Performance Comparison of SPIN based Push-Pull Protocols

Ravish Khosla, Xuan Zhong, Gunjan Khanna, Saurabh Bagchi, and Edward J. Coyle

School of Electrical and Computer Engineering
Purdue University, West Lafayette, Indiana 47907
Email: (rkhosla, zhongx, gkhanna, sbagchi, coyle)@purdue.edu

Abstract— **Multiple data-centric protocols - which can broadly be classified as *push-pull*, *push-only*, or *pull-only* - have been proposed in the literature. In this paper we present a framework to develop an insight into the characteristics of push-pull protocols. The performance of push-pull protocols is critically dependent on the time-out settings used to trigger failure recovery mechanisms. We perform a study of how to choose optimal timeouts to achieve best performance. Our starting point is a recently proposed SPIN-based protocol, called Shortest-Path Minded SPIN (SPMS), in which meta-data negotiations take place prior to data exchange in order to minimize the number of data transmissions. We propose a redesign of SPMS, called SPMS-Rec, which reduces the energy expended in the event of failures by requiring intermediate relay nodes to try alternate routes. Our simulation results show that SPMS-Rec outperforms SPMS, and thus SPIN, yielding energy savings while reducing the delay when multiple nodes fail along a route. We further propose a modification to SPMS-Rec through request suppression which helps in reducing redundant data transmissions.**

Index Terms—**Sensor Networks, Reliability, Time-out, Data dissemination**

I. Introduction

Sensor networks are increasingly getting used in various applications which require collection and analysis of data. Data dissemination is an important part of any sensor network. Data dissemination protocols have to face several challenges due to the unique characteristics of sensor networks. Sensor nodes are battery operated and it is often infeasible to recharge or replace the batteries, especially when they are placed in hostile terrains. Thus, data dissemination protocols have to be energy efficient, consuming the least amount of energy as possible. The data dissemination has to achieve a certain minimum level of reliability to be useful. Reliability can suffer due to many reasons; including the fact that communication is over a shared wireless channel. Various fading effects and shared channel are some factors causing packet loss. Protocols which do collision avoidance (e.g. CSMA-CA protocols) may not be feasible to be used in sensor networks due to limited resources. Thus, the data dissemination protocol must be tolerant to such failures caused by fading, collisions, and node failures.

For any-to-any communication, two classes of protocols have been widely studied – push-based and pull-based. In the push-based mechanism, the sensors push the data proactively towards the sink. Examples are flooding and directed diffusion [9]. In the pull-based mechanism, the sink(s), oftentimes mobile, queries the sensors for data. Example protocols are SAFE [8] and TTDD [7]. Neither class of protocols leverages the inherent redundancy in the data sources in order to minimize the amount of data exchanges. It is a fact that in many deployments a particular part of the sensor field is covered by multiple sensors or detection of an event happens at multiple sensors. For example, in the CSO mitigation project [14], the storm event being a non-point event can be detected by multiple sensors embedded in the underground channels at different points.

SPIN [10] is the first in the class of *push-pull* protocols. It initially proposed the idea of exchanging compact data descriptors prior to data exchange to reduce the redundant data transmissions. Thus, a three-way handshake between the source and the destination occurs through advertisement of data (broadcast by source), request for data (destination to source), and data transmission (source to destination). The key point is that only interested nodes need request for the data after examining the meta-data in the advertisement thus avoiding redundant transmissions as compared to flooding. Shortest Path Minded SPIN (SPMS) [4] extends SPIN by performing multi-hop communication for the request and the data, within a node's maximum transmission radius termed as '*zone*'. It assumes that the network is connected at the minimum transmission power and a node is capable of transmitting at different power levels. Each node maintains a fixed number of alternate routes to any other node in the zone and the energy cost of reaching the destination for each route. SPMS reduces the energy and the delay compared to SPIN and is designed to be resilient to the failures of the intermediate relay nodes.

We propose a new protocol SPMS-Rec which is designed to further reduce the energy and delay for data dissemination in case of large failures in the communication path. Intermediate nodes (between source and destination) provide local re-transmissions to achieve a high reliability.

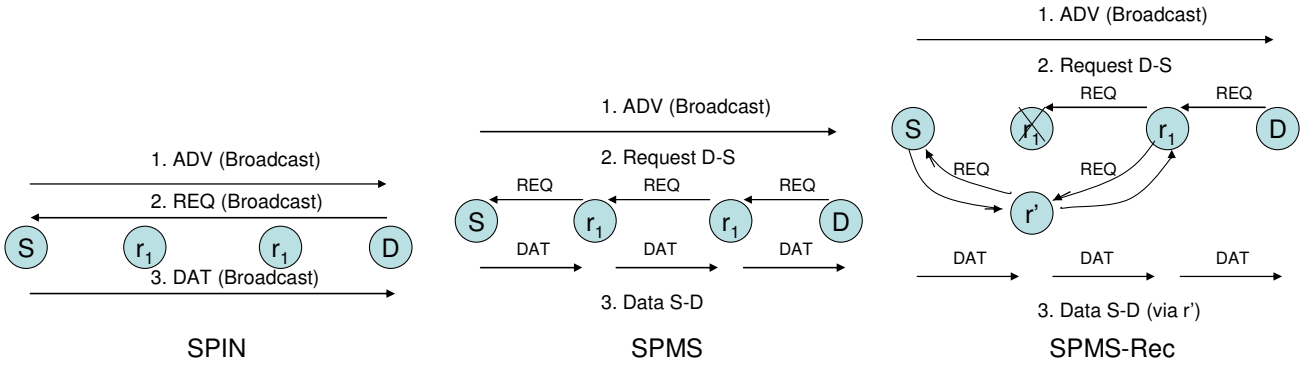


Figure 1: Illustration of data dissemination under SPIN, SPMS and SPMS-Rec.

We also propose a variant to SPMS-Rec called request suppression (SPMS-Rec-RS) aimed at reducing the number of requests for the same data. We implement SPIN, SPMS, SPMS-Rec and SPMS-Rec-RS in GTSNets[1]. Through extensive experiments we study the reliability, energy and delay of these protocols for data dissemination under failure conditions. We observe that SPMS-Rec outperforms SPIN and SPMS in energy and delay savings while providing a high reliability.

II. SPMS-REC & EXTENSIONS

A. *SPMS-Rec*

SPMS-Recursive (SPMS-Rec) is designed to reduce the energy consumption for data dissemination when node and link failures happen in the network during the three-way handshake for the transmission of a data packet. In SPMS if a relay node, say node α, between the destination-source pair (D, S) during the request transmission phase finds its next hop unavailable then it simply drops the packet and node D times out. Node D then tries alternate means of sending the request to node S. This wastes energy and significantly increases the delay because the request may already have traversed multiple hops to get to the intermediate node α where the failure occurred. This consequently is inefficient in energy and in end-to-end latency. The inefficiency is magnified if the failure happens when the data is on its way from S to D The problem gets exacerbated with increasing failures. The cause of this problem in SPMS is that relay nodes are stateless and thus can not detect or recover from failures. Responsibility for the three way handshake thus rests completely with the source and destination nodes.

SPMS-Rec solves this problem by enabling each intermediate node to infer a broken route if an expected follow-up event does not occur. For example, if an intermediate node has forwarded a request packet, it expects to see the corresponding data packet within a timeout interval. On detecting the broken route, the intermediate node attempts to reach the end point through an alternate route, and failing that may try direct communication with the end point. This process requires the intermediate nodes to maintain transient state information about the handshakes being routed through them.

Each node maintains a routing table for neighbors within its zone using the Distributed Bellman-Ford (DBF) algorithm. There are two entries for each zone neighbor – the first corresponding to the primary choice for the next hop to reach the neighbor, and the second corresponding to the secondary choice. The preference for a path is determined by the energy cost of reaching the destination and not the number of hops, henceforth referred to as the "shortest path". By power level, we mean one of the discrete transmission power levels supported by the hardware. The transmission power level to reach the destination directly is also maintained.

If a node has data to disseminate, it broadcasts an ADV to all the nodes within the zone. The interested nodes do not send a REQ immediately but wait for a timeout T_{OUT}^{ADV} for the data to come to a closer node. Each node maintains a PRONE, which is the energy-wise preferred source to get the data from and SCONE, which is the alternative source of data if the PRONE fails or is unreachable. The failure free operation of SPMS-Rec is identical to that of SPMS. The destination node D sends a REQ to the PRONE, say node S, in a multi-hop fashion through the shortest path. It starts a timer $T_{OUT}^{DAT}(D)$. Let node I_r be a relay node receiving the REQ from D intended for the final destination node S. Node I_r first tries to send the REQ through shortest path and in the process also starts a timer $T_{OUT}^{DAT}(I_r)_1$ within which it expects to get the data. Each time a node forwards a REQ packet it stores some transient state information in its request-table. For each active request, the request-table contains a unique identifier of the node that forwarded the REQ to node I_r. Once $T_{OUT}^{DAT}(I_r)_1$ timer expires, node I_r tries to send the request through the alternate path to node S. In the process it starts another timer $T_{OUT}^{DAT}(I_r)_2$. When $T_{OUT}^{DAT}(I_r)_2$ expires, the relay node either decides to send the REQ directly (i.e., one-hop) or drops it. This decision is made due to the observation that if node I_r is too far from node S, it is desirable that node I_r does not try direct transmission and instead let the previous node (closer to node D) time out and

try alternate means of reaching node *S*. Assuming a node index increasing from *S* to *D,* the index of node I_r beyond which direct transmission should not be tried can be computed by node *D* [13]. This can be piggybacked on the REQ as a Relay To Send (RTS) value which is decremented at each hop. If RTS falls to zero, this implies that the holder of the request can send the request directly, if multi-hop communication fails.

When the source node *S* receives the REQ, it initiates the DAT transmission to the node *D*. Node *S* adds the unique request ID to the DAT and forwards it to the next hop relay node which had sent it the REQ. Each relay node forwards the DAT back on the same route from which the REQ had arrived. This design is motivated by the underlying assumption that the topology between *S* to *D* does not change either due to mobility or failures in the time between the REQ and the DAT. This ensures that the failed routes that were discovered during the REQ phase are avoided in the DAT phase. In case of a failure in the DAT propagation, the destination times out due to a data timer $T_{OUT}^{DAT}(D)$and restarts the REQ process through an alternate path. In summary, SPMS-Rec avoids costly end-point timeouts by having intermediate nodes store transient state information to enable them to locally detect and recover from failures.

B. Extension of SPMS-Rec: SPMS-Rec Request Suppression (SPMS-Rec-RS)

The goal of SPMS-Rec-RS is to reduce the number of redundant transmissions by suppressing unnecessary requests. A possible scenario in SPMS-Rec is the following: Assume Node *P* has originated or forwarded the request for some data d_i from node *S* and while it is waiting for the data, it gets a request for d_i directed to *S* from another node *Q*. Instead of forwarding the request to *S* as in SPMS-Rec, *P* can choose to suppress the forwarding if it is already trying to get the data or already has the data. The first case happens if *Q*'s T_{OUT}^{ADV} is set too low or if *P* experiences unanticipated delay in the path from *S*. The second case happens if *P* already received the data d_i but its advertisement was not heard by *Q* due to a collision. In either case, *P* sends the data back to *Q*. However if *P* exhausted all the ways of getting the data, then it should send the request on behalf of *Q* to *S*. Redundant traffic is thus reduced at the cost of performing more computations at intermediate nodes without affecting the delay. This variant causes some increased state to be maintained at the intermediate nodes. In SPMS-Rec, each intermediate node keeps track of the request packets which it has relayed. In SPMS-Rec-RS, apart from nodes whose request packets have been forwarded, id of nodes whose requests have been suppressed, also need to be maintained.

C. Setting Timeout Values

In all the push-pull protocols, the timeout settings are critical as they directly affect the performance metrics of energy, delay, and reliability. For SPMS, setting a large timeout causes the network to operate at low congestion levels, leading to fewer collisions and retransmissions. Hence the reliability is better and the transmission energy spent is lower; however the delay and listening energy increase. Setting very low timeouts runs counter to the spirit of the protocols since a node should allow a nearer node sufficient time to get the data and to re-advertise so that it can request the data from the nearer node. Similarly in SPMS-Rec, an intermediate node must wait for all the nodes on the path from itself to the destination (of REQ) to try out all possible ways of reaching destination and then start trying its own alternate paths. Setting low timeouts would cause many redundant transmissions and waste energy.

It is difficult to analytically determine optimal timeout values. The timeout determines the number of contending nodes and hence the congestion in the network, which in turn affects the MAC delay, further making an analytical solution difficult. Therefore we experimentally determine the timeout for different network sizes to achieve user defined QoS parameters.

III. EXPERIMENTS

We build simulation models of SPIN, SPMS, SPMS-Rec, and SPMS-Rec-RS in GTSNets (Georgia Tech Sensor Network Simulator) [1], which is capable of scaling beyond the capabilities of ns-2. The simulator consists of customizable network layers, a subset of which is modified for our models. The layer 2 MAC has been implemented as a modified version of IEEE 802.11 MAC without RTS/CTS/ACK while retaining the functionality of CSMA/CA. Without the use of ACK, collisions cannot be detected and hence are treated as failures equivalent to node failures, since they cannot be distinguished.

The model parameters have been taken from the datasheet of the latest Crossbow Mica2 motes [2]. Of importance is that each node has 31 power levels – from -20 dBm to 10 dBm at increments of 1dBm. The control packets (ADV and REQ) are each 12 Bytes, while the data packet is 250 Bytes. The nodes are placed at a constant spacing on a square grid with wrapping around to avoid boundary conditions. Thus the size of the grid increases with the number of nodes in the network. Each node generates packets according to an exponential inter-arrival time. Each node on receiving the ADV makes a decision whether it is interested in the data. The output metrics of interest are reliability, energy, and delay. All the results presented are averaged over all the unique data packets in the network at the end of a given run. The important simulation parameters are shown in Table 1.

Table 1: Default parameter values for experiments

Sensor spacing	18.2 m	Number of data packets per node	10
Network MTTF	25 s	Fraction of nodes interested in data	0.1
Fraction of nodes liable to fail	0.25	Inter-packet time (λ)	1 s

A. *Failure free operation – SPIN and SPMS*

In the first set of experiments, we quantify the advantage of SPMS over SPIN in the failure free case. It may be argued that the advantage of SPMS over SPIN has already been demonstrated in [4]. However, as would have become evident to the discerning reader, the performance of both protocols is very dependent on the setting of the timeout values. In our experiment, we perform a systematic study of the behavior of the two protocols with varying timeout values and come up with a reasoned timeout setting for the comparison.

Also, one has to be cautious in comparing the protocols since three primary metrics are involved – reliability, energy, and delay and for contrasting the value of a given metric, the others have to be comparable. While performing this experiment, we uncovered the poor performance of SPIN with simply natural collisions in the network and therefore modified SPIN by having interested nodes wait a random time before sending the request. The comparison of SPMS to this modified SPIN is given. Energy (per packet) is computed as the sum of the battery energy drained for all the nodes in the network divided by the total number of unique data packets introduced in the network. For the energy calculation, the transmission and reception energies are taken into account but not the idle listening energy since that can be optimized by an independent sleep-awake protocol. However, importantly, since SPMS has better delay performance than SPIN, including the idle listening energy will simply increase the energy improvement, provided a sleep-awake protocol of similar efficiency is applied to both. The delay is measured from the time the first advertisement for the data is sent out to when the data packet is received at a node and is averaged over all the received packets. The reliability is defined as $N_h/(N_h+N_d+N_{ADV})$ where, N_h is the number of nodes that have the data (excluding the source nodes), N_d is the number of nodes that have sent the REQ but have not received the DAT, and N_{ADV} is the number of nodes that have not heard the ADV (independent of its interest in the data). The reason for the N_{ADV} in the denominator is that we want to penalize the protocol when a node does not hear the ADV for a particular data packet as this may be caused by collision with an existing transmission induced by the protocol. In all the simulations, the network is a single zone.

Two protocols can be compared in energy and delay only when they have the same reliability. Not doing so will favor a protocol that achieves a lower energy simply by satisfying a fewer number of nodes in the network. When the original SPIN protocol (henceforth referred to as basic-SPIN) was simulated, a low reliability was found especially with larger number of nodes, even without explicit injection of failures. This is because SPIN transmits REQ and DAT at the power level required to reach the destination node and thus causes many collisions in a dense network. Since there is no backup mechanism when a DAT or REQ gets lost, the reliability suffers greatly. Retransmitting at a higher power level, especially the large sized DAT, introduces more congestion in the network and may degrade the reliability further. In our simulations, we follow an alternate approach of increasing the reliability of SPIN by introducing random backoffs before sending the REQ to a source that has advertised the data. To have meaningful comparisons with SPMS, we have to increase reliability of SPIN to a high enough threshold, 90% for the experiments, by tuning the random backoff period. This has the effect of making SPIN operate in a lower congestion region compared to SPMS to achieve the same reliability. The experimental results show that the modified version of SPIN (henceforth referred to as mod-SPIN), performs worse in both energy and delay compared to SPMS.

B. *Exploration of timeout values*

For simulating SPMS, we need to determine the various timeouts to be used in the protocol, namely T_{OUT}^{ADV}'s and T_{OUT}^{DAT}'s.. We model each of the timeouts to be proportional to the square of the number of hops between the destination and the source following the empirical model of MAC layer delay as Gn if n nodes are contending for the channel and there is little congestion [11]. Thus, each timeout is represented by $T_{unit} * h^2$ where T_{unit} is a normalized constant and h is the hop distance between this node and source.

The determination of T_{unit} is done in each topology by sweeping through a range of T_{unit}'s and picking the value for which the non-idle energy is minimized while satisfying the constraints Reliability > R_0 and Delay < D_0 where R_0 and D_0 are suitable user defined values. A sample is shown for a 100 node topology in Figure 2. The value of T_{unit} is normalized with respect to a DAT packet transmission time. If we choose R_0 = 90%, D_0 = 17 s, then to achieve an energy minimum, T_{unit} (normalized) is approximately 1200. The plots show that as T_{unit} increases, the reliability increases and the energy drops as less redundant transmissions are now required and lesser number of collisions are caused. However the delay increases due to increasing T_{unit}. A similar strategy is used to find the value of the maximum back-off needed in mod-SPIN to satisfy the delay and the reliability constraints.

C. *Lower bound on protocol's energy requirement*

The energies considered so far have been at the end of the simulation run. However to achieve a particular reliability, one can find out the minimum energy required by plotting a sample path of the number of nodes satisfied with time. Snapshots of the system are taken at discrete time points and as soon as the fraction of interested nodes receiving the data (f_r) reaches the user defined level R_0, the energy consumed at that instant is the minimum energy required to achieve this reliability. A sample path for a 144 node topology for both SPMS and mod-SPIN, with T_{unit} for SPMS and for SPIN obtained as above, is shown in Figure 3(a).

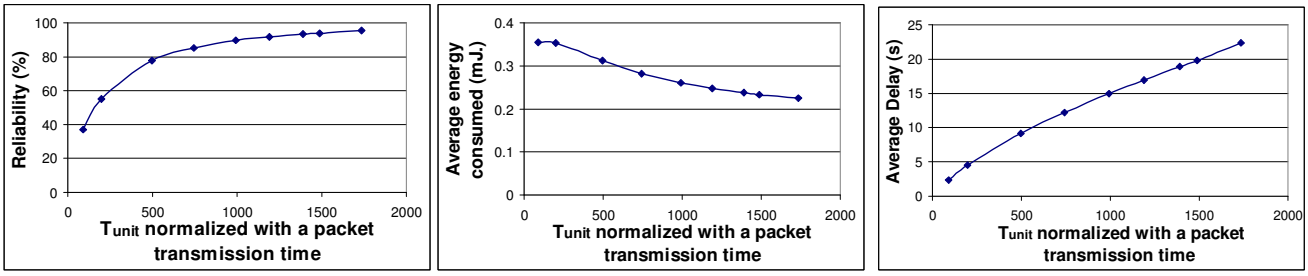
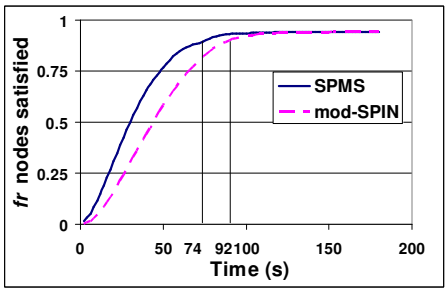
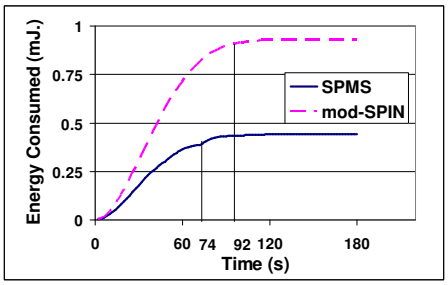


Figure 2: Reliability, Energy and Delay of SPMS in a 100 node topology

Thus if R_0 = 90%, the time to achieve this reliability for SPMS is about 74 sec and 92 sec for mod-SPIN. This shows that SPMS achieves faster dissemination of data than SPIN, even if it has the same eventual reliability as SPIN (94%). Thus the minimum energy required is read from the energy curve to be about 0.395 mJ for SPMS and about 0.9 mJ for SPIN. For the failure free case, the energy and delay of SPMS and mod-SPIN are compared in Figure 3(b). The results show that SPMS saves about 15-45% in delay and 18-56% in energy compared to mod-SPIN.



(a)



(b)

Figure 3: (a) Fraction of interested nodes satisfied for SPMS and mod-SPIN with 144 nodes; (b) Energy consumed for SPMS and mod-SPIN with 144 nodes.

D. Failure Operation of SPMS, SPMS-Rec and SPMS-Rec-RS

In this experiment, we compare SPMS and SPMS-Rec in failure scenarios, again with a certain fraction f_r = 25% of the nodes failing. In SPMS-Rec we model all the timeouts as $T_{unit}*h^3$, except for direct transmissions which are still modeled as $T_{unit}*h^2$. This is because a node *D* which is *h* hops away from the data source *S* has to allow sufficient time for the nodes in the path to *S* to try their best and alternate paths and possibly direct transmissions before *D* times out. T_{unit} is determined without failures.

The comparison of SPMS and SPMS-Rec in terms of energy and delay is shown in Figure 4(a). For low values of MTTF (1 s), it was found that reliability of SPMS is lower than SPMS-Rec, hence preventing a valid comparison in energy and delay. In order to actually measure energy efficiency of the protocol, we normalize the total energy consumed by the number of packets received. The results show that as the number of nodes increase, the energy and delay of both protocols increase. The delay increases as the data flows through larger number of zones with the increase of network size. SPMS-Rec shows increasing energy savings over SPMS with network size, with a 13% improvement at 196 nodes. The reason for the slight decrease in energy at 196 nodes is due to the normalization – the overall energy increases, but since the number of data received increases faster, the energy consumed per packet decreases. SPMS-Rec also shows an improvement over SPMS in delay increasing with network size, which is expected as SPMS-Rec prevents costly timeouts at the end points due to failures, allowing intermediate nodes to try alternate paths to the data source.

We compare the performance of SPMS-Rec-RS with SPMS-Rec. SPMS-Rec-RS aims to reduce the number of redundant transmissions in the network, hence the metric used to compare it with SPMS-Rec is the number of duplicate data received by a node for each data item it is interested in, normalized by the number of data packets received. The results in Figure 3(c) show that SPMS-Rec-RS improves the number of redundant data received significantly, about 63-87%.

IV. CONCLUSIONS

In this paper we proposed a new reliable protocol (SPMS-Rec) for data dissemination in sensor networks and compared the performance of existing SPIN-based protocols. The variants of SPIN-based push-pull protocols compare favorably to the baseline SPIN protocol in the failure free case. SPMS-Rec is found to give energy and delay savings over SPMS in the presence of failures. SPMS-Rec-RS is shown to be successful in suppressing redundant data transmissions. Future work aims at providing a framework so that a user can select a data dissemination protocol based on the application parameters like fraction of interested nodes (number of sinks), number of sources, node density etc.

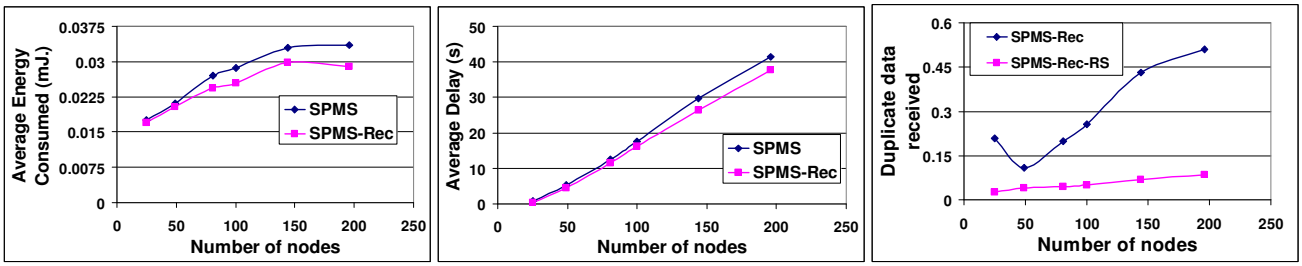


Figure 4: Comparison of (a) Energy and (b) Delay for SPMS and SPMS-Rec for various numbers of nodes; (c) Effectiveness of SPMS-Rec-RS

REFERENCES

[1] E. Ould-Ahmed-Vall, G. Riley, and B. Heck, "GTSNetS: The Georgia Tech Sensor Network Simulator," 7th ACM Intl. Symp. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM), 2004.

[2] Crossbow Technology Inc., MICA2 Mote http://www.xbow.com

[3] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation based protocols for disseminating information in wireless sensor networks," *ACM Wireless Networks*, Vol. 8, Mar-May 2002, pp.169-185.

[4] G. Khanna, S. Bagchi, and Y-S Wu, "Fault Tolerant Energy Aware Data Dissemination Protocol in Sensor Network," *IEEE Dependable Systems and Networks Conference (DSN)*, 2004, pp. 739-748.

[5] J. Heidemann, F. Silva and D. Estrin, "Matching Data Dissemination Algorithms to Application Requirements," *Proceedings of the ACM Conference on Embedded Networked Sensor Systems, (SenSys)*, 2003.

[6] S. Kim, S. H. Son, J. Stankovic, S. Li, and Y. Cho, "SAFE: A Data Dissemination Protocol for Periodic Updates in Sensor Network," *International Workshop on Data Distribution for Real-Time Systems*, 2003.

[7] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang, "A two-tier data dissemination model for large-scale wireless sensor networks", *8th ACM Intl. Conf. on Mobile Computing and Networking (MobiCom)*, September 2002.

[8] S.J. Park, R. Vedantham, R. Sivakumar and I. F. Akyildiz, "A Scalable Approach for Reliable Downstream Data Delivery in Wireless Sensor Networks," *5th ACM Intl. Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc)*, pp. 78 – 89, 2004.

[9] C Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," *6th Intl. Conf. on Mobile Computing and Networking (MobiCOM '00)*, 2000.

[10] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, October 2002, pp. 660-670.

[11] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas on Communications*, Vol.18, no.3, pp.535-547, Mar 2000.

[12] J. H. Kim and J. K. Lee, "Performance analysis of MAC protocols for wireless LAN in Rayleigh and shadow fast fading," I*EEE GLOBECOM '97*, Vol. 1, Nov 1997.

[13] Ravish Khosla, "Reliable Data Dissemination in Energy Constrained Sensor Networks", Master's Thesis, School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, August 2006.

[14] U. of Notre Dame, Purdue University, et al. "Detection and Control of Combined Sewer Overflow Events Using Embedded Sensor Network Technology," [Online] http://www.nd.edu/~lemmon/CSO