

A Fast and Reliable Protocol for Wireless Sensor Networks in Critical Conditions Monitoring Applications*

Azzedine Boukerche
PARADISE Research Laboratory
SITE – University of Ottawa
K1N 6N5 Ottawa, Canada
boukerch@site.uottawa.ca

Richard Werner Nelem Pazzi
DC - Universidade Federal de São
Carlos, CP 676
13565-905 São Carlos, SP, Brazil
richard@dc.ufscar.br

Regina Borges Araujo
DC - Universidade Federal de São
Carlos, CP 676
13565-905 São Carlos, SP, Brazil
regina@dc.ufscar.br

ABSTRACT

Sensor networks are increasingly being deployed for fine-grain monitoring of physical environments subjected to critical conditions such as fire, leaking of toxic gases and explosions. A great challenge to these networks is to provide a fast, reliable and fault tolerant channel for events diffusion, which meets the requirements of query-based, event-driven and periodic sensor networks application scenarios, even in the presence of emergency conditions that can lead to node failures and path disruption to the sink that receives those events. This paper presents a fault tolerant and low latency algorithm, which we refer to as PEQ (Periodic, Event-Driven and Query-Based Protocol), that meets sensor networks requirements for critical conditions surveillance applications. The algorithm uses the publish/subscribe paradigm to disseminate requests across the network and an ack-based scheme to provide fault tolerance. The algorithm was implemented using NS-2 simulator and compared to the Directed Diffusion paradigm. Important metrics were evaluated showing that the proposed algorithm can be a proper solution to meet constraints and requirements of events delivery in critical conditions monitoring applications.

Categories and Subject Descriptors

C.2.2 [Computer System Organization]: computer-communication networks –*Network protocols, Routing protocols.*

General Terms

Algorithms, Measurement, Performance.

Keywords

Sensor networks, routing protocol, publish/subscribe.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MSWiM'04, October 4–6, 2004, Venezia, Italy.

Copyright 2004 ACM 1-58113-953-5/04/0010...\$5.00.

*This work was supported by the Canada Research Chair Program, NSERC, Canada Foundation for Innovation Funds, and OIT/Distinguished Researcher Award.

1. INTRODUCTION

With the recent developments in wireless networks and multifunctional sensors with digital processing, power supply and communication capabilities, wireless sensor networks are being largely deployed in physical environments for fine-grain monitoring in different classes of applications [1] [15]. One of the most appealing applications is security surveillance and critical conditions monitoring. In a prison, for instance, it is important to keep a reliable monitoring of the physical environment, especially when emergency situations emerge, such as prisoner rebellions that can lead to incendiary fire conditions and losses of human lives and patrimony. In such situations, it is important that information can be “sensed” from the physical environment while the emergency state is in progress, since more precise information can be used by security and rescue teams for operation management and better strategic decisions. However, in order to keep the information flowing from the sensors during the emergency, a wireless sensor network solution has to cope with the failure of sensor nodes (sensors can be burnt, have their propagation jeopardized by interferences, such as water or dense smoke present in the environment, can be malfunctioning etc). Thereby, wireless sensor network solutions for such environments have to be fault tolerant and reliable, and to provide low latency, besides fast reconfiguration and energy saving. In terms of energy savings, in a silent monitoring state, sensor nodes can be programmed to notify about events in a periodic fashion (send temperature at every 10 minutes) or event-driven fashion (send temperature only when above 60°C). In these cases the interest may not change for quite some time.

Some existing energy saving solutions take that into consideration and switch some nodes off, leading the nodes to an inactive state – these are waken up only when interest matches the events “sensed” [11]. On the other hand, in query-based application scenarios, queries (new interests) can be propagated to sensors arbitrarily, according to the application and/or user’s will and so, some existing energy saving solutions may not be adequate because the transition from inactive state to data transfer state can be costly in terms of energy use when many arbitrary transitions are necessary [5]. Moreover, energy saving and fault tolerance support can present conflicting interests when more paths, involving inactive nodes, have to be quickly set up because of failure in nodes of previous paths. This paper describes PEQ, the wireless sensor network algorithm, which supports, simultaneously, Periodic, Event-driven and Query-based

application scenarios. PEQ can provide low latency for event notification, fast broken path reconfiguration, and high reliability in the delivery of the events with low energy dissipation. Low latency is achieved by the use of the shortest path for the delivery of events. Fast subscriptions of new interests (for query-based scenarios) are provided by the concept of driven delivery of events, in which new subscriptions to a sensor region are speed up by using the inverse path used for event notifications. This has an impact on energy saving, since less traffic is disseminated through the network for both event notification and broken path reconfiguration.

The network nodes can trigger fault tolerance when they detect a node failure, in which case the nodes find, cooperatively, the fastest path, with smallest possible number of transmissions. The provision of low latency, high reliability even in the presence of failures, fast subscription of new interests and energy saving makes PEQ a choice algorithm to support applications in areas ranging from Health care (body vital signs monitoring, medical instruments, localization of objects and people in health-care facilities, laboratories, etc.) to Transportation (traffic control, vehicle supervision and control, etc), Government (environmental control, meteorological services, key national symbols, e.g., cultural institutions and national sites and monuments), Manufacturing (including chemical industry and defense industrial base) and Miscellaneous (smart super markets, tourism guides, entertainment etc.). The sensor network is configured through a hop tree, which is built at the configuration time. The publish/subscribe paradigm is used to promote the interaction between sensors and sink. Subscriptions to the nodes are propagated to the sensors through the hop tree created. In order to better describe the algorithm, a grid model is used. However, the solution can be applied to mesh and dense randomly deployed sensor node networks as well.

The paper is organized as follows. Section 2 describes the PEQ algorithm, showing snapshots of important parts of the code, such as the network configuration, subscription and notification of events and network reconfiguration in face of node failures. Section 3 discusses the experiments performed, the simulation scenarios, metrics used and results obtained. In section 4 interesting related works are discussed and compared to PEQ. Finally, conclusions are described in section 5, followed by References.

2. DESCRIPTION OF PEQ

The main motivation for the work described here is driven by the need to provide support for all of the following requirements simultaneously: low latency, reliability, fast path recovery in the presence of failures and energy savings. Although several interesting solutions have been reported in the literature, they basically do not attend all three requirements at the same time. Moreover, some solutions either require special hardware or sophisticated processing at the nodes. The basic idea of the PEQ algorithm is to use ordinary nodes, with no special hardware and a simple processing at each node by using the hop level as the main information to minimize data transmission. In the presence of failures, a switch to a fast recovery mode is done keeping the exchange of information among neighbor nodes to a minimum, differently from other solutions. PEQ is a routing algorithm, which is realized in three steps. The first step comprises the construction of the hop tree. The sink starts the process of

building the hop tree, which will be used as a configuration and subscription message propagation mechanism to the sensor network. The second step involves the propagation of subscriptions to the sensor network. Finally, the last step is responsible for delivering events from the sensors to the sink, by using the fastest and less costly route, in terms of energy savings.

Next sections describe the publish/subscribe paradigm as the mechanism for sensors/sink interaction, followed by the description of the routing steps. It is assumed that the nodes are disposed as a grid so that the transmission coverage of one sensor node is capable of reaching its eight neighbor nodes. However, the solution can be applied also for mesh networks model as well as dense randomly deployed sensor node networks, as shown along the text.

2.1 The publish/Subscribe Paradigm for Sensors/Sink Interaction

Sensors networks can have thousands of nodes, each one producing an event that is delivered to one or more static or mobile sinks. Several communication paradigms can be used to promote the interaction between sensors that produce events and sinks that consume those events. Examples of such paradigms include message passing, remote invocations, notifications, shared spaces and message queuing. The basic problem with these paradigms is that they fail to promote full decoupling between participants, making the system less flexible and less scalable [9]. Eugster and colleagues [8] make an excellent study of all these paradigms and compares them to the publish/subscribe paradigm, which has received increased attention because it decouples consumers and producers in time (publishers and subscribers do not need to be active in the interaction at the same time), space (publishers and subscribers do not need to know each other) and flow (publishers and subscribers do not need to be synchronized to interact). In the publish/subscribe interaction paradigm, one or more sinks receive events notification from a sensor network. The sink expresses interest in a sensor by subscribing to certain information it requires from the sensor. When the sensor detects that information, it publishes it by firing an event, which is sent to the sink(s) through a notification message in an asynchronous way.

The publish/subscribe mechanism can be used to convey and receive notification of the following types of interest: periodic information, query-based and event-driven. PEQ algorithm has a simple and effective solution for the subscription propagation and notification of events, as described in the next sections.

2.2 Building the Hop Tree

In the wireless sensor network considered here, one node does not have a global understanding of the network, i.e., a node only knows a small amount of information about its nearest neighbors (those that are within its coverage reach). In a first moment, each node knows only the hop level, of a hop tree, that it is in. The hop tree is started by a sink, which transmits to its neighbor(s) an attribute-value pair called hop.

The algorithm for building the hop tree is based on flooding the network, starting from the sink, with a hop value, which is stored, incremented and transmitted to its neighbor nodes. These neighbor nodes store the received hop value, increment it and transmit it to its neighbor nodes and so on until the whole sensor

network is configured with different levels of hops. Because the communication among the network nodes is through radio frequency, all the neighbors of a node receive the transmission. So, one node that has already transmitted, will receive its neighbor's transmission, generating a loop. In order to avoid these useless transmissions that cause energy waste, a set of rules was established as part of the algorithm for the hop diffusion. One of the local rules establishes that when a node receives a hop from its neighbor, it checks this value against its local hop value. If the local hop value is greater than the received one, the node updates its hop, increment this value and retransmit it to its neighbors. In case the locally stored hop is smaller or equal to the received hop, the node does not update its hop and does not transmit it. Figure 1 shows the initial configuration of a mesh network.

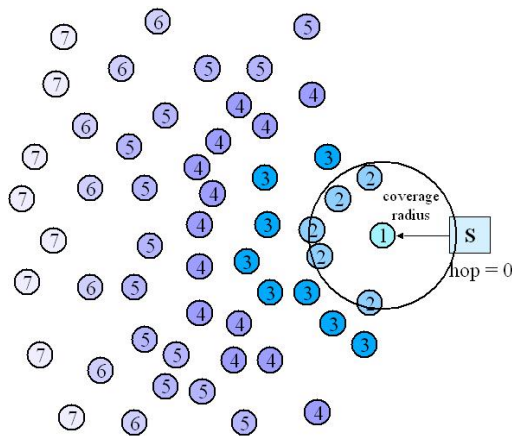


Figure 1. Hop configuration in a mesh network.

The data structure used in the algorithm comprises three tables: configTable, routingTable and subscriptionTable. The configTable holds the configuration parameters associated with a sink. A node uses the routingTable to forward messages to its neighbor nodes. Finally, the subscriptionTable is used to store subscriptions a node receives. The routingTable has four fields: sinkID, senderID, destID, and coordinates. The coordinates attribute is used to indicate the position of the node, so an application can know where the readings come from, and a sink can send a subscription to a region delimited by coordinates, instead of sending to specific source IDs.

2.3 The Subscription Message Propagation

In the publish/subscribe paradigm, for a sink to be notified about the events that are captured from the physical environment by the sensors, it needs to subscribe to one or more nodes for a given information, by setting one or more criteria (temperature > 60oC, presence of smoke, etc) that have to be matched before any event is sent. By sending events only when they match a criterion, it reduces network traffic, causing less waste of energy and extending the sensors network life. After the initial configuration of the network, the only information a node has is the hop level it is in. This information alone is not enough for efficient subscription propagation. In the absence of any information about which node of the network can satisfy the sink interest, one way to propagate the initial subscription is to flood the network with this interest. Each node of the network keeps a small subscription table and a routing table. Each record of the subscription table

represents a different subscription. During the subscription message propagation, when a node receives this message, it compares the coordinate attribute to its own coordinates. If they are the same, it means that the subscription is meant to this node and so, it is stored in its subscriptions table. Otherwise, the node only re-transmits the subscription as part of the algorithm. During the subscription propagation, when a node receives a transmission, it sets its routingTable destID to be the node that has transmitted and sets its sinkID to be the corresponding sink that has sent the subscription. This information will be used to forward data back to the sink. When a node needs to forward data to a sink, it checks its routingTable and forwards the packet to the destID corresponding to the sinkID.

2.4 Sending the Notification Message

When information is captured from the physical environment by a sensor, it checks its subscription list to determine if there is any registered interest. If a criterion is met, the node verifies the senderID of the node that transmitted the subscription. After that, the node assembles an event notification message that contains the following attributes: type, value, coordinates, sinkID and send them to its neighbors. When each neighbor node receives the message, it compares the received destID with its own ID. If the result is true, the node stores coordinates and senderID in its routing table, gets the destID of the routing table and each node repeats the algorithm until the notification reaches the sink.

Due to the initial configuration characteristic of the network, the maximum allowed number of neighbors that transmits to a node is three. Each node processes messages only from the nodes that are in a previous hop level. This characteristic makes it easy to select the neighbor that transmitted faster, besides avoiding messages loops. Supposing that a sink S sends a subscription to the network, and considering that the top-left-most node is the sensor that produces an event that meets the subscription criterion of sink S, the path that is created down to the sink for the sending of the notification message can be seen in Figure 2.

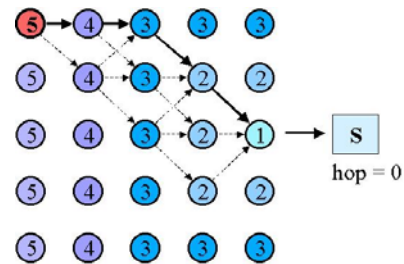


Figure 2. Path for notification delivery.

Note that the arrows indicate the links that could form alternative paths, depending only on the choice each node makes for the fastest node that delivered the subscription. An important feature of the configuration of the hops values can be observed in the notification transmission phase. When a node receives a transmission from a neighbor node, it only retransmits the message if the node has higher hop number (one unity more). For instance, only the nodes with hop = 4 retransmit the information received from nodes with hop = 5, and so on.

The path used to forward data from the source node to the sink can also be used to forward subscription to the source node, as

shown in Figure 3. In order to be able to use this reverse path, the nodes in the path send data to the senderID they get from their routingTables. This is useful when subscriptions of query-driven type have to be supported. Otherwise, if the node does not have a matching value for coordinates, the node transmits without specifying the senderID, so that all neighbors will transmit the subscription message. According to this algorithm, one node transmits only if its hop value matches the hop value received.

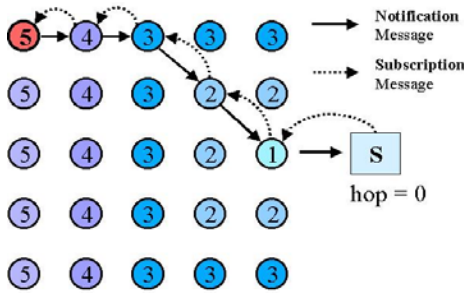


Figure 3. Driven delivery of subscriptions.

Because the driven delivery of subscriptions use the same path created for notification messages, only the nodes comprising this path spend energy for transmission. The other nodes either receive and do not transmit (as is the case of the neighbor nodes to the path nodes) or do not even receive messages. Figure 4 shows a map that represents the energy consumed by the network when using the referred path. The darker nodes denote a larger expenditure of energy.

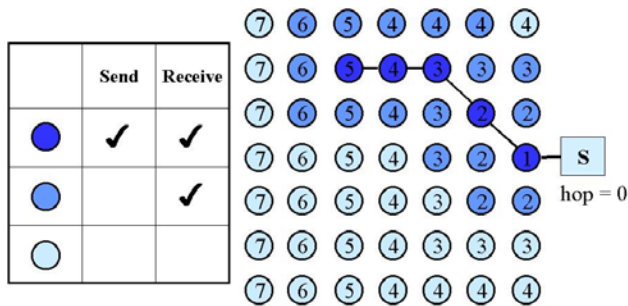


Figure 4. Energy map of the network.

2.5 Path Repair Mechanism

The path created for sending the notification message is unique and efficient (promotes low latency and saves energy). It can also be used for the driven delivery of new subscriptions (for query-based scenarios, for instance, that may require random subscriptions). However, because the path is unique, any failure in one of its nodes will cause disruption, preventing the delivery of the event as well as the subscriptions. Possible causes of failure include: low energy, physical destruction of one or more nodes, communication blockage, etc. Many routing algorithms for sensor networks have been proposed in the literature. Some are based on periodic flooding mechanisms [11] [12], rooted at the sink, to repair broken paths and to discover new routes to forward traffic around faulty nodes. This mechanism is not satisfactory in terms of energy saving because it wastes a lot of energy broadcasting repairing messages. Furthermore, during the interval of network flooding, these algorithms are unable to route data around failed nodes, causing data losses.

The PEQ algorithm offers an ack-based path repair mechanism. This repairing mechanism consists of two parts: failure detection at the destination node and selection of a new destination. Right after the initial configuration phase, each node has only one destination node to forward data to the sink, due to the single (shortest) path created.

When a node, named sender, needs to forward data to its destination, it simply sends the data packet and sets a timeout and waits for the neighbor's acknowledgment. If the sender receives its neighbor's ACK, it can infer that the neighbor is alive. The neighbor node sends the ACK message right after it has forwarded the original packet, therefore the sender node knows that its packet was properly forwarded, and it does not need to retransmit the packet nor choose another neighbor. If the sender node does not receive the ACK message, a problem must have occurred with the neighbor and another node should be selected as the new target. Then the sender immediately broadcasts a SEARCH message to its neighbors. The nodes will reply with a message to the sender containing their hop level and identification. The next step is to select a new destination. The sender chooses the neighbor with lower hop level to be its new destination. The sender then updates its routing table to ease the forwarding of subsequent packets. In order to avoid creating closed paths (loops), the sender sets its own hop level to be the destination hop level plus one. If any neighbor does not reply the SEARCH message, the sender node has to retransmit this message. If the node is isolated, the only solution is to increase its radio range. Note that the backtrack mechanism is implemented here, as any node may respond the request, even nodes with higher hop levels, including the originator of the packet.

A disrupted path is shown in Figure 5. After the repairing mechanism is exploited, the path is reconstructed as can be seen in Figure 6. It is obvious that, if all neighbors of a node fail, this node will be isolated and its transmission will not reach any neighbor. One solution would be to configure the radio module of the node to increase its coverage area, but this will spend more

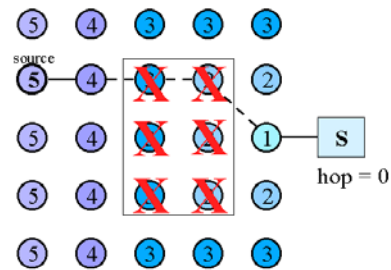


Figure 5. Region with destroyed nodes.

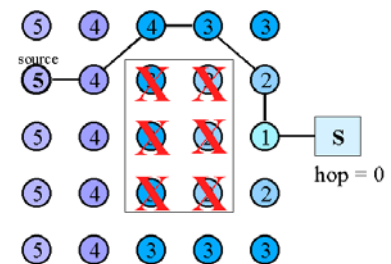


Figure 6. Repaired path.

energy. Another solution is to provide fault tolerance through the establishment of multiple paths from the nodes to the sink.

3. SIMULATION EXPERIMENTS

This section describes the investigation of the performance evaluation of PEQ through an extensive set of simulation experiments. PEQ results are compared to the well-known Directed Diffusion communication scheme for sensor networks [11].

3.1 Simulation Scenario and Metrics

The PEQ algorithm was programmed in C and implemented using NS-2 simulator [18]. The simulation scenarios consist of several sensor fields with different sizes, ranging from 100 to 500 nodes. The nodes were randomly placed in the field and a fixed workload of five sources and one sink was used. Sources were placed at the left side of the field and the sink at the right side so that when the network size is increased, path numbers between sources and sink are also increased (due to the number of nodes). Thus, the impact of network size on PEQ's performance can be more noticeable than randomly selecting the sources and the sink.

Table 1 lists the simulation parameters used. The input values were basically based on the values reported for Directed Diffusion (DD) in [11]. For different input values, such as the data rate for each source (which was chosen to be 10 events per second, instead of 2 per second as reported for DD), DD was simulated with the same values used by PEQ. In this case, the authors found that 2 events per second might not be enough for people localization in emergency situations. The interval for sending a repairing event was set to 20 seconds. Nodes in the network were configured to dissipate energy according to the parameters of Table 1 (adopted from [11]), and their radio range was set to 20 meters to more closely mimic realistic sensor radio modules.

Table 1. Simulation parameters [11].

Parameters	Values
Simulation time (s)	500
Number of nodes	100-500
Source data rate (events/s)	10
Repairing event interval (s)	20
Radio range (m)	20
Transmit Energy (mW)	14.88
Receive Energy (mW)	12.50
Dissipation in Idle (mW)	12.36
Dissipation in Sleep (mW)	0.016

PEQ and DD were tested with the same simulation scenarios and parameters. Each value measured was taken from a mean of 20 simulations. Event delivery delay and delivery rate are critical metrics for the performance of supervision applications. Moreover, dissipated energy may have a large impact on these delays. Thus, PEQ is evaluated through the following metrics:

- **Sink-Source-Sink Delay** - latency from the moment a subscription is sent to the moment the first corresponding event is received at the sink;

- **Average Delay** - average latency from the moment an event is transmitted to the moment it is received at the sink;
- **Average Event Delivery Ratio** - number of distinct received events to the number of originally sent events ratio;
- **Average Dissipated Energy** - total dissipated energy to the number of nodes ratio.

3.2 The Effectiveness of PEQ

The sink-source-sink provides the time it takes from the moment a subscription message is sent to the moment the first event is received. This metric is particularly important in query-based applications, which demand a fast and reliable response, like people localization queries for finding people in a building subject to an emergency situation, such as detection of fire or high temperatures, for instance. Low delay is important in this example since people can move fast in a few seconds and so, information can be out of date very quickly, preventing actions that could save lives if taken in a secure time.

PEQ uses the subscription message to propagate the initial configuration that builds the path to the sink and when the source receives the subscription, it uses this path to deliver data to the sink. DD, on the other hand, propagates the subscription (interest) and, when the source receives it, it propagates an exploratory event to the sink using multiple paths - the sink will reinforce one of these paths. The PEQ path creation has fewer steps and is faster than DD, resulting in lower delays, as shown in Figure 7.

The average latency from the moment an event is transmitted to the moment it is received at the sink is shown in Figure 8.

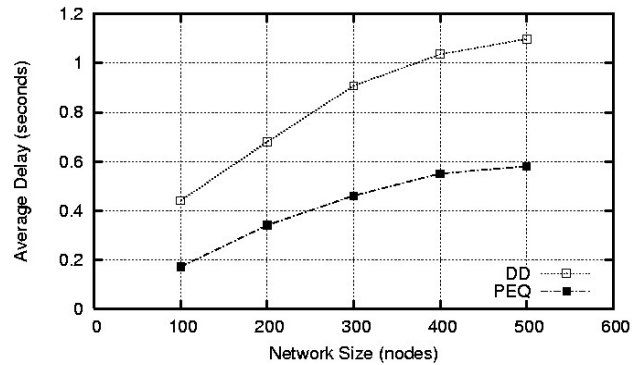


Figure 7. Sink-source-sink delay.

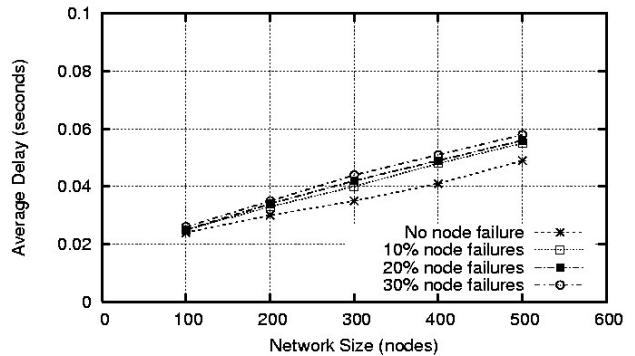


Figure 8. Average delay with node failures.

Turning off a fixed fraction of nodes simulated node failures. These nodes were randomly chosen from the sensor field and turned off at a random time during the simulation. As network size is increased, the delay gets higher due to the greater number of hops an event has to travel from source to sink. This makes sense, as, in order to repair a broken path, the algorithm has to find "alive" nodes. Since the number of failed nodes increases, the new created path becomes longer. PEQ always tries to find the shortest path to the sink. As shown in the graph, for a fixed network size, say 500 nodes, the delay increases from 0.049 to 0.058 seconds, an acceptable latency for the application scenario considered in this paper. By comparing PEQ with DD, PEQ algorithm resulted in lower delays, as observed in Figure 9.

This happens because of the complex mechanism DD uses to find a path. The graph plots the comparison of the algorithms in a sensor field with 30% of node failures. Sensor network reliability can be measured by its average event delivery ratio, which reflects the success rate of events transmissions to the sink. Figure 10 shows that PEQ is able to maintain a reasonable event delivery ratio even at a high percentage of node failures. Compared with DD, PEQ has a better ratio because it uses fewer messages to set up paths and propagate events, resulting in a smaller amount of packet collisions, as shown in figure 11.

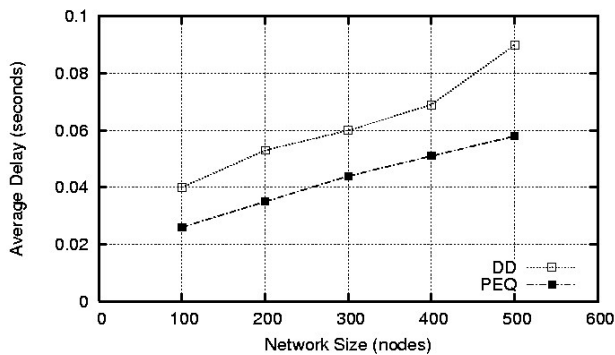


Figure 9. Average delay in event delivery.

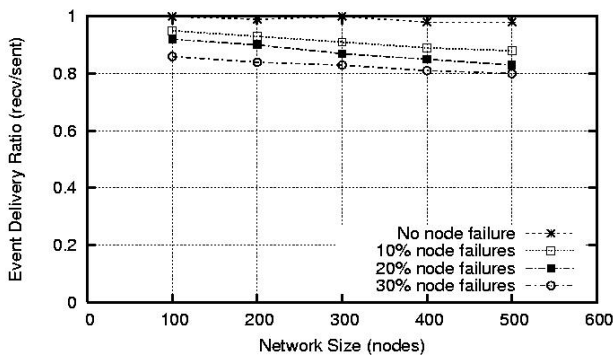


Figure 10. Average event delivery ratio with node failures.

Figure 12 plots the average dissipated energy per node on the presence of various node failure percentages. Because of space restrictions, it was chosen to show only the comparison graph with 30% of node failure. The other percentage values showed similar advantages of PEQ over DD. Figure 13 shows the average dissipated energy. As a large number of nodes remains in idle

state when not transmitting or receiving, the dissipated energy does not change very much, comparing to DD. One would expect the algorithm to spend more energy finding alternate paths due to node failures but, the fact that idle radio modules spend as much energy as receiving transmissions, the idle time energy utilization absolutely dominates all simulations.

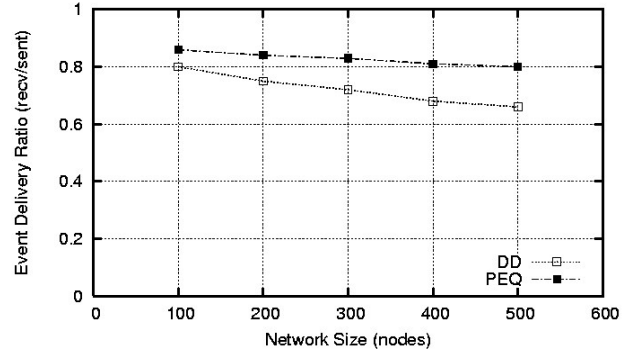


Figure 11. Average event delivery ratio.

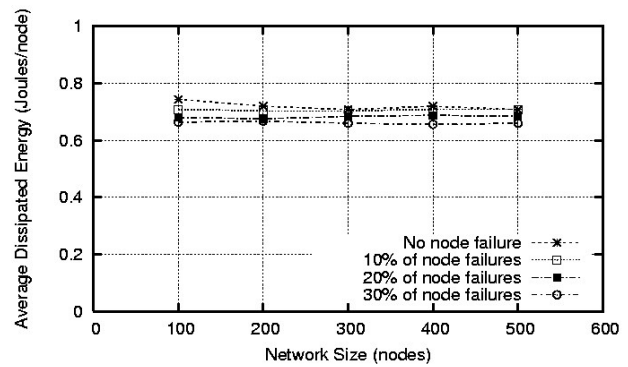


Figure 12. Average dissipated energy with node failures.

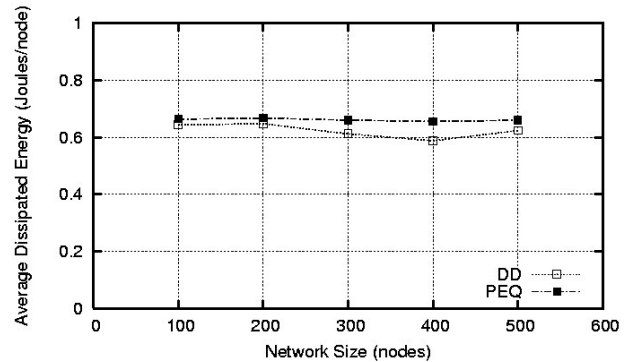


Figure 13. Average dissipated energy.

All experiments reported above were conducted with a fixed event rate of 10 events per second. However, it is important to know how a varying event rate can impact on average delay and event delivery ratio. Experiments with varying rate of 5, 10 and 15 events per second were carried out. Figure 14 shows that an increase in the event rate has raised the latency observed, especially when 15 or more events per second were generated by the sources. This of course was expected since a rate of 15 events

per second generates much more traffic and therefore more packet collisions and losses, as shown in Figure 15, where the average delivery ratio decreases as more events per second are generated. The sensor field considered here has only one sink, so that the algorithm could be better evaluated in a traffic jam situation. The nodes closer to the sink have to deal with a great number of packets per second, limiting the performance of the network and impacting on its lifetime. One solution would be to throw more sinks and distribute the load among them, but it is subject of future work.

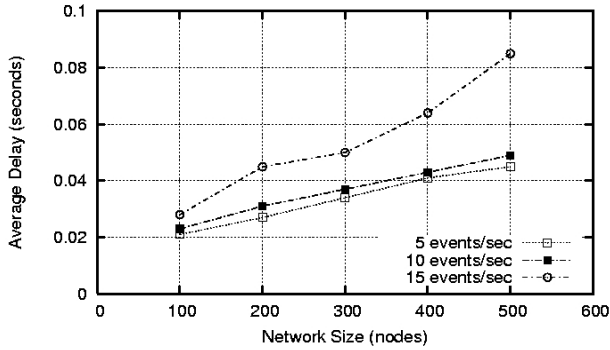


Figure 14. Average delay.

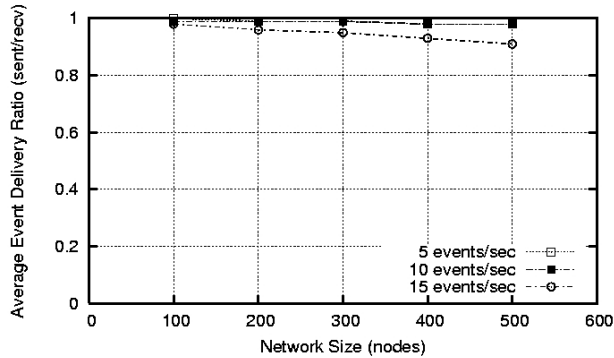


Figure 15. Average delivery ratio.

4. RELATED WORK

As mentioned above, the main motivation for the work described here are the challenging requirements posed by different types of scenarios for critical conditions monitoring applications: periodic, event-driven and query-based. These scenarios demand fast path set up for query-based subscriptions, such as localization of people in a building during a fire emergency, low latency for event delivery and reliability (high event delivery ratio), with minimum energy dissipation. Because energy saving is an important issue to preserve and extend a sensor network lifetime, various energy saving based solutions were reported in the literature. It is well known that a sensor in idle state consumes almost the same amount of energy than when it is awake, and that energy savings means to turn off communications completely (sleep mode) [1] [11]. STEM [17] provides a good solution for energy saving when the sensors have to be switched to the data path mode (awake state) once in a while, i.e., when application scenario is basically event-driven. However, when the state switch from sleep to awake has to be made often, such as when

different types of subscriptions are requested (query-based or periodic), then the switch could outperform the savings of energy. EAD [3], a network level energy aware routing protocol, uses novel concepts of neighboring broadcast scheduling and distributed competition among neighbors based on residual energy in order to set up a backbone for reliable delivery of notification. It is a great challenge to meet energy saving and fault tolerance requisites simultaneously, because these requirements can be conflicting. For instance, the multi-path version of the Directed Diffusion paradigm [10] uses multiple routing paths to transfer data, so that node failures in one path can be overcome by sending the data through multiple paths what increases energy consumption and can cause packet collisions. ARRIVE [13] and INSENS [7] use variations of this concept to cope with node failures. Deng and colleagues [6] proposes a solution for tolerance to both random node failure and area failure. It dynamically repairs a routing path between a sensor node and a sink, by storing in a node its parent node. The problem with this solution is that the nodes have to exchange information through a protocol that has three phases what can be a cause for extra delays. Because results were not shown in the paper, it remains to be seen if new routes can be found keeping latency low. The PFR protocol [4] is inspired by the probabilistic multi-path of the Directed Diffusion paradigm. Basically, it favors transmissions towards the sink using nodes within a zone around an imaginary line connecting a source node to the sink. The protocol forms a “thin zone” of nodes to propagate the data to the sink. The capacity of estimating the direction of a received transmission can increase the node energy consumption and cost, as the node needs to have a magnetometer module. An extended version of PFR (SW-PFR) introduces sleep-awake periods in order to save energy [16]. Variable Transmission Range Protocol - VRTP [2] copes with fault tolerance and energy saving by allowing the data transmission range to vary in such a way to overcome node failures or obstacles. Network lifetime is increased since critical sensors (those that are close to the sink) are not overused; however a more complex hardware has to be deployed. Algorithms that deal with node failures include SPIN (SPMS) [14], which uses meta-data exchange prior to exchange of data to decide if a node requires the data. It uses shortest distance multi-hop routing for the request and data transfers, saving energy and reducing end-to-end delay. SPMS fault-tolerance mechanism consists in keeping both the current and the second shortest route in the routing table. When node failures occur in the current shortest path, the second path can be chosen. However, in an emergency situation such as a fire condition, many nodes can failure, including nodes from pre-defined and stored paths.

This paper describes PEQ, a simple algorithm that uses ordinary sensor node hardware with short radio range. By using flooding for the nodes initial configuration, it establishes a low latency shortest path for event delivery using a small amount of information for the routing mechanism (basically the hop level and routing table). Node failure detection is based on acknowledgements. If a failure is detected, unlikely other solutions that uses 3 way protocol, PEQ broadcasts a SEARCH message to its neighbors, and receives a reply with their hop level and identification. The neighbor with lower hop level is chosen as the new destination, and loop back is avoided. Apart from good performance in terms of average delay and average delivery ratio, as shown in the experiments above, one good point about PEQ is

that the logic in the nodes is simple, and the amount of information held at each node is small compared to many existing solutions, making node processing very fast. In other paradigms, like Directed Diffusion [11], every node stores every interest message, even if the node does not publish a matching event. In our algorithm, each intermediate node has a routing table to direct incoming messages, and it does not have to run a complete matching algorithm every time it receives a message, what reduces processing time at each node, reducing latency.

5. CONCLUSIONS

Sensor networks are increasingly being used for continuous sensing, event detection, location sensing as well as micro-sensing in applications areas ranging from health care, to transportation, finance, defense, food, government, manufacturing, fire fighting, and much more. One of the most appealing applications is security surveillance and critical conditions monitoring. Sensor networks are increasingly being deployed for fine-grain monitoring of physical environments subjected to critical conditions such as fire, leaking of toxic gases and explosions. A great challenge to these networks is to provide a fast, reliable and fault tolerant channel for events diffusion, which meets the requirements of query-based, event-driven and periodic sensor networks application scenarios, even in the presence of emergency conditions that can lead to node failures and path disruption to the sink that receives those events.

This paper describes PEQ, a wireless sensor network algorithm that provide low latency for event notification, dynamic broken path reconfiguration, high reliability in the delivery of the events with low energy dissipation. Low latency is achieved by the use of the shortest path for the delivery of events. New subscriptions to a sensor region are speed up by using the reverse path used for event notifications. Individual nodes, instead of a sink-based mechanism, trigger fault tolerance mechanism locally. Important metrics were evaluated and compared to the Directed Diffusion paradigm, showing that PEQ has lower average delay, lower sink-source-sink delay, and better Average Event Delivery Ratio, making it a good candidate to meet constraints and requirements of events delivery in emergency situations monitoring applications.

6. REFERENCES

- [1] Boukerche, A. and I. Nikolettseas, "Protocols for Data Propagation in Wireless Sensor Networks", Chapter 2, in *Wireless Communications systems and networks*, Edt. M. Guizani, Kluwer Publ., pp. 23-51, 2004.
- [2] Boukerche, A. Chatzigiannakis, I. and Nikolettseas, S. A New Energy Efficient and Fault-tolerant Protocol for Data Propagation in Smart Dust Networks using Varying Transmission Range. Annual Simulation Symposium. Proceedings of the 37th ACM/IEEE Annual Simulation Symposium - ANSS (2004).
- [3] Boukerche, A. Cheng, X. And Linus, J. Energy-Aware Data-Centric Routing in Microsensor Networks. In *MSWiM'03*, September 19, 2003, San Diego, California, USA. (2003).
- [4] Chatzigiannakis, I., Dimitriou, T., Nikolettseas, S. and Spirakis, P. A Probabilistic Algorithm for Efficient and Robust Data Propagation in Smart Dust Networks. CTI Technical Report (2002).
- [5] Chatzigiannakis, I., Nikolettseas, S. and Spirakis, P. A Comparative Study of Protocols for Efficient Data Propagation in Smart Dust Networks. In *Proc. 2nd ACM Workshop on Principles of Mobile Computing – POMC'2002* (2002).
- [6] Deng, J., Han, R. and Mishra, S. A Robust and Light-Weight Routing Mechanism for Wireless Sensor Networks. To appear at 2004 DIWANS - 1st Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks, Florence - Italy, June 29 (2004).
- [7] Deng, J., Han, R. and Mishra, S. INSENS: Intrusion-Tolerant Routing in Wireless Sensor Networks. Poster paper. In the 23rd IEEE ICDCS 2003, Providence, RI, May (2003).
- [8] Eugster, P. T., Felber, P., Guerraoui, R. and Kermarrec, A. The many faces of publish/subscribe. *ACM Comput. Surv.* 35(2): 114-131 (2003).
- [9] Eugster, P. T., Guerraoui R. and Sventek, J. Distributed Asynchronous Collections: Abstractions for Publish/Subscribe Interaction. Elisa Bertino (Ed.): *ECOOP 2000, LNCS 1850*, pp. 252-276. Springer-Verlag Berlin Heidelberg (2000).
- [10] Ganesan, D., Govindan, R., Shenker, S. and Estrin, D. Highly resilient, energy efficient multipath routing in wireless sensor networks. *MC2R*, 1(2) (2002).
- [11] Intanagonwivat, C., Govindan, R. and Estrin, D. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *Proc. 6th ACM/IEEE Intl Conf. on Mobile Computing – MOBICOM'2000* (2000).
- [12] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister, "System architecture directions for networked sensors," in *Proceedings of ACM ASPLOS IX*, November (2000).
- [13] Karlof, C., Li, Y. and Polastre, J. Arrive: Algorithm for robust routing in volatile environments. Technical Report UCBCSD-02-1233, Computer Science Department, University of California at Berkeley, May (2002).
- [14] Khanna, G., Bagchi, S. and Wu, Y. Fault Tolerant Energy Aware Data Dissemination Protocol in Sensor Networks. *IEEE DSN*. Jun 28 to Jul 1, Florence, Italy (2004).
- [15] Min, R., Bhardwaj, M., Cho, S., Sinha, A., Shih, E., Wang, A. and Chandrakasan, A. Low-Power Wireless Sensor Networks, *VLSI Design 2001*, January (2001).
- [16] Nikolettseas, S., Chatzigiannakis, I., Antoniou, A., Euthimiou, H., Kinalis, A. And Mylonas, G. Energy Efficient Protocols for Sensing Multiple Events in Smart Dust Networks. *Proc. 37th Annual ACM/IEEE Simulation Symposium (ANSS'04)*, IEEE Computer Society Press, pp. 15 -24, 2004.
- [17] Schurgers, C., Tsiatsis, V., Ganeriwal, S. and Srivastava, M. Topology Management for Sensor Networks: Exploiting Latency and Density. In *Proc. MOBICOM'2002* (2002).
- [18] The Network Simulator ns-2. www.isi.edu/nsman/ns.