Research Article

Skyline-Based Aggregator Node Selection in Wireless Sensor Networks

Aziz Nasridinov, Sun-Young Ihm, and Young-Ho Park

Department of Multimedia Science, Sookmyung Women’s University, Seoul 140-742, Republic of Korea

Correspondence should be addressed to Young-Ho Park; yhpark@sm.ac.kr

Received 22 May 2013; Accepted 18 July 2013

Academic Editor: Tai-hoon Kim

Copyright © 2013 Aziz Nasridinov et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In order to achieve the equal usage of limited resources in the wireless sensor networks (WSNs), we must aggregate the sensor data before passing it to the base station. In WSNs, the aggregator nodes perform a data aggregation process. Careful selection of the aggregator nodes in the data aggregation process results in reducing large amounts of communication traffic in the WSNs. However, network conditions change frequently due to sharing of resources, computation load, and congestion on network nodes and links, which makes the selection of the aggregator nodes difficult. In this paper, we study an aggregator node selection method in the WSNs. We formulate the selection process as a top-k query problem, where we efficiently solve the problem by using a modified Sort-Filter-Skyline (SFS) algorithm. The main idea of our approach is to immediately perform a skyline query on the sensor nodes in the WSNs, which enables to extract a set of sensor nodes that are potential candidates to become an aggregator node. The experiments show that our method is several times faster compared to the existing approaches.

1. Introduction

Recently, wireless sensor networks (WSNs) have been used in many applications, such as military target tracking and surveillance [1], meteorological hazards [2], wildlife monitoring [3], natural disaster relief [4], and healthcare [5]. A WSN consists of a sink node, also called a base station, and a group of sensor nodes. Each sensor node has a wireless radio transceiver, a power source, a small microcontroller, and multitype sensors that enable the sensor node to sense and exchange the data with other sensor nodes. On the other hand, the base station is a gateway for the WSN to communicate with the external applications. The base station collects the sensor data from the sensor nodes and combines it into a form requested by the applications.

In a typical WSN, the sensor nodes have limited resources such as battery power, computing capability, and memory. Communication is a dominant source of energy consumption in the WSNs [6, 7]. Thus, the general approach is to jointly process the sensor data, generated by the different sensor nodes while transmitting it to the base station. This process is called as a data aggregation process. By processing, combining, and filtering the sensor data, the data aggregation process reduces a number of data transmissions and improves the bandwidth energy utilization in the WSNs.

In WSNs, the aggregator nodes perform the data aggregation process. The aggregator nodes receive the sensor data from neighboring nodes, perform the data aggregation process, and forward the filtered data to the base station. Careful selection of the aggregator nodes in the data aggregation process results in reducing large amounts of communication traffic in the WSN. However, network conditions change frequently due to sharing of resources, computation load, and congestion on network nodes and links, which makes the selection of the aggregator nodes difficult [8, 9].

Several data aggregation protocols have been proposed to solve the selection problem of aggregator nodes, which can be categorized into two types: tree-based data aggregation protocols [10–13] and cluster-based data aggregation protocols [14–18]. In tree-based data aggregation protocols, the aggregator node is determined, and the data paths of sensor nodes include the determined data aggregator nodes. The main issue of tree-based data aggregation protocols is the construction of an energy efficient data aggregation tree, which is time consuming in the large WSN. On the other hand, in cluster-based data aggregation protocols, sensor
2. Data Aggregation in WSN

In this section, we briefly explain the data aggregation process in the WSNs.

A WSN is a collection of sensor nodes with limited battery power, computing capability, and memory. Since the communication is a main source of energy consumption in WSN, it is preferable to jointly process the sensor data, generated by the different sensor nodes while forwarding it to the base station. This process is called a data aggregation process. One of the advantages of data aggregation process is that when the base station initiates the query on the WSN, rather than sending each sensor node’s data to the base station, one of the sensor nodes performs the data aggregator process. Thus, the data aggregation process reduces redundant data transmissions and improves the overall lifetime of the WSN.

Figure 1 demonstrates a data aggregation process. In a typical data aggregation process, three types of nodes are used, such as base station, sensor nodes, and aggregator nodes. Sensor nodes sense the data from the target region and send it to the aggregator nodes. The aggregator nodes collect the sensor data from the multiple sensor nodes, perform the data aggregation process using aggregation function, and send the aggregated data to the upper aggregator node or to the base station. The base station collects the aggregated data from the WSN and combines it into a form requested by the applications.

The simplest way to perform the data aggregation process is to determine data aggregator nodes in the network. Recall from Section 1 that network conditions change frequently due to sharing of resources, computation load, and congestion on network nodes and links, which makes the selection of the aggregator nodes difficult. In this paper, we study an aggregator node selection method in the WSNs.

3. Related Study

Several data aggregation protocols have been proposed to solve the selection problem of aggregator nodes, which can be categorized into two types: tree-based data aggregation protocols [10–13] and cluster-based data aggregation protocols [14–18]. In this section, we only describe representative data aggregation protocols.

3.1. Tree-Based Data Aggregation Protocols. In tree-based data aggregation protocols, the aggregator node is determined, and the data is transformed to the base station through the determined data aggregator nodes.

Madden et al. [10] proposed the tiny aggregation (TAG), service for aggregation in a low-power, distributed, and wireless environments. TAG has two attributes. First, TAG enables users to express simple, declarative queries for the data collection and aggregation, by borrowing an idea from the aggregation operators in database query language. Second, it semantically distributes and executes aggregation queries in the sensor network in a timely and power-efficient manner and preserves important properties of the WSNs, such as the resource constraints and loss communication.

Lindsey et al. [11] proposed power-efficient gathering in sensor information systems (PEGASIS), which reduces energy cost to increase the lifetime of the WSNs. The method insists that it is near optimal in terms of energy cost for the data gathering application in the WSNs. The main idea of the PEGASIS is to form a chain among the sensor nodes. In order to evenly distribute the energy usage in the WSNs, each sensor node communicates only with a close neighboring node and takes turns forwarding the data to the base station.
Ding et al. [12] proposed an efficient energy-aware distributed heuristic to generate the aggregation tree, called energy-aware distributed aggregation tree (EADAT). The EADAT algorithm makes no assumption of a local network topology and is based on residual power. It uses two techniques, such as neighboring broadcast scheduling and distributed competition among neighbors, which makes the EADAT algorithm efficient. The simulation analysis demonstrates that EADAT algorithm performs well in terms of network lifetime, energy saving, data delivery ratio, and the protocol overhead.

3.2. Cluster-Based Data Aggregation Protocols. In cluster-based data aggregation protocols, sensor nodes are divided into clusters. In each cluster, a cluster head is selected. Cluster head aggregates the sensor data locally and forwards the aggregation result to the base station.

Heinzelman et al. [16] proposed a low-energy adaptive clustering hierarchy, called low-energy adaptive clustering hierarchy (LEACH), protocol architecture for microsensor networks. The LEACH is divided into two phases, such as set-up phase and steady-state phase. In set-up phase, cluster structures are organized. Then, in the steady-state phase, the data is forwarded from the nodes to the cluster head and to the base station. LEACH uses a randomized rotation of the cluster head in order to evenly distribute energy usage among the sensor nodes. The experiment results demonstrate that LEACH reaches the performance needed under the tight constraints of the wireless channel.

Younis and Fahmy [17] proposed protocol, called hybrid energy-efficient distributed clustering (HEED). HEED selects a cluster head according to both residual energy and node proximity to its neighbors. HEED defines the mean of the minimum power levels, called average minimum reachability power (AMRP), required by all nodes within the cluster ranges to reach the cluster head. AMRP provides a good estimation of the communication cost in each cluster. In order to select a cluster head, each sensor node sets its probability of becoming a cluster head that considers the initial percentage of cluster heads, the current residual, and the initial energy of the sensor node. This process continues until each node selects its cluster head. The simulation results show that HEED can prolong the network lifetime and support scalable data aggregation.

Buttyan and Schaffer [18] proposed a position-based aggregator node election in wireless sensor networks (PANEL), which uses the geographical position information of the sensor nodes in order to select an aggregator node. In PANEL, at the beginning of each phase, a metric called a reference point is calculated in each cluster by each node. Once the reference point is calculated, the nodes in the cluster select the sensor node that is the closest to the referent point as the aggregator node for the given phase. In each phase, the reference points are recalculated, and the aggregator node selection procedure is reperformed, in order to ensure load balancing meaning that each sensor node can have the equal probability to become an aggregator node.

4. Proposed Method

This section will describe the proposed method in detail. We start with general assumptions for our approach. Then, we explain the aggregator node selection process using skyline. Finally, we describe a tree building process of the cluster heads.

4.1. Skyline Sensor Nodes. Considering that a set of sensor nodes are scattered in a field, in this paper, we make following assumptions: we assume that each sensor node can perform following functions: sensing, aggregation, and forwarding. Sensor nodes are static and they are aware of their geographical position, that is, not instrumented with GPS-capable antennae. We further assume that a sensor network is subdivided into the clusters. In each cluster, we select a cluster head, in other words an aggregator node. An aggregator node performs the data aggregation process locally, and forwards the aggregated sensor data to the base station. Figure 2 demonstrates an example of a cluster-based data aggregation process.

It is important to note that an aggregator node should be selected according to multiple attributes, such as distance from the base station, power consumption, battery life, and communication cost. When a number of sensor nodes are large in a cluster, it may take a long time to compute the combination of these attributes and select an optimal aggregator node. However, we can perform a look up at just the top few results, ranked by a small set of attributes values that define an aggregator node. Thus, we propose to formulate the selection process as a top-k query problem, where we efficiently solve the problem by using a modified SFS algorithm. The main idea of our approach is to immediately perform a skyline query on the sensor nodes in the WSN, which enables to extract a set of sensor nodes that are potential candidates to become an aggregator node. First, we briefly introduce skyline queries, and then, we describe how to apply them in our approach.

Given a set of sensor nodes with \( n \) attributes, a skyline query choses those sensor nodes that are not dominated by
Definition 1 (dominance relationship). Consider a cluster C and the sensor nodes \( s_1 \) and \( s_2 \) that belong to the cluster \((s_1, s_2) \in C\). We say that \( s_1 \) dominates \((s_1 > s_2) s_2 \) if and only if \( s_1 \) is better than or equal to \( s_2 \) by all attributes and definitely better in at least one attribute.

Definition 2 (skyline sensor node). The skyline of \( C \), denoted as \( SA(C) \), is a set of sensor nodes in \( C \) that are not dominated by any other sensor node of \( C \), that is, \( SA(C) = \{ s_1 \in C | \forall s_2 \in C : s_2 < s_1 \} \).

Figure 3 shows an example of skyline sensor nodes of a certain cluster. The sensor nodes are represented as points in the 2-dimensional space, with the coordinates of each point indicating the values of the sensor nodes in two attributes, such as power consumption and communication cost. From the figure, we can observe that \( s_1, s_2, s_4 \), and \( s_5 \) are not dominated by other sensor nodes, meaning that there are no other sensor nodes that offer both shorter power consumption and communication cost than these nodes.

4.2. Aggregator Node Selection Algorithm. We assume that the proposed approach considers aggregator node selection in a medium-scale sensor network. In a typical medium-scale sensor network, the number of nodes can reach 200–300 sensor nodes, where the potential candidate to be an aggregator node does not exceed 30–40 sensor nodes, according to the size of each cluster. However, it is important to mention that in a large-scale sensor network, potential candidates to be an aggregator node can be huge, which means that there is a need to build an index before applying our approach.

Determining the aggregator nodes of a certain cluster requires pairwise comparisons of the attributes of the candidate aggregator nodes. This process can be expensive in terms of computation time if the number of candidate services is large. Several efficient algorithms have been proposed for skyline computation. Given that, for the problem considered here, the process of determining the skyline aggregator nodes is independent of any individual base station request or usage context, it does not need to be conducted online at request time. Therefore, we make use of any of the existing methods for determining the skyline aggregator nodes offline in order to speed up the service aggregator selection process later at request time. For this purpose, we used SFS algorithm, which preorders the data points in skyline according to their scores obtained by a monotone functions \( f \), such that if \( f(s_i) < f(s_j) \), then it is guaranteed that \( s_i < s_j \). In other words, the function corresponds to a topological sort with respect to the dominance criteria. The SFS method is generally approached as a baseline in benchmarking research; hence, it is suitable for the formulating solution to the skyline query.

SFS is a skyline algorithm based on presorting and uses no index structures. Algorithm 1 [20] describes the steps of extended SFS. It takes an array \( C[1, \ldots, n] \) of tuples as input which is assumed to fit in the main memory. It returns the skyline set \( S[C[1, \ldots, n]] \) as output. SFS maintains an array \( S \) holding skyline tuples and visits all tuples in sorted order (line 3). For each tuple \( D[i] \), SFS performs dominance tests with all skyline tuples in \( S \) (line 4). If no skyline tuples in \( S \) dominate \( D[i] \), SFS inserts \( D[i] \) into \( S \).

4.3. Aggregator Node Traversal Algorithm. In each cluster, we select a cluster head in order to aggregate the sensor data locally and transmit the aggregation result to the base station. However, this approach is also inefficient, because there can be many cluster heads in the large WSNs, which leads to the selection problem among cluster heads. Thus, we propose an aggregator node traversal algorithm, in which we form a tree structure to transmit aggregated data by multihopping through other cluster heads. Algorithm 2 presents an aggregator node traversal algorithm.

In Algorithm 2, given an ordered tree \( T \) with root \( r \), we aim at receiving a list of aggregator nodes to traverse through the tree. The algorithm starts with a BuildTree function (line 1) that builds a tree of selected aggregator nodes. In this procedure, the aggregator node that has the best combination of attributes, calculated in Algorithm 1, is selected as a root node (lines 2 and 3). Then, by checking each child nodes, the algorithm recurs down the left or right subtree and builds a tree of aggregator nodes (lines 3–16). Once BuildTree function builds a tree of aggregator nodes, the algorithm calls aggregator node traversal function, which traverses a tree in a postorder manner. A postorder traversal involves first postorder traversing the subtrees rooted at each of the children of a node and then visiting the node itself, starting at the root.
**Algorithm 1:** The extended Sort-Filtering-Skyline algorithm.

**Input:**
- Input an array $C[1,\ldots,n]$ of data points

**Output:**
- Output the skyline set $S(C[1,\ldots,n])$
  1. $S \leftarrow \emptyset$
  2. sort $C[1,\ldots,n]$ topologically with respect to the dominance criteria (Definition 1)
  3. for $i = 1$ to $n$
do

  (4) if $\forall e \in S$, $e \neq C[i]$ then

  (5) insert $C[i]$ into $S$

end for

(7) return $S$

**Definition 3 (postorder traversal).** Let $T$ be an ordered rooted tree with root $r$. If $T$ consists only of root $r$, then root $r$ is the postorder traversal of $T$. Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at $r$ from left to right in $T$. The postorder traversal begins by visiting $T_1$, then $T_2$ in postorder until $T_n$, and ends by visiting $r$.

It is important to note that the discussed Algorithms 1 and 2 are mutually related to each other. Algorithm 1 selects a set of aggregator nodes according to their attributes, such as distance from the base station, power consumption, battery life, and communication cost. However, there can be many cluster heads in the large WSNs, which leads to selection problem among cluster heads. Algorithm 2 solves this problem by forming a tree structure to transmit aggregated data by multihopping through other cluster heads which results in significant energy savings.

**5. Performance Evaluation**

In this section, we present performance evaluation of our approach. The aim of the experiment is to compare the computation time of the proposed approach with the method when the data aggregation process is not used and with the method of clustering.
5.1. Experiment Results. Experiments were carried out on a 2.4 GHz Pentium processor with 512MB of RAM running Windows XP Professional. For implementation of our proposed approach, we used C++ programming language. Data size used in our experiments consists of 1K, 10K and 100K, data. The following experiments are carried out.

We compare node selection time. Graphs in Figures 4(a), 4(b) and 4(c) demonstrate this comparison. In all of these figures, x-axis represents aggregation node selection time in milliseconds and y represents d dimensions in universe. d dimensions can be interpreted as the sensor node attributes such as distance from the base station, power consumption, battery life, and communication cost. Data size used in our experiments consists of 1K (a), 10K (b), and 100K (c) data.

From the graphs in Figures 4(a), 4(b) and 4(c), we can observe that the proposed approach outperforms the method when the data aggregation process is not used and with the method of clustering by up to two times. This is because the main idea in our approach is to perform a skyline query on the sensor nodes in WSNs in order to extract among those sensor nodes that are potential candidates for the leading role and those that cannot possibly become an aggregator node. Our approach selects a set of leading aggregator nodes according to their attributes, such as distance from the base station, power consumption, battery life, and communication cost. Thus, we can select aggregator nodes more efficiently. On the other hand, the method of clustering suffers from load balancing as it shows the next best result. The method when data aggregation process is not used uses a random aggregator selection algorithm. Thus, it shows the worst performance in selection aggregate node.
6. Conclusion

In this paper, we have studied an aggregator node selection method in the WSNs. We have proposed to formulate the selection process as a top-k query problem, where we efficiently solve the problem by using a modified SFS algorithm. Our approach selects a set of aggregator nodes according to their attributes, such as distance from the base station, power consumption, battery life, and communication cost. Thus, we can reduce large amounts of communication traffic by sending only aggregated data through selected aggregator nodes, instead of individual sensor data, to the base station. The experiments have showed that our method is several times faster comparing to the existing approaches. We have also provided an in-depth analysis on the major factors that impact the performance of previous approaches.

Acknowledgments

This work was supported by the SRC Research Center for Womens Diseases of Sookmyung Womens University (2009).

References

[1] I. Bekmezci and F. Alagoz, “Energy efficient, delay sensitive, fault tolerant wireless sensor network for military monitoring,” International Journal of Distributed Sensor Networks, vol. 5, no. 6, pp. 729–747, 2009.

[2] G. Manes, G. Collodi, R. Fusco, L. Gelpi, and A. Manes, “Continuous remote monitoring in hazardous sites using sensor technologies,” International Journal of Distributed Sensor Networks, vol. 2012, Article ID 317020, 13 pages, 2012.

[3] A. J. Garcia-Sanchez, E. Garcia-Sanchez, E. Losilla et al., “Wireless sensor network deployment for monitoring wildlife passages,” Sensors, vol. 10, no. 8, pp. 7236–7262, 2010.

[4] E. Cayirci and T. Coplu, “SENDROM: sensor networks for disaster relief operations management,” Wireless Networks, vol. 13, no. 3, pp. 409–423, 2007.

[5] H. Alemdar and C. Ersoy, “Wireless sensor networks for healthcare: a survey,” Computer Networks, vol. 54, no. 15, pp. 2688–2710, 2010.

[6] G. J. Pottie and W. J. Kaiser, “Wireless integrated network sensors,” Communications of the ACM, vol. 43, no. 5, pp. 51–58, 2000.

[7] Y. C. Lee and A. Y. Zomaya, “On effective slack reclamation in task scheduling for energy reduction,” Journal of Information Processing Systems, vol. 5, no. 4, pp. 175–186, 2009.

[8] J. Subhlok, P. Lieu, and B. Loweikamp, “Automatic node selection for high performance applications on networks,” in Proceedings of the 7th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP ’99), pp. 163–172, Atlanta, Ga, USA, May 1999.

[9] Y. H. Park, K. Y. Whang, B. S. Lee, and W. S. Han, “Efficient evaluation of linear path expressions on large-scale heterogeneous XML documents using information retrieval techniques,” Journal of Systems and Software, vol. 79, no. 2, pp. 180–190, 2006.

[10] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, “TAG: a tiny aggregation service for ad-hoc sensor networks,” in Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI ’02), pp. 131–146, December 2002.