Optimising Quality of Information in Data Collection for Mobile Sensor Networks

Farshid Hassani Bijarbooneh, Pierre Flener, Edith Ngai, and Justin Pearson

Uppsala University, Department of Information Technology, Box 337, SE–75105 Uppsala, Sweden

Email: {Farshid.Hassani, Pierre.Flener, Edith.Ngai, Justin.Pearson}@it.uu.se

Abstract—**Wireless sensor networks have become increasingly popular for environmental and activity monitoring, such as temperature, pollution, parking space, traffic, and crowd monitoring. Mobile users can collect and visualise sensing data by communicating with wireless sensors along their walks using Bluetooth or NFC. They can also share the sensing data on the Internet through 3G or WiFi connectivity. Nevertheless, mobile users may not be able to collect all the data from the sensors due to limited contact times and batteries. It is crucial to collect data with a maximum amount of information from the available resources. In this paper, we tackle the problem by prioritising the sensing data to maximise the data utility considering the quality of information of the sensing data and the communication overhead. We formulate the optimisation problem and propose a greedy algorithm for clustering the sensors and scheduling the data collection. Our greedy algorithm coordinates the mobile users in the sensing field in order to avoid the collection of redundant sensing data. We evaluate the data utility and energy consumption of the proposed algorithm using real mobility traces from the North Carolina state fair. The results demonstrate that our algorithm can significantly improve data utility at low communication overhead compared with an existing algorithm.**

I. INTRODUCTION

Wireless sensor networks consist of distributed, wireless-enabled, embedded devices capable of employing a variety of electronic sensors. Each node in a wireless sensor network is equipped with one or more sensors in addition to a microcontroller, wireless transceiver, and energy source. Wireless sensors can be deployed for pollution monitoring, radiation detection, traffic and parking management in smart cities [1], [2], and so on. For example, pollution sensors are deployed along roads to monitor the level of particles from car emissions, such as carbon monoxide (CO), carbon dioxide (CO_2), and nitrogen dioxide (NO_2). Cameras could be deployed to monitor crowd, crimes, or traffic, and to locate parking spaces. Wireless sensors are also essential for industrial control, logistics, retail, smart agriculture, home automation, and e-health [3].

The most attractive feature of wireless sensor networks is their autonomy. When deployed in the field, a wireless sensor is capable of communicating with every other node in range through Bluetooth or near field communication (NFC), which creates an ad hoc mesh network for relaying information to and from the gateway node. With the advent of mobile phones, mobile users can act as mobile gateway nodes to collect data from the wireless sensors and share the data on the Internet or with other mobile users. Opportunistic data collection with

mobile devices allows wireless sensors to be deployed at almost any location without considering the density of wireless sensors or the underlying network infrastructure. Coupled with the low cost and almost limitless supply of available sensing modules, wireless sensor networks offer much flexibility and potential for numerous application-specific solutions.

Mobile users are collecting information of interest sensed by the surrounding wireless sensors along their walks. Their trajectories are random and uncontrollable. Due to the limited contact time and wireless communication range, the mobile users may not be able to collect all the data from the wireless sensors. As contact might be scarce and short, there is a need to maximise the value of information to be received by the mobile users. Recently, quality of information (QoI) has been suggested to measure the *value* of sensed data, which is a multidimensional and application-specific metric affected by multiple factors, which include its location, its sensing modalities, ambient noise levels, sensing channel conditions, fault status, and physical process dynamics [4], [5].

In this paper, we consider the QoI of sensing data and maximise the utility in data collection for mobile users in wireless sensor networks. We define *data utility* as a metric computing the sum of the qualities of information from the sensed data divided by their communication overhead in data collection. The wireless sensors in the sensing field periodically take measurements from the environment and cache the sensed data in their buffers. When the wireless sensors are approached by the mobile users, the mobile phones carried by the mobile users can communicate with the sensors to collect the cached data. We formulate the optimisation problem and propose a two-step greedy algorithm to maximise data utility in data collection. The mobile users cluster the sensing field to avoid the collection of redundant data and prioritise the collection of sensing data according to their data utility. We demonstrate the efficiency of the proposed algorithm by simulations using real mobility traces from the North Carolina state fair.

The contributions of this paper are as follows. (1) We present a novel QoI-aware data collection scheme that utilises voluntary mobile users to collect data from their surrounding sensors. (2) We formulate the optimisation problem and propose a greedy algorithm for maximising the QoI per communication overhead from the collected data. (3) We conduct extensive simulations with real mobility traces to evaluate the QoI and communication overhead of our proposed solution.

The remainder of this paper is organised as follows. In

Section II we discuss the related work. In Section III, we give the system overview. In Section IV, we describe the formulation of our optimisation problem on QoI-aware data collection. In Section V, we present our two-step greedy algorithm for maximising data utility in data collection. In Section VI, we conduct extensive simulations to evaluate our solution based on real mobile traces. Finally, we summarise and conclude the paper in Section VII.

II. RELATED WORK

Data collection has been widely studied for stationary wireless sensor networks. Gnawali *et al.* [6] present the state-of-the-art routing protocol for a sensor network where the nodes are forwarding data directly to a sink. They consider stationary WSNs that have static routes from the wireless sensors to the sink. With the advancement of mobile devices, mobile nodes have been considered to improve the performance of data collection. Many works consider mobile devices walking pre-defined paths to collect data from stationary sensors through wireless communication. Shah *et al.* [7] present an architecture using moving entities, called *data mules*, to collect sensing data. Gatzianas and Georgiadis [8] consider a sensor network where multiple mobile agents collect the data. They suggest the sink to stay at specific sojourn points to collect data periodically with a distributed algorithm. Similarly, Gu *et al.* [9] propose a partition-based algorithm to schedule the movement of mobile elements, which minimises the required moving speed and eliminates buffer overflow. Bisnik *et al.* [10] further study the problem of quality coverage on event detection using mobile sensors. They also analyse the effect of controlled mobility to the fraction of events captured. In this paper, in departure from these works, we consider mobile users with *uncontrolled* mobility in data collection. The mobile users do not walk along pre-defined paths. They collect sensing data automatically while they are moving naturally under normal activities.

Uncontrollable mobility patterns have been considered recently for mobile sensor networks. Kusy *et al.* [11] propose an algorithm to predict the mobility pattern of mobile sinks using a mobility graph. They precompute and store routing states in the network in order to improve the reliability of wireless sensor networks involving mobile sinks. In addition, Lee *et al.* [12] introduce data stashing in order to forward the collected data to multiple mobile sinks. They predict the movement of the mobile sinks and stash the data in advance in a set of selected relay nodes and wait for the mobile sinks to arrive. As a mobile sink passes by the relay nodes, they forward the stashed data to the mobile sink. This method involves an offline procedure of learning the trajectories of the mobile sinks in order to predict a set of potential future trajectories. This offline procedure is formulated as the multiple sequence alignment problem, which is NP-hard. Li *et al.* [13] also propose a ubiquitous data collection scheme that allows mobile users to collect network-wide sensing data from static wireless sensors. The proposed scheme can efficiently form a new data collection tree or locally update the previous data collection tree with the movement of the mobile users. Existing work has thus explored data collection for mobile sensor networks with *small* variations. However, none of them has been focusing on optimising the quality of information (QoI) considering the limited contact time and the remaining battery life of the mobile devices.

Recent work has already explored the optimisation of QoI for fundamental networking operations such as rate control, scheduling, and routing. Tan *et al.* [14] consider the design of QoI-aware routing in sensor networks. Urgaonkar *et al.* [15] consider a model for QoI-aware scheduling in task processing networks focusing on the accuracy and freshness of the sensing data. Using dynamic programming and optimal stopping theory, they characterise the optimal scheduling policy that maximises the average utility delivered by the network. Wang et al. [16] investigate maximising QoI subject to cost constraints in data fusion systems. They consider data fusion applications that try to estimate or predict some current or future state for target tracking, path planning, and sensor node localisation. Rather than optimising generic network-level metrics such as latency or throughput, they explore resource-efficient sensor network operation by directly optimising an application-level notion of quality, namely prediction error. Although application-level notions of QoI have thus been explored for WSNs, the optimisation of QoI for data collection in sensor networks by mobile users with uncontrolled mobility remains to be further explored.

III. SYSTEM OVERVIEW

A. Network Model

We consider a sensing field with stationary sensors deployed to monitor environmental data, such as noise level, temperature, pollution, person count, and unusual events. This sensing information could be interesting to mobile users who are participating in different kinds of activities. For example, participants in a state fair may want to obtain information about free spaces in the parking lots and numbers of people in restaurants, shows, or special events at particular locations. Figure 1 shows the map of a state fair, where locations of interest are identified and numbered by the organiser. They could be the locations of popular booths, restaurants, shows, parking lots, and toilets, which are all ideal places for deploying stationary sensors for monitoring the environment and activities.

Since local area networks may not be available in an open field, the wireless sensors deployed are not necessarily connected to any fixed network infrastructure. Opportunistic data collection by mobile users with smart phones could be beneficial to deliver the data from the wireless sensors to mobile users and the Internet. Most smart phones are equipped with Bluetooth, which enables them to communicate with the surrounding wireless sensors. Smart phones allow their users to visualise the data and upload it to the Internet through 3G connectivity. This supports large-scale data dissemination for both current participants in the state fair and future participants on the Internet. A base station in our network model

Fig. 1: The map of a state fair, with the numbers indicating locations where stationary sensors could be deployed to monitor unusual events, amounts of crowd, noise level, temperature, etc.

communicates with the mobile nodes through 3G internet connectivity and receives the uploaded data for aggregation and coordination.

B. Protocol Design

In our setup, as the mobile nodes move in the sensing field, the data collection is repeated every time a fixed time period, called a *round* (typically 30 seconds), elapses. A new round begins when a mobile node enters the sensing field, or when the previous round ends. Each round includes the following three phases:

- Phase 1: Each mobile node maintains two lists of its immediate neighbour sensor nodes (one hop away from the mobile node) as candidate sink nodes, namely one list for the current round and one list for the previous round. At the beginning of each round, if the new list of candidate sink nodes of a mobile differs from the previous one (due to the movement of the mobile node to a new location out of the vicinity of the candidate sink nodes from the previous round), then the mobile node requests the sink nodes to initiate a neighbourhood discovery with a given maximum number of hops. The neighbourhood discovery protocol is a variation of the tree discovery in [17] (also see route computation in [6]), where the broadcast message in the neighbourhood discovery protocol differs from the one in [17] and [6] by the information it carries. The broadcast message sent from a sensor node using the neighbourhood discovery protocol needs to carry extra information required by the quality of information scheme, namely the quality of its information, the size of the data in its buffer, and the identifiers of its neighbour sensor nodes in addition to the primary information of the one in [17] and [6]. Each mobile node reports the discovered communication links and the quality of information of the sensor nodes to the base station, and requests a scheduling of the sensor nodes for the data collection.

- Phase 2: Preprocessing is done on the base station, and it aggregates the information received from all the mobile nodes, and computes a list of shortest paths and hop counts from the sensor nodes to each mobile node. The base station uses our clustering algorithm (further explained in Section V-A) to assign a cluster of sensor nodes to each mobile node, and solves an optimisation problem to schedule the data collection (further explained in Section V-B) by maximising the data utility and minimising the communication overhead.

- Phase 3: Each mobile node receives the scheduling of the data collection from the base station, and initiates the data collection by requesting the data from the sensor nodes in the order specified in the schedule. The schedule information is cached at the mobile node, to be reused in the next round if the mobile node is not moved out of the vicinity of its immediate neighbour sensor nodes (the candidate sink nodes).

Throughout this paper, we use QoI-aware collection protocol (QoIACP) to refer to the protocol design above.

IV. Problem Formulation

In each round, let M be the set of mobile nodes moving freely to collect data from the stationary sensors, and let S be the union of all sensor nodes that are reachable by the mobile nodes with a given limit $MaxHopCount$ on the number of hops. Throughout this paper, the identifiers of constants always start with uppercase letters, and the identifiers of unknowns (or variables) and indices always start with lowercase letters.

As explained in Section III-B, in each round, the result of the neighbourhood discovery is reported to the base station by each mobile node. The neighbourhood discovery data for a mobile node $m \in M$ contains the list of sensor nodes that are reachable by m within $MaxHopCount$ hops. The neighbourhood discovery data also includes the quality of information $Q[s]$ of each sensor node s, in the range from 0 to 100, the size of the buffered data $D[s]$ in kB for sensor node s, the set $SinkNodes[m]$ of the sink nodev identifiers of mobile node m, and the set $Neighbours[s]$ of the neighbours of the reachable sensor nodes by m. The base station performs a preprocessing of the neighbourhood discovery data for all the mobile nodes, and computes the number $Hops[s, m]$ of hops for a shortest path from each sensor node s to each mobile node m.

Let $x[s, m]$ be a Boolean variable with value 1 if the sensor node s is selected for the data collection by the mobile node m, and 0 otherwise. We require the sensor nodes to deliver data with the highest quality of information and the lowest cost (communication overhead), therefore the data utility gained for each pair of sensor node s and mobile node m is:

$$u[s, m] = \frac{D[s] \cdot Q[s] \cdot x[s, m]}{Hops[s, m]} \qquad (1)$$

where $u[s,m]$ represents the data utility for communication between sensor node s and mobile node m. The data size generated in each round is equal to the data rate (bytes per second) of s multiplied by the length T in seconds of each round. In our work, the data rate is 200 bytes per second for all sensor nodes, and the data is buffered for 10 rounds, hence the total size of the buffered data is $D[s] = 60$ kB if $T = 30$ seconds.

The objective is to maximise the total data utility gained for the communication between the sensor nodes and the mobile nodes:

$$\texttt{maximise} \quad \sum_{m \in M} \sum_{s \in S} u[s,m] \quad (2)$$

Note that maximising the total data utility (2) causes an increase in the communication costs as the data with higher utility might be routed from sensor nodes several hops away from the mobile nodes.

The *capacity constraints* enforce that the size of the data sent to any mobile node does not exceed the capacity of that mobile node during any round:

$$\forall m \in M : \quad \sum_{s \in S} D[s] \cdot x[s,m] \leq C[m] \cdot T \quad (3)$$

where $C[m]$ is the capacity in kB/second of the mobile node $m \in M$. In our work, the maximum bandwidth for data collection from mobile nodes is 10 kB/second, indicating that each mobile user can collect 300 kB of data in each round if $T = 30$ seconds.

The *communication constraints* enforce that any sensor node sends data to at most one mobile node in each round:

$$\forall s \in S : \quad \sum_{m \in M} x[s,m] \leq 1 \quad (4)$$

Indeed, recall that not every sensor node is selected for data collection at every round.

In a solution to our problem formulation with the objective function (2) and just the constraints (3) and (4), a sensor node s with a high quality of information might be selected for data collection even if it is located a few hops away from a mobile node m, and that might cause path overlapping. Therefore, it is actually also necessary to guarantee that at least one of the shortest paths from s to m passes only sensor nodes that are not used for the data collection by any other mobile node.

Figure 2a shows a simple example of WSN with four sensor nodes s_1 to s_4 and two mobile nodes m_1 and m_2. The communication links are shown with solid and dashed edges, where the dashed edges are the communication links created by the mobile nodes with their immediate neighbours (potential sink nodes). The number shown above a sensor node is the quality of information of the corresponding sensor. We assume the capacity for all mobile nodes is 10 kB/second, the time period T of each round is 10 seconds, and the data size buffered by all sensor nodes is 60 kB. We assume the number of hops from a sensor node s to a mobile node m is the number of edges on the shortest path from s to m (for example, $Hops[s_1, m_1] = 1$, $Hops[s_3, m_1] = 2$, and so on). In

this example, due to the capacity constraints (3), each mobile node can collect data from only one sensor node, and the maximum utility is gained, if data is collected from s_3 and s_4 by either m_1 or m_2. Figure 2b shows a possible solution, with $x[s_4, m_1] = 1$ and $x[s_3, m_2] = 1$ (with a total data utility of $\frac{60 \cdot 90}{3} + \frac{60 \cdot 60}{2} = 3600$) indicated by colouring the sensor nodes to the colour of the selected mobile nodes, and $x[s,m] = 0$ for all other sensor nodes s and mobile nodes m, indicated by keeping the blue colour of the sensor nodes s_1 and s_2. The path from sensor node s_4 to mobile node m_1 (indicated by red edges) overlaps the path from sensor node s_3 to m_2 (indicated by green edges). The overlap occurs at sensor node s_3, and requires s_3 also to relay the data from s_4 to m_1, which increases the communication overhead of s_3, reduces the overall energy efficiency, and increases the latency of the data delivery. Figure 2c shows a solution where path overlapping is avoided by m_2 collecting the data from sensor node s_2 instead of s_3, with a total data utility of $\frac{60 \cdot 90}{3} + \frac{60 \cdot 25}{1} = 3300$. Note that avoiding path overlap decreases the total data utility by $3600 - 3300 = 300$ units compared to the solution in Figure 2b.

To enforce a no-overlap constraint for paths, we solve the problem in two steps. First, we use a clustering algorithm (further explained in Section V-A), and then we schedule the data collection (further explained in Section V-B) within each cluster. The clustering algorithm guarantees that path overlapping does not occur, and the data collection scheduling maximises the total data utility (2), subject to just the capacity constraints (3) and the communication constraints (4).

V. ALGORITHM DESIGN

A. Clustering

Our clustering algorithm `SelectClusters` partitions the sensor nodes reachable from the mobile nodes with a given limit $MaxHopCount$ of hops into disjoint clusters and assigns each cluster to a mobile node. The clustering algorithm aims at balancing the clusters between the mobile nodes, such that each cluster includes as many sensor nodes as possible, while avoiding starvation of the other clusters. Our clustering algorithm `SelectClusters` for each round is listed in Algorithm 1. In our algorithms, the identifier of a variable is written with *italic* font, and the identifier of a function is written with `typewriter` font.

`SelectClusters` takes the hop count matrix $Hops$, the integer $MaxHopCount$, the array $SinkNodes$ of sink nodes, and the array $Neighbours$ of neighbour sensor nodes as an input, and outputs an array called $cluster$, where $cluster[m]$ represents the set of sensor nodes assigned to the mobile node m. `SelectClusters` uses the function `path` (line 14), where $\texttt{path}(s, m)$ takes a sensor node s and a mobile node m as input, and returns **true** if at least one shortest path from s to m only passes through sensor nodes that are already in $cluster[m]$, and otherwise returns **false**.

The `SelectClusters` algorithm creates an array $clusterQueue$, where $clusterQueue[m]$ represents a first-in first-out queue of sensor nodes that can be a member of
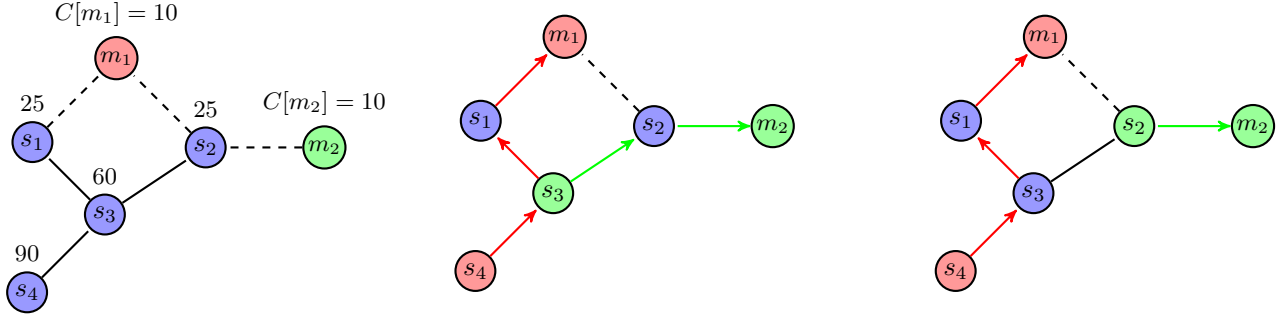
(a) WSN with four sensor nodes and two mobile nodes

(b) Paths overlapping at sensor node s_3

(c) Path overlapping avoided

Fig. 2: Motivating example for the no-overlap constraint for paths.

$cluster[m]$, and initialises each $clusterQueue[m]$ to the (one-hop) neighbours of mobile node m (lines 3–5). Let $sensorIsNotUsed[s]$ be **true** if sensor node s is not yet a member of any $cluster[m]$, and **false** otherwise. Each $sensorIsNotUsed[s]$ is initially set to **true** (lines 7–9). We use a breadth-first strategy to visit all the sensor nodes in $clusterQueue[m]$ (lines 10–12), and the algorithm quits if $clusterQueue[m]$ is empty for all $m \in M$ (line 10). In each iteration, we first select a mobile node m (line 11), and then while $clusterQueue[m]$ is not empty, we dequeue a sensor node s from that queue (line 13). If s is not yet a member of any cluster and at least one shortest path from s to m only passes through sensor nodes already in $cluster[m]$ (line 14), then we add s to $cluster[m]$ (line 15) and mark s as used (line 16). We then iterate over each neighbour s' of s (line 17), and enqueue s' to $clusterQueue[m]$ (line 19), if s' is not used yet in any cluster, the number of hops from s' to m is at most $MaxHopCount$, and s' is not already in $clusterQueue[m]$ (line 18). If a sensor s is added to a cluster, then we require to switch to the next mobile node in order to balance the clusters, which is achieved by breaking out of the inner while loop (line 22).

B. Data Collection

We propose the scheduling algorithm `SelectMaxUtility` for the data collection in each round. It is listed in Algorithm 2. `SelectMaxUtility` aims at maximising the total data utility achieved for the scheduling of the data collection from the sensor nodes.

The `SelectMaxUtility` scheduling algorithm takes the clusters $cluster$ created by the `SelectClusters` algorithm, the array D of sensor node data sizes, the array C of mobile node capacities, the set M of mobile nodes, and the set S of sensor nodes as an input, and outputs an array $schedule$, where $schedule[m]$ is a first-in first-out queue of sensor nodes scheduled for data collection by mobile node m. Each mobile

---

Algorithm 1 The `SelectClusters` algorithm

Input: $M, S, Hops, MaxHopCount, SinkNodes, Neighbours$
Output: $cluster$: disjoint sensor clusters for each mobile node
1: **for all** m **in** M
2:    $cluster[m] \leftarrow \emptyset$
3:    **for all** s **in** $SinkNodes[m]$
4:       **enqueue**($clusterQueue[m]$, s)
5:    **end for**
6: **end for**
7: **for all** s **in** S
8:    $sensorIsNotUsed[s] \leftarrow$ **true**
9: **end for**
10: **while** $clusterQueue[m] \neq \emptyset$ for some m
11:    **for all** m **in** M
12:       **while** $clusterQueue[m] \neq \emptyset$
13:          $s \leftarrow$ **dequeue**($clusterQueue[m]$)
14:          **if** $sensorIsNotUsed[s]$ **and** $path(s, m)$ **then**
15:             $cluster[m] \leftarrow cluster[m] \cup \{s\}$
16:             $sensorIsNotUsed[s] \leftarrow$ **false**
17:             **for all** s' **in** $Neighbours[s]$
18:                **if** $sensorIsNotUsed[s']$ **and**
                     $Hops[s', m] \leq MaxHopCount$ **and**
                     s' **not in** $clusterQueue[m]$ **then**
19:                   **enqueue**($clusterQueue[m], s'$)
20:                **end if**
21:             **end for**
22:             **break**
23:          **end if**
24:       **end while**
25:    **end for**
26: **end while**
27: **return** $cluster$

---

node m starts the data collection process, as it receives the scheduling queue $schedule[m]$ from the base station. The data collection is requested from the sensor nodes in $schedule[m]$ in the order given there.

The `SelectMaxUtility` scheduling algorithm iterates over all the mobile nodes (line 7), and for each mobile node selects sensor nodes i from the set $cluster[m]$, such that

Algorithm 2 The `SelectMaxUtility` data collection algorithm

    Input: $cluster, D, C, M, S$
    Output: $schedule$: queue of sensor nodes for each mobile node
1: **for all** m **in** M
2:     $schedule[m] \leftarrow \emptyset$
3:     **for all** s **in** S
4:        $x[s, m] \leftarrow$ **false**
5:     **end for**
6: **end for**
7: **for all** m **in** M
8:     $totalDataSize \leftarrow 0$
9:     $sensorSelected \leftarrow$ **true**
10:    **while** $sensorSelected$
11:      $sensorSelected \leftarrow$ **false**
12:      $S' \leftarrow \{i \in cluster[m] \mid \neg x[i, m] \ \wedge$
13:          $u[i, m] = \max u[*, m]\}$
14:      **if** S' is not empty **then**
15:        **for all** s' **in** S'
16:          $totalDataSize \leftarrow totalDataSize + D[s']$
17:          $sensorSelected \leftarrow$ **true**
18:          **if** $totalDataSize \leq C[m] \cdot T$ **then**
19:            $x[s', m] \leftarrow$ **true**
20:            **enqueue**$(schedule[m], s')$
21:          **else**
22:            $sensorSelected \leftarrow$ **false**
23:          **end if**
24:        **end for**
25:      **else**
26:        $sensorSelected \leftarrow$ **false**
27:      **end if**
28:    **end while**
29: **end for**
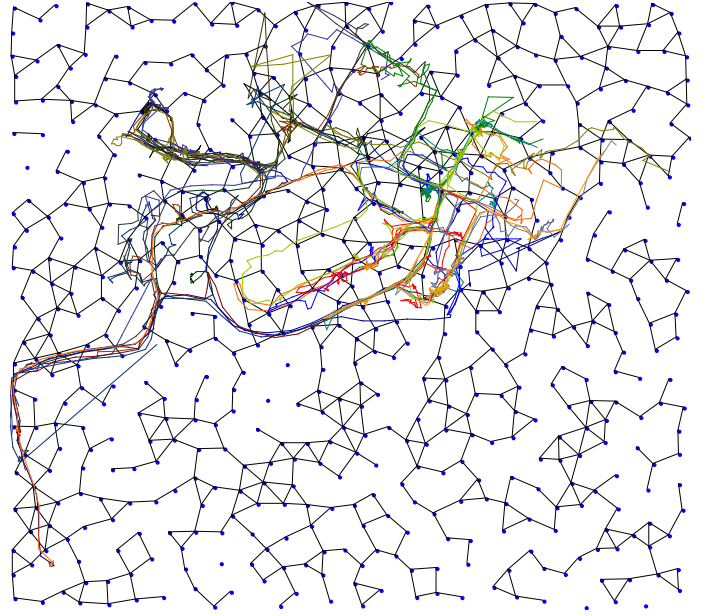30: **return** $schedule$

Fig. 3: Our random deployment of 600 sensor nodes in the sensing field. The sensor nodes are shown with dots, and the edges represent the communication links between the sensor nodes. The location traces of the mobile nodes for 180 rounds of the simulation are shown with coloured lines.

VI. EXPERIMENTS

A. Experiment Setup

We experiment with mobile traces collected by participating visitors to a state fair (North Carolina, USA) [18]. The data set includes GPS traces of 19 mobile phones every 30 seconds over a period of at least 90 minutes and at most 208 minutes, and covering a sensing field of size 1.2 km by 1 km. We use the first 90 minutes of the GPS traces in our simulation, as all the mobile nodes are active in the sensing field during this period. The simulation is over 180 rounds, as the length of each round is set to 30 seconds. We randomly place 600 sensor nodes in the sensing field with random quality of information. The placement requires that no two sensor nodes can be placed closer than 25 meters. The transmission range of the sensor nodes and the mobile nodes is set to 50 meters, indicating that a neighbour sensor node is detected within 50 meters [19], [20]. In each round, the size of the data stored in the buffer of each sensor node is 60 kBytes, and the capacity of each mobile node is 300 kBytes. We assume the base station is a server connected to the Internet, and communicates with the mobile nodes through 3G.

Figure 3 shows the sensing field with 600 randomly deployed sensors and the communication links as the edges. The location trace of each mobile node is shown by a coloured line during the 180 rounds of the simulation.

each sensor node i is not yet assigned for data collection ($x[i, m] =$ **false**) and achieves the maximum data utility $u[i, m]$ among all sensors for m (line 12). Sensor s' is assigned for data collection only if the capacity constraint (3) is satisfied. Let $totalDataSize$ be the total size of the data in bytes to be collected from the sensors by the current mobile node. In each iteration of the while loop (line 10), we select a set S' of sensor nodes (line 12), and if S' is not empty, then we iterate on all sensor nodes $s' \in S'$ (line 15), the total data size $totalDataSize$ is incremented by the data size $D[s']$ (line 16), and if the capacity constraint (line 18) is satisfied, then the sensor node s' is assigned for data collection by mobile node m (line 19), and it is enqueued in $schedule[m]$ (line 20). We exit the while loop (line 10) under two conditions. First, if the capacity constraint for mobile node m (line 18) is not satisfied, then the boolean variable $sensorSelected$ is set to **false** (line 22), and the next mobile node is selected as the condition of the while loop becomes **false** (line 10). Second, if the set S' of sensor nodes is empty (due to $x[s, m] =$ **true** for all sensors in $cluster[m]$, or the cluster $cluster[m]$ is empty), then $sensorSelected$ is also set to **false** (line 26) and causes the while loop to exit.

B. Results and Analysis

We evaluate the performance of our data collection protocol QoIACP in terms of data utility, quality of information, and communication overhead. We compare the performance of our QoI-aware data collection algorithm with our extended implementation of the CTP protocol [6]. Our implementation of CTP extends it by constructing the data collection tree periodically when the mobile nodes are moving in the sensing field. The mobile nodes keep collecting sensed data in their data collection trees with minimum communication cost.

In order to measure further the quality of our results, we construct an integer programming (IP) model from the problem formulation in Section IV by just keeping the objective function (2), the capacity constraints (3), and the communication constraints (4), while allowing path overlapping to occur. The IP model is solved to optimality with the solver *Gurobi Optimizer* [21], and the resulting metrics, namely total data utility, total quality of information, and communication overhead are used in our evaluation as an upper bound for the corresponding metrics computed for the QoIACP protocol, because the latter also enforces the no-overlap constraints. The optimal results of the IP model do not, in general, provide a tight upper bound on the data utility due to the problem relaxation, hence the optimal solution to the data collection problem including the no-overlap constraints might be far below this upper bound.

Our simulation of QoIACP and the data collection algorithms is implemented in C++, and the IP model is run under *Gurobi Optimizer* (revision 5.1) and Mac OS X 10.8.2 64 bit on an Intel Core 2 Duo 2.53 GHz with 3MB L2 cache and 8GB RAM. The runtimes of the IP model and our data collection algorithms are on average below 50 milliseconds for any number of mobile nodes (varying from 1 to 19) in our instance data.

Figure 4 compares the total data utility over 180 rounds achieved by the QoIACP protocol versus the CTP protocol and the IP model, upon varying the number of mobile nodes. The results show that QoIACP achieves a high total data utility, with at most a 4% gap from the upper bound computed by the IP model, compared to the 15% gap between CTP and the IP model for the largest instance (19 mobile nodes).

Figure 5 compares the total quality of information gained over 180 rounds by the QoIACP protocol versus the CTP protocol and the IP model, upon varying the number of mobile nodes. The total quality of information is the sum of the normalised information value from the collected sensing data. It is computed by the numerator in the data utility function in Eq. (1):

$$\sum_{s \in S} \sum_{m \in M} D[s] \cdot Q[s] \cdot x[s, m] \qquad (5)$$

As expected, the results show a large gap between the CTP protocol and our QoIACP protocol.

Figure 6 compares the total data utility achieved by the QoIACP protocol versus the CTP protocol over 180 rounds. The results show that QoIACP achieves a higher data utility
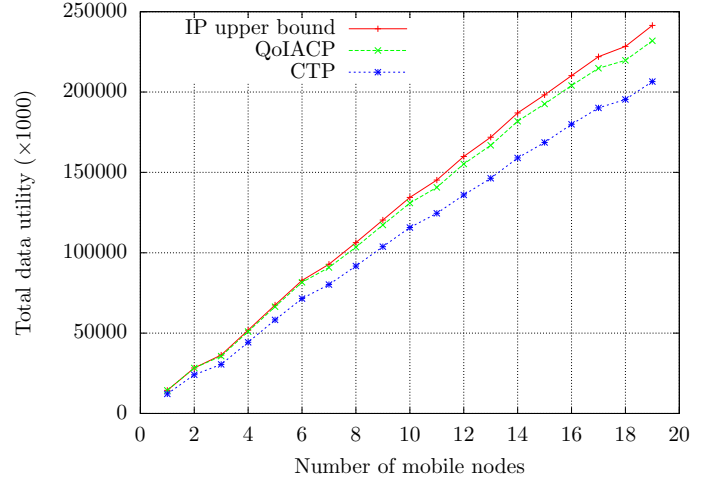

Fig. 4: Total data utility over 180 rounds (90 minutes), upon varying the number of mobile nodes, for our QoIACP protocol, our variant of the CTP protocol [6], and the upper bound from our IP model.
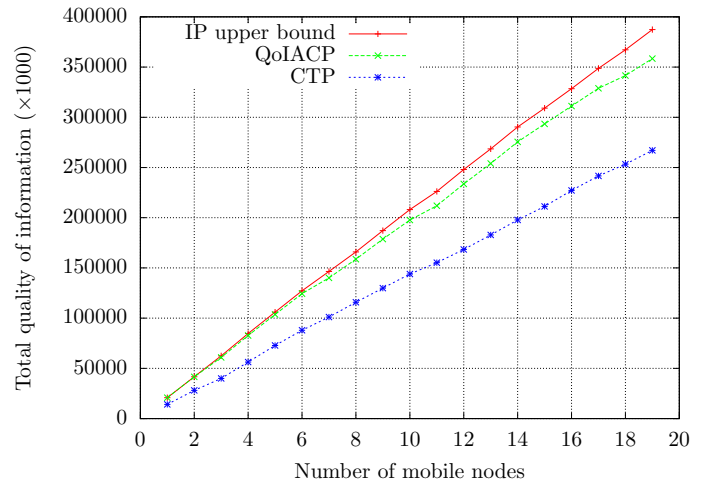

Fig. 5: Total quality of information gained over 180 rounds (90 minutes), upon varying the number of mobile nodes, for our QoIACP protocol, our variant of the CTP protocol [6], and the upper bound from our IP model.

than the CTP protocol in 80% of the rounds, and the utilities are equal in the other 20% of the rounds.

Figure 7 compares the total communication overhead over 180 rounds by the QoIACP protocol versus the CTP protocol, upon varying the number of mobile nodes. The total communication overhead of the sensors is measured by the number of transmissions for the sensors to report their sensing data to the mobile nodes. The CTP protocol is tailored to minimise the communication overhead, and as a result the data collection from the sensor nodes with the CTP protocol achieves the least amount of communication overhead on the sensor nodes.
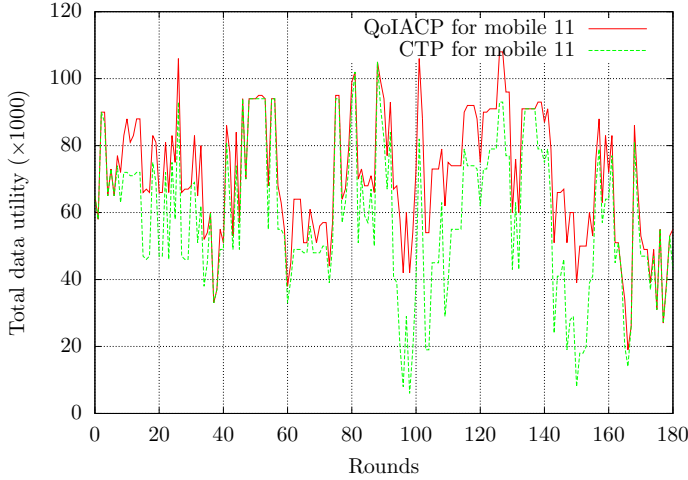
Fig. 6: Data utility achieved by our QoIACP protocol and our variant of the CTP protocol [6] for the mobile node 11 in a simulation with 19 mobile nodes, over 180 rounds (90 minutes).
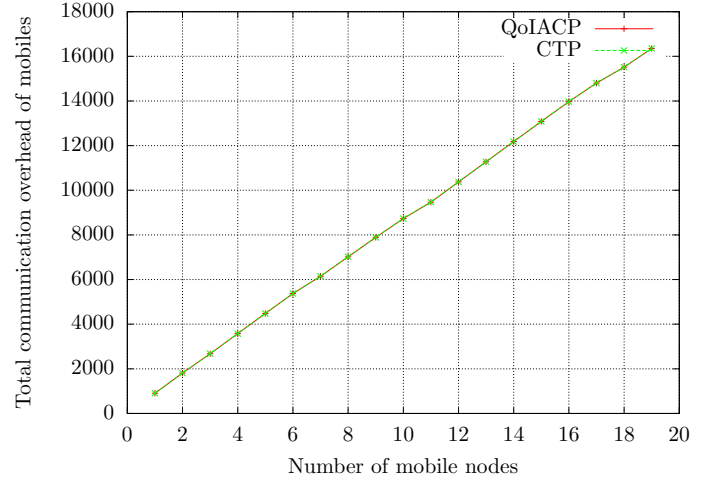

Fig. 8: Total communication overhead of the mobile nodes over 180 rounds (90 minutes), upon varying the number of mobile nodes, for our QoIACP protocol and our variant of the CTP protocol [6].
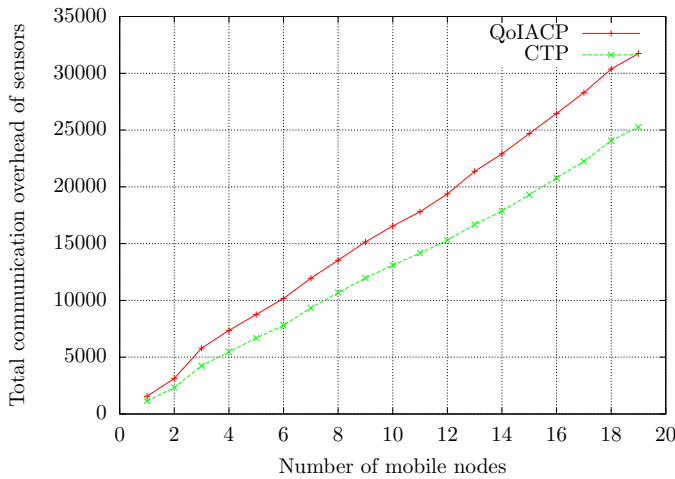

Fig. 7: Total communication overhead of the sensor nodes over 180 rounds (90 minutes), upon varying the number of mobile nodes, for our QoIACP protocol and our variant of the CTP protocol [6].
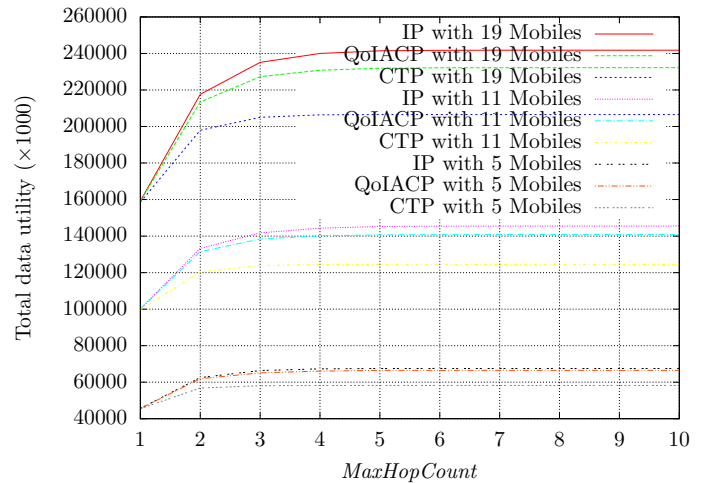

Fig. 9: Total data utility over 180 rounds (90 minutes) in a simulation with 5, 11, and 19 mobile nodes, upon varying $MaxHopCount$ between 1 and 10.

The sensor nodes with the QoIACP protocol are required to spend more energy for the communication in order to achieve a higher data utility.

Figure 8 compares the total communication overhead over 180 rounds by the QoIACP protocol versus CTP, upon varying the number of mobile nodes. The total communication overhead of the mobile nodes is computed as the number of transmissions required for the data collection between the mobile nodes and their sink nodes. The results show that both QoIACP and CTP achieve the same total mobile communication overhead, mainly due to the capacity of the mobile nodes being fully utilised for the data collection in each

round by both the QoIACP protocol and the CTP protocol.

Figures 9 and 10 show the trends of total data utility and communication overhead for instances with 5, 11, and 19 mobile nodes as $MaxHopCount$ is increasing from 1 to 10. The results show that, regardless of the choice of protocol, after 5 hops there is nearly no gain for the data collection by increasing the maximum number of hops. Therefore, in our QoIACP protocol, it is best to choose $MaxHopCount = 5$ for all mobile nodes.

VII. Conclusion

We have presented a novel quality-of-information (QoI) aware data collection protocol (QoIACP) for wireless sensor
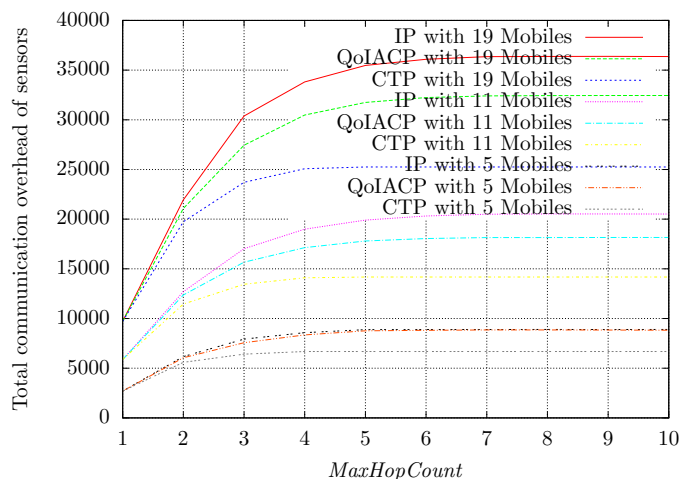
Fig. 10: Total communication overhead of the sensor nodes over 180 rounds (90 minutes) in a simulation with 5, 11, and 19 mobile nodes, upon varying $MaxHopCount$ between 1 and 10.

networks with mobile users. The protocol is designed to optimise data utility, which measures the normalised QoI value of collected data per transmission. A hybrid methodology is used in our QoIACP protocol with a distributed neighbourhood discovery protocol, but centralised clustering and data collection scheduling for coordination among multiple mobile users. We have formulated the optimisation problem and proposed a clustering algorithm and a scheduling algorithm for the data collection in order to maximise the total data utility. A relaxation of the problem constraints has been used to model the problem with integer programming (IP); this model is then solved to optimality to provide a usually non-sharp upper bound on the quality of our results. We have evaluated our QoIACP protocol by extensive simulations using real mobility traces. We have also compared our QoIACP protocol with an extended implementation of the CTP protocol [6].

The simulation results have demonstrated that our QoIACP protocol can achieve very high data utility, as close as 4% to the upper bound on an optimal solution. It obtains much higher QoI (more than 25%) and data utility (more than 11%) than the CTP protocol. The communication overhead of mobile users remains the same between QoIACP and CTP, while the sensors in QoIACP may have slightly higher communication overhead for reporting sensing data with high QoI.

As future work, we plan to extend QoIACP to a fully distributed implementation for real deployment. Apart from maximising data utility, we would also like to extend the work for load balancing for multiple mobile users.

ACKNOWLEDGEMENTS

REFERENCES

[1] M. Fazio, M. Paone, A. Puliafito, and M. Villari, "Heterogeneous sensors become homogeneous things in smart cities," in *Proceedings of the Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, ser. IMIS. IEEE Computer Society, 2012, pp. 775–780.

[2] L. Filipponi, A. Vitaletti, G. Landi, V. Memeo, G. Laura, and P. Pucci, "Smart city: An event driven architecture for monitoring public spaces with heterogeneous sensors," in *Proceedings of the Fourth International Conference on Sensor Technologies and Applications*, ser. SENSORCOMM. IEEE Computer Society, 2010, pp. 281–286.

[3] C. Chong and S. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247–1256, 2003.

[4] Z. M. Charbiwala, S. Zahedi, Y. Kim, Y. H. Cho, and M. B. Srivastava, "Toward quality of information aware rate control for sensor networks," in *International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks*, 2009.

[5] C. Bisdikian, "On sensor sampling and quality of information: A starting point," in *Proceedings of the Fifth Annual IEEE International Conference on Pervasive Computing and Communications*, ser. PerCom Workshops, 2007, pp. 279–284.

[6] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the Seventh ACM Conference on Embedded Networked Sensor Systems*. ACM Press, 2009, pp. 1–14.

[7] R. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a three-tier architecture for sparse sensor networks," in *Proceedings of the First International IEEE Workshop on Sensor Network Protocols and Applications*, ser. SNPA. IEEE Communications Society Press, 2003, pp. 30–41.

[8] M. Gatzianas and L. Georgiadis, "A distributed algorithm for maximum lifetime routing in sensor networks with mobile sink," *IEEE Transactions on Wireless Communications*, vol. 7, no. 3, pp. 984–994, 2008.

[9] Y. Gu, D. Bozdag, E. Ekici, F. Ozguner, and C.-G. Lee, "Partitioning-based mobile element scheduling in wireless sensor networks," in *Proceedings of the Second Annual IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks*, ser. SECON. IEEE Communications Society Press, 2005, pp. 386–395.

[10] N. Bisnik, A. Abouzeid, and V. Isler, "Stochastic event capture using mobile sensors subject to a quality metric," in *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom. ACM Press, 2006, pp. 98–109.

[11] B. Kusy, H. Lee, M. Wicke, N. Milosavljevic, and L. Guibas, "Predictive QoS routing to mobile sinks in wireless sensor networks," in *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*, ser. IPSN. IEEE Computer Society, 2009, pp. 109–120.

[12] H. Lee, M. Wicke, B. Kusy, O. Gnawali, and L. Guibas, "Data stashing: energy-efficient information delivery to mobile sinks through trajectory prediction," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*. ACM, 2010, pp. 291–302.

[13] Z. Li, Y. Liu, M. Li, J. Wang, and Z. Cao, "Exploiting ubiquitous data collection for mobile users in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, 2012.

[14] H.-X. Tan, M.-C. Chan, W. Xiao, P.-Y. Kong, and C.-K. Tham, "Information quality aware routing in event-driven sensor networks," in *Proceedings of the 29th conference on Information communications*, ser. INFOCOM. IEEE Press, 2010, pp. 1703–1711.

[15] R. Urgaonkar, E. Ciftcioglu, A. Yener, and M. Neely, "Quality of information aware scheduling in task processing networks," in *Proceedings of the International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, ser. WiOpt, 2011, pp. 401–406.

[16] D. Wang, H. Ahmadi, T. Abdelzaher, H. Chenji, R. Stoleru, and C. Aggarwal, "Optimizing quality-of-information in cost-sensitive sensor data fusion," in *International Conference on Distributed Computing in Sensor Systems and Workshops*, ser. DCOSS, 2011, pp. 1–8.

[17] S. Banerjee and S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks," in *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies*, ser. INFOCOM, vol. 2, 2001, pp. 1028–1037.

[18] I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong, "CRAWDAD Daily GPS track log collected from State Fair, North Carolina, USA." Available from http://crawdad.cs.dartmouth.edu/ncsu/mobilitymodels/GPS/NC_State_Fair, Jul. 2009.

[19] S. Gao, H. Zhang, and S. Das, "Efficient data collection in wireless sensor networks with path-constrained mobile sinks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 4, pp. 592–608, 2011.

[20] S. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, "Energy efficient schemes for wireless sensor networks with multiple mobile base stations," in *IEEE Conference on Global Telecommunications*, ser. GLOBECOM '03, vol. 1, 2003, pp. 377–381.

[21] Gurobi Optimization, Inc., "Gurobi optimizer reference manual," 2012, available from http://www.gurobi.com.