Report Number: WUCSE-2005-31

2005-07-06

Real-time Power Aware Routing in Wireless Sensor Networks

Octav Chipara, Zhimin He, Guoliang Xing, Qin Chen, Xiaorui Wang, Chenyang Lu, John Stankovic, and Tarek Abdelzaher

Many mission-critical wireless sensor network applications must resolve the inherent conflict between the tight resource constraints on each sensor node, particularly in terms of energy, with the need to achieve desired quality of service such as end-to-end real-time performance. To address this challenge we propose the Real-time Power-Aware Routing (RPAR) protocol. RPAR achieves required communication delays at minimum energy cost by dynamically adapting the transmission power and routing decisions based on packet deadlines. RPAR integrates a geographic forwarding policy cognizant of deadlines, power, and link quality with new algorithms for on-demand power adaptation and efficient neighborhood discovery. Simulations based... **Read complete abstract on page 2.**

Recommended Citation

Chipara, Octav; He, Zhimin; Xing, Guoliang; Chen, Qin; Wang, Xiaorui; Lu, Chenyang; Stankovic, John; and Abdelzaher, Tarek, "Real-time Power Aware Routing in Wireless Sensor Networks" Report Number: WUCSE-2005-31 (2005). *All Computer Science and Engineering Research.*
https://openscholarship.wustl.edu/cse_research/950

Real-time Power Aware Routing in Wireless Sensor Networks

Octav Chipara, Zhimin He, Guoliang Xing, Qin Chen, Xiaorui Wang, Chenyang Lu, John Stankovic, and Tarek Abdelzaher

Complete Abstract:

Many mission-critical wireless sensor network applications must resolve the inherent conflict between the tight resource constraints on each sensor node, particularly in terms of energy, with the need to achieve desired quality of service such as end-to-end real-time performance. To address this challenge we propose the Real-time Power-Aware Routing (RPAR) protocol. RPAR achieves required communication delays at minimum energy cost by dynamically adapting the transmission power and routing decisions based on packet deadlines. RPAR integrates a geographic forwarding policy cognizant of deadlines, power, and link quality with new algorithms for on-demand power adaptation and efficient neighborhood discovery. Simulations based on a realistic radio model of MICA2 motes show that RPAR significantly reduces the number of deadline misses and energy consumption when compared to existing real-time and energy-efficient routing protocols and beacon based neighborhood management schemes.

2005-31

Real-time Power Aware Routing in Wireless Sensor Networks

Authors: Chipara, Octav; He, Zhimin; Xing, Guoliang; Chen, Qin; Wang, Xiaorui; Lu, Chenyang; Stankovic , John; Abdelzaher, Tarek

July 6, 2005

Abstract: Many mission-critical wireless sensor network applications must resolve the inherent conflict between the tight resource constraints on each sensor node, particularly in terms of energy, with the need to achieve desired quality of service such as end-to-end real-time performance. To address this challenge we propose the Real-time Power-Aware Routing (RPAR) protocol. RPAR achieves required communication delays at minimum energy cost by dynamically adapting the transmission power and routing decisions based on packet deadlines. RPAR integrates a geographic forwarding policy cognizant of deadlines, power, and link quality with new algorithms for on-demand power adaptation and efficient neighborhood discovery. Simulations based on a realistic radio model of MICA2 motes show that RPAR significantly reduces the number of deadline misses and energy consumption when compared to existing real-time and energy-efficient routing protocols and beacon based neighborhood management schemes.

Real-time Power Aware Routing in Wireless Sensor Networks

[†]Octav Chipara, [‡] Zhimin He, [†]Guoliang Xing, [‡]Qin Chen, [†]Xiaorui Wang, [†]Chenyang Lu, [‡]John Stankovic and [*]Tarek Abdelzaher
[†] Washington University in St. Louis [‡] University of Virginia [*]University of Illinois at Urbana-Champaign

Abstract— **Many mission-critical wireless sensor network applications must resolve the inherent conflict between the tight resource constraints on each sensor node, particularly in terms of energy, with the need to achieve desired quality of service such as end-to-end real-time performance. To address this challenge we propose the *Real-time Power-Aware Routing* (RPAR) protocol. RPAR achieves required communication delays at minimum energy cost by dynamically adapting the transmission power and routing decisions based on packet deadlines. RPAR integrates a geographic forwarding policy cognizant of deadlines, power, and link quality with new algorithms for on-demand power adaptation and efficient neighborhood discovery. Simulations based on a realistic radio model of MICA2 motes show that RPAR significantly reduces the number of deadline misses and energy consumption when compared to existing real-time and energy-efficient routing protocols and beacon based neighborhood management schemes.**

I. Introduction

Real-time interaction with physical environments is crucial for many wireless sensor network (WSN) applications. For example, a surveillance system must notify users within a few seconds after an intruder is detected to start pursuing actions in time. Similarly, a fire fighter may rely on timely temperature updates to remain alert to the surrounding fire conditions. Late delivery of sensor data may endanger the fire fighter. Moreover, different data in a system may have different deadlines. For instance, the validity intervals (and hence, update deadlines) of the locations of different intruders such as pedestrians and motor vehicles may depend on their velocities. To support such applications, the underlying communication protocols must adapt their behavior based on packet deadlines to reduce the number of deadline misses.

Supporting real-time communication in WSNs is extremely challenging. Low-power wireless networks usually have unreliable links and limited bandwidth. Furthermore, link quality can be heavily influenced by environmental factors [1][2]. As a result, communication delays in WSNs are highly unpredictable. Moreover, many WSN applications must operate for months or years without wired power supplies. Real-time communication protocols designed for WSNs must therefore balance real-time performance with energy efficiency.

To address the above challenges, we propose the *Real-time Power-Aware Routing (RPAR)* protocol to provide energy-efficient soft real-time communication in WSNs. RPAR achieves required communication delays by dynamically adapting the transmission power and routing decisions based on packet deadlines and network conditions. RPAR has several novel features. First, it employs a new forwarding policy that identifies the most energy efficient forwarding choice (combination of neighbor and transmission power) that meets the desired latency. This policy enables RPAR to meet required packet deadlines at minimum energy cost. Second, a key component of RPAR is a novel on-demand neighborhood manager. Our approach includes power adaptation and neighbor discovery schemes that can quickly identify forwarding choices that meet packet deadlines while introducing minimum overhead. Moreover, RPAR addresses a range of important practical issues in WSNs including probabilistic links, scalability, and severe memory and bandwidth constraints.

The rest of the paper is organized as follows. We first analyze the impact of transmission power on communication latency via an empirical study on XSM2 motes (Section II). We identify the design goals for real-time power-aware routing (Section III). Next, we present the design of the routing algorithm (Section IV) and its neighborhood manager (Section V). We evaluate the performance of RPAR through simulations based on a realistic radio model of MICA2 motes (Section VI). We conclude the paper after discussions on related work (Section VIII) and open issues (Section VII).

II. Impact of Transmission Power Control

In this section we study the impact of transmission control on real-time performance in WSNs. We first measure the relationship between transmission power and communication delay under a light load on XMS2 motes. We then discuss the tradeoff between communication delay and channel capacity when the workload causes contention and interference.

A. Delay Considerations

To understand the impact of power control on the end-to-end delay we performed a set of experiments in an office environment using XSM2 motes[1]. Five XSM2 motes are placed in a line. The first mote injects a packet into the network at a rate of 4 packets per second. Each mote forwards the packet to its next neighbor. When a packet reaches the end of the line, the last mote changes the packet's direction and sends it back to the source. Each mote runs B-MAC [3]. The automatic retry request (ARQ) is used to retransmit a packet at most five times to improve reliability. The transmission power is varied from -18dbm to 0dbm in increments of 1dbm. The one-hop distance is varied from 5 feet to 40 feet, in increments of 5 feet. One hundred packets are sent at each power level.

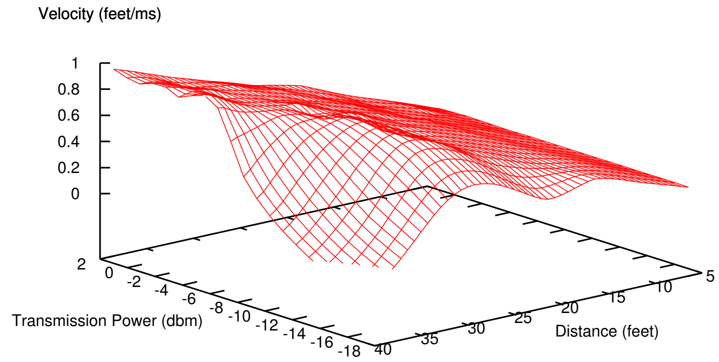


Fig. 1. Impact of transmission power and one-hop distance on delivery velocity

To evaluate the impact of transmission power and one-hop distance on end-to-end delay, we measure the *delivery velocity* of each packet. The delivery velocity is defined as the total distance the packet travels divided by its end-to-end delay. As shown in Figure 1, transmission power has a significant impact on delivery velocity. For example, when the one-hop distance is 20 feet, increasing the transmission power results in more than a two-fold improvement in delivery velocity from 0.25 feet/ms at -18dbm to 0.54 feet/ms at 2dbm. This is because increasing transmission power can effectively improve the quality of a wireless link [4] and, hence, reduce the number of transmissions needed to deliver

[1] Both XSM2 and MICA2 uses the ChipCon CC1000 radio.

a packet. In addition, the delivery velocity increases as the one-hop distance increases within a range, but drops sharply when the one-hop distance exceeds the range due to degrading link quality[2]. Higher transmission power achieves a longer drop-off range. Therefore, far-away nodes that have poor quality links at low power, are transformed into reliable communication neighbors when the transmission power is increased. Such nodes achieve high delivery velocities.

Our experiments demonstrate that transmission power control is a highly effective mechanism for controlling communication delay under light workload by controlling the transmission power and the one-hop distance.

B. *Capacity Considerations*

Increasing transmission power has the side effect of reducing the maximum achievable throughput of a WSN due to increased channel contention and interference [6]. We focus on real-time applications in which meeting the deadlines of real-time packets is more important than total throughput. For example, in a surveillance application timely delivery of the location of a intruder is more important to the user than delivering a large amount of non-critical data such as room temperatures. RPAR is designed to dynamically achieve the desired tradeoff between communication delay and throughput by adapting the transmission power based on required communication delays. When deadlines are tight, RPAR trades capacity and energy for shorter communication delay by increasing the transmission power. On the other hand, when deadlines are loose, RPAR lowers the transmission power to increase throughput and reduce energy consumption. This adaptive approach is a key feature of RPAR.

It is also important to note that the reduced capacity is a problem only when the workload approaches the network capacity. Recent advances in real-time capacity theory has shown that the performance degradation can be avoided as long as the amount of high-priority data transmitted in the network is small enough not to trigger capacity bottlenecks. In [7] the authors derive a lower bound on the maximum amount of real-time traffic that may be transmitted without triggering the capacity bottlenecks. This bound may be used to perform off-line analysis of the capacity requirement or on-line for admission control or congestion control. We discuss how to integrate RPAR with such techniques in Section VII-A.

III. Problem Formulation

The primary goal of RPAR is to increase the number of packets that meet their deadlines and, hence, reduce the miss ratio[3]. At the same time, RPAR minimizes the energy consumed for transmitting packets under the constraint that the deadlines are met. In this paper, we focus on minimizing transmission energy. We discuss how RPAR can be integrated with sleep scheduling to reduce the energy wasted by idle listening in Section VII-C. Other design goals of RPAR include adapting to different deadline requirements, accounting for realistic radio properties, and being scalable.

WSN applications have varied communication requirements resulting in workloads with diverse deadlines. A real-time power-aware routing protocol should dynamically find the best trade-off between latency, energy consumption and message latency: when routing packets with tight deadlines a higher transmission power is used to lower the latency and meet the deadline at the price of reduced network capacity and increased energy consumption; conversely, when routing packets with lax deadlines a lower transmission power may be used to improve energy efficiency and network capacity while still meeting the deadlines. Recent empirical studies on real WSN platforms like MICA2 motes reveal that wireless links are highly probabilistic and asymmetric[1]. A routing protocol should account for these practical issues. Finally, RPAR is designed to minimize the communication overhead and make routing decisions based solely on local information. Such a feature enables RPAR to scale to large WSNs composed of potentially hundreds of nodes with extreme resource constraints.

In this paper, we assume that each node knows its location. The location may be obtained via GPS or other localization services. We also assume that the radio can adjust its transmission power. For example, the Chipcon CC1000 radio on MICA2 and XSM2 motes can tune its power between -20dbm and 10dbm. Although RPAR varies the transmission power of packets, it does not require a power-aware MAC protocol. Instead, it is designed to work with existing CSMA protocols such as B-MAC [3]. We assume that the MAC protocol does not use RTS/CTS. This is because in WSNs that have small packets and limited bandwidth the RTS/CTS scheme introduces a high overhead [8].

IV. Design of RPAR

RPAR is comprised of four components: a velocity assignment policy, a delay estimator, a forwarding policy and a neighborhood manager. RPAR uses the velocity assignment policy to map a packet's deadline to a velocity requirement. The delay estimator evaluates the one-hop delay of each neighbor-power pair (N, p) in the neighbor table, i.e. the time it takes the node to deliver a packet to neighbor N at power level p. Based on the information provided by the delay estimator and the velocity requirement, RPAR forwards the packet using the most energy efficient neighbor-power pair in its neighborhood table that meets the velocity requirement. For the remainder of the paper we refer to a neighbor-power pair as a *forwarding choice*. When the forwarding policy fails to find a forwarding choice that satisfies the velocity requirement in the neighbor table, the neighborhood manager attempts to find new forwarding choices that will.

The rest of this section describes the velocity assignment policy, forwarding policy, and delay estimator in detail. A key feature of RPAR is its novel neighborhood manager which is presented in Section V.

A. *Dynamic Velocity Assignment Policy*

Before a node S forwards a packet, it uses the velocity assignment policy to map a packet's deadline to a *required velocity*:

$$v_{req}(S, D) = \frac{d(S, D)}{slack} \tag{1}$$

where, $d(S, D)$ is the Euclidean distance from node S to the destination node D. The $slack$ is the amount of time left before the deadline expires. Upon transmitting the packet, the source node initializes the $slack$ to the deadline. At each hop, the slack is decremented to account for queuing, contention and transmission delays. It is important to note that if the required velocity is satisfied on every hop then the deadline is also met. Therefore, RPAR maps the global problem of meeting end-to-end deadlines to the local problem of meeting the required delivery velocity on each hop.

This *dynamic* velocity assignment policy adapts the velocity requirement based on dynamic network conditions. If a packet is late then the required velocity is increased so that the packet may catch up. Conversely, if the packet is early the required velocity is decreased. We note that a similar velocity assignment policy is used for packet prioritization at the MAC layer in [9]. In contrast, RPAR uses the velocity assignment policity to make local routing decisions.

B. *Forwarding Policy*

RPAR forwards a packet to the most energy efficient forwarding choice that meets the packet's velocity requirement. Consider the case

[2] The drop-off range of delivery velocity corresponds to the boundary of the gray area reported in [1][5].

[3] The miss ratio is the fraction of packets that do not meet their deadlines.

when node S forwards a packet toward destination D using a forwarding choice (N, p). The velocity provided by (N, p) is:

$$v_{prov}(S, D, (N, p)) = \frac{d(S, D) - d(N, D)}{delay(S, (N, p))} \quad (2)$$

The one-hop delay $(delay(S, (N, p)))$ is estimated by the delay estimator (see next subsection). $d(S, D) - d(N, D)$ is the progress made towards destination by forwarding a packet to N. A forwarding choice (N, p) meets the velocity requirement if $v_{prov}(S, D, (N, p)) \geq v_{req}(S, D)$.

RPAR then evaluates the energy consumption of all forwarding choices that meet the velocity requirement. The cost associated with the forwarding choice (N, p), when S routes a packet towards destination D is:

$$E(S, D, (N, p)) = E(p) \cdot R(S, (N, p)) \cdot \frac{d(S, D)}{d(S, D) - d(N, D)} \quad (3)$$

where $E(p)$ is the energy consumed in transmitting the packet at power level p. $R(S, (N, p))$ is the expected number of transmissions before S successfully delivers a packet to N when transmitting at power level p. R is computed by the delay estimator to be discussed in Section IV-C. $\frac{d(S,D)}{d(S,D)-d(N,D)}$ is the expected number of hops to the destination. (3) estimates the expected energy consumption of routing the packet from the current node to the destination. We note that (3) resembles the routing metric proposed in [10] which outperformed greedy geographic routing when a fixed transmission power is used in lossy WSNs. In contrast, our forwarding policy applies this metric to forwarding choices with different power levels.

C. Delay Estimator

RPAR uses a novel delay estimator to predict the delays of different forwarding choices. The one-hop delay depends on three components: the contention delay ($delay_{cont}$), the transmission time for the packet and its acknowledgement ($delay_{tran}$), and the expected number of transmissions (R) [4]:

$$delay(S, (N, p)) = (delay_{cont}(S) + delay_{tran}) \cdot R(S, (N, p)) \quad (4)$$

Since the transmission time of the packet and its acknowledgement are constants determined by packet size and network bandwidth, the main function of the delay estimator is to monitor the link quality and the contention delay. We observe that the contention delay is independent of the forwarding choice when RTS/CTS is not used. Hence, we divide the delay estimation into two parts: use a single contention estimator *per node* and a link quality estimator *per forwarding choice*. This reduces the delay estimator's storage cost.

To measure link quality each node counts the number of retransmissions until a packet is successfully delivered. RPAR transmits each acknowledgement at the same power level used by the sender to transmit the packet. The link quality estimator keeps track of the total number of retransmissions and produces a link quality estimate using Jacobson's algorithm [11]. Jacobson's algorithm works by estimating the average and variation in the observed link quality to produce a conservative link-quality estimate.

The contention estimator monitors the contention delay for each packet transmission. Similarly to the link quality, the contention delay estimate is computed using Jacobson's algorithm. Equation 4 is used to compute the one-hop delay estimate based on the values of the contention and link quality estimators.

RPAR's delay estimator is designed to support real-time communication in dynamic environments. First, it estimates the conservative link quality to avoid deadline misses. Existing link estimators are designed to estimate the *average* link quality [5][12]. For example, the link estimator proposed in [5] is based on windowed mean with EWMA (WMEWMA). However, such an approach is not suitable for real-time communication since routing decisions based on average delays can still result in a large number of deadline misses. In contrast, our delay estimator adopts the Jacobson's algorithm to calculate conservative estimations of contention delays and link quality. Second, unlike the existing link estimators that require neighbors to exchange link quality information periodically [5][12], our delay estimator gets immediate link quality feedback by monitoring the number of transmissions until the sender receives an acknowledgement. Since forwarding decisions are made on packet-by-packet basis, RPAR is responsive to variations in link quality in dynamic environments.

D. Summary

While parts of RPAR are inspired by earlier work, a key novelty of RPAR lies in the unique combination of dynamic velocity assignment, geographic forwarding, and link estimation techniques for power-efficient real-time communication. This enables RPAR to dynamically adjust the trade-off between latency, capacity and energy consumption based on packet deadlines. When forwarding packets with tight deadlines, RPAR spends additional energy and reduces the network capacity, to ensure low latency. Conversely, when forwarding packets with lax deadlines, RPAR reduces the energy consumed and improves the network capacity. By making this trade-off dynamically, RPAR is well-suited for real-time applications with diverse deadline requirements.

V. Neighborhood Management

As packets are routed through a node, the neighborhood manager is responsible for finding energy-efficient forwarding choices that meet the velocity requirements of incoming packets. Achieving this goal is difficult because of the following reasons.

Due to probabilistic link quality, a node hears from a large number of potential neighbors while only a fraction of them have good link quality. This raises two potential problems: it is impossible to maintain statistics for all neighbors due to memory constraints and polluting the neighbor table with low-quality links deprecates the routing performance. We note that when power control the number of possible forwarding choices explodes[5]. Therefore, the neighborhood manager must be even more selective in maintaining forwarding choices in the table.We solve this problem by adopting the FREQUENCY algorithm [13] to maintain in the neighborhood table only those forwarding choices that are frequently used in routing packets. The details of our solution are presented in Section V-A.

More importantly, a critical issue for supporting real-time communication is to minimize the time it takes a node to find a forwarding choice that meets the velocity requirement. To solve this the neighborhood manager is invoked *on-demand* whenever a packet does not meet the velocity requirement. The neighborhood manager has two options for discovering new forwarding choices that satisfy the velocity requirement. According to the *Power Adaptation* policy, the quality of a link to a neighbor already present in the table may be improved by increasing the transmission power. Alternatively, RPAR may discover new neighbors through *Neighbor Discovery*. When a node consistently meets the deadline requirements the power adaptation policy is invoked to reduce the transmission power. This enables RPAR to improve energy efficiency and network capacity by considering new forwarding choices that incrementally decrease the transmission power and may still meet the velocity requirements. The Power Adaptation and Neighbor Discovery are presented in sections V-B and V-C, respectively.

[4] In case of a failed transmission, the sender waits for the transmission time of acknowledgement before retransmitting the packet.

[5] For example, CC1000 radio uses 31 power levels to control the transmission power between -10dbm to 20dbm. Thus, the number of possible forwarding choice increased 31 times.

A. Table management

The problem of neighborhood management with memory constraints was first addressed by Woo et. al, in their design of the MT protocol. In MT, the FREQUENCY algorithm is used to maintain the neighbors that have high link quality in the neighbor table. In contrast, we uses the FREQUENCY algorithm [13] to maintain the forwarding choices used frequently for routing in the neighbor table. The FREQUENCY algorithm works as follows. We associate a frequency counter with each forwarding choice. When a forwarding choice is used for routing its frequency counter is incremented while the frequency counters of all other forwarding choices are decremented. When the neighbor manager inserts a new forwarding choice and the table is full, the forwarding choice with the smallest frequency count is evicted. This enables RPAR to optimize its performance in terms of miss ratio and energy efficiency based on the set of velocity requirements of incoming packets.

B. Power Adaptation

The goal of the power adaption is to find the most energy efficient forwarding choices that still meet the velocity requirement. When velocity requirements cannot be satisfied for a packet, the power adaptation scheme increases the transmission power to improve the velocity provided by neighbors already in the neighbor table. On the other hand, when the velocity requirements are met, it attempts to improve the energy efficiency and network capacity by decreasing the transmission power. If a forwarding choice is eligible for power adaptation, then a new forwarding choice is inserted in the neighbor table according to a multiplicative increase and linear decrease scheme as discussed below.

When the forwarding policy cannot find a forwarding choice that meets the velocity requirement in the neighbor table, RPAR determines the neighbor whose power should be increased. We call a neighbor *eligible* for power adaptation if its link quality may be improved by increasing the transmission power. A neighbor is eligible if (i) the estimated link quality of all the existing forwarding choices associated with it are below a threshold; and (ii) there is no forwarding choice that transmits to it at the maximum power. The power of the link to the neighbor with the maximum velocity among all eligible neighbors is increased, by multiplying it by a small tunable factor α $(\alpha > 1)$. This process is repeated until a forwarding choice that meets the velocity requirement is found or no forwarding choices in the neighborhood table are eligible. In the latter case, the neighbor discovery is invoked (see Section V-C).

The power adaptation can also decrease the transmission power, to improve energy efficiency and network capacity. When the neighbor table contains forwarding choices that meet the velocity requirement of incoming packets, RPAR decreases the power of the most energy efficient forwarding choice by β (a tunable parameter) until one of the following conditions is satisfied: (i) the minimum power has been reached; (ii) the the link drops below a threshold; or (iii) when there are two consecutive power levels such that at the lower level the velocity requirement is not met but at the higher power level the velocity requirement is met.

A large value for α reduces the time until the lowest power level which has good link quality is identified. However, it wastes energy and reduces network capacity since a lower transmission power may suffice for meeting the packet deadline. A large value for β reduces the time when energy and network capacity is wasted while a large β may result in deadline misses or packet losses due to low link quality. We tune α so that four iterations are necessary to increase the power from the default power level to the maximum power. We decrease the power level by $\beta = 1$ at each iteration.

The power adaptation scheme provides a responsive mechanism for adapting to variations in link quality. A key benefit of this scheme is that it does not require *any* additional overhead packets.

C. Neighbor Discovery

When RPAR cannot find a viable forwarding choice despite increasing the transmission power through power adaptation, the neighbor discovery component is invoked to find new neighbors. The goal for neighbor discovery is to identify nodes that meet the velocity requirement. The neighbor discovery mechanism should introduce small communication and energy overhead while minimizing the time it takes to discover neighbors which are useful in meeting the velocity requirement.

In the following discussion we assume that a packet sent by node S and destined for D failed to meet its velocity requirement v_{req}. S starts the neighbor discovery by broadcasting a request to route (RTR) packet at some power p. Some node N_i hears the RTR and replies. Upon receiving the reply, RPAR inserts in its neighbor table the new forwarding choice (N_i, p). We need to address three issues: (1) What is the transmission power level p at which a RTR is transmitted? (2) How can we maintain the communication overhead to a minimum? (3) How can we minimize the transient time until a neighbor that meets the velocity requirement is found?

When the neighbor discovery is triggered because there is no neighbor closer to the destination in the neighbor table, RPAR broadcasts a RTR at the default power level. This failure usually occurs when a node routes a packet to a new destination. We chose to transmit the RTR at the default power level to reduce the impact of neighbor discovery on network capacity and energy usage. In contrast, if a neighbor closer the destination is in the table, the RTR is broadcast at the maximum power. This ensures that far away nodes which may provide high delivery velocities receive the RTR.

Since the RTR is broadcast, a large number of nodes may reply. This results in high network contention. The common solution for this is to let the replying nodes pick a randomized delay before transmitting. A node withdraws from replying if it hears replies from other nodes. This simple scheme has a drawback: although a large time window reduces the chance of packet collisions, it prolongs the time needed to find a viable neighbor that meets the velocity requirement. To find a new neighbor quickly while reducing collisions, our neighbor manager restricts the set of replying nodes to include only those that may help in meeting the velocity requirement.

A node replies only if it satisfies the following conditions: (i) it makes progress toward destination, (ii) it is not already present in the neighbor table and (iii) the maximum velocity that may be achieved by selecting it as next hop is higher than the velocity requirement. To verify that a node makes progress to the destination, we include the sender and destination locations in the RTR. In addition, the RTR also contains a list of node IDs which are already in the table and should not reply. Finally, a neighbor N replies if the following inequality is satisfied:

$$v_{req}(S,D) \leq v_{max}(S,D,N) = \frac{d(S,D) - d(N,D)}{delay_{cont}(S) + delay_{tran}} \quad (5)$$

where v_{max} is the maximum velocity that N can provide[6]. v_{max} is computed based on the minimum possible delay for S to transmit a packet $(delay_{cont}(S) + delay_{tran})$. From (5), the maximum distance between any eligible neighbor N and destination D can be derived as follows.

$$d_{max}(N,D) = d(S,D) - v_{req}(S,D) \cdot (delay_{cont}(S) + delay_{tran}) \quad (6)$$

S piggybacks d_{max} in the RTR, and a neighbor N that hears the RTR will reply only if $d(N,D) \leq d_{max}(N,D)$.

[6] This is achieved when the quality of the link between S and N is one (R(S,(N,p)) = 1)

D. Summary

In sharp contrast to earlier neighborhood management techniques that rely on periodic beacons [5], our power adaptation and neighbor discovery schemes are triggered by routing failures in an *on-demand* fashion. The reactive approach enables our neighborhood manager to respond quickly to changes in the network conditions and packet deadlines, while introducing low overhead when network and workload remain unchanged.

VI. Experimental Evaluation

We implement RPAR in a Matlab-based network simulator called Prowler [14]. To create a realistic simulation environment, we configure Prowler based on the characteristics of the MICA2 mote from Crossbow. Accordingly, a node can transmit packets at 31 power levels ranging from -20 dBm to 10 dBm, with current consumption from 3.7 mA to 21.5 mA. The bandwidth is 40 Kbps. Prowler uses the log-normal shadowing path-loss propagation model at the physical layer. A collision occurs if a receiver receives two overlapping packets with signal strengths over the receiver's sensibility. We implement the probabilistic link model from USC [15] in Prowler. Experimental data shows that the USC model produces unreliable and asymmetric links similar to MICA2 motes[4]. The MAC protocol in Prowler employs a simple CSMA scheme similar to TinyOS's MAC protocol, B-MAC [3]. As mentioned earlier we did not implement the RTS/CTS mechanism due to its high overhead on low-bandwidth WSNs. To improve the reliability we use ARQ. The maximum number of retransmissions is five. The size of the data packet and acknowledgment packets are 760 and 200 bits, respectively.

We evaluate RPAR's real-time performance and energy efficiency. The following performance metrics are used: *miss ratio* defined as the fraction of the packets that are not successfully delivered within their deadlines and the total transmission energy per data packet. We compare RPAR with two protocols that consider velocity or energy efficiency, respectively. The first baseline protocol called MaxV is inspired by SPEED protocol [12] which supports soft real-time communication by enforcing a uniform delivery velocity of packets across the network. However, to reduce the delay MaxV always chooses the neighbor with the maximum velocity. The second baseline, MinE, is an energy-efficient geographic routing protocol that selects as next hop the most energy efficient forwarding choice according to Equation 3. This routing scheme significantly outperforms greedy geographic routing in terms of energy efficiency in unreliable wireless networks[10]. Unlike RPAR, these baseline protocols operate at a fixed transmission power level. We use $protocol_L$ and $protocol_H$ to denote the protocol (MaxV or MinE) that operates at the default power level (0 dBm) and the maximum power level (10 dBm).

In simulations, we focus on the "many-to-one" traffic pattern which is common in WSN applications. In each simulation, 130 nodes are deployed in a 150m × 150m region divided into 11.5m × 15m grids. A node is randomly positioned in each grid. To increase the hop count between sources and the sink, we choose the sources from the left-most grids of the topology and the sink from the middle of the right-most grids. To simulate random data sources, the interval between two consecutive packets sent by a source is the sum of a constant (300ms) and a random value that follows an exponential distribution. We vary the mean of the exponential distribution to create different network workloads. Each result is the average of five runs. The 90% confidence interval of each data point is also presented.

We start by evaluating the performance of the three forwarding policies in the case when the neighborhood table of each node contains all forwarding choices. The link quality estimators are initialized according to the USC link model. This set of experiments is designed to quantify the best-case performance of the forwarding policies in the presence of perfect knowledge of the neighborhood and link qualities. Next, we consider the case when the neighborhood table has limited size and the link quality of each forwarding choice is estimated on-line. Finally, we evaluate the impact of different workloads on RPAR.

A. Forwarding Policy

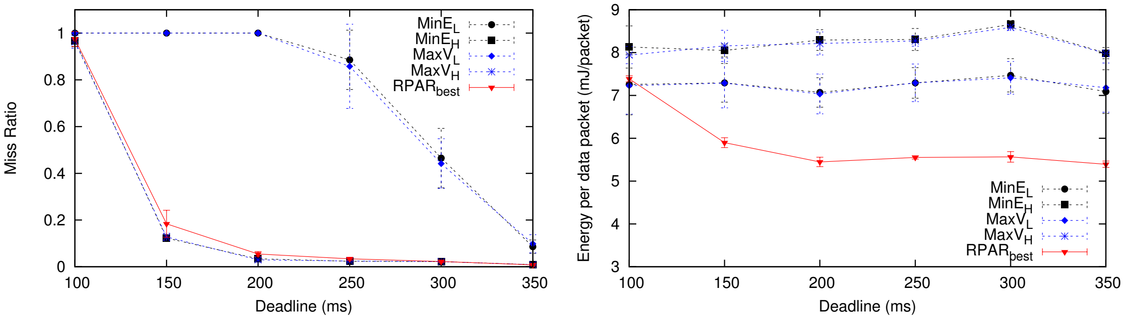
The first set of experiments use a light workload generated by three sources. Each source sends a packet every 4s. To evaluate the capability of RPAR to adapt to different real-time requirements, we vary the packet deadline between 100ms and 350ms. The deadlines are configured to be tight. This is reflected in the significant miss ratio of the baselines transmitting at default power. Even though this workload is light it still exhibits contention due to unreliable links, ARQ, and high node density. Figure 2(a) shows the miss ratio as the deadline is varied. The forwarding policies that use the default transmission power, MinE_L and MaxV_L, start missing packets when the deadline is $350ms$. As the deadline is decreased, they miss an increasing number of deadlines up to 200ms when none of the transmitted packets meets their deadline. In contrast, their counterparts that use the maximum transmission power, MinE_H and MaxV_H, have significantly lower miss ratios. This result confirms our observation Section II that using a high transmission power can effectively reduce communication delay under light load.

Figure 2(b) shows that the baseline protocols using high transmission power consume significantly more energy per packet. In contrast, RPAR consistently achieves both desired real-time performance and energy efficiency under different deadlines. As shown in Figure 2(a), RPAR achieves miss ratios close to MinE_H and MaxV_H. At the same time, as shown in Figure 2(b), RPAR consumes less energy than MinE_L and MaxV_L for all deadlines except 100ms. This is because our forwarding policy selects the most energy-efficient combination of neighbor and transmission power instead of a fixed transmission power. Note the correlation between the energy consumption and the deadline. RPAR spends additional energy to meet tighter deadlines. This shows the desired trade-off between real-time performance, energy efficiency and network capacity.

B. Performance with Neighborhood Management

This set of experiments is designed to evaluate the performance of the forwarding policies running in conjunction with the neighborhood management policies. In the following experiments, RPAR uses the on-demand neighborhood discovery scheme described in Section V. Similar to the MT protocol [5], the baselines used a neighborhood manager that uses beacons for neighborhood discovery and the FREQUENCY algorithm for table management. In all experiments each node sends beacons with a period of 20s using the same transmission power as the data packets. When the periodic beacon scheme is used, data packets start to be transmitted after 40s to allow for the neighborhood table link quality estimators to be initialized. The neighborhood management of the baselines is designed to maintain in the neighbor table only those nodes that have good link quality. However, unlike our neighborhood manager, it was not designed to support either power-control or real-time communication.

The size of the neighbor table is set to 360 bytes for all protocols. However, the considered protocols manage the allocated space differently. For the baselines, we assume they store a data structure similar to that used in MT. Even though RPAR must store the power level for each forwarding choice, it can allocate more entries in the neighbor table since its link estimator requires less storage than that used by MT. MT's link estimator must keep track of the number of received packets, the sequence number of the last received packet, and the number packets lost to compute the link quality. In contrast, our link estimator requires



(a) Miss ratio

(b) Total energy consumption per data packet

Fig. 2. Performance of considered protocols when deadline is varied. The neighborhood table is prefilled.

only the average and variation of the number of transmission per packet to estimate link quality.

The performance of RPAR is affected by the quality of the forwarding choices found in the neighborhood table. As such we consider three versions of RPAR. RPAR$_{best}$ quantifies the performance of the forwarding policy when the table is pre-filled, representing the best-case performance of RPAR. RPAR$_{cold}$ starts with an empty table and builds its neighborhood table according to the neighborhood management scheme described in Section V. RPAR$_{cold}$ is indicative of the worst-case performance of RPAR as in practice the neighborhood table is usually not empty. Therefore, we introduce RPAR$_{warm}$ that approximates the average-case performance when some forwarding choices are already in the table after routing the first 50 packets.

Figure 3 shows the performance of the forwarding policies used in combination with their respective neighborhood management policies. Figure 3(a) indicates that the miss ratio of all considered forwarding policies increased. This can be attributed to the imperfect knowledge of forwarding choices and the limited space allocated for the neighborhood table. The beacon-based neighborhood manager has a significantly higher impact on the baseline protocols than our on-demand neighborhood manager has on RPAR. In contrast to the previous set of experiments where the baselines using the maximum transmission power had miss ratios comparable to those attained by RPAR, in these experiments RPAR clearly outperforms them. When the deadline is 350ms, the benefit of our neighborhood manager is minimal. The miss ratio of RPAR$_{cold}$ and RPAR$_{warm}$ compared to that of RPAR$_{best}$ increased by an additional 3.5% and 1% respectively. Similarly, the performance of MaxV$_H$ using its neighborhood manager increased by 5% compared to the case when no neighborhood manager was used. However, as the deadline tightens the benefit of the new neighborhood manager becomes evident. At 150ms, the miss ratio of RPAR$_{cold}$ and RPAR$_{warm}$ compared to RPAR$_{best}$ increased *only* 4.7% and 1.2% respectively. In contrast the miss ratio of MaxV$_H$ jumped by 30.5%. Two factors contributed to the improved performance of RPAR's neighborhood discovery over the beacon scheme. First, our neighborhood manager is deadline aware in that it discovers and keeps forwarding choices that satisfy the velocity requirement in the neighborhood. This is particularly apparent in the case of tight deadlines when a few neighbors provide the required velocity. Second, our on-demand power adaptation and neighbor discovery schemes can find good forwarding choices more quickly than the periodic beacons. Finally, our link estimator is also more agile since it gets immediate feedback via the ACK packet.

Figure 3(b) shows the total energy consumed per data packet including the energy spent for transmitting the overhead packets. Figure 3(c) shows the energy consumed for transmitting *only* the overhead packets. Figures 3(b) and 3(c) indicate the energy consumed by the baselines accounts for a large fraction of the total energy consumed per data packet. In contrast RPAR using the on-demand neighborhood discovery scheme consumes significantly less energy. The reduction in energy consumption is attributed to both our forwarding policy (see Figure 2(b)) and our neighborhood manager which introduces much lower overhead in terms of energy. While the beacon period may be increased to lower the energy consumption, this will further degrade the real-time performance of the baselines. Note the correlation between the overhead and the deadlines in the case of on-demand neighbor discovery: more energy is spent to identify nodes that provide higher delivery velocities. Most of the overhead energy is spent during the first part of the simulation to perform the initial neighbor discovery. This explains the difference between in the energy consumption of RPAR$_{cold}$ and RPAR$_{warm}$.

C. Impact of Varying the Workload

Next we consider the case when the workload is varied while the deadline is fixed. Figures 4(a) and 4(b) show the experimental results for the case when the workload is varied by changing the number of sources from 4 to 10. Each source generates data with an inter-packet time of 6s and the deadline is fixed at 300ms. Figures 4(c) and 4(d) show the experimental results for the case when the average inter-packet time is varied from 1s to 5s and the number of sources is fixed at 3. Figure 4(a) and 4(c) show the miss ratio when the workload is varied. The graphs indicates the same trend as the previous experiment: the forwarding policies that use the default power level have a high miss ratios, while the forwarding policies using maximum transmission power have low miss ratios. The performance of RPAR is similar to that of MaxV$_H$ in terms of miss ratio. In terms of energy efficiency RPAR outperforms all the baselines except when the inter-arrival time is 1s.

VII. Discussion

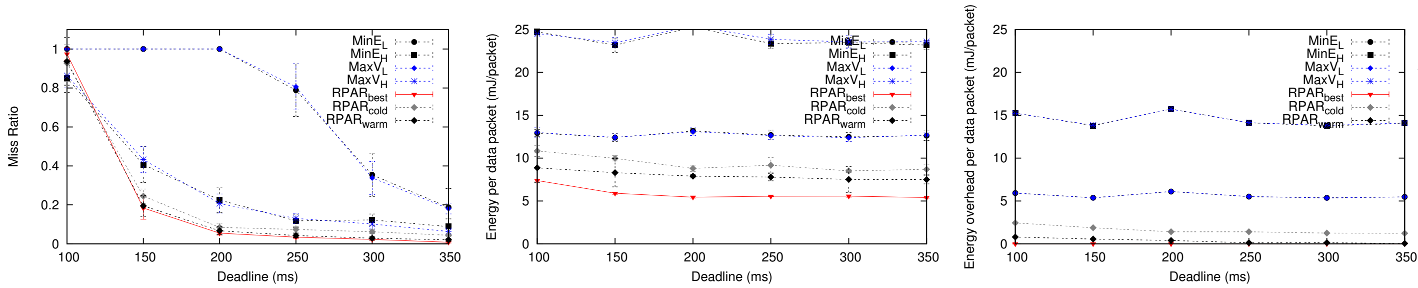
We now identify several open issues that have not been addressed in our current work, and discuss how RPAR address them.

A. Integration with Congestion Control Protocols

RPAR's power adaptation policy exhibits a pathological behavior when a node is congested, as exemplified by the following scenario: due to high contention, a node needs a large number of retries to transmit a packet correctly, due to high collision probability. Hence, RPAR increase the transmission power, worsening the situation. There are known solutions to deal with this problem. First, at the MAC layer, several methods have been proposed to allow a node to disambiguate between a packet being lost due to collision or due to poor link quality [16]. Such feedback from the MAC layer would prevent RPAR from needlessly increasing the transmission power. Second, many congestion control protocols have been proposed for WSNs [17] [18]. When a node detects that is congested RPAR should stop increasing the transmission power. This prevents the power control from worsening the congestion and allows the congestion protocols to alleviate it.

B. Handling Holes

A known problem with greedy geographic forwarding is that it may fail to find a route in the presence of network holes. Such holes may appear due to voids in node deployment or node failures. RPAR partly mitigates this issue through power control: if the diameter of the hole is smaller than the transmission range at the maximum power, then RPAR will identify a transmission power that is sufficient to transmit the packet across the hole. As a result, RPAR may still use the greedy geographic



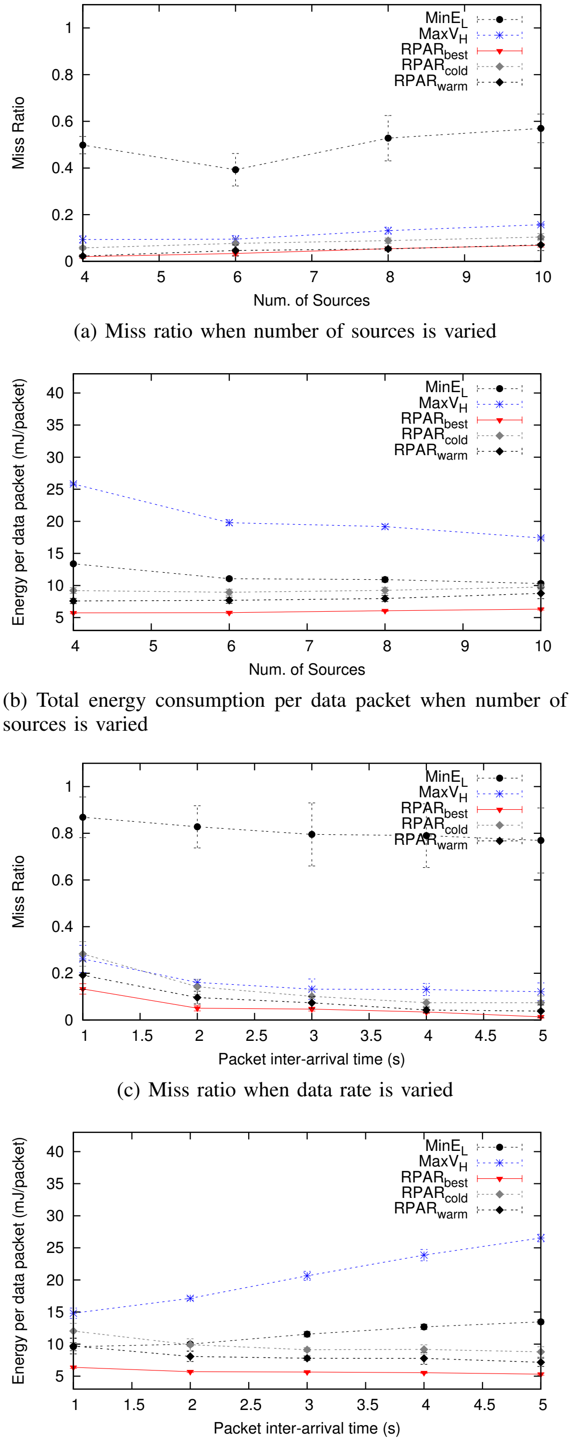
(a) Miss ratio when deadline is varied

(b) Total energy consumption per data packet when deadline is varied

(c) Total overhead energy when deadline is varied

Fig. 3. Performance of considered protocols when deadline is varied (with neighborhood management).

routing heuristic even in the presence of small holes. However, other mechanisms such as face routing in [19] [20] [21] [22] are necessary for routing packets around holes with large diameters. Integration of face routing with RPAR will be part of our future work.



(a) Miss ratio when number of sources is varied

(b) Total energy consumption per data packet when number of sources is varied

(c) Miss ratio when data rate is varied

(d) Total energy consumption per data packet when data rate is varied

Fig. 4. Performance of considered protocols when the workload is varied (with neighborhood management)

Large holes may have a negative impact on our protocol, since the Euclidean distance becomes a poor approximation of the actual path length. A first step towards solving this issue is RPAR's deadline assignment policy. By recomputing the required velocity based on the observed progress towards the destination RPAR adapts to the observed path dilation. Better performance may be obtained by computing the dilation of a routing path. This may be achieved by computing the boundary of the whole using a protocol such as Boundhole [23].

C. Integration with Power Management

RPAR aims to minimize the energy for packet transmission which is only a part of the total energy consumption of a network. To further minimize the energy consumption, a WSN needs to integrate RPAR with a power management protocol that reduces the energy wasted on idle listening. We consider two classes of power management techniques and describe how RPAR may be integrated with them.

An effective approach to reducing idle listening power is to maintain a connected backbone composed of nodes that are always active, while other nodes typically follow a periodic sleep schedule to save energy (e.g., [24] [25]). The backbone is used for routing and buffering packets destined at sleeping nodes. The last-hop delay due to a sleeping node is usually bounded by the period of its duty cycle and can be easily accounted for when routing packets. When delivering a packet to a sleeping node, the deadline needs to be adjusted to account for the additional sleep delay the packet will encounter before being delivered.

Sleep scheduling algorithms alternate periods of sleep and activity. Of particular interest to real-time applications are sleep scheduling algorithms that adjust their periods of sleep and activity based on observed workload to minimize the impact of sleep schedules on message latency, such as T-MAC [26], 802.11e Power Saving Mode, ESSAT [27] and on-demand power management [28]. As the packet is routed towards the destination, RPAR's deadline assignment policy can account for the additional delay introduced by sleep scheduling by subtracting it from the slack.

VIII. Related Work

RPAR improves upon some existing real-time protocols and complements others. The closest related works are SPEED and MM-SPEED. SPEED [12] bounds the end-to-end communication delay by enforcing a uniform communication speed throughout the network. MM-SPEED [29] extends SPEED to support multiple delivery speeds and to provide differentiated reliability. Both protocols use fixed transmission power. RPAR can achieve better performance in terms of energy and deadline miss ratio than the above protocols by considering a richer set of possible forwarding choices: at each hop we select not only the neighbor but also the transmission power. Through power control a better trade-off between latency, channel capacity and energy consumption may be achieved. This makes RPAR well-suited for applications that require power-efficient real-time communication with diverse deadline requirements. Another unique of feature of RPAR is that it is specially designed to handle unreliable wireless links prevalent in wireless networks [1] [30]. In contrast, neither SPEED nor MM-SPEED explicitly considers unreliable links in its design. Moreover, previous works did not address the important practical issue of neighborhood management which is a key component of our work.

RPAR complements RAP [9] and SWAN [31]. RAP [9] prioritizes traffic through a velocity monotonic scheduling scheme which considers both a packet's deadline and distance to the destination. SWAN provides differentiation between real-time and non real-time traffic. A key feature of SWAN is its ability to keep the network below a utilization bound through rate control. Integration of RPAR with packet scheduling and rate control may further improve the real-time performance in WSNs.

The state of the art neighborhood management is described as part of the MT protocol [5]. MT uses periodic beacon scheme to exchange neighbor information. A fundamental drawback of this approach is that it trades off responsiveness and communication overhead. The key benefit RPAR's neighborhood manager is that it reduces the time until forwarding choices that meet a packet's velocity requirement are found. This is accomplished through the deadline-aware power adaptation and neighborhood discovery policies: the neighborhood manager is invoked on-demand, whenever a packet does not meet its velocity requirement, and, through power adaptation and neighbor discovery, new forwarding choices that may meet the velocity requirement are found. Additionally, unlike a periodic beacon scheme which incurs a uniform overhead for the life-time of the network, the RPAR's neighborhood manager causes communication overhead when deadlines are missed.

While RPAR strives to reduce the miss ratio, it is also concerned with reducing energy consumption whenever possible. In this regard, RPAR is similar to work on power-aware routing which finds energy-efficient routes by varying transmission power. Singh et al. propose five power-based routing metrics that minimize power consumption or extend system lifetime [32]. Li et al. propose an online power-aware routing scheme to optimize system lifetime [33]. Chang and Tassiulas propose a local algorithm to maximize the network lifetime when the message rates are known[34]. Sankar et al. formulate maximum lifetime routing as a maximum flow problem and propose a distributed algorithm [35]. Chang et al. propose a linear programming approach to maximize the system lifetime based on a multi-commodity flow formulation[34]. Power aware routing schemes have been implemented on real wireless network platforms [36]. Gomez and Campbell [37] argue for dynamically adjusting the transmission power to improve capacity and energy efficiency. RPAR also exploits the benefits of variable transmission control to provide power-efficient real-time communication.

IX. Conclusions

We have developed the RPAR protocol to support energy-efficient real-time communication in WSNs. RPAR has several important features. First, it employs a power-adaptive forwarding policy that is cognizant of both communication delay and energy cost. By combining power control and real-time routing, our forwarding policy enables RPAR to meet packet deadlines while reducing the energy consumption. Second, RPAR features novel on-demand power adaptation and neighbor discovery schemes that can quickly identify desirable forwarding choices at minimum overhead. RPAR also addresses key practical issues in WSNs including limited memory, unreliable links, and scalability. The advantages of RPAR over several existing real-time and energy efficient routing and neighborhood management techniques have been demonstrated through realistic simulations of MICA2 motes. In the future we plan to evaluate RPAR on our physical testbed and integrate RPAR with congestion control techniques.

References

[1] Zhao, J., Govindan, R.: Understanding packet delivery performance in dense wireless sensor networks. In: SenSys '03. (2003) 1–13

[2] Cerpa, A., Wong, J., Kuang, L., Potkonjak, M., Estrin, D.: Statistical model of lossy links in wireless sensor networks. In: IPSN 2005. (2005)

[3] Polastre, J., Hill, J., Culler, D.: Versatile low power media access for wireless sensor networks. In: SenSys. (2004)

[4] Zuniga, M., Krishnamachari, B.: Analyzing the transitional region in low power wireless links. In: SECON. (2004)

[5] Woo, A., Tong, T., Culler, D.: Taming the underlying challenges of reliable multihop routing in sensor networks. In: SenSys. (2003) 14–27

[6] Gupta, P., Kumar, P.R.: The capacity of wireless networks. IEEE Trans on Information Theory **46** (2000)

[7] Abdelzaher, T.F., Prabh, S., Kiran, R.: On real-time capacity limits of multihop wireless sensor networks. In: IEEE RTSS. (2004)

[8] Woo, A., Culler, D.E.: A transmission control scheme for media access in sensor networks. In: MOBICOM. (2001) 221–235

[9] Lu, C., Blum, B.M., Abdelzaher, T.F., Stankovic, J.A., He, T.: Rap: A real-time communication architecture for large-scale wireless sensor networks. In: RTAS'02. (2002)

[10] Seada, K., Zuniga, M., Helmy, A., Krishnamachari, B.: Energy efficient forwarding strategies for geographic routing in wireless sensor networks. In: SenSys 2004 Proceedings. (2004)

[11] Jacobson, V.: Congestion avoidance and control. In: SIGCOMM '88. (1988)

[12] He, T., Stankovic, J., Lu, C., Abdelzaher, T.: Speed: A stateless protocol for real-time communication in sensor networks. In: ICDCS. (2003)

[13] Demaine, E.D., L&#243;pez-Ortiz, A., Munro, J.I.: Frequency estimation of internet packet streams with limited space. In: Proceedings of the 10th Annual European Symposium on Algorithms. (2002) 348–360

[14] Simon, G., Volgyesi, P., Maroti, M., Ledeczi, A.: Simulation-based optimization of communication protocols for large-scale wireless sensor networks. In: 2003 IEEE Aerospace Conference. (2003)

[15] Zhou, G., He, T., Krishnamurthy, S., Stankovic, J.A.: Impact of radio irregularity on wireless sensor networks. In: MobiSYS '04. (2004)

[16] Woo, A., Whitehouse, K., Jiang, F., Polastre, J., Culler, D.: The shadowing phenomenon: implications of receiving during a collision. In: UCB Technical Report. (2004)

[17] Hull, B., Jamieson, K., Balakrishnan, H.: Mitigating congestion in wireless sensor networks. (In: SenSys '04)

[18] Wan, C.Y., Eisenman, S.B., Campbell, A.T.: Coda: congestion detection and avoidance in sensor networks. (In: SenSys '03)

[19] Karp, B., Kung, H.T.: Gpsr: greedy perimeter stateless routing for wireless networks. In: MOBICOM '00. (2000) 243–254

[20] Bose, P., Morin, P., Stojmenovic, I., Urrutia, J.: Routing with guaranteed delivery in ad hoc wireless networks. Wireless Networks **7** (2001)

[21] Kuhn, F., Wattenhofer, R., Zhang, Y., Zollinger, A.: Geometric ad-hoc routing: of theory and practice. In: PODC '03. (2003) 63–72

[22] Kim, Y.J., Govindan, R., Karp, B., Shenker, S.: Geographic routing made practical. (In: USENIX 2005)

[23] Liu, J., Zhao, F., Petrovic, D.: Information-directed routing in ad hoc sensor networks. In: WSNA '03. (2003) 88–97

[24] Chen, B., Jamieson, K., Balakrishnan, H., Morris, R.: Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In: MOBICOM. (2001) 85–96

[25] Xu, Y., Heidemann, J., Estrin, D.: Geography-informed energy conservation for ad hoc routing. In: MOBICOM. (2001) 70–84

[26] van Dam, T., Langendoen, K.: An adaptive energy-efficient mac protocol for wireless sensor networks. In: SenSys 2003 Proceedings. (2003)

[27] Chipara, O., Lu, C., Roman, G.C.: Efficient power management based on application timing semantics for wireless sensor networks. (In: ICDCS '05)

[28] Zheng, R., Kravets, R.: On-demand power management for ad hoc networks. In: INFOCOM '03. (2003)

[29] Felemban, E., Lee, C.G., Ekici, E., Boder, R., Vural, S.: Probabilistic qos guarantee in reliability and timeliness domains in wireless sensor networks. In: INFOCOM '05. (2005)

[30] Son, D., Krishnamachari, B., Heidemann, J.: Experimental study of the effects of transmission power control and blacklisting in wireless sensor networks. In: SECON. (2004)

[31] Ahn, G.S., Campbell, A., Veres, A., Sun, L.H.: Swan: Service differentiation in stateless wireless ad hoc networks. In: INFOCOM '02. (2002)

[32] Singh, S., Woo, M., Raghavendra, C.S.: Power-aware routing in mobile ad hoc networks. In: MOBICOM. (1998)

[33] Li, Q., Aslam, J., Rus, D.: Online power-aware routing in wireless ad-hoc networks. In: MOBICOM. (2001)

[34] Chang, J.H., Tassiulas, L.: Energy conserving routing in wireless ad-hoc networks. In: INFOCOM. (2000)

[35] Sankar, A., Liu, Z.: Maximum lifetime routing in wireless ad-hoc networks. In: INFOCOM. (2004)

[36] Doshi, S., Bhandare, S., Brown, T.X.: An on-demand minimum energy routing protocol for a wireless ad hoc network. SIGMOBILE (2002)

[37] Gomez, J., Campbell, A.: A case for variable-range transmission power control in wireless multihop networks. (In: INFOCOM 2004)