Ad Hoc Networks 8 (2010) 694–707

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

A delay-aware reliable event reporting framework for wireless sensor–actuator networks

Edith Ngai [a,*], Yangfan Zhou [b], Michael R. Lyu [b], Jiangchuan Liu [c]

[a] *Department of Information Technology, Uppsala University, Uppsala, Sweden*
[b] *Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China*
[c] *School of Computing Science, Simon Fraser University, Burnaby, British Columbia, Canada*

ARTICLE INFO

Article history:
Received 21 January 2009
Received in revised form 6 October 2009
Accepted 25 January 2010
Available online 10 February 2010

Keywords:
Sensor–actuator networks
System design
Delay-aware
Reliability
Data transport
Simulations

ABSTRACT

Wireless sensor–actuator networks (WSANs) greatly enhance the existing wireless sensor network architecture by introducing powerful and possibly even mobile actuators. The actuators work with the sensor nodes, but can perform much richer application-specific actions. To act responsively and accurately, an efficient and reliable reporting scheme is crucial for the sensors to inform the actuators about the environmental events. Unfortunately, the low-power multi-hop communications in a WSAN are inherently unreliable; frequent sensor failures and excessive delays due to congestion or in-network data aggregation further aggravate the problem.

In this paper, we propose a general reliability-centric framework for event reporting in WSANs. We argue that the reliability in such a real-time system depends not only on the accuracy, but also the importance and freshness of the reported data. Our design follows this argument and seamlessly integrates three key modules that process the event data, namely, an efficient and fault-tolerant event data aggregation algorithm, a delay-aware data transmission protocol, and an adaptive actuator allocation algorithm for unevenly distributed events. Our transmission protocol adopts smart priority scheduling that differentiates event data of non-uniform importance. We further extend the protocol to handle node and link failures using an adaptive replication algorithm. We evaluate our framework through extensive simulations; the results demonstrate that it achieves desirable reliability with minimized delay.

1. Introduction

The advances of hardware and software technologies for embedded systems have turned micro-sensors with radio transceivers into reality [1–4]. Wireless sensor networks (WSNs), constructed by a group of sensors, have been suggested for numerous novel applications, such as monitoring for harsh environments and protecting national borders. Recently, actuator nodes, which have much stronger computation and communication power than uni-purpose micro-sensors, have also been introduced [5]. An actuator can perform diverse tasks, such as processing the data reported from the sensors and accordingly interacting with the environment; a mobile actuator (e.g., a robot) could even change its location periodically to serve the application better.

The sensors and actuators can form a powerful and yet cost-effective hybrid network, that is, a wireless sensor–actuator network (WSAN). While the functionalities of the actuators are application specific, a well-designed communication model between the two types of nodes is crucial to a WSAN. In particular, given that the actuators need accurate event data from the sensors to perform corresponding actions, reliability is an important concern in the sensor–actuator communication. Unfortunately, the

* Corresponding author. Tel.: +46 70 167 9360.
E-mail address: edith.ngai@it.uu.se (E. Ngai).

1570-8705/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2010.01.004

low-power multi-hop communications often employed in a WSAN are inherently unreliable; frequent sensor failures and excessive delays due to congestion or in-network data aggregation further aggravate the problem.

In this paper, we focus on the design of a generic framework for reliable event reporting in WSANs. We argue that the reliability in this context is closely related to the delay, or the freshness of the events, and they should be jointly optimized. We also suggest that the non-uniform importance of the events can be explored in the optimization. We therefore present a delay- and importance-aware reliability index for the WSANs. Our framework seamlessly integrates three key modules to maximize the reliability index: (1) a multi-level data aggregation scheme, which is fault-tolerant with error-prone sensors; (2) a priority-based transmission protocol, which accounts for both the importance and the delay requirements of the events. A latency-oriented fault-tolerant transmission protocol is also provided as an extension to cope with transmission failures; and (3) an actuator allocation algorithm, which smartly distributes the actuators to match the demands from the sensors.

Our framework is fully distributed, and is generally applicable for diverse WSANs. Within this generic framework, we present an optimized design for each of the modules, and also discuss their interactions. The performance of our framework is evaluated through extensive simulations. The results demonstrate that our framework can significantly enhance the reliability of event reporting; it also makes more effective use of the expensive actuators.

The remainder of this paper is organized as follows: Section 2 presents related work. In Section 3, we outline our network model and the problem to be solved. The reliable event reporting framework is presented in Section 4, together with detailed descriptions of each module. In Section 5, we provide simulation results for our framework. Finally, we conclude the paper in Section 6.

2. Related work

Wireless sensor networks (WSNs) have been extensively studied recently for a wide range of applications [1–3,6]. Similar to WSNs, WSANs can be applied for environmental monitoring, forest fire detection, chemical attack detection, etc. Sensors, which detect specific events in the environment, will report to the actuators for immediate actions. Efficient and reliable event reporting is therefore an important issue in WSANs. Some related protocols are proposed for wireless ad hoc networks [7–10], but they are impractical for large scale dynamic sensor networks. For WSNs, He et al. [11] proposed a real-time communication protocol SPEED, which combines feedback control and non-deterministic quality of service (QoS) aware geographic forwarding. Lu et al. [12] described a packet scheduling policy, called Velocity Monotonic Scheduling, which inherently accounts for both time and distance constraints. Felemban et al. [13] proposed Multipath and Multi-Speed Routing Protocol (MMSPEED) for probabilistic QoS guarantee in WSNs. Multiple QoS levels are provided in the timeliness domain by using different delivery speeds, while various requirements are supported by probabilistic multipath forwarding in the reliability domain. Huang et al. [14] proposed a spatiotemporal multicast protocol, called "mobicast", which provides reliable and just-in-time message delivery to mobile delivery zones. Ergen et al. [15] presented a routing algorithm that maximizes the sensor network lifetime, and further incorporates delay guarantees into energy efficient routing by limiting the length of paths from each sensor to the collection node.

For reliable data transport with transmission failures, Aidemark et al. [16] presented a framework for achieving node-level fault tolerance (NLFT) using time-redundant task scheduling in the nodes. Ganesan et al. [17] described the use of multipath routing for energy-efficient recovery from node failures in wireless sensor networks, proposing and evaluating the classical node-disjoint multipath and the braided multipath designs. Djukic and Valaee [18] used path diversification to provide QoS, which bounds on the end-to-end delay and probability of packet loss (PPL) by transmitting packets with erasure codes and multiple paths. Jain et al. [19] considered the problem of routing in a delay-tolerant network in the presence of path failures. Jain proposed an approach that improves the probability of successful message delivery by applying a combination of erasure coding and data replication. Wang and Wu [20] studied direct transmission and flooding on delay and fault-tolerant mobile sensor network (DFT-MSN) and introduced an optimized flooding scheme that minimizes the transmission overhead of flooding. There are also related works in the general embedded or delay-tolerant network settings. For example, Khanna et al. [21] suggested that the failure of any node in a path can be detected and circumvented using backup routes. Assayad et al. [22] proposed a bi-criteria scheduling heuristic in data-flow graphs to maximize the system's reliability and minimize the system's run-time. Dubois-Ferriere et al. [23] introduced a scheme for error-correction that exploits temporal and spatial diversity through packet combining. Lou and Kwon [24] proposed a hybrid multipath data collection scheme for secure and reliable data collection in wireless sensor networks.

Our work is motivated by the above studies. The key difference is that we focus on the interactions between sensors and actuators, not uniform network nodes. In this context, additional considerations are needed to address the heterogeneous characteristics and the unique interactions within the network.

There have been studies exploring heterogenous sensor networks, e.g. [25–27], but they do not cope with the special features of actuators. For WSAN, Hu et al. [28] proposed an anycast communication paradigm. This constructs an anycast tree rooted at each event source and updates the tree dynamically according as the sinks join and leave the network. Cayirci et al. [29] offered a power-aware many-to-many routing protocol. Actuators register the data types of interest by broadcasting a task registration message; the sensors then build their routing tables accordingly. Melodia et al. [30] further presented a distributed coordination framework for WSANs based on an event-driven clustering paradigm. All sensors in the event area forward their readings to the appropriate actors by the data aggregation trees. While these works have explored the potentials of WSANs, reliability issues, and in particular, the reliability of event reporting from sensors to actuators, have yet to be addressed.

3. Network model and objective

In this section, we present an WSAN model and list our design objectives of the reliable event reporting framework.

3.1. Network model

We consider a wireless sensor–actuator network (WSAN) that consists of a collection of sensor nodes *s* and actuator nodes *a*. The field covered by this network is divided into virtual grid cells for event monitoring, as illustrated in Fig. 1. We assume that the sensors and actuators are aware of their locations, and hence, the associated grids. The location information can be obtained either through GPS [31] or various localization techniques [32–35].

Each sensor is responsible for collecting event data in its associated grid cell. Since malfunctioned sensors may give inconsistent readings, the data in the same grid cell will be aggregated to form a consistent mean value before reporting. A subset of the sensors in the field, referred to as *reporting nodes*, *v*, are responsible for forwarding the aggregated event data to the actuators for further actions. As we will show later, the aggregation occurs in a distributed manner, along with the data flow toward the reporting node *v*. Also note that the communications from the sensors to the actuators follow an anycast paradigm, that is, event reporting is successful if any of the actuators receives the report.

We focus on the reliable event data transmission from the sensors to the actuators. The corresponding actions that the actuators should perform are out of the scope of this paper, and are really application specific. It is however worth noting that, for most of such applications, strict reliability as in TCP is often not necessary and even impossible given the errors/distortions arising in aggregation and transmission; on the other hand, timely delivery is much more important, as it not only enables shorter response times for the actuators, but also implies more accurate decisions since the data are fresher.

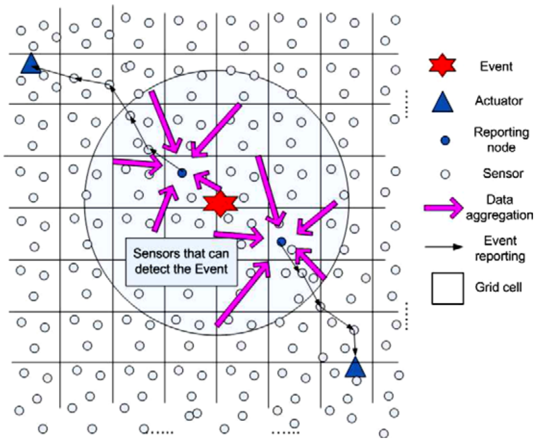

Fig. 1. An Illustration of the WSAN model and event reporting from sensors to actuators.

We thus propose a reliability index, which measures the probability that the event data are aggregated and received correctly within predefined latency bounds. Since the events may have different importance, depending on their types, urgency, and seriousness, our index and reporting framework also accommodates such differences. To realize this, each sensor in our framework maintains a priority queue, and, during transmission, important event data are scheduled with higher priorities. Beyond this differentiation in individual nodes, the queue utilization also serves as a criterion for next-hop selection in routing toward actuators (see Table 1).

3.2. Design objective

We now give a formal description of the system parameters, and our objective is to maximize the overall reliability index, $\mathbb{R}$, across all the events, as follows:

Objective

Maximize

$$\mathbb{R} = \sum_{\forall e} \left(\frac{Imp(e)}{\sum_{\forall e} Imp(e)} * r_e \right), \tag{1}$$

where $r_e = \frac{|Q_e|(1-f)}{N_e}$.

Subject to

$$D_{q_e} \leqslant B_e, \quad \forall e. \tag{2}$$

Reliable event reporting requires both accurate and timely delivery of the sensing data. The parameter Q_e represents the set of data reports of event *e* that reach the actuator within the latency constraint. However, the sensing data may be incorrect due to malfunctions of sensors. They may also have flipped bits if there are transmission failures. Aggregation failures occur only if malfunctioned sensors dominate a grid cell, while transmission failures are usually caused by node or link failures. We denote the aggregation and transmission failure rate of the reported data by *f* in our problem formulation. The reliability index r_e for an event *e* is then calculated as $\frac{|Q_e|(1-f)}{N_e}$, which represents the proportion of data reports that arrive an actuator within delay bound B_e and without failure in data

Table 1
System parameters.

Event	e
Data report of event *e*	q_e
Set of data reports of event *e* that can reach the actuator within the latency constraint	Q_e
Importance of event *e*	$Imp(e)$
Latency bound for reporting event *e*	B_e
End-to-end delay of data report q_e	D_{q_e}
Number of data reports for event *e*	N_e

aggregation and transmission. Each event is also associated with an importance index $Imp(e)$ depending on the event type, level of the event, etc. which might be application specific. The overall reliability of the system $\mathbb{R}$ is then calculated as the weighted sum of the importance of all the events and their corresponding reliability index r_e. Our objective is to maximize the overall reliability of the system in providing reliable event reporting.

4. The reliable event reporting framework

Our framework addresses the whole process for event reporting, and integrates three generic modules to achieve the above reliability objective. Specifically, when an event (e.g., a fire) occurs, the sensors located close to the event will detect it. After aggregation, which removes redundancy and inconsistent readings, the reporting nodes will forward the reports to the actuators. Such forwarding is delay- and importance-aware, implemented through prioritized scheduling and routing in each sensor. It is further enhanced to cope with transmission failures by an adaptive replication algorithm. We also provide an actuator allocation module that determines the locations of the actuators. It ensures a balanced and delay-minimized allocation of actuators to process the unevenly distributed events in the network.

Fig. 2 illustrates the workflow of our framework. When an event is detected, the sensing data will be aggregated and forwarded to the reporting nodes. The reporting nodes will then report the aggregated data to the closest actuators. Actuator allocation is adopted for more efficient coordination among the actuators. The actuators, after receiving the event data, will carry out proper reactions to the environment. We now offer detailed descriptions of the above three modules, namely, data aggregation, event reporting, and actuator allocation.

4.1. Grid-based data aggregation

In a densely deployed sensor network, multiple sensors may sense the same event with similar readings. Hence, it is preferable to aggregate the data before reporting to the actuators. Dividing the working space into grid cells provides a simple and efficient design for data aggregation. Sensors in the same grid cell are likely to provide redundant data about the environment. Hence, aggregating the sensing data in the same grid cell can reduce redundant data, so as to improve the network performance and reduce energy consumption for communications. Moreover, it can provide fault tolerance among the sensor readings.

Our grid-based aggregation algorithm works as follows (see Fig. 3). We divide the network into a number of grid cells. The size of grid cells could be determined according to the expected precision of the event location. The precision also depends on the size of the events in the network. Since the sensors in the same grid cell are expected to collect similar readings of the same event, the grid cell size should not be larger than the size of events. If users expect event location of higher precision, they can divide the area into smaller grid cells. Note that smaller grid size can provide higher precision on the event location in a dense network, though larger grid cell can reduce more data for better network performance. Moreover, it is important to ensure that the nodes in the same grid cell are connected with each other given the wireless range of devices. To balance between the precision and network performance, we could choose the maximum grid cell size that can provide enough precision of data, such that the network can still achieve good performance by reducing much redundant data.

Readings from the sensors in the same grid cell will be aggregated before reporting to the actuator. Each aggregation node is elected by the nodes in the same grid cell. For example, the node that is located closest to the sink can be selected as the aggregation node. Alternatively, the nodes in the same grid cell may take turn to be the aggregation node to balance their energy consumptions.

For each grid cell, there is an aggregating node that first collects the event data, $\langle x_1, x_2, \ldots, x_n \rangle$, and finds their median *med*. It will compare each data x_i with *med* and filter out those with a significant deviation (e.g., greater than a predefined threshold Δd). These data could be from malfunctioned sensors, which will then be blacklisted and discarded. Note that a large deviation from the average reading does not necessarily indicate of a malfunctioning

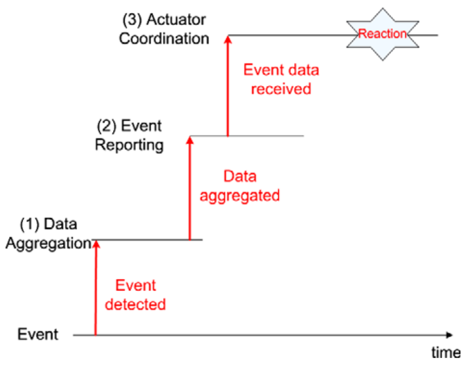

Fig. 2. Workflow of the framework.

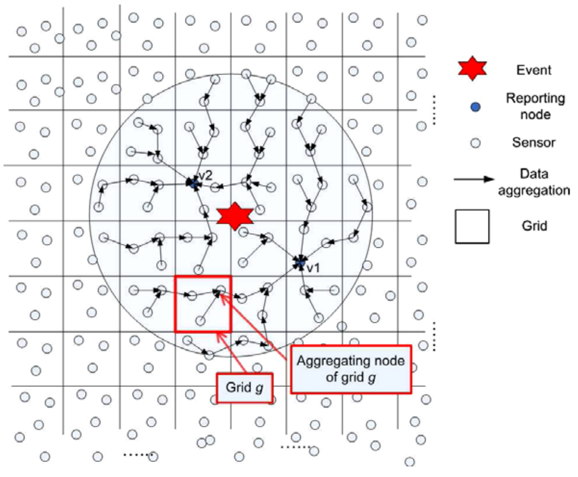

Fig. 3. Grid-based data aggregation.

sensor sometimes. However, given a reasonable size of the grid cell, the sensors in the same grid cell are likely to measure similar readings of the same event. Apart from that, we can compare the sensor readings with the neighboring grid cells to confirm the occurrence of the event. Our framework is also flexible to integrate with other existing fault detection and event detection mechanisms [36,37].

Algorithm 1. Data aggregation

```
Define: $\overline{x_g}$ as aggregated data mean of grid cell $g$;
for each sensor $s$ receive data $x_i$ do
  if multiple $x_i \in g$ and $s$ is the aggregating node then
    find the median $med$ among data $\langle x_1, x_2, \dots, x_n \rangle$;
    for each data $x_i \in g$ do
      if $|x_i - med| > \Delta d$ then
        blacklist node $i$
      end if
    end for
    $\overline{x_g}$ = mean of the un-blacklisted data $x_i \in g$
  end if
end for
```

Then, the aggregating node will calculate the mean value $\overline{x_g}$ from the remaining data in grid cell g (Algorithm 1). We consider the aggregated data to be reliable if more than half of the sensors in the grid cell are normal. The reliability for the aggregated data from grid cell g thus can be evaluated as:

$$1 - f_g = 1 - \sum_{i=\lceil (N_x+1)/2 \rceil}^{N_x} \binom{N_x}{i} (f_s)^i (1 - f_s)^{N_x - i},$$

where f_g is the failure probability of grid cell g on data aggregation, N_x is the number of nodes in grid cell g, and f_s is the fraction of sensors that are malfunctioned.

Data aggregation in this context refers to the computation of statistical means and moments, as well as other cumulative quantities that summarize the data obtained by the network. Such accumulation is important for data analysis and for obtaining a deeper understanding of the signal landscapes observed by the network. The technique can also avoid the expensive transmission of all sensing data to the actuators so as to reduce the number of messages and increase the network lifetime. Other than calculating the mean from the sensor readings, we may program the aggregation node to calculate other statistical values, including maximum, minimum, and median according to different requirements of the applications.

The aggregating node may serve as the reporting node to forward the aggregated data to actuators. The aggregation, however, can be easily extended to multiple levels, where a reporting node is responsible for further collecting and aggregating the data from the aggregating nodes in surrounding grid cells, as shown in v(Fig. 3). For the 2-level case, each aggregation node independently decides whether it will serve as a reporting node according to a probability p_v. Here, $p_v = \frac{1}{N_g}$, where N_g is the number of sensors in a grid cell. Our proposed approach is designed for balancing the energy consumption of the sensors. It is simple and easy to implement, though the random reporting node may not be the closest node to the actuator. Given grid cells of small size, the distances from different nodes in the same grid cell to the nearby actuator will be approximately the same. If large grid cells are adopted, our approach can select the closest node to the actuator as the reporting node. Other bidding algorithms for reporting node selection could be used in our framework, e.g., those in [38].

4.2. Priority-based event reporting

The routing and transmission protocol for event reporting from the reporting nodes to the actuators is the core module in our framework. The key design objective here is to maximize the number of reports reaching the destination within their latency bound, and, for different event types, to give preference to important events. To this end, we adopt a priority queue in each sensor, which plays two important roles: (1) prioritized scheduling to speed up important event data transmission and (2) queue utilization as an index for route selection to meet the latency bounds.

In our preemptive priority queues, the packets for the event data are placed according to their data importance, and each priority is served in a first-in-first-out (FIFO) discipline. The packet experiences a delay at a node as it waits to be transmitted to the link. The length of the queuing delay depends on the number of packets in the higher priority queues and the number of earlier-arriving packets that are queued and waiting for transmission across the link. The average waiting time of a packet at a node for transmission also increases when there are network contention and interference among the wireless links.

Fig. 4 shows how node i forwards packets to its neighbors j_1, j_2, and j_3. The geographical distances from j_1, j_2, and j_3 to actuator a are represented by $\|j_1, a\|$, $\|j_2, a\|$, and $\|j_3, a\|$, respectively. Only the neighbors, which provide satisfactory advancement from i to a, will be considered as the next hop. Furthermore, the queue utilization of the

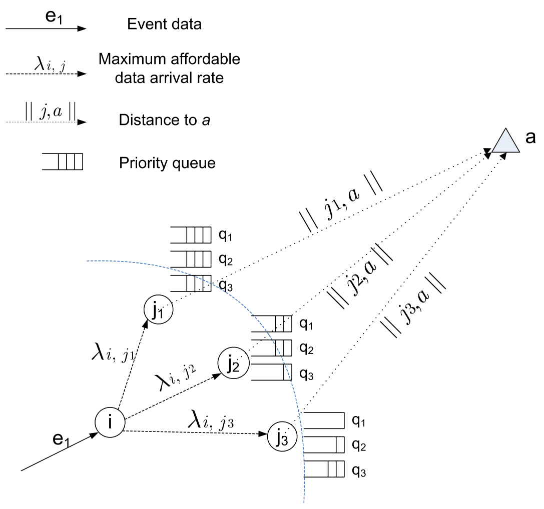

Fig. 4. Maximum affordable arrival rate from i to j.

neighbors is considered in route selection. For example, the data e_1 flowing into i has the highest priority, so it will be served by the highest priority queue q_1. Among all the neighbors of i, j_3 is selected as it provides e_1 with the best service by an empty highest priority queue q_1.

The average waiting time of a packet in the highest priority queue is $\overline{d_{q_1}} = \overline{R} + \overline{S}\overline{N_{q_1}}$, where $\overline{R} = \frac{1}{2}\sum_{k=1}^{K}\lambda_k\overline{S^2}$ is the mean residual service time in the node, $\overline{N_{q_1}}$ is the mean number of packets in the first queue, K is the number of priority queues, λ_k is the arrival rate of the packets in priority queue k, and $\overline{S}$ and $\overline{S^2}$ are the expectation and second moment of the service time of the sensor. We assume the packet arrival is Poisson. $\overline{S}$ and $\overline{S^2}$ can be obtained in each individual sensor by observing the time it takes to serve a packet.

By Little's theorem, $\overline{N_{q_1}} = \lambda_1\overline{d_{q_1}}$, and the load of priority k is $\rho_k = \lambda_k\overline{S}$, hence the waiting time of a packet in the first priority queue is:

$$\overline{d_{q_1}} = \frac{\overline{R}}{1-\rho_1}.$$

Similarly, the average waiting time of a packet in the second priority queue is:

$$\overline{d_{q_2}} = \overline{R} + \overline{S}\overline{N_{q_1}} + \overline{S}\overline{N_{q_2}} + \overline{S}\lambda_1\overline{d_{q_2}} = \frac{\overline{R} + \rho_1\overline{d_{q_1}}}{1-\rho_1-\rho_2}.$$

The average waiting time $\overline{d_{q_k}}$ of a packet in the kth priority queue is:

$$\overline{d_{q_k}} = \frac{\overline{R}}{(1-\rho_1-\cdots-\rho_{k-1})(1-\rho_1-\cdots-\rho_k)}.$$

Sensors periodically exchange control information with neighboring nodes through beacon messages or piggyback messages. A control message contains information such as waiting time and data rate to the actuators. However, the number of control messages is small compared with data messages. A node sends control messages infrequently or only if there is a sudden change of the network traffic.

When routing the event data packets, a sensor should not select a next hop that is busy in forwarding important data. On the contrary, it selects a next hop that has a smaller queueing time for the corresponding priority, or it may select a next hop that it can preempt the data packets with lower importance.

More formally, consider node i that receives a new event data $data_e$. Given the control message it has received from neighbor j, node i can obtain $\langle a, \overline{S}, \lambda_{high}, \lambda_{low}\rangle$, where a is the target actuator, $\overline{S}$ is the expected service time of node j, $\lambda_{high} = \sum_{\forall k, imp(data_k) \geqslant imp(data_e)}\lambda_k$ is the sum of all λ_k of the data that are equal or more important than $data_e$, and $\lambda_{low} = \sum_{\forall k, imp(data_k) < imp(data_e)}\lambda_k$ is the sum of all λ_k of the data that are less important than $data_e$.

Node i needs to ensure that the end-to-end latency for $data_e$ is no more than the latency bound B_e. To this end, it first estimates the advancement $h_{i,j}$ towards the actuator a from i to j, and then the maximum hop-to-hop delay from i to j, $delay_{i,j}$.

$$h_{i,j} = \frac{\|a,i\| - \|a,j\|}{\|a,i\|}.$$

So,

$$delay_{i,j} \leqslant B_e * h_{i,j}.$$

Since $delay_{i,j} = d_q + d_{tran} + d_{prop} + d_{proc}$, the maximum queueing delay $d_{q_{max}}$ is:

$$d_{q_{max}} = B_e * h_{i,j} - (d_{tran} + d_{prop} + d_{proc}).$$

Only neighbors with $d_{q_{max}} > 0$ will be considered as the next hop; otherwise the latency bound cannot be met. Among these candidates, node i starts inspecting the neighbors with both $\lambda_{low} = 0$ and $\lambda_{high} = 0$, followed by the remaining neighbors. Here, $\lambda_{low} = 0$ implies that it is not forwarding any event data with importance lower than that considering by node i; if node i forwards the data to this node, it will not affect the waiting time of the existing packets in that node; Similarly, $\lambda_{high} = 0$ means that it is not transmitting any data with higher importance, so the data from node i, if forwarded, can be served with the highest priority. For each of the candidates mentioned above, node i calculates the maximum data rate λ_i that it can forward while satisfying the latency bound:

$$d_{q_{max}} > \frac{\overline{R}}{(1-\lambda_{high}\overline{S})(1-\lambda_{high}\overline{S}-\rho_i)},$$

and

$$\rho_{i,j} < 1 - \lambda_{high}\overline{S} - \frac{\overline{R}}{(1-\lambda_{high}\overline{S})d_{q_{max}}},$$

where $\rho_{i,j} = \lambda_{i,j}\overline{S}$ is the maximum affordable load of j for handling data from i on event e.

Then the event data packets are forwarded to the neighbor with the highest $h_{i,j}$ and satisfactory $\lambda_{i,j}$, which is the closest to the destination with enough capacity for transmission. The affordable packet arrival rate $\lambda_{i,j}$ is satisfactory when it is greater than the data rate received by i. Each intermediate node updates the latency bound B_e before forwarding the packet to the next hop, according to this equation:

$$B_e = B_e - (t_{depart} - t_{arrive}) - d_{tran} - d_{prop},$$

where $(t_{depart} - t_{arrive})$ is the elapse time of the packet in a node, d_{tran} can be computed using the transmission rate and the length of the frame containing the packets, and d_{prop} is the propagation time, which is in the order of several microseconds in wireless transmission.

After the transmission starts, the sensor will update its $\overline{S}$ and the routes regularly to make sure the transmission can be completed within the latency bound. If the latency bound is not met, the sensor has to forward the packets to another route. In the worst case, if no alternative can be found, the sensor may inform the previous node to select another route in the future [38].

4.3. Coping with transmission failures

Besides excessive delays, packets might be dropped due to errors/failures in links or nodes along a transmission path. To cope with such failures, we extend the above routing algorithm by adopting adaptive packet replication in transmission. In this work, we target at coping with transmission failures due to channel fading, corrupted packets with flipped bits or bad functioning of nodes. The link capacity is supposed to be large enough to support the transmission rates with only small number of collisions in our design. Our scheme will be applied only when the replicated packet rates are affordable by the existing link capability.

We assume that each packet has a reliability requirement R_{req}, which is initialized as being proportional to its event importance. We define link loss rate $L_{i,j}$ as the packet loss rate from node i to its next hop j, and path success rate P_j as the probability that a packet from node j reaches the actuator (destination) successfully. Instead of forwarding a packet to the next hop with the highest $h_{i,j}$ and satisfactory $\lambda_{i,j}$, node i can forward the packet to multiple next hops with an adaptively determined replication factor r_f. Consider an illustration in Fig. 5, where the potential next hops for node i are $j_1,\ j_2$, and j_3, with observed link loss rates form i to $j_1,\ j_2$, and j_3 being $L_{i,j_1},\ L_{i,j_2}$, and L_{i,j_3}, respectively. The corresponding path success rates $P_{j_1},\ P_{j_2}$, and P_{j_2} from $j_1,\ j_2$, and j_3 to actuator a can accordingly be estimated. Node i may check whether using a single next hop can meet the reliability requirement R_{req}. If not, it needs to decide the replication factor r_f and forward the packet to multiple next hops. The implementation of our scheme can facilitate broadcasting from a node to multiple nodes in one time to save energy.

We now discuss the above procedure in detail. First, node i selects the top k neighbors with the highest $h_{i,j}$ with satisfactory $\lambda_{i,j}$, and estimates their link loss rates $L_{i,j}$ based on periodical feedbacks from the neighbors. Node i then estimates the path success rate P_j from i to actuator a via j as follows:

$$P_j = (1 - L_{i,j})^{1/h_{i,j}}.$$

Node i will allocate the packets to its neighbors according to their $\lambda_{i,j}$. The neighbors with higher $\lambda_{i,j}$ will be allocated with more packets to balance the load. The proportion of the packets to neighbor j, $prop_j$, is given by:

$$prop_j = \frac{\lambda_j}{\sum_{j=1}^{k} \lambda_j}.$$

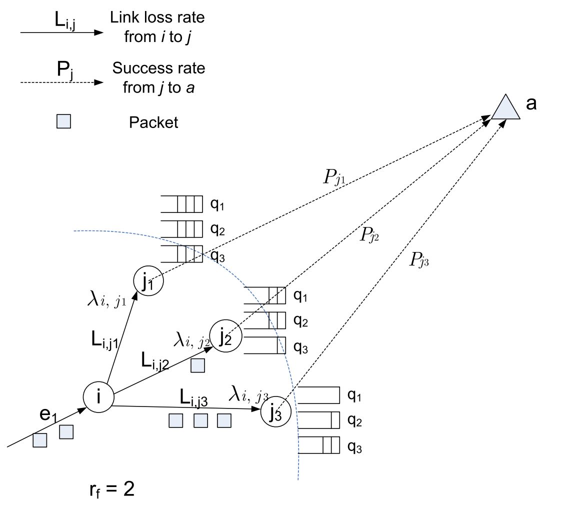

Fig. 5. Forwarding packets with replication factor $r_f = 2$.

The probability that the packet can be delivered successfully from i to actuator a by these k neighbors, P_i, can then be estimated as:

$$P_i = \sum_{j=1}^{k} \left(\frac{\lambda_j}{\sum_{j=1}^{k} \lambda_j} * P_j \right). \tag{3}$$

Then, i determines the replication factor r_f with the following equation:

$$P_i * r_f \geqslant R_{req}.$$

The replication factor r_f must be greater than R_{req}/P_i, where R_{req} is initialized as the required event reliability, or the event importance in our work, by the reporting node v. Each of the above neighbors will be allocated with proportion $prop_j$ of packets from i. The corresponding path success rate P_j will become the required reliability R_{req} of that particular path from j to the actuator.

Each node j that receives the packets will select its next hop m' with the highest $h_{j,m}$ and satisfactory $\lambda_{j,m}$. Similarly, the path success rate must be no less than R_{req}:

$$(1 - L_{j,m'})^{1/h_{h_{j,m'}}} \geqslant R_{req}.$$

If the link loss rate from j to m' satisfies the above equation, data will be forwarded to m', and it follows that:

$$(1 - \bar{L}_1)(1 - \bar{L}_2)(1 - \bar{L}_3) \cdots (1 - \bar{L}_n) > R_{req},$$

and

$$(1 - \bar{L}_2)(1 - \bar{L}_3) \cdots (1 - \bar{L}_n) > R_{req}/(1 - \bar{L}_1),$$

where the $\bar{L}_1, \bar{L}_2, \ldots, \bar{L}_n$ are the respective packet loss rates of the links on the path.

Node j then updates the reliability requirement R_{req} and forwards it with the packets to the selected neighbor m':

$$R'_{req} = R_{req}/(1 - L_{j,m'}).$$

In the case that $L_{j,m'}$ does not satisfy the required reliability, node j will look for the neighbor with the next highest $h_{j,m}$ and satisfactory $\lambda_{j,m}$. The procedure repeats until it goes through all the potential neighbors with high $h_{j,m}$ and $\lambda_{j,m}$. If no single neighbor meets the reliability requirement, node j will forward packets to multiple neighbors, as node i does [39]. If all neighbors cannot satisfy this reliability requirement, the node will send feedback message to the previous hop. The previous hop will then look for alternative neighbors. If no alternative neighbors can be found, the node will again send feedback message to its previous hop. The process is repeated until alternative paths are found or the feedback message reaches the source. Then, the source can decide to relax the reliability requirement or stop sending the messages.

We analyze the extra message overhead of the scheme as follows. Consider that a node i forwards its packets at replication factor r_f, i may send the replicated packets to multiple neighbors to distribute the load. Given that replication occurs only at i along the paths and the average path length from i to its closest actuator is H, the total number of packets transmitted by the intermediate nodes at different hops along the paths to the actuator are:

$$\begin{aligned}
&\text{1st hop}(i) = r_f \lambda_{in},\\
&\text{2nd hop} = r_f(1 - L_{i,j})\lambda_{in},\\
&\text{3rd hop} = r_f(1 - L_{i,j})^2\lambda_{in},\\
&\ldots\\
&(H-1)\text{th hop} = r_f(1 - L_{i,j})^{H-1}\lambda_{in},
\end{aligned}$$

where $L_{i,j}$ is the average link loss rate from i to neighbor j and λ_{in} is the data rate from i to the actuator. Since our scheme is applied only if there are transmission failures which cause packet loss, the extra message overhead keeps decreasing along the paths. According to the estimation in our scheme, the packets that can reach the actuator eventually are roughly at the rate of $\lambda_{in}R_{req}$. If we consider a network with no transmission failure and packet replication for comparison, the intermediate nodes along the paths will transmit at λ_{in} constantly. Our scheme indeed provides minimum extra overhead to achieve the required reliability in message delivery.

It is true that packet replication may bring extra overheads, but the replication factor is carefully selected to provide the required reliability with minimum number of redundant packets. Since the event data are aggregated before reporting, all aggregated data are important as they may report different events in the network. For applications with lower reliability requirements, however, our scheme can be relaxed to allow certain level of transmission failures, rather than guaranteeing absolute reliability.

4.4. Actuator allocation

The sensors report the occurrence of the detected events to their closest actuators. Actuators have to work collaboratively to provide fast response to the events. Given that the actuators are much more powerful than the ordinary sensors, such coordination can be achieved through direct one-hop communications using another wireless channel. Since the actuators can communicate with each other directly, packet delay from the sensors to the actuators becomes the major concern for fast event reporting and response.

Since the reports are triggered by events, we suggest that an actuator allocation be performed according to the event occurrence frequency. If the actuators are located closer to the events, the packet delay from the reporting sensors to the actuators will be shorter. Intuitively, the locations with more events should be allocated more actuators, so as to reduce the reporting distances. Such an allocation can be performed in the initial stage based on pre-estimated frequencies, or, with mobile actuators, performed periodically to accommodate event dynamics.

Algorithm 2. Actuator allocation

```
ActuatorAllocation (Field A, int ActuatorNum)
TotalFreq ← $\sum_{\forall g_i \in A} freq_{g_i}$
TmpFreq ← 0;
i ← 0;
while TmpFreq < TotalFreq/2 do
  TmpFreq ← TmpFreq + $freq_{g_i}$;
  i++;
endwhile
A1 ← $\bigcup_{k=0}^{i} g_i$;
A2 ← A − A1 ;
ActuatorAllocation(A1, ActuatorNum/2);
ActuatorAllocation(A2, ActuatorNum-ActuatorNum/2);
end ActuatorAllocation
```

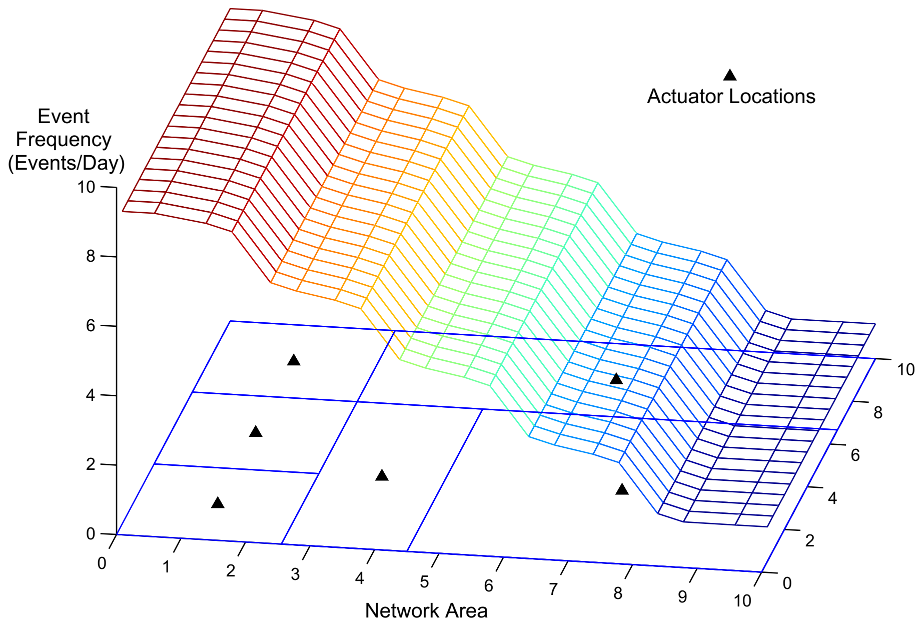

Fig. 6. Actuator allocation with 6 actuators.

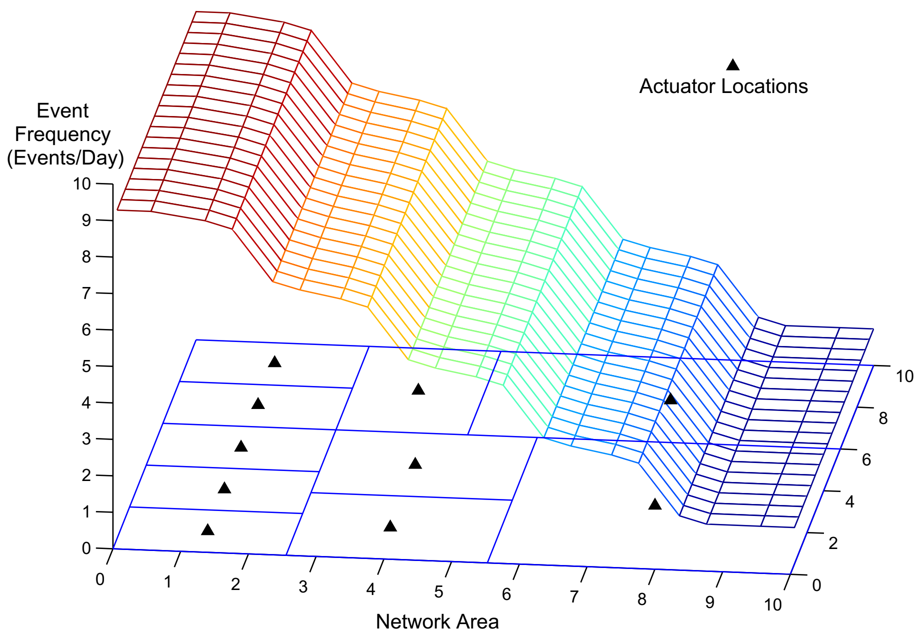

Fig. 7. Actuator allocation with 10 actuators.

Let $freq_g^t$ be the event occurring frequency of grid g at time t. For each time interval T, $freq_g^t$ will be calculated and updated by this formula:

$$freq_g^t = \alpha * N_{event}/T + (1-\alpha) * freq_g^{t-1},$$

where $freq_g^{t-1}$ is the event frequency in the previous time interval, α is a constant smoothing factor between 0 and 1, and N_{event} is the number of events happened in the current time interval.

Algorithm 2 gives an allocation that balances the load of the actuators as well as minimizing the anycast distances. In this algorithm, first, the event frequency $freq_g^t$ of all grid cells will be summed. Then, the field A will be equally divided into two, denoted by $A1$ and $A2$, according to the frequency distribution. That is, $A1$ and $A2$ have the same event occurrence frequency and each is allocated half of the actuators. The process repeats recursively for $A1$ and $A2$, until each subfield contains only one actuator.

Figs. 6 and 7 demonstrate our actuator allocation results with 6 and 10 actuators, respectively. The event frequency was measured and displayed as a curved surface in each figure. The actuator allocation results are shown on the plane of network area. In practice, the actuator allocation algorithm can be executed by one designated actuator after collecting the event frequency information. It then informs the allocation result to other actuators, which may then move to the corresponding locations.

5. Performance evaluation

We have conducted *ns-2* [40] simulations for our proposed reliable event reporting framework. The simulation settings are mainly drawn from [11], which are summarized in Table 2.

The sensors follow uniformly random distribution in a 200 m × 200 m area. Event data of high priority and low priority with constant bit rate (CBR) are generated in the network area. Sensors will aggregate the event data and report them to the actuator. There is a delay bound for delivering the data from the sensors to the actuator.

5.1. Reliability of event reporting

In the first set of experiments, we evaluate the reliability of our event reporting algorithm. To this end, we generate four events randomly in the network and vary their data rates from 10 pkt/s to 80 pkt/s. Two of the four events are high priority events with importance 1.0 (events 2 and 4), while the two are low priority events with importance 0.3 (events 1 and 3). Each packet should be reported to the actuator within the latency bound of 2 s.

We first assume that all the reports are routed to the same actuator. We fix the locations of the events and

Table 2
Simulation parameters

Network size	200 m × 200 m
No. of sensors	100
Node placement	Uniform
Radio range	40 m
MAC layer	IEEE 802.11
Packet size	32 bytes
No. of actuators	1–6
No. of concurrent events	3–10
B_e	2 s

change the seed to generate different sensor locations. Fig. 8 shows the on-time reachability of the four events with our priority-based event reporting with event importance (PREI). For comparison, we also show the result with the geographic routing protocol (GRP) [41,42], where greedy forwarding is employed and there is no differentiation regarding the event types. We can see that our PREI achieves much higher on-time reachability for the important events (event 2 and 4). The reachability for the low important events however is lower than that in GRP. This follows our design objective that important events will be served with higher priority and better quality routes. Note that, even if two different events are of the same importance, their reachabilities could be different, depending on their locations. This also happens when we compare the average delay. However, our PREI generally performs better for the same event.

Fig. 9 further shows the average delays in the PREI and GRP. It is clear that the delay in PREI is generally lower than that in GRP. This is because the PREI considers the workload of the neighbors when selecting the route. An interesting observation is that, in PREI, the average delays of the more important events are not necessarily lower than the less important events; e.g., the delay for event 1 is lower than all others, even though its importance is not high. The reason is that this event is closer to the actuator than the others. We find the average per-hop delays are generally lower for important events. Also note that the actuator allocation algorithm can mitigate this delay, as will be examined later.

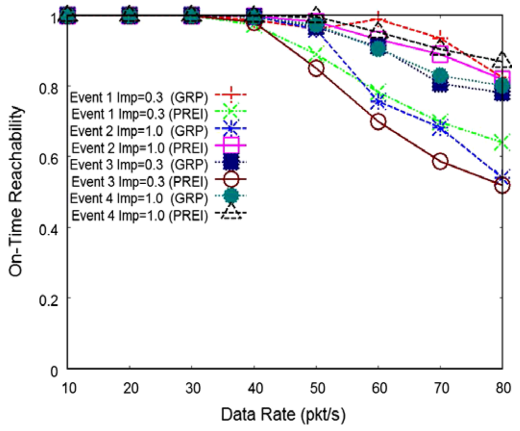

Fig. 8. On-time reachability.

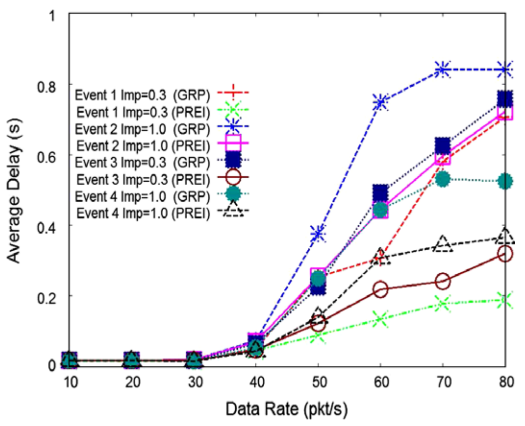

Fig. 9. Average delay.

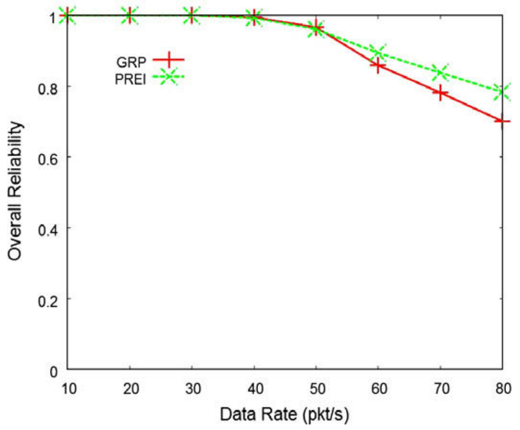

Fig. 10. Overall reliability.

Finally, Fig. 10 shows the overall reliability index, $\mathbb{R}$, of the two protocols. Again, it demonstrates that PREI outperforms GRP, and the gap increases when the data rate becomes higher.

5.2. *Coping with transmission failures*

We further study the performance of our extended routing protocol in coping with transmission failures. We fix the data rate at 15 pkt/s and vary the link failure probability f. This term means that there is a probability f for each link to encounter a transmission failure when forwarding a packet to the next hop. Two events are generated randomly in the network with the event importances 1.0 and 0.4 respectively.

Fig. 11 shows the reliability of the two events with our latency-oriented fault-tolerant data transport protocol (LOFT), which is an extension of PREI. We can see that LOFT achieves much higher event reliability than PREI for both types of events in resistant to transmission failures. Also, the reliability of the more important event (event 1) is higher than that of the less important event (event 2) in LOFT. This follows our design objective that important events should be guaranteed with higher reliability. On the contrary, the reliability of the two events are similar in the PREI, even though priority-based routing is applied. This is because PREI has no mechanism to handle link failures, and hence cannot provide any differentiation of the reliability among different events. Since the data rate of 15 pkt/s is relatively low to the network, there is no packet loss due to overload. Therefore, the reliability of both events are degraded to the same extent by link failures.

Fig. 12 further shows the average delay in LOFT and PREI. PREI performs somewhat better than LOFT when the link failure probability is low. This is because it always

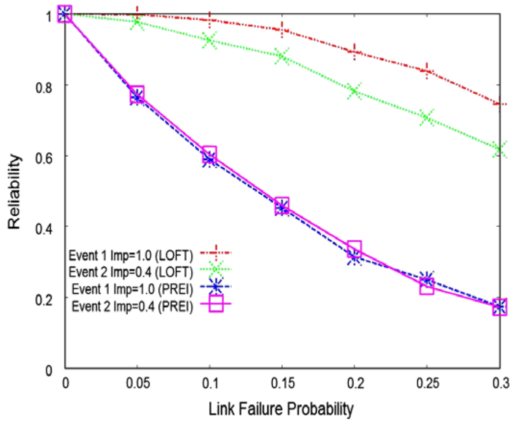

Fig. 11. Event reliability with transmission failures.

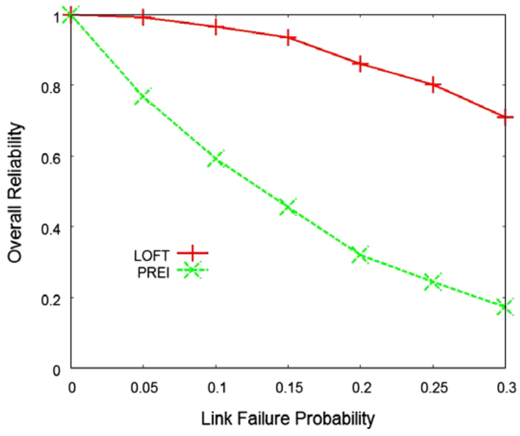

Fig. 13. Overall reliability with transmission failures.

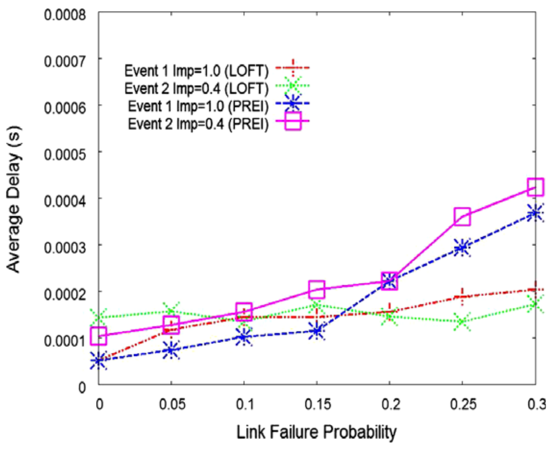

Fig. 12. Average delay with transmission failures.

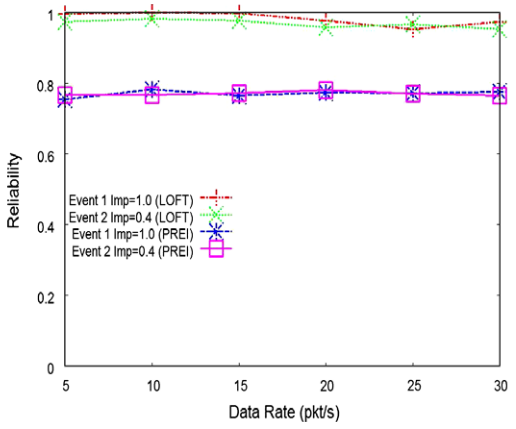

Fig. 14. Event reliability with link failure probability 0.05.

selects the next hop with the lightest workload, while LOFT estimates and considers also the link packet loss rates when selecting the route. However, it is clear that the delay in LOFT is lower than that in PREI when the link failure probability increases. The reason is that replication is applied in LOFT, so packets are routed through multiple paths. Intuitively, we would expect that it achieves a lower data delivery delay.

The overall reliability index $\mathbb{R}$ of the two protocols is shown in Fig. 13. This demonstrates the LOFT outperforms PREI, and the gap increases when the link failure probability becomes higher.

We further study the effect of data rates to the performance of our protocol. We fix the link failure probability as 0.05 and vary the data rates. Fig. 14 shows that our LOFT protocol can achieve nearly perfect reliability, while PREI can only achieve reliability close to 0.8. It also indicates that the reliability achieved is independent of the data rates.

Similarly, Fig. 15 shows that LOFT achieves small and comparable average delay with PREI. Note that, the average delay of the less important event (event 2) in LOFT increases with the data rates. It is because the traffic load of the network increases with replication under a high data rate. The queuing and transmission times may then become non-negligible for the low-priority packets. Fig. 16 again shows that the overall reliability of LOFT is higher than that of PREI.

5.3. Actuator allocation

In this experiment, we show the effectiveness of our actuator allocation algorithm. To emulate the non-uniform event occurrences, we divide the whole field into three, with the event occurrence probability 0.6, 0.333, and 0.067, respectively.

Our simulator generates events according to the above probability with data rate 60 pkt/s, and it allows different number of concurrent events in the network as represented in the *x*-axis of Figs. 17 and 18.

Fig. 17 gives the on-time reachability with different number of concurrent events. We first focus on 2 and 3 actuators only, and investigate the impact of using more actuators later. We can see from Fig. 17 that the reliability with actuator allocation outperforms that without allocation (i.e., random distribution). While the more actuators there are, the better performance we can expect, we notice that the effect of allocation is remarkable. In fact, the performance of a 2-actuator system with allocation is very close to that of 3-actuator without allocation, and even outperforms it when there are few concurrent events.

Fig. 18 shows the corresponding average delay. Not surprisingly, 3-actuator with allocation achieves the lowest delay. Similar to the on-time reachability, the delay for the 2-actuator with allocation is close to the case of 3-actuator without allocation. The results suggest that actuator

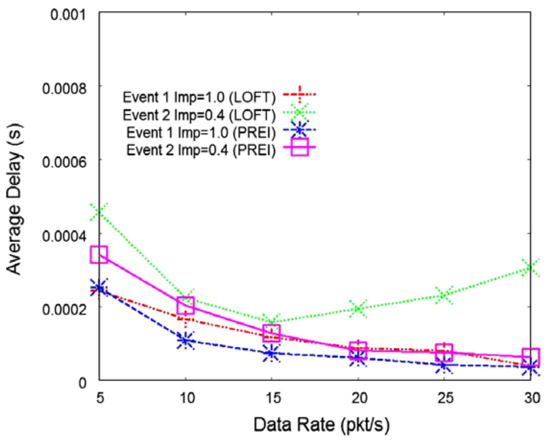

Fig. 15. Average delay with link failure probability 0.05.

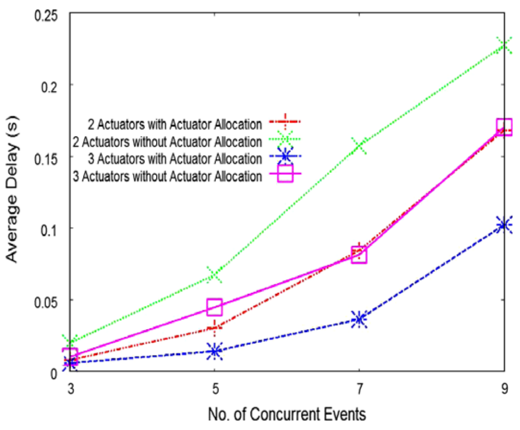

Fig. 18. Average delay with actuator allocation.

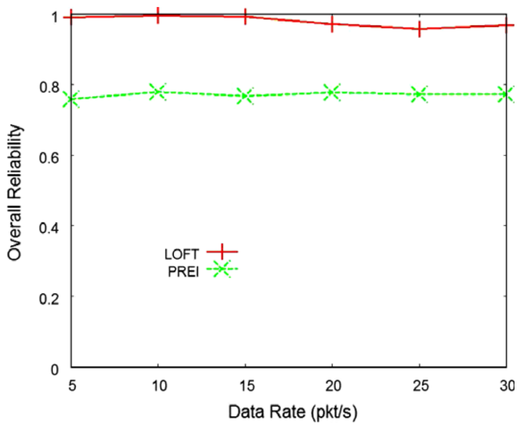

Fig. 16. Overall reliability with link failure probability 0.05.

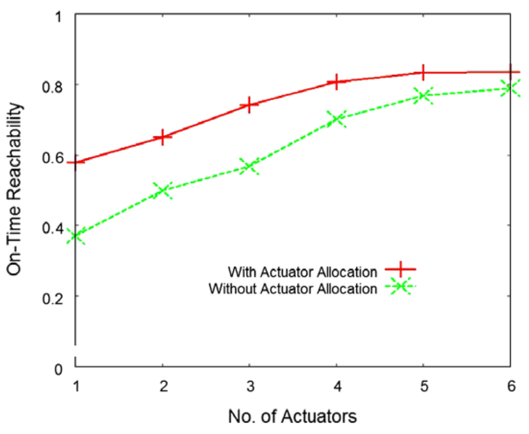

Fig. 19. On-time reachability vs. no. of actuators.

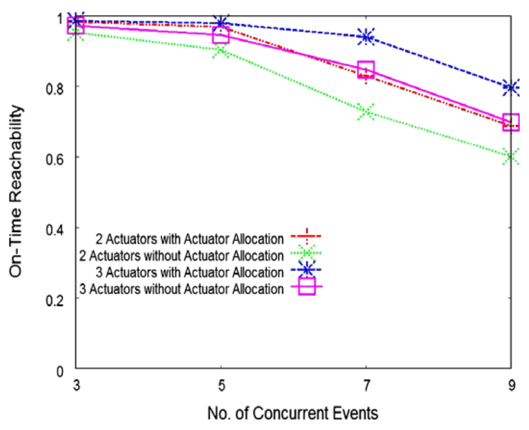

Fig. 17. On-time reachability with actuator allocation.

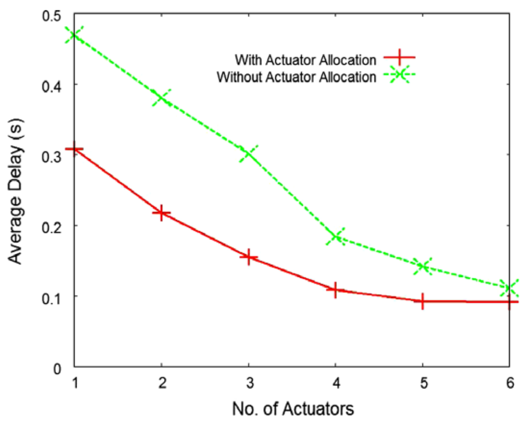

Fig. 20. Average delay vs. no. of actuators.

allocation is an effective tool for improving the efficiency of event reporting.

To further investigate the impact of the number of actuators, we fix the number of concurrent events at 10 and vary the number of actuators from 1 to 6. Fig. 19 shows the on-time-reachability as a function of the number of actuators with and without actuator allocation. Again, event reporting with actuator allocation achieves higher on-time reachability than that without actuator allocation with the same number of actuators. Intuitively, given more actuators, we can generally expect better performance, even if they are randomly deployed. This can be verified from the figure. We can see that the on-time reachability monotonically increases with more actuators, while the difference between the two schemes (with/without allocation) becomes smaller.

Similar trends can also be found in Fig. 20, which shows the average delay of event reporting as a function of the number of actuators.

6. Conclusion

In this paper, we focused on reliable event reporting from sensors to actuators in a wireless sensor–actuator network (WSAN). We argued that the reliability in this context is closely related to the delay, or the freshness of the events, and they should be jointly optimized. We also suggested that the issue of non-uniform importance of the events can be explored in the optimization. Following this argument, we proposed a general delay- and importance-aware event reporting framework. Our framework seamlessly integrates three key modules to maximize the reliability index: (1) a multi-level data aggregation scheme, which is fault-tolerant with error-prone sensors; (2) a priority-based transmission protocol (PREI), which accounts for both the importance and delay requirements of the events; A latency-oriented fault-tolerant transmission protocol (LOFT) as an extension to PREI, with the ability to cope with transmission failures; and (3) an actuator allocation algorithm, which smartly distributes the actuators to match the demands from the sensors.

Within this generic framework, we presented an optimized design for each of the modules, and also discussed their interactions. We evaluated the performance of our framework through simulations. The results demonstrated that our framework makes effective use of the actuators, and can significantly enhance the reliability of event reporting.

Acknowledgements

This work was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Project CUHK4158/08E. The work of E. C.-H. Ngai was supported by the Swedish Governmental Agency for Innovation Systems under the VINN Excellence Center WISENET and the VINNMER programme (Project No. 2009-04092). The work of J. Liu was supported by the Natural Sciences and Engineering Research Council of Canada under a Discovery Grant and a Strategic Project Grant.

References

[1] I.F. Akyildiz, W. Su, T. Sandarasubramaniam, Wireless sensor networks: a survey, Computer Networks 38 (5) (2002) 393–422.
[2] D. Estrin, R. Govindan, J. Heidemann, S. Kumar, Next century challenges: scalable coordination in sensor networks, in: Proc. of ACM MobiCom, Seattle, Washington, US, 1999.
[3] G.J. Pottie, W.J. Kaiser, Wireless integrated network sensors, Communications ACM 43 (5) (2000) 551–558.
[4] E.C.-H. Ngai, M.R. Lyu, J. Liu, A real-time communication framework for wireless sensor–actuator networks, in: Proc. of the IEEE Aerospace Conference, Big Sky, Montana, US, March 2006.
[5] I.F. Akyldiz, I. Kasimoglu, Wireless sensor and actor networks: research challenges, Ad Hoc Networks 2 (4) (2004) 351–367.
[6] C. Micheloni, G.L. Foresti, C. Piciarelli, L. Cinque, An autonomous vechicular for video surveillance of indoor environments, IEEE Transactions on Vechicular Technology 56 (2) (2007) 487–498.
[7] T. Chen, J. Tsai, M. Gerla, Qos routing performance in multihop multimedia wireless networks, in: Proc. IEEE Sixth Int. Conf. Universal Personal Commun., 1997, pp. 557–561.
[8] R. Sivakumar, P. Sinha, V. Bharghavan, Cedar: core extraction distributed ad hoc routing algorithm, IEEE Journal on Selected Areas in Communications 17 (8) (1999) 1454–1465.
[9] S. Chen, K. Nahrstedt, Distributed quality-of-service routing in ad hoc networks, IEEE Journal on Selected Areas in Communications 17 (8) (1999) 1488–1505.
[10] B. Hughes, V. Cahill, Achieving real-time guarantees in mobile ad hoc wireless networks, in: Proc. of the 24th IEEE Real-Time Systems Symp. (RTSS 2003), December 2003.
[11] T. He, J. Stankovic, C. Lu, T. Abdelzaher, SPEED: a real-time routing protocol for sensor networks, in: Proc. of the IEEE ICDCS, Providence, RI, US, May 2003, pp. 46–55.
[12] C. Lu, B.M. Blum, T.F. Abdelzaher, J.A. Stankovic, T. He, RAP: a real-time communication architecture for large-scale wireless sensor networks, in: Proc. of the IEEE RTAS, San Jose, CA, US, September 2002.
[13] E. Felemban, C.-G. Lee, E. Ekici, MMSPEED: multipath multi-speed protocol for QoS guarantee of reliability and timeliness in wireless sensor networks, IEEE Transactions on Mobile Computing 5 (6) (2006) 738–754.
[14] Q. Huang, C. Lu, G.-C. Roman, Mobicast: just-in-time multicast for sensor networks under spatiotemporal constraints, in: Proc. of the 2nd International Workshop on Information Processing in Sensor Networks, 2002, pp. 442–457.
[15] S.C. Ergen, P. Varaiya, Energy efficient routing with delay guarantee for sensor networks, ACM Wireless Networks 13 (5) (2007) 679–690.
[16] J. Aidemark, P. Folkesson, J. Karlsson, A framework for node-level fault tolerance in distributed real-time systems, in: Proc. of the IEEE DSN, Yokohama, Japan, June 28 – July 1, 2005.
[17] D. Ganesan, R. Govindan, S. Shenker, D. Estrin, Highly-resilient, energy-efficient multipath routing in wireless sensor networks, Mobile Computing and Communication Review 1 (2) (2001).
[18] P. Djukic, S. Valaee, Reliable packet transmissions in multipath routed wireless networks, IEEE Transactions on Mobile Computing 5 (5) (2006) 548–559.
[19] S. Jain, M. Demmer, R. Patra, K. Fall, Using redundancy to cope with failures in delay tolerant network, in: Proc. of the ACM SIGCOMM, Pennsylvania, US, August 2005.
[20] Y. Wang, H. Wu, DFT-MSN: the delay fault tolerant mobile sensor network for pervasive information gathering, in: Proc. of the IEEE Infocom, 2006.
[21] G. Khanna, S. Bagchi, Y.-S. Wu, Fault tolerant energy aware data dissemination protocol in sensor networks, in: Proc. of the IEEE DSN, Florence, Italy, June 28 – July 1, 2004.
[22] I. Assayad, A. Girault, H. Kalla, A bi-criteria scheduling heuristic for distributed embedded systems under reliability and real-time constraints, in: Proc. of the IEEE DSN, Florence, Italy, June 28 – July 1, 2004.
[23] H. Dubois-Ferriere, D. Estrin, M. Vetterli, Packet combining in sensor networks, in: ACM Sensys, San Diego, California, US, November 2005.
[24] W. Lou, Y. Kwon, H-SPREAD: a hybrid multipath scheme for secure and reliable data collection in wireless sensor networks, IEEE Transactions on Vehicular Technology 55 (4) (2006) 1320–1330.
[25] R.K. Sahoo, A. Sivasubramaniam, M.S. Squillante, Y. Zhang, Failure data analysis of a large-scale heterogeneous server environment, in: Proc. of the IEEE DSN, Florence, Italy, June 28 – July 1, 2004.
[26] V.P. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, N. Shroff, A minimum cost heterogeneous sensor network with a lifetime constraint, IEEE Transactions on Mobile Computing 4 (1) (2005).

[27] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, S. Singh, Exploiting heterogeneity in sensor networks, in: Proc. of the IEEE Infocom, Miami, FL, US, March 2005.
[28] W. Hu, S. Jha, N. Bulusu, A communication paradigm for hybrid sensor/actuator networks, in: Proc. of the 15th IEEE Intl. Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Bacelona, Spain, September 2004.
[29] E. Cayirci, T. Coplu, O. Emiroglu, Power aware many to many routing in wireless sensor and actuator networks, in: Proc. of the 2nd European Workshop on Wireless Sensor Networks (EWSN), Istanbul, Turkey, 31 January – 2 February 2005, pp. 236–245.
[30] T. Melodia, D. Pompili, V.C. Gungor, I.F. Akyildiz, A distributed coordination framework for wireless sensor and actor networks, in: Proc. of ACM Mobihoc, Urbana-Champaign, IL, US, 2005, pp. 99–110.
[31] J.G. McNeff, The global positioning system, IEEE Transactions on Microwave Theory and Techniques 50 (March) (2002) 645–652.
[32] L. Hu, D. Evans, Localization for mobile sensor networks, in: Proc. of ACM Mobicom, Philadelphia, PA, US, 26 September – 1 October 2004, pp. 99–110.
[33] A. Savvides, C.C. Han, M.B. Srivastava, Dynamic fine-grained location in ad hoc networks of sensors, in: Proc. of ACM Mobicom, Philadelphia, PA, US, 2001, pp. 166–179.
[34] T. He, C. Huang, B.M. Blum, J.A. Stankovic, T. Abdelzaher, Range-free localization schemes for large scale sensor networks, in: Proc. of ACM Mobicom, San Diego, CA, US, 2003, pp. 81–95.
[35] D.K. Goldenberg, P. Bihler, Y.R. Yang, M. Cao, J. Fang, A.S. Morse, B.D.O. Anderson, Localization in sparse networks using sweeps, in: Proc. of ACM Mobicom, 2006, pp. 110–121.
[36] S. Zahedi, M. Szczodrak, P. Ji, D. Mylaraswamy, M. Srivastava, R. Young, Tiered architecture for on-line detection, isolation and repair of faults in wireless sensor networks, in: Military Communications Conference, 2008. MILCOM 2008. IEEE, November 2008, pp. 1–7.
[37] T. Banerjee, B. Xie, D.P. Agrawal, Fault tolerant multiple event detection in a wireless sensor network, Journal of Parallel and Distributed Computing 68 (9) (2008) 1222–1234.
[38] E.C.-H. Ngai, Y. Zhou, M.R. Lyu, J. Liu, Reliable reporting of delay-sensitive events in wireless sensor–actuator networks, in: Proc. of the IEEE MASS, Vancouver, Canada, October 2006.
[39] E.C.-H. Ngai, Y. Zhou, M.R. Lyu, J. Liu, LOFT: a latency-oriented fault tolerant transport protocol for wireless sensor–actuator networks, in: Proc. of the IEEE Globecom, Washington, DC, US, November 2007.
[40] K. Fall, K. Varadhan, The ns manual, December 2003, <http://www.isi.edu/nsnam/ns>.
[41] B. Karp, H. Kung, GPSR: greedy perimeter stateless routing for wireless networks, in: Proc. of ACM MobiCom, Boston, Massachusetts, US, 2000.
[42] H. Frey, I. Stojmenovic, On delivery guarantees of face and combined greedy-face routing algorithms in ad hoc and sensor networks, in: Proc. of ACM Mobicom, Los Angeles, US, September 2006, pp. 390–401.

Edith C.H. Ngai is currently an Assistant Professor in Department of Information Technology, Uppsala University, Sweden. She received her B.Eng, M.Phil., and Ph.D. from Department of Computer Science and Engineering, The Chinese University of Hong Kong. She did her post-doc in Department of Electrical and Electronic Engineering, Imperial College London, United Kingdom. Her research interests include wireless sensor networking, mobile computing, network security, QoS routing, and video information processing. Previously, she has conducted research in VIEW Laboratory, CUHK, Hong Kong, Network Modelling Lab, Simon Fraser University, Vancouver, Canada, and Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing, China, and the Intelligent Systems & Networks (ISN) Group, Imperial College London, United Kingdom, and the Networked & Embedded Systems Laboratory (NESL) at UCLA.

Yangfan Zhou is currently a postdoctoral research associate in Department of Computer Science and Engineering, The Chinese University of Hong Kong (CUHK). He received an MPhil and a PhD from CUHK, and a BSc from Peking University. He is interested in distributed computing and networking particularly in sensor networks, Web services, cloud computing, and cyberphysical systems. His current research is on their software engineering issues (e.g., fault management, fault tolerance, reliability engineering, testing, and debugging). Before joining CUHK, for many years he has been working as an engineer in information technology industry, where he is now also active in technology consulting.

Michael R. Lyu is currently a Professor in the Computer Science and Engineering department of the Chinese University of Hong Kong. He worked at the Jet Propulsion Laboratory as a Technical Staff Member from 1988 to 1990. From 1990 to 1992 he was with the Electrical and Computer Engineering Department at the University of Iowa as an Assistant Professor. From 1992 to 1995, he was a Member of the Technical Staff in the Applied Research Area of the Bell Communications Research, Bellcore. From 1995 to 1997 he was a research Member of the Technical Staff at Bell Laboratories, which was first part of AT&T, and later became part of Lucent Technologies. Dr. Lyu's research interests include software reliability engineering, distributed systems, fault-tolerant computing, web technologies, mobile networks, digital video library, multimedia processing, and video searching and delivery.

Jiangchuan Liu is an Associate Professor in the School of Computing Science at Simon Fraser University, British Columbia, Canada. From 2003 to 2004, he was an Assistant Professor in the Department of Computer Science and Engineering at The Chinese University of Hong Kong. He was also a Microsoft Research Fellow, and worked at Microsoft Research Asia (MSRA) in the summers of 2000, 2001, 2002, and 2007. His research interests are in networking, in particular, multimedia communications, overlay/peer-to-peer networking, and wireless sensor/mesh networking. He is an Associate Editor of IEEE Transactions on Multimedia