Supple: A Flexible Probabilistic Data Dissemination Protocol for Wireless Sensor Networks

Aline Carneiro Viana [1,4], Thomas Herault [1,2,3], Thomas Largillier [1,2,3], Sylvain Peyronnet [1,2,3], Fatiha Zaïidi [2,3]

[1] INRIA Saclay - Ile de France, France;
[2] Univ Paris-Sud, LRI, UMR 8623, Orsay, F-91405, France;
[3] CNRS, Orsay, F-91405, France;
[4] Technical University Berlin, TKN, Germany.
aline.viana@inria.fr,{herault, largillier, syp, zaidi}@lri.fr

ABSTRACT

We propose a flexible proactive data dissemination approach for data gathering in self-organized Wireless Sensor Networks (WSN). Our protocol Supple, effectively distributes and stores monitored data in WSNs such that it can be later sent to or retrieved by a sink. Supple empowers sensors with the ability to make on the fly forwarding and data storing decisions and relies on flexible and self-organizing selection criteria, which can follow any predefined distribution law. Using formal analysis and simulation, we show that Supple is effective in selecting storing nodes that respect the predefined distribution criterion with low overhead and limited network knowledge.

Categories and Subject Descriptors

B.8.2 [**Performance and Reliability**]: Performance Analysis and Design Aids; C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Network topology, Distributed networks*; C.2.2 [**Network Protocols**]

General Terms

Algorithms, Design, Simulation, Theory

Keywords

Free sink trajectories, mobile sinks, proactive dissemination

1. INTRODUCTION

In this paper, we narrow our focus on zone monitoring wireless sensor network (WSN) applications: a large number of sensors is deployed to collect data or events in a specified geographic area. These applicabilities are interesting in wildlife observation, intruder detection, and meteorological surveillance, and usually require autonomy, cooperation, and self-organization capabilities from the sensor network. Collected data by sensors are later gathered by an entity called sink (a node with no resource limitation), which will store and process the whole network data. A sink can be static or mobile. In the former case, connectivity to at least one sink in the network has to be assured in order to guarantee a good information retrieval. In the later case, the mobile sink has the flexibility to move over the network and gather the collected data, and the multi-hop sink connectivity is not always required [3, 14]. In both cases, depending on how data is organized in the network, sink connectivity and/or trajectory to the sink have to be defined. For these reasons, data management reveals to be an important design issue in monitoring-based WSNs.

A key challenge in this context is *how to efficiently distribute and store monitored data such that it can be later sent to or retrieved by a sink.* A failure in this process might result in data loss. Much work has been carried out on data dissemination in WSNs. Basically, these works can be categorized as reactive or proactive. In the reactive approaches, sensors have to react to indications of the position of static sinks or of trajectory taken by mobile sinks, so that their monitored data can be sent accordingly to the sinks location [11, 15, 2]. In proactive approaches, the monitored data should be disseminated through the deployment region in advance so that (i) connected paths may be later established between storing nodes and the static sinks or (ii) the mobile sink may later visit the storing nodes [20, 7, 22]. In the latter case, the way the data dissemination is performed will determine if the sink trajectory may be either predetermined with controlled mobility [23, 7] or free by following an uncontrolled mobility pattern [22]. In this paper, *we focus on proactive data dissemination strategy and on how to select well distributed storing nodes in WSN. Sinks may be then either static and located on the border of the network, or mobile with its trajectory being unknown to the sensors.*

Additionally, given the resource-limited characteristics of sensors, any data dissemination mechanism developed for them must be simple and incur low overhead. Assuring *an efficient proactive data dissemination by only using local inferred neighborhood knowledge while respecting resource constraints* is thus an interesting challenge in WSNs and the focus of our research. In fact, the difficulty in selecting well distributed storing nodes in a network strongly depends on the amount of network knowledge available to sensors. If every sensor has a complete knowledge of the network, selecting storing nodes becomes trivial. Otherwise, if sensors are only aware of their neighborhood and have no location

MSWiM'10, October 17–21, 2010, Bodrum, Turkey.

information, ensuring that a set of well selected storing nodes emerges from individual decisions is challenging.

To counter these issues, this paper presents a flexible proactive data dissemination protocol, called *Supple*. Supple empowers sensors with the ability to make storing decisions that rely on neighborhood information only and flexible selection criteria, which can follow any predetermined distribution law. Although a large amount of effort has been invested in designing data dissemination algorithms for WSNs [22, 7, 2], the provision of a lightweight data distribution strategy adaptable to any criterion of storing nodes' selection (e.g., storage capability, energy constraints, network location, equal storing load distribution, etc) has not received similar attention. Hence, the novelty of Supple is *its flexibility in selecting good storing nodes respecting the established selection criterion without having a global network view.* Additionally, in a network with n nodes, Supple guarantees that the contact and data gathering by a sink of only m storing nodes, where $m << n$, will allow it to get a representative amount of data of the whole network.

Supple is based on three phases: neighborhood discovery, weight distribution, and data dissemination. Sensors use a simple tree-based structure, constructed during the neighborhood discovery phase, which allows weight distribution among nodes. Weights are based on predefined criterion of selection as well as distribution law, and are used by sensors at the data dissemination phase. At this phase, sensors then make on the fly forwarding and data storing decisions based on their own weights and the weights of their neighbors. Supple takes advantage of the bias among different sensors' weights for good data dissemination. This behavior can be used for *uniform data distribution*, by assigning equal weights to all sensors as well as for *specific data distribution*, by assigning high weights to specific selected set of sensors. This later distribution can be useful in cases where, to avoid network disconnections, only border nodes are used for storing activities: referred here as *location-based data distribution.*

We provide a detailed formal analysis of Supple and analytically compare it with other data dissemination techniques, such as RaWMS [1] and flooding. Although Supple achieves the same properties as RaWMS for uniform data dissemination, we show it has a much better message complexity (i.e. Supple uses exponentially fewer messages). Moreover, Supple has the advantage of being flexible: allows disseminating data on any subset of the nodes with any distribution. Finally, by simulations, we study the performance of Supple on a large set of topologies and compare it to RaWMS [1]. The simulation results largely confirm our theoretical analysis. They show Supple is practical and effective in distributing data among selected storing nodes that respect the predefined distribution criterion with limited network knowledge. Simulation results also show that Supple outperforms RaWMS in term of robustness to message losses.

In the remainder of this paper: Section 2 presents the system model; Section 3 describes the Supple approach and its formal analysis; Section 4 discusses simulated results and Section 5 the related works; Section 6 concludes the paper.

2. RATIONALE AND SYSTEM MODEL

In Supple, each node determines its own group of storage nodes independently of other nodes, without any implication on the randomness of each group. The storing selection flexibility feature of Supple allows its combination with data gathering strategies based both on static or mobile sinks, with the condition of accordingly setting the predefined selection criterion. For instance, in the case of static sinks located close to the border of the network, a location-based data dissemination may be used, where only border nodes would perform storing activities.

Case of study. We consider an application where a large number n of sensors are randomly scattered on a given geographic area for collecting data or monitoring events. Data is then gathered by a finite set of static or mobile sinks.

Nodes. All sensors are uniquely identified. Sensors are all *equal* in terms of computational, memory, and communication capabilities. No synchronization is required. Following the proactive data dissemination's procedures, each sensor node in the network is provided with a *partial view* regarding some other nodes (including for itself). In this way, each node may act as a storage node for some other nodes in the WSN, but not for all of them. By slight abuse of terminology, we use the term *view* both for the actual information stored at a given node p and for the IDs of the nodes whose information is stored at p. The size of views will be analyzed in the following sections. To counter the limited buffer of sensors, we consider the use of power-aware compression algorithms to deal with the main drawback of partial views of s entries at storage nodes [17]. Generally, if the latency is not an application issue the data collected by sensors can be locally compressed before being disseminated, reducing the network traffic and thus prolonging the network lifetime. For instance, algorithms like the one presented in [17] reaches compression rations up to 70% on environmental datasets.

Communication. Each node i is able to wirelessly communicate with a set of *neighbors* that are within i's transmission range: a transmission disk centered on i with radius t. We assume *bidirectional* communication and that the average density of nodes $d_{avg} = \frac{\pi t^2 n}{a^2}$ is such that the resulting communication network is connected [6]. The network is thus modeled as a 2-dimensional *Unit Disk* graph $G^2 = (n, t)$, being $G = (V, E)$ where V is the set of network nodes and E models the one-to-one neighboring links.

Limited initial knowledge. Initially a node $i \in V$ only knows its identity, the fact that no two nodes have the same identity, and a parameter $W(i)$ that defines its weights in the network ($W : S \to \mathbb{N}$ for $S \subseteq V$ is called the weighting function). Weights are initially assigned to nodes based on an external criterion of storing nodes' selection. For uniform selection, all sensors will have the same weight and then, the same chances to be selected as storing node by another node. On the other hand, if the criterion is a location-based selection only nodes at the specified location will be used as storing nodes. For instance, if the criterion is a border selection strategy, each sensor i located on the border of the network will have $W(i) = 1$ and $W(i) = 0$ if it is an insider node (cf. [19]). Supple may also rely on the use of dynamic weights among selected storing nodes: e.g. to give to nodes located on the network border and having higher storing or energy capabilities, a higher probability of storing. In this case, an external mechanism should provide this information and accordingly assign the parameter $W(i) > 0$ to each node i. This however, is not the focus of this paper. Hereafter, we use the term *target set* to refer to the subset of nodes with weight greater than 0, i.e. the storing nodes. Finally, nodes do not know their position and we do not use

Supple Input: Graph $G = (V, E)$
Input: Target set $S \subseteq V$
Input: Weighting function $W : S \to \mathbb{N}$
Input: View size $s \in [1, \ldots, |V|]$

1. Construction of the tree $T(G)$ from G
2. Propagation of the weights of the target set nodes
3. For each node $i \in V$
   send $data(i)$ to the root of $T(G)$
4. The root propagates each $data(i)$, $r(s)$ times according to the probabilities induced by the weights over the target set

Figure 1: Principle of Supple.

Weight distribution
Input: Tree $T(G)$
Input: Target set $S \subseteq V$
Input: Weighting function $W : S \to \mathbb{N}$

For each node $i \in T(G)$ create a triple $(l_i, W(i), r_i)$
For each node $i \in T(G)$ in a breadth-first search starting from the leaves
   do let $j :=$ left child of i in $l_i = l_j + W(j) + r_j$
   do let $p :=$ right child of i in $r_i = l_p + W(p) + r_p$

Figure 2: Weight distribution in $T(G)$.

any geographic knowledge in our algorithm. The presented hereafter approach relies solely on node connectivity.

3. SUPPLE: FORMAL PRESENTATION

In this section, we formally present the Supple algorithm. The Supple's goal is to allow each node sending its collected data to a *target set*, to be latter gathered by the sink. Using Supple, we can ensure that each storing node of the target set has a view of controlled size s containing data collected by any s nodes in the network, which are chosen according to the Supple algorithm. The general principle of Supple is described in Fig. 1. Further details are provided in the following sections.

3.1 Tree construction

Let $G = (V, E)$ be the graph that represents the network. The first step of Supple relies on a tree construction: a tree-based routing structure $T(G)$ initiated by a central-localized node in the network and that is at least binary. The constructed tree $T(G)$ embeds the connectivity of the network and ensures that sampling a node according to a given distribution can be done with a logarithmic number of hops. In particular, Supple requires a bootstrap phase where $T(G)$ is constructed using a cost metric propagated in 1-hop `Hello` messages. The constructed $T(G)$ structure is thus, an aggregation of the shortest paths from each sensor to the central-localized node based on a cost metric, which can represent any application requirement: hop count, loss, delay among others. An important set of routing protocols in WSNs are based on the construction of a tree-based routing topology rooted at the sink [11, 16]. Other tree-based structures that can be used in conjunction with Supple are PeerNet [4] and Tribe [21]. PeerNet in particular, constructs a binary tree. Finally, *Supple can be adaptable to any kind of structure, the only requirement being the routing capability.*

In the rest of the paper, we consider that the $T(G)$ construction is performed with a hop count metric and that the tree is binary. The complexity of the tree construction is then $\mathcal{O}(n)$. Note that this is done for the sake of clarity and it is not a limitation of our method.

3.2 Weight distribution

The flexibility of Supple is given by the fact that the data dissemination can be adapted to any target set (denoted by S). For this, all nodes of the target set have a weight assigned through a function $W : S \to \mathbb{N}$. The probability of sending data to a particular node i is given by the weight assigned to node i with respect to the sum of all weights of storing nodes in the target set. Although allowing the use of any selection criterion of storing nodes (e.g. uniform, location-based, energy-based, etc), *we consider in this paper, the uniform selection criterion in order to allow the comparison with the related approach RaWMS [1].* For this reason, $|S| = V$ and for all nodes $i \in S$, $W(i) = 1$. This will give to all nodes the same chances to be selected as storing nodes. Once equal weights are assigned to nodes, nodes perform the weight distribution over the tree, as depicted in Fig. 2. The idea behind this algorithm is to initialize each node $i \in S$ with a triple $(l_i, W(i), r_i)$, where l_i (resp. third component r_i) is the weight of the left (resp. right) subtree of i and $W(i)$ is the weight of the node i in the target set.

It is clear that the complexity of the whole weight distribution process is $\Theta(n)$. Additionally, the weight distribution only requires a field of at most $\log n + \log |W|$ bits in the usual `Hello` packet.

3.3 Data dissemination

The data dissemination is the most important phase of Supple. This phase ensures the properly data propagation at storing nodes. Since we consider here that nodes have the same weight, this phase has to ensure a uniform distribution of nodes' data among the target set.

The idea is the following. Firstly, all nodes must send their data to the root of the tree (i.e. the node that started the tree construction), as detailed in Fig. 3. When the root receives new data from one of its children, it means that a node is propagating its information for dissemination into the target set. The root propagates then, $r(s)$ times the data to its children. This will ensure the views are of size s (cf. Proposition 1) . The propagation by the root is done according to the weights of its left and right subtree and also to its own weight (in the case the root is also in the target set). The *Forward_data* algorithm depicted in Fig. 4 formally presents this local propagation. Moreover, it must be noted that messages are forwarded asynchronously, i.e., there is no reason for the root to finish the $r(s)$ sequential data sending of a node to start sending data of another node. Thus, each node forwards messages coming from its parent according to the *Forward_data* algorithm (cf. Fig. 4). It is worth noting that the algorithm naturally stops when the message is received by a node whose left and right component of the triple equals to 0 (i.e., at the leaf level).

At the end of the data dissemination, all nodes of the target set will have, with high probability, a view of size s. This view is randomly composed by nodes' data distributed according to the weights given on the target set. In the case weights are equal for all nodes, we naturally achieve view of size s with uniformly disseminated data.

The complexity of the data dissemination is the keypoint

Table 1: Comparison between Supple, RaWMS, and flooding.

	# rounds	msgs per round	total msgs	msg size	mem. overhead	additional overhead
Supple	$r(s)$	$n \cdot \log n$	$n \cdot r(s) \cdot \log n$	1	view size s	3 integers per node
RaWMS	$r(s)$	n^2	$n^2 \cdot r(s)$	1	view size s	
flooding	1	n^2 broadcasts	n^2 broadcasts	1	linear	mem. for flooding

Data dissemination
Input: Tree $T(G)$ with a triple $(l_i, W(i), r_i)$ for each $i \in T(G)$

For each node $i \in V$ do
    Send data to its parent
On reception from a child by node i do
    if $i \neq$root then forward to parent
    if $i =$root then do $r(s)$ times
        Forward_data
On reception from the parent by node i do
    Forward_data

Figure 3: Algorithm for the data dissemination.

Forward_data
Input: Tree $T(G)$ with a triple $(l_i, W(i), r_i)$ for each $i \in T(G)$
Input: viewsize s
Input: data d
(code for node i)
Pick at random uniformly $x \in [0, l_i + W(i) + r_i]$
If $x < l_i$ then send d to left child
If $l_i \leq x \leq l_i + W(i)$ then store d in own view
If $W(i) + l_i < x$ then send d to right child

Figure 4: Algorithm for forwarding data down into the tree.

of Supple. Each node sends its data to the root, which implies $\mathcal{O}(n \log n)$ messages. Then each data is propagated $r(s)$ times through the tree (from the root to the leaves), resulting in $\mathcal{O}(n \cdot r(s) \cdot \log n)$ messages. Finally, the complexity in term of messages of the whole process is $\mathcal{O}(n \cdot r(s) \cdot \log n)$. In this way, Supple outperforms the message complexity of the related work closest to Supple, the RaWMS[1], which achieves a complexity in term of messages of $\Theta(n^2 \cdot r(s))$.

3.3.1 Formal analysis

Here, we address the problem of computing the number of times a message must be sent through the tree in order to ensure that the size of the views will be, with high probability, s. Intuitively, if each disseminated node data would have reached a different storing node, then in order to obtain a view of size s, it would have been enough to start s data sending at each node, during the data dissemination phase. Nevertheless, two data sending started at the same node i have a non-negligible probability of reaching the same storing node j. Thus, in order to obtain the target view size s, each node should start a larger number $r(s)$ of sending, where $r(s) > s$. The following proposition gives an explicit lower bound for the value of $r(s)$, depending on s.

PROPOSITION 1 (COMPUTATION OF $r(s)$). *Let n be the number of nodes in the tree $T(G)$. To obtain, with high probability, a view of size s, $r(s)$ messages must be sent, where:*

$$r(s) = \begin{cases} n \ln(\frac{n}{n-s}) & if\ s \neq n, \\ n \ln n & if\ s = n \end{cases}$$

PROOF. Let first consider the case where $s \neq n$. We want to compute the number of messages that must be sent in order to obtain with high probability, s different nodes, performing the data dissemination according to uniformly distributed weights. The number of unreachable nodes p after r(s) messages is: $\mathbb{E}(p) = n - s$, which can be rewritten as

$$n(1 - \frac{1}{n})^{r(s)} = n - s$$

Using the classic inequality $(1-x)^y \leq e^{-y.x}$ this means:

$$e^{(-r(s)/n)} \geq \frac{n-s}{n}$$

By choosing $r(s) = n \ln(\frac{n}{n-s})$, we obtain the aimed expectation, i.e. balanced views of size s.

We now prove the case where $s = n$. In this case we have $n \cdot e^{(-r(s)/n)} = 0$. Using $r(s) = (k+1) \cdot n \ln n$ (with $k \in \mathbb{N}$), we obtain:

$$n \cdot e^{(-r(s)/n)} = n^{-k}.$$

This means that the probability of a collision can be as small as we want. □

Another point of interest is the relationship between the size s of the view and the size $|S|$ of the target set. Indeed, if the target set size $|S|$ is too small with respect to the view size s, not enough space will be available to store data of nodes of the whole network. Hence, even if the data stored by all nodes in the target set is gathered, it will not provide a representative amount of network data. The following proposition addresses the relationship between s and $|S|$.

PROPOSITION 2 (RELATIONSHIP BETWEEN s AND $|S|$). *Let $|V| = n$. If the view size of each node is limited to s, then the target set S must contain at least $\Theta(\frac{n}{s} \ln n)$ nodes in order to guarantee with high probability a good data storing and a satisfying data gathering by a sink.*

PROOF. We use the solution of the coupon collector's problem that can be found in [18] (pages 57–63). In our case the number of trials x is $s \times |S|$, in [18] it is proven that $x \geq n \ln n$, thus we have $|S| \geq \frac{n}{s} \ln n$. □

The proposition 2 only gives hints to Supple users in order to make sure that the size of the target set is large enough to store the nodes' data of the whole network. In the case where $|S| = n$, the Proposition 2 also gives the *minimum number of storing nodes m in the target set to be contacted by a sink in order to gather a representative amount of data of the whole network*. Note that only the quantity of $(\frac{n}{s} \ln n)$ target nodes has to be respected and the sink is free to contact *any* node in the target set.

3.4 Summary

Here, we discuss the pros and cons of Supple and compare its complexity to the RaWMS strategy. The data dissemination of Supple has the flexibility and the self-organizing feature of being adaptable to any kind of data distribution, which is dictated by the way weights are distributed among nodes. In the particular case of uniform distribution, this data dissemination is done more efficiently than in the related approaches, thanks to the tree structure. Indeed, this implies an exponential improvement of the number of messages used to obtain the uniform distribution.

Additionally, Supple requires a small additional overhead in term of memory: only a triple of integers. Finally, Supple is robust to messages losses and failures of storing nodes (as stated in section 4), since the data of each node is replicated in $r(s)$ storing nodes. If the sink is mobile, no path construction among storing nodes and the sink is necessary, since the sink will directly contact the storing nodes. For the special case where $s = \sqrt{n}$ (i.e. $s \neq n$), we get $r(s) \approx \frac{n}{n-s} \approx \sqrt{n}$. This means that for relatively small view sizes, there is a very little chance of getting collisions and that by only contacting $m = \sqrt{n}$ storing nodes a sink can get a representative view of the whole data in the network (i.e., $s * m = \sqrt{n} * \sqrt{n} = n$).

Regarding the assignment of dynamic weights to nodes, such as energy-based weights, it is enough to associate energy-level thresholds with weights and to redo the weights distribution phase each time the node energy falls below the threshold. In this way, the probability of a node to be selected as a storing node would be given by its weight and consequently, by its remaining energy. Note that here, we only discuss how the flexible data dissemination Supple can be performed according to the location and weights of nodes. So that, Supple can be adapted to any mechanism of weight assignment. The specification of these mechanisms, however, is not our focus here.

Otherwise, with the construction of the tree, nodes close to the root become hot spots, what can cause battery depletion of nodes and consequently, tree disconnections. This can be avoided or minimized by the use of specific policies for dynamically modifying the root of the tree, or by the generation and maintenance of multiple and different trees. In this paper, however, our main goal was to evaluate the main features of the Supple approach. Those kinds of improvements are left for future work.

Concerning the complexity, the most important results are the following. The propagation of the weights is done with $\mathcal{O}(n)$ messages. The data dissemination for each node that sends its data, is done with $\mathcal{O}(r(s) \cdot \log n)$ messages. Thus, the total complexity in term of messages for the data dissemination for all nodes is $\mathcal{O}(n \cdot r(s) \cdot \log n)$, which is then the total number of messages used by Supple. Table 1 summarizes the differences with RaWMS[1] and flooding.

4. PERFORMANCE ANALYSIS

4.1 Evaluation methodology

Supple is evaluated through simulation using a home-made simulator. Each simulation comprises a dissemination and a gathering phase. In the first phase, each node performs Supple or RaWMS for data dissemination. In the gathering phase, a mobile sink performs as many visits as necessary to get a representative amount of data of the network, meaning getting n different entries of storing nodes' views.

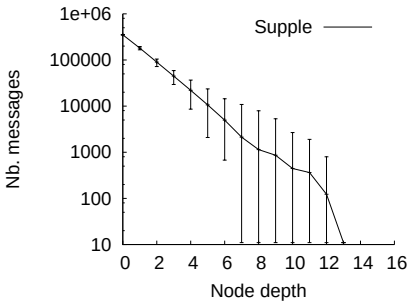

Figure 5: Average number of messages sent by a node in Supple as a function of its depth in the tree.

4.1.1 Experimental setup

Four scenarios have been considered, each experiment has been repeated 25 times and the results represent the mean value of these experiments. In the first scenario, the same topology and tree are used in all experiments in order to evaluate the effects of the random choices performed at the *Forward_data* (cf. Fig. 4). In the second scenario, again the same topology is used but different trees are generated for each experiment. By comparing these results to the previous one, we show how the tree construction impacts the performance of the protocol. In the third scenario, we compare the Supple protocol to the RaWMS protocol on 25 different topologies. Finally, the fourth scenario allows the evaluation of the performance of both protocols in the presence of message losses. Simulations stop when no more message is circulating in the network.

It is worth mentioning that, in order to compare the dissemination capabilities of Supple and RaWMS, no salvation mechanism was added to deal with message losses (e.g., in RaWMS, this mechanism establishes that if a low level acknowledgment is not received for the just sent message, then another random neighbor is chosen by the RW process). Its implementation could improve the performance of both protocols under message losses.

4.1.2 Simulation parameters

Our simulations involves scenarios with $n = 1,000$ nodes placed at uniformly random locations in a square area. The average number of nodes in the communication range of any node was set to a target average density $d_{avg} = 24$. Only connected topologies were considered, where a binary tree was constructed from the root (cf. section 3.1).

For comparison reasons with RaWMS, we set: (1) the size of the target set to $|S| = V = n$ and assign $W(i) = 1$ for all nodes $i \in S$; and (2) nodes view size to $s = \lceil \sqrt{1,000} \rceil = 32$ entries and consequently, $r(s) = \lceil \sqrt{1,000} \rceil = 32$ (cf. Section 3.3.1). As a buffer management policy, we consider a *size-based policy*, which removes the oldest entry to make room for the new information in the view. Finally, both protocols, Supple and RaWMS, handle multiple-entry collisions: when adding a new entry to the view, they first check if it is already known, thus no entry is added twice.

4.1.3 Evaluation metrics

To evaluate the cost and efficiency of Supple, we evaluate (1) the messages overhead, which counts the amount of

transmitted messages by each node (i.e., we want to ensure that the impact of both the tree and the random choices on the communication are negligible) and (2) the efficiency in data gathering, which is the accumulated amount of collected information after a node is visited by the sink. We know that, at the worst case, all the data can be gathered by visiting all the nodes in the target set. Nevertheless, we want to know how fast the data can be collected. In fact, by well distributing data among storing nodes (and in the case $|S| = n$), Supple allows a mobile sink to perform free trajectories and get a representative amount of information of the whole network by visiting a small number m, where $m << n$, of storing nodes of the target set. In order to evaluate this property, we consider a mobile sink will cross the network and randomly visit nodes, gathering the data in their views.

4.2 Simulated results

4.2.1 Communication overhead

Fig. 5 represents the average number of messages sent by a node as a function of its depth in the tree. The number is the mean value obtained over 25 experiments, where the tree is the same and only the random choices made in the algorithm are different (cf. Fig. 4). We sum all the messages sent by nodes at a certain depth and then divide this number by the number of nodes which are at this particular depth. As expected, the closer the node is to the root in the tree, the more messages it has to send. Thus, as discussed in Section 3.4, the use of multiples trees and consequently multiples and well distributed roots, could help on the distribution of message overhead among nodes in the network.

Additionally, we have measured the total number of messages seen at the end of each experiment. It equals to an average of $3.1967E+06$ with a very small variance for the 25 experiments, which demonstrates the limited influence of the random choices of the protocol. Instead, RaWMS [1] presents much higher overhead results for a smaller network: up to $5E+03$ messages for a network size of 800 nodes. When different trees are built on each experiment, the number of messages equals to $3.2E+06$ in average and the variance is still small. This shows that the underlying network, the constructed tree, and the random choices performed during Supple deployment do not really impact the behavior of the protocol as long as it is "well balanced" (i.e. each non-leaf node has at least two children). In addition, both results confirm the message overhead analysis discussed in Section 3.4.

4.2.2 Efficiency in data gathering

Here, a mobile sink is firstly placed in a random position in the network, it visits the nodes in this position, and then chooses the next node to visit, trying to avoid revisiting an already visited node (as introduced in [5]). When the sink visits a node it gathers this node's data and all the information in its view. This procedure is repeated until the sink has collected all the network information. Fig. 6(a) and 6(b) show the amount of accumulated collected entries (represented in the graph by the number of gotten IDs of the stored data) the mobile sink gathers per visited node. The results indicate that Supple gives similar results to the ones given by RaWMS. In particular, after visiting any $2.3\sqrt{n} \approx 73$ nodes, the sink is able to collect information from about 90% of the nodes, as implied by the analysis in Section 3 and [5]. Thus, these results confirm that Supple with an exponential improvement of the number of messages (cf. Table 1 in Section 3) compared to RaWMS, allows the mobile sink to achieve a high representative view (i.e. 90%) of the whole network data, by only visiting a relative small number of nodes network, i.e. 73 nodes over $1,000$ (7.3% of nodes). Additionally, Fig. 6(c) and 6(d) show that random choices as well as the use of different trees at the tree construction phase do not affect the good performance given by Supple in the amount of collected information (i.e. 90% of total data when only 7.3% of network nodes are visited).

4.2.3 Loss resilience

Fig. 6(e) shows the performance in data gathering of both protocols, when the probability of loosing a message is 5%. The performance of Supple is not affected by the loss rate, whereas RaWMS is unable to tolerate this low level of failures. This is clearly understandable since the RaWMS protocol exchanges exponentially many more messages than Supple. The probability for a message to be delivered in the RaWMS protocol with 5% of messages loss is $0.95^{1998} \simeq 3.10286848 \times 10^{-45}$, which is close enough to zero to result in no message reception. In particular, the effect of message losses in the RaWMS efficiency is only reduced when unicast packet retransmissions and the salvation mechanism is implemented, as shown in results in [1].

Finally, Fig. 6(f) shows the averaged number of data the sink has gathered information after having visited 200 nodes, under different loss rates. We can see that when the message loss percentage is higher than 1‰, the performance of the RaWMS protocol decreases rapidly, while the Supple protocol still keeps good performances. This is again explained by the fact that RaWMS exchanges exponentially many more messages than Supple.

4.3 Discussion

Supple provides *an exponential improvement of the number of messages used*, when compared to RaWMS. The choice to use a tree, however, introduces a high overhead of messages transmission to the root and its vincinity. A solution to this consists in creating multiple trees with different roots and load balancing the data dissemination on the different trees, alleviating the communications requirements imposed to a unique root. Such solution can be easily implemented by randomly selecting uniformly distributed nodes in the network for initiating the tree construction.

Additionally, the simulations demonstrate that Supple behaves as predicted by the theoretical analysis: a very high proportion of n data can be distributed in a way that, it is still possible to gather most of the network data by only visiting a small portion $m = 2.3\sqrt{n}$ of the target set, where $m << n$. Regarding efficiency in data gathering, Supple provides *the same quality of data dissemination as the RaWMS protocol, but with more flexibility,* since the storing nodes may be selected following any criterion of distribution. The simulations also illustrated that, due to its relatively small number of message transmissions when compared to RaWMS, *Supple can tolerate a much higher failures rate without requesting additional link reliability or extra salvation mechanisms.*

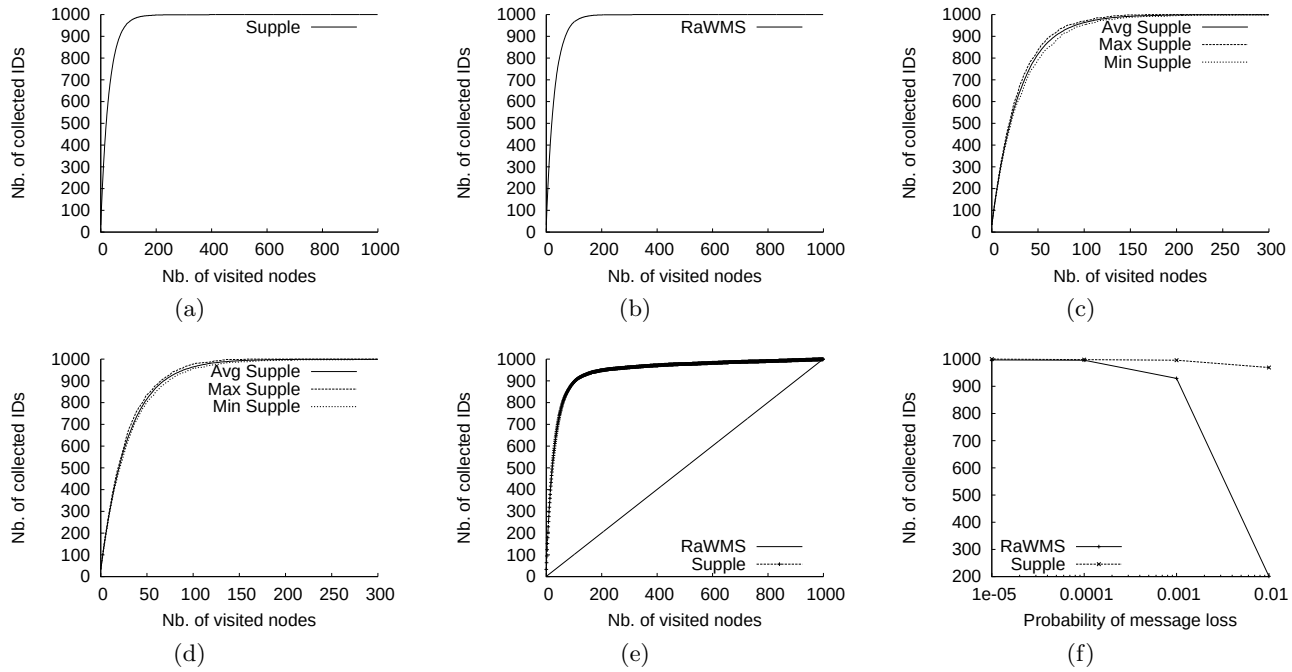

Figure 6: Amount of collected information per visited node in $1,000$-node network. Given by (a) Supple strategy, (b) RaWMS strategy, (c) Supple strategy with fixed tree and (d) Supple strategy with a different tree per experiment. (e) Under 5% of loss rate. (f) Amount of collected information after visiting 200 nodes and under different loss rates.

5. DATA DISSEMINATION IN WSNS

In the literature, much work has been carried out on data dissemination in WSNs. The way the data dissemination is performed depends, however, on how the sink gathers the monitored data made available by the sensors in the network. In a general point of view, data aggregation allows a structural organization of the network topology. This organization can be: *reactive* or *proactive*. By surveying the literature, it appears that early research on reactive approaches in wide-area static sensor networks can be traced back to Directed Diffusion [11] and SPIN [9]. [25, 10] are other examples of works dealing with static sink and reactive dissemination strategies. [25] establishes a data collection tree by query propagation from sinks. [10] is based on a isobar mapping, which allows to build a topographic map of a space populated by sensors and data aggregation with nodes similarity depending on collected data. In this last years, [13] has proposed an approach that combines the push and pull queries strategies, known as an hybrid approach. Among the proactive approaches, Ratnasamy et al [20] proposed the use of a Distributed-Hash-Table structure on top of the geographic routing protocol (GPSR) to support data-centric storage. In [8], a clustering-based protocol is proposed to transmit data to the base station. On the other hand, the presence of sinks that can move and directly collect data from sensor nodes in a monitored area, avoids the necessity for sensor-to-sink path maintenance in the network. [15, 2] are examples of reactive dissemination approaches where the mobile sink follows a controlled, and thus predictable, trajectory. In this case, the sink must visit some predefined nodes to retrieve a representative view of the monitored data. On the other hand, reactive dissemination approaches where the mobile sink follows a free, i.e. uncontrolled, trajectory, requires the sensors to track the sink mobility in order to adapt or influence the data dissemination [24, 12]. In summary, these reactive proposals must dedicate a significant amount of resources to track the sink and to forward on-the-fly the data to be collected towards the mobile sink.

In proactive data dissemination strategies with predictable sink mobility, data is sent by sensors to a well selected subset of nodes, typically forming a virtual structure, to be later retrieved by the mobile sink [23, 7]. On the other hand, if the mobile sink performs an uncontrolled mobility, no structure can be defined. In this case, the dissemination should be performed in a way that allows the sink following a free trajectory to retrieve a representative view of the monitored area by visiting a relatively small number of any nodes in the network [22].

Our protocol Supple deals with proactive data dissemination in WSNs and can be adaptable to static or mobile sinks' visits. Supple allows data dissemination to a subset of nodes in the network by following any previously defined selection criterion. One example of selection criterion concerns a subset of border nodes, which can be an interesting storing option if sinks are located close to the network border or follow a controlled trajectory defined by the border nodes location. Otherwise, by uniformly distributing data over the target set $|S| = n$, Supple can be also adapted to the case where the mobile sink performs uncontrolled mobility. Nevertheless, contrarily to the approaches presented in [22, 1], Supple allows an efficient data dissemination with a lower communication overhead.

6. CONCLUSION AND FUTURE WORK

In this paper, we presented Supple, a proactive data dissemination protocol. Supple is an efficient approach to distribute and store monitored data such that it can be later sent to or retrieved by a sink. We presented a formal analysis of Supple and performed simulations to study its performance compared to RaWMS. The simulated results confirmed the formal analysis. Supple limits the load in each storage node to s and enables each node to determine its own group of storage nodes independently of other nodes without any implication on the randomness of each group. Hence, in the case where mobile sinks are used, the visit of $m << n$ nodes can guarantee the gathering of a representative data view of the whole network. For the special case where the view size of nodes is limited to $s = \theta(\sqrt{n})$ and $r(s) \approx \frac{nk}{n-s} \approx \sqrt{n}$, the communication complexity equals to $\theta(n \cdot \sqrt{n} \cdot \log n)$ and $m = \sqrt{n}$. In this way, Supple allows an efficient data dissemination with an exponential improvement of the number of messages compared to RaWMS. The flexibility of Supple is based on the criterion of nodes selection. This one can be used in future work to select nodes based on their resources and to dynamically and accordingly reconstruct the tree.

7. REFERENCES

[1] Z. Bar-Yossef, R. Friedman, and G. Kliot. RaWMS - Random Walk based lightweight Membership Service for wireless ad hoc networks. *ACM Transaction on Computer Systems*, 26(2):1–66, June 2008.

[2] S. Basagni, A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. M. Wang. Controlled sink mobility for prolonging wireless sensor networks lifetime. *Wireless Networks Journal (WINET)*, 14(6):831–858, Dec. 2008.

[3] I. Chatzigiannakis, A. Kinalis, and S. Nikoletseas. Efficient data propagation strategies in wireless sensor networks using a single mobile sink. *Computer Communications*, 31(5):896–914, 2008.

[4] J. Eriksson, M. Faloutsos, and S. Krishnamurthy. Scalable ad hoc routing: The case for dynamic addressing. In *Proc. of IEEE Infocom*, Mar. 2004.

[5] R. Friedman, G. Kliot, and C. Avin. Probabilistic quorum systems in wireless ad hoc networks. In *Proc. of the IEEE DSN*, June 2008.

[6] P. Gupta and P. Kumar. Critical power for asymptotic connectivity in wireless networks. In *Proc. of Stochastic Analysis, control, optimization and applications*, pages 547–566, 1998.

[7] E. B. Hamida and G. Chelius. A line-based data dissemination protocol for wireless sensor networks with mobile sink. In *Proc. of IEEE ICC*, May 2008.

[8] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocols for wireless microsensor networks. In *Proc. of Hawaaian International Conference on Systems Science*, Jan. 2000.

[9] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proc. of ACM Mobicom*, pages 174–185, Aug. 1999.

[10] J. M. Hellerstein, W. Hong, S. Madden, and K. Stanek. Beyond average: Toward sophisticated sensing with queries. In *Proc. of Snd International Workshop , ISPN*, LNCS. Springer, 2003.

[11] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proc. of ACM Mobicom*, pages 56–67, Aug. 2000.

[12] A. Kinalis and S. Nikoletseas. Adaptive redundancy for data propagation exploiting dynamic sensory mobility. In *Proc. of 11th ACM MSWiM*, pages 149–156, Oct. 2008.

[13] X. Liu, Q. Huang, and Y. Zhang. Balancing push and pull for efficient information discovery in large-scale sensor networks. *IEEE Transaction on Mobile Computing*, 6(3):241–251, 2007.

[14] J. Luo and J. P. Hubaux. Joint mobility and routing for lifetime elongation in wireless sensor networks. In *Proc. of the IEEE INFOCOM 2005*, Miami, FL, USA, Mar. 2005.

[15] J. Luo, J. Panchard, M. Piorkowski, M. Grossglauser, and J. P. Hubaux. Mobiroute: Routing towards a mobile sink for improving lifetime in sensor networks. In *Proc. of IEEE/ACM DCOSS06*, pages 480–497, San Francisco, CA, USA, June 2006.

[16] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks. *Proc. of ACM SIGOPS Operating Systems Review*, 36, 2002.

[17] F. Marcelloni and M. Vecchio. A simple algorithm for data compression in wireless sensor networks. *Communications Letters, IEEE*, 12(6):411–413, June 2008.

[18] R. Motwani and P. Raghavan. Randomized algorithms. *ACM Computing Survey*, 28(1):57–63, 1996.

[19] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proc. of ACM Mobicom*, Apr. 2003.

[20] S. Ratnasamy, D. Estrin, R. Govindan, B. Karp, and S. Shenker. Data-centric storage in sensornets. In *Proc. of ACM Sicomm*, Feb. 2002.

[21] A. C. Viana, M. D. Amorim, S. Fdida, and J. F. Rezende. Indirect routing using distributed location information. In *Proc. of IEEE PERCOM*, Mar. 2003.

[22] A. C. Viana, A. Ziviani, and R. Friedman. Decoupling data dissemination from mobile sink's trajectory in wireless sensor networks. *IEEE Communications Letters*, Mar. 2009.

[23] Z. Vincze, D. Vass, R. Vida, A. Vidacs, and A. Telcs. Adaptive sink mobility in event-driven multi-hop wireless sensor networks. In *Proc. of 1st International Conference on Integrated Internet Ad Hoc and Sensor Networks*, pages 30–31, May 2006.

[24] H. Yang, F. Ye, and B. Sikdar. SIMPLE: Using swarm intelligence methodology to design data acquisition protocol in sensor networks with mobile sinks. In *Proc. of the IEEE INFOCOM*, Apr. 2006.

[25] Y. J. Zhao, R. Govindan, and D. Estrin. Residual energy scans for monitoring wireless sensor networks. Technical report, UC Los Angeles, 2002.