

COORDINATED CONVERGECAST IN WIRELESS SENSOR NETWORKS

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

ABSTRACT

Convergecast is a common communication pattern across many sensor network applications featuring data-flows from many different source nodes to a single sink node during a very short time window. Convergecast in wireless ad hoc network requires proper coordination among the nodes to avoid high packet collision rates near the sink node. A coordinated convergecast framework for achieving high convergecast reliability has been developed. In this paper, we extend this framework and study the effectiveness of packet aggregation and duplication within this framework. We analyze the performance of the coordination strategies via simulation and show the tradeoffs among reliability, latency, and throughput. We also compare a set of coordinated convergecast strategies with the existing routing strategy designed and implemented for a real-world application.

I. INTRODUCTION

Convergecast refers to a communication pattern in which the flow of data from a set of nodes to a single node in the network. Many sensor network applications require convergecast for data collection. For example, consider a sensor network deployed to determine the location of a sniper in an urban environment. In general sensor nodes have limited computational capability. As a result, they cannot execute sophisticated algorithms to determine the location from which the shot was fired [9]. In such applications all the packets generated in the network have to reach the base station for further analysis.

The issue of convergecast has only been addressed by few [3], [1], [5]. Convergecast is a many-to-one process which results in a non-uniform bandwidth demand across the network space: more bandwidth is needed for nodes closer to the sink. 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.

A simple and effective convergecast in wireless sensor networks has recently been proposed [5], in which the nodes in the 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 main contribution in this paper is an extension of this simple convergecast coordination framework. The extension includes an inward delay mechanism and optional packet aggregation and duplication mechanisms within a layered routing architecture. Our simulation results show that this inward delay mechanism works better than the original outward delay mechanism for some performance metrics, aggregation further improves the overall performance of convergecast, and duplication increases delivery rates with higher energy cost.

The remainder of the paper is organized as follows. Section II presents the extended convergecast framework. Section III describes packet aggregation and duplication in a layered routing architecture. Section IV discusses the simulation model and analyzes simulation results. Section V concludes the paper with some future work along this direction.

II. CONVERGECAST WITH RADIAL COORDINATION

The main part of the convergecast protocol is a hop-distance based temporal coordination heuristic for adding transmission delays to avoid collisions. In this section, we will first introduce the framework and present two delay strategies: outward and inward delays.

A. Temporal Coordination Heuristic

Temporal coordination is our key strategy to reduce the collision and packet loss. Rather than sending or relay a packet immediately, a node v in the network will wait for time T_v before sending the packet through the radio. We expect this to help reduce the congestion around the sink node in the convergecast process. The question is then how does each node v decide what is their T_v . Assume the “sink-to-me” hop-distance h_v is

available from the network initialization phase or from the query broadcast phase, let T_v be $T(h_v)$ where T is a delay coordination function. Note that although for asymmetric networks, “sink-to-me” and “me-to-sink” may be different, it should not change this coordination scheme in a significant way.

For the outward delay strategy, inner nodes start sending packets first, the further away a node is from the sink, the more nodes need to send a packet before it does, so the longer it should wait to avoid unnecessary collisions. For the inward delay strategy, outer nodes start sending packets first, while the packets of inner nodes are delayed to avoid collisions. In either case, the form of the coordination function $T(h)$ depends on the topology of the network.

B. Delay Coordination Functions

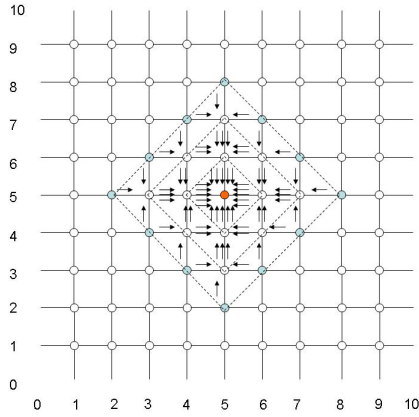


Fig. 1. An Example of Convergecast on a Square-lattice Network

Figure 1 shows a square-lattice network. In this network, each node has 4 neighbors and the number of nodes that are h -hops away from a sink is $4h$. In general, if sensor nodes are uniformly deployed (connectivity forming a regular graph) and the density of the network is d , namely, each node has d neighbors, the number of nodes that are h -hops away from a sink is dh .

Assume each node sends a packet to the sink and no packet aggregation is possible or present, also assume the average packet transmission time for one hop is τ .

For the outward delay scheme, transmissions are started from inside to outside. Since the total number of nodes that are less than h -hops away from the sink is

$$d + 2d + 3d + \dots + d(h-1) = \frac{d}{2}h(h-1), \quad (1)$$

the minimum expected transmission time for all the packets initiated in less than h hops away from the sink

to be received at the sink is:

$$t_o(h) = \frac{d}{2}h(h-1)\tau. \quad (2)$$

As a result, if the nodes that are h -hops away wait $t_o(h)$ 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 packets immediately.

For the inward delay scheme, transmissions are started from outside to inside. Assume H is the hop-distance of the outmost nodes. The total number of nodes that are larger than h -hops away from the sink is

$$d(h+1) + d(h+2) \dots + dH = \frac{d}{2}(h+H+1)(H-h), \quad (3)$$

the minimum expected transmission time for all the packets initiated in larger than h hops away from the sink to be received at the sink is:

$$t_i(h) = \frac{d}{2}(h+H+1)(H-h)\tau. \quad (4)$$

As a result, if the nodes that are h -hops away wait $t_i(h)$ before start sending, the potential of collision with the transmissions of packets that are generated larger than h -hops away could be reduced significantly comparing to sending the packets immediately.

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:

$$t(h) = ah^2 + bh + c \quad (5)$$

where a , b and c are constants that can be obtained off-line given the application scenario (e.g., 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 as well, even when all packets sent out by the nodes at other hop-distance away will have no collision with packets sent by these nodes. To take this factor into account, we added an additional random offset:

$$T(h) = t(h) + rdh\tau \quad (6)$$

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 (6) by the formula in (2), we have the outward delay formula as:

$$T_o(h) = \left[\frac{h-1}{2} + r \right] dh\tau \quad (7)$$

Replace $t(h)$ in (6) by the formula in (4), we have the inward delay formula as:

$$T_i(h) = \left[\frac{(h + H + 1)(H - h)}{2} + rh \right] d\tau \quad (8)$$

C. Local Coordination Heuristic

A local coordination function can be obtained even when the network is not uniformly deployed. In this case, formula (7) or (8) 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 on the information at this node. In practice, τ can be either fixed off-line, to be sent with the query, or to be adaptive online. In simulation (Section IV), we will show that one can use τ as tunable parameter for tradeoffs among latency, throughput and delivery ratios.

III. LAYERED ROUTING ARCHITECTURE

In this section, we will first introduce a layered routing architecture and then present the routing components that can be used to form a convergecast protocol using the delay strategies described in the previous section.

A. Layered Routing Architecture

Our layered routing architecture is not related to the IEEE seven-layer OSI network architecture. This layered architecture was inspired by the component wiring structure in NesC/TinyOS [2] and was in fact implemented initially in TinyOS and later in EmStar [4].

A routing component is a module which handles events (e.g., *Packet_Received*, *Packet_Sent*, *Clock_Tick*) and executes commands (e.g., *Send_Packet*). A routing component A is wired to another routing component B , denoted $A \rightarrow B$, if A passes commands to B and B passes events to A . The minimum configuration of layers includes the MAC layer, a routing layer, and the application layer, with the MAC layer at the bottom and the application layer at the top. The MAC layer performs the low-level communication protocol (e.g., GenericComm in TinyOS). The routing layer forwards the received packet (via broadcast or selectively to the next hop) or passes the received packet up if it is the destination of that packet. The application layer generates the application scenarios. This layered architecture allows for the sharing of common components by different algorithms. For example, many algorithms need neighborhood management and transmission queues, and flooding-based routing may add transmission delays to forwarding packets to reduce collisions. For a given application scenario, one should select the right algorithm with a set of routing components. It is the

algorithm designer's choice to put individual functions into different layers so that common functions can be shared by different algorithms.

B. Basic Convergecast Layers

In the layered architecture, the basic convergecast protocols have the following layers: *application* \rightarrow *initialization* \rightarrow *convergecast* \rightarrow *delay_transmit* \rightarrow *transmit_queue* \rightarrow *MAC*.

- *initialization*: One of the common components for initialization is flooding from the sink node if the sink node is known initially. Flooding from the sink node can be used to generate hop distances to the sink or a spanning tree rooted at the sink node.
- *convergecast*: There are two ways a convergecast can be implemented: (1) using a convergecast tree, which can be established during the initialization phase or the query broadcast phase, or (2) using directional flooding [7], i.e., flooding towards the sink according to the hop distances. We have done experiments on both and found that directional flooding works a lot better in noisy radio environments where the lost of packets is significant. We present here the flooding strategies only. Each node maintains the hop distance to the sink node, and each packet is transmitted with the hop distance of the node. A packet will be rebroadcast if and only if the hop distance of this node is no larger than that of the sender. If the packet will be rebroadcast, the delay time will be calculated according to the delay coordination function.
- *delay_transmit*: This layer maintains a timeout queue and the counts for all the packets it has heard. Each packet sending from this layer will be put into the timeout queue, and a timer will be set according to the time delay of the packet. When the timer expires, the corresponding packet will be retrieved from the timeout queue. If during the waiting time, the node hears the same packet N times, then with probability of $p(N)$ it forwards the packet, where p is a monotonic non-increase function of N with $p(1) = 1$, $p(\infty) = 0$, and $p(N_1) \geq p(N_2)$ if $N_1 \leq N_2$. This layer can be used with any flooding protocol, e.g., constrained flooding [12][14] or the comb-needle model[6], to reduce collision and increase energy efficiency.
- *transmit_queue*: This layer maintains a buffer of outgoing packets when the radio transceiver is busy sending or receiving.

C. Additional Convergecast Layers

Using the layered architecture, one can incorporate other routing components into the convergecast framework. Components that would improve the convergecast performance include *duplicate_transmit* and *aggregate_queue*.

- *duplicate_transmit*: Duplicated transmission using priority management has been implemented in DFRF [7], a directed flooding routing which has been applied to the application of Shooter Localization [9]. Using this component, each packet has an age field which is zero the first time the packet is heard and will be increased every time the packet is transmitted. The older the age, the lower the priority. Packets in the transmit queue are ordered by the priorities. There is a maximum age, and packets that reach the maximum age will be dropped. Duplicated transmission increases the delivery ratio, at the expense of extra energy consumption. It should only be used in applications where high delivery ratios are critical.
- *aggregate_queue*: Packet aggregation is an efficient mechanism that increases delivery ratios and reduces energy consumption at the same time, if packet sizes are fixed. In this layer, maximum N packets will be assembled into one packet before sending, and a packet will be dissembled to N packets after receiving. However, packet aggregation is only possible when the packet data size is much smaller than the maximum data size of the transmission frame at the MAC layer. In the shooter localization application [9], maximum four packets are aggregated into one packet for transmission.

Adding both layers, a convergecast protocol becomes:
application \rightarrow *initialization* \rightarrow *convergecast* \rightarrow
delay_transmit \rightarrow *duplicate_transmit* \rightarrow
aggregate_queue \rightarrow *MAC*.

IV. SIMULATION RESULTS

Two convergecast experiments are conducted and performances are evaluated. 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, delivery ratios and energy consumption. The second example is from a real application “shooter localization” [9] where the network topology and event traces are obtained from the real experiments.

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

A. Radio Model

Our simulation of the convergecast coordination strategy is performed in Rmase [13], built on Prowler [8] which provides a probabilistic radio propagation model and the CSMA MAC protocol for Mica motes [2].

The radio model attempts to simulate the probabilistic nature in wireless sensor communication observed by many [15][10]. It 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:

$$P_{rec,ideal}(d) \leftarrow P_{transmit} \frac{1}{1+d^\gamma}, \text{ where } 2 \leq \gamma \leq 4 \quad (9)$$

$$P_{rec}(i,j) \leftarrow P_{rec,ideal}(d_{i,j})(1+\alpha(i,j))(1+\beta(t)) \quad (10)$$

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 $\sigma_\beta > 0$. In (10), α is static depending on locations i and j only, and β is dynamic which changes over time. 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. In our experiments, the radio model is set 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 radio transmission rate is set to be 40Kbps, and each packet has a fixed size of 960 bits, i.e., the maximum possible throughput can be as high as 40 packets per second.

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

B. Performance Metrics

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

- *Latency*: The latency of a data packet measures the time delay of the packet from the source to the

sink. If n packets have arrived, latency is given by $\sum_i d_i/n$, where d_i is the latency of the i th packet. 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 of data measures the number of data packets per second received at the sink.
- *Delivery ratio*: The delivery ratio of a network measures the total number of data packets received at the sink vs. the total number of data packets sent from all the sources.
- *Energy consumption*: The energy consumption is the sum of used energy of all the nodes in the network from the beginning, where the *used energy of a node* is the sum of the energy used for communication, including transmitting, receiving, and idling. All the convergecast algorithms in our study do not use sleep schedules and every node uses the same power for transmission. Therefore, the energy consumption is linear to the total number of packets sent in the network. In this simulation, for simplicity, we set energy consumption proportional to the total number of transmissions. A realistic energy model has been built in Rmase and used in research for minimum energy configuration of sensor networks [11].
- *Energy efficiency*: Energy efficiency is defined to be the ratio between the total number of data packets received at the sink vs. the total energy consumption in the network.

C. Performance Tradeoffs using Radial Coordination

The simulation is performed on a uniformly deployed 10x10 network, as shown in Figure 2. 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 ($p_{error} = 0.05$).

Assume that every node except the sink needs to send out a packet within 0.25 second time frame. By varying the delay coefficient τ in formula (7) or (8), one can control the latency and the delivery ratio. In this experiment, four strategies are compared: inward and outward delay coordinations with and without aggregation (where 4 packets are packed into one for transmission). A total of 10 random runs were conducted for each test. Figure 3 show the tradeoffs between latency and delivery ratios by varying the delay coefficient t . For the outward and inward delay schemes, τ is set to be $0.025t$ and $0.0036t$

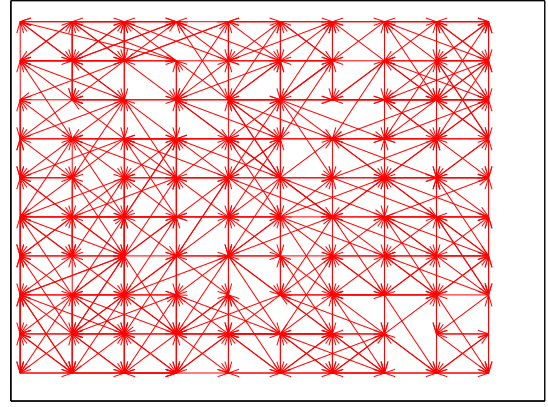


Fig. 2. Snapshot of Connectivity for a 10x10 Network

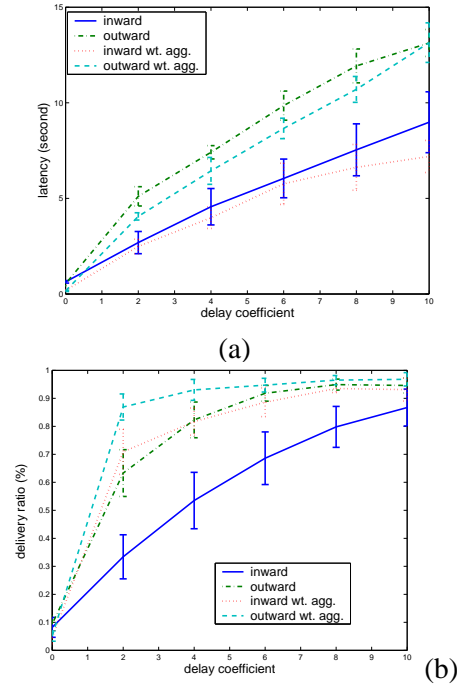


Fig. 3. Performance Tradeoff using Radial Coordination: (a) Latency vs. Delay Coefficient; (b) Delivery ratio vs. Delay Coefficient

seconds, respectively. We see that for all strategies, both latency and delivery ratios increase with the increase of delay coefficients. Without the radial coordination (i.e. $\tau = 0$), the delivery ratio is only less than 10 percent, and with radial coordination, the delivery ratio 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 delivery ratios are important, one may want to increase τ ; on the other hand, if the low latency is important, one should choose a small τ instead.

Figure 4 show the results comparing different strate-

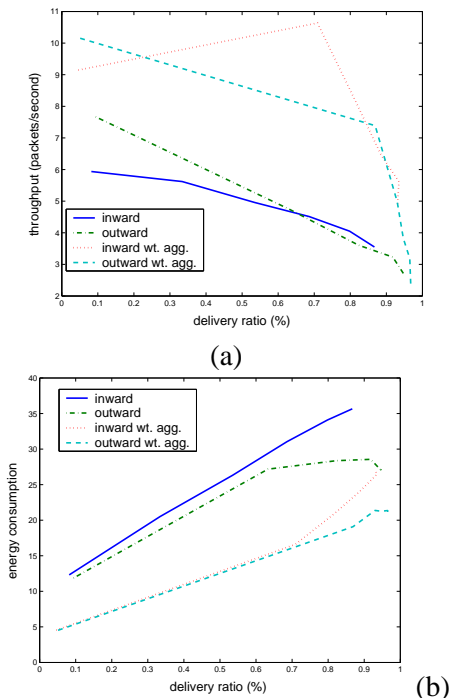


Fig. 4. Performance Comparisons using Radial Coordination: (a) Throughput vs. delivery ratios (b) Energy Consumption vs. delivery ratios

gies under the same delivery ratios. We see that (1) inward strategies have high throughput, and (2) the use of aggregation always improves the performances.

D. Shooter Localization

Shooter localization [9] is a real example 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.

The original algorithm used in shooter localization DFRF [7] is compared with the four convergecast strategies using radial coordination. Similar to our algorithms, the original algorithm, “directional flooding”, uses “hop-distance” to flood the packets to the sink node, where “hop-distance” is generated in the initialization phase. No extra delay is added to packets. Using duplicated transmission, each packet may be transmitted up to 3 times to increase the reliability and 4 packets are aggregated into one before transmission. Four strategies are used to compare with the original algorithm: (1) outward with aggregation, (2) inward with aggregation,

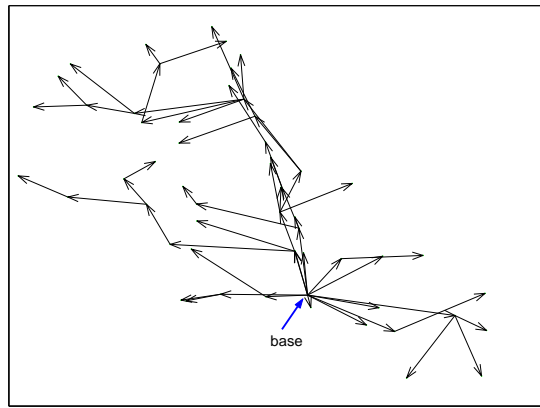


Fig. 5. Node Distribution in Shooter Localization Experiment

(3) outward with aggregation and duplication, and (4) inward with aggregation and duplication. Figure 6 shows the latency (a), throughput (b), delivery ratio (b) and energy efficiency (d) for all the algorithms applied to this application scenario. We can see that (1) outward with aggregation and duplication has the highest delivery ratios, (2) inward has lower latency than that of outward, and (3) inward without duplication has the highest energy efficiency.

V. CONCLUSION

Convergecast is frequently used for supporting various tasks in sensor network applications, yet its performance has not been extensively studied. In this paper, we have investigated the problem of packet loss in convergecast process in large scale networks due to congestion and collisions near the sink, and proposed a coordinated convergecast framework for achieving higher convergecast reliability. Using a layered routing architecture, we have also integrated other routing components such as aggregation for efficiency and duplication for high reliability. We studied the performance of various strategies via simulation and show the tradeoffs among reliability, latency, throughput and energy consumption. We also demonstrated in a real application scenario that radial coordination achieved dramatic improvement in reliability comparing to the original protocol for the application. From our simulation, we see that even though coordinated convergecast improves reliability, its throughput is far from the maximum possible limit. For future work, we will investigate optimal scheduling for approaching maximum convergecast throughput.

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] U. Berkeley. Berkeley wireless embedded systems, 2003. <http://webs.cs.berkeley.edu/>.
- [3] I. Chlamtac and S. Kutten. Tree-based broadcasting in multihop radio networks. *IEEE Transactions on Computers*, 36(10):1209–1223, 1987.
- [4] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, and D. Estrin. EmStar: A software environment for developing and deploying wireless sensor networks. In *USENIX04*, 2004.
- [5] Q. Huang and Y. Zhang. Radial coordination for convergecast in wireless sensor networks. In *Proc. IEEE 1st workshop on Embedded Networked Sensors (EmNeTS-I)*, 2004.
- [6] X. Liu, Q. Huang, and Y. Zhang. Combs, needles, haystacks: Balancing push and pull for discovering in large-scale sensor networks. In *Proc. 2nd ACM International Conference on Embedded Networked Sensor Systems (SenSys04)*, 2004.
- [7] M. Maroti. Directed Flood-Routing Framework. In *Proc. of the 5th International Middleware Conference*, October 2004.
- [8] G. Simon. Probabilistic wireless network simulator. <http://www.isis.vanderbilt.edu/projects/nest/prowler/>.
- [9] G. Simon, M. Maroti, A. Ledeczi, G. Balogh, B. Kusy, and A. Nadas. Sensor network-based countersniper system. In *Proc. 2nd ACM International Conference on Embedded Networked Sensor Systems (SenSys04)*, 2004.
- [10] 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.
- [11] G. Xing, C. Lu, Y. Zhang, Q. Huang, and R. Press. Minimum power configuration in wireless sensor networks. In *Sixth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc05)*, 2005.
- [12] 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.
- [13] Y. Zhang, M. Fromherz, and L. Kuhn. Rmase: Routing modeling application simulation environment. <http://www.parc.com/era/nest/Rmase>.
- [14] Y. Zhang, M. Fromherz, and L. Kuhn. Smart routing with learning-based qos-aware meta-strategies. In *Proc. Quality of Service in the Emerging Networking*, Lecture Notes in Computer Science 3266, 2004.
- [15] 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.

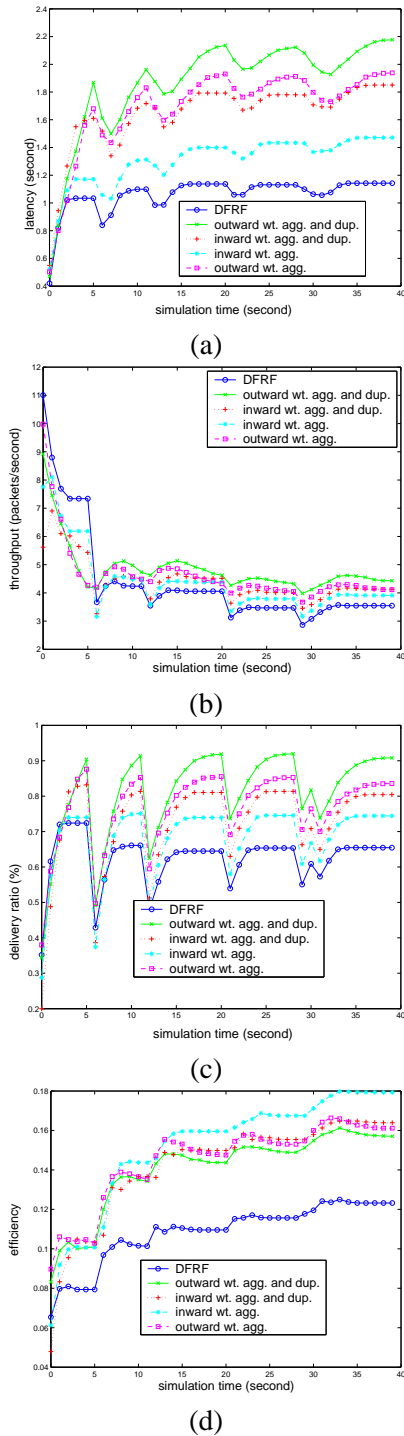


Fig. 6. Performance Comparison for Shooter Localization: (a) Latency (b) Throughput (c) Delivery ratio (d) Energy efficiency

Acknowledgement: This work is funded in part by the Defense Advanced Research Project Agency contract #F33615-01-C-1904. Thanks also to an anonymous reviewer for valuable comments.