A Data Fusion Protocol for WSN Performance and Data Retrieval

Vasco Pereira, Edmundo Monteiro, Jorge Sá Silva
CISUC - Department of Informatics Engineering – University of Coimbra
Coimbra, Portugal
e-mail: {vasco, edmundo, sasilva}@dei.uc.pt

Abstract—The specific capabilities of Wireless Sensor Networks (WSNs), such as fast deployment and flexibility, together with the low cost solutions that can be achieved, bring new opportunities for an all-new range of applications, far from the typical scenarios of low requirements and high redundancy. Some of these new scenarios involve the assurance of predefined performance levels. To guarantee that these goals are achieved, a careful constant monitoring must exist. However, this monitoring depletes nodes energy (especially in middle nodes that have to retransmit the information to the sink), increases interference between nodes and causes an overhead in the used bandwidth, a foreseeable result if measuring performance of a large multilevel WSN. In the protocol to be presented, data fusion is used to enhance the benefits of continuous monitoring while minimizing its implied overhead, using the specific characteristics of WSNs in its favor.

Keywords—fusion; wireless sensor networks, monitoring

I. INTRODUCTION

The use of Wireless Sensor Networks (WSNs) in performance controlled scenarios implies not only that the network must assure a pre-determined quality of service (QoS) level, to provide for application specific demands, but also that the QoS must be maintained over time. The latter implies the use of continuous performance monitoring and of alerts that notify potential malfunctions. The typical implementation of network monitoring involves the retrieval of performance data from all the nodes to a central unit. As data is (most of the times) delivered in a multi-hop fashion [1], each node being responsible for the transmission of all the data from their children, that creates huge amounts of data that must be retransmitted through the network, wasting not only valuable energy but also creating substantial overhead over the wireless medium. At the same time, monitoring will, by itself, degrade the performance of the network, as normal sensed data will have more difficulty to be transmitted.

While many protocols have already been created to reduce the amount of regular sensed data that needs to be transmitted, their application to performance has problems, as performance does not have the same needs as regular data nor its information can be condensed the same way. Performance monitoring demands that different metrics use different fusion functions, the use of alerts and a specific cadence of reporting. Moreover, original performance data is not needed most of the times if performance levels are guaranteed. In order to deal with the specificities of performance monitoring and retrieval in WSNs with controlled performance, including those that operate in critical scenarios (e.g. industrial and health) a new protocol is now proposed. This protocol merges different techniques, including data fusion, collective metrics, healthy metric intervals and an alert system. The result is a protocol that permits the continuous collection of performance data from all nodes with reduced overhead, allowing for the natural fluctuations in performance values that result from the specific nature of WSNs and providing for an immediate notification of the sink whenever a malfunction is detected.

This paper has 5 sections. Following Introduction, Section II describes work already done in the collection of data and in the monitoring of WSNs. Section III describes the proposed protocol and Section IV evaluates it using simulation. Last section presents the conclusions.

II. RELATED WORK

Some work has already been done in networks that perform data analysis and use aggregation operations before transmitting data. In this section the focus is not in the multiple works using aggregation and fusion to collect regular sensed data but in the works that more closely relate to performance retrieval.

Directed Diffusion [2] is a data-centric approach to the dissemination of data in a WSN. The results are transmitted node by node and can be aggregated in order to save energy. Directed diffusion is not suited for networks with a permanent broadcast of data, nor to use specific aggregates for each type of data, being more useful in a query-driven model. TAG [3] (Tiny AGgregation) is a generic aggregation service for ad-hoc networks. It provides a declarative interface for data collection and aggregation inspired by database query languages and distributes those queries in the network. This protocol is suited for generic data and only works in a query-driven data-delivery model. In [4] Zhao et al. describe an architecture for sensor network monitoring and propose a protocol to continuously compute aggregates of network properties (e.g. loss rates, energy levels, packet counts). The protocol uses digest functions whose input is the contribution value of each node. These digests are continuously diffused throughout the all network and allow for all nodes to have a constant knowledge of the network. In spite of saving energy and allowing for a general knowledge of specific network properties, the proposed approach is not indicated for critical applications as it relies on changing spanning trees, does not deliver data towards a base station nor has any alert mechanism in case a critical value is detected in a specific...
network property. Several monitoring tools such as Sympathy [5], Memento [6] or DiMo [7], were also proposed to address the problem of monitoring WSNs and detect failures. In general these tools use poor aggregation and do not provide for a constant reporting of the network performance.

Collective metrics are a new set of metrics that result of the fact that information may not be generated from a single node, but from the contribution of many nodes. As an example, collective loss would be the total number of lost packets in the network (or in a specific group of nodes) in a specified time interval [8].

III. PROTOCOL DESCRIPTION

The proposed protocol has 3 main features. The first is that each node periodically sends fused performance messages that contain the state of the network branch for which it is the top node. Each fusion message contains one or more collective metrics, together with information about the samples it contains (i.e., the number of individual values that form the fused metric) and the number of fusion operation it has undergone. The second is that if an out-of-bound metric value is detected in a node (a value out of the range of predefined healthy values for the metric), the value of that metric is not sent together with others in the fused message, but uses a specific alert message. Also, specific alerts are produced if a node has not received a message from one of its neighbors for a specified maximum time, informing the sink that a node has died or has some kind of malfunction. The third feature is that at any time the sink may query any node about the value of any metric. This feature is essential to debug.

Fusion can be performed in all nodes or in selected nodes, depending on the nodes where the protocol is active for a specific metric. This is controlled during deployment and can be changed after by the sink (through setup messages). In all cases there is no need to select cluster-heads or specific nodes to make fusion, therefore reducing the maintenance overhead in the network. Fusion is always done where available, and always in the path to the sink.

A. Setup

The protocol starts by defining the metrics to be calculated in nodes, as well as their healthy ranges (the acceptable values that the metric may have and that translate the performance requirements of a specific WSN or application). The definition of the healthy values may be done automatically or explicitly. Specifics on how to determinate healthy ranges are outside of the scope of the protocol and depend on the specific WSN applications used and existing environment. If done automatically, the sink sends a node a message that requires the value of specific metrics. For this method to be successful, the network should be in a state that mimics the best-expected operation conditions. After collecting all responses from nodes, the sink defines (with a predefined margin), the acceptable ranges for each metric. This method does not impose the values to a network; it defines healthy ranges taking into consideration the existing WSN characteristics. If the definition of healthy values is done explicitly, specific values are sent by the network administrator to all nodes, based on the requirements of the application to run. Also, during setup, the reporting interval (time between performance updates), warnings tolerated (number of times a metric may be out of the healthy interval without an alarm being sent, to accommodate one-time fluctuations of metric values that represent transient situations) and fusion settings are initialized. While the initial setup is done at network setup, it can also be changed dynamically during the network operation. After setup the protocol is ready to begin the performance retrieval.

B. Operation

Once in operation, each node using the proposed protocol may receive data from upstream nodes (nodes closer to the sink) and from downstream nodes (nodes farther from the sink). Upstream nodes deliver setup messages and topology change messages to their downstream neighbors, as well as requests for specific sink metric queries. On receiving a setup or topology change message from an upstream node, each node modifies its own Management Information base (MIB) to reflect the changes (MIB in this context is used generically as a management table). On receiving a specific metric request, the node calculates the requested metric and immediately delivers it to the sink. On receiving a message from one of its 1-hop downstream neighbors, a node always saves the time of the packet reception, as each node is responsible for the detection of the death or malfunction of its direct neighbors. From all the messages received from downstream neighbors, only metric report messages of fused metrics are treated (METRIC_REPORT_FUSION packets). All other messages are just forwarded. On receiving a metric report message with fused data, the node reads its contents and saves them in its MIB (Fig. 1).

1 procedure process_packet_from_child()
2 if packet_source_is_a_one-hop_neighbor
3 update_MIB_with_last_contact_date()
4 endif
5 if packet_type == METRIC_REPORT_FUSION
6 update_MIB_with_metrics_data()
7 else
8 forward_received_packet_to_sink()
9 endif
10 end procedure

Fig. 1 - Reception of packets from downstream nodes.

At regular intervals (UPDATE_TIME), defined in the setup, each node calculates its own metrics and validates if they are within the expected healthy interval (Fig. 2).

1 procedure update_report()
2 while(TRUE)
3 wait(UPDATE_TIME)
4 for i = first_metric to last_metric
5 value = calculate_individual_metric(i)
6 if limits_crossed()
7 send_ALERT()
8 else
9 update_MIB_with_metric_data(i, value)
10 endif
11 Next i
12 Calculate_fused_metrics()
13 send_METRIC_REPORT_FUSION()
14 end
15 end procedure

Fig. 2 – Metrics update report.
If out-of-bounds metrics are found (and if the warning threshold was surpassed) an alert is generated and immediately send to the sink. Also, the time of last contact of all 1-hop neighbors is checked. If the last contact surpasses the defined threshold, an alert message is generated and immediately sent to the sink. Finally, it reads the values of the metrics to be fused that were received from other nodes, fuses them with its own metrics and delivers a \texttt{MERIC\_REPORT\_FUSION} packet to its upstream neighbor.

IV. EVALUATION

In order to assess the capabilities of the proposed protocol, simulation tests were accomplished.

A. Simulation details

A network of 32 nodes using a hierarchical tree was used. The logical topology is depicted next (Fig. 3).

![Simulation topology](image)

Fig. 3 - Simulation topology.

The simulation was done using the Cooja simulator [9] emulating 32 Tmote Sky nodes, each with a transmission range set to 40m. The area simulated was of 240x120m. All nodes report to the same sink (node 0) hierarchically. While in a real scenario interference may allow for the reception of a packet, in Cooja, overlapping packets always cause a packet drop, what potentiates the number of losses in networks with congestion. ContikiMAC Radio-Duty Cycle Protocol [10], which proposes some enhancements over X-MAC, was used. It has periodical wake-ups to listen for packets from neighbors, while remaining with radio off the rest of the time to save energy. The Framer layer uses “Framer-802154” a ContikiOS implementation compatible with standard IEEE 802.15.4 (2003). The ContikiOS Rime stack [11], which is a set of custom lightweight networking protocols designed specifically for low-power wireless networks is also used. The transmission used in not reliable.

In the simulation, regular data packets (that simulate sensed data) contain 12 bytes of data. The \texttt{METRIC\_REPORT\_FUSION} packets, which contain the performance metrics, carry for simulation purposes 3 metrics (energy, delay per-hop and number of data packets already sent). Performance packets and regular data packets are generated and sent by all nodes in the network (with the exception of the sink) with a cadence of one performance packet and one data packet generated and sent every 6 seconds. This cadence is the same along all simulations for an easy comparison between them. For fusion purposes, averages were calculated.

Four different strategies are compared in the simulation. The first (\textit{Metrics fused+data}) uses the fusion protocol for performance retrieval and also includes an independent feed of regular data from all nodes in the topology. The second (\textit{Metrics fwd+data}) uses dedicated individual packets to send performance data and there is also a constant feed of data from all nodes in the topology. Each of these packets is just forwarded to the sink, with no processing done in any of the intermediate nodes. In the third (\textit{Metrics&data fused}), performance data and regular data transmissions are done using the same packets. Both performance and regular data are fused along the path to the sink. Finally, the fourth strategy (\textit{Metrics&data fwd}) is the same as the third but in this case no fusion is done, packets containing both performance data and regular data are just forwarded to the sink.

B. Simulation

Tests were made to compare the energy spent and the number/type of packets generated. 70 cycles of performance reporting were simulated in each test made, and an average of the results is presented. Reliability is also addressed.

The average total energy spent during the simulation time is showed in Fig. 4, together with the maximum, minimum and standard deviation of the set of tests made. As can be seen, the approach with better results is \textit{Metrics&data fused}, fusion of both performance and regular data. The worst option is to use \textit{Metrics fwd+data}, which sends performance and regular data in different packets, both not fused along the network. It is also clear that the two first strategies had a worse behavior than the last two. This result is due to the fact that the first two strategies send regular data separated from performance data, producing more traffic and consequently also more congestion and interference, resulting in worse global results. An individual node analysis showed that the energy spent is higher in nodes closer to the sink. This tendency happens independently of the strategy used. However, as nodes get closer to the sink, the difference between the strategies widens. The justification of the differences relies in the number of packets generated and forwarded by each strategy (more packets demand more energy for radio transmission) and also on the fact that by having more packets being transmitted, the interferences are also higher, leading to frequent aborted transmissions. The number of packets per node is analyzed in Fig. 5.

![Total energy spent in the network](image)

Fig. 4 - Total energy spent in the network by strategy used (average).

Lost packets presented are the sum of packets sent originally from the referenced node that did not arrive destination. In fact, most of the drops that happen in the leaf
nodes of the topology are in fact losses that occurred while other nodes were forwarding the messages and not by the actual transmission made by original sending nodes.

The average reliability of the network was also calculated, based on the samples delivery rate, as strictly considering the packets would be unfair. In fact, when using the fusion protocol most of the transmissions are made between 2 neighbor nodes (only alerts go directly to the sink), what makes it easier to prevent losses, especially in network areas with less traffic congestion. When using strategies that do not include the fusion protocol, all messages are sent end-to-end (from the original sender to the sink), being subject to problems along the all path to the sink. So, instead of comparing the raw values of packets sent, received and lost, the comparison will be made considering the information that actually arrives to the sink, the one that effectively counts for the monitoring of the network. In this case, all the reception of packets by intermediate nodes will not be considered. As each metric report packet contains the number of samples that constitutes each metric reported, this value will be used to compare the effective report of metrics that reach the sink. In fact, each sample can be counted as information from a different node, which was fused along the path to the sink. Alerts and packets of regular data are already counted as end-to-end. The average number of samples (including alerts and regular data) sent and lost in the network is presented next (Fig. 6).

The averages obtained for reliability were 60.65% for Metrics fused+data, 44.15% for Metrics fwd+data, 86.70% for Metrics&data fused and 45.66% for Metrics&data fwd. This difference remains stable across all tests (standard deviation, maximum and minimum values obtained, are all in a close range).

C. Analyss

The proposed protocol is both energy and packet-efficient, when compared to a solution that implies the transmission of individual performance concurrently with regular data, or performance data within regular data packets. By reducing the packets needed to transport the metrics it contributes to lower interferences and fewer retransmissions due to collisions, which not only save energy but also contribute to reduce the performance degradation that measuring the performance necessarily brings to the network. The protocol can be used in two ways. The first implies the use of performance packets fused along the network and is suited to scenarios where not all nodes send regular data, or where all nodes send regular data but at a different cadence of performance update needs, or when regular and performance packets report to different sinks. The second extends the functionality of metrics transmission to regular data, by allowing performance and regular data to be sent within the protocol packets, being suited to scenarios where performance and regular data share the same cadence of reporting and similar fusion needs. Both ways provide for a continuous monitoring of the global performance of the network while enabling the control of specific individual metric values through the use of alerts. A drawback of the protocol is that the performance retrieved does not give a picture of the global network in a specific time, as the samples that arrive at the sink are from different moments, the moments when they were collected in their source nodes. Also, in order to fuse metrics from different nodes at each hop, some relay time is due. The relay imposed does not affect any alerts, only metrics within healthy bounds. The relay is also dependent both on the cadence that is set in the protocol and on the radio duty cycle of the node. However, the drawback identified is not a problem in real deployments as the network manager still has a constant knowledge of the global performance of the network, being assured that if a malfunction is detected an alert is immediately generated and sent to the sink. By using the proposed protocol, an update of the global performance of the network is received at each metric reporting interval.

V. CONCLUSION

In this paper, the collection of performance data was focused. A new protocol to deal with the specificities of WSNs monitoring, using data fusion, was proposed. The protocol was specified and evaluated using simulation. When compared to solutions that use no fusion it presents better results both in the energy used as in the number of packets generated.

REFERENCES