Data Collection in Delay Tolerant Mobile Sensor Networks using SCAR

Cecilia Mascolo
Dept. of Computer Science, University College London
Gower Street, London
WC1E 6BT, United Kingdom
c.mascolo@cs.ucl.ac.uk

Mirco Musolesi
Dept. of Computer Science, University College London
Gower Street, London
WC1E 6BT, United Kingdom
m.musolesi@cs.ucl.ac.uk

Bence Pásztor
Dept. of Computer Science, University College London
Gower Street, London
WC1E 6BT, United Kingdom
b.pasztor@cs.ucl.ac.uk

1. OVERVIEW AND MOTIVATION

Sensor devices are now present in virtually all sorts of items, from vehicles and furniture to humans and animals. This generates networks of wireless connected devices with topologies which could be very dynamic. The monitoring abilities of these devices range from pollution and temperature to health-care and mobility. The amounts of data generated by these applications are usually quite large, however, fortunately, the data is, in most cases, also delay tolerant, in the sense that it can wait in the network for quite a while before being collected.

The scenario we envisage in this demonstration is one where mobile sensor nodes (e.g., animals, vehicles or humans) route data through each others in order to reach sink nodes, which can be either mobile or fixed. The fixed nodes are intended as nodes connected to a backbone network and therefore able to forward the data to the appropriate destination.

The challenges offered by this scenario are many and include the quantity of data to be shipped to the sinks, the potentially scarce communication power (i.e., energy and bandwidth) of the nodes, the possible communication and sensor hardware faults, the mobility and the scarce buffer size of the nodes.

Different techniques could be employed for mobile sensor data gathering. A basic strategy would be to only allow data delivery when sensors are in direct proximity of the sinks. This technique has very little communication overhead, given that messages are only sent directly from the sensor node generating messages to the sink. However, depending on how frequently sensor nodes meet the sinks, the delivery of the data might be very poor. This is particularly true if the sinks are very few and spread out. More refined techniques include epidemic-style approaches [6], which would spread the data over the sensor network, so that eventually a sink could be reached. This approach has very good delivery ratio if buffers are sufficiently large, however the overhead in terms of communication and, therefore, energy is quite high. Some solutions have been presented in literature, including the use of data mules with scheduled routes [3].

Energy and, therefore, communication overhead is an issue, the spreading of the messages needs to be carefully controlled and traded off for the delivery ratio. This is even more true if the nodes have limited memory so that the buffer size is small and very few messages can be stored. Moreover, we consider scenarios where the route of the potential message carriers may not be known a priori, as in some of the cases of systems where devices attached to animals or humans are exploited to deliver the data to the sinks.

Starting from these considerations, we have developed SCAR (Sensor Context-Aware Routing) [5], a routing approach which uses prediction techniques over context of the sensor nodes (such as previously encountered neighbors, battery level, etc.) to foresee which of the sensor neighbors are the best carriers for the data messages. We further adopt different classes of messages in order to achieve an intelligent buffer management. In this demonstration, we will show the feasibility of the implementation of this protocol, based on Kalman filter prediction techniques [1]. We will also prove that our protocol is able to deliver the data to the sinks without the a priori knowledge of the routes of the carriers.

We now first introduce the protocol, then we give an overview of its implementation on sensors running Contiki [2]. We will then describe the setting of the actual demonstration.

2. SCAR ROUTING

We provide the key concepts of the protocols. These are useful to appreciate the technical contribution of our demonstration. Additional details can be found in the appendix of this paper. The complete description of the protocol are presented in [5].

The decision process by which nodes select the best carriers is based on prediction of the future evolution of the system. Our solution relies on the analysis of the history of the movement pattern of the nodes and their colocation with the sinks and on the evaluation of the current available resources of the sensors. In particular, each node evaluates its buffer size is small and very few messages can be stored. Moreover, we consider scenarios where the route of the potential message carriers may not be known a priori, as in some of the cases of systems where devices attached to animals or humans are exploited to deliver the data to the sinks.

Starting from these considerations, we have developed SCAR (Sensor Context-Aware Routing) [5], a routing approach which uses prediction techniques over context of the sensor nodes (such as previously encountered neighbors, battery level, etc.) to foresee which of the sensor neighbors are the best carriers for the data messages. We further adopt different classes of messages in order to achieve an intelligent buffer management. In this demonstration, we will show the feasibility of the implementation of this protocol, based on Kalman filter prediction techniques [1]. We will also prove that our protocol is able to deliver the data to the sinks without the a priori knowledge of the routes of the carriers.

We now first introduce the protocol, then we give an overview of its implementation on sensors running Contiki [2]. We will then describe the setting of the actual demonstration.

2. SCAR ROUTING

We provide the key concepts of the protocols. These are useful to appreciate the technical contribution of our demonstration. Additional details can be found in the appendix of this paper. The complete description of the protocol are presented in [5].

The decision process by which nodes select the best carriers is based on prediction of the future evolution of the system. Our solution relies on the analysis of the history of the movement pattern of the nodes and their colocation with the sinks and on the evaluation of the current available resources of the sensors. In particular, each node evaluates its buffer size is small and very few messages can be stored. Moreover, we consider scenarios where the route of the potential message carriers may not be known a priori, as in some of the cases of systems where devices attached to animals or humans are exploited to deliver the data to the sinks.

Starting from these considerations, we have developed SCAR (Sensor Context-Aware Routing) [5], a routing approach which uses prediction techniques over context of the sensor nodes (such as previously encountered neighbors, battery level, etc.) to foresee which of the sensor neighbors are the best carriers for the data messages. We further adopt different classes of messages in order to achieve an intelligent buffer management. In this demonstration, we will show the feasibility of the implementation of this protocol, based on Kalman filter prediction techniques [1]. We will also prove that our protocol is able to deliver the data to the sinks without the a priori knowledge of the routes of the carriers.

We now first introduce the protocol, then we give an overview of its implementation on sensors running Contiki [2]. We will then describe the setting of the actual demonstration.

2. SCAR ROUTING

We provide the key concepts of the protocols. These are useful to appreciate the technical contribution of our demonstration. Additional details can be found in the appendix of this paper. The complete description of the protocol are presented in [5].

The decision process by which nodes select the best carriers is based on prediction of the future evolution of the system. Our solution relies on the analysis of the history of the movement pattern of the nodes and their colocation with the sinks and on the evaluation of the current available resources of the sensors. In particular, each node evaluates its buffer size is small and very few messages can be stored. Moreover, we consider scenarios where the route of the potential message carriers may not be known a priori, as in some of the cases of systems where devices attached to animals or humans are exploited to deliver the data to the sinks.
While moving, the sensors will transfer their data to other sensors only if these have a higher probability to deliver the data to sinks (i.e., they are better carriers). The calculation of the delivery probability is local and it does not involve any distributed computation. Nodes exchange information about their current delivery probability and their available buffer space with the neighbors only periodically.

3. PROTOTYPE IMPLEMENTATION

We have implemented SCAR on the Tmote Sky nodes using the Contiki Operating System \[\text{[1]}\]. The Kalman filter predictor has been implemented to allow decision making on forwarding and buffering. As shown in the demonstration, our implementation is very lightweight.

The sensors are equipped with low-power microprocessors. As these are simple, 16-bit processors (TI MSP430), they do not provide floating-point support, which is essential for the calculation of the Kalman predictions. To enable the algorithm to work, an emulator had to be used. The 16-bit processor limits the accuracy of the calculations, however, for the purpose of the predictions, this is acceptable. The algorithm is implemented using protothreads running in the Contiki kernel. Each thread is given a task, such as calculating and sending the predictions of the sensor, receiving other mote's information (routing messages) and storing it in a table, and sending and receiving the actual messages. The threads are executed periodically (in case of routing messages), or when necessary (if there is a message in the buffer to send). The messages are sent as UDP packets, using Contiki libraries. Each mote is given a unique IP address during set-up. Messages are stored in a buffer as C structures the buffer is limited to \(1\) XX (10?) number of messages. Information about other sensors are also stored as C structures, and updated every time a new routing message is received. Also, when a message reaches the sink the message with the path (the IP addresses of the sensors traversed) is displayed, and deleted from the buffer.

4. SCENARIO AND DEMO DESCRIPTION

The aim of this demonstration is to show how the protocol is able to select the right carriers for the data generated by the sensors and then to deliver them to the sinks. As we have explained, the choice of the carrier(s) is based on information related to battery power, connectivity, buffer space of the sensors.

The setting of the demonstration is shown in Figure 1 and will be as follows:

- T-mote Sky A, acting as a data (temperature) source, will be placed in a corner (far from the demonstration booth) of the demonstration hall;
- T-mote Sky B, connected to a laptop through USB, acting as a sink and will be placed on the booth table; sensors A and B are not in reach with each other as indicated in the picture; the laptop will act as a display for the received data;
- 5 T-Mote Skys attached to people or on Lego MindStorms; these will represent the mobile carriers. These sensors will have different conditions: in particular, sensors C, D, E are often in reach of the source A, while F and G are not and instead they are often in reach of sink B. Initially only sensor C has a full buffer, while sensor D and E do not, later also the buffer on sensor D will fill up. Sensor F has high battery levels while sensor G has low power.

The demonstration will show how initially node A running SCAR will select node E and/or D as first hops, while later in time, it will only select C. The next hop for the data then will be F as it has higher battery. From F the data has a chance to reach sink B as F is frequently roaming towards B. When the data reaches the sink, a message showing the path the message took, the time, and the temperature value, will be displayed. We also will show the times related to the decision making process guided by the Kalman filters in the different stages.

Depending on the space allocated for the demonstration, we can use some foil to screen the motes’ antenna to restrict their range and allow the demonstration to run in more constrained environments.

5. REFERENCES


\[\text{Cecilia: fix}\]
APPENDIX

A. THE KEY CONCEPTS OF THE SCAR PROTOCOL

In this appendix, we present a more detailed overview of the SCAR protocol.

Each sensor that is the source of data tries to place bundles on a number of neighboring nodes which have the best chance to deliver them to a sink node. Each node maintains an ordered list of the neighboring nodes (including itself) decreasingly ordered according to their delivery probabilities. Each node then replicates the bundle to the first $R$ nodes ($R - 1$ nodes if the node itself is in the first $R$ positions of the list). The value of $R$ is specified by the user and it can be considered as a priority level associated to the data retrieved by the sensor.

The replica sent to the node with the highest delivery probability is labeled as master copy. The other replicas are labeled as backup copies. These can be overwritten if buffers are full, whereas master copies are deleted only when sensors exchange the data with the sinks. In general, this distinction is used for an intelligent management of the buffer that gives higher priority to master copies.

Each node keeps monitoring if there are neighbors with better probability of delivery than its own. If this is the case, the data bundles are shifted from one buffer to the other. This, however, implies that the data bundles are only replicated on a number of nodes in the first hop, while they are forwarded (i.e., deleted from one node and copied on another), later if the carriers, while roaming, find either a sink or a better carrier.

As we are in a sensor network, the high level of faults in the nodes implies that we need to allow for some more replication on the data. However, if the amount of data generated by the sensors is considerable, the approach of replication adopted by both epidemic-like protocols and in [?] incurs in heavy overheads. We replicate less but we try to control the replication in an intelligent way by predicting the future evolution of the system. In other words, data are replicated $R$ times, with $R$ that may be order(s) of magnitude less than the number of sensors composing the system.

The delivery probability of the nodes also keeps into account the energy level of the nodes, so to avoid that some best carriers become strong attractors and run into low battery problems more quickly than others. In other words, we will show that as the battery level decreases, the probability of being selected decreases.

In order to select the best carrier(s) for the data bundles, we use a mechanisms based on the estimation of the future behavior of each sensor node based on the history of its colocation with sinks, its changing rate of connectivity (i.e., its mobility), and its power level.

Each node predicts, using time series forecasting techniques, the evolution of its context described by a set of attributes. In particular, we consider three indicators describing its colocation with the sinks, its changing degree of connectivity and its battery level.

More specifically, a utility function is associated to each context indicator. Our aim is to maximize each attribute, in other words, to choose the node that presents the best trade-off between the attributes representing the relevant aspects of the system for the optimization of the bundles delivery process. Analytically, considering $k$ attributes with associated utility functions $U_1(s_i), ..., U_k(s_i)$, the problem can be reformulated as a multiple criteria decision problem [4] with $k$ goals:

$$Maximize\{U(s_i)\} = f(U_1(s_i), ..., U_k(s_i))$$

(1)

The combined goal function using the the so-called Weights method can be defined as

$$Maximize\{\sum_{j=1}^{n} w_j U_j(s_i)\}$$

(2)

where $w_1, w_2, ..., w_n$ are significance weights reflecting the relative importance of each goal.

In our case, the solution is very simple, since it consists in the evaluation of the function $f(U_1, ..., U_k)$ using the values predicted for each node and in the selection of the node(s) with the maximum such value.

The overall utility function $U(s_i)$ gives a measure of the probability that a node $s_i$ ability of delivering bundles to the sinks (i.e. of being co-located with them in the future). The delivery probability of each sensor will be equal to its composed utility function. More formally, the delivery probability of a sensor $s_i$ is defined as

$$P(s_i) = U(s_i)$$

(3)

Two devices are co-located if they are in the same transmission range (i.e. one hop distance). Therefore, this utility function is computed considering its relative mobility (calculated by evaluating its change degree of connectivity history), its colocation with sinks, and its survivability (calculated by considering its battery level history)\(^2\). We associate a utility function to each of these indicators, respectively $U_{cdc}(s_i), U_{coloc}(s_i)$ and $U_{battery}(s_i)$, and we compose these utility functions using a weighted sum as follows:

$$U(s_i) = w_{cdc} U_{cdc}(s_i) + w_{coloc} U_{coloc}(s_i) + w_{bat} U_{bar}(s_i)$$

(4)

\(^2\)Even if we take into consideration only these three context indicators, our framework allows for the integration of other utility functions describing other aspects of the system that may be important to improve the performance of the storing-and-forwarding strategy.