliability. We study the performance of the new protocol via simulation and show the tradeoffs among reliability, latency, throughput and energy consumption. To compare this protocol with other existing convergecast protocols, a real application scenario is used. Simulation shows that this protocol can achieve dramatic improvement in reliability comparing to existing convergecast protocols.

1. Introduction

Many sensor network applications require broadcast and convergecast for data dissemination or collection. Convergecast refers to a communication pattern in which the flow of data from a set of nodes to a single node in the network. One may view convergecast as the opposite to broadcast or multicast, in which the flow of data is from a single node to a set of nodes in the network. Figure 1 is a simple example that illustrates the characteristics of the broadcast and convergecast in a network. On the left is a broadcast example in which node A is the message source and nodes B, C, D are the expected recipients. B hears A's message directly, and forwards a copy of the message to nodes C and D. On the right is a convergecast example in which node A is the destination node. Nodes B, C, and D each have a message destined to node A. Node B serves as a relay node for nodes C and D.

The issue of designing efficient and reliable broadcast protocol for wireless ad hoc networks has received much attention in recent years [8, 9, 7, 12, 2, 13]. However, the issue of convergecast has only been addressed by few [3, 1].

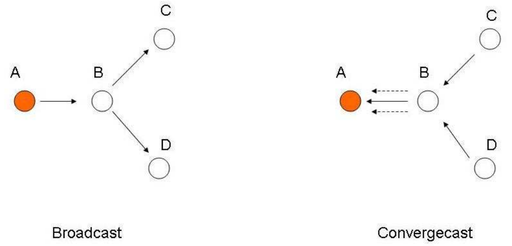


Figure 1. A Simple Example of Broadcast and Convergecast

At the first glance, it appears that convergecast is a reverse process of broadcast. However, there is a critical difference between convergecast and broadcast that makes convergecast to be more than a reverse process of broadcast. In a broadcast, the datum being disseminated is identical to all recipients. As a result, the datum only need to traverse any edge in the network at most once, and the bandwidth utilization for a broadcast process is essentially uniform across the network. On the other hand, convergecast is a many-to-one process and the data flowing to the recipient from different nodes are likely to be different. The result is a non-uniform bandwidth demand across the network space: more bandwidth is needed for nodes closer to the sink, as seen in the example in Figure 1. In the convergecast process, edge BA is traversed by three packets while CB and DB are traversed by one packet. A general analysis in Section 2 shows that the sink node is inevitably an ultimate bottleneck in convergecast. In a large-scale wireless network, this bottleneck phenomenon in convergecast potentially impacts its reliability and throughput, and demands special attention.

In this paper, we investigate strategies for reliable and efficient convergecast in wireless sensor networks. Note that in wireless networks, collision is a major adversary to communication reliability and wastes resource (energy, bandwidth, time). Collisions occur when multiple nodes simultaneously transmit over the same channel to the same node or a receiver is in the range of another transmission over the same channel. Many schemes, especially MAC level ones, such as CSMA, CSMA/CA, FDMA, CDMA, TDMA, have been proposed and used to reduce collisions in wireless communication. The concern of simplicity and scalability for large scale wireless sensor networks has lead to the adoption of CSMA [14][15] and its close variants [16][18] as the MAC layer strategies in the popular Mica mote platform. For the same reason, in this paper we focus on investigating convergecast mechanisms on top of CSMA type MAC layer. It is well-known that in a CSMA type scheme, packet-loss-ratio increases when the load on a link increases. This is also recently observed by Zhao et al. [18] on the Mica mote platform. This points to a potential problem in convergecast: without proper data flow control or coordination, the contention at the bottleneck could dramatically increase packet loss and reduce the reliability of convergecast, which is also confirmed in our simulation study.

Our main contribution in this work is a simple convergecast coordination strategy called “radial coordination” in which the nodes in the a convergecast process adjust their transmission time according to a quadratic formula based on the estimated hop distance to the sink in a 2D network. Our simulation results show that this radial coordination combined with a constrained flooding routing strategy dramatically increases success rates in the convergecast process and there is a controllable parameter that one can adjust for the tradeoff among success rates, latency, throughput and energy consumption.

The remainder of the paper is organized as follows. Section 2 presents the network model in this study. Section 3 presents our convergecast algorithm. Section 4 describes our simulation method and simulation results. Section 5 discusses alternatives and future work, followed by a conclusion section.

2. Motivation and Model

2.1. Motivation

Access of information in a large-scale sensor networks often involves a broadcast process followed by a convergecast process. In the broadcast process, the query for specific information is sent from the query node to all nodes in the network that potentially have the matching information. In the convergecast process, all nodes having the relevant information send their answers to the querying node. For convenience, henceforth we call the querying node as the “root” node. The streams of data converging to the root create a flood of information to the root and an uneven demand of bandwidth across the space, which leads to the need for proper coordination in the convergecast process to reduce potential collisions.

Let’s use a simple example in a uniformly distributed network to see how the load is distributed in space in a convergecast process. Assume the root is interested in the detailed readings of every sensor node in the network within certain distance R_{max}. It sends out a query using spatially constrained broadcast (geocast). All nodes within the circle of radius R_{max} reply with their readings. Note that nodes closer to the root need to rely the data for nodes further away. The number of relevant nodes outside a circle of radius R is proportional to $R_{max}^2 - R^2$ in a uniformed distributed network. So the amount of data that the nodes between $R - r$ and $R + r$ needs to reply is proportional to $R_{max}^2 - R^2$, where r is the communication range of the sensor nodes. Note that in a wireless network, the max amount of bandwidth available in an area is roughly proportional to the size of the area. So the bandwidth available in the area between $R - r$ and $R + r$ is proportional to $(R + r)^2 - (R - r)^2 = 4Rr$. Assume the data outside pass through the area evenly, then the data load per unit area is

$$L(R) \propto \frac{R_{max}^2 - R^2}{4Rr} \qquad (1)$$

We can see that as R gets smaller, the load increases fast.

It is well-known that the probability of collision and packet loss increases as the communication load increase on the CSMA MAC layer. A goal of this work is to find a good strategy to reduce the packet loss by exploiting the characteristics of convergecast. In this paper, we first consider the case in which every node in the network needs to participate and no data aggregation or piggy-backing process is present in the convergecast process. The potential impact of data aggregation and piggy-backing to our model of study is marginal and is discussed in Section 5.

2.2. Model

Convergecast were traditionally investigated in networks with symmetric links, i.e., a node A can hear another node B directly guarantees B can hear A directly. In such a network, a broadcast tree rooted at the query node could be built in the broadcast process and be used by the convergecast process for the routing of answers to the query nodes. The answer available in a node can be routed to the root by simply reverse the path the corresponding query traversed.

Our simulation of the convergecast coordination strategy is based on a probabilistic radio propagation model and the CSMA MAC protocol for Mica motes. Such sensor networks have been demonstrated to have asymmetric links. Therefore the broadcast tree is better viewed as a directed graph. The reverse path for each path on the broadcast tree might not exist. As a result, the conventional reverse traversal scheme for convergecast might not work, without extra symmetric link maintenance or data acknowledgement. In fact, we will show that a version of constrained flooding works better than tree-based convergecast for this type of network, with or without radial coordination.

We use a radio propagation model provided by the Prowler [10] network simulator. It attempts to simulate the probabilistic nature in wireless sensor communication observed by many [18][15]. The propagation model determines the strength of a transmitted signal at a particular point of the space for all transmitters in the system. Based on this information the signal reception conditions for the receivers can be evaluated and collisions can be detected. The transmission model is given by:

$$P_{rec,ideal}(d) \leftarrow P_{transmit}\frac{1}{1+d^{\gamma}}, \quad where\ 2 \le \gamma \le 4 \qquad (2)$$

$$P_{rec}(i,j) \leftarrow P_{rec,ideal}(d_{i,j})(1+\alpha(i,j))(1+\beta(t)) \qquad (3)$$

where $P_{transmit}$ is the signal strength at the transmitter and $P_{rec,ideal}(d)$ is the *ideal* received signal strength at distance d, α and β are random variables with normal distributions $N(0,\sigma_{\alpha})$ and $N(0,\sigma_{\beta})$, respectively. A network is asymmetric if $\sigma_{\alpha} > 0$ or $\sigma_{\beta} > 0$. A node j can receive a packet from node i if $P_{rec}(i,j) > \Delta$ where $\Delta > 0$ is the threshold. There is a collision if two transmissions overlap in time and both could be received successfully. Furthermore, an additional parameter p_{error} models the probability of a transmission error caused by any other reasons.

Since the CSMA MAC model and its characteristic is relatively well-known, we omit the detailed description due to page limit.

3. Convergecast with Radial Coordination

Our convergecast protocol contains two essential elements. The first is a hop-distance based temporal coordination heuristic. The second is a constrained flooding mechanism. Note that convergecast is often appear in two types of scenarios. One is query-driven convergecast, the other is event-driven convergecast. In a query-driven scenario, a convergecast takes place after a query is broadcasted. In a event-driven scenario, a convergecast takes place after an event occurs. The events include physical events such as a gunshot is heard by the sensors, and temporal triggers such as periodic snapshots of the network state. While the coordination strategy used is the same, the details in the temporal coordination is slightly different for the two scenarios. In the next we use the query-driven example for describing our algorithm detail and point out the difference in an event-driven scenario.

3.1. Temporal Coordination Heuristic

Temporal coordination is our key strategy to reduce the collision and packet loss. Rather than sending a reply immediately after receiving the query, a node i in the network will wait for time T_i before sending the reply. We expect this to help reduce the congestion around the root node in the convergecast process. The question is then how does each node i decide what is their T_i. We suggest to use the “root-to-me” hop-distance available from the proceeding query broadcast as a variable, and let T_i be $T(h_i)$ where h_i is the hop distance from the root to node i in the broadcast tree. Note that although for asymmetric networks, “root-to-me” and “me-to-root” may be different, it should not change this coordination scheme in a significant way. We call $T(h)$ the *coordination function* or the “nice” function.

The form of the coordination function $T(h)$ depends on the topology of the network. The intuition is that, the further away a node is from the root, the more nodes need to send a reply before it does, so the longer it should wait to avoid unnecessary collisions. How many more nodes there are depend on the topology of the network. In the next we first derive it on a 2-D square-lattice network, then a general formula is developed for a randomly deployed network.

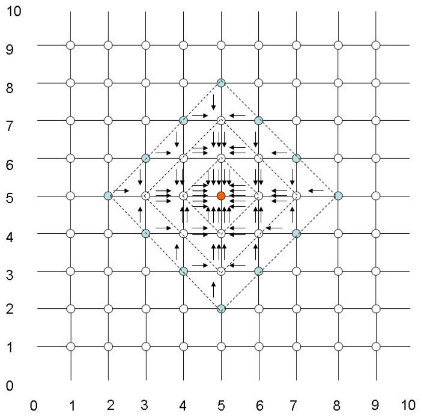


Figure 2. An Example of Convergecast on a Square-lattice Network

3.1.1. Square-Lattice Network

In a square-lattice network, as shown in Figure 2, the number of nodes that are h-hops away from a sink is $4h$. The total number of nodes that are less than h-hops away from the sink is

$$4 + 8 + 12 + ... + 4(h-1) = 2h(h-1) \qquad (4)$$

Assume each node sends a packet to the sink and no data aggregation is possible (e.g., the data is not compressible) or present, also assume the average packet transmission time for one hop is τ, we have the expected transmission time for all the packets initiated in less than h hops away from the sink to be received at the sink:

$$t(h) = 2h(h-1)\tau \tag{5}$$

As a result, if the packets that are h-hops away wait $t(h)$ before start sending, the potential of collision with the transmissions of packets that are generated less than h-hops away could be reduced significantly comparing to sending the response right after receiving the query.

Consider the following two additional factors we can get a more accurate timing estimate.

- On average it took $h\tau$ time for a node h-hops away from the root to receive the query. Note that the nodes one hop away could try to send the reply immediately after receiving the query. As a result, if all transmissions are coordinated well, during the $h\tau$ time, $h-1$ packets could have been received by the root. Note that this term shows up only in a query-driven scenario, and does not occur in an event-driven scenario.
- The nodes that are h-hops away do not need to wait until all packets less than h-hops away have been received to start transmitting, because their packets could take $h\tau$ time to reach the root. So we can pipeline these packets earlier.

These two factors led us to refine and adjust the waiting time formula to be as follows

$$t(h) = 2h(h-1)\tau - (h-1)\tau - h\tau \tag{6}$$

(Note in an event-driven convergecast scenario, the second term does not exist. So $t(h) = 2h(h-1)\tau - h\tau$, instead. Note also that in a event-driven convergecast scenario, the prior knowledge of the hop count h can come from a bootstrapping phase, so this formula based on hop count is still practically applicable.)

3.1.2. General Uniform Networks In general, assume the density of the network is d, namely, each node has d neighbors, the total number of nodes that are less than h-hops away from the sink is

$$d + 2d + 3d + ... + d(h-1) = \frac{d}{2}h(h-1) \tag{7}$$

and the general waiting time formula becomes

$$t(h) = \frac{d}{2}h(h-1)\tau - (h-1)\tau - h\tau \tag{8}$$

One can apply a quadratic form transmission delay function for replying to a query or responding to an event in any 2D uniformly deployed network as:

$$t(h) = ah^2 + bh + c \tag{9}$$

where a, b and c are constants that can be obtained offline given the application scenario (e.g., query-based or event-driven, density (d), bandwidth (τ), etc.).

However, according to this formula, all the nodes at h-hops away would send out at the same time, which will cause collision at the sink, even though all packets sent out by the nodes that are less than h-hops away have been received by the sink by that time. To take this factor into account, we added an additional random offset:

$$T(h) = t(h) + rdh\tau \tag{10}$$

where r is a uniformly distributed random number from 0 to 1, and dh corresponds to the number of nodes at distance h for a uniformly distributed network. Replace $t(h)$ in (10) by the formular in (8), and when d is large, we have

$$T(h) = [\frac{h-1}{2} + r]dh\tau \tag{11}$$

3.1.3. Local Coordination Heuristic A local coordination function can be obtained even when the network is not uniformly deployed. In this case, formula (11) still can be used as a local coordination function, where d is the number of neighbors of this node. The coordination strategy is totally local, depending only the information at this node. In practice, τ can be either fixed offline, to be sent with the query, or to be adaptive online. In simulation (Section 4), we will show that one can use τ as tunable parameter for tradeoffs among latency, throughput, success rates, and energy consumption.

3.2. Constrained Flooding

Constrained flooding is a routing strategy that we have developed for wireless sensor networks [17]; it has been applied to various applications including convergecast and shown to have overall high performance metrics, in terms of high success rates and low energy consumption. Even though the temporal coordination heuristic described in the previous section can be applied to tree-based convergecasting as well, in our simulation experiments, it does not achieve as high success rate as the constrained flooding mechanism.

Constrained flooding in general has an initialization phase and a routing phase. The initialization phase establishes a “potential field” by propagating from the sink. In query-based scenarios, the initialization is at the time of query broadcasting. In event-driven scenarios, the initialization is the bootstrapping from the base station. Each node maintains the “distance” value from the node to the sink. This value is obtained at the initialization and can also be learnt during the routing process [17]. During the routing process, each node will rebroadcast the received packet only if the node is not too much "further" away from the sink than its sender. Furthermore, a delay is added so that the node with "shorter distance" will transmit earlier and a probabilistic suppression mechanism is applied. It works as follows. Each packet contains a number indicating how far away the sender node is expected to be from the root. When a node hears a packet which is sent or relayed by a node that is further away, then it will try to forward the packet in a probabilistic manner. It will wait for a certain period of time according to its relative "distance" to the sink before the attempt to forward. Note this waiting period is corresponding to the relative distance rather than the absolute distance we used in the temporal coordination heuristics. If during the waiting time, the node hears the same packet N times, then with probability of $1/N$ it forwards the packet.

Combining the temporal coordination heuristic with constrained flooding, the delay time of transmission is set according to the delay function in (11). The probabilistic suppression mechanism is used to further reduce collision and improve efficiency.

4. Simulation Results

We have simulated the radial coordination strategy described in the previous section in two scenarios, using Prowler [10]. In these experiments, the radio model is set according to Section 2, with $\sigma_\alpha = 0.45$, $\sigma_\beta = 0.02$, and $p_{error} = 0.05$. The threshold reception is set to be $\Delta = 0.1$.

The first experiment is on a uniformly deployed network with all the nodes sending one packet almost at the same time (within 0.25 seconds). This example is used for theoretically analyzing the tradeoffs among latency, throughput, success rates and energy consumption. The second example is from a real application "shooter localization" [11] where the network topology and traces are obtained from the real experiments.

In the next, we first describe the performance metrics that are used for comparisons of routing strategies.

4.1. Performance Metrics

We have developed a set of performance metrics for comparing different routing algorithms, including latency, throughput, success rates, energy consumption, etc.

- *Latency*: The latency of a packet measures the time delay of the packet from the source to the destination. For any destination, if n packets have arrived, latency for that destination is given by $\sum_i d_i/n$, where d_i is the latency of the ith packet. The latency of the network is then averaged by the number of destinations. Note that we use latency rather than the number of hops as the metric since latency consists of not just the number of hops, but also the length of transmission queues, the random delays at the MAC layer, and deliberately added delays in routing algorithms to avoid collisions.
- *Throughput*: Throughput at a destination measures the number of packets per second received at the destination. The throughput of the network is the sum of the throughput from all destinations.
- *Success rate*: The success rate of a network measures the total number of packets received at all the destinations vs. the total number of packets sent from all the sources.
- *Energy consumption*: The energy consumption is the sum of used energy of all the nodes in the network. In the current version, we assume that each transmission consumes energy proportional to its signal strength. Therefore, the energy consumption is proportional to the total number of packets sent in the network. A more complicated energy model can be built as well.

4.2. Performance Tradeoffs using Radial Coordination

The simulation is performed on a uniformly deployed 10x10 network, as shown in Figure 3. Note that not only the connectivity is not symmetric ($\sigma_\alpha = 0.45$), but also there is a small dynamic variance ($\sigma_\beta = 0.02$) and a probability of error ($p_{error} = 0.05$).

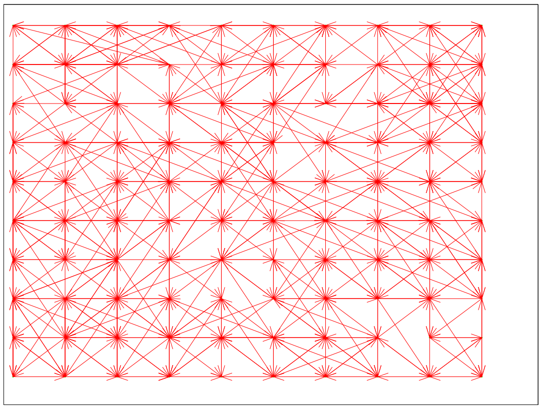


Figure 3. Snapshot of Connectivity for a 10x10 Network

Assume that every node except the sink needs to send out a packet within 0.25 second time frame. Using the radial coordination scheme, the actual sending is delayed according to the temporal coordination heuristic in (11). The delay of transmission is recorded in the latency metric. By varying the delay coefficient τ in formula (11), one can control the latency and the success rate. A total of 10 random runs were conducted and Figures 4 (a)-(d) show the performance metrics with respect to the delay coefficient τ, which is varying from 0 to 9000. The results show that with the increase of τ, the success rate increases. However, the latency also also increases and the throughput decreases. Without the radial coordination (i.e. $\tau = 0$), the success rate is only less than 10 percent, and with radial coordination, the success rate can go above 90 percent. For different applications with different performance requirements, one can select the best τ that fits the purpose. For example, if high success rates are important, one may want to increase τ; on the other hand, if the high throughput or low latency is important, one should choose a small τ instead.

4.3. Shooter Localization

Shooter localization [11] is a real example of the event-driven type of convergecast; the application is used for getting all the acoustic data that are above certain threshold to the base station, so that the source of the sound can be located. In this experiment, the network topology (Figure 5), the sink location and the event trace data are obtained from the actual experiments on the hardware platform. The network is about 10 hops.

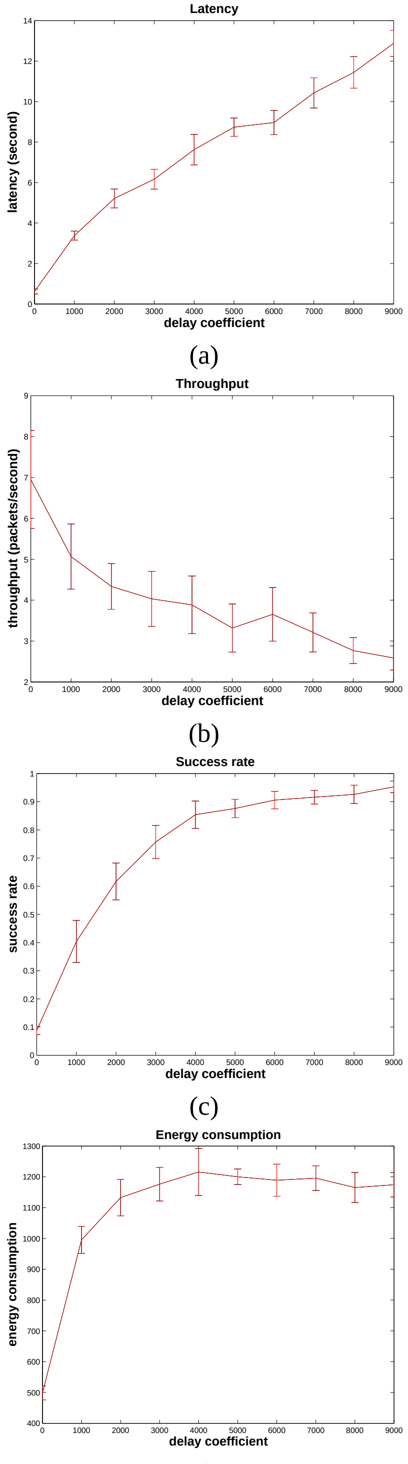
Three algorithms are compared for this scenario:

- Gradient flooding with duplication: this algorithm is actually used in the shooter localization application, implemented in TinyOS/NesC. Like constrained flooding, a potential field is generated in the initialization phase and data flow to the sink in the direction of the potential field. No data suppression mechanism is used, however, each packet may be transmitted up to 3 times to increase the reliability.
- Backbone tree: this algorithm uses directed diffusion [6] to create a backbone tree [5] in the initialization phase and passes packets to parents during routing.
- Radial coordination: apply radial coordination presented in this paper to reduce collision, with τ set to be 1000 and 4000.

Figure 6 shows the latency (a), throughput (b), success rate (b) and energy consumption (d) for all the algorithms applied to this application scenario. We can see that (1) tree-based convergecast does not work as well as flooding-based strategies for this application, and (2) by sacrificing the latency metric, one can get higher success rates.

5. Discussion and Future Work

We have performed various tests on the first experiment (the 10 x 10 uniformly deployed network). We have



(a)

(b)

(c)

(d)

Figure 4. Tradeoff of Performance Metrics using Radial Coordination

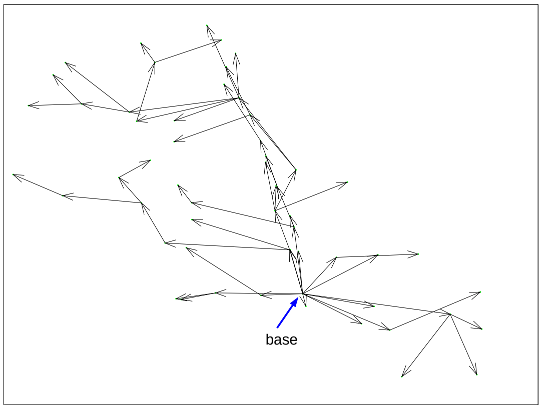


Figure 5. Node Distribution in Shooter Localization Experiment

added radial coordination to tree-based convergecast as well. However, for the tree-based convergecast, even with τ going up to 9000, the maximum success rate is still less than 0.5. For the results shown in the previous section, we have applied radial coordination to both original and forwarding packets. We have also tested applying radial coordination to original but not forwarding packets. However, the maximum success rate is less than 0.5. By changing the distribution of the random variable r in formula (11) from $[0, 1)$ to $[0, \delta)$ where $\delta > 0$, the performance changes as well. By removing the random delay term we can only get the success rate up to 0.7. We also found that by varying signal strength/network density, the performance curves have the same shapes, although energy consumption and latency may shift up or down. The denser the network, the higher the energy consumption and the smaller the latency.

Radial coordination is a general mechanism that can be used with other routing strategies, for example, use with packet duplication and aggregation. When data compression and aggregation is present in the convergecast process, the exact form for the coordination function depends on the compression properties of the aggregation operators used. Application specific analysis of appropriate temporal coordination functions in the presence of data aggregation is a topic for future research. For simplicity, we have only varied one variable τ to obtain the tradeoff curves. By varying more variables in the general coordination formula in (9), one will get better performance curves overall. Given an estimated communication pattern, one can obtain the coefficients by an offline optimization procedure.

The impact of unreliable wireless link on convergecast has recently been addressed in [1]. A TDMA-based approach is investigated in [1] and a heuristic convergecast tree construction and channel allocation algorithm is proposed. Our study complements the previous work by investigating efficient and reliable convergecast protocols on CSMA MAC layer. Very recently, a TCP like congestion control mechanism [4] has been proposed to reduce the congestion in convergecast scenarios. Our approach may be viewed as a congestion prevention scheme that complements the reactive control scheme.

Our investigation has been focused on 2D networks. In a one dimensional network, one can derive the coordination function which is linear to the hop distance. Similarly, in a 3-dimensional network, the coordination function shall take a cubic form on the hop distance. Experimental validation of these cases is left for future work.

In addition, we have only investigated the benefit of radial coordination for convergecast on top of the CSMA MAC layer. Future work includes the the study of its impact on convergecast on other types of MAC layers such as S-MAC and TDMA, and how the scheme should interact with power control strategies.

6. Conclusion

Convergecast is frequently used for supporting various tasks in sensor network applications yet its optimization has not been extensively studied. In this paper, we investigate the problem of packet loss in convergecast process in large scale networks due to congestion and collisions near the sink, and propose a coordinated convergecast protocol for achieving higher convergecast reliability. We studied the performance of the new protocol via simulation and show the tradeoffs among reliability, latency, throughput and energy consumption. This protocol is compared with other existing convergecast protocols and shows dramatic improvements. We also demonstrated in a real application scenario this protocol achieved dramatic improvement in reliability comparing to existing convergecast protocols.

Acknowledgement: This work is funded in part by the Defense Advanced Research Project Agency contract# F33615-01-C-1904.

References

[1] V. Annamalai, S. Gupta, and L. Schwiebert. On tree-based convergecasting in wireless sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, volume 3, page 1942, 2003.

[2] J. Cartigny, D. Simplot, and I. Stojmenovic. Localized minimum-energy broadcasting in ad-hoc networks. Technical Report 8, IRCICA/LIFL, Univ. Lille 1, Lille, France, 2002.

[3] I. Chlamtac and S. Kutten. Tree-based broadcasting in multihop radio networks. *IEEE Transactions on Computers*, 36(10):1209–1223, 1987.

[4] C. T. Ee and R. Bajcsy. Congestion control and fairness for many-to-one routing in sensor networks. In *ACM SenSys04 (to appear)*, 2004.

[5] T. He, S. Krishnamurthy, J. A. Stankovic, and T. F. Abdelzaher. Energy-efficient surveillance system using wireless sensor networks. In *MobiSys04*, June 2004.

[6] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proc. 6th Int'l Conf. on Mobile Computing and Networks (ACM Mobicom)*, Boston, MA, 2000.

[7] H. Lim and C. Kim. Multicast tree construction and flooding in wireless ad hoc networks. In *Proceedings of 3rd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2000.

[8] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 152–162, August 1999.

[9] W. Peng and X. Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. In *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, 2000.

[10] G. Simon. Probabilistic wireless network simulator. http://www.isis.vanderbilt.edu/projects/nest/prowler/.

[11] G. Simon, A. Ledeczi, and M. Maroti. Sensor network-based countersniper systems. In *ACM SenSys04 (to appear)*, 2004.

[12] I. Stojmenovic, M. Seddigh, and J. D. Zunic. Internal nodes based broadcasting in wireless networks. In *HICSS*, 2001.

[13] B. Williams and T. Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, pages 194–205, 2002.

[14] A. Woo and D. E. Culler. A transmission control scheme for media access in sensor networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 221–235. ACM Press, 2001.

[15] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of the First International Conference on Embedded Networked Sensor Systems*, pages 14–27, Los Angeles, Nov 2003.

[16] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, New York, NY, USA, June 2002.

[17] Y. Zhang and M. Fromherz. Message-initiated constraint-based routing for wireless ad-hoc sensor networks. In *Proc. IEEE Consumer Communication and Networking Conference*, 2004.

[18] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the First International Conference on Embedded Networked Sensor Systems*, pages 1–13, Los Angeles, Nov 2003.

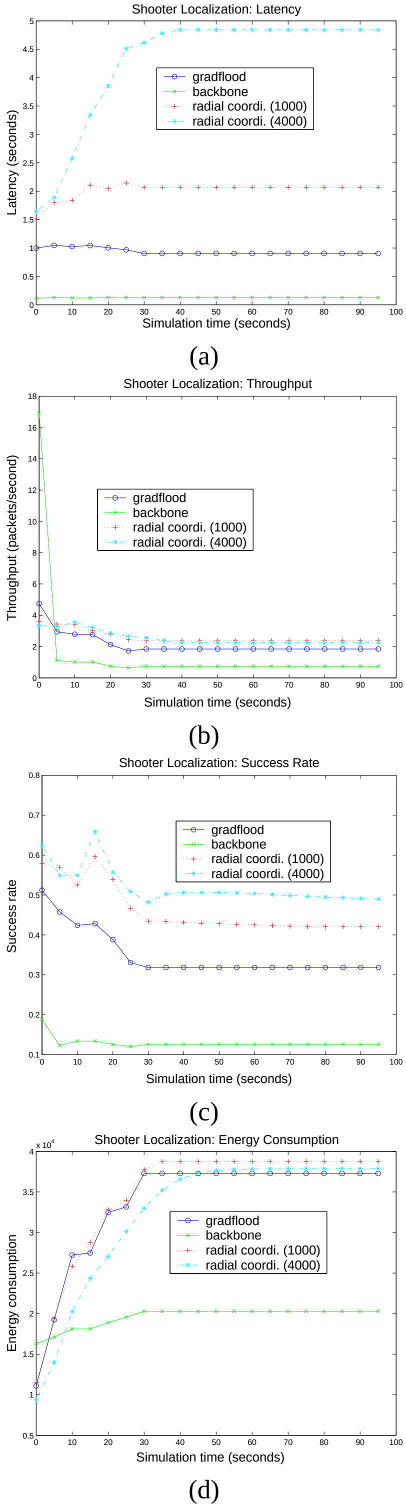


Figure 6. Performance Comparison for Shooter Localization: (a) Latency (b) Throughput (c) Success Rate (d) Energy consumption