

## A ROBUST AND EFFICIENT FLOODING-BASED ROUTING FOR WIRELESS SENSOR NETWORKS

YING ZHANG

MARKUS FROMHERZ

*Palo Alto Research Center (PARC) Inc.  
3333 Coyote Hill Road, Palo Alto, California, U.S.A.  
{yzhang, fromherz}@parc.com*

Received 1 June 2006

Revised 1 September 2006

Flooding protocols for wireless networks in general have been shown to be very inefficient and therefore are mainly used in network initialization or route discovery and maintenance. In this paper, we propose a framework of constrained flooding protocols. The framework incorporates a reinforcement learning kernel, a differential delay mechanism, and a constrained and probabilistic retransmission policy. This type of protocol takes the advantages of robustness from flooding, but maintains energy efficiency by constraining retransmissions. Without the use of any control packets, such a protocol adapts to the specific routing requirements of the task and the dynamic changes of the network. We analyze this framework in simulation using some real-world applications in sensor networks.

*Keywords:* Sensor Networks; Ad-hoc Routing Protocols; Real-time Reinforcement Learning.

### 1. Introduction

Routing in sensor networks has very different characteristics than that in traditional communication networks. First of all, address-based destination specification is replaced by a more general feature-based specification [10], such as geographic location [12] and information gain [5, 35], or a fixed but maybe mobile sink. Secondly, routing metrics are not just shortest delays, but also energy usage [27] and information density [5, 35]. Thirdly, in addition to peer-to-peer communication, multicast (one-to-many) and converge-cast (many-to-one) are major traffic patterns in sensor networks. Even for peer-to-peer communication, routing is more likely to be source or sink driven than table-based [21], and source/sink pairs often are dynamic (changing from time to time) or mobile (moving during routing).

In the last several years, many routing protocols have been developed for wireless sensor networks, including Grid Routing [4], Directed Routing [16], Mint Routing [25], Backbone Routing [7], and the Constraint-based Routing Framework [28, 31]. Most of

these routing protocols have been implemented on Berkeley motes [1], a widely used sensor network platform. We may classify these routing mechanisms into two categories: *structure-based* protocols and *structure-less* protocols. Structure-based protocols generate a routing tree (e.g., Backbone [7], Mint [25], Grid [4], Adaptive Tree [31, 33]), or one or more routing paths (e.g., AODV [17], DSR [11], Multipath routing [6]) before sending data. For dynamic networks, in addition to initialization, periodic or repair-based maintenance is required to keep the structure up to date. Structure-less protocols, instead of building and maintaining a routing structure, establish a "potential field". Data from upstream flows downstream according to the potential field. Similar to structure-based protocols, however, the maintenance of the potential field is necessary for dynamic networks. Flooding (e.g., Directed Routing [16], Gradient Broadcast [26], Gradient Routing [19], Constrained Flooding [30, 31]) and real-time search (e.g., Ant-Routing [3, 34], Q-Routing [2, 13], GEAR [27], LRTA\* [29, 31]) protocols belong to this category.

Constrained flooding has been proposed as one of the meta-routing strategies [30, 31] for constraint-based routing [28]. Constrained flooding consists of a real-time reinforcement kernel, shared by other meta-routing strategies [31] in constraint-based routing, and a constrained retransmission policy with differential delays. In this paper, we will present an in-depth study of the constrained flooding protocols, performance analysis and comparisons with other peer routing protocols for some real-world applications. We will show that constrained flooding protocols are not only robust with high success rates but also, unlike other flooding protocols, energy efficient.

There is not much work on flooding-based routing strategies; most flooding strategies are associated with route discovery, e.g., [18]. However, flooding was suggested as a robust multicast approach for highly mobile fast-moving ad-hoc networks since traditional approaches fail to work in such scenarios [8]. Efficient flooding techniques for multicast were also proposed via gossip [14, 15] or via multipoint relaying [20]. A couple of flooding-based routing protocols have been proposed recently, such as GRAD [9], GRAB [26] and DFRF [16]. GRAD is a basic gradient flooding protocol where all nodes with smaller cost to the sink retransmit the data. GRAB adds credit control to the basic gradient routing which effectively reduces the degree of redundancy and energy consumption. DFRF incorporates priorities and multiple redundant transmissions to further increase success rates. DFRF has been implemented on Berkeley motes and applied to Shooter Localization [23], one of the most successful applications in sensor networks.

The major elements of the constrained flooding (CF) framework are the following:

**Real-time learning of potential/cost field:** other than most flooding protocols where the potential field has to be established and maintained, CF incorporates real-time reinforcement learning techniques to build and update the potential field. No extra control packets are necessary.

**Temperature control for robustness:** unlike other gradient routing, where only nodes with smaller cost retransmit the packet, CF adds an extra variable, temperature,

with the intent that only nodes with cost difference smaller than the temperature participate in the relay. The temperature will be cooled down over time to reduce the number of transmissions and increase energy efficiency.

**Delayed transmission according to the gradient descent:** instead of adding random delays to retransmission packets to reduce broadcast collision, CF adds delays according to the cost difference. The smaller the cost, the sooner is the retransmission.

**Probabilistic retransmission based on data frequency:** the node will retransmit the delayed packet based on the number of times the same packet has been heard. The more times the packets have been heard, the less likely that it will be retransmitted by this node. This way, packets at nodes with larger cost (with larger delays) are likely to be suppressed, significantly reducing the number of transmissions.

Note that not all elements are novel and some of them (e.g. delayed, probabilistic and counter-based retransmission) have been proposed before [24]. CF integrates constraint-based message specification with reinforcement learning strategies and applies them effectively to control the broadcast storm. CF can also incorporate credit control from GRAB and priority management from DFRF in a layered routing architecture. Furthermore, CF is a meta-routing strategy that can be used with general routing specifications, of which geographical or energy-aware routings are canonical examples.

The remainder of the paper is organized as follows. Section 2 presents the constrained flooding framework and its main components. Section 3 analyzes the protocols in this framework via simulation and compares routing performances with other protocols for some real-world applications. Section 4 concludes the paper.

## 2. Constrained Flooding

In this section, we will first present the basic elements of constrained flooding protocols and then introduce a layered routing architecture with the components for implementing constrained flooding protocols.

### 2.1. Constrained Flooding Protocols

The basic elements of constrained flooding include (1) a real-time reinforcement learning kernel for establishing and maintaining the potential field, (2) a constrained propagation policy, (3) differential delays based on cost difference, and (4) probabilistic retransmission based on data frequency.

#### 2.1.1. Potential field generation

The potential field of a network is the mapping from nodes to cost-to-go values. The definition of the cost-to-go depends on routing objectives. For example, if the shortest path is the routing objective, cost-to-go is the minimum number of hops from the current node to the sink node. In general, if the local cost function is  $o$ , the cost-to-go at node  $v^0$  is  $\min_p \sum_i o(v^i)$  where  $p: v^0, v^1, \dots, v^{n-1}$  is a path from  $v^0$  to the sink node.

Most flooding protocols generate the potential field by flooding from the sink node. Each node maintains the estimated cost-to-go  $c$ . The sink has zero cost-to-go and all the other nodes have infinite initial values. Starting from the sink node, a packet is sent out advertising its cost-to-go. Whenever a node  $v$  receives a packet from a neighbor with cost-to-go as  $c'$ , it calculates its new cost-to-go as  $\min(c, o(v)+c')$  and sends a packet with its cost-to-go. After a complete flooding from the sink node, every node will have a cost-to-go value and the potential field is established. There are two problems with this method. First of all, if the network changes, one has to regenerate the potential field by flooding from the sink. For the sink to recognize the changes in the network, network performance metrics, e.g., packet loss rate, have to be maintained. If the network is highly dynamic, this method would have both high energy cost and high loss rates. Periodic flooding can be performed as well, again, with the extra energy cost. Secondly, if the sink is not fixed, for example, in the case of geographical routing, one has to locate the sink first through an initial flooding.

For real-time reinforcement learning, in particular Q-learning, each node maintains not only its estimated cost-to-go, but also cost-to-go values of its neighbors. Initially all the cost-to-go values are set to zero, if there is no knowledge about the sink. Every packet (not just the control packets) has a piggybacked cost-to-go field and all the nodes (excluding the ones in sleep state) are set to be in promiscuous listening mode. Whenever a non-sink node  $v$  overhears a packet, whether it is the designated receiver or not, it updates the corresponding cost-to-go value of the neighbor where the packet was sent from, and re-estimates its own cost-to-go using the formula

$$c \leftarrow (1 - \alpha)c + \alpha(o(v) + \min_w c'(w)) \quad (1)$$

where  $0 < \alpha \leq 1$  is the learning rate,  $w$  is a neighbor of this node, and  $c'$  is the neighbor's cost-to-go. There are two advantages with this method. First of all, initial flooding from the sink is useful but unnecessary, since packets from any direction help establish the potential field. Similarly, when the network changes, no flooding from the sink is needed, ongoing packets in the network will update the potential field. Secondly, if one has knowledge about the sink, cost-to-go can be estimated initially rather than set to zero uniformly. For example, in the case of geographical routing with the shortest path objective, the cost-to-go at any node can be estimated by  $d/R$ , where  $d$  is the Euclidean distance between the node and the sink, and  $R$  is the maximum radio range. When there are no network holes, this estimation of the potential field can be accurate enough for efficient geographical flooding. When there are network holes, the potential field would be automatically updated during routing.

### 2.1.2. *Constrained propagation*

All flooding protocols piggyback the cost-to-go value in each data packet, and most of them use a simple rule to forward packets: if a node receives a packet with cost-to-go not lower than its own, it retransmit the packet. This simple strategy works if all the cost-to-go values are accurate. If the cost-to-go values are not fully learned yet (e.g., there is a

network hole in geographical routing) or if the network changes, this simple rule causes the loss of packets.

In constrained flooding, this simple rule is generalized using an extra variable, temperature  $T$ : if a node with cost-to-go  $c$  receives a packet with cost-to-go  $c'$ , and if  $c - c' \leq T$ , it retransmits the packet. This rule is identical to the simple rule when  $T = 0$ . However, in this generalized form,  $T$  can be changing during routing. For example,  $T$  can be set to large values initially and reduced gradually (cooling down) with the learning of the potential field.

The temperature cooling-down process may vary in this general framework. For example, one can reduce  $T$  after every packet received:

$$T \leftarrow kT / (k + 1) \quad (2)$$

where  $k = T_0$ , the initial temperature.

### 2.1.3. Differential delays

Most flooding protocols add random delays to forwarding packets to avoid collision. In constrained flooding, delays are added to forwarding packets according to the cost difference  $\Delta = c' - c$ : the larger the difference, the smaller the delay. The delay function  $\delta$  has the property that if  $\Delta_1 \geq \Delta_2$ ,  $\delta(\Delta_1) \leq \delta(\Delta_2)$ . For example, the delay function

$$\delta(\Delta) = D / e^\Delta \quad (3)$$

where  $D$  is a constant, works quite well.

### 2.1.4. Probabilistic retransmissions

For most flooding protocols, all the forwarding packets will be transmitted after random delays. For constrained flooding, after the designated delay time, a probabilistic policy will be used to decide whether or not to retransmit the packet: the more times the same packet has been heard, the less possible the packet will be retransmitted. Let  $p$  be a probability function for retransmission and  $C$  denote the number of times the packet has been heard in the past, then  $p$  should satisfy (1)  $p(1) = 1$ , i.e., if only one such a packet is heard, it should be transmitted, and (2)  $p(C_1) \leq p(C_2)$  if  $C_1 \geq C_2$ . An example of such a function is

$$p(C) = 1 / C^\gamma \quad (4)$$

where  $\gamma \geq 0$ . Here  $\gamma$  can be tuned to trade off between robustness and energy cost. The smaller the  $\gamma$ , the more packets are to be transmitted and the higher the energy cost.

### 2.1.5. Constrained flooding protocols

Like most flooding protocols, in addition to a data field, constrained flooding protocols have the cost-to-go field and a unique identifier, which can be a sequence ID or a time stamp at the source. The cost-to-go is used to generate and update the potential field as well as to control the direction of flooding, and the unique identifier is used to check duplicates and to count the number of times such a packet has been heard. Furthermore,

constrained flooding protocols have each sender's (including relay nodes) address in any packet, so that neighborhood structure can be maintained.

A flowchart for constrained flooding protocols is shown in Fig. 1. Module 1 updates the temperature (e.g., using Eq. (2)) and its cost-to-go as per Eq. (1), module 2 determines if the packet shall be forwarded based on the cost difference and the current temperature at this node, module 3 adds delay to the forward packet (e.g., according to delay function Eq. (3)), and module 4 makes the final decision (e.g., according to probabilistic function Eq. (4)) of sending the packet after the designated delay time.

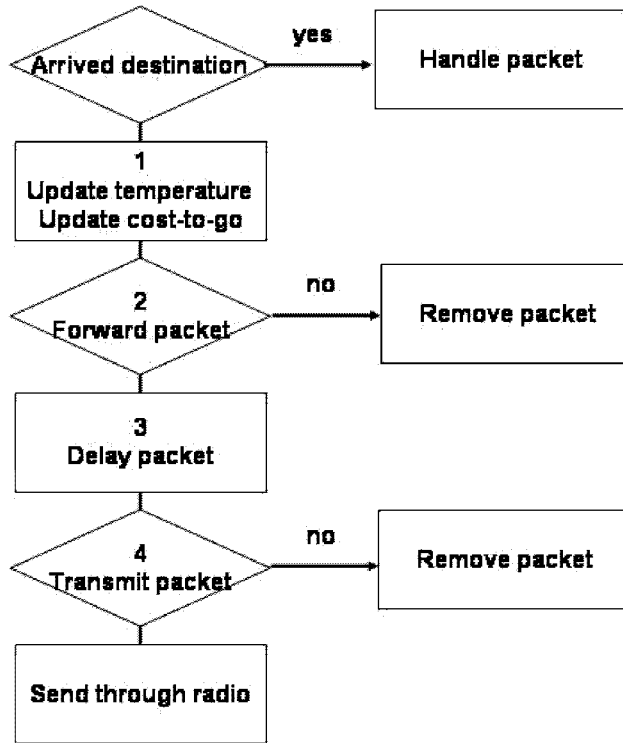


Fig. 1. Flowchart for constrained flooding protocols.

## 2.2. Constrained Flooding Components

The constrained flooding framework can also incorporate mechanisms proposed by other flooding protocols, such as credit control [26], priority management for duplicated transmissions [16], and packet aggregations [16]. On the other hand, components that are used in constrained flooding, such as delayed and probabilistic retransmission, can be applied to any flooding-based routing. In this section, we advocate a layered routing architecture, then describe the essential components for constrained flooding as well as other components that can be used within the framework.

### 2.2.1. Layered routing architecture

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*. Fig. 2 shows the general architecture. 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 (i.e., *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. Furthermore, one can have a set of *control layers* between the routing layer and the application layer, and a set of *support layers* between the MAC layer and the routing layer. The control layers are for network initialization and maintenance, e.g., building and maintaining a routing table or a spanning tree. The support layers are for adding features such as transmitting and receiving queues, data aggregation/fragmentation, neighborhood management, confirming/delaying transmissions, and checking duplications. It is the algorithm designer's choice to put individual functions into different layers so that common functions can be shared by different algorithms.

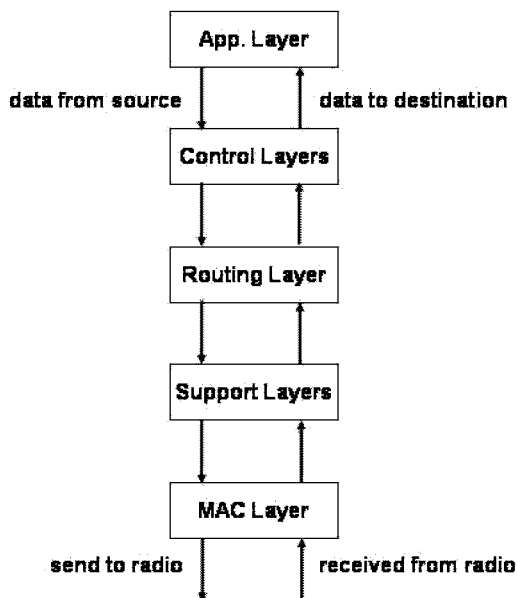


Fig. 2. A general layered routing architecture.

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 control and support routing components.

### 2.2.2. Constrained flooding layers

The basic constrained flooding protocols have the following layers: *application* → *constrained\_flood* → *delay\_transmit* → *neighborhood* → MAC, where *constrained\_flood* is the routing layer, *delay\_transmit* and *neighborhood* are support layers.

- *constrained\_flood*: This layer implements the basic functions of modules 1, 2 and 3 in Fig. 1. It maintains the cost-to-go of this node as well as the cost-to-go values of its neighbors. The value of cost-to-go at this node will be added to each packet passing this layer, which will be transmitted with the packet. Delays are calculated for forwarding packets.
- *delay\_transmit*: This layer implements the function of module 4 in Fig. 1. It maintains a timeout queue and the counts for all the packets it has heard. After the timeout for each packet, the probabilistic transmission rule is applied.
- *neighborhood*: Each packet sent from this layer will piggyback the node's address, which will be retrieved from the packet received at this layer. A list of neighbors is maintained.

Using the layered architecture, one can incorporate other routing components into the constrained flooding framework. These components include *initialization*, *credit\_control*, *duplicate\_transmission*, and *aggregate\_queue*.

- *initialization*: Initial flooding from the sink node generates an initial potential field effectively.
- *credit\_control*: Credit control for flooding has been used in GRAB [26] to control the width of flooding, trading off energy usage and success rates.
- *duplicate\_transmission*: Duplicated transmission using priority management has been implemented in DFRF [16] to increase packet success rates.
- *aggregate\_queue*: Packet aggregation assembles a number of packets into one packet before sending and disassembles a packet into individual packets after receiving. In the shooter localization application [23], maximum four packets are aggregated into one packet.

## 3. Performance Evaluations

In this section, we evaluate the routing performance of the constrained flooding framework. We first describe our simulation environment and routing performance

metrics, and then study the properties of the algorithm by varying parameters and finally compare its performance with peer routing protocols using two real-world scenarios.

### 3.1. Network Model

Our protocol study uses the radio propagation model and the MAC layer communication model provided by Prowler [22, 36], a probabilistic sensor network simulator.

The simple radio model in Prowler attempts to simulate the probabilistic nature in wireless sensor communication observed by many [25, 37]. 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:

$$P_{rec,ideal}(d) \leftarrow P_{transmit} \frac{1}{1+d^\gamma} \tag{5}$$

where  $2 \leq \gamma \leq 4$  and

$$P_{rec}(i, j) \leftarrow P_{rec,ideal}(d_{i,j})(1 + \alpha(i, j))(1 + \beta(t)) \tag{6}$$

where  $P_{transmit}$  is the signal strength at the transmitter and  $P_{rec,ideal}(d)$  is the *ideal* received signal strength at distance  $d$ ,  $\alpha$  and  $\beta$  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 (6),  $\alpha$  is static depending on locations  $i$  and  $j$  only, and  $\beta$  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 for any other reason. The default radio model in Prowler has  $\gamma = 2$ ,  $\sigma_\alpha = 0.45$ ,  $\sigma_\beta = 0.02$ ,  $\Delta = 0.1$  and  $p_{error} = 0.05$ . Fig. 3 shows a snapshot of the radio reception curves in this model.

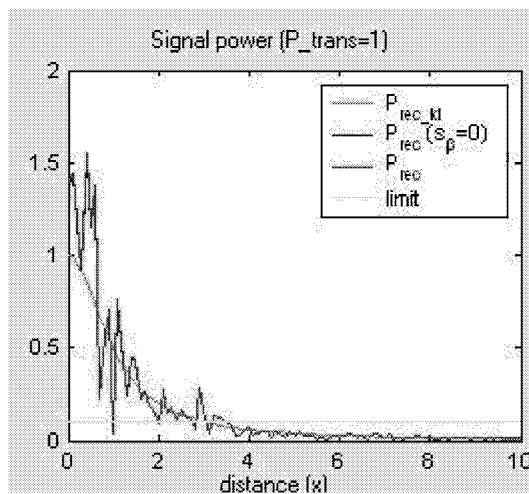


Fig. 3 Snapshot of radio reception curves for the default model.

The MAC layer communication is modeled by a simplified event channel that simulates the Berkeley notes' [1] CSMA MAC protocol. When the application emits the *Send\_Packet* command, after a random *Waiting\_Time* interval the MAC layer checks if the channel is idle. If not, it continues the idle checking until the channel is found idle. The time between idle checks is a random interval characterized by *Backoff\_Time*. When the channel is idle the transmission begins, and after *Transmission\_Time* the application receives *Packet\_Sent* event. After the reception of a packet on the receiver's side, the application receives a *Packet\_Received* or *Collided\_Packet\_Received* event, depending on the success of the transmission. The data rate is 40Kbits per second and each packet has 960 bits.

### 3.2. Performance Metrics

Some routing performance metrics are used in this paper for comparing different routing protocols, including: latency, success rates, and energy efficiency.

- *Latency*: The latency of a packet measures the time delay of the packet from the source to the sink. For any sink, if  $n$  packets have arrived, latency for that sink is given by  $\sum_i d_i/n$ , where  $d_i$  is the latency of the  $i$ th packet. The latency of the network is then averaged by the number of sinks. Note that we use latency rather than the number of hops as the metric since latency is caused by 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.
- *Success rates*: The success rate of a network measures the total number of packets received at all the sinks vs. the total number of packets sent from all the sources.
- *Energy efficiency*: Energy efficiency is defined to be the ratio between the total number of packets received at the sink(s) vs. the total number of packets sent in the network.

### 3.3. Properties of Constrained Flooding

We study some properties of constrained flooding using a network of 100 nodes, placed in a 10x10 grid with small random offsets. The average number of neighbors is around 6. The number of hops from the source to the destination is around 10. The default radio model in Prowler is used. The source sends 50 packets, one per second, to the destination. The basic constrained flooding layers are used. In all of our tests, learning rate  $\alpha$  is set to 1. Following is a set of results, averaged over 10 random runs.

#### 3.3.1. Efficiency vs. temperature over time

One property of this algorithm is that with the cooling of temperatures, the overall efficiency increases. Fig. 4 shows a scenario with initial temperature  $T_0 = 20$ , probability function parameter  $\gamma = 1$  and delay constant  $D = 0.025$  second.

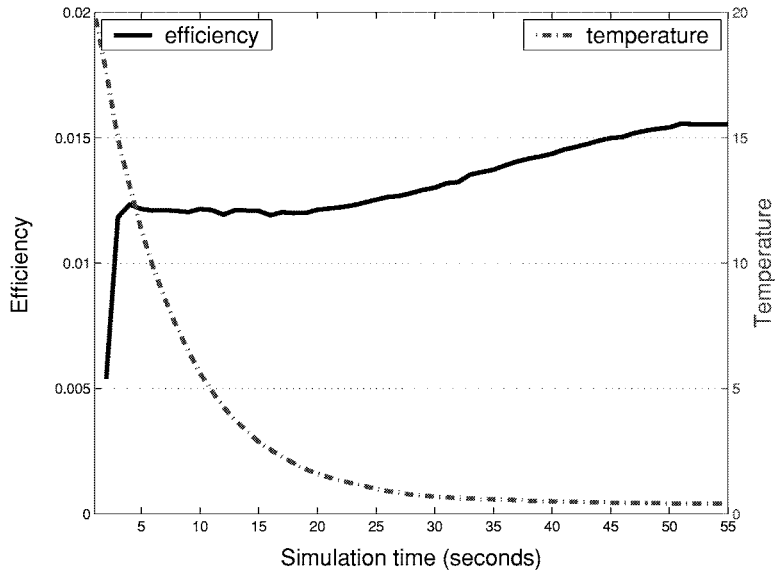


Fig. 4. Efficiency vs. temperature over time.

3.3.2. Efficiency vs. latency with different delay constant

By reducing delay constant  $D$ , one can decrease latency, but at the same time reduce efficiency, since less packets would be filtered out. Fig. 5 shows a scenario with  $T_0 = 15$  and  $\gamma = 1$ .

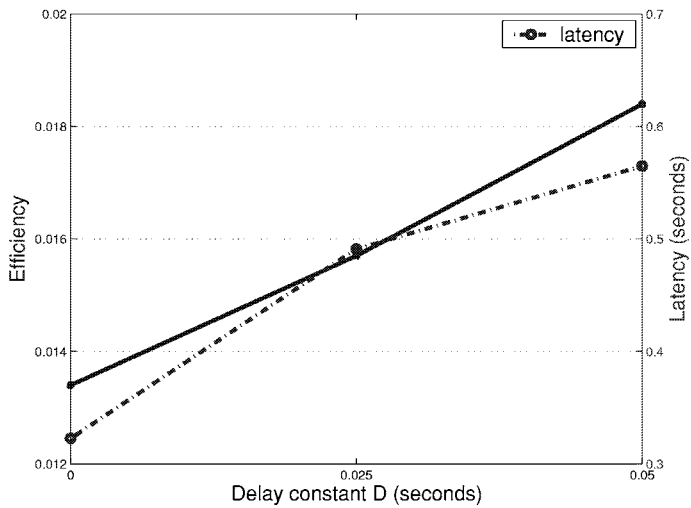


Fig. 5. Efficiency vs. latency with different delay constant.

### 3.3.3. Performance with varying $T_0$ and $\gamma$

Fixing  $D=0.025$ , we study the effects of  $T_0$  and  $\gamma$  on efficiency and success rates. Fig. 6 shows that by increasing initial temperature, efficiency decreases. However,  $\gamma$  affects efficiency in a different way for different  $T_0$ . When  $T_0 = 20$ , the increase of  $\gamma$  increases efficiency. Fig. 7 shows that by increasing initial temperature, success rates increase. Again,  $\gamma$  affects success rates in a different way for different  $T_0$ . When  $T_0 = 20$ , the increase of  $\gamma$  increases success rates.

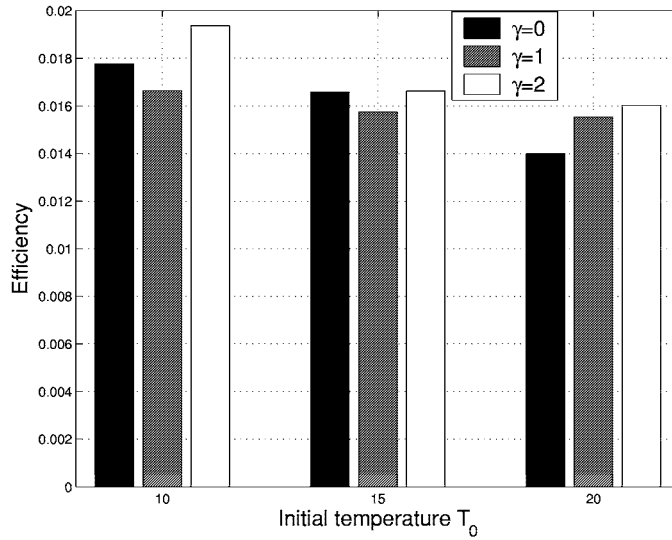


Fig. 6. Efficiency with varying  $T_0$  and  $\gamma$

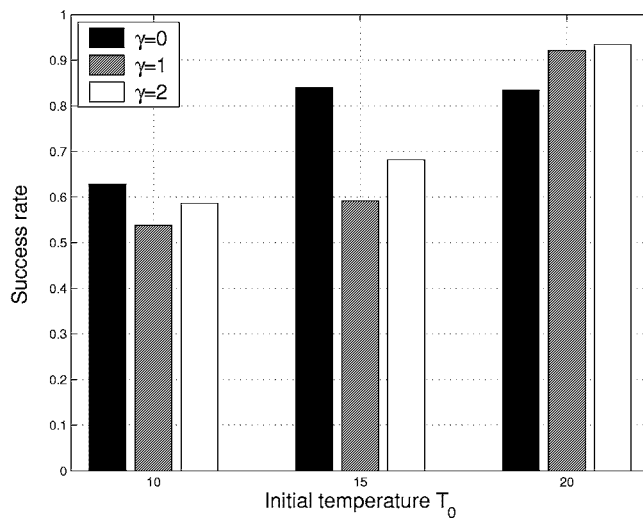


Fig. 7. Success rate with varying  $T_0$  and  $\gamma$

### 3.4. Pursuer Evader Games

The Pursuer Evader Game (PEG) has been a standard testbed for sensor networks. In this testbed, sensors are deployed in a regular grid with random offsets. An evader is a tank or a car whose movement can be detected by the sensor nodes. A pursuer is another vehicle that is going to track the evader down. The communication problem in this task is to route packets sent out by sensor nodes who detected the evader to the mobile pursuer.

In our test, the network is a  $7 \times 7$  sensor grid with small random offsets. The transmit signal strength is set to 1, and the maximum radio range is about  $3d$  (a dense network), where  $d$  is the standard distance between two neighbor nodes in the grid.

Fig. 8 shows an instance of the connectivity of such a network.

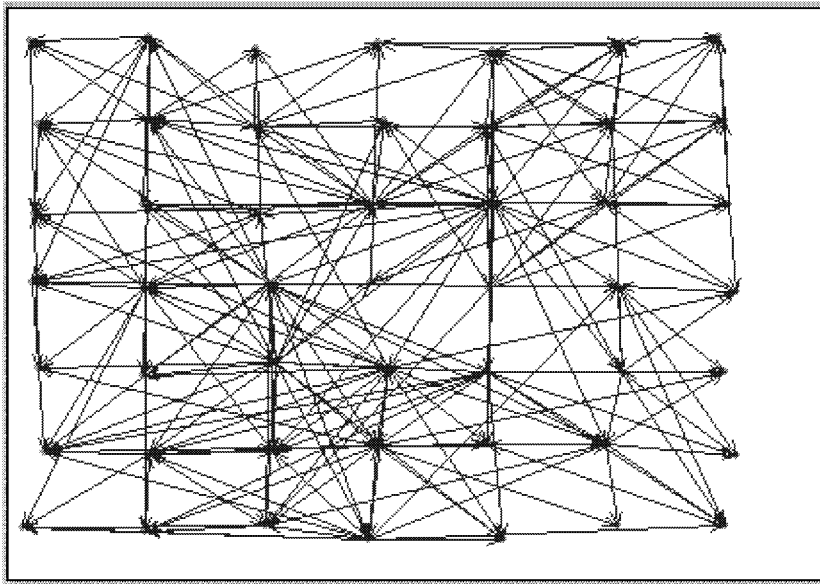


Fig. 8. Instance of radio connectivity in PEG.

In this simulation, the source is dynamic, changing from node to node, following the movement of the evader, and the sink is mobile. The average speed of both the evader and the pursuer is set to  $0.2d/s$ . The source starts at the center and the sink starts at a corner.

Two peer routing algorithms are tested in this application to compare the routing performance with CF:

- *Directed Routing (DFRF)* [16] -- directional flooding using a "potential field" established in initialization, and sending a packet possibly a couple of times for extra high success rates;
- *Grid Routing (Grid)* [4] -- a spanning-tree based routing with "grid" information;

- *Grid Routing with Duplication (Grid+Dup)* -- grid routing used with the duplication module to increase success rates;
- *Constrained Flooding (CF)* -- basic CF with initialization, where, just like DFRF, the potential field is established in initialization but are continuously improving by learning.

Figs. 9, 10, and 11 show the latency, success rates, and energy efficiency, respectively.

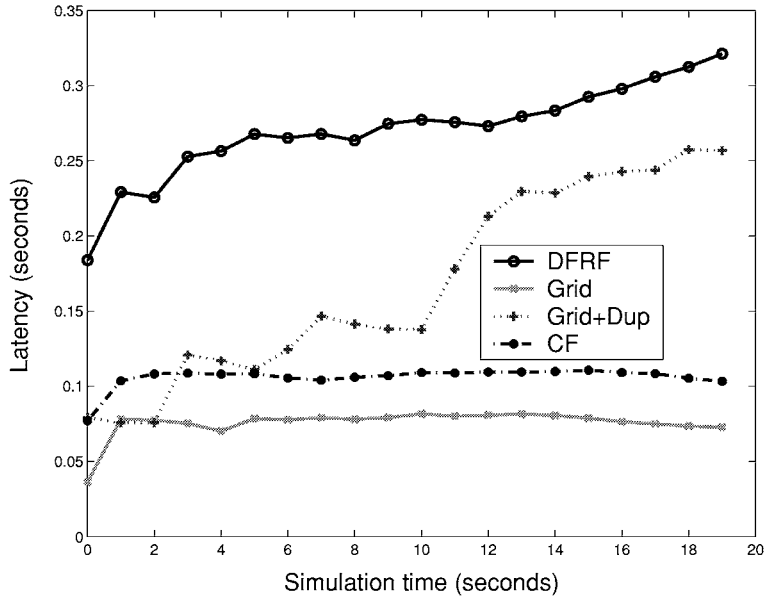


Fig. 9. PEG: Latency

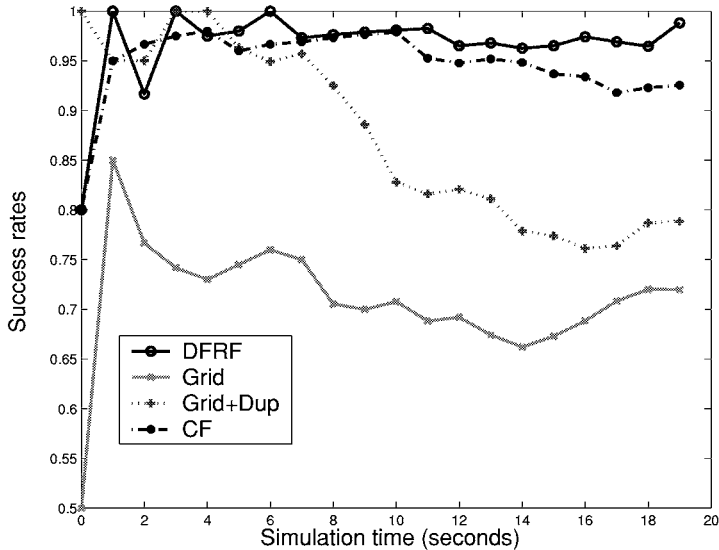


Fig. 10. PEG: Success rates

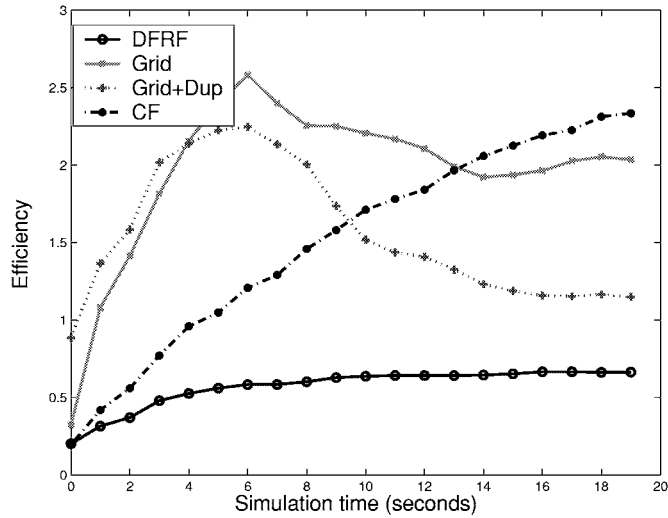


Fig. 11. PEG: Energy efficiency

From the performance results we see that both DFRF and CF have high success rates. However DFRF has lower energy efficiency than CF, grid routing improves success rates with the duplication component but also decreases the energy efficiency significantly. Both CF and grid routing without duplication have low latency. Note that we have also tested AODV [17] for this application scenario, which has near-zero success rates, due the fast moving target.

### 3.5. Shooter Localization

Shooter localization [23] is a successful sensor network application. For this application, all the acoustic data that are above certain threshold have to be sent to the base station, so that the source of the sound can be located. The communication problem in shooter localization is a typical convergecast [9, 12]. In this experiment, the sensor distribution, the sink location and the event trace data are obtained from the actual experiments on the hardware platform. The network is about 10 hops (Fig. 12).

Three algorithms are compared for this scenario:

- *Directed Routing (DFRF)* [16] -- this algorithm is actually used in the shooter localization application, implemented in TinyOS/NesC; DFRF integrates both duplicate transmissions (up to 3 times) and packet aggregations (up to 4 packets).
- *Backbone tree (Backbone)* [7] -- this algorithm uses directed diffusion [10] to create a backbone tree in the initialization phase and passes packets to parents during routing. To handle the problem of asymmetric links, it establishes a symmetric link neighborhood in the initialization;
- *Constrained Flooding (CF)* -- basic CF with initialization;

- *Constrained Flooding with Aggregation and Duplication (CF+Agg+Dup)* -- basic CF with initialization, aggregation and duplication components.

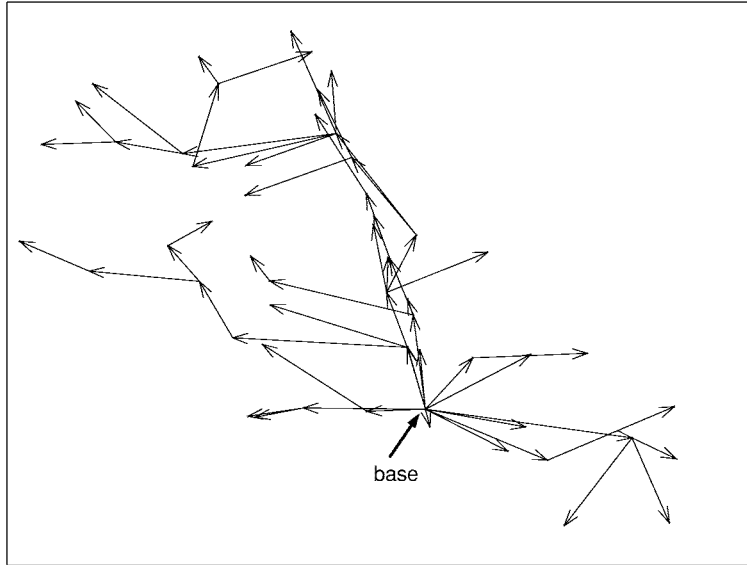


Fig. 12. Node distribution in the shooter localization experiment

Figs. 13, 14 and 15 show latency, success rates and energy efficiency, respectively.

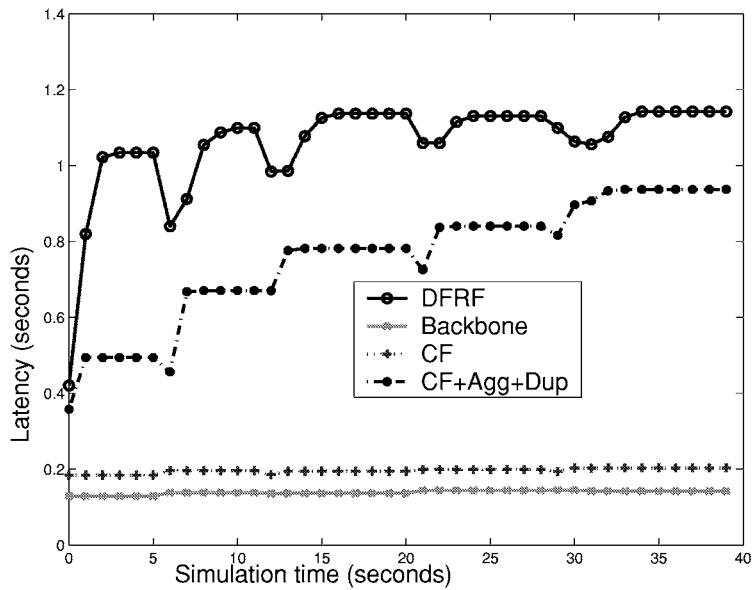


Fig. 13. Shooter Localization: Latency

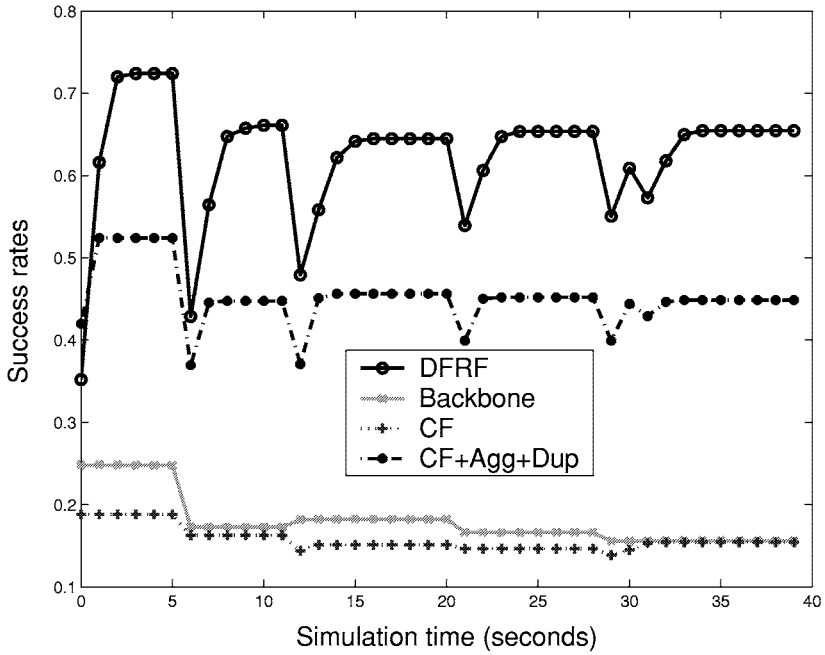


Fig. 14. Shooter Localization: Success rates

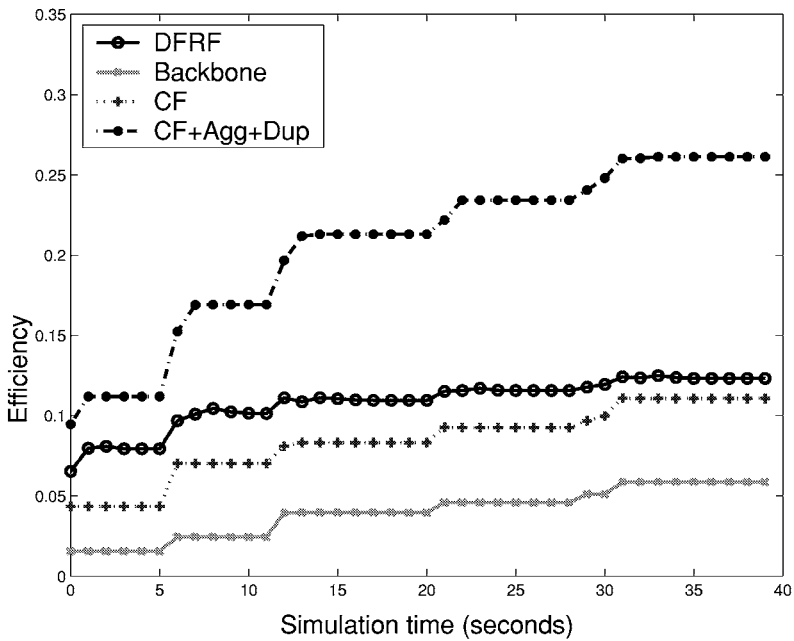


Fig. 15. Shooter Localization: Energy efficiency

We see that DFRF has the highest success rates. CF+Agg+Dup increases the success rates and has the highest energy efficiency. We have also tried the augmentation of

aggregation and duplication to the Backbone layers but it does not seem to increase success rates either.

#### 4. Conclusions

We have presented in this paper the framework of constrained flooding (CF), its basic elements and its augmented utilities in a layered routing architecture. We have shown in our experiments that CF can be both robust and energy efficient, and our framework is flexible to incorporate additional components, such as initialization, credit control, aggregation and duplication.

CF is a type of meta-routing strategy for the constraint-based routing, that can be used with different routing specifications, e.g., geographical or energy-aware routing. Experiments with routing objectives other than shortest paths are considered as future work.

#### References

1. U.C. Berkeley, "*Berkeley Wireless Embedded Systems*," <http://webs.cs.berkeley.edu/>.
2. J.A. Boyan and M.L. Littman, "Packet Routing in Dynamically Changing Networks: A Reinforcement learning Approach," In *Advances in Neural Information Processing Systems*, volume 6, pages 671-678. Morgan Kaufmann Publishers, Inc., 1994.
3. G. D. Caro and M. Dorigo, "AntNet: Distributed Stigmergetic Control for Communication Networks," *Journal of Artificial Research*, **9**, pages 317-365, 1998.
4. Y. Choi, M.G. Gouda, H. Zhang, and A. Arora, "*Routing on a Logical Grid in Sensor Networks*," Technical Report TR04-49, Department of Computer Science, The University of Texas at Austin, 2004.
5. M. Chu, H. Haussecker, and F. Zhao, "Scalable Information-Driven Sensor Querying and Routing for Ad-hoc Heterogeneous Sensor Networks," *Int. Journal on High Performance Computing Applications*, June, 2002.
6. D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-Resilient Energy-Efficient Multipath Routing in Wireless Sensor Networks," *Mobile Computing and Communications Review* 1(2), 2002.
7. T. He, S. Krishnamurthy, J. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh, "*An Energy-Efficient Surveillance System Using Wireless Sensor Networks*," In Proc. 2nd International Conference on Mobile Systems, Applications and Services (MobiSys04), 2004.
8. C. Ho, K. Obraczka, G. Tsudik, and K. Viswanath, "*Flooding for Reliable Multicast in Multi-hop Ad-hoc Networks*," In Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M), 1999.
9. 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.
10. C. Intanagonwiwat, R. Govindan, and D. Estrin, "*Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks*," In Proc. 6th Int'l Conference on Mobile Computing and Networks (ACM Mobicom), Boston, MA, 2000.

11. D.B. Johnson and D.B. Maltz, "Dynamic Source Routing in Ad-hoc Wireless Networks," In T. Imielinski and H. Korth, editors, *Mobile Computing*, pages 153-181. Kluwer Academic, 1996.
12. B. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," In Proc. 6th Int'l Conf. on Mobile Computing and Networks (ACM MobiCom), Boston, MA, 2000.
13. S. Kumar and R. Miikkulainen, "Confidence-based Q-Routing: An On-line Adaptive Network Routing Algorithm," In Proceedings of Artificial Neural Networks in Engineering. 1998.
14. L.Li, J. Halpern, and Z. Haas, "Gossip-based Ad-Hoc Routing," In IEEE InfoCom, 2002.
15. J. Luo, P. Eugster, and J. Hubaux, "Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks," In IEEE InfoCom, 2003.
16. M. Maroti, "Directed Flood-Routing Framework. Technical Report," ISIS-04-502, Institute for Software Integrated Systems, Vanderbilt University, 2004.
17. C.E. Perkins and E.M. Royer, "Ad hoc On-demand Distance Vector Routing," In Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications, pages 90--100, February 1999.
18. S. Pleisch, M. Balakrishnan, K. Birman, and R. van Renesse, "MISTRAL: Efficient Flooding in Mobile Ad-hoc Networks," In 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing, 2006.
19. R. Poor, "Gradient Routing in Ad hoc Networks," <http://www.media.mit.edu/pia/Research/ESP/texts/poorieecpaper.pdf>.
20. A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying: An Efficient Technique for Flooding in Mobile Wireless Networks," Technical Report 3898, March 2000. <ftp://ftp.inria.fr/INRIA/publication/publi-ps-gz/RR/RR-3898.ps.gz>.
21. E. Royer and C. Toh, "A Review of Current routing Protocols for Ad hoc Mobile Wireless Networks," *IEEE Personal Communications*, April 1999.
22. G. Simon, "Probabilistic Wireless Network Simulator," <http://www.isis.vanderbilt.edu/projects/nest/prowler/>.
23. 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.
24. Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad hoc Network," *ACM Wireless Networks*, 8(2), March 2002.
25. A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," In Proc. 1st ACM International Conference on Embedded Networked Sensor Systems (SenSys03), 2003.
26. F. Ye, G. Zhong, S. Lu, and L. Zhang, "A Robust Data Delivery Protocol for Large Scale Sensor Networks," In *Information Processing in Sensor Networks*, Lecture Notes in Computer Science, number 2634, page 658, 2003.
27. Y. Yu, R. Govindan, and D. Estrin, "Geographical and Energy Aware Routing: a Recursive Data Dissemination Protocol for Wireless Sensor Networks," Technical Report UCLA/CSD-TR-01-0023, UCLA Computer Science Department, May 2001.
28. 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.

29. Y. Zhang and M. Fromherz, "Search-based Adaptive Routing Strategies for Sensor Networks," In AAAI04 workshop on Sensor Networks, 2004.
30. Y. Zhang and M. Fromherz, "Constrained Flooding: A Robust and Efficient Routing Framework for Wireless Sensor Networks," In 20th International Conference on Advanced Information Networking and Applications, 2006.
31. 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.
32. Y. Zhang and Q. Huang, "Coordinated Convergecast in Wireless Sensor Networks," In MilCom05, 2005.
33. Y. Zhang and Q. Huang, "Adaptive Tree: A Learning-based Meta-routing Strategy for Sensor networks," In 3rd IEEE Consumer Communications and Networking Conference, 2006.
34. Y. Zhang, L. Kuhn, and M. Fromherz, "Improvements on Ant-routing for Sensor Networks," In Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence, Lecture Notes in Computer Science 3172, 2004.
35. Y. Zhang, J. Liu, and F. Zhao, "Information-Directed Routing in Sensor Networks using Real-time Reinforcement Learning," In Combinatorial Optimization in Communication Networks. Kluwer Academic Publishers, 2005.
36. Y. Zhang, G. Simon, and G. Balogh, "High-level Sensor Network Simulations for Routing Performance Evaluations," In Third International Conference on Networked Sensing Systems, 2006.
37. J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," In Proc. of 1st International Conference on Embedded Networked Sensor Systems (SenSys03), 2003.