Abstract

Monitoring spatio-temporal continuous fields using wireless sensor networks has emerged as a novel and efficient solution. The development of energy efficient query dissemination and data collection algorithms for environments where only a small subset of nodes has relevant readings is a challenging problem if no information about the location of these nodes is available. Monitoring these data requires not only an initial discovery stage but also a continuous search for new relevant data due to field variations in time. One solution to this problem is to let nodes cooperate and decide jointly which data are relevant and their locations. Trails to relevant data can be distributively generated as insect colonies do using pheromone-based indirect communication. In this work, we propose PhINP, a probabilistic Pheromone-based In-Network Processing mechanism to monitor information using WSNs. Our proposal takes advantage of both a pheromone-based iterative strategy to direct queries towards nodes with relevant information and query- and response-based in-network filtering strategies to reduce the quantity of nodes selected to answer the query. Additionally, nodes use reinforcement learning to improve the routing performance of queries. We demonstrate by extensive simulations that using PhINP mechanism the query routing cost can be reduced by approximately 60% over flooding, applying the same in-network filtering strategy, with an
error below 1%.

**Keywords:** Bio-inspired Networking, Computational Intelligence, In-Network Processing and Filtering, Monitoring System, Routing Algorithms and Protocols, Swarm Intelligence, Wireless Sensor Networks.

1. Introduction

A wireless sensor network (WSN) consists of a set of small size, low cost, and low energy devices deployed over an area in order to monitor physical phenomena (such as temperature, pollution, noise level, etc). The information collected by sensor nodes can be either proactively or reactively relayed from them to a sink node using multi-hop communication. However, sensor nodes have constrained energy, sensing, processing and communication capabilities mainly due, amount others reasons, to the fact that they are battery-powered devices [1]. As data communication (both data transmission and reception) is the main source of energy consumption in WSNs, an efficient resource management is a key factor in its design, which is addressed in [2]. This challenges us to develop efficient routing algorithms in order to maximize the network lifetime. A complete survey on routing techniques in WSNs is treated in [3].

In this work, we address the problem of monitoring continuous fields using WSNs where a subset of nodes has relevant readings but no information about the node's location is available a priori. We are interested in problems such as what is the top (i.e., maximum) reading?, or more generally what are the 10% top readings?, and then monitor these points over time. The simplest solution to these problems is to let nodes send their readings to the sink node which processes data in a centralized way (warehousing approach). A more efficient approach, mainly in large sensor networks, is to process inside the network all readings and sent to the sink node only those which are considered relevant. This last approach is known as *in-network processing* (INP). Among INP techniques, *filtering* is the one that has less computational and memory requirements. Other techniques like data aggregation and data fusion typically demand more hardware resources. Filtering can be implemented either at *data-level*, by deleting repeated data messages due to the high spatial correlation, which is dependent of the resolution of the sensor readings, or at *node-level*, by selecting only a subset of sensor nodes to send their readings to the sink. In this paper, we focus on the second case. Generally, node selection can be implemented by either *fixed* or *adaptive filtering* using static or dynamic thresholds, respectively. Static thresholds are used for searching readings within a specific range of values $X$ [4] [5], while dynamic thresholds, for readings without a predefined range or when there is not information of the data range. In this last case, the threshold level is updated in the filtering process. Examples of threshold-based filtering are summarized in Table 1.

Fixed filtering is based on static thresholds so it can be implemented locally by each node despite the actual readings available at other nodes. On the contrary, adaptive filtering depends on these readings to update the thresholds [6]. Filtering can be implemented straightforward by simply disseminating queries all over the network (i.e., flooding). Even if
the cost of disseminating a query is high in terms of message exchange, the cost of collecting data is much lower as only a few nodes do response. This approach guarantees that all relevant nodes are selected for reporting their readings. When monitoring fields, the same query needs to be processed periodically. To avoid flooding the network at each query, one solution is to use proactive schemes based on disseminating a query only once but letting nodes to report their readings if data have changed. However, this scheme is not suitable for large WSNs monitoring dynamic continuous fields, where much data changes are expected.

Table 1. Information obtained using threshold-based filtering techniques.

<table>
<thead>
<tr>
<th>Static threshold (fixed filtering)</th>
<th>Dynamic threshold (adaptive filtering)</th>
</tr>
</thead>
<tbody>
<tr>
<td>max (value &gt; X), min (value &lt; X), range (X₁ &lt; value &lt; X₂), iso-contour (value = X), count (number of nodes with value &gt; X), multi-dimensional / complex (value₁ &gt; X₁ &amp; value₂ &lt; X₂)</td>
<td>max, min, top-k, top-percentual, snapshot, maximal range, multi-dimensional / complex, skyline</td>
</tr>
</tbody>
</table>

In this work, we consider a reactive scheme where only the first query is flooded to all nodes but the following ones are directed, based on an iterative process, towards areas where relevant nodes are located. In this way, we aim at disseminating the same query periodically but at a minimum cost. Besides reducing the cost of data gathering through filtering, it is worth noting that query dissemination may require much message exchange. This work investigates how to decrease the cost of query dissemination by integrating computational intelligence (CI) techniques on nodes that can help to direct queries and avoid flooding the network. CI is a set of nature-inspired computational methodologies to address complex problems of the real world in which traditional methodologies are ineffective or infeasible. A detailed survey of CI paradigms and their potential applications in WSNs is given in [7], in which two bio-inspired paradigms for energy-aware routing are highlighted: swarm intelligence (SI) (such as ant colony optimization, pheromone-based strategies, etc.) and reinforcement learning (RL). Both paradigms have low computational and memory requirements, high flexibility, and optimality. A detailed survey and future directions of SI-based routing protocols is treated in [8].

In this context, we propose PhINP, a pheromone-based in-network processing mechanism using broadcast communication model, which combines advantages of in-network filtering, pheromone-based and reinforcement learning algorithms. The proposal is able to implement both fixed and adaptive in-network filtering by means of a directed query dissemination strategy. For simplicity, we focus the discussion on finding maximum values, which is a particular case of adaptive filtering. Robustness and adaptability of the routing mechanism are emphasized due to the variability of conditions in WSNs (i.e. failure of nodes, introduction of new nodes, packet loss, etc). Additionally, a motivation for this work was to develop cooperative schemes to decentralize the control of the sink node. A sink node that requires information in a moment can become sensor node in a later time (role changing). This scheme enables to extend the network lifetime. Nodes in the neighborhood of the sink have high energy requirements due to they cooperate forwarding many responses.

The rest of the paper is organized as follows. Section 2 discusses both related work on
pheromone-based routing protocols and algorithms and the main differences with our proposal. Section 3 introduces the PhINP mechanism and the network model, while query dissemination, data gathering, and pheromone-level update phases are described in Subsections 3.1, 3.2, and 3.3 respectively. Section 4 analyzes results obtained by simulation and discusses main results. Finally, Section 5 concludes the work.

2. Related Work

Bio-inspired networking and communication protocols and algorithms have emerged as an efficient and robust solution. They are based on biological systems, as result of millions of years of evolution, with characteristics such as adaptability to environmental conditions, resiliency to failures and damages, collaborative operation and self-organization. By looking the nature as a source of inspiration, we can deal with large scale networks in which the absence of centralized control and unattended resolution of potential failures are key factors [9] [10]. Swarm intelligence is a relatively novel field which studies the collective behavior of multi-agent systems that coordinate using decentralized control and self-organization. In this kind of distributed and collaborative intelligence, an individual member has not the ability to perform an action efficiently. However, each member, using a kind of indirect communication named stigmergy, contributes to perform a task in an efficient way. Pheromone-based routing mechanisms mimic the cooperative foraging behavior of ant colonies using two phases to improve information discovery and gathering. In the exploration phase, ants lay down pheromone trails. Upon finding food, ants return to their colony reinforcing the trails with more pheromone. Other ants tend to follow these paths and to reinforce them releasing more pheromone. Over time, pheromone tends to evaporate to erase unused paths. In order to avoid stagnation, in which all ants tends to use the same trails, losing the ability to discover new relevant information sources, a certain probability of exploration of new trails may be considered.

This pheromone-based approach has inspired several routing mechanisms, which we classified by applying three criteria. 1) In terms of how is started the data sending process, three approaches can be defined, proactive [11] [12] (sources report values to sink after an event), reactive [13] (sources report values to sink on-demand based on queries), and hybrid approaches (a combination of proactive and reactive approaches), which are shown in Figure 1. Reactive schemes typically have better scalability than proactive ones because they require less control overhead [14]. In scenarios where data changes quickly, a reactive approach is more convenient. 2) In terms of the communication model, most works have considered unicast communication model but broadcast and hybrid models [13] have been also proposed. The first one does not take advantage of the broadcast nature of wireless communication (a transmitted packet is received by all neighbor nodes in the communication range). In addition, broadcast both enables multipath exploration and increases the robustness regarding packet loss and failed nodes, which is very common in WSNs. 3) Finally, in terms of the initiator of the process, three approaches have been proposed, source-initiated [11] [14] [15] [16], sink-initiated [17], and hybrid approaches.
This work proposes a pheromone-based sink-initiated 2-phase reactive routing mechanism which takes advantage of local and temporal pheromone levels in sensor nodes to direct messages to nodes with relevant information. The main differences between existing mechanisms and the proposed one are: (i) both exploration and query dissemination are integrated in one phase, (ii) the pheromone level is neither transported in the query message nor exchanged between neighbor nodes, this level is a local parameter of each node, (iii) the pheromone level is a simple value indicating the probability of the node to continue relaying the query rather than a pheromone vector or matrix with partial pheromone levels in function to each neighbor, (iv) most pheromone-based routing mechanisms are address-based, instead our proposal is data-based, which is more suitable for in-network processing, and (v) the list of visited nodes is not included in the query message, which is very common in pheromone-based algorithms. In this work we increase the analysis of PhINP mechanism of a previous work [18] both analyzing the parameter setting to obtain good performance (i.e., good trade-off between error and cost) and comparing the performance of PhINP with Gossip and Flooding mechanisms, applying the same in-network filtering strategies.

3. PhINP Mechanism

We consider a network conformed by sensor nodes disseminated over a square area, in which there is not a defined network topology. It is established in an ad-hoc manner based on local interactions (broadcast radio links), which are function of both node positions and communication range. In this context, one node takes the role of sink when it is accessed by a user requiring information. The surface, representing the physical magnitude to be sensed, changes continuously in time. Nodes are tasked with storing data obtained from sensing. Sink node, initiating a query process, has no clue about the information's location. PhINP mechanism makes use of three phases, *query dissemination*, *data gathering*, and *pheromone-level update*, which are described in the following subsections.

3.1 Query Dissemination

This phase is based on iterative search space reduction, adapting the query dissemination to the data distribution. It efficiently combines both a *pheromone-based strategy* to direct queries to nodes with relevant readings and *in-network filtering*, in order to limit no relevant nodes to answer. The query message is conformed by type (query or data message), identification, and threshold value (e.g., maximum value). It is worth mentioning that, in
order to implement the *maximum function*, nodes need to update the query threshold if their sensed value is larger (i.e., dynamic threshold). In order to reduce the communication cost, each node can relay the same query message only once.

On receiving a query message, each node must make two decisions, *if to answer the query reporting its data to the sink after a period of time*, and *if to continue relaying the query to its neighbors*. The first decision is based on both the query's threshold value and the node's reading, and the second one is based on the node's pheromone level. These two decisions generate four different kinds of node's behaviors, which are detailed in Figure 2 in case of searching for maximum reading. First, the node checks whether it has already received the same query or not by reading the message identification (id). To limit overloading the network, a received query is discarded if it has already been processed. This lets each node to forward a query message at most once. Next, the node reads the threshold value of the query message to determine the relevance of its reading. The information gain $\Delta E$ represents the difference between this threshold and the local sensed value. If $\Delta E < 0$, the reading is considered relevant and the node self-configures to answer the query after a period of time (Figure 2 a, c). Otherwise, the node has not relevant readings for the requirement (Figure 2 b, d). Finally, the node must decide if to relay the query to its neighbor nodes based on its pheromone level ($\lambda$). As higher $\lambda$, more likely to relay the query. If the node's pheromone level is low, the probability to continue relaying the query is low (Figure 2 c, d), otherwise, the query is likely relayed (Figure 2 a, b), changing the threshold value of query message if there is information gain (Figure 2 a) and increasing the hop level of the query message. Figure 2 and Algorithm 1 describe this process, in which extreme situations ($\lambda=0.1$ and $\lambda=0.9$) are considered.

![Figure 2. Query dissemination phase: (a) answer and relay query, (b) only relay, (c) only answer, (d) no action.](image-url)
Algorithm 1 - Query dissemination algorithm

1. id ← getId(queryMsg)
2. hops ← getHops(queryMsg)
3. thresholdValue ← getThreshold(queryMsg)
4. ΔE ← (thresholdValue – sensedValue)
5. // Discard already forwarded queries
6. if idTable [id] == true then
7. delete(queryMsg)
8. else
9. updateIdTable(id)
10. end if
11. // Answer decision
12. if ΔE ≤ 0 then
13. dataMsg ← createMsg(sensedValue)
14. scheduleSend(dataMsg)
15. answer ← true
16. end if
17. // Query relaying decision
18. if λ > rand() then
19. if ΔE ≤ 0 then
20. updateMsg(queryMsg, sensedValue) // Update threshold value
21. end if
22. send(queryMsg)
23. else
24. delete(queryMsg)
25. end if

3.2 Data Gathering

At the end of query dissemination phase, each node learns its distance to the sink (hop level) updating the minimum hop number from query messages received by multipaths. The response phase is triggered by all nodes selected after query dissemination phase. The triggering time is inversely proportional to the hop distance between sink and the selected node. This mechanism allows to disable (filter) the answer of previously selected nodes near to the sink node. In data gathering phase, a multipath routing strategy limited by hop level is applied to guarantee that sensor readings arrive at the sink, providing low latency and robustness to node failures and unreliable links. On receiving a data message, each node, based on the hop level information, decide whether to relay the message to sink node, if is so,
it puts its hop level in the message to relay. This is done when the hop level of the receiving
node is less than that included in message by sending node. Otherwise, the message is
discarded. This process is detailed in Figure 3 and Algorithm 2.

Algorithm 2 – Data gathering algorithm

1. hops ← getHops (dataMsg)
2. thresholdValue ← getThreshold (dataMsg)
3. ΔE ← thresholdValue – sensed
4. // Filter selected nodes to answer
5. if ΔE > 0 then
6. unscheduleSend (dataMsg)
7. answer ← false
8. end if
9. // Discard already received data messages
10. if idTable [id] == true then
11. delete (dataMsg)
12. else
13. updateIdTable (id)
14. end if
15. // Data message relay decision
16. if hops > minHops then
17. Msg ← setHops (minHops - 1)
18. send (dataMsg)
19. relayDataMsg ← true
20. else
21. delete (dataMsg)
22. end if

3.3 Pheromone-level update

Each sensor node periodically updates its pheromone level, which increases its
probability of forwarding a query message. The update rate is setup by the query process and
in general it is assumed equal to the monitoring frequency (i.e., the frequency at which
queries are disseminated). The first query message encourages all nodes to participate in this
phase by setting their maximum pheromone level (λ = 1). This guarantees that all nodes
receive the query (flooding) and that all nodes with relevant readings are properly selected.
After each iteration (i.e. query dissemination and data gathering phases), all nodes decrease
their pheromone level by $\lambda_{\text{dec}}$ up to a lower bound given by $\lambda_{\text{min}}$. Only nodes which have been
either selected to answer the query (i.e., send sensed data) or relayed data back to the sink,
increase their pheromone level by $\lambda_{inc}$ up to an upper bound given by $\lambda_{max}$ (such as in MAX-MIN Ant System [19]). In our proposal, the $\lambda_{max}$ level is limited to 1, and the $\lambda_{min}$ level can be adjusted based on the application requirements. In this last case, it is a trade-off between query cost and error, because for low $\lambda_{min}$ values the query cost tends to be low but the error increases, and vice versa. The goal of this phase is to reinforce those paths directed towards nodes with relevant data, which is described in Algorithm 3.

Figure 3. Data gathering phase: data messages are relayed to the sink based on the node's hop level

Algorithm 3 - Pheromone update algorithm

1. if $\lambda > \lambda_{min}$ then
2. $\lambda = \lambda - \lambda_{dec}$
3. end if
4.
5. if answer == true or relayDataMsg then // pheromone reinforcement
6. if $\lambda < \lambda_{max}$ then
7. $\lambda = \lambda + \lambda_{inc}$
8. end if
9. end if

4. Main Results

PhINP mechanism was evaluated with a discrete event network simulator developed using Omnet++ [20]. Query error, query cost and active nodes metrics were considered in order to compare the performance of this proposal with respect to existing ones. The first
metric represents the gap between the best reported data to the sink after a query iteration and 
the actual optimal one (i.e., the relative error). The second metric considers the number of 
nodes on average involved in forwarding the query message. Note that the maximum query 
cost is equal to 1, which means that all nodes relayed the query once (i.e., flooding). This cost 
is proportional to energy consumption as power requirements are dominated by 
communications. The latter metric depicts the proportion of active nodes in the network, that 
is, nodes with residual energy to keep running.

The performance of PhINP was compared with broadcast-based flooding and gossip 
mechanisms. However, the same in-network filtering strategies are applied in the three 
mechanisms. The only difference is how the query is disseminated in the network. Flooding 
is a deterministic approach, where the probability of each node to relay the query message is 
equal to one. As a consequence, queries are disseminated to all nodes. Gossip is a 
probabilistic approach, where the probability of each node to relay the query is based on the 
gossip level, which is fixed and equal in all nodes. This parameter can take values from 0 to 
1. Low gossip levels mean that the query has low probability to be relayed for each node and, 
as a consequence, few nodes participate in the query dissemination, and viceversa. Gossip 
applying a gossip level equal to one behaves as flooding. Finally, PhINP is an adaptive 
approach, in which the probability to relay the query is based on node's pheromone level. 
This parameter is individually adapted between $\lambda_{\text{min}}$ and $\lambda_{\text{max}}$ in each iteration in order to 
obtain paths or trails from sink to sensor nodes with relevant information. These levels are 
limited between 0 and 1.

The simulation scenario is conformed by nodes uniformly deployed over a square surface 
with a constant node density of $2.5 \times 10^{-3}$ nodes/square meter, assuming a circular 
communication range of 40 meters. Therefore, if the network size is increased, the coverage 
area is extended, maintaining a constant node density. The sensed field is modeled as the sum 
of $S$ decreasing exponential functions with different values of position, amplitude, and 
decrement. Sensor readings are function of node positions in the field. The field model 
assumes multiple independent sources such as air conditioners, lights, speakers, etc. Our 
simulations consider the case of $S=160$ and report the average results between 2000 and 5000 
random simulations.

In the followings subsections, we analyze the convergence and sensitivity of PhINP, the 
performance compared with flooding and gossip mechanisms, and the robustness of PhINP 
respect to packet loss and failed nodes obtained through simulation.

4.1 Convergence and sensitivity

Since the proposed scheme first floods the network, the error cost remains null in the 
following iterations if there are no changes in the sensed field. In fact, the error remains near 
to 0 due to a low probability of disconnected nodes in some scenarios. Under these 
conditions, the convergence of the algorithm can be analyzed. At each iteration, paths to 
relevant data are reinforced, decreasing the query cost. This behavior is shown in Figure 4 for 
different network sizes. Note that the cost is almost independent of the network size. Figure 4 
shows that, for a low $\lambda_{\text{min}}$ level, the convergence is slower than for high levels, but with a
lower query cost. In the first case, a convergence is approximately obtained at 12 iterations. As a consequence, a trade-off exists between convergence time and minimum cost.

Since monitoring implies to be able to track changes in the sensed field at minimum cost, we evaluate the sensitivity of the proposed scheme to these changes. Ideally, once the scheme converges to some minimum query cost, it should be able to still detect relevant data with some error. The sensitivity to changes depends on the minimum level of pheromone $\lambda_{\text{min}}$ that nodes can have. We evaluate this sensitivity for $\lambda_{\text{min}} = 0.1$, considering a network with 400 nodes. To this end, after convergence (12 iterations), the amplitude of each data source is changed randomly up to 90%. In Figure 5, different $\lambda_{\text{inc}}/\lambda_{\text{dec}}$ relationships are analyzed. The larger the $\lambda_{\text{inc}}/\lambda_{\text{dec}}$ relationship, the faster the convergence to a minimum error; however, the cost tends to increase in the following iterations. As a conclusion, a $\lambda_{\text{inc}}/\lambda_{\text{dec}}$ relationship of 2 is adequate for continuous fields with slow variation, however, for fields with high variation a value of 4 is more convenient.

![Figure 4. Convergence and sensitivity: query cost for (a) $\lambda_{\text{min}} = 0.1$ and (b) $\lambda_{\text{min}} = 0.5$.](image)

![Figure 5. Convergence and sensitivity: query error (a) and cost (b) for different $\lambda_{\text{inc}}/\lambda_{\text{dec}}$ relationships.](image)

From simulations, we concluded that a good error-cost trade-off can be obtained using $\lambda_{\text{dec}} = 0.1$, $\lambda_{\text{inc}} = 0.2$ or 0.4 (according to the field dynamic), $\lambda_{\text{min}} = 0.1$ and $\lambda_{\text{max}} = 1$. 

www.macrothink.org/npa
4.2 Performance

The performance of PhINP mechanism is compared with broadcast-based flooding and gossip query dissemination schemes for a network size of 400 nodes, which is shown in Figure 6. This comparison considers only query disseminating cost (i.e., query cost) as the cost of replies is negligible with respect to the query one. Flooding disseminates the query to all nodes, while gossip to only half of nodes (using a gossip level = 0.5), which are randomly selected as the query is propagated in the network. PhINP is configured using $\lambda_{\text{dec}} = 0.1$, $\lambda_{\text{inc}} = 0.2$, $\lambda_{\text{min}} = 0.1$ and $\lambda_{\text{max}} = 1$. Our proposal performs like flooding in terms of errors but shows a cost reduction of 60%. As expected, gossip is able to limit the query cost but with a high error. These results validate that our proposal is able to search and learn where relevant data are located and reduce the cost of disseminating the query to these areas.

![Figure 6. Performance comparison respect to amplitude change](image)

Finally, network lifetime (analyzing active nodes) and search error is analyzed for gossip, flooding and PhINP mechanisms. We suppose that each sensor node is powered by two AA alkaline batteries, with 9360 Joules each one. A standard IEEE 802.15.4 radio transceiver at 0 dBm power output approximately requires 0.12 mJ for the transmission or reception of a byte [21]. A packet size of 20 kB is defined, which enables either the transmission or reception of 7800 packets for each node, considering negligible the energy for sensing and computing task. After that, a sensor node fails generating holes in the network. Additionally, we consider that the sink node has unlimited energy, because it could be connected to the power line. Figure 7 shows the query cost and active nodes as a function of the query iteration.

We can see that PhINP outperforms similar to Flooding respect to the query error, but the network using flooding has a fast decrement of the active nodes from 300 iterations, with a convergence to 60% active nodes. Gossip and PhINP schemes, instead, performs the quantity of active nodes of flooding, increasing the network lifetime. Analyzing the energy levels in the network, we note that nodes near to sink node have high energy consumption due to the task of response relaying. Network lifetime can be improved applying sink node selection given a period of time.
4.3 Robustness

Finally, the robustness of PhINP to network changes is analyzed. To this end, we evaluate its performance under packet loss conditions. Figure 9 shows the query error and cost as a function of both the network size and packet loss probability using $\lambda_{\text{dec}} = 0.1$, $\lambda_{\text{inc}} = 0.2$, $\lambda_{\text{min}} = 0.1$ and $\lambda_{\text{max}} = 1$. Probabilities up to 10% of packet loss keep the error below 5% for networks of up to 400 nodes. Under different packet loss conditions, the cost remains almost the same. The robustness of PhINP respects to failed nodes and query iteration is shown in Figure 7 a.
5. Conclusions

In this work, we propose a simple and efficient mechanism to be used for monitoring continuous fields using wireless sensor networks. The proposed scheme tackles the query dissemination problem applying in-network filtering, reinforcing distribution paths using an iterative pheromone-based process. Results show that the query cost can be significantly reduced while keeping errors very low, outperforming flooding and gossip mechanisms. Ongoing works includes the implementation of PhINP mechanism in real sensor networks.

Acknowledgement

This work is partially funded by the FONCyT - National Technological University IP-PRH 2007 Postgraduate Grant Program and by the SECyT - National University of Cordoba 2010-2011 Research Program.

References


Copyright Disclaimer

Copyright reserved by the author(s).

This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/3.0/).