RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks

Chenyang Lu Brian M. Blum Tarek F. Abdelzaher John A. Stankovic Tian He

Department of Computer Science
University of Virginia
{chenyang, bmb5v, zaher, stankovic, th7c}@cs.virginia.edu

Abstract

Large-scale wireless sensor networks represent a new generation of real-time embedded systems with significantly different communication constraints from traditional networked systems. This paper presents RAP, a new real-time communication architecture for large-scale sensor networks. RAP provides convenient, high-level query and event services for distributed micro-sensing applications. Novel location-addressed communication models are supported by a scalable and light-weight network stack. We present and evaluate a new packet scheduling policy called velocity monotonic scheduling that inherently accounts for both time and distance constraints. We show that this policy is particularly suitable for communication scheduling in sensor networks in which a large number of wireless devices are seamlessly integrated into a physical space to perform real-time monitoring and control. Detailed simulations of representative sensor network environments demonstrate that RAP significantly reduces the end-to-end deadline miss ratio in the sensor network.

1. Introduction

With the advances in MEMS devices and embedded processors and radios, it will soon be feasible to deploy large-scale sensor networks to perform distributed microsensing control of physical environments [10]. For example, a surveillance system may use a large network of acoustic sensors to detect and track vehicles in a security area. Similarly, biometric sensors can be deployed in airports and around vital targets to detect harmful bio-agents and issue alarms to command and control centers during potential bio-attacks. These smart sensors and actuators are equipped with low-power processors and short-range radio transceivers [8]. They will automatically form multi-hop ad hoc networks to communicate both among themselves and to remote base stations (e.g., PDA's).

Because distributed micro-sensing involves direct interaction with a physical environment, data communication in sensor networks often has timing constraints in the form of *end-to-end deadlines*. Surveillance may require the position of an intruder be reported to a command center within 5 sec so that pursuing actions can be initiated in time. Data in a system may have different deadlines due to different validity intervals. The validity intervals (and hence, update deadlines) of the locations of different intruders such as pedestrians and motor vehicles may depend on their movement speeds. For example, locations of tanks should have shorter update deadlines than those of pedestrians. Similarly, the location of an intruder should have a shorter update deadline than the temperature measurement of a region because the former can change faster than the latter. Therefore, sensor network protocols should support *real-time* communication by minimizing the packet *deadline miss ratio*, i.e., the percentage of packets that meet their end-to-end deadlines.

While sensor networks share the notion of timing constraints with more traditional embedded systems, they differ in two respects. First, individual sensors are typically very small in size and resource capacity. Hence, the philosophy of sensor networks relies on resource dedication rather than sharing. In other words, individual sensor devices and nodes are likely to be dedicated for individual tasks, thereby eliminating much of the need for sophisticated CPU scheduling in a multitask environment.

Second, it is envisioned that sensors nodes will operate in groups, since individual nodes are too limited and unreliable to perform useful activities from the application's perspective. Group activities require coordination and communication among member nodes. Thus, the main schedulable resource becomes the wireless communication channel. Progress of user-level activities and their ability to meet end-end deadlines are therefore determined by scheduling of the communica-

tion medium rather than scheduling of the processor. Towards that end, new real-time communication architectures are required for ad hoc wireless environments.

Wireless communication scheduling differs from CPU scheduling in that it has an inherent notion of distance. In sensor networks, the distance is determined by the physical locations of source and destination. These locations impose distance constraints on messages, in addition to time constraints, calling for communication scheduling policies that are cognizant of both time and space.

The first contribution of this paper is RAP, a real-time communication architecture for large-scale wireless sensor networks. RAP provides a set of convenient, high-level real-time query and event services to real-time distributed micro-sensing applications. Query and event services are based on novel location-addressed communication models supported by a scalable and light-weight network stack.

RAP implements a novel *Velocity Monotonic Scheduling* (VMS) policy suitable for packet scheduling in sensor networks. VMS is based on a notion of packet *requested velocity*. Each packet is expected to make its end-to-end deadline if it can move toward the destination at its requested velocity, which reflects its local urgency. Compared with non-prioritized packet scheduling, VMS improves the deadline miss ratios of sensor networks by giving higher priority to packets with higher requested velocities. VMS can outperform deadline-based packet scheduling because velocity more accurately reflects the local urgency at each hop when packets with the same deadline have different distances to their destinations. Since the requested velocity can be locally determined assuming that each sensor knows its own location, a combination of VMS and geographic forwarding (GF) provides a localized and scalable protocol for real-time communication on sensor networks.

The final contribution of this paper is a detailed simulation study of the real-time performance of several routing protocols and packet scheduling algorithms in a typical sensor network scenario. Our simulation experiments demonstrate that, for sensors far away from their base station, a combination of GF and VMS reduces the deadline miss ratio by as much as 72.1%, compared to existing wireless communication, and by as much 28.1% compared to GF with deadline-based prioritization. To our best knowledge, ours is the first detailed performance study on deadline issues in multi-hop wireless sensor network settings under overload conditions.

In the following sections, we discuss the key characteristics of sensor networks, present the design of RAP, report a set of simulation experiments with sensor network configurations, and conclude the paper by summarizing our key results and future work.

2. Real-time Communication in Sensor Networks

In this section, we describe the characteristics of sensor networks and communication models on sensor networks. This analysis serves as a basis for our design of real-time communication protocols.

Sensor networks are an instance of mobile ad hoc networks (MANET) [9] that have recently attracted a lot of interest and visibility due to flexibility, the feasibility of their deployment at low costs, and the increasing number of potential application domains. In general, mobile ad hoc networks depend on peer-to-peer communication protocols that do not require a fixed infrastructure such as centralized servers and access points. Sensor-networks are different from their traditional ad hoc wireless counterparts in that they have a larger scale, higher density, smaller devices, and a tighter interaction with a physical environment. Energy conservation is important in wireless networks. It is especially critical in sensor networks because of form-factor constraints, which preclude the use of large batteries or power supplies, while expecting the sensor nodes to last for a long time.

In most envisioned sensor network applications, a large number of sensors are deployed in an area and a small number of more powerful nodes (such as PDAs) form possibly mobile interface stations which serve as the entry points to the sensor network. In the following, we shall call such interface stations, *base stations*. A user may query the physical environment through such base stations. Alternatively she may also register for an *event*. The occurrence of the event will automatically trigger a specified query. A query can specify timing requirements including rate, start time, duration, and end-to-end deadlines. For example, a user can register for a virus_found event in a rectangular area with coordinates (10,10,20,20), and specify a query on the event to report the density of the detected virus. If a virus is found, the density of the viruses should be reported to the base station from where they are found every 1.5 sec for a duration of 30 min. Every reading should reach the base station within an end-to-end deadline of 5 sec.

Communication in a sensor network can be divided into two categories: *local coordination* and *sensor-base*

communication. Before sending information to the base station, sensors within the local area often need to coordinate among themselves to aggregate data and generate a reliable result. For example, acoustic sensors may need triangulation among multiple nodes to decide the location of a tank. Local coordination often occurs within a distance of one or a few radio radii. Sensor-base communication is responsible for reporting the aggregated data to the base station, which often spans many (e.g., tens of) hops. Consider a communication radius of 30 m of common short-range radios transceivers. It is conceivable to have more than 10,000 nodes and tens of hops of communication in a coverage area of several square kilometers. Since sensor-base communication typically travels a much larger number of hops than local coordination messages, in this paper we focus on the former type of communication.

Unlike IP networks, sensor-base communication directly uses *location* as the target address. Instead of querying a sensor with ID 1002, a user or application queries a geographic region. The identities of sensors that happen to be located in that region are not important. Any sensors in that region that receive the query may initiate local coordination to aggregate the requested data. A leader may be elected to send the query result back to the base station. If continuous monitoring is required, the query may report the desired measurement periodically through the multi-hop ad hoc network. The base station can attach its location to the query message so that the query results can also be addressed by location (assuming no two base stations are at a same location).

The communication between sensors and the base stations is asymmetric since one single query from the base station often starts a long-running periodic data flow from the sensors to the base station.

Communication in sensor networks can suffer from "hot regions", i.e., areas where the network is seriously congested. Hot regions are often caused by a set of related events that synchronously trigger a large number of data flows toward the base station. Examples of related events include correlated measurement of the same environmental activity, or correlated environmental activities such as a group of new targets entering a security area, or a bio-attack on a part of an airport. Maximizing the number of packets that make their deadlines in overload conditions is critical in sensor networks.

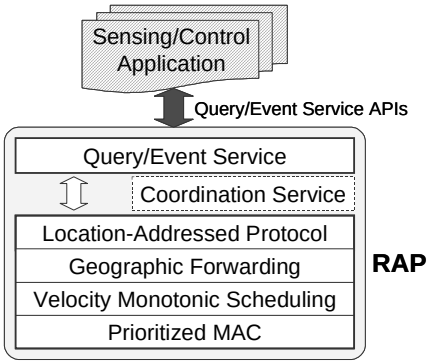


Figure 1 The RAP communication architecture

3. Design of RAP

We now present the design of RAP to support real-time communication in large-scale sensor networks. Given the unique characteristics of sensor networks, the goal of RAP includes the following:

- Provide general service APIs that are suitable for distributed micro-sensing and control in sensor networks
- Maximize the number of messages meeting their end-to-end deadlines
- Scale well with large number of nodes and hops
- Introduce minimum communication and processing overhead.

The architecture of RAP is shown in 0. Sensing and control applications interact with RAP through a set of query/event service APIs. A Query/Event Service layer submits the query or event registration to an area. The Query/Event Service at the sensors in that area then (periodically or aperiodically) sends query results back to the base station. If an event is registered, the query is started only if the registered event happens. The sensor-base communication is supported by a network stack including a transport-layer Location-Addressed Protocol (LAP), a Geographic Forwarding (GF) routing protocol, a Velocity Monotonic (packet) Scheduling (VMS) layer, and a prioritized MAC. This network stack embodies a set of efficient and localized algorithms to improve the end-to-end deadline miss ratio of sensor-base communication. This network stack is the focus of this paper.

The coordination service is responsible of dynamic group management and data aggregation among sensors (e.g., multiple sensors coordinate to determine the location of a target through triangulation). The coordination services are part of our on-going research and not addressed in this paper.

We now describe the Query/Event service APIs and the network protocol stack in detail in the following subsections.

3.1. Query/Event Service APIs

Applications may submit queries or register for events through a set of query/event service APIs. The API provides a high-level abstraction to applications by hiding the specific location and status of each individual node. These APIs allow applications to specify the timing constraints of queries. The underlying layers of RAP are responsible for orchestrating the sensing and communications of relevant sensors to accomplish all query and event services.

RAP provides the following query/event service APIs.

query{attribute_list, area, timing_constraints, querier_loc}

Issue a query for a list of attributes in an area. A query has timing constraints. If a period is specified for a command, query results will be automatically sent from an area to the issuer of the query in every period. For example, the following query requires the average density of the viruses in an rectangular area (10,10,12,12) be reported to the base station of the querier every 1.5 sec. Every reading should reach the base station within an end-to-end deadline of 5 sec. The query includes the location of its base station so that query results can be sent back using LAP. In this paper we assume the location of the base stations are fixed.

```
query {
    virus.count,
    area=(10,10,12,12),
    period=1.5,deadline=5,
    base=(100,100)
}
```

register_event{event, area, query}

Register for an event. A query is triggered once an event occurs. For example, the following API call registers a virus_count query for a virus_found event. If any viruses are found in a rectangular area with coordinates (10,10,100,100), returns the average density of the viruses of the 2×2 square area centered at the event location (X_{event},Y_{event}) every 1.5 sec. Every reading should reach the base station within an end-to-end deadline of 5 sec.

```
register_event {
    virusFound(0,0,100,100),
    query {
        virus.count,
        area=(Xevent-1,Yevent-1,Xevent+1,Yevent+1),
        period=1.5, deadline=5,
        base=(100,100)
    };
```

A query or event is sent to every node in the specified area. Query results are sent back to the base station based on its location provided by the query or event registration.

3.2. Location-Addressed Protocol

LAP is a connectionless transport layer in the network stack. LAP is similar to UDP except that all messages are addressed by location instead of IP address. Three types of communication are supported by LAP: unicast, area multicast, and area anycast.

- *Unicast* delivers a message to a node that is closest to the destination location. Unicast can be used by sensors to send query results to base stations.
- *Area multicast* delivers a message to *every* node in a specified area. Area multicast can be used to register for an event or send a query to an area, for coordination among nodes in a local group.
- *Area anycast* delivers a message to at least one node in a specified area. Area anycast can also be used for sending a query to a node in an area. The node can initiate group formation and coordination in that area.

Since this paper is concerned with real-time issues in overload conditions, in the rest of this paper we focus on unicast from sensors to base stations because this form of communication contributes to most of the real-time traffic in sensor networks.

3.3. Geographic Forwarding

Since communication destinations are identified by geographic location, we assume the routing layer is aware of physical geography. A router can determine the physical location of the destination relative to itself and forward the packet in the general direction of the destination. Geographic forwarding (GF) [16] has been proposed in earlier wireless literature and evaluated in the MANET environments.

More precisely, GF makes a greedy decision to forward a packet to a neighbor if 1) it has the shortest geographic distance to the packet's destination among all immediate neighbors; and 2) it is closer to the destination than the forwarding node. When such nodes do not exist in a region, the GPSR protocol [16] can be used to route packets around the perimeter of the depleted region. The only state on each node maintained by GF and GPSR is a table of the locations of immediate neighbors. Because GF uses immediate neighborhood

information to make localized routing decisions, it is highly scalable with regard to the number of nodes, network diameter, and the rate of change in topology [16]. GF works best in sensor networks that usually have high node densities and support location-addressed communication. Location addressed communication means that GF can be used without a location directory service [12], which could introduce extra management and communication overhead.

High node density causes two desirable properties of GF in sensor networks. First, the greedy forwarding algorithm described above has a high success probability in finding a good path from source to destination resulting in efficient communication. Second, the number of hops is approximately proportional to the distance that a packet has to travel. Hence, the distance between a node and a packet's destination can serve as an indication of the packet's hop count.

3.4. Velocity Monotonic Scheduling

A key component of real-time communication architectures is the packet scheduling policy which determines the order in which incoming packets at a router are forwarded to an outgoing link. In existing ad hoc networks, packets are typically forwarded in FCFS order. FCFS scheduling does not work well in real-time networks where packets have different end-to-end deadlines and distance constraints. Instead, competing packets should be prioritized based on their local urgency. In the context of sensor networks, packet scheduling should be both *deadline-aware* and *distance-aware*. Deadline-aware means that a packet's priority should relate to its deadline. The shorter the deadline, the higher the packet priority. Distance-aware means that a packet's priority should relate to its distance from the destination. The longer the distance, the higher the packet priority.

An example is shown in Figure 2. In scenario 1, both sensors A and B send periodic flows to a base station C. Packets from A and B compete at nodes D, E, and F because of possible collision of transmissions from B, F and D. They should also be prioritized in the network-layer queues in node E. Similarly, in scenario 2 flows from A and B will compete at nodes E, F, G, and H. Assume that both flows share a same deadline in each graph, then A's packets should have higher priorities than B's packets because A's packets have to travel farther than packets from B, and therefore should move faster in the competing regions.

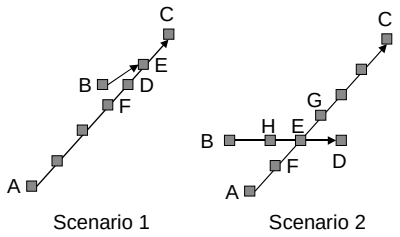


Figure 2 Scenarios of distance-aware scheduling

Since packet priority should be decided based on both distance and deadlines, we propose *Velocity Monotonic Scheduling* (VMS). VMS assigns the priority of a packet based on its *requested velocity*. A packet with a higher requested velocity is assigned a higher priority. VMS improves the number of packets that meet their deadlines because it assigns the "right" priorities to packets based on their different urgencies on the current hop. VMS also solves the fairness problem described in [18] in sensor networks because packets that are far away from the base station will be tend to have higher priorities when it competes against other packets that are closer to the destination.

We investigate two priority assignment policies: Static Velocity Monotonic (SVM) and Dynamic Velocity Monotonic (DVM), depending on whether requested velocity is computed dynamically or statically.

3.4.1. Static Velocity Monotonic

SVM computes a fixed requested velocity at the sender of each packet. Assume a packet is sent from a sender at (x_0,y_0) to a destination at (x_d,y_d), and has an end-to-end deadline D sec, then SVM sets its requested velocity to:

$$V = dis(x_0,y_0,x_d,y_d)/D \quad (1)$$

where $dis(x_0,y_0,x_d,y_d)$ is the geographic distance between (x_0,y_0) and (x_d,y_d). The requested velocity of a packet is fixed throughout the network.

3.4.2. Dynamic Velocity Monotonic

DVM *dynamically* re-calculates the requested velocity of a packet upon its arrival at each intermediate node. Assume a packet arrives at a node at location (x_i,y_i); its destination is at (x_d,y_d); it has an end-to-end deadline D sec, and its elapsed time, i.e., the time it has been in the network, is T_i sec; its requested velocity V_i at (x_i,y_i) is set to:

$$V_i = dis(x_i,y_i,x_d,y_d)/(D-T_i) \quad (2)$$

At the sender node (x_0,y_0), the elaped time $T_0=0$ and the requested velocity is initialized to $V=dis(x_0,y_0,x_d,y_d)/D$. The requested velocity of a packet will be adjusted based on its actual progress (i.e., actual velocity). A packet's requested velocity increases by re-applying Eq. 2 at subsequent nodes if its previous progress towards to the destination is slower (e.g., due to a hot region) than its previous requested velocity. On the other hand, its requested velocity decreases if it moves faster than its previous requested velocity. This is so this packet can give way to other more urgent packets. Note that DMV requires clock synchronization.

3.4.3. Priority Queue

Each packet is assigned a priority based on its requested velocity and queued at the network layer when there are multiple outstanding packets. Several options are available for implementing priority queues. One approach is to insert all packets into a single queue ordered by priority. When the queue us full, higher priority incoming packets overwrite lower priority ones. The benefit of this solution is that it most accurately reflects the order of requested velocities, and allows all packets to share the same buffer regardless of their priority. The approach, however, requires implementing a data structure whose insertion time, in the worst case, grows logarithmically in the number of packets. To bound the queue insertion overhead, another approach currently used in our simulation is to maintain multiple FIFO queues each corresponding to a fixed priority level. Each priority corresponds to a range of requested velocities. A packet is first mapped to a priority, and then inserted into the FIFO queue that corresponds to its priority. This approach is more efficient because no ordering needs to be performed for every incoming packet. The per-packet overhead is logarithmic only in the number of priority levels, not the number packets. Assuming that packets that miss their deadlines are useless, priority queues can actively drop packets that have missed their deadlines to avoid wasting bandwidth.

3.5. MAC-layer prioritization

Local prioritization at each individual node is not sufficient in wireless networks because packets from different senders can compete against each other for a shared radio communication channel. To enforce packet priorities, MAC protocols should provide *distributed prioritization* on packets from different competing nodes. Extensions [1][15] of the IEEE 802.11 wireless LAN protocol [18] have been developed to provide distributed prioritization. In this paper we implement two of such extensions proposed by Aad and Castelluccia [1]. We modified two components of the standard 802.11 implementation: the initial wait time after the channel becomes idle, and the backoff window increase function. These mechanisms are chosen because they introduce minimal overhead and can be ported to light-weight CSMA/CA protocols [23] that are more suitable to sensor networks than 802.11. We now briefly describe the mechanisms. The detailed description and analysis of these mechanisms are available in [1].

3.5.1. Initial Wait Time after Idle

802.11 sets a DIFS counter once the communication channel has become idle. Before sending an RTS (Request To Send) packet, a node will wait a random period of time between 0 and DIFS. To prioritize this process we set the DIFS parameter based on the packet priority: DIFS = BASE_DIFS * PRIORITY. Packets with a higher priority (corresponding to a low PRIORITY value) on average choose a smaller waiting period than another packet with a lower priority.

3.5.2. Backoff Increase Function

802.11 doubles its backoff window, CW, to extend a node's waiting period when a transmission collision occurs. We modified 802.11 to increase CW in accordance with the packet priority[1]:

CW=CW*(2+(PRIORITY-1)/MAX_PRIORITY)

MAX_PRIORITY is the maximum value of priority (corresponding to the lowest priority). The backoff counter of a node with a pending lower priority packet increases faster than a node with a pending packet with a higher priority.

The above two mechanisms give high priority packets high probability to get the channel in both the contention avoidance and contention phases.

In summary, RAP integrates a set of light-weight protocols to satisfy the following key requirements of large-scale sensor networks.

- RAP provides general query and event service APIs as a convenient high-level service abstraction suited for distributed micro-sensing applications.
- RAP increases the number of packets meeting their end-to-end deadlines by prioritizing the

[1] The backoff function is slightly changed from the original extension to mitigate its stability problem observed in [1].

transmission of contending packets based on their requested velocities.

- RAP scales well in large-scale sensor networks because it is composed of efficient and localized protocols and algorithms at every layer. The only states GF maintains are the locations of immediate neighbors. VMS determines a packet's priority only based on locally available information. No connection state is maintained inside the network.

4. Experimentation

We ran a set of simulation experiments to evaluate the aforementioned real-time packet scheduling and prioritization protocols on sensor networks for a biometric sensing application. We implemented GF, VMS, and the 802.11 extensions on the GloMoSim wireless network simulator [4] developed by UCLA.

4.1. Network configuration

We tuned the network parameters to approximately simulate the Berkeley motes [8], a state-of-the art network sensors. We generated a square region of 136×136 m^2 divided into 100 13.6×13.6 m^2 grids. 100 nodes were simulated with one node randomly placed in each grid.

The other network parameters are listed as follows:

- Radio communication radius: 30.5 m
- Packet size: 32 bytes for each count packet, 160 bytes for each detail packet
- Bandwidth: 200 kbps. Current version of MICA motes available to us supports a bandwidth of 100kbps. Future versions are expected to have a much higher capacity. Due to limitations of the GloMoSim simulator, we had to send data flows on top of the UDP/IP stack that contribute to 28 B overhead. In a real implementation we expect to eliminate the UDP/IP headers.

4.2. Application Workloads

We simulate a bio-sensor application that monitors viruses in an area. Users can register for events and query bio-sensors, which generate periodic data flows to a base station. Data flows have different rate and timing requirements. We assume that a base-station sends two different queries: count and detail to various locations.

Count:

```
registerEvent {
    virusFound(0,0,136,136),
    query {
        virus.count,
        area=(Xevent-1,Yevent-1,Xevent+1,Yevent+1),
        period=Pc, deadline=Dc
        base = (134.07, 128.06)
    };
};
```

Detail:

```
query {
    detail,
    area=(x-1,y-1,x+1,y+1),
    period=Pd, deadline=Dd
    base=(134.07, 128.06)
};
```

A user registers a *count* query with a virusFound event in the whole 136×136 m^2 squared area. A virusFound event is generated when a grid detects a specified virus at location (X_{event}-1,Y_{event}). This event triggers a query virus_count, which periodically reports the density of the detected virus in the area (X_{event}-1,Y_{event}-1,X_{event}+1,Y_{event}+1) to the base station.

A user can also directly submit a *detail* query to get more detailed data collected at a location. On the other hand *detail* generates periodic flows that send detailed information about a grid to the base station for further analysis. While a large number of *count* flows may be generated (e.g., during a bio-attack), the user may only query the *details* of a small number of important grids. We assume that packets (called *count packets*) returned by *count* queries have longer deadlines and a smaller size than packets (called *detail packets*) returned by *detail* queries.

We simulate a scenario that correlated events (i.e., a bio-attack) result in two hot regions each covering approximately a square of 54.4×54.4 m^2. A hot region locates at the southwest corner. The other hot region is close to the center of the region. The two hot regions are on a same diagonal to the base station to generate a worst-case congestion situation. Each hot region generates multiple flows to a base station on the northeast corner of the region. In addition, a small number of other flows are generated from other randomly picked locations. A total of 31 nodes send CBR flows representing count flows, with a subset of 15 of these nodes also sending CBR flows representing detail flows. All flows are started with a uniformly randomized time within a window of 5 s to simulate synchronous events common in sensor networks.

We varied the rates and deadlines between the count and detail flows to better understand the effect of these parameters on different protocols. The table below lists the configurations that we tested in our simulations.

rate (1/s) count : detail	deadline (s) count : detail
0.67 : 0.30	50 : 5
0.76 : 0.35	25 : 5
0.80 : 0.36	10 : 5

The rates and number of flows were chosen such that the network is close to its breaking point where packets start to miss their deadlines.

4.3. Implementation of Protocols

Before we investigate packet scheduling algorithms, an important design decision in RAP is the routing protocol. Our investigations focus on two routing protocols, Dynamic Source Routing (DSR) [11] and GF [16]. DSR is an ad hoc routing protocol designed for traditional ID-based MANET. Previous performance studies [5] showed that DSR outperforms other major ID-based routing protocols in term of packet delivery ratio. GF is a location-based routing protocol suitable for location-addressed communication. While DSR and GF are not new, they have not been previously studied in term of deadline miss ratio in sensor networks. We compare their deadline miss ratios in Section 4.4.

At the packet scheduling layer we compare SVM and DVM against two baselines: FCFS and a deadline-based scheduling algorithm that we call DS. DS assigns a fixed priority to packets based on their end-to-end deadlines. In our workload, all count packets are assigned priority 3 (the lowest), and all detail packets are assigned priority 1 (the highest). At the MAC layer, 802.11 and its extensions were used in combination with other protocols. We now list the combination of protocols in the following table. The first column contains the acronyms that are used to represent the combinations in the same row.

	Routing	Scheduling	MAC
DSR/FCFS	DSR	FCFS	802.11
GF/FCFS	GF	FCFS	802.11
GF/DS	GF	DS	802.11 extension
GF/SVM	GF	SVM	802.11 extension
GF/DVM	GF	DVM	802.11 extension

DS, SVM, and DVM actively dropped packets that already miss their deadlines, while DSR and GF did not actively drop packets to be consistent with original specifications. Only greedy forwarding is implemented for GF. We did not implemented GPSR. The beacon period of GF is 5 sec.

The network-layer queues can hold a total of 300 packets for each configuration. DSR and GF had a single FIFO queue with 300 entries, while DS, SVM, and DVM each had three FIFO queues corresponding to different priorities. The mapping from a velocity to a priority is shown in the following table. The velocity ranges in SVM are chosen to balance the number of flows in each priority level. The velocity ranges in DVM initially assigned priority 2 or 3 to all flows. This allowed raising some packets' priorities to priority 1 if they are delayed.

Priority	Velocity Range (m/s)	
	SVM	DVM
1	(10, ∞)	(40, ∞)
2	(5, 10]	(10, 40]
3	(0, 5]	(0, 10]

Six repeated runs were made for each of the nine combinations of the rates and deadlines. The main performance metric is the deadline *miss ratio*, i.e., the percentage of generated packets that are received by the base station within their deadlines. The mean of six runs and its 90% confidence intervals for each data point for deadline (5:10) sec are shown in all data points of miss ratios.

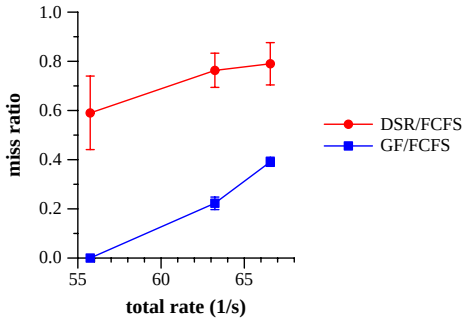


Figure 3 Overall deadline miss ratio of DSR and GF with deadlines (5,10)

4.4. Routing: DSR and GF

First we compare the overall deadline miss ratio of DSR/FCFS and GF/FCFS (see Figure 3). DSR has a significantly higher miss ratio than GF. We found that DSR lost a large number of packets due to queue overflow, while GF lost no packets due to queue overflow. The high percentage of packet drop in DSR is caused by its aggressive route-caching. In DSR, each node caches overheard routes. When a node receives a route discovery packet from another node, it checks its route cache and informs the sender of the requested route if it is available in the route cache. In the hot regions in our network, only the first flow needs to flood the network to acquire a route to the base station. All the later flows from the same region will be informed of the existing route causing most packets from the hot region to go through the *same* route (close to the diagonal line). In overload conditions, nodes on the shared route ran out of queuing space and lost packets. This problem can be common in sensor networks because of their correlated traffic patterns. In sensor networks related events (i.e., a bio-attack) can start a large number of flows from a

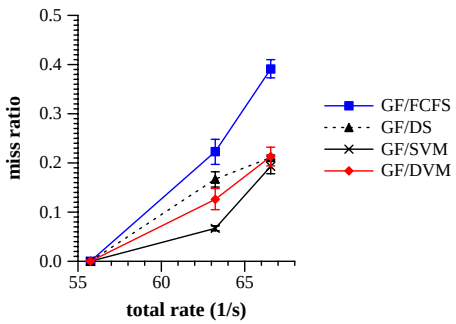
same small region almost synchronously. GF does not have the overflow problem because it delivers a packet through a straight line from its source to the base station. Packets from different sensors are routed through different nodes because the source sensors have different directions toward the base station. This an important reason that GF is more suitable than DSR in sensor networks.

4.5. Packet scheduling

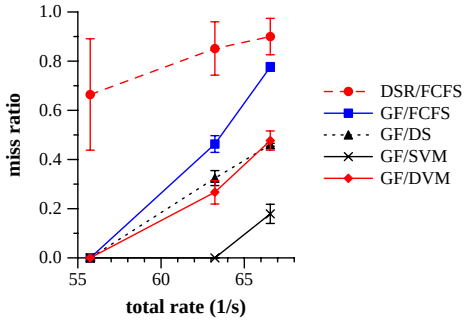
Now we compare different packet scheduling algorithms. The overall deadline miss ratios for deadlines of (5,10) s are presented in Figure 4a. The overall deadline miss ratio of DSR/FCFS is not shown in this figure because it is significantly higher than all other protocols and cannot fit in the scale (the maximum miss ratio of 0.5) of the graph. Since packets close to the base station are more likely to meet their deadlines and tend to "dilute" the difference between different algorithms, we also present the miss ratio of the subset of flows from the farther hot region at the southwest corner of Figure 4b.

From both figures, all prioritization-based packet scheduling (DS, SVM, and DVM) achieved miss ratios that were significantly lower than the protocols using FCFS. In particular GF/SVM achieved a significantly lower deadline miss ratio than all other protocols. As shown in Figure 4b, when the highest overall rate is 66.6 packets/s, only 17.9±3.9% of all packets from the farther hot region missed their deadlines with SVM, compared with a miss ratio of 77.6±1.7% for GF (with FCFS), 46.0±0.6% for DS. This result demonstrates SVM's advantage of considering both distance and deadlines in packet prioritization.

However, DVM did not outperform DS. We suspect that this is due to our current implementation that has only three priority levels, which adversely affects the flexibility of priority adjustment in DVM. We plan to investigate other implementations of DVM (such as a single packet queue ordered velocity) that allows better granularity of control by DVM.



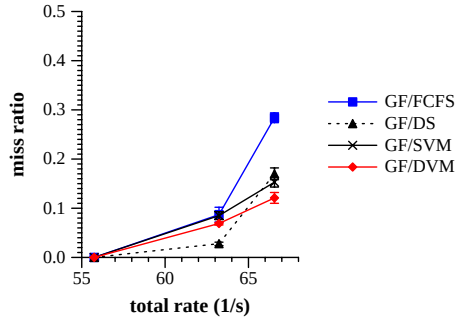
(a) Overall Deadline Miss Ratios



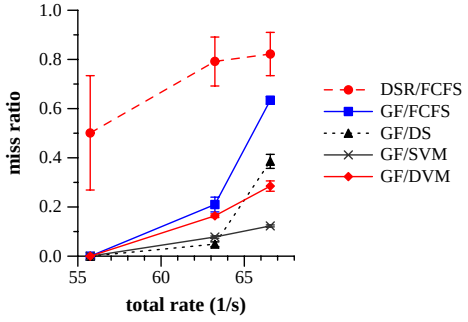
(b) Deadline Miss Ratios of Flows from the far corner

Figure 4 Deadline miss ratio with deadlines (5,10)

The miss ratios of all flows and the flows from the remote hot regions with deadlines (5, 25) s and (5, 50) s are presented in Figure 5ab and Figure 6ab, respectively. All velocity-based and deadline-based packet scheduling still significantly outperform GF and DSR with FCFS scheduling. Moreover, SVM consistently achieves the lowest miss ratio in both cases. The difference between SVM and DVM decreases as the difference between deadlines of the two types of flows is increased. DS and SVM perform almost identically when deadline is (5, 50) s. Since the distances from each sensor to the base station stays the same, deadlines become a more dominant factor in requested velocity with significantly different deadlines.

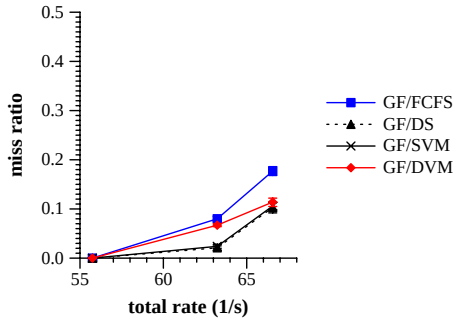


(a) Overall Deadline Miss Ratios

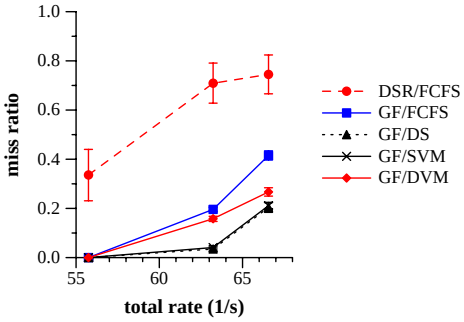


(b) Deadline Miss Ratios of Flows from the far corner

Figure 5 Deadline miss ratio with deadlines (5,25)



(a) Overall Deadline Miss Ratios



(b) Deadline Miss Ratios of Flows from the far corner

Figure 6 Deadline miss ratio with deadlines (5,50)

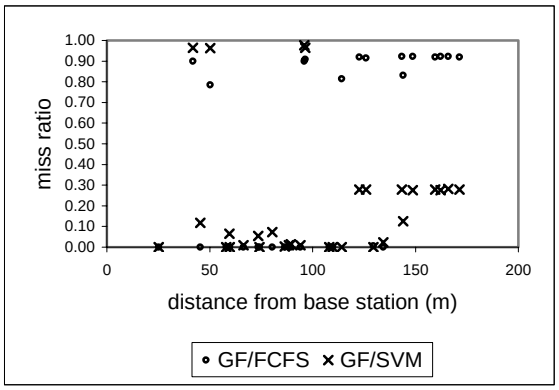


Figure 7 Miss ratio vs distance between source and destination (Deadline: (5:10) s; Rates: (0.8, 0.36)/s)

4.5.1. Distance fairness

SVM achieves better *fairness* to flows from sensors that are far away from the base station. This form of fairness is important to sensor networks because it affects how well sensor networks can scale. A sensor network cannot provide sufficient service if all remote sensors cannot report to the base station in time! We show the fairness by plotting the miss ratio of packets as a function of its sender's distance from the base station in a typical run with the highest rate in Figure 7. GF (with FCFS) significantly discriminates against remote sensors. Almost all packets that are from sensors more than 120 m away from the base station miss their deadlines. In contrast, SVM reduces the deadline miss ratio of remote sensors to about 30%. SVM is also fairer than DS and DVM that both achieved a miss ratio of about 60% for those packets (not shown due to space limitations).

In summary, SVM consistently achieves lower deadline miss ratios than both FCFS and the deadline based scheduling policy in all experiments. The performance improvement of SVM is especially significant for data flows generated from far away from their base station. For these flows SVM reduces the deadline miss ratio by as much as 57.6%, compared to FCFS, and by as much 28.1% compared to deadline-based scheduling policy.

5. Related Work

There are significant research results on real-time communications on single-hop wired LANs (e.g., [24][25]), multi-hop wired LANs (e.g., [14]), ATM (e.g., [17][18][19]), and the Internet (e.g., [6][13][22][21]). A good survey about real-time network architecture for packet-switched network is [3]. However, there have been few published works on real-time multi-hop sensor networks, which has significant different constraints from previous real-time networks.

Directed diffusion [10] is a data-driven communication paradigm for sensor networks. Users can broadcast interests to sensor networks. Sensors whose data match an interest report their data to the node that posts the interest. Our event service is similar to the interests in directed diffusion. The difference is that RAP allows users to specify the deadlines of queries on events. RAP's network protocol stack priorities the transmission of packets based on their requested velocities. In contrast, directed diffusion treats all data the same without considering their deadlines. Directed diffusion does not support location-addressed communication.

There has been significant research on routing protocols targeted at traditional MANET systems with a smaller scale than sensor networks. Broch et. al. [5] presented detailed simulation results of four representative routing protocols including DSDV [1], TORA [1], DSR [11], and AODV [21] in small MANET with radio communication similar to wireless LAN cards. Their results showed that reactive routing protocols introduces less overhead packets and achieve higher data throughput. DSR and AODV need to flood the

network to establish a route. This may introduce high overhead for large-scale sensor networks. Flooding can be partly avoided through aggressive caching of overheard routes on each node, but it can cause the queue overflow problem as described in Section 0. DSR writes the IDs of every node on the route to the packet header, which can cause significant overhead in sensor networks with many hops. Karp and Kung [16] presented geographic forwarding protocols and demonstrated that they scale better than DSR in term of network diameters and moving speed.

At the MAC layer Woo and Culler [23] proposed a MAC protocol with adaptive rate control to achieve fairness among nodes regardless of their distance from the base station in a sensor network. However, their MAC does not provide prioritization for packets with different deadlines. Timing constraints are not considered in their protocol.

Several prioritization and real-time architectures of wireless LANs have been proposed in the literature. In [2] Adamou et. al. presented a fair scheduling algorithm on Wireless LAN. Choi and Shin [7] proposed a Time-Division Duplexed LAN architecture for both real-time and non-real time communication. These solutions are not designed for multi-hop networks. Kanodia et. al. [15] proposed a MAC-layer prioritization mechanism for 802.11. Their solution depends on the RTS/CTS mechanism and requires all nodes to overhear to RTS/CTS even when they are not sending or receiving data. The overhearing requirement prevents nodes from sleeping, which can be vital for improving the power efficiency in sensor networks [23].

Our work on VMS is inspired by coordinated multi-hop scheduling [15] developed by Kanodia et. al. They proposed three priority index assignment policies for multi-hop wireless networks. The Time-To-Live (TTL) policy assigns priority to a packet based on its TTL counter, while each node decreases TTL by the time it spent in that node. The TTL-based priority can dynamically adapt packet priorities based on its progress. We note that TTL-based priority may not handle scenario 2 in Figure 2 well because A and B's packets may have a similar TTL despite the fact that they have different distances to travel after E. The fixed per-node allocation decreases the priority index on each node by a per-node constant. The uniform delay budget (UDB) allocation assigns a fixed priority index to a packet based on its end-to-end deadline divided by the end-to-end hop count. UDB utilizes *per-hop* velocity computed based on end-to-end hop count, while VMS is based on *geographic* velocity computed based on the geographic distance to the destination. UDB requires underlying routing protocols to provide the end-to-end hop count for each flow, which is obtained at the cost of route discovery and maintenance overhead. UDB cannot work with GF, which does not provide hop count. In comparison, VMS does not require hop count and is a perfect match with GF.

6. Conclusions

Real-time communication is a critical service for future sensor networks to provide distributed micro-sensing in volatile environments. We present RAP, a new real-time communication architecture for large-scale sensor networks. RAP provides convenient, high-level query and event services for distributed micro-sensing applications. Novel location-addressed communication models are supported by a scalable and light-weight network stack. We exploit the notion of velocity in real-time communication protocols on sensor networks. Velocity reflects the local urgency of a packet by capturing both key constraints in sensor networks, namely, the end-to-end deadline and the communication distance. We present Velocity-Monotonic Scheduling as a suitable scheduling policy to minimize deadline miss ratios in multi-hop sensor networks. Detailed simulations of sensor network environments demonstrate that RAP significantly reduces both the end-to-end deadline miss ratio in the sensor network. In the future we will investigate the schedulability analysis and admission control algorithms for VMS in order to provide deadline guarantees. We will also develop coordination protocols in sensor networks and implement RAP in a physical testbed of Berkeley motes [8].

Acknowledgements

We thank Sang Son and Chengzhi Li for their helpful discussions on this work. The work reported in this paper was supported, in part, by the DARPA NEST program under grant F33615-01-C-1905, and the DoD MURI program administered by the ONR under Grant N00014-01-1-0576.

7. References

[1] I. Aad and C. Castelluccia, "Differentiation Mechanisms for IEEE 802.11," *IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001.

[2] M. Adamou, S. Khanna, I. Lee, I. Shin, S. Zhou, "Fair Real-time Traffic Scheduling over A Wireless LAN," *Proceedings of the 22nd IEEE Real-Time Systems Symposium*, RTSS 2001, London, UK, December 3-6, 2001

[3] Aras, Kurose, Reeves, Schulzrinne. "Real-Time Com-

munication in Packet-Switched Networks", Proc. of the IEEE, Vol. 82 No. 1, Jan. 1994, pp. 122--139.

[4] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla. "GloMoSim: A Scalable Network Simulation Environment," *UCLA Computer Science Department Technical Report 990027*, May 1999.

[5] J. Broch, D. A. Maltz, D. B. Johnson, Y. Hu, and J. Jetcheva. "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols." In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 1998)*, ACM, Dallas, TX, October 1998.

[6] B. Choi, D. Xuan, C. Li, R. Bettati, and W. Zhao, "Scalable QoS Guaranteed Communication Services for Real-Time Applications," in *Proc. of the IEEE International Conf. on Distributed Computing Systems*, (ICDCS), April 2000.

[7] S. Choi and K. G. Shin, "A Unified Wireless LAN Architecture for Real-Time and Non-Real-Time Communication Services," IEEE/ACM Transactions on Networking, 8(1), February 2000.

[8] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, " System Architecture Directions for Network Sensors," ASPLOS 2000.

[9] IETF Working Group on Mobile Ad Hoc Networks, http://www.ietf.org/html.charters/manet-charter.html.

[10] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *the Sixth Annual International Conference on Mobile Computing and Networks (MobiCOM 2000)*, August 2000, Boston, Massachusetts.

[11] D. B. Johnson and D. A. Maltz. "Dynamic Source Routing in Ad Hoc Wireless Networks." In *Mobile Computing*, edited by Tomasz Imielinski and Hank Korth, Chapter 5, pages 153-181, Kluwer Academic Publishers, 1996.

[12] J. Li, J. Jannotti, D, S. J. De Couto, D. R. Karger, and R. Morris, "A Scalable Location Service for Geographic Ad Hoc Routing," in the *Proceedings of the 6th ACM International Conference on Mobile Computing and Networking (MobiCom '00)*, Boston, Massachusetts, August 2000, pages 120-130.

[13] J. Liebeherr, D. E. Wrege, and D. Ferrari, "Exact Admission Control in Networks with Bounded Delay Services," IEEE/ACM Transactions on Networking, 1996.

[14] D. Kandlur, K. G. Shin, and D. Ferrari, "Real-time Communication in Multi-hop Networks," *IEEE Trans. on Parallel and Distributed Systems*, October 1994, pp. 1044-1056.

[15] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. W. Knightly, "Distributed Multi-Hop Scheduling and Medium Access with Delay and Throughput Constraints," *International Conference on Mobile Computing and Networking (MobiCom 2001)*, Rome, Italy, July 16-21, 2001.

[16] Karp, B. and Kung, H.T., "Greedy Perimeter Stateless Routing for Wireless Networks," in *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, Boston, MA, August, 2000, pp. 243-254.

[17] S. Kweon and K. G. Shin, "Providing Deterministic Delay Guarantees in ATM Networks," *IEEE/ACM Transactions on Networking*, 6(6), December 1998.

[18] C. Li, R. Bettati, and W. Zhao, "Static Priority Scheduling for ATM Networks," in *Proc. of IEEE Real-Time Systems Symposium*, Dec. 1997.

[19] J. K. Ng, S. Song, and W. Zhao, "Integrated Delay Analysis of Regulated ATM," in *Proc. of IEEE Real-Time Systems Symposium*, Dec. 1997.

[20] B. O'Hara and A. Petrick, *IEEE 802.11 Handbook - a Designer's Companion*, IEEE Press, 1999.

[21] I. Stoica and H. Zhang, "Providing Guaranteed Services Without Per Flow Management," *SIGCOMM*, 1999.

[22] S. Wang, D. Xuan, R. Bettati, and W. Zhao, "Providing Absolute Differentiated Services for Real-Time Applications in Static-Priority Scheduling Networks," *IEEE INFOCOM 2001*.

[23] Woo and D. Culler. "A Transmission Control Scheme for Media Access in Sensor Networks," *ACM/IEEE International Conference on Mobile Computing and Networks (MobiCOM 2001)*, Rome, Italy, July 2001.

[24] W. Zhao , J. A. Stankovic , K. Ramamritham, "A Window Protocol for Transmission of Time-Constrained Messages," *IEEE Transactions on Computers*, 39(9), p.1186-1203, Sept. 1990.

[25] K. M. Zuberi, K. G. Shin, "Design and Implementation of Efficient Message Scheduling for Controller Area Network," 49(2), *IEEE Transactions on Computers* February 2000.