IMPROVING DATA FRESHNESS TO ENHANCE THE QUALITY OF OBSERVATIONS IN WIRELESS SENSOR NETWORKS

by

Ramya Bala Ammu

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Computer Science

Boise State University

May 2011

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Ramya Bala Ammu

Thesis Title: Improving Data Freshness To Enhance The Quality Of Observations In Wireless Sensor Networks

Date of Final Oral Examination: 28 January 2011

The following individuals read and discussed the thesis submitted by student Ramya Bala Ammu, and they evaluated her presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Dr. Sirisha Medidi	Chair, Supervisory Committee
Dr. Murali Medidi	Member, Supervisory Committee
Dr. Jyh-haw Yeh	Member, Supervisory Committee

The final reading approval of the thesis was granted by Dr. Sirisha Medidi, Chair of the Supervisory Committee. The thesis was approved for the Graduate College by John R. Pelton, Ph.D., Dean of the Graduate College.

Dedicated to my family

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Sirisha Medidi for her valuable support and guidance in mentoring me during the course of my education. It has been a wonderful learning experience with her. This thesis would not have been possible without her help and support. I would like to thank Dr. Murali Medidi for the help and inspiration he provided throughout my research. I would like to thank Dr. Jyh-haw Yeh for being on my committee and for his guidance during my study at BSU. I am indebted to my husband, my parents, and my in-laws for providing constant encouragement and support and without whom I would not have completed this endeavour. Finally, I would like to thank all members of swanlab for their tremendous support in completing my thesis.

ABSTRACT

Wireless Sensor Networks (WSNs) are used to achieve either continuous monitoring or event-detection in the area of interest. In continuous monitoring applications, each sensor node transmits its sensed data to the sink node (base station) periodically; while in event-detection driven applications, nodes report to the sink node once an event occurs. Continuous monitoring applications require periodic refreshed data at the sink node. Data reaching the sink node after a certain threshold is not useful for processing or analysis because it is stale. Data freshness along with reliable data delivery is critical in such applications. Current protocols in this area measure freshness only in terms of latency or delay of packets received at the sink node. However, improving overall delay alone does not necessarily improve data freshness. To address the needs of continuous monitoring applications, we adapt an existing protocol that provides packet-level reliability and augment it with prioritization and rate control mechanisms to improve data freshness. We implemented our protocol using the ns-2 simulator for evaluating its performance and identified metrics for measuring data freshness. Results show that prioritization and rate control improves data freshness significantly.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Advances in technology have led to the development of tiny, low cost and low power devices or sensor nodes equipped with multiple parameter sensing, processing, and communication capabilities. A Wireless Sensor Network (WSN) consists of a large number of sensor nodes that may be deployed randomly and densely. A Wireless Sensor Network typically consists of a base station and a group of sensor nodes. Sensor nodes are small electronic components capable of sensing different types of information from the environment, such as temperature, light, humidity, and radiation. The nodes process this information and forward it to a user, or, in general, a base station. Each node consists of processing capability (one or more microcontrollers, CPUs, or DSP chips), may contain multiple types of memory (program, data, and flash memories), have an RF transceiver (usually with a single omni-directional antenna), have a power source (e.g., batteries and solar cells), and may accommodate various sensors and actuators. The base station can query the sensor nodes for information and is responsible for the collection of data and relaying to other networks. Sensor networks are used in several applications, including habitat and ecosystem monitoring, seismic monitoring, rapid emergency response, perimeter security and surveillance, ground water contamination monitoring, and etc. Sensor nodes have limited power, storage, and processing capability, which necessitates the need for light weight protocols in sensor networks.

Wireless Sensor Networks (WSNs) are used to achieve either continuous monitoring or event detection in the area of interest. In continuous-monitoring applications, each sensor node transmits its sensed data to the base station periodically; while in event-detection driven applications, nodes report to the base station once an event occurs. Continuous-monitoring applications require periodically refreshed data at the base station. For instance, an equipment-monitoring application in hospitals requires fresh data periodically to analyze the machinery status and take necessary and preventive action when required. Data freshness is critical in such applications.

So far, data freshness was measured only in terms of latency or delay of packets received at the base station [30], where the focus is on how long a node should wait before it aggregates data packets such that delay is reduced. In fact, they do not address if sufficient data from an area or source has been received as desired by the application. To improve data freshness, it is necessary for the data packets to reach the base station reliably. One protocol that improves packet-level reliability using the concept of monitors was proposed in [36]. In order to cater to the needs of continuous-monitoring applications and improve data freshness in our light weight protocol, we will tailor the monitor's approach that provides packet-level reliability, and adapt prioritization and rate-control mechanisms to improve data freshness.

1.1 Organization of Thesis

The rest of this thesis is organized as follows. Chapter 2 reviews the related work in the area. The motivation and the proposed protocol design for improving data freshness is explained in Chapter 3. Chapter 4 presents the performance evaluation of the protocol. Finally, the conclusion and future research are presented in Chapter 5.

CHAPTER 2

RELATED WORK

Wireless Sensor Networks (WSNs) can be typically used to achieve either continuous monitoring or event detection in an area of interest. In continuous-monitoring applications, each sensor node transmits its sensed data to the sink (base station) periodically; while in event-detection driven applications, once an event occurs, it is reported to the sink. In continuous-monitoring applications, such as equipment monitoring in hospitals, fresh data is required periodically to analyze the machinery status and take necessary and preventive action when required. Data reaching the base station after a certain threshold is not useful for processing or analysis. Data freshness along with reliable data delivery is critical in such applications. Many protocols have been proposed to minimize latency and provide reliability in WSNs.

2.1 Latency

Data freshness can also be improved by minimizing the latency in sensor networks. Several protocols were proposed for minimizing the latency [5, 8, 12, 14, 16, 17, 24, 25, 28, 31, 34]. For example, a surveillance system may require the position of an intruder be reported to a command center within a specified time interval so that pursuing actions can be taken in time. In such applications, it is required that data reaches the information sink within a threshold time. This requires data freshness as data received after a certain threshold

is considered stale and will not be utilized for evaluation or analysis. In [5], the authors proposed RAP, a new real-time communication architecture for large scale wireless sensor networks. RAP provides convenient high-level query and event services for distributed micro-sensing applications. A new packet scheduling policy called Velocity Monotonic Scheduling is used. This scheduling policy assigns a velocity to every packet and transmits the packets according to their velocity: a higher velocity packet has higher priority. This ensures that urgent packets reach the destination within a short period of time. Sift [12], a MAC protocol for wireless sensor networks, performs well when spatially correlated contention occurs and adapts well to the sudden changes in the number of sensors that are trying to send data. Sift is ideal for sensor networks where it is often sufficient that any of R of N sensors that observe an event report it, with the remaining nodes suppressing their transmissions. Sift is based on the intuition that when we are interested in low latency for the first R reports, it is important for the first few successful slots to be contention free. To tightly bound contention latency, a fixed-size contention window is used. However, Sift does not address energy efficiency. Also, it is designed for only one-hop transmission. Psift [24] is an improvement over Sift and it works for multi-hop sensor networks as well.

A new topology management scheme called Sparse Topology and Energy Management (STEM) [28] has been proposed to reduce energy consumption in a monitoring state to minimum while ensuring satisfactory latency for transitioning to the transfer state. The majority of the time, the network is only sensing its environment. This is referred to as a monitoring state. Once an event occurs, data has to be forwarded to the sink and the network transitions to the transfer state. STEM reduces energy consumption of the nodes by putting them into a sleep state. In the monitoring state, instead of full sleep, a node goes into low power listen mode. However, in return for this energy reduction, a certain amount of latency will be introduced to wakeup the nodes. Nodes in this design have three

states: sleep, active, or listening. The initiator node polls the target node continuously. As soon as the target node hears the poll, the link between the nodes is activated. Once the link is activated, data transfer takes place using a MAC protocol. To wake up a node, a wake-up message is sent to the node in the form of a beacon packet (STEM-B, meaning beacon-based) or a simple tone (STEM-T, meaning tone-based), resulting in two variants of STEM. The topology management in STEM is specifically geared toward those scenarios where the network spends most of its time waiting for events to happen without forwarding traffic. Simulation results show a considerable improvement over GAF in both scenarios. Though this scheme has many advantages, it suffers from the energy consumption due to continuous polling and requires extra radio on the sensor nodes.

Latency minimized energy efficient MAC protocol [8] is a hop-ahead reservation scheme that minimizes the latency and increases energy efficiency. It does this by reserving the next hop's channel in advance. It is useful in time-critical applications where sensed events are to be reported immediately to the sink to take remedial or defensive actions. LEEM assists in sending the packets with minimum delay by using dual frequency radio set up. Since the next hop's channel is reserved in advance, the intermediate nodes in the data path forward the packet as soon as it is received. In an event-driven sensor network, the nodes spend a lot of time sensing the event. In order to reduce this energy, the control channel radios are kept in a lower power sleep mode. The control channel radio is made active periodically to check for any data transmissions and to activate the data channel. In LEEM, nodes are resynchronized every hour. The synchronization helps in making reservations and reduces the delay. In a synchronized network, since each node knows the time at which its next hop node is active, it need not send continuous wake-up signals. This results in lower energy consumption. The reservation scheme in LEEM helps in eliminating the set up latency at the intermediate nodes. When a sensor node's control channel radio is in sleep mode and

an event occurs, the sensor node waits for the next hop node to become active. It then requests the next hop node to activate its data channel radio by sending a request packet. The receiver agrees by sending back an ACK. This procedure continues throughout the data path until the packet reaches the data sink. Whenever the data transfer takes place, the receiver of the packet reserves the channel for K hops ahead. If the value of K is one, it is a one-hop reservation scheme, otherwise it is an N-hop reservation scheme. When the current transmission gets completed at the receiver, the next hop channel becomes ready. Hence, the delay in setting up the next channel is avoided except at the first hop. Although LEEM shows significant improvement over other protocols like STEM and PTW in terms of energy consumption and latency, it is not applicable for applications that have a continuous occurrence of events.

DMAC [20] is an energy-efficient and low-latency MAC protocol that is designed and optimized for applications in which the major traffic pattern consists of data collected from several source nodes to a sink through a unidirectional tree. DMAC uses a staggered active/sleep schedule to solve the data-forwarding interruption problem and enables continuous data forwarding on the multi-hop path. In DMAC, data prediction is used to enable active slot requests when multiple children of a node have packets to send in the same sending slot, while a More-to-Send packet is used when nodes on the same level of the data-gathering tree with different parents compete for channel access. Nodes that are out of the hearing range of both the sender and receiver are unaware of the ongoing transmissions, and therefore go to sleep until the next cycle/interval. The data-forwarding process will then stop at the node whose next hop towards the sink is out of the overhearing range because it is in sleep mode. This is the data-forwarding interruption problem. In DMAC, the activity schedule of nodes on the multi-hop path are staggered to wake up sequentially. An interval is divided into receiving and sending the sleep periods. In the

receiving state, a node is expected to receive a packet and send an ACK packet back to the sender. In the sending state, a node will try to send a packet to its next hop and receive an ACK packet. In sleep mode, nodes turn off the radio to save energy. The receiving and sending periods have the same length, which is enough for one transmission and one reception. Based on the depth of the node in the tree, the node skews its wake-up scheme. Since nodes on a data path wake up sequentially to forward a packet to the next hop, sleep delay is reduced. Every node periodically turns to receiving, sending, and sleep states. When a node has multiple packets to send at a sending slot, it needs to increase it own duty cycle and request other nodes on the multi-hop path to increase their duty cycles. A more-data flag is added in the MAC header to indicate the request for an additional active period. The receiver checks the more-data flag and if the flag is set, it also sets the more-data flag. With this mechanism, DMAC can react quickly to traffic variations to be both energy efficient and maintain low latency. In order to avoid collision and interference, DMAC uses data-prediction schemes and More-To-Send packets.

A centralized MAC protocol based on TDMA was proposed in [31]. Initially all nodes have the same amount of energy and they wake up in their transmit schedule even when they don't have any data. A TDMA frame is constructed by the cluster head and is transmitted to all nodes in the network. This specifies when a node can transmit or receive a frame. The selection of the cluster head follows an innovative strategy of threshold-based reliability calculation. When the sensor nodes detect an event, information is forwarded from the sensor nodes to the cluster head, which aggregates the data and sends it to the sink. Cluster heads assign time slots to all nodes in the network. Nodes transmit the information received in their respective time slots. In order to balance between energy and latency, the wake-up schedule is used that will have a transition from TDMA approach to an On-Demand approach for critical information, which should be transmitted continuously so latency can

be reduced. In the wake-up phase, nodes consider data as critical by comparing them with previously transmitted information. When the data is critical, nodes send a wake-up tone signal, which will be recognized by all the nodes in the data path during the listen period. This helps in reducing latency for critical data.

In [34], a Dynamic TDMA protocol for Wireless Sensor Networks (SPARE MAC) has been proposed. In [34], time is organized in frames and each frame is divided in a signaling sub frame (SSU, consisting of N slots), a wake-up Slot (WS), and a DATA Sub Frame (DSU, consisting of M slots). Each sensor node must choose one of the N slots and will use this to broadcast control packets. The wake-up slot is used to send out tones to notify the neighboring nodes that they have to wake up during the next SSU. The DSU contains slots that are used for receiving data packets. Each sensor has to choose one or more of these and these slots form the Reception schedule of that node. To address the hidden terminal problem, SPARE MAC uses Wake-up Reliable Reservation Aloha protocol, according to which each sensor has to chose one of the control slots in the SSU and uses it for transmitting signaling packets. During SSU, a slot is declared free only if no one is reported using it within 2 hops while in DSU, and a slot is chosen if no other node at one hop has chosen it. To avoid the long delays caused by collisions, a dynamic technique to reserve reception slots is proposed. In this, the transmitters count the consecutive collisions and send a control packet when the threshold is reached. At the receiver, a rate-estimation algorithm is used to allow the receiver to know the number of active sources. Dynamic Bandwidth Adjustment in SPARE MAC makes it possible for any sender to dynamically change its own Reception Schedule by adding, removing, or changing some of its slots.

Providing prioritized treatment in continuous monitoring applications helps in improving data freshness. Based on this requirement, many techniques have been proposed to give

prioritized treatment while maintaining the network performance. The remote homing-based solution in [37] provides support for service differentiation in WSNs. Each source node has a local home and a remote home. It sends the normal traffic through the local home and critical information through the remote home. The remote home resides more than one layer away from the source in the multi-layered architecture. By sending data through the remote home, the transmission delay is reduced. When critical information is transmitted, certain layers in the multi-layered architecture are bypassed so that information reaches the sink with minimum latency. Reliability is addressed using dual remote homing. This is achieved by selecting the primary route and back up route, and sending the critical information through both these routes. This ensures that critical information reaches the sink. However, this increases congestion in the network.

Different mechanisms to provide differentiated services for time-critical information flows are explored in [33]. The first mechanism prioritizes the packet-transmission process at each node. When a high priority packet and low priority packet reach a node, the node first transmits the high priority packet. The second mechanism prioritized the access to the wireless medium. Once a channel has been sensed idle for a certain period of time, a node with high priority packets will transmit first. However, in both of these approaches, the reliability of the packets is not considered. It was also observed that a number of high priority packets were being dropped due to collisions. Hence, a third mechanism was used in which the channel from the source to the sink is reserved for the entire duration of the high priority flow. However, this approach has not been applied to a network where there are multiple event centers. A TDMS-based scheduling is presented in [18] to improve throughput in a heterogeneous wireless sensor network. The proposed scheduler is at a proxy level and thus is transparent to both the application layer as well as the MAC layer. The scheduler activates multiple stations simultaneously to maximize aggregate throughput

of the network. However, it is specific to an application and might not be applicable in general to all the applications.

An adaptive-forwarding scheme to provide service differentiation was proposed in [3]. The adaptive-forwarding scheme uses the multi-path and multi-packet forwarding mechanisms based on the packet priority. This paper assumes that packets of high priority have to reach the sink at a higher probability and minimum latency. In order to ensure this, the authors introduced the concept of multi-path forwarding and multi-packet forwarding. In multi-path forwarding, the high priority packets are transmitted along multiple paths and in multi-packet forwarding, the copies of high priority packets are transmitted. Both of these ensure high reliability. An adaptive-forwarding scheme combines both of these approaches to ensure high reliability. However, in this approach, the congestion increases in the network and there is an additional overhead caused due to multiple transmissions of the same packet.

A priority-based rate-control mechanism for service differentiation is proposed in [39]. The high priority real-time traffic is distinguished from low priority non-real-time traffic and the input traffic is serviced based on the priority. Four types of traffic are considered: real-time traffic, high priority non-real-time traffic, medium priority non-real-time traffic, and low priority non-real-time traffic. Separate queues are used for each traffic. Each packet has a traffic class assigned to it. Arriving packets are sent to different queues according to their traffic class and packets are transmitted according to their priority.

Multi-path routing is proposed in [11] to provide prioritized treatment to time-critical applications. The paths are classified during multi- path routing based on their route length and critical queries are routed through a set of paths with minimum route length. The rest of the traffic is spread with the objective of uniform node utilization in the network. For a given source-sink pair, multiple paths are classified based on their optimality, which is

determined by the number of hops taken by a packet on a path to reach its destination. The set of paths that deliver the packets with a delay less than or equal to that acceptable to time-critical applications are called critical paths and are used to route the critical query traffic. Though time-critical information is sent along the paths with minimum length, this paper [11] does not discuss their reliability.

PRIMP [19] is proposed for sensor networks to offer extended life time and robust network fault tolerance. PRIMP addresses the slow start up issue that occurred in data centric routing schemes. PRIMP uses the concept of virtual sources. When an interest from a sink flows down the network and reaches a neighbor, the virtual sources of the neighbor nodes are set as the four corners of the sub area in which the neighbor node resides. Interest flows from the neighbor to all of these virtual sources. These virtual sources forward the interest only when the node is nearer to the actual source. Otherwise, it is dropped. The drawback of this approach is it is not scalable.

Data freshness in sensor networks has also been approached from the concept of network security. Many protocols have been proposed with this goal. In [13], the authors proposed a secure data aggregation technique that can be used to transmit aggregated packets securely to the sink. The data sent from each leaf node is encrypted and the encrypted message is sent to the group leader. All of the sensor nodes are divided into groups and each group has a group leader. Once the group leader receives all the messages from the nodes, it aggregates them and forwards them. The messages sent from the leaf nodes are not decrypted by the group leader. They are only decrypted at the sink. This ensures that the packets are sent securely to the sink. Along with the message, each leaf nodes sends the time stamp to the group leader. The group leader know if the data is fresh based on the time stamp. If it finds the data to be old, it ignores it. This way data freshness is improved. The paper [13] mainly deals with security of the aggregated packets. The

data freshness is ensured by using a time stamp, which is an overhead because each time data is forwarded, the time stamp of the received data packet has to be checked. In [7], the novel application of body area sensor networks to monitor soccer players in a soccer field is presented. The authors outline the challenges in experimental data collection and elaborate on the design choices made. A multi-hop routing protocol that balances between resource consumption and delay has been proposed. In order to minimize the delay, the authors implemented an off-line flooding-based protocol where once a data sample is generated, it is continuously transmitted in every slot. Similarly, when a neighbor picks up the sample, it is retransmitted by the neighbor. Since every available node participates in the forwarding process, this mechanism is guaranteed to achieve the possible lowest delay at the cost of resource consumption. The authors next propose a random forwarding scheme that reduces the resource consumption at the cost of increased delay. The authors proposed a tunable scheme that allows for good delay performance and lower resource consumption. However, there is a chance of old data being transmitted within the network. To ensure data freshness, player j forwards data for player i only if the most recent sample in its window is less than A samples old. The experimental results show that the tunable scheme proposed achieves lower delay and resource consumption.

An important challenge to effective data delivery in wide area environments is maintaining the data freshness of objects using solutions that can scale to a large number of clients without incurring significant server overhead. Policies for maintaining data freshness are traditionally either push-based or pull-based. Push-based policies involve pushing data updates by servers; they may not scale to a large number of clients. Pull-based policies require clients to contact servers to check for updates; their effectiveness is limited by the difficulty of predicting updates. In [15], an adaptive pull-based solution to this challenge is proposed. Several techniques that use update history to estimate the freshness of cached

objects are presented in this paper and update patterns for which each technique is most effective are identified. Adaptive policies that can (automatically) choose a policy for an object based on its observed update pattern are introduced. Using the time the object was last modified, clients or caches can use TTL, a pull-based policy. TTL estimates how long an object remains fresh in the cache as a function of its last modification time. The aggregate history-based policy uses the aggregate history that is learned from the past updates to a set of objects. The individual history policy uses the individual history to estimate the freshness of a cached object. Experimental results prove that the IndHist and AggHist policies for cyclic objects significantly improve the accuracy of estimates of an object's freshness. In [4], analysis of the definitions of data freshness and metrics are presented and a taxonomy based upon the nature of data, the type of application, and synchronization policies underlying the multi-source information is proposed. The way freshness is defined is analyzed and presented. A data integration system is an information system that integrates data of different independent data sources and provides the users with a uniform access to the data by means of a global model. Data freshness comprises a family of quality factors, each one representing some freshness aspect and having its own metrics. Freshness is hence referred to as quality dimension. Two sub-dimensions of freshness are the currency factor, which captures the gap between the extraction of data from the sources and its delivery to the users, and the timeliness factor, which captures how often data changes or how often new data is created in a source. The metrics proposed for measuring data freshness are the currency metric, which measures the time elapsed since the source data changed without being reflected in the materialized view, the obsolescence metric, which measures the number of updates to a source since the data-extraction time, and the freshness-rate metric, which measures the percentage of extracted elements that are up to date. Systems that evaluate freshness are analyzed. Few such systems are data

warehousing systems, mediation systems, caching systems, and replication systems. In [2], energy-efficient security mechanisms for data integrity in body bio sensor networks are proposed. A security scheme that identifies attacks on data freshness and preserves message integrity is constructed. A body bio sensor network is a group of wireless sensor nodes used to measure biological parameters that can provide valuable medical information. In the applications of body bio sensor networks, it is critical to maintain confidentiality and privacy of sensitive medical information. Malicious capture of private data may cause harm to the patient. Few of the security challenges in the case of the bio sensor networks are authentication, confidentiality, freshness, and message integrity. Freshness guarantees that the information obtained from a node is recent and not a replay of old packets. Security threats that may be targeted at wireless bio sensor networks are data-freshness attack, and authentication attack. In a data-freshness attack, an adversary deliberately introduces a delay in the packet transmission or replays old messages to cause disruption in data aggregation. The authors proposed a solution to maintain data freshness by periodically calculating the round trip time delay of a packet. The base station relays a permissible RTT delay value to other nodes in the time field of a request packet. When the base station fails to receive a response in the permissible time period, network abnormality is suspected.

2.2 Reliability

Along with data freshness, reliable data delivery is very critical in sensor networks. WSNs consist of a large number of sensor nodes densely deployed in the areas of interest. On detecting the event, sensor nodes generate packets, and forward them to the base station or sink with the help of neighboring nodes. At the base station, these packets are processed and the necessary action is taken. This clearly shows the need for a reliable data transport

in WSNs. Many protocols have been proposed for providing reliable data delivery in sensor networks [1, 10, 27, 32, 35, 36].

Reliable Data Transport In Sensor Networks (RMST) [32] is proposed to provide reliability at the transport layer. This scheme adds reliable data transfer to directed diffusion. RMST is designed for delivering large blocks of data in multiple segments from a source node to a sink node. For example, this is required when time series data has to be transmitted. Reliability in RMST refers to the eventual delivery to all subscribing sinks of any and all fragments related to a unique RMST entity. A unique RMST entity is a data set consisting of one or more fragments from the same source. RMST does not include any real-time guarantees. In RMST, receivers are responsible for detecting whether or not a fragment needs to be re-sent. In the cached mode, the sink node and all intermediate nodes on an enforced path cache segments and check the cache periodically for missing segments. When a node detects missing segments, it generates a NACK message, which travels back to the source along the reinforced path. The first node *A*, having missing segments in its cache, forwards them again towards the sink (and thus towards the requesting node). If *A* can retrieve all requested segments from its cache, then *A* drops the NACK packet; otherwise, it is forwarded further upstream. Both the segments and the NACK packets are represented in terms of attributes, to be compatible with directed diffusion. In the non-cached mode of RMST, only the sink node has such a cache, but not the intermediate nodes; therefore, NACKs travel back to the source node (which clearly also needs to cache the segments). Some of the services offered by RMST are loss detection and repair: losses are detected at each node using a watch-dog timer. The timer handler sends a NACK for the missing fragments that have aged too long. Another important feature of RMST is a back channel at each node, which is a channel towards the source. This is used for sending NACK to the sender for missing fragments. RMST combines a NACK-based

transport-layer protocol with S-ARQ to achieve the best results. RMST does not guarantee reliability when nodes fail and does not address the congestion in sensor networks.

Witness-Aided Routing Protocol (WAR) [1] is designed to support unicast routing over both unidirectional and bidirectional links in ad hoc networks while maintaining low-bandwidth utilization and reliable-packet delivery. WAR is based on the notion of a witness host, which plays a central role in the routing and recovery process. The neighborhood of a mobile host is a union of two sets: the incoming and the outgoing neighborhood. The incoming neighborhood of host *A* is the set of mobile hosts one hop away from *A*, whose transmissions it can hear. Similarly, the outgoing neighborhood of *A* is the set of mobile hosts on each hop away from *A*, which can hear its transmissions. The two neighborhoods are the same if the links are bidirectional. A witness is a host that can overhear a transmission that was not destined to it. Thus, all witness hosts of *A* are members of *N out(A)*. When a witness host is also a member of *N in(A)* and *W* belongs to *N out(A)*. WAR is a reactive protocol and has three major components: route discovery, packet forwarding, and route recovery. The discovered route is invoked by a source host *S* every time it needs to route to a destination host *D*, and it does not have one already cached. Host *S* broadcasts a route-request message and starts a local timer to decide when to re-send the request in case no response arrives. All hosts in *N out(s)* hear the request and they replicate it so the route-request message propagates until it reaches *D*. Since the route request is a broadcast, it is likely that it arrives at the destination on many different paths. *D* will choose one of these and will send a route reply message to *S*. WAR uses source routing in order to deliver packets from a source host to it, so at each intermediate host, the packet contains information about the next hop in the route. Before *A* delivers a packet to the next host *B* in the route, it removes its id from the list of remaining hosts. A packet is considered to be successfully delivered if *A* receives a passive acknowledgment from

B or if *A* receives a positive acknowledgment from any of its witness hosts. Otherwise, *A* assumes that the route is broken and initiates a route discovery procedure. During the route-recovery process, WAR attempts to quickly and inexpensively bridge the gap created in the route. This would allow the packet to travel to its destination as opposed to delaying it until *S* finds another route.

Directed diffusion [10] aims to optimize robustness, scalability, and energy efficiency for data-centric network interests. Events that are to be recorded are propagated by sink nodes throughout the network via flooding. These initial requests for information stipulate that responses should be sent at a low data rate. Upon receiving responses from multiple sources, a sink can use positive and negative reinforcement to increase and decrease gradient (*i.e.,* paths) data rates, respectively. Reinforcement mechanisms can vary; this paper uses interest messages with modified interval values for this purpose. Rules governing the propagation of control messages are executed locally; that is, each node decides individually without a global state how it will react to control messages. The adoption of a hop-to-hop paradigm obviates the need for complex routers and universally unique identifiers. Additionally, it can result in highly adaptive networks, in which broken links can be avoided automatically according to local policies. In *NS*2 experiments, the authors found that directed diffusion dissipated less energy than both flooding (a watermark) and omniscient multicast (a representation of attainable performance with traditional end-to-end architectures). Additionally, it incurred comparable delay to omniscient multicast, and substantially out-performed flooding. Two particularly important performance-enhancing factors of directed diffusion are negative reinforcement, which allows for the pruning of superfluous gradients, and duplicate suppression, which takes advantage of the incorporation of application layer semantics in the communication protocol (made possible by directed diffusion) to avoid transmitting redundant messages.

Event-driven sensor networks rely on the collective effort of a number of sensor nodes. Reliable event detection at the sink is based on the collective information provided by the sensor nodes and not on any individual report. To achieve event-to-sink reliability, [27] has been proposed. ESRT seeks to achieve reliable event detection with minimum energy expenditure and congestion control. It has been tailored for use in sensor networks with adaptability to dynamic topology, collective identification, energy conservation, and biased implementation at the sink. Reliability is measured as the number of data packets received at the sink. To measure reliability, the concept of observed event-level reliability and desired event-level reliability have been introduced. Observed event reliability (ri) is the number of packets received in decision interval i at the sink. Desired event reliability R is number of data packets required for reliable event detection. If the observed event reliability is greater than desired reliability, the event is deemed to be reliably detected. Otherwise, appropriate actions have to be taken to achieve this reliability. The reporting rate of a sensor node is defined as the number of packets sent out per unit time by that node. ESRT configures the reporting frequency f such that the desired event detection accuracy is achieved with minimum expenditure. Five different characteristic regions have been identified based on reporting frequency f, r, and R. The five states are (No Congestion, Low Reliability), (No Congestion, High Reliability), (Congestion, High Reliability), (Congestion, Low Reliability), and (Optimal Operating Region). The motive of ESRT is to maintain operation in Optimal Operating Region. The network can reside in any one of these states. Depending on the current state (Si) of the network, ESRT finds the updated reporting frequency (Fi+1) and broadcasts it to all the source nodes. If the observed event reliability is less than the desired reliability, ESRT increases the reporting frequency of the nodes. If the observed reliability is more than the desired level, the reporting frequency is decreased to avoid congestion and reduce the energy consumption. To detect the current

state of the network, the sink must be able to detect congestion in the network. The sensor nodes detect congestion using the buffer sizes and set the congestion notification bit. Once the sink receives a packet with this bit set, it knows that congestion will take place and will update the reporting frequency accordingly.

However, with in-network data aggregation in place, packet-level reliability is mandatory because aggregated packets contain a lot of information and loss of one packet results in loss of a lot of data. In order to address this issue, a sensor-to-sink reliable transport protocol is proposed in [36], which is suitable for data aggregation and provides reliable upstream packet delivery by configuring the inactive nodes as monitors. Monitors assist in reliable loss detection and recovery in case of sudden node failures and congestion. The aim is to have a minimum set of nodes acting as monitors for reducing the energy consumption and also to keep all the nodes in the data path monitored. The configuration of monitors is done using a distributed heuristic that requires only one-hop information. This heuristic activates a minimal set of monitors when the flow starts and deactivates them when the flow stops. Each node is assigned a rank, which is initialized to zero initially. The rank of the node is incremented when nodes on the data path forward a rank-increment message. Similarly, after a flow is finished, the nodes decrement their rank. In the monitor-configuration process, if the data packet is received at a node and there is no monitor configured for that node, the node will broadcast a monitor-request message to its one-hop neighbor with the highest rank, which replies with a monitor-agree message. The node will record its monitor and will inform all of its other neighbors of its monitor. The neighbor nodes, once they receive this message, will update their rank and monitor accordingly. In case of packet losses due to congestion, retransmission to the same forwarder might not be successful. Hence, the retransmitted packets will be sent to the monitor node. The monitor will then choose a new route altogether to forward the packet.

This will ensure reliable packet delivery.

As nodes in WSNs are battery operated, it is not practically possible to replace the batteries frequently. Several techniques have been proposed to reduce the energy consumption [6, 8, 9, 20–22, 26, 28, 29, 38, 40–42]. Data aggregation and data compression are few of the techniques used for increasing energy efficiency. In [29], the authors proposed the cascading effect, which is a mechanism to implement data aggregation. One way of performing power-efficient data collection in sensor networks is to process data as it flows from information sources to sinks. This technique is called in-network data aggregation and is very useful in reducing the energy consumption. To implement in-network data aggregation, the authors proposed the cascading effect, which defines when to clock out data as it is aggregated by nodes on its way to the sink.

In [26], authors proposed a mechanism to reduce the energy consumption in continuous-monitoring and event-driven sensor networks. The number of data packets transmitted are reduced with the help of data compression. To further achieve energy efficiency, all the nodes in an event center are formed into a separate cluster. This clustering mechanism helps in improving data suppression, thereby reducing transmission of redundant data. Coding by ordering scheme is used for data compression. The basic idea of this scheme is to accommodate the order of data gathered by an intermediate node in such a way as to omit the explicit transmission of a certain number of packets. The NP/CSMA technique is used in the cluster-formation phase. For the continuous-monitoring application, all nodes in the event center transmit data using the LEACH protocol. The LEACH protocol groups sensors into clusters and rotates the cluster head to balance the energy consumption. All the cluster members transmit data to the cluster head at a predefined frequency using a TDMA scheme. At the end of each, TDMA frame-data compression is applied at the cluster head depending on the number of packets received from the cluster members and cluster head transmits

the packets directly to the sink. In addition to the clustering mechanism, transmission of irrelevant data is avoided at each node. Each sensor node stores its previously transmitted data and compares the new data with this previous data. The new data is transmitted only if there is a mismatch. Though this protocol achieves energy efficiency, it does not work when there are events occurring frequently. Also, relevant data transmission might not be useful when the same information is transmitted alternatively.

Our approach is different from the above mentioned techniques, in that we improve data freshness along with packet-level reliability. In continuous-monitoring applications, such as equipment monitoring in hospitals, fresh data is required periodically to analyze the machinery status and take necessary and preventive action when required. Data freshness along with reliable data delivery is critical in these applications. We adapt the concept of monitors used in [36] to improve reliability and use concepts of prioritization and rate control to improve data freshness.

CHAPTER 3

DATA FRESHNESS AND RELIABLE DATA DELIVERY

3.1 Motivation and Design Considerations

Wireless Sensor Networks (WSNs) can be used to achieve either continuous monitoring or event detection in the area of interest. In continuous-monitoring applications, each sensor node transmits its sensed data to the sink node periodically while in applications driven by event detection, nodes report to the sink node once an event occurs. Continuous-monitoring applications require periodic refreshed data at the sink nodes. For instance, an equipment-monitoring application in hospitals requires fresh data periodically to analyze the machinery status and take necessary and preventive action when required. Data freshness is critical in these applications. However, resource constraints and wireless errors pose a major challenge in achieving data freshness along with reliable data delivery of packets in sensor networks.

It is important in continuous-monitoring applications for the data to reach the sink node within a certain threshold so that appropriate decisions can be made on fresh data. However, data freshness cannot be improved without providing reliability. WSNs are generally organized in a multi-hop topology. Since messages travel multiple hops, it is important to have a high reliability on each link. When techniques such as data aggregation are used to minimize the energy consumption, packet-level reliability is very critical as the correlated data packet contains a lot of information and loss of a single packet would result

in a huge amount of data loss. Hence, we need protocols that provide data freshness and reliability in such applications.

Many protocols provide event-level reliability for the upstream traffic. One such protocol is Event-To-Sink-Reliabile Transport Protocol (ESRT) [27]. ESRT employs a rate control mechanism for congestion avoidance. However, data freshness may not be guaranteed in ESRT as the congestion-avoidance mechanism depends on the state of the network for improving reliability. The network state in ESRT is identified with the buffer overflow and the congestion bit, which in turn adjusts the reporting rate of the sensor nodes accordingly. The correct state of the network might not be identified in scenarios where the packets could be dropped due to channel access failures; the buffer might not have excess packets to indicate congestion in the network. Also, since ESRT only deals with event-level reliability, the packets that carry the congestion bit could be dropped. There is no additional mechanism in ESRT that will ensure that these packets will reach the base station reliably. If these packets are dropped, the base station might not know the correct state of the network, which may not guarantee data freshness. In our protocol, improving reliability and data freshness does not depend on the state of the network. Packet-level relibility could be improved in our protocol by the additional infrastructure of monitors [36], which mitigate congestion and collisions in the network. Data freshness could be improved in our protocol by prioritizing the flows and adjusting the reporting rate based on the feedback from the base station.

In our protocol, the base station identifies the areas or sources from which it has not received a sufficient number of packets as desired by the application for every processing interval. Only the packets that reach the base station within a certain threshold will be considered. Sources from which the base station has not received the expected number of packets become the priority sources and the packets generated by these sources become the higher priority packets during a processing interval. We expect that the proposed protocol

increases the source recovery and reduces the recovery time. So far data freshness was measured only in terms of latency or delay of packets received at the base station [30]. However, improving the overall delay does not necessarily improve data freshness. We studied the effectiveness of the proposed protocol in terms of average delay of higher priority packets, the number of higher priority packets that have reached the base station within a certain interval, the percentage of sources that have recovered, and the average interval for the sources to recover in addition to the average delay of the packets received at the base station.

3.2 Assumptions

For ease of explanation, we make the following assumptions in the proposed protocol:

(a) The network is densely deployed, does not have any holes, and its outer boundary is identified.

(b) The nodes can be divided into active and inactive sets such that the network is connected and there is a connection between every pair of active nodes.

(c) All the nodes know the status of their one-hop neighborhood node (active/inactive) by local broadcast mechanisms.

(d) The processing interval and the expected number of packets required at the base station are identified.

In the following sections, we explain our protocol design in detail. We first describe the initial set up, which explains the formation of the data-gathering tree where every node configures a forwarder for itself and there is path from every source node to the base station. We then explain the monitor-configuration process where an inactive node is configured as

a monitor for a link between two active nodes. We augmented the monitor configuration proposed in [36] to suit the requirements of our protocol. In the proposed protocol, the inactive node with the minimum number of active nodes in its one-hop neighborhood is selected as the monitor. This helps in reducing the average latency by minimizing contentions and collisions. Finally, we elaborate on prioritization and rate-control mechanisms to improve data freshness.

3.3 Monitor Configuration for Reliable Data Transfer

3.3.1 Initial Setup

In multi-hop wireless sensor networks, sensor nodes forward the sensed information towards the base station with the help of a forwarder. The forwarder can be chosen dynamically or statically. In our approach to improve reliability and data freshness, we select the forwarder statically. The initial process of selecting forwarders consists of the selection of active and inactive nodes and the formation of data-gathering tree. Initially, we select a subset of nodes as active nodes randomly such that the network is connected and there is a path from every active node to the base station. The remaining set of nodes will be the inactive nodes. This careful division of nodes into active and inactive will ensure that there are no holes in the network. Choosing only a subset of nodes will also increase the energy efficiency.

The process of forming of a data-gathering tree is initiated by the base station. The base station broadcasts a *FORWARDER-REQ* message. All the nodes that receive the broadcast message will set the base station as their forwarder and will also set their hop distance as one. All the active nodes that receive this message include their hop distance and will relay this message to all the nodes away from the base station. Inactive nodes that receive

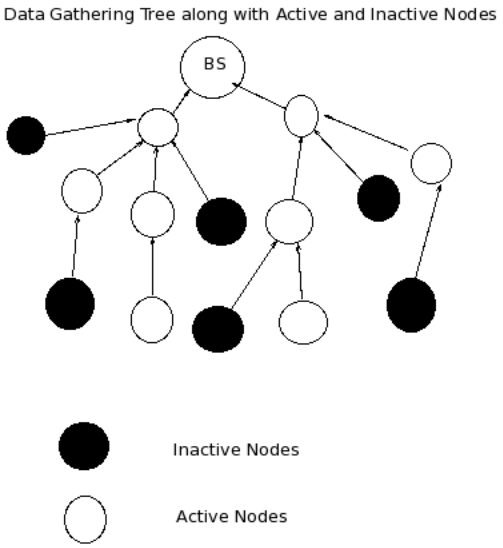


Figure 3.1: Data-gathering tree

this message will only set the base station as their forwarder and set their hop distance from the base station. This process continues until all the nodes in the network configure their forwarder. Since the *FORWARDER-REQ* message is a broadcast, there is a possibility that a node could receive the *FORWARDER-REQ* message even after it has configured its forwarder. In this scenario, the node resets its forwarder if the source of the new message is closer to the base station than its forwarder. This ensures that the forwarder is the node closest to the base station. Figure 3.1 shows the data-gathering tree along with the active and inactive nodes.

3.3.2 Monitor-Configuration Process

After the formation of the data-gathering tree, every node has a forwarder configured. A set of monitor nodes is chosen from the set of inactive nodes. The link between every pair of active nodes will be monitored by an inactive node configured as the monitor. To reduce the collisions and ensure that every link has a monitor, the monitor-configuration process

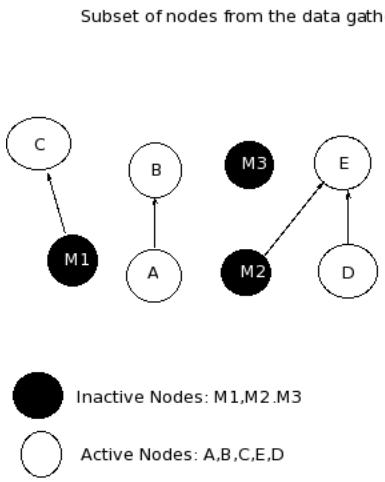


Figure 3.2: Subset of nodes from the data-gathering tree

is done in three stages. In the first stage, all the nodes at even hop distance from the base station configure the monitor. During the second stage, all the nodes at odd hop distance from the base station complete the monitor-configuration process. To ensure that every link has a monitor, in the final stage, all nodes without a monitor configured in the first two stages will initiate the monitor-configuration process.

The process of monitor configuration for a link is as follows. In a link, the node that is farthest from the base station sends a *NEIGHBOR-REQ* message to its forwarder. The forwarder replies with a *NEIGHBOR-REPLY* message including its inactive one-hop neighbors. The sender node upon receiving this message from its forwarder, forms the set of common inactive neighbors between itself and its forwarder. It then sends an *ACTIVE-NEIGHBOR-REQ* message to all the common neighbors. The common neighbors then reply with an *ACTIVE-NEIGHBOR-REPLY* message including their forwarder and the number of active neighbors. The sender node waits for a reasonable amount of time based on the number of common neighbors and then chooses the common neighbor with the minimum number of active nodes in its one-hop neighborhood, and also ensures that the forwarder of

the common neighbor is not itself nor its forwarder. Additionally, it also ensures that the common neighbor has at least three active nodes in its one-hop neighborhood. It then sends a *MONITOR-REQ* message to the chosen node. The selected node could receive a number of monitor-request messages. It waits for a reasonable amount of time, and then sends back a *MONITOR-AGREE* message to a node by selecting the node at random. The node on receiving the acceptance message, sends a *MONITOR-NOTIFICATION* message to all of its one-hop neighbors. Active neighbors and inactive neighbors, other than the configured monitor node that receive the *MONITOR-NOTIFICATION* message, ignore the message. When the inactive node selected as monitor receives the *MONITOR-NOTIFICATION* message, it checks if the message contains information on acting as monitor. If so, it registers the nodes that it has to monitor and sends an *ACK* message to the sender node to inform it about its confirmation as monitor. When the originating node receives the *ACK* message from the monitor node, it registers the monitor for the link between itself and its forwarder. The process will continue for all the active nodes in the network, which results in all the links between active nodes being configured with a monitor node. In case of packet losses, monitor nodes would re-route the packet to its forwarder, which is in a different data path.

The selection of the monitor node based on the minimum number of active nodes in its one-hop neighborhood helps in reducing the average latency. A monitor is configured to monitor only one link. Though this increases the number of nodes selected as monitor, it reduces the delay for the monitor node while forwarding the packets. Additional monitors are not configured to support the existing monitors as this increases the overhead. Also, nodes that are one hop away from the base station do not configure monitors because these nodes have the base station as their forwarder and it is not necessary to have a monitor.

For example, consider the subset of nodes in Figure 3.2. Figure 3.3 illustrates the monitor-configuration process for the link between *A* and *B* in the subset. From the figure,

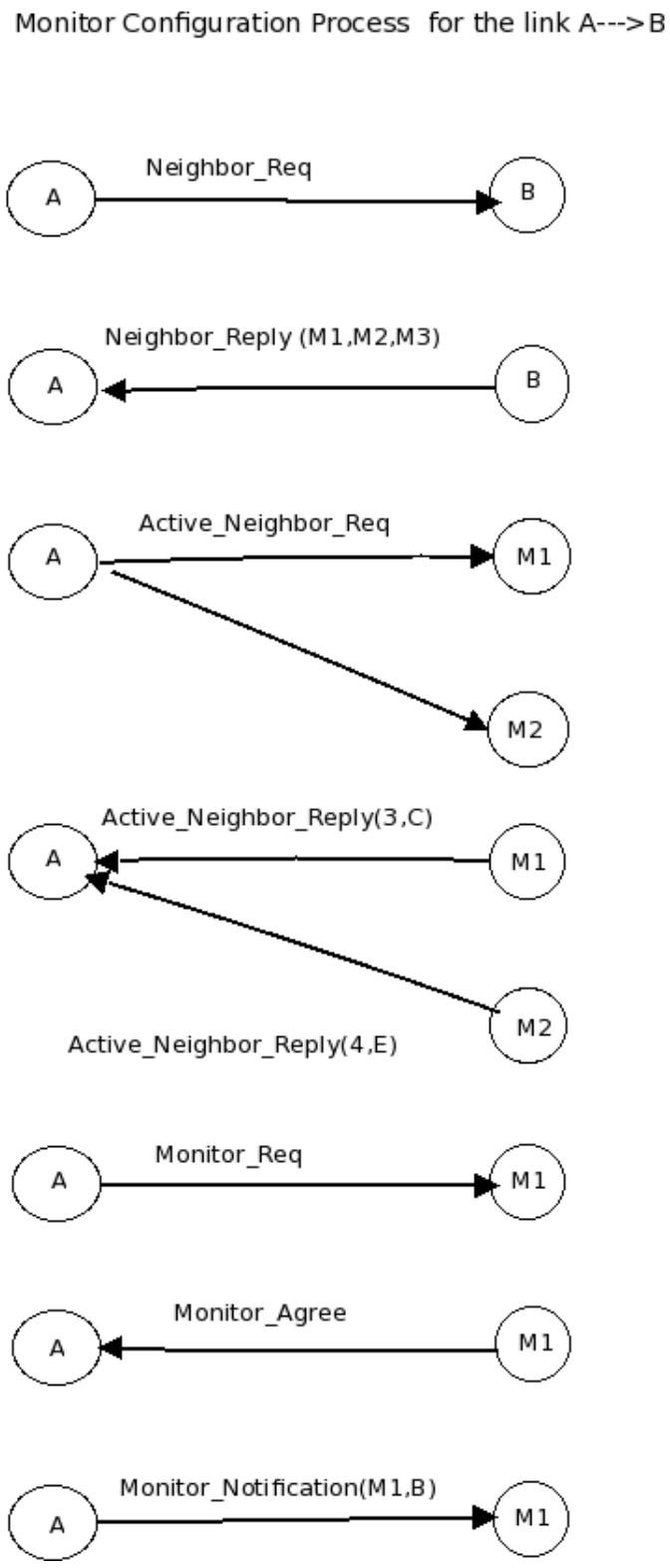


Figure 3.3: Monitor Configuration

it can be observed that $M1$ and $M2$ are the common inactive neighbors for A and B. $M1$ is selected as the monitor since it has the minimum number of active nodes in its one-hop neighborhood compared to $M2$ and its forwarder is neither node A nor node B. After choosing node $M1$ as the monitor, node A completes the monitor-configuration process with a *MONITOR-NOTIFICATION* message. Similarly, every node configures a monitor for itself and its forwarder. In the final stage, the monitor is selected as the common inactive neighbor and a node for which the *MONITOR-REQ* message has not been sent before. This helps in increasing the probability that the link gets a monitor in the final stage.

3.3.3 Monitor Functionality

Consider a link between two active nodes A and B. Let $M1$ be the monitor node for the link $A \rightarrow B$. Whenever node A transmits a packet to node B, the monitor node caches that packet in its buffer. It then waits for a reasonable amount of time for node B to transmit that packet. When node B transmits that packet, the monitor node overhears that transmission as it is also the one-hop neighbor of node B. It then removes the packet from its cache as it knows that the packet has been successfully transmitted from node B. If the packet transmitted by node A does not reach B due to collision or congestion, or node B is unable to transmit the packet to its forwarder, the monitor node $M1$ assumes that the packet is lost and it retransmits the packet to its own configured forwarder, which is in a different path altogether. Before the monitor node $M1$ transmits the packet, it waits for a reasonable amount of time for node B to transmit the packet. This way, the monitor nodes help in rerouting the dropped packets and increasing the reliability. However, if for some reason the monitor node $M1$ does not overhear the transmission of the packet from node B after node B successfully transmitted the packet, it incorrectly assumes that the packet has been dropped and retransmits the packet, resulting in duplicates.

As mentioned above, the monitor node helps in the reliable delivery of packets. However, they can only reroute packets when the packets are received by the monitor. In MAC 802.11, every node, before it actually transmits the data, tries to get the channel access. The four-way handshake between a node and its forwarder is as follows. Initially, when node *A* wants to transmit a packet, it sends a ready-to-send (RTS) message, informing the forwarder that it is ready to send data and checks if the forwarder is ready to accept data. The forwarder node *B* sends a clear-to-send (CTS) message informing node *A* that it can send data. Node *A* then sends the actual data to node *B*. Upon successful reception of data, node *B* sends an ACK message to node *A*. If for any reason node *A* does not get the CTS from node *B*, it retransmits the RTS message a maximum number of times and then drops the data packet. If the packet drop is due to channel-access failure, then node *A* directly transmits the RTS message to the monitor node instead of dropping the packet. If the monitor node replies with a CTS message, node *A* transmits the actual data to the monitor node directly instead of its forwarder. This helps in reliable delivery of the packet.

3.4 Mechanisms for Improving Data Freshness

In continuous-monitoring applications, such as equipment monitoring in hospitals, fresh data is required periodically to analyze the machinery status and take necessary and preventive action when required. Data freshness along with reliable data delivery is critical in these applications. Our protocol aims at improving data freshness and seeks the information from the sensor nodes whenever required. Data freshness can be improved by prioritizing the flows based on the feedback from the base station and adjusting the reporting rate of the sources.

3.4.1 Overview of the Protocol Stack and Transmission Process

The protocol stack for a sensor node consists of the application layer, the transport layer, the network layer, the link layer (LL), an ARP module connected to the link layer, an interface priority queue (IFq), the MAC layer, and a network interface (netIF), all connected to the channel. For all outgoing (into the channel) packets, the packets are handed down to the LL by the Routing Agent. The LL hands down packets to the interface queue. The MAC layer dequeues the packets from the interface queue and transmits the packet to the physical layer. When the MAC layer receives an acknowledgement that the packet has been successfully transmitted or the packet has been dropped, it dequeues the next packet from the queue for transmission. This is the normal process of transmission of packets from the network layer to the physical layer. At each and every layer of the protocol stack, header information is added to the packet at every layer by the sending node. At the receiving node, each layer unwraps its own header part from the packet, performs its functionality, and passes the packet to the upper layers. Each layer can access only its own header of the packet. When the receiving node knows that it is not the destination of the packet, it passes the packet to its forwarder following the same procedure.

Feedback Message

The source nodes keep transmitting the sensed information periodically to the base station at a given rate. The base station identifies the areas or sources from which it has not received a sufficient amount of information for every processing interval. The base station then classifies the sources into three categories: sources from which it received more information, sources from which it received enough information, and sources from which it received less information. Next, the base station sends this information to the sources in

the network through broadcast messages. Broadcasting reduces the delay but increases the network traffic. To minimize the downstream traffic and prevent packet drops due to the downstream traffic, each node re-broadcasts only once and duplicates are ignored.

3.4.2 Prioritization Process

We assume that the base station has the capability to know if it has received sufficient information from the sensor nodes or not. When a source node receives the feedback message from the base station, it checks if it is among the priority sources from which the base station did not get enough information. If it is, the source node then sets the priority of the packets it is generating to high in the link layer header of the packet. Since the priority is set in the header of the link layer, when the packet is inserted in the interface queue to be sent to the MAC layer for transmission, it can be sent based on its priority. Once a source node sets the priority of the packet to high, every other node in the path of the packet to the base station will forward this packet considering it to be a packet of higher priority. At each node, packets go to different queues based on their priority. While dequeueing, the queue with the highest priority is served first followed by the queue with lower priority. This ensures that packets with higher priority are transmitted first. The same process takes place at every node, including the monitor nodes. Thus, prioritization could help in improving data freshness.

3.4.3 Rate-Control Mechanism

To further improve data freshness in our protocol, we augment the well-known rate-control mechanism to suit the requirements of our protocol. The source nodes keep sending the information to the base station at a fixed-packet interval rate. The base station has a processing interval after which it checks if it has received enough information from every

source node. Based on the information it has received, it sends feedback in the form of a broadcast. When the source nodes receive feedback, they will send additional packets to the base station until the next processing interval if the feedback indicates that the base station has not received enough information from them. On the contrary, if the feedback indicates that the base station has received more information from a source than required, the node sends fewer packets during that processing interval. If the feedback indicates that the base station has received sufficient information, the source node continues to send the same number of packets as earlier. This helps the sources to recover faster and could potentially improve data freshness. Consider a scenario where the base station did not receive enough information from a node during two consecutive intervals. When the node sends more packets to the base station for two consecutive intervals, it will increase the traffic in the network. Also, if sending more packets did not help in getting enough information to the base station in one processing interval, it can be inferred that this did not help the node to recover. In this scenario, sending more packets again does not make sense, instead we reduce the traffic from all other nodes in the network(i.e, we reduce the overall network traffic). This could increase the source recovery and reduce the recovery time.

Algorithm1 is the pseudocode of the functionality at the base station.

For every processing interval, the base station checks if it has received the expected number of packets from every source node or not. Accordingly, it adds the source node to the list of priority sources if the information received from the node is less than the expected number of packets. If the information received is more than the expected number, it adds the node to the list of non-priority sources, and if the information received is equal to the expected number, it checks if it has to reduce the overall network traffic. If so, it adds the node to the list of non-priority sources and otherwise ignores it. The base station then sends this information to all the sources through broadcast messages.

Algorithm 1 Functionality at the base station

Let x be the expected number of packets required at the base station for every processing interval.
srcCnt is the total number of the sources in the network.
InfoReceived is the information received at each processing interval.
$Cnt \leftarrow x$
while $ProcessInterval \leq lastProcessInterval$ **do**
  if $node = BaseStation$ **then**
    $HasIncreasedReportingRate \leftarrow false$
    for $i = 0$ to $srcCnt$ **do**
      Check if enough information is received from source or not
      if $InfoReceived < Cnt$ **then**
        Add the node to list of PrioritySources
      else if $InfoReceived > Cnt$ **then**
        Add the node to list of NonPrioritySources
      else if $InfoReceived = Cnt$ **then**
        If there is a priority source for two consecutive intervals
        Add the node to list of NonPrioritySources
      end if
    end for
    SendFeedback Message including the above information.
  end if
end while

Algorithm2 is the pseudocode of the functionality at the source nodes.

Once a source node receives the broadcast message from the base station, it checks if it is among the list of priority sources. If it is, it sets the priority of the packets it is generating to high and sends more packets during that processing interval. However, the node will not send more packets in two consecutive intervals. If it has sent more packets in the previous interval, it will not send in the next interval again. If the node is among the non-priority sources, it will send fewer packets or the same number of packets to the base station during that interval.

Algorithm 2 Functionality at the source nodes

```
HasIncreasedReportingRate ← false
IncreaseReportingRate ← false
DecreaseReportingRate ← false
if node = SourceNode then
  if FeedbackReceived then
    if node is among the PrioritySources then
      SetPriority ← true
      if HasIncreasedReportingRate = true then
        IncreaseReportingRate ← false
        HasIncreasedReportingRate ← false
      else
        IncreaseReportingRate ← true
      end if
    end if
    if node is among the NonPrioritySources then
      SetPriority ← false
      DecreaseReportingRate ← true
    end if
  end if
  if IncreaseReportingRate = false and DecreaseReportingRate = false then
    send same number of packets as before
  end if
  if IncreaseReportingRate = true and DecreaseReportingRate = false then
    HasIncreasedReportingRate ← true
    send additional packets
    IncreaseReportingRate ← false
  end if
  if DecreaseReportingRate = true and IncreaseReportingRate = false then
    send fewer packets
    DecreaseReportingRate ← false
  end if
end if
```

CHAPTER 4

PERFORMANCE EVALUATION

To evaluate the performance of our protocol, extensive simulations have been conducted using ns-2 [23] simulator. To evaluate the effectiveness of the mechanisms to improve data freshness in our protocol, we compare our protocol with the base network, which uses monitors but has no priority queue and no rate-control mechanism incorporated. To study the performance of our protocol with the prioritization and rate-control mechanism, we evaluate our protocol in two stages: one having prioritization and no rate-control, and the other having both prioritization and rate-control. To our knowledge, current protocols measure data freshness only in terms of latency or delay of all packets received at the base station [30]. However, improving the overall delay alone does not necessarily improve data freshness. In continuous-monitoring applications, a base station needs to get fresh data from all the sources periodically and the sources from which the information was missing need to recover quickly. Increasing the number of sources that can recover and reducing the recovery time helps in improving data freshness. Hence, we identified the following metrics to study the performance of our protocol in addition to the average delay of all packets received at the base station

(a) Average delay of higher priority packets.

(b) The number of higher priority packets that have reached the base station within the identified freshness interval.

(c) The percentage of sources that have recovered.

(d) The average interval for the sources to recover.

For every processing interval, the base station expects a sufficient number of packets from all the sources as desired by the application. Only the packets that reach the base station within the identified freshness interval will be considered. Sources from which the base station has not received the expected number of packets become the priority sources and the packets generated by these sources become the higher priority packets for the next processing interval. We expect that the proposed protocol increases the source recovery and reduces the recovery time. The following are the metrics we identified to measure the freshness of data in addition to the average delay of the packets received at the base station.

- Average Delay Of Higher Priority Packets: This metric indicates the average delay of the higher priority packets received at the base station. Higher delay implies that the prioritization and rate control did not help in improving the data freshness while lower delay indicates that the protocol employed improved the data freshness.

- Freshness Ratio: This is the ratio of the total number of higher priority packets received at the base station within the identified freshness interval over the total number of higher priority packets received at the base station. Lower freshness ratio implies that the higher priority packets could not be received at the base station as required by the application.

- Recovery Percentage: This metric represents the total number of priority sources that have recovered over the total number of priority sources. This metric reflects how the network is recovering. Any source could become a priority source irrespective of whether it recovered or not, and a non-priority source could become a priority

source anytime due to the nature of sensor networks and traffic conditions. When the expected number of packets at the base station is more, sources could become priority sources more than once due to congestion. As the expectation reduces, the number of priority sources reduces. Recovery percentage could vary when the expected number of packets at the base station varies.

- Average Recovery Interval: This is the average of all intervals taken by the priority sources. This metric reflects how quickly the network has recovered. If the recovery interval is high, it means that the technique being evaluated is not responding quickly.

The metric used in the previous work [30] to measure data freshness is insufficient as it mainly focuses on the delay of all packets received at the base station. It does not capture the requirements of continuous-monitoring applications in terms of the base station getting sufficient fresh data from all the sources and how quickly the sources are able to recover in case of missing information. This entails the need for an additional set of metrics to capture the demands of such applications. By examining the usefulness of the metrics identified, we can conclude that the additional metrics along with the previous metric provide a comprehensive view of the system's performance and an accurate baseline for comparison.

4.1 Simulation Setup

The simulations were run with the standard simulation parameters [36] as mentioned in Table 4.1. A subset of 30 nodes were considered to be active nodes and the remaining nodes were considered as inactive nodes. For data packets, Constant Bit Rate (CBR) traffic is generated. In all the experiments, each data point taken is an average of 20 independent runs. The simulations were run for a total simulation time of 350 sec.

Table 4.1: Simulation Parameters

Parameter	Value
Area	$1000m$ x $1000m$
Transmission radius	$250m$
Total number of nodes	150
Number of sources	20
Data Packet size (bytes)	516
Packet interval rate	$0.1(5KB/s)$ to $0.35(3.75KB/s)$

We identified the freshness interval to be four processing intervals for our simulation to give a sufficient amount of time for the packets to reach the base station even in highly congested scenarios. We used a processing interval of 3 sec for our simulations and the number of processing intervals were 17. We varied the total number of packets required at the base station (75%, 50%, and 25% of the packets sent) to reflect the demands of such applications. The packet interval is varied from 0.1 sec to 0.35 sec when the expectation at the base station is 75% of the packets sent; our simulations showed that at lighter loads (packet interval greater than 0.35), no additional mechanism is required to deliver data freshness. Similarly, the packet interval is varied from 0.1 sec to 0.3 sec when the expectation is 50% and 25% of the packets sent. For rate control, we increased the reporting rate to 70% of the total number of the packets required at the base station and decreased the reporting rate to 50% of the additional packets sent. We conducted experiments by varying the increase rate and decrease rate and these were found to be the optimal values. Increasing the reporting rate by a value more than 70% introduces additional traffic, worsening the performance of the protocol, and increasing the rate by a value less than 70% does not help in increasing the performance. Similarly, decreasing the reporting rate by a value more than 50% triggers a chain reaction, degrading the performance, and decreasing the rate by a value less than 50% does not increase the performance. Similar behavior can be seen

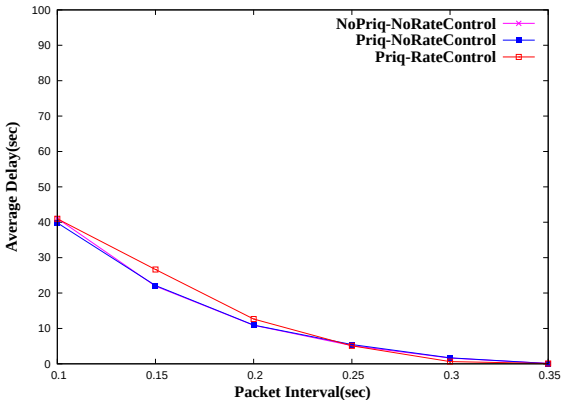
with only increase and only decrease. The increase and decrease in the reporting rate is the same when the expected number of packets is 75%, 50%, and 25%.

4.2 Simulation Results

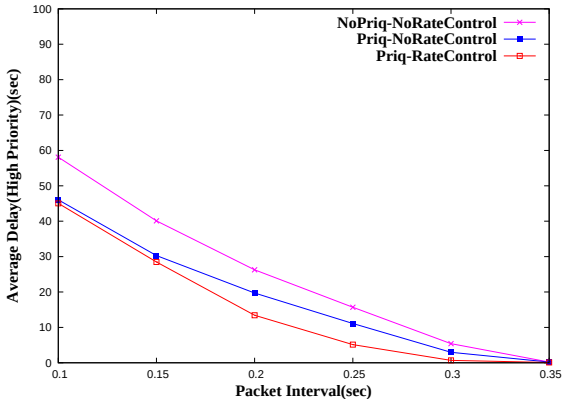
We evaluated the performance of the proposed protocol with different expectations at the base station. We first show the performance when the expectation at the base station is high (75%). We plotted all the graphs for all the metrics by varying the packet interval to show how the proposed protocol performs with different traffic loads.

The average delay of packets received at the base station is shown in Figure 4.1(a). The average delay of the packets received is almost the same in all three cases. The decrease in delay of higher priority packets leads to an increase in the delay of lower priority packets and vice versa, resulting in the average delay being the same in all the three cases. However, the slight increase in average delay with rate control for higher traffic loads is due to the slight increase in the overall traffic in the network due to rate control.

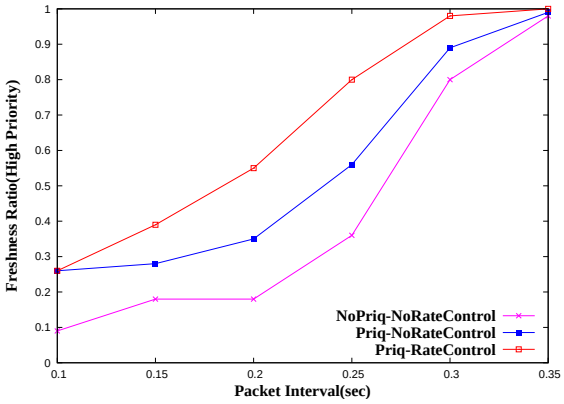
The average delay of higher priority packets is reduced with the proposed protocol as shown in Figure 4.1(b). At higher traffic loads, the delay of higher priority packets is more for all the three cases when compared to the delay at lower traffic loads. At higher traffic loads, the network becomes congested, resulting in delay and an increase in packet drops. Also, since the total number of packets required at the base station (75 percent of the packets sent) is high, the probability of the base station getting a sufficient amount of information from every source node decreases in such congested scenarios, increasing the number of higher priority packets. When a rate-control mechanism is employed, an additional amount of traffic is introduced in the network. Even with this increase in traffic, the average delay of higher priority packets is lower than the case when there is no rate



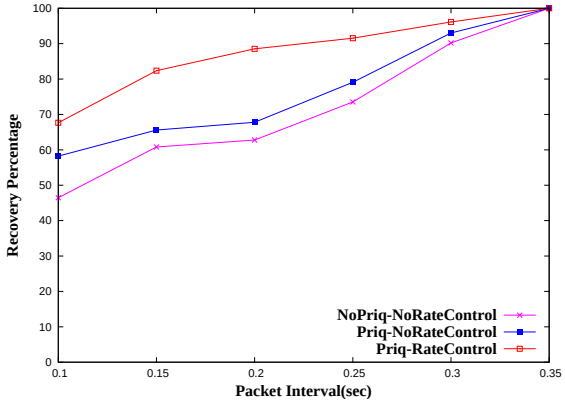
(a) Average Delay



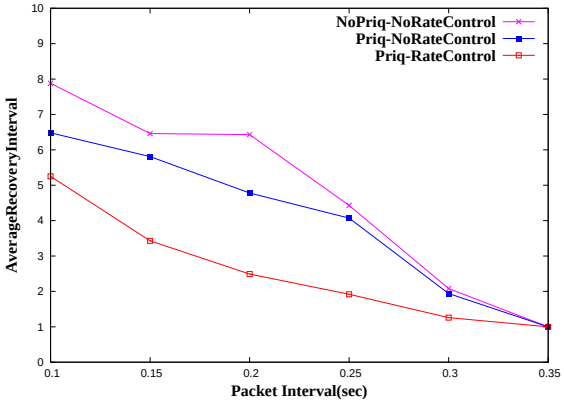
(b) Average Delay of Higher Priority Packets



(c) Freshness Ratio



(d) Recovery Percentage



(e) Average Recovery Interval

Figure 4.1: Performance when base station requires 75% of the packets sent

control.

The total number of higher priority packets that have reached the base station within the freshness interval is shown in Figure 4.1(c). The improvement in freshness ratio of higher priority packets with rate control is reflective of the decrease in the delay of higher priority packets with rate control. The improvement in freshness ratio with rate control is lower at highly congested scenarios, as the congestion in the network prevents higher priority packets from reaching faster. When the network traffic is less (for packet interval of 0.35), the freshness ratio reaches a maximum value even without prioritization and rate control. Hence, the curves converge at this data point.

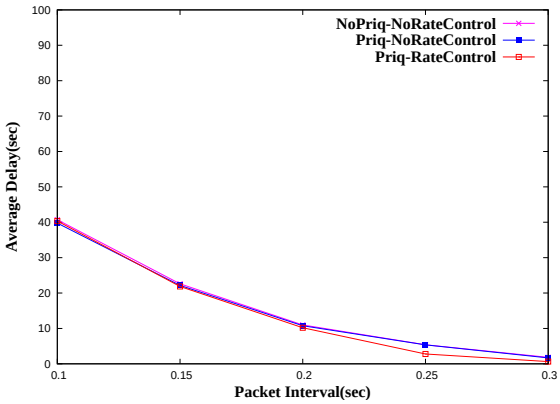
Figures 4.1(d) and 4.1(e) show the effectiveness of the proposed protocol with prioritization and rate control in terms of recovery percentage and the recovery interval. The increase in the recovery percentage with rate control increases as the network load increases as shown in Figure 4.1(d). Prioritization with rate control improves the recovery percentage. However, when the network is highly congested (for packet interval of 0.1 sec), the gains due to rate control are not very high because the congestion in the network introduces additional delays.

The results when the expected number of the packets at the base station is 50% of the packets sent are shown in Figure 4.2. Figure 4.2 shows that the proposed protocol with prioritization and rate control performs better than with no prioritization and no rate control and with prioritization and no rate control. From Figure 4.2(a), we can see that the average delay did not worsen with the proposed protocol. Also, from Figures 4.2(b) and 4.2(c), we can infer that the average delay of higher priority packets decreased and the freshness ratio increased with the proposed protocol. However, the improvement due to rate control increases as the network load reduces. At lower traffic loads, since the expected number of packets at the base station is only 50 percent, the recovery of the priority sources is faster

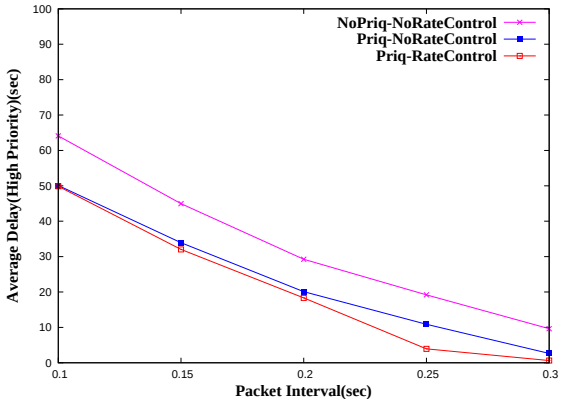
and the recovery percentage is high, resulting in an increase in the freshness ratio and the reduction in the average delay of higher priority packets. At higher traffic loads, the improvement in freshness ratio is less as the recovery at such congested scenarios is less, increasing the number of higher priority packets flowing in the network. This additional traffic introduces delay, reducing the improvement in freshness ratio. As expected, the recovery percentage increased and the recovery interval reduced with rate control as shown in Figure 4.2(d) and 4.2(e).

Figure 4.3 shows the results when the expected number of the packets at the base station is 25% of the packets sent. The average delay of the packets received at the base station is almost the same in all three cases as shown in Figure 4.3(a). The average delay of higher priority packets reduced and the freshness ratio improved as shown in Figures 4.3(b) and 4.3(c). As the expected number of packets at the base station is less, prioritization alone shows significant improvement. With rate control, there is a slight increase in performance. From Figures 4.3(d) and 4.3(e), we can see that the recovery percentage increased and the average recovery interval reduced with the proposed protocol.

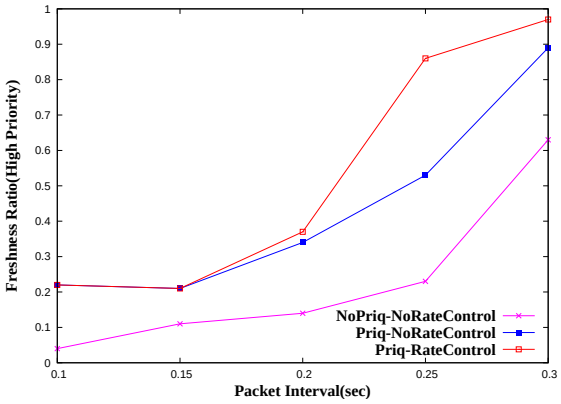
From Figures 4.1, 4.2, and 4.3, we can see that the proposed protocol performs as expected with all the metrics with different expectations at the base station. Data freshness in terms of the average delay of higher priority packets, freshness ratio of higher priority packets, recovery percentage, and average recovery interval improved significantly with the proposed protocol. The average delay of the packets received at the base station did not worsen with the proposed protocol.



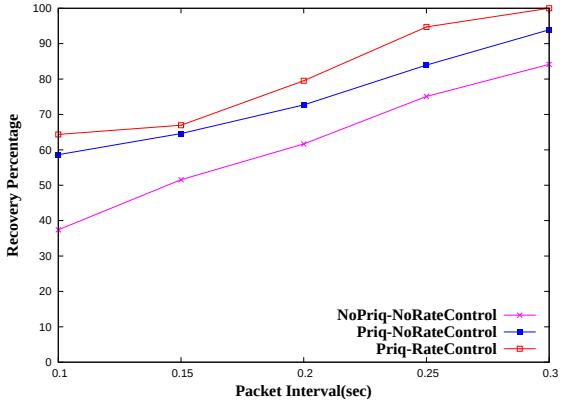
(a) Average Delay



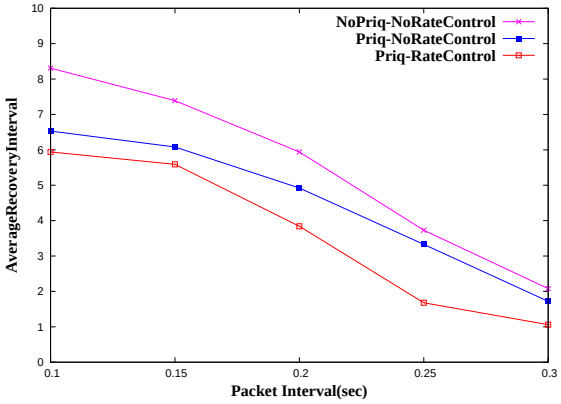
(b) Average Delay of Higher Priority Packets



(c) Freshness Ratio

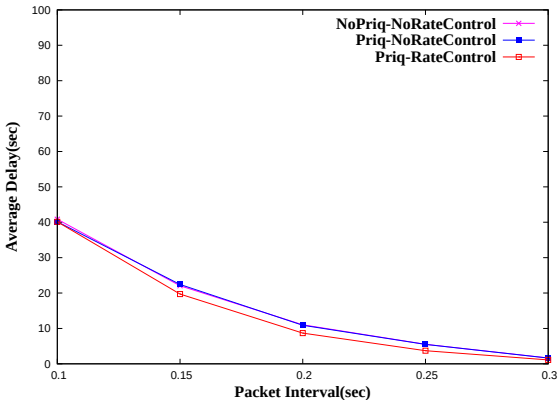


(d) Recovery Percentage

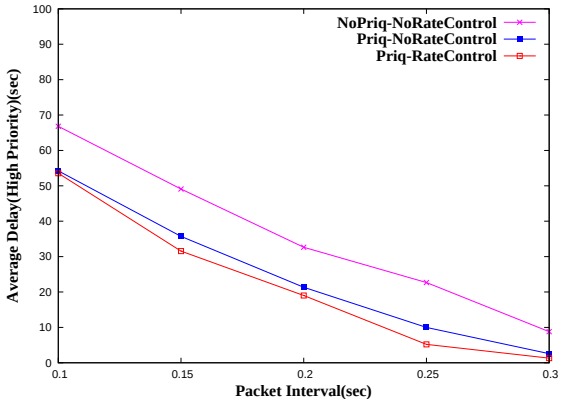


(e) Average Recovery Interval

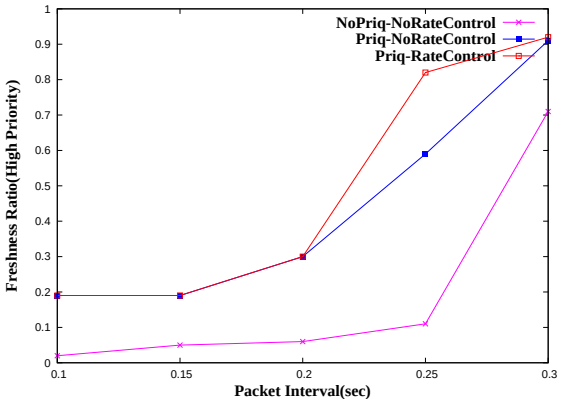
Figure 4.2: Performance when base station requires 50% of the packets sent



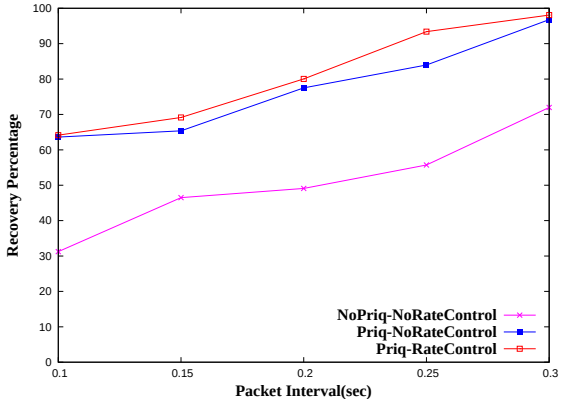
(a) Average Delay



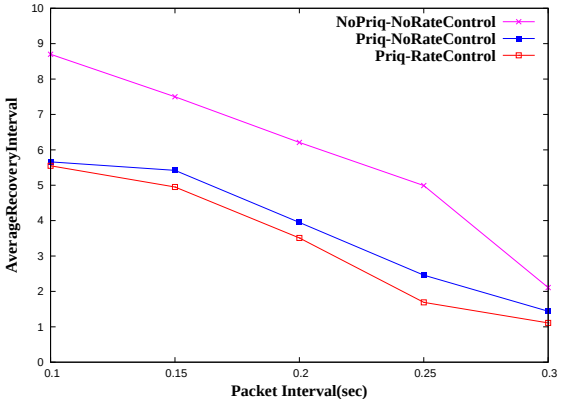
(b) Average Delay of Higher Priority Packets



(c) Freshness Ratio



(d) Recovery Percentage



(e) Average Recovery Interval

Figure 4.3: Performance when base station requires 25% of the packets sent

CHAPTER 5

CONCLUSIONS

In continuous-monitoring applications using sensor networks, each sensor node transmits its sensed data to the sink node periodically. Such applications require that the base station gets fresh data periodically. Data reaching the sink node after a certain threshold is not useful for processing or analysis because it is stale. Data freshness is critical in these applications. To improve data freshness, data packets should reach the base station reliably. One protocol that improves packet-level reliability using the concept of monitors was proposed in [36].

Current protocols that aim at improving data freshness measure freshness only in terms of latency of all packets received at the base station [30]. However, improving overall delay alone does not necessarily improve data freshness. Also, they do not address the areas or sources from which sufficient information is not received as desired by the application. In order to address the needs of continuous-monitoring applications and improve data freshness, we augmented the monitor configuration in the monitors's approach [36] for providing packet-level reliability and adapted prioritization and rate-control mechanisms to improve data freshness. We identified metrics to measure the freshness of data in terms of the average delay of higher priority packets, the number of higher priority packets that have reached the base station within the identified freshness interval, the percentage of sources recovered, and the recovery interval for a source node in addition to the average delay of all

packets received at the base station to capture the requirements of continuous-monitoring applications and provide a comprehensive view of the system's performance.

Our simulation results show that the proposed protocol improves data freshness significantly. We also ensured that the average delay of the packets received at the base station did not worsen with the proposed protocol. The work and the contributions proposed in this thesis may be extended giving rise to several lines of future work. As part of future research, we plan to study our protocol in conjunction with other packet-level reliability protocols.

REFERENCES

[1] I. Aron and S. Gupta. A witness-aided routing protocol for mobile ad-hoc networks with unidirectional links. *MDA*, 1999.

[2] V.B. Balasubramanyam, G. Thamilarasu, and R. Sridhar. Security solution for data integrity in wireless biosensor networks. *ICDCSW*, 2007.

[3] S. Bhatnagar, B. Deb, and B. Nath. Service differentiation in sensor networks. *Fourth international Symposium on Wireless Personal Multimedia*, 2001.

[4] M. Bouzeghoubi. A framework for analysis of data freshness. *Proceedings of the 2004 international workshop on Information quality in information systems*, 2004.

[5] C.L. Brian, M.B. Tarek, F. Abdelzaher, J.A. Stankovic, and T. He. RAP: A real-time communication architecture for large-scale wireless sensor networks. *Real-Time and Embedded Technology and Applications Symposium*, 2002.

[6] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *ACM MobiCom*, pages 85–96, 2001.

[7] A. Dhamdhere, H. Chen, A. Kurusingal, V. Sivaraman, and A. Burdett. Experiments with wireless sensor networks for real-time athlete monitoring. *IEEE SENSEAPP*, 2010.

[8] M. Dhanaraj, B.S. Manoj, and C.S.R. Murthy. A new energy efficient protocol for minimizing multi-hop latency in wireless sensor networksd. *IEEE PerCom*, pages 117–126, 2005.

[9] Q. Dong. Maximizing system lifetime in wireless sensor networks. *IPSN*, pages 13–19, 2005.

[10] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. *Mobile Computing and Networking*, 2000.

[11] N. Jain, D.K. Madathil, and D.P. Agrawal. Exploiting multi path routing to achieve service differentiation in sensor networks. *IEEE International Conference on Network*, 2003.

[12] K. Jamieson, H. Balakrishnan, and Y.C. Tay. SIFT: A MAC protocol for event driven wireless sensor networks. *Third European Workshop on wireless sensor networks Zurich, Switzerland*, 2006.

[13] K. Kifayat, M. Merabti, Q. Shi, and D. Llewellyn-Jones. Applying secure data aggregation techniques for a structure and density independent group based key management protocol. *Information Assurance and Security*, 2007.

[14] H. Kim and S. Min. Priority based QoS MAC protocol for wireless sensor networks. *IEEE International Symposium on Parallel and Distributed Processing*, 2009.

[15] B. Laura, A. Gal, and L. Raschid. Adaptive pull-based data freshness policies for diverse update patterns. *ACM Transactions on Database Systems (TODS)*, 2006.

[16] H. Le, H. Guyennet, V. Felea, and N. Zerhouni. Low latency MAC for event-driven wireless sensor networks. *Mobile Adhoc and Sensor Networks*, 2007.

[17] D. Lee and K. Chung. An energy efficicnet and low latency MAC protocol using RTS aggregation for wireless sensor networks. *International Conference on Advanced Technologies for Communications*, 2008.

[18] D. Lim, J. Shim, T.S. Rosing, and T. Javidi. Scheduling data delivery in heterogeneous sensor networks. *IEEE International Symposium on Multimedia*, 2006.

[19] Y. Liu and W.K.G. Seah. PRIMP: A priority based multi path routing protocol for sensor networks. *Personal, Indoor and Mobile Radio Communications*, 2004.

[20] G. Lu, B. Krishnamachari, and C. Raghavendra. An adaptive energy-efficient and low-latency MAC for data gathering in wireless sensor networks. *IEEE IPDPS*, pages 224–231, 2004.

[21] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel. Delay efficient sleep scheduling in wireless sensor networks. *IEEE INFOCOM*, pages 2470–2481, 2005.

[22] M. Medidi and Y. Zhou. Extending lifetime with differential duty cycles in wireless sensor networks. *IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1033–1037, November 2007.

[23] NS-2 network simulator. http://www.isi.edu/nsnam/ns.

[24] K. Nguyen, T. Nguyen, C.K. Chaing, and M. Motani. PSIFT: A prioritized MAC protocol for event driven wireless sensor networks. *Communications and Electronics*, 2006.

[25] S.N. Parmar, S. Nandi, and A.R. Chowdary. Power efficient and low latency MAC for wireless sensor networks. *SECON*, 2006.

[26] M.E. Rivero-Angeles and N. Bouabdallah. Event reporting on continuous monitoring wireless sensor networks. *IEEE Globecomm*, 2009.

[27] Y. Sankarasubramaniam, B.A. Ian, and F. Akyildiz. ESRT: Event-to-sink reliable transport in wireless sensor networks. *ACM Mobihoc*, 2003.

[28] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Optimizing sensor networks in the energy-latency-density design space. *IEEE Transactions on Mobile Computing*, pages 70–80, 2002.

[29] I. Solis and K. Obraczka. The impact of timing in data aggregation for sensor networks. *IEEE Intl. Conference on Communications (ICC)*, 2004.

[30] I. Solis and K. Obraczka. In-network aggregation trade-offs for data collection in wireless sensor network. *International Journal of Sensor Networks*, 2006.

[31] V. Srikanth and L. Ramesh Babu. Reliable and low latency MAC for event critical applications in wireless sensor networks. *IJCSNS Jan*, 2009.

[32] F. Stann and J. Heidemann. RMST: Reliable data transport in sensor networks. *SNPA*, 2003.

[33] W.L. Tan, O. Yue, and W.C. Lau. Performance evaluation of differentiated services mechanisms over wireless sensor networks. *VTC-2006*, 2006.

[34] F. Turati, M. Cesana, and L. Campelli. SPARE MAC enhanced: A dynamic TDMA protocol for wireless sensor networks. *IEEE Globecomm*, 2009.

[35] C.Y. Wan, A.T. Campbell, and L. Krishnamurthy. PSFQ: A reliable transport protocol for wireless sensor networks. *WSNA*, 2002.

[36] J. Wang and S. Medidi. Topology control for reliable sensor-to-sink data transport in sensor networks. *IEEE Intl. Conference on Communications (ICC)*, 2008.

[37] J. Wang and V.M. Vokkarane. Optimal remote homing for providing service differentiation in information aware multi layered wireless sensor networks. *IEEE Intl. Conference on Communications (ICC)*, 2006.

[38] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for adhoc routing. *ACM MobiCom*, pages 70–84, 2001.

[39] M.H. Yaghmaee and D.A. Adjeroh. Priority-based rate control for service differentiation and congestion control in wireless multimedia sensor networks. *The International Journal of Computer and Telecommunications Networking*, 2009.

[40] X. Yang and N. Vaidya. A wake-up scheme for sensor networks: Achieving balance between energy saving and end-to-end delay. *IEEE RTAS*, pages 19–26, 2004.

[41] Y. Zhou and M. Medidi. Energy-efficient contention-resilient medium access for wireless sensor networks. *IEEE Intl. Conference on Communications (ICC)*, 2007.

[42] Y. Zhou and M. Medidi. Sleep-based topology control for wakeup scheduling in wireless sensor networks. *IEEE Intl. Conference on Sensor Mesh and Ad Hoc Communications and Networks (SECON)*, pages 304–313, June 2007.