A Spatiotemporal Communication Protocol for Wireless Sensor Networks

Tian He, *Member*, *IEEE*, John A. Stankovic, *Fellow*, *IEEE*,
Chenyang Lu, *Member*, *IEEE*, and Tarek F. Abdelzaher, *Member*, *IEEE*

Abstract—In this paper, we present a spatiotemporal communication protocol for sensor networks, called SPEED. SPEED is specifically tailored to be a localized algorithm with minimal control overhead. End-to-end soft real-time communication is achieved by maintaining a desired delivery speed across the sensor network through a novel combination of feedback control and nondeterministic geographic forwarding. SPEED is a highly efficient and scalable protocol for sensor networks where the resources of each node are scarce. Theoretical analysis, simulation experiments, and a real implementation on Berkeley motes are provided to validate the claims.

Index Terms—Wireless sensor networks, routing, real-time, spatiotemporal, geographic forwarding.

1 Introduction

Many sensor network applications, such as battlefield surveillance and earthquake response systems, are designed to interact with fast changing events in the real world. It is often necessary for the underlying communication infrastructure to meet real-time constraints [9], [16], [4]. In surveillance systems [8], for example, communication delays within sensing and actuating loops directly affect the quality of tracking. Our spatiotemporal communication protocol, called SPEED, is inspired by following observation: In wired networks, the end-to-end delay is independent of the physical distance between the source and destination. While in multihop wireless sensor networks, since communication is physically bounded, the end-to-end delay depends not only on single hop delay (time-constraints), but also on the distance a packet travels (spatial-constraints). In view of this, the key design goal of this work is to support a spatiotemporal communication service with a desired delivery *speed* across the sensor network, so that end-to-end delay is proportional to the distance between the source and destination. We deem this service as one type of soft real-time communication, because it achieves a predicable end-to-end communication delay under given distance (spatial) constraints.

The key contribution of SPEED is achieving the spatiotemporal requirements through a novel combination of a time-aware feedback control mechanism and a spatial-aware nondeterministic geographic forwarding scheme. We evaluate SPEED using GloMoSim [22]. The performance results show that SPEED 1) reduces the number of packets that miss their end-to-end deadlines, 2) reacts to transient congestion in the most stable manner, and 3) efficiently handles voids [12] with minimal control overhead. To demonstrate its applicability, we also implement SPEED on the Berkeley motes [3]. The results show that SPEED helps balance the traffic load to increase the system lifetime.

• *T. He, J.A. Stankovic, and T.F. Abdelzaher are with the Department of Computer Science, University of Virginia, Charlottesville, VA 22903-2442. E-mail: {tianhe, stankovic, zaher}@cs.virginia.edu.*
• *C. Lu is with the Department of Computer Science and Engineering, Washington University, St. Louis, MO 63130. E-mail: lu@cs.wustl.edu.*

Manuscript received 24 Oct. 2003; revised 2 Sept. 2004; accepted 6 Jan. 2005; published online 22 Aug. 2005.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-0195-1003.

2 State-of-the-Art

Several routing protocols have been developed for ad hoc wireless networks. Sensor networks can be regarded as a subcategory of such networks, but with a number of different requirements.

In sensor networks, location is more important than a specific node's ID. For example, tracking applications only care where a target is located, not the ID of the reporting node. In sensor networks, such spatial-awareness [7] is necessary to make the sensor data meaningful. Therefore, it is natural to utilize spatial-aware routing. A set of location based routing algorithms have been proposed. Finn [5] proposed a greedy geographic forwarding protocol with limited flooding to circumvent the voids inside the network. GPSR [12] by Karp and Kung use perimeter forwarding to get around voids. Similarly, GOAFR [14] combines the greedy routing with the adaptive face routing to provide an asymptotically optimal path to the destination. Geographic distance routing (GEDIR) [20] guarantees loop-free delivery in a collision-free network. LAR [13] by Ko and Vaidya improves the efficiency of the on-demand routing algorithms by restricting packet flooding in a specified "request zone." Basagni et. al. propose a distance routing algorithm [2] for mobility (DREAM), in which each node periodically updates its location information to other nodes. An updating rate is set according to a distance effect in order to reduce the number of control packets. Recently, Huang et al. [4] proposed Mobicast protocol extended geocast by providing just-in-time information dissemination to nodes in a mobile delivery zone.

SPEED also utilizes geographic location to make localized routing decisions. The key difference is that SPEED takes timely delivery into account and is designed to be the first spatiotemporal-aware communication protocol for sensor networks. Moreover, SPEED provides an alternative solution to handle voids other than approaches based on planar graph traversal [12], [14] and limited flooding [5].

Reactive routing algorithms such as AODV [17] and DSR [10] maintain routing information for a small subset of possible destinations, namely, those currently in use. If no route is available for a new destination, a route discovery process is invoked. Route discovery broadcasts can lead to significant delays in a sensor network with a large network diameter. This limitation makes on-demand algorithms less suitable for real-time applications.

Several real-time protocols have been proposed for ad hoc and sensor networks. SWAN [1] uses feedback information from the MAC layer to regulate the transmission rate of nonreal-time TCP traffic in order to sustain real-time UDP traffic. RAP [16] uses velocity monotonic scheduling to prioritize real-time traffic and enforces such prioritization through a differentiated MAC Layer. Kanodia et al. [11] proposed a service differentiation for delay-sensitive traffic by prioritizing 802.11. Woo and Culler [21] proposed an adaptive MAC layer rate control to achieve fairness among nodes with different distances to the base station. All of these algorithms work well by locally degrading a certain portion of the traffic. However, this kind of local MAC layer adaptation cannot handle long-term congestion where routing assistance is necessary to divert traffic away from any hotspot. SPEED provides a combination of MAC layer and network layer adaptation that effectively deals with such issues. To the best of our knowledge, no routing algorithm has been specifically designed to provide soft real-time guarantees under spatiotemporal constraints for sensor networks.

3 Design Goals

The key design goal of the SPEED algorithm is to support a spatiotemporal communication service with a desired delivery *speed* across the sensor network, so that end-to-end delay is proportional to the distance between the source and destination. It should be noted that delivery speed refers to the approaching rate along a straight line from the source toward the destination. Unless the packet is routed exactly along that straight line, delivery speed is smaller than the actual speed of the packet in the network. For example, if the packet is routed in the opposite direction from the destination, its speed is negative. Our algorithm ensures that this condition never occurs.

More specifically, SPEED satisfies the following design objectives.

1. **Soft Real-Time:** We define the soft real-time guarantee provided by SPEED as delay guarantee per unit delivery distance (speed guarantee). Under this guarantee, we can obtain a predictable end-to-end communication delay under given spatial (distance) constraints before hand. Consequently, applications can make admission control to deliver packets that are able to meet end-to-end deadlines.
2. **Minimal State Architecture:** The physical limitations of sensor networks, such as large scale, high failure rate, and constrained memory capacity necessitate a minimal state approach. SPEED only maintains immediate neighbor information. It does not require a routing table as in DSDV [18] nor per-destination states as in AODV [17]. Thus, its memory requirements are minimal.
3. **Minimum MAC Layer Support:** SPEED does not require real-time MAC support. The feedback control scheme employed in SPEED allows all existing best effort MAC layers.
4. **QoS Routing and Congestion Management:** Most reactive routing protocols can find routes that avoid network hot spots during the route acquisition phase. Such protocols work well when traffic patterns do not fluctuate during a session. However, these protocols (e.g., [10]) are less successful when congestion patterns change rapidly compared to the session lifetime. When a route becomes congested, such protocols either suffer a delay or initiate another round of route discovery. As a solution, SPEED uses a novel backpressure rerouting scheme to reroute packets around large-delay links with minimum control overhead.
5. **Traffic Load Balancing:** In sensor networks, the bandwidth and energy are scarce resources compared to a wired network. Because of this, it is valuable to utilize several simultaneous paths to carry packets from the source to the destination. SPEED uses nondeterministic forwarding to balance each flow among multiple concurrent routes.
6. **Localized Behavior:** Pure localized algorithms are those in which any action invoked by a node should not affect the system as a whole. In algorithms such as AODV, DSR, and TORA, this is not the case. In these protocols, a node uses flooding to discover new paths. In sensor networks where thousands of nodes communicate with each other, broadcast storms may result in significant power consumption and possibly a network meltdown. To avoid that, all distributed operations in SPEED are localized to achieve high scalability.
7. **Void Avoidance:** In some scenarios, pure greedy geographic forwarding may fail to find a greedy path to the destination, even when one actually exists. SPEED handles the void the same way as it handles congested areas and guarantees that if there is a greedy route between the source and destination, it will discover it.

Note, while SPEED does not use routing tables, SPEED does utilize location information to carry out routing. Because of this, we assume that each node is location-aware [11].

4 Speed Protocol

SPEED maintains a desired delivery speed across sensor networks with a two-tier adaptation included for diverting traffic at the networking layer and locally regulating packets sent to the MAC layer. It consists of the following components:

- An API.
- A delay estimation scheme.
- A neighbor beacon exchange scheme.
- A Nondeterministic Geographic Forwarding algorithm (NGF).
- A Neighborhood Feedback Loop (NFL).
- Backpressure Rerouting.
- Last mile processing.

As shown in Fig. 1, NGF is the routing module responsible for choosing the next hop candidate to support the desired delivery speed. NFL and Backpressure Rerouting are two modules to reduce or divert traffic when congestion occurs, so that NGF has available candidates from which to choose. The last mile process is provided to

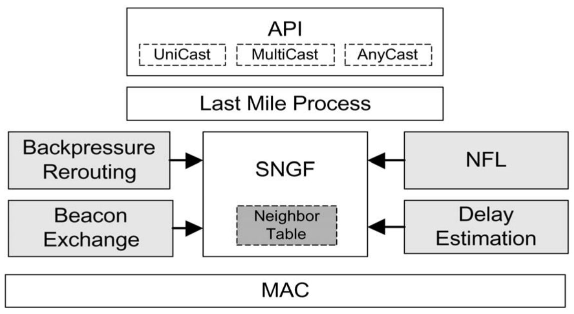

Fig. 1. SPEED protocol.

support three types of real-time communication services, namely, real-time unicast, real-time area-multicast, and real-time area-anycast, for sensor networks. Delay estimation is the mechanism by which a node determines whether congestion has occurred. And beacon exchange provides geographic location of the neighbors so that NGF can perform geographic based routing. The details of these components are discussed in the subsequent sections, respectively.

4.1 Application API and Packet Format

The SPEED protocol provides four application-level API calls:

- AreaMulticastSend (position, radius, packet): This service identifies a destination area by its center position and radius. It sends a copy of the packet to every node inside the specified area with a speed above a certain desired value.
- AreaAnyCastSend (position, radius, packet): This service sends a copy of the packet to at least one node inside the specified area with a speed above a certain desired value.
- UnicastSend(Global_ID, packet): In this service the node identified by Global_ID receives the packet with a speed above a certain desired value.
- SpeedReceive(): this primitive permits nodes to accept packets targeted to them.

There is a single data packet format in SPEED. It contains the following major fields:

- PacketType: the type of communication—Area Multicast, AreaAnyCast or Unicast.
- Global_ID: only used in Unicast communication to identify a destination node.
- Destination Area: Describes a three-dimensional space with a center point and radius in which the packets are destined.
- TTL: Time To Live field is the hop limit used for last mile processing.
- Payload.

4.2 Delay Estimation

We use single hop delay as the metric to approximate the load of a node. We notice that the delays experienced by broadcast packets and unicast packets are quite different due to different handling inside the MAC layer. Unicast packet delay is more appropriate for making routing decisions. In a scarce bandwidth environment, we cannot afford to use probing packets to estimate the single hop delay. Instead, we use the data packets passing this node to perform this measurement. Delay estimation is done at the sender, which timestamps a packet ($\mathrm{T}_{\mathrm{arriving}}$) entering the tail of network output queue and time-stamps the packet ($\mathrm{T}_{\mathrm{departure}}$) when the last bit of this packet is sent out. Single trip delay equals the interval between $\mathrm{T}_{\mathrm{arriving}}$ and $\mathrm{T}_{\mathrm{departure}}$. Propagation delay is ignored. In case of transmission failures, $\mathrm{T}_{\mathrm{departure}}$ is set and used for calculation only at a successful transmission.

We compute the current delay estimation by combining the newly measured delay with previous delays via the exponential weighted moving average (EWMA) [15] as following:

$$Delay(k) = \alpha^{*} Delay_{new} + (1-\alpha)^{*} Delay(k-1).$$

We argue that this delay estimation is a better metric than average queue size for representing the congestion level of the wireless network, because the shared media nature of the wireless network allows the network to be congested even if queue sizes are small.

4.3 Neighbor Beacon Exchange

Similar to other geographic routing algorithms, every node in SPEED periodically broadcasts a beacon packet to its neighbors. This periodic beaconing is only used for exchanging location information between neighbors. We argue that the beaconing rate can be very low when nodes inside the sensor network are stationary or slow moving. Moreover, piggybacking [16] methods can also be exploited to reduce this beacon overhead.

In addition to periodic beaconing, SPEED uses an on-demand backpressure beacon, to quickly notify the upstream nodes of traffic changes inside the network. As shown in the evaluation (Section 6.4), our on-demand beacon scheme introduces only a small overhead in exchange for a fast response to congestion.

In SPEED, each node keeps a neighbor table to store information passed by the beaconing. Each entry inside the table has the following fields: (NeighborID, Position, SendToDelay, ExpireTime). The ExpireTime is used to timeout this entry. If a neighbor entry is not refreshed after a certain timeout, it is removed from the neighbor table. SendToDelay is the delay estimation to the neighbor node identified by the NeighborID field. The details of obtaining this value have been discussed in Section 4.2.

4.4 Nondeterministic Geographic Forwarding

Before elaborating on Nondeterministic Geographic Forwarding (NGF), we first introduce three definitions:

- The Neighbor Set of Node i: NS_i is the set of nodes within the radio range of node i. Note, we do not assume that the radio is a perfect circle. SPEED works with irregular radio patterns.
- The Forwarding Candidate Set of Node i: A set of nodes that belong to NS_i and are closer to the destination. Formally, $FS_i(Destination) = \{node \in NS_i \mid L - L_next > 0\}$, where L is the distance from node i to the destination and L_next is the distance from the next hop forwarding candidate to the destination. These nodes are inside the cross-hatched shaded area as shown in Fig. 2. We can easily obtain $FS_i(Destination)$ by scanning the NS set of nodes once.

  It is worth noticing that the membership of the neighbor set only depends on the radio range, but

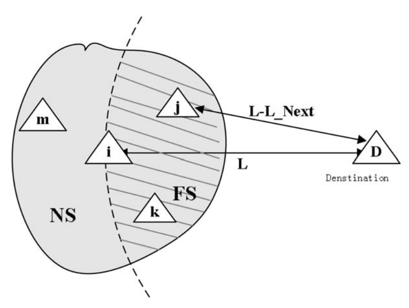

Fig. 2. NS and FS definitions.

the membership of the forwarding set also depends on destination area.

- Relay Speed. Relay speed is calculated by dividing the advance in distance from the next hop node j by the estimated delay to forward a packet to node j. Formally, $Speed_i^j(Destination) = \frac{L-L\_next}{HopDelay_i^j}$.

Since in SPEED nodes keep the Neighbor Set (NS), but do not keep a routing table or flow information, the memory requirements are only proportional to the number of neighbors.

Based on the destination of the packet and the current FS, the Nondeterministic Geographic Forwarding (NGF) portion of our protocol routes the packets according to the following rules:

1. Packets are forwarded only to the nodes that belong to the $FS_i(Destination)$. If there is no node inside the $FS_i(Destination)$, packets are dropped and a backpressure beacon is issued to upstream nodes to prevent further drops (for more details, see Section 4.6). To reduce the chance of such drops, we provide a lower bound of node density that can virtually eliminate these drops. The theoretical analysis on this issue is provided in Section 5.1.
2. SPEED divides the neighbor nodes inside FS_i $(Destination)$ into two groups. One group contains the nodes that have relay speeds larger than a certain desired speed $S_{setpoint}$, the other contains the nodes that cannot sustain such desired speed. The $S_{setpoint}$ is a system parameter that depends on the communication capability of the nodes and desired traffic workload a sensor network should support. For given bandwidth T, packet size L and Radio Range R, following inequality should hold:
$$0 \leq S_{setpoint} \leq RT/L.$$
3. The forwarding candidate is chosen from the first group, and the neighbor node with highest relay speed has a higher probability to be chosen as the forwarding node. To trade off between load balancing and optimal delivery delay, we use following discrete exponential distribution function:
$$P(x=i) = \frac{(Speed_i)^K}{\sum\limits_{i=0}^{N} (Speed_i)^K} \quad 1 \leq i \leq N.$$
In this distribution function, N is the number of forwarding candidates inside the first group. K is used to trade off between load balance and optimal delivery delay. A larger K value leads to a shorter end-to-end delay; while a smaller K value achieves a better load balance.
4. If there are no nodes belonging to the first group, a relay ratio is calculated based on the Neighborhood Feedback Loop (NFL), which is discussed in more detail in Section 4.5. Whether a packet drop will really happen depends on whether a randomly generated number between (0, 1) is bigger than the relay ratio. In SPEED, a packet is dropped only when no downstream node can guarantee the single hop speed set point $S_{setpoint}$ and dropping packets must be performed to reduce the congestion. Though one can consider buffering packets as an alternative to the dropping, however, we argue that under real-time and small memory constraints, dropping is often a better choice.

NGF provides two nice properties to help meet our design goals. First, since NGF sends packets to the downstream node capable of maintaining the desired delivery speed, soft real-time end-to-end delivery is achieved with a theoretical delay bound: $DelayBound = L_{e2e}/S_{setpoint}$, where L_{e2e} is the distance between the source and destination. $S_{setpoint}$ is the uniform speed to be maintained across the sensor network. Second, NGF can balance traffic and reduce congestion by dispersing packets into a large relay area. This load balancing is valuable in a sensor network where the density of nodes is high and the communication bandwidth is scarce and shared. Load balancing also balances the power consumption inside the sensor networks to prevent some nodes from dying faster than others.

NGF provides MAC layer adaptation and reduces the congestion by locally dropping (or optionally buffering) packets. This adaptation is good enough to deal with transient overshoot inside the sensor networks. But, if such congestion remains for a relatively long time, network layer adaptation is desired to redirect traffic to a less congested area, which is discuss further in Section 4.6.

4.5 Neighborhood Feedback Loop (NFL)

The Neighborhood Feedback Loop (NFL) is the key component in maintaining the single hop relay speed. The NFL is an effective approach to maintaining system performance at a desired value. This has been shown in [19], where a low miss ratio of real-time tasks and a high utilization of the computational nodes are simultaneously achieved. Here, we want to maintain a single hop relay speed above a certain value $S_{setpoint}$, a performance goal desired by the designer.

We deem it a miss when a packet delivered to a certain neighbor node has a relay speed less than $S_{setpoint}$, or if there is a loss due to collision. The percentage of such misses is called this neighbor's miss ratio. The responsibility of the NFL is to force the miss ratios of the neighbors to converge to a set point, namely zero.

As shown in Fig. 3, the MAC layer collects miss information and feeds it back to the Relay Ratio controller. The Relay Ratio controller calculates the relay ratio and feeds that into the NGF where a drop or relay action is made. The Relay Ratio controller currently implemented is a multiple inputs single output (MISO) proportional controller that takes the miss ratios of its neighbors as inputs and proportionally calculates the relay ratio as the output to the NGF. Formally, it is described by the following formulas.

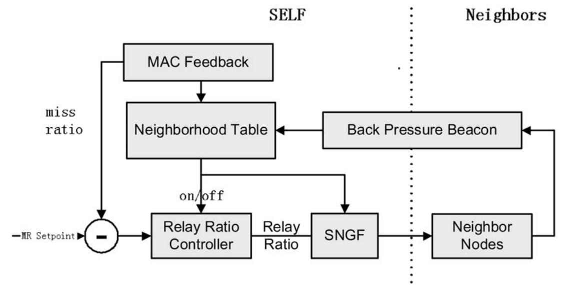

Fig. 3. Neighborhood Feedback Loop (NFL).

$$u = 1 - K\frac{\sum e_i}{N} \quad if\ \forall e_i > 0$$

$$u = 1 \quad if\ \exists e_i = 0,$$

where e_i is the miss ratio of the neighbor i inside the FS set, N is size of the FS set. u is the output (relay ratio) to NGF. And, K is the proportional gain.

It should be noted that the Relay Ratio controller is activated only when all nodes inside the forwarding set (FS) cannot maintain the desired single hop relay speed $S_{setpoint}$ and a drop is absolutely necessary to maintain the single hop delay. Such a scheme ensures that rerouting has a higher priority than dropping. In other words, SPEED does not drop a packet as long as there is another path that can meet the delay requirements.

By reducing the sending rate to the downstream nodes, the neighborhood feedback loop can maintain a single hop relay speed. However, this MAC layer adaptation cannot solve the hotspot problem, if the upstream nodes, which are unaware of the congestion, keep sending packets into this area. In this case, backpressure rerouting (network layer adaptation) is necessary to reduce the traffic injected into the congested area.

4.6 Back-Pressure Rerouting

Backpressure rerouting is naturally generated from the collaboration of neighbor feedback loop (NFL) routines as well as the nondeterministic geographic forwarding (NGF). To be more explicit, we introduce this scheme with an example (Fig. 4).

Suppose in the lower-right area, heavy traffic appears, which leads to a low relay speed in nodes 9 and 10. Through the MAC layer feedback, node 5 detects that nodes 9 and 10 are congested. Since NGF reduces the chance of selecting nodes 9 and 10 as forwarding candidates and routes more packets to node 7, it reduces the congestion around nodes 9 and 10. Since all neighbors of 9 and 10 react

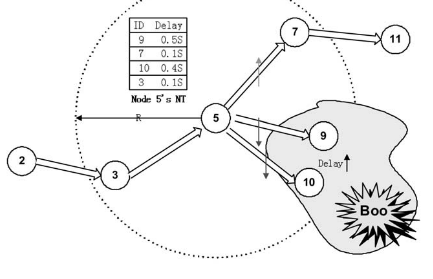

Fig. 4. Backpressure rerouting case one.

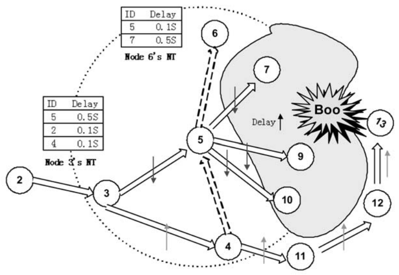

Fig. 5. Backpressure rerouting case two.

the same way as node 5, eventually nodes 9 and 10 are able to relay packets above the desired speed.

A more severe case could occur when all the forwarding neighbors of node 5 are also congested as shown in Fig. 5. In this case, the neighborhood feedback loop is activated to assist backpressure rerouting. In node 5, a certain percent of packets are dropped in order to reduce the traffic injected into the congested area. At the same time, an on-demand backpressure beacon is issued by node 5 with the following fields: (ID, Destination, AvgSendToDelay).

AvgSendToDelay is the average SendToDelay of all nodes inside $\mathrm{FS}_{\mathrm{ID}}(\mathrm{Destination})$. In our example, when the destination is node 13, AvgSendToDelay is the average delay from node 5 to nodes 7, 9, and 10.

When a neighbor receives the back-pressure beacon from node 5, it determines whether node 5 belongs to its $FS(\mathrm{Destination})$. If node 5 does, this neighbor modifies the SendToDelay for node 5 according to the AvgSendToDelay. For example, only node 3 considers node 5 as a next hop forwarding candidate to the destination where node 13 resides. If node 5 is not in the $FS(\mathrm{Destination})$, then this neighbor ignores the backpressure beacon. This backpressure mechanism can reduce the chance of "false congestion indication," to ensure that traffic from node 4 to node 6 is not affected by the backpressure beacon.

If, unfortunately, node 3 is in the same situation as node 5, further backpressure is imposed on node 2. In the extreme case, the whole network is congested and the backpressure proceeds upstream until it reaches the source, where the source quenches the traffic flow to that destination.

Backpressure rerouting is a network layer adaptation used by SPEED to reduce the congestion inside the network. In this case, no packet needs to be sacrificed. Network layer adaptation has a higher priority than MAC layer adaptation used by NGF and NFL. A drop via the feedback loop is only necessary when the situation becomes so congested and there is no alternative to maintaining a single hop speed other than dropping packets.

4.7 Void Avoidance

Greedy geographic-based algorithms have many advantages over the traditional MANET routing algorithms for real-time sensor network applications. They do not suffer route discovery delay and tend to choose the shortest path to the destination. Moreover without flooding, they have relatively low control packet overhead. Unfortunately, they also have a serious drawback. In many cases, they may fail to find a path even though one does exist. To overcome this, SPEED deals with a void the same way it deals with congestion. As shown in Fig. 6, if there is no downstream node to relay packets from node 2 to node 5, node 2 sends

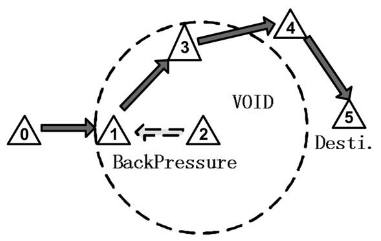

Fig. 6. Void avoidance scheme.

out a backpressure beacon containing fields: (ID, Destination, ∞). The upstream node 1 that needs node 2 to relay the packets to that destination sets the SendToDelay for node 2 to infinity and stop sending packets to node 2. If node 3 does not exist, further backpressure occurs until a new route is found. It should be admitted that our scheme of void avoidance is not guaranteed to find a path if there is one as in GPSR [16], but it is guaranteed to find a *greedy* path if one exists. To maintain real-time properties, we do not allow backtracking to violate our desired speed setpoint. However, as we can see from the evaluation Section 6.6, such a simple scheme can significantly reduce packet loss due to voids in high-density sensor networks.

4.8 Last Mile Process

Since SPEED is targeted at sensor networks where the ID of a sensor node is not important, SPEED only cares about the location where sensor data is generated.

The last mile process is so-called because only when the packet enters into the destination area will such a function be activated. The NGF module mentioned above controls all previous packet relays.

The last mile process provides two novel services that fit the scenario of sensor networks: Area-multicast and Area-anycast. The area in this case is defined by a center-point (x, y, z) and a radius, in essence, a sphere. More complex area definitions can be made without jeopardizing the design of this last mile process.

Nodes can differentiate the packet type by the PacketType field mentioned in Section 4.1. If it is an anycast packet, the nodes inside the destination area deliver the packet to the transport layer without relaying it onward. If it is a multicast packet, the nodes inside the destination area which first receive the packet coming from the outside of the destination area set a TTL. This allows the packet to survive within the diameter of the destination area and be broadcast within a specified radius. Other nodes inside this destination area keep a copy of the packet and rebroadcast it. The nodes that are outside the destination area just ignore it. The last mile process for unicast is nearly the same as multicast, except that only the node with a specified global_ID inside the destination area delivers the packet to the transport layer. If the location service is precise, the estimated destination area for a given node will be much smaller than single radio coverage. As a result, additional flooding overhead for the unicast packets is negligible (sometimes zero). We note that the current implementation of the last mile process is relatively straightforward. More efficient and robust techniques are desired for future research.

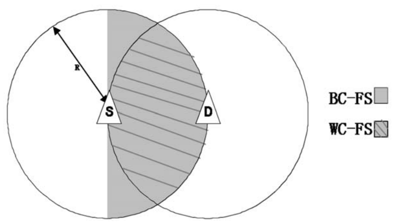

Fig. 7. Forwarding areas.

5 Protocol Analysis

This section provides a protocol analysis of several practical issues related to the SPEED protocol.

5.1 Impact of Node Density

One basic assumption of sensor networks is their relatively high node density. It is an interesting research issue to determine the impact of node density on routing performance. Specifically, in the geographic-based algorithm, we want to find the lower bound of node density that can probabilistically guarantee no void that can prevent a greedy geographic forwarding step from happening.

In GF-based algorithms, a node forwards packets to next hop nodes that are nearer to the destination. The area where such qualified nodes reside is called the *forwarding area (FA)*.

Assume the nodes are randomly distributed inside the system, the larger the size of the forwarding area, the higher is the probability that there is a candidate to be chosen. In fact, the forwarding area size is not constant; it depends on how far away the sending node is from the destination node.

As shown in Fig. 7, when the destination node is infinitely far away from the sending node, the forwarding area is the largest (Best Case Forwarding Size) and when the destination node is exactly R away from the sending node, the available forwarding size is the worst case forwarding size (WCFS). For guaranteeing purposes, we only consider the worst case, even though most of the time the forwarding size is nearer to the best case. In the worst case, the forwarding size is calculated by (1). For the purpose of analysis, here we use R as a nominal radio radius.

$$WCFS = \left[2\cos^{-1}\frac{1}{2} - \frac{\sqrt{3}}{2}\right] R^2. \tag{1}$$

Now, we consider the worst-case forwarding area. We desire to know the lower bound of node density that satisfies the following condition: We desire to know the lower bound of node density that satisfies the following condition:

$$\mathrm{P}\,(\text{At least one node inside the FA}) > 1 - \varepsilon. \tag{2}$$

Equivalently,

$$\mathrm{P}\,(\text{No nodes inside the FA}) <= \varepsilon. \tag{3}$$

Assuming a uniform distribution, according to (3), the following condition must hold (the size of the area covered by the sensor network is denoted by AreaSize >> WCFS):

$$\left(1 - \frac{WCFS}{AreaSize}\right)^{AreaSize\ \times Density} \le \varepsilon. \tag{4}$$

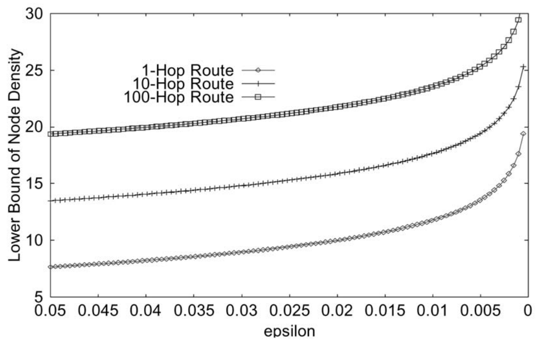

Fig. 8. Lower bound of node densities.

Since the left-hand side of the inequality is a monotonically increasing function when the AreaSize increases and monotonically decreasing when node density increases, the lower bound of the node density is achieved when AreaSize is infinite:

$$\lim\left(1-\frac{WCFS}{AreaSize}\right)^{AreaSize\ \times Density} = e^{-WCFS\times Density} \leq \varepsilon$$

$$\text{Hence}:\ Density \geq \frac{-\ln\varepsilon}{WCFS}. \tag{5}$$

As for the greedy geographic-based routing algorithm without backpressure, we must guarantee that for every hop they can find a forwarding candidate. More formally, to guarantee:

$$\begin{aligned}&\text{P(successfully deliver packets to a destination}\\&\text{through \#hop greedy forwarding)} >= 1-\varepsilon.\end{aligned} \tag{6}$$

Assume voids follow an identical independent distribution (iid), equivalently:

$$[\text{P(At least one node inside the FA)}]^{\#\text{hop}} > 1-\varepsilon. \tag{7}$$

Following the same derivation from (2) to (5), we get the lower bound of node density:

$$Density \geq \frac{-\ln\left(1-\sqrt[\#hop]{1-\varepsilon}\right)}{WCFS}. \tag{8}$$

Fig. 8 shows the lower bound of node densities that can probabilistically guarantee that there is no void that can prevent greedy routing under different ε values and lengths of the routes. For example, for a 10-hop route, it is statistically guaranteed that 99 percent delivery ratio in worst case if the node density inside networks is above 16 node/nominal range.

On the other hand, we need to guarantee there exist a greedy path to the destination for the SPEED protocol. We observe that SPEED cannot enforce backpressure at the source node, where no upstream node exists. After the first hop relay, the backpressure effectively reduces packet lost due to the void at subsequent hopes as mentioned in Section 4.7. To simplify the analysis, we only consider first hop loss due to the void. This approximation slightly overestimates the delivery ratio of SPEED and serves as an upper bound. According to inequality 7, Fig. 9 plots the estimate delivery rate under different node densities.

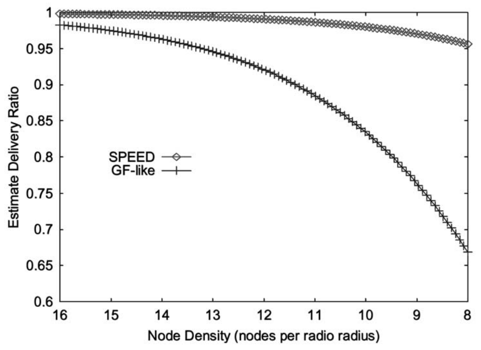

Fig. 9. Estimate Deliver ratio.

We note that the result we obtained from the formal analysis (Fig. 9) is quite similar to the results obtained through simulation (Section 6.6); For example, both simulation and formal analysis have about 95 percent delivery ratio for SPEED at the density of 8 nodes per radio circle.

5.2 Analysis of Localization Impact

Theoretically, it is desirable for location-based routing algorithms to have a perfect localization service; however, in practical, in order to obtain higher location accuracy, systems have to increase the cost of the localization via sophisticated devices or additional communication overhead. Although more accurate location information is preferable, the desired level of granularity should depend on a cost/benefit analysis of the protocols that utilize this information. In this section, we investigate the impact of localization errors on the SPEED protocol. Specifically, we investigate the pseudovoid problem caused by localization errors, which leads to routing failures in SPEED and other location-based routing algorithms.

5.2.1 Pseudovoid Problem

Location-based routings normally follow a greedy forwarding rule, as long as there is at least a node inside forwarding area. Due to the localization error, some nodes, which are actually located inside the forwarding area of a sender, might be mistaken by the sender to be outside (Fig. 10a).

More specifically, the pseudovoid problem happens when all nodes inside the forwarding area get localization results that are outside of the forwarding area of the sender. We note that the forwarding area can be shifted by the localization error of the sender. Assuming that the localization error of each node follows an identical independent distribution (iid),

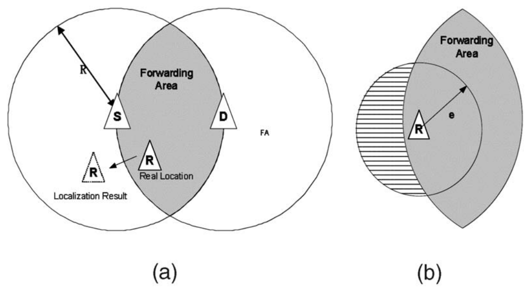

Fig. 10. Pseudovoid problem.

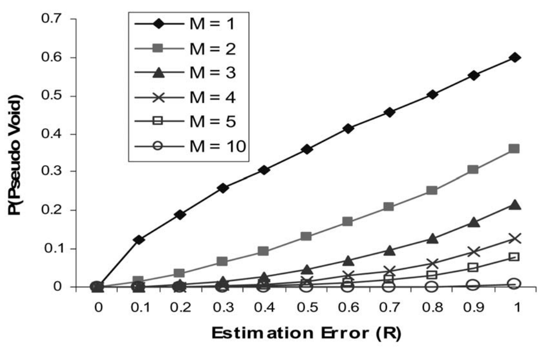

Fig. 11. Probability of pseudovoid problem.

$$\begin{aligned}&\mathrm{P(pseudovoid)} = \prod_{\mathrm{i=1}}^{\mathrm{M}} \mathrm{P}(\mathit{Localization}(N_i) \notin \\ &FA(\mathit{Localization}(\mathit{sender}))|\mathrm{Location}(Ni) \in \\ &FA(\mathit{Location}(\mathit{sender}))),\end{aligned} \tag{9}$$

where FA is the forwarding area, N_i is an arbitrary node inside forwarding area, and M is the number of nodes inside forwarding area.

Assume localization error is omnidirectional and maximum distance between the estimated location and the real location is e. According to Fig. 10b, P (pseudovoid) equals:

$$\prod_{i=1}^{M}\left[\frac{1}{\pi e^2} \times \iint_{m^2+n^2\le e^2} \left[\iint_{FA(m,n)} \frac{Size_{dashedArea}}{\pi e^2} dxdy\right] dmdn\right], \tag{10}$$

where $FA(m,n)$ denotes the forwarding area of the sender when the localization result of the sender is (m, n). We obtain the value of (8) through numerical integration. We only consider the worst-case in which the destination is exactly R away from sending node. The results are shown in Fig. 11.

Fig. 11 gives us insight on the relationship between node density (M denotes the number of nodes inside forwarding area), estimation error (in the unit of radio range) and the probability of the pseudovoid problem occurring. For example, in order to reduce chance of drop due to the pseudo void problem below 5 percent, the number of nodes inside the forwarding area should be equal to or larger than 3 when localization error is as much as half radio range. Based on the worse case forwarding area size given in (1), the corresponding node density should be 7.67 node/nominal radio circle

$$Density \ge \frac{\pi R^2}{WCFS} \times 3 = 7.67\, nodes/circle. \tag{11}$$

Fig. 11 also demonstrates that when node density is sufficiently high (M = 5 & corresponding node density $>$ 12.78), statistically the pseudoproblem rarely happens ($<$ 0.2 percent) when the localization error below half radio range. This theoretical analysis is consistent with our simulation result in [11].

6 Experimentation and Evaluation

We simulate SPEED on GloMoSim [22], a scalable discrete-event simulator developed by UCLA. This software provides a high fidelity simulation for wireless communication with detailed propagation, radio and MAC layers. Table 1 describes the detailed setup for our simulator. The communication parameters are mostly chosen in reference to the Berkeley Telos mote specification.

In our evaluation, we compare the performance of seven different routing algorithms: AODV [17], DSR [10], GF [20], GPSR [12], SPEED, SPEED-S, SPEED-T. We adopt both ad hoc routing protocols (AODV and DSR) and sensor network protocols (GF, GPSR).

GF forwards a packet to the node that makes the most progress toward the destination. GPSR has identical performance as GF when network density is relatively high, however, it achieves better delivery ratios in sparse networks. SPEED-S and SPEED-T are reduced versions of SPEED. SPEED-S replaces the NGF with a MAX-SPEED routing algorithm that geographically forwards the packets to nodes that can provide a max single hop relay speed. SPEED-T replaces the NGF with a MIN-DELAY routing algorithm that geographically forwards packets to nodes that have a minimum single hop delay. Both reduced versions have no backpressure rerouting mechanisms.

In our evaluation, we present the following set of results:

1. end-to-end delay under different congestion levels,
2. miss ratio,
3. control overhead,
4. communication energy consumption, and
5. packet delivery ratio under different node densities.

All experiments are repeated 16 times with different random seeds and different random node topologies. We also implement SPEED on the Berkeley. The results obtained from this testbed show a load balance feature of SPEED protocol (see Section 6.8).

TABLE 1
Simulation Settings

Routing	AODV, DSR, GF, GPSR, SPEED, SPEED-S, SPEED-T
MAC Layer	802.11 (Simplified DCF)
Radio Layer	RADIO-ACCNOISE
Propagation model	TWO-RAY
Bandwidth	200Kb/s
Payload size	32 Byte
TERRAIN	(200m, 200m)
Node	100 Nodes, Uniform placement
Radio Range	40m

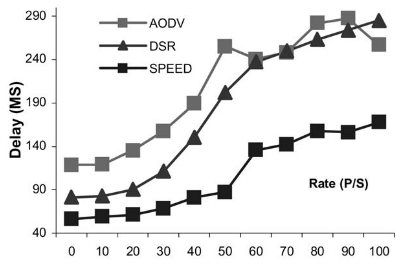

Fig. 12. E2E delay versus congestion.

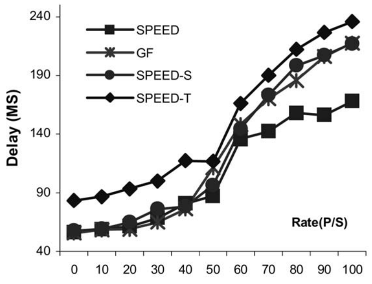

Fig. 13. E2E delay versus congestion.

6.1 Sensor Network Traffic Pattern

There are two typical traffic patterns in sensor networks: a base station pattern and a peer-to-peer pattern. The base station pattern is the most representative one inside sensor networks. For example, in surveillance systems, multiple sensors detect and report the location of an intruder to the control center. In tracking systems, a base station issues multiple tracking commands to a group of pursuers. In a different respect, the peer-to-peer pattern is usually used for data aggregation and consensus in a small area where a team of nearby motes interact with each other. The end-to-end delay in the base station pattern is the major part of delay for the sensing-actuation loop, and is therefore the focus of our evaluation.

6.2 Congestion Avoidance

In a sensor network, where node density is high and bandwidth is scarce, traffic hot spots are easily created. In turn, such hot spots may interfere with real-time guarantees of critical traffic in the network. In SPEED, we apply a combined network and MAC layer congestion control scheme to alleviate this problem.

To test the congestion avoidance capabilities, we use a base station scenario, where six nodes, randomly chosen from the left side of the terrain, send periodic data to the base station at the middle of the right side of the terrain. The average hop count between the node and base station is about eight to nine hops. Each node generates 1 CBR flow with a rate of 1 packet/second. To create congestion, at time 80 seconds, we create a flow between two randomly chosen nodes in the middle of the terrain. This flow then disappears at time 150 seconds into the run. This flow introduces a step change into the system, which is an abrupt change that stress-tests SPEED's adaptation capabilities to reveal its transient-state response. In order to evaluate the congestion avoidance capability under different congestion levels, we increase the rate of this flow step by step from 0 to 100 packets/second over several simulations.

Fig. 12 and Fig. 13 plot the end-to-end (E2E) delay for the six different routing algorithms. At each point, we average the E2E delays of all the packets from the 96 flows (16 runs with six flows each). The 90 percent confidence interval is within 2 to 15 percent of the mean, which is not plotted for the sake of legibility.

Under the no or light congestions, Fig. 12 and Fig. 13 show that all geographic based routing algorithms have short average end-to-end delay in comparison to AODV and DSR. There are several factors accounting for this outcome. First, the route acquisition phase in AODV and DSR leads to significant delays for the first few packets, while geographic-based routing does not suffer from this. We argue that without an initial delay cost, geographic-based routing is more suitable for real-time applications like target tracking where the base station sends the actuation commands to the sensor group, which is dynamically changing as the target moves. In such a scenario, DSR and AODV need to perform route acquisition repeatedly in order to track the target. Second, the route discovered through flooding and path reversal has relatively more hops than greedy geographic forwarding. The reason for even higher delay in AODV than DSR is that DSR implementation intensively uses a route cache to reduce route discovery and maintenance cost. As shown in Fig. 13, SPEED-T has higher delay than GF, SPEED-S, and SPEED, because SPEED-T only uses hop delay to make routing decision and disregards the progress each hop makes, which leads to more hops to the destination in wireless multihop networks. Instead, under lightly congested situation, GF, SPEED-S, and SPEED tend to forward a packet at each step as close to the destination as possible, thereby reducing the number of hops and the end-to-end delay.

Under the heavy congested situations (Fig. 12 and Fig. 13), each routing algorithm responds differently. SPEED performs best. For example, SPEED reduces the average end-to-end delay by 30 percent to 40 percent in the face of heavy congestion in comparison to the other algorithms considered. The key reasons for SPEED's better performance are:

1. DSR, AODV, and GF only respond to severe congestion, which leads to link failures (i.e., when multiple retransmissions fail at the MAC layer). They are insensitive to long delays as long as no link failures occur.
2. DSR, AODV, and GF routing decisions are not based on the link delays, and therefore may cause congestion at a particular receiver even though it has long delays.
3. DSR and AODV flood the network to rediscover a new route when the network is already congested.
4. SPEED-T and SPEED-S do not provide traffic adaptation. When all downstream nodes are congested, SPEED-T and SPEED-S cannot reduce or redirect the traffic to uncongested routes.
5. SPEED not only locally reduces the traffic through a combination of NGF and Neighborhood Feedback loops in order to maintain the desired speed, but also diverts the traffic into a large area through its backpressure rerouting mechanism. This combination leads to lower end-to-end delay.

6.3 E2E Deadline Miss Ratio

The deadline miss ratio is the most important metric in soft real-time systems. We set the desired delivery speed $S_{setpoint}$ to 1km/s, which leads to an end-to-end deadline of

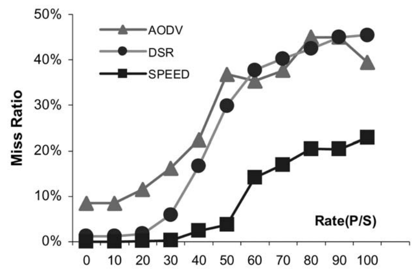

Fig. 14. MissRatio versus congestion.

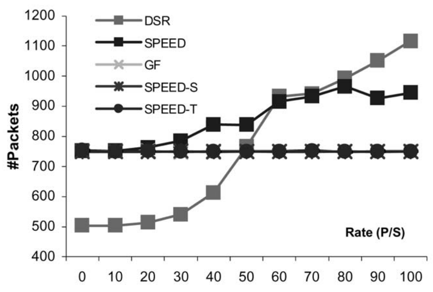

Fig. 16. Control overhead comparison.

200 milliseconds. In the simulation, some packets are lost due to congestion or forced-drops. We also consider this situation as a deadline miss. The results shown in Fig. 14 and Fig. 15 are the summary of 16 randomized runs.

AODV and DSR do not perform well in the face of congestion because both algorithms flood the network in order to discover a new path when congestion leads to link failure. This flooding just serves to increase the congestion. GF only switches the route when there are link failures caused by heavy congestion. The routing decision is based solely on distance and does not consider delay. SPEED-T only considers the single hop delay and does not take distance (progress) into account, which leads to a longer route. SPEED-S provides no adaptation to the congestion and cannot prevent packets from entering the congestion area. Only SPEED tries to maintain a desired delivery speed through MAC and network layer adaptations and, therefore, has a much less miss ratio than other algorithms. Due to its transient behavior, SPEED still has about a 20 percent miss ratio when the network is heavily congested. Future work is needed to reduce the convergence time in order to improve the performance.

Comparing Fig. 14 and Fig. 15, we argue that purely localized algorithms without flooding outperform other algorithms when traffic congestion increases. Generally, the less state information a routing algorithm depends on, the more robust it is in the face of packet loss and congestion.

6.4 Control Packet Comparison

Except for AODV, all other routing algorithms studied use a relatively low number of control packets. Most control packets in DSR and AODV are used in route acquisition. Because AODV initiates route discovery (flooding) whenever a link breaks due to congestion, it requires a large number of control packets. DSR uses a route cache extensively, so it can do route discovery and maintenance with a much lower cost than AODV. The only control packets used in GF, SPEED-S, and SPEED-T (Fig. 16) are periodic beacons, whose number is constant at 750 under different congestion levels. In addition to periodic beacons, SPEED uses two types of on-demand beacons to notify neighbors of the congestion. This costs SPEED more control packets than the other three geographic based routing algorithms (Fig. 16).

6.5 Energy Consumption

Under energy constraints, it is vital for sensor nodes to minimize energy consumption in radio communication to extend the lifetime of sensor networks. From the results shown in Fig. 17, we argue that geographic-based routing tends to reduce the number of hops in the route, thus reducing the energy consumed for transmission. AODV performs the worst as a consequence of sending out many control packets during congestion. DSR has larger average hop counts and more control packets than other geographic-based routing algorithms. SPEED-T only takes delay into account, which leads to longer routes. Fig. 17 shows that SPEED has nearly the same power consumption as GF and SPEED-S when the network is not congested. Under such situations, SPEED tends to choose the shortest route and does not require any on-demand beacons. Under heavy congestion, SPEED has slightly higher energy consumption than GF and SPEED-S, mainly because SPEED delivers more packets to the destination than the other protocols when heavily congested.

6.6 Node Density Impact

The typical density of a sensing-covered sensor network system [12] is about 20 to 30 nodes/radio range in order to provide high fidelity in localization, detection, and tracking. In the previous evaluations, we use a 12 nodes/radio range as a typical setting. However, it is important to understand how SPEED performs under very low-density settings.

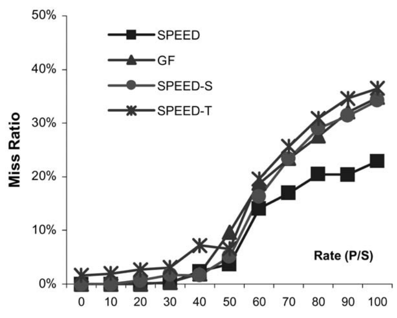

Fig. 15. MissRatio versus congestion.

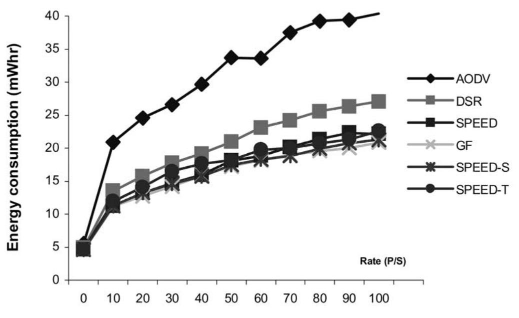

Fig. 17. Transmission energy consumption.

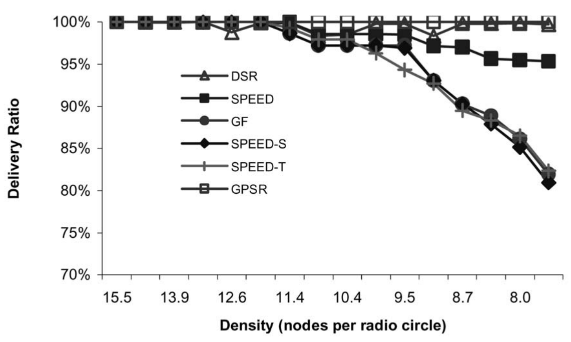

Fig. 18. Deliver ratio versus densities.

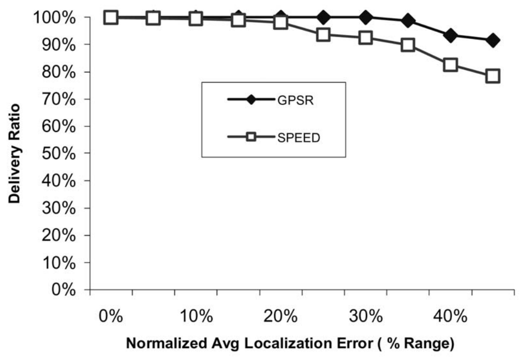

Fig. 19. Delivery ratio versus location errors.

This experiment evaluates the end-to-end delivery ratio of all routing algorithms under different node densities. To eliminate packet loss due to congestion, we only use four flows with a rate of 0.5 packets/second, these flows go from the left side of the terrain to the base station at the right side of the terrain. To change the density of the network, instead of increasing the number of nodes in the terrain, we keep the number of nodes constant at 100, and increase the side length of the square terrain from 180 to 250 in steps of 5 meters. It is no surprise that DSR performs best in the delivery ratio, since it uses flooding to discover the route. Theoretically, DSR should have 100 percent delivery ratio (Fig. 18) as long as the network is not partitioned. All other geographic based algorithms have 100 percent delivery ratio when the network has high density ($>$ 12 nodes/per radio range). However, when the network density is reduced below 9 nodes/per radio range, GF, SPEED-S and SPEED-T degrade performance rapidly. Only SPEED manages to deliver 95 percent of its packets to the destination. It should be pointed out that as shown in Fig. 18, GPSR [16], another well-known geographic-based routing algorithm, permits backtracking and can achieve 100 percent delivery rate as long as the network is not partitioned. However, SPEED drops 5 percent of its packets, because these packets need backtracking in order reach the destination. If these packets were to backtrack, these packets would have a negative delivery speed. This is not allowed by SPEED for the sake of maintaining the real-time properties, which is not supported by GPSR. We note that GPSR defaults to the GF protocol when node density is high. The E2E deadline miss ratio of GF in Fig. 15 is significantly higher than SPEED when the network becomes congested.

6.7 Location Error Impact

While most work in location-based routing assumes perfect location information, the fact is that erroneous location estimates are virtually impossible to avoid. In this experiment, we investigate the tolerable localization error bound for the SPEED protocol. To prevent congestion and, therefore, isolate the effects of localization error, the traffic loads are set to the rate of 1 packet/second. We increase the localization error from 0 percent to 50 percent of the radio range in steps of 5 percent to measure the end-to-end delivery ratio. Fig. 19 shows that when the localization error is below 20 percent of the radio range, SPEED can achieve almost 100 percent delivery. We note that because the GPSR protocol allows backtracking, GPSR is more flexible in dealing with location error impact; hence, it achieves a slightly better performance in this case, as shown in Fig. 19.

6.8 Implementation on Motes

We have implemented the SPEED protocol on the Berkeley motes platform with a code size of 6036 bytes (code is available at [10]). Three applications including data placement, target tracking, and CBR are built on top of SPEED.

Due to the physical limitations of the motes, it is extremely difficult to perform as extensive evaluation as we did in the wireless simulator. Considering the space limitation, we only present partial results here as a study in developing a more complete solution on a mote testbed. In the experiment, we use 25 motes to form a 5 by 5 grid. To evaluate the load balancing capability of SPEED, we send a CBR flow from node 24 to node 0 which is the base station. We collect the number of packets relayed by intermediate motes (1 to 23) and compare this with the result obtained from the GF protocol which we also implemented on the motes.

GF relays packets via a fixed route, which leads to unbalance traffic, for example, in Fig. 20, node 14 sends out 98 packets while node 13 does not sent out any packets. SPEED uses nondeterministic forwarding, which can balance energy consumption. We argue that in sensor networks, balanced energy consumption can prevent some nodes from dying faster than others, therefore increasing the network lifetime.

7 Conclusion

Many excellent protocols have been developed for ad hoc networks. However, sensor networks have additional requirements that were not specifically addressed. This paper is the first to address real-time requirements under spatial constraints by maintaining a desired delivery speed across the network through a novel combination of time-aware feedback control and spatial-aware nondeterministic geographic forwarding. The two-tier adaptation at both the

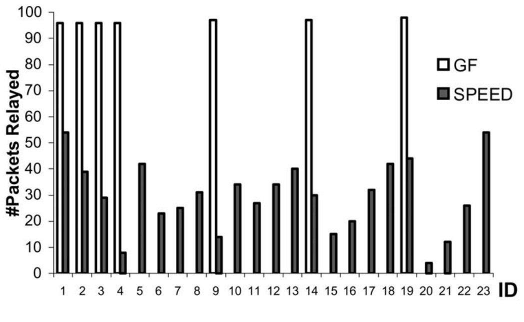

Fig. 20. Traffic balance.

MAC and network layers improves the end-to-end delay and provides good response to congestion and voids. Our simulations on GloMoSim and our implementation on Berkeley motes confirm SPEED's improved performance compared to DSR, AODV, GF, SPEED-S, and SPEED-T.

ACKNOWLEDGMENTS

This work was supported in part by the DAPRPA IXO offices under the NEST project (grant number F336615-01-C-1905), the MURI award N00014-01-1-0576, and US National Science Foundation grant CCR-0098269.

REFERENCES

[1] G.S. Ahn, A.T. Campbell, A. Veres, and L.H. Sun, "SWAN: Service Differentiation in Stateless Wireless Ad Hoc Networks," *Proc. IEEE INFOCOM,* June 2002.

[2] S. Basagni et al., "A Distance Routing Effect Algorithm for Mobility (DREAM)," *Proc. ACM/IEEE MobiCom,* Oct. 1998.

[3] CrossBOW MICA2 Mote Specifications: http://www.xbow.com/, 2005.

[4] Q. Huang, C. Lu, and G.C. Roman, "Spatiotemporal Multicast in Sensor Networks," *Proc. ACM SenSys,* Nov. 2003.

[5] G.G. Finn, "Routing and Addressing Problems in Large Metropolitan-Scale Internetworks," ISI/RR-87-180, USC/ISI, Mar. 1987.

[6] T. He, L. Gu, B. Blum, J. Xie Nest Project Source Code, http://sourceforge.net/projects/vert/, 2005.

[7] T. He, C. Huang, B.M. Blum, J.A. Stankovic, and T.F. Abdelzaher, "Range-Free Localization Schemes in Large Scale Sensor Networks," *Proc. MobiCom,* Sept. 2003.

[8] T. He, S. Krishnamurthy, J.A. Stankovic, T.F. Abdelzaher et al., "Energy-Efficient Surveillance System Using Wireless Sensor Networks," *Proc. MobiSys,* June 2004.

[9] T. He, J.A. Stankovic, C. Lu, and T.F. Abdelzaher, "SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks," *Proc. Int'l Conf. Distributed Computing Systems,* May 2003.

[10] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing,* chapter 5, pp. 153-181, Kluwer Academic Publishers, 1996.

[11] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E.W. Knightly, "Distributed Multi-Hop Scheduling and Medium Access with Delay and Throughput Constraints," *Proc. IEEE Mobicom,* July 2001.

[12] B. Karp and H.T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proc. IEEE MobiCom,* Aug. 2000.

[13] Y.B. Ko and N.H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," *Proc. IEEE MobiCom,* Oct. 1998.

[14] F. Kuhn, R. Wattenhoffer, and A. Zollinger, "Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing," *Proc. MobiHoc,* June 2003.

[15] J.F. Kurose and K.W. Ross, *Computer Networking A Top-Down Approach Featuring the Internet,* ISBN 0-201-47711-4, Addison Wesley Longman Inc., 2005.

[16] C. Lu, B.M. Blum, T.F. Abdelzaher, J.A. Stankovic, and T. He, "RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks," *Proc. IEEE Real-Time and Embedded Technology and Applications Symp.,* Sept. 2002.

[17] C.E. Perkins and E.M. Royer, "Ad-Hoc On Demand Distance Vector Routing," *Proc. Workshop Mobile Computing Systems and Applications,* Feb. 1999.

[18] C.E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *Proc. SIGCOMM Symp. Comm. Architectures and Protocols,* pp. 212-225, Sept. 1994.

[19] J.A. Stankovic, T. He, T.F. Abdelzaher, M. Marley, G. Tao, S. Son, and C. Lu, "Feedback Control Scheduling in Distributed Systems," *Proc. IEEE RTSS,* Dec. 2001.

[20] I. Stojmenovic and X. Lin, "GEDIR: Loop-Free Location Based Routing in Wireless Networks," *Proc. IASTED Int'l Conf. Parallel and Distributed Computing and Systems,* Nov. 1999.

[21] A. Woo and D. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks," *Proc. IEEE Mobicom,* July 2001.

[22] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks," *Proc. PADS,* 1998.

Tian He received the PhD degree from the University of Virginia, Charlottesville, in 2004, and the MS degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2000, and the BS degree from the Nanjing University of Science & Technology, Nanjing, China in 1996. He is author and coauthor of 25 papers at international conference and journals. His research interests include wireless sensor networks, real-time embedded systems, and distributed systems. He is a member of the ACM and IEEE. He will join the University of Minnesota-Twin City this fall as an assistant professor.

John A. Stankovic received the PhD degree from Brown University. He is the BP America Professor in the Computer Science Department at the University of Virginia. He recently served as chair of the department, completing two terms. He is a fellow of both the IEEE and the ACM. He also won the IEEE Real-Time Systems Technical Committee's Award for Outstanding Technical Contributions and Leadership. Professor Stankovic also serves on the Board of Directors of the Computer Research Association. Before joining the University of Virginia, Professor Stankovic taught at the University of Massachusetts where he won an outstanding scholar award. He has also held visiting positions in the Computer Science Department at Carnegie-Mellon University, at INRIA in France, and Scuola Superiore S. Anna in Pisa, Italy. He was the Editor-in-Chief for the *IEEE Transactions on Distributed and Parallel Systems* and is a coeditor-in-chief for the *Real-Time Systems Journal*. His research interests are in distributed computing, real-time systems, operating systems, and wireless sensor networks.

Chenyang Lu received the BS degree from the University of Science and Technology of China, Hefei, China, in 1995, the MS degree from the Chinese Academy of Sciences, Beijing, China, in 1997, and the PhD degree from the University of Virginia, Charlottesville, in 2001, all in computer science. He is an assistant professor in the Department of Computer Science and Engineering at Washington University in St. Louis. He is author and coauthor of more than 35 refereed publications. His current research interests include distributed real-time embedded systems and middleware, wireless sensor networks, and adaptive QoS control. He received a US National Science Foundation CAREER award in 2005. He is a member of the IEEE.

Tarek F. Abdelzaher received the BSc and MSc degrees in electrical and computer engineering from Ain Shams University, Cairo, Egypt, in 1990 and 1994, respectively. He received the PhD degree from the University of Michigan in 1999. Since 1999, he has been an assistant professor at the University of Virginia where he founded the Quality of Service (QoS) Laboratory. He has authored/coauthored more than 60 refereed publications. He was Guest Editor for the *Journal of Computer Communications* and the *Journal of Real-Time Systems*, and is coeditor of *IEEE Distributed Systems Online*. He served on numerous program committees, and held several steering positions including program chair of RTAS 2004. He is a patent holder on adaptive Web systems, an IEEE member, and a US National Science Foundation CAREER award recipient. Tarek's research interests include QoS provisioning, real-time computing, operating systems, computer networking, and sensor networks.