

Energy-Efficient Aggregate Query Evaluation in Sensor Networks

Zhuoyuan Tu Weifa Liang

Department of Computer Science

Australian National University

Canberra, ACT 0200, Australia

{zytu,wliang}@cs.anu.edu.au

Abstract

Wireless sensor networks have been widely used for communication and surveillance environments. Data collected by the sensor devices needs to be extracted and aggregated for a wide variety of purposes. Due to the serious energy constraint imposed on such a network, it is a great challenge to perform aggregate queries efficiently. In this paper, we consider the energy-efficient aggregate query evaluation in a sensor network database with an objective to prolong the network lifetime. We first propose a heuristic algorithm by introducing a node capability concept which balances the residual energy and the energy consumption at each node. We then present an improved algorithm by incorporating group aggregation and conditional clauses into our routing protocol. We finally evaluate the performance of the proposed algorithms against the existing algorithms through simulation. The experimental results show that the proposed algorithms significantly outperform the existing ones in terms of network lifetime.

Keywords: *Sensor network, energy efficiency, aggregate query, network lifetime, group aggregation, sensor database.*

1 Introduction

In the recent years, inevitable research tendency has been moving towards wireless sensor networks, which consist of a large number of active, low-battery-supplied sensors for the powerful data collection and monitoring systems. This new class of sensors are differentiated from the other wireless devices in that their autonomous, tether-less nature makes users free from the concerns like configuration of network routes, recharging of batteries and tuning of parameters [16]. An example of such small sensor devices is the Intel *mote 2* based on the modular design of the original Berkeley motes (see Fig. 1), which runs on the top of the *TinyOS* operating system [6] and measures approximately 2mm x 2.5mm x 1mm, consisting of an *ARM7* core processor, a *Bluetooth* radio, *RAM* and *FLASH* memory, a small battery pack, as well as various *I/O* options. Such motes can also be connected to *PCs*, *PDA*s, the *WWW*, and existing wired networks, given the gateway and interface products.



(a) The Top Side



(b) The Bottom Side

Figure 1: Intel Mote 2

Due to the commercial maturation on both hardware and operating system¹, the mote-based sensor networks will soon be widely used for the various purposes. Applications range from the intelligent civilian surveillance, the fine-grained natural habitat monitoring, to the local vibration prediction, and the military object recognition. Recent advances in microelectronic technologies empower these sensor devices to monitor information at previously unobtainable resolutions [3]. Using these sensor devices, biologists are able to obtain the ambient conditions for endangered plants and animals every few seconds. Security guards can detect the subtle temperature variation in storage warehouses to avoid fire. Medicine manufacturers can maintain the strict requirements on the environmental parameters during product manufacturing. These

¹MICA Motes from Crossbow, Inc. http://www.xbow.com/Products/Wireless_Sensor_Networks.htm.

inexpensive sensor devices gracefully exhibit their blatant monitoring superiority, specially where the macro sensing counterparts are not suitable to be deployed. Prospective application will arise in a more dynamic manner. Instead of being physically-fixed, future sensor devices are capable of locating each other and routing data in ignorance of the network topology, thereby allowing the network topology to change as sensors move [14].

Such sensor networks, however, have severe resource constraints on communication, computation and energy consumption. First, the typical communication distances for low power wireless radios used in sensor devices are around a hundred feet, which means sensor nodes need to use multihop routing protocols to communicate with peers out of their transmission ranges. Limited transmission range, limited connecting quality, coupled with limited bandwidth, result in the variable latency and the frequent packet drop. Second, sensor nodes have limited computing ability and memory size. For example, Berkeley *Mica* mote has a 4-MHz, 8-bit Atmel processor and a 518KB storage. This restricts the complexity of data processing and the size of intermediate results stored on a sensor node [19]. Third, sensor nodes are extremely low-power-supplied. The current commercial motes are usually equipped with a pair of AA batteries, and their lifetime can range from days to months or even years. For example, if a 2200 mAh pair of batteries are used naively on 15 mA current, the lifetime of motes will be $2200 / 15 = 146$ hours, which is approximately 6 days. In term of 1% operational duty cycle between active and sleep modes, however, individual nodes can achieve the lifetime in one year.

In summary, the energy conservation issue is of paramount importance in sensor networks, because one node failure, due to its battery exhaustion, can paralyze the entire networks. Therefore, the *network lifetime* of a wireless sensor network is defined as the time of the first node failure in it [1].

1.1 Sensor Data and Queries

To meet various monitoring requirements, a large number of sensor devices are deployed distributedly to monitor the physical phenomenon. Each sensor node operates as an autonomous data source. Data from different nodes has the same data schema and collectively forms a distributed relational table, so the interested data from the table can be extracted or aggregated for the user requirements. Therefore, a sensor network

can be treated as a distributed database system, where sensor data generated by each sensor node can be viewed as a horizontal fragment of a relational table, as shown in Figure 2. Here we assume that during each time interval a sensor node only produces

Sensor_id	Building_no.	Level_no.	Room_no.	Temperature
255	103	2	302	26.3
255	103	2	302	26.5
255	103	2	302	27.2
255	103	2	302	28.6

Figure 2: A fragment of a relational table produced by Sensor no. 255

a tuple (a row in Fig.2). An attribute (a column in Fig.2) of a tuple is either the information about the sensor node itself (e.g., its id or location), or the data detected by the node (e.g., the temperature in its vicinity). Note that in some complicated sensor network systems, more than one type of sensor devices need to be used for extensive information collection. For example, there are three kinds of sensors employed to monitor temperature, light and humidity respectively. Such a complicated sensor network can still be viewed as a database which has several different relational schemas and consists of multiple tables from different types of sensors. For simplicity, in this paper we only consider the sensor network database with one relational schema.

To interact with a sensor network, a preferred way is to use *declarative queries*. Instead of inventing a new language to express such queries, a SQL-style query is adopted. That is, users simply specify the data that they are interested in and pose various SQL-like queries over the sensor network as follows.

```

SELECT    {attributes, aggregates}
FROM      sensors
WHERE     condition-of-attributes
GROUP BY {attributes}
HAVING    condition-of-aggregates
DURATION  time interval

```

As the interface between the users and the sensor network, a special sensor node called the *base station* is distinguished with unlimited energy supply. It is used to broadcast the user queries throughout the entire network and collect the query results back to the

users. The semantics of the above query template are similar to those in SQL, except that the DURATION clause specifies the interval between two consecutive query results and thus supports the long running, periodic queries. Such a declarative query model, as a level of abstraction, spares the clients from the concerns of relevant sensors locating, detailed data processing and results gathering, thus becomes a prevalent interaction approach between users and sensor network.

1.2 In-Network Aggregation

To respond to a user aggregate query, the network can process in either centralized or in-network manner. In the centralized implementation, all the messages generated by the sensor nodes are transmitted to the base station directly, and the interested information will then be extracted and computed centrally at the base station. However, this centralized processing is very expensive due to the tremendous energy consumption on the message transmission. By virtue of the autonomous, full-fledged computation ability of sensor nodes, it is possible to filter or combine messages locally. Since the energy consumption in sensor networks is dominated by the wireless communication, the application will benefit in the energy efficiency by aggregating the data inside the network rather than simply transmitting them to the base station. Such an *in-network aggregation* manner can dramatically decrease the number of the message transmissions by pushing the partial computation tasks from the base station into the intermediate nodes, so has been used as a prevalent approach for the energy savings in sensor networks.

To process an aggregate query, the in-network aggregation will be divided in two phases: the *distribution* phase, which broadcasts the aggregate query down into the network; and the *collection* phase, in which the sensed messages are collected and aggregated at each relay node and then continually routed up to its parent [14]. In the distribution phase, a tree rooted at the base station and spanning all the sensor nodes is constructed for the given aggregate query. The construction of the routing tree is initiated from the base station (the *root*) and pursues along with the propagation of the query. In the collection phase, each relay node (except the leaf nodes) is allowed to perform data aggregation locally before transmitting the result to its parent. Ultimately, the aggregate result will be relayed to the base station.

During the data aggregation, each node in the routing tree will be assigned into

a group according to the distinct value of a list of Group-By attributes in a SQL-like query (see above). Messages from different nodes are merged into one message at each relay node in the tree if they belong to the same group [17]. For example, if we pose a query “the average temperature in each building”, each sensor node will first generate its own sensed message and collect the messages from its descendants in the tree, and then use SUM and COUNT functions in SQL to compute the average temperature for each group (each building) before forwarding the result to its parent. In the end, all the messages in the same building will be merged into one message and the number of messages finally received by the base station is equal to the number of buildings, so that the transmission energy consumption will be dramatically reduced. Inspired by the energy efficiency from this group aggregation, we will later use it to improve our algorithm and show its benefit in details.

1.3 Our Contributions

To prolong the network lifetime when dealing with aggregate query evaluation, in this paper we first propose an algorithm by introducing the node capability concept to balance the residual energy and the energy consumption at each node. We then present an improved algorithm which allows the group aggregation to reduce the energy consumption. We finally conduct experiments by simulation. The experimental results show that the proposed algorithms outperform the existing ones.

The rest of the paper is organized as follows. Section 2 provides the network model and the problem definition. Section 3 introduces the node capability concept and a heuristic algorithm. Section 4 presents an improved algorithm with the group aggregation and pursues the further improvements by incorporating the *where* and *having* conditional clauses into consideration. Section 5 conducts extensive experiments to evaluate the performance of the proposed algorithms. Section 6 reviews the related work. Section 7 concludes the paper.

2 Preliminaries

2.1 Network Model

Assume that a sensor network consists of n homogeneous energy-constrained sensor nodes and an infinite-energy-supplied base station s deployed over an interested region. Each sensor periodically produces sensor data as it monitors its vicinity. The communication between two sensor nodes is done either directly (if they are within the transmission range of each other) or through the relay nodes. The network can be modeled as a directed graph $M = (N, A)$, where N is the set of nodes with $|N| = n + 1$ and there is a directed edge $\langle u, v \rangle$ in A if node v is within the transmission range of u . The energy consumption for transmitting a m -bit message from u to v is modeled to be $md_{v,u}^\alpha$, where $d_{v,u}$ is the distance from u to v and α is a parameter that typically takes on a value between 2 and 4, depending on the characteristics of the communication medium.

2.2 Problem Definition

Given an aggregate query issued at the base station, the problem is to evaluate the query against the sensor network database by constructing a spanning tree rooted at the base station. We aim to find the energy-efficient routing tree protocol to maximize the number of queries that can be evaluated before the first node failure in the network, such that the network lifetime is maximized. We refer to this problem as *the lifetime-maximized routing tree problem* (LmRTP for short).

3 Algorithm LmNC

In this section we introduce the node capability concept and propose a heuristic algorithm called the **Lifetime-maximized Network Configuration** (LmNC) for LmRTP based on the capability concept.

3.1 Capability Concept

Given a node v , let $p(v)$ be the parent of v in a routing tree. The energy consumption for transmitting a m -bit message from v to $p(v)$ is $E_c(v, p(v)) = md_{v,p(v)}^\alpha$, where $d_{v,p(v)}$ is the distance between v and $p(v)$. Let $E_r(v)$ be the residual energy of v before evaluating the current query. Assume that the length of the message by every node is the same (m -bit), then the capability of node v to $p(v)$ is defined as

$$C(v, p(v)) = E_r(v)/E_c(v, p(v)) - 1 = E_r(v)/md_{v,p(v)}^\alpha - 1. \quad (1)$$

If v has k descendants in the routing tree, then the energy consumption at v to forward all the messages (its own generated message and the messages collected from its descendants) to its parent $p(v)$ will be $(k+1)md_{v,p(v)}^\alpha$, given that there is no aggregation at v . If after this transmission, v will exhaust its residual energy, then $E_r(v) = (k+1)md_{v,p(v)}^\alpha$. From Eq. (1), it is easy to derive that $k = E_r(v)/md_{v,p(v)}^\alpha - 1 = C(v, p(v))$. So, if there is no aggregation at v , the capability of node v to $p(v)$, $E_c(v, p(v))$, actually indicates the maximum number of descendants that it can support by its current residual energy.

3.2 Algorithm Description

The basic idea behind algorithm LmNC is as follows. Since a node with the larger capability can have more descendants in the routing tree (if data aggregation is not allowed), it should be placed closer to the tree root to prolong the network lifetime.

Based on this idea, we propose an algorithm LmNC, where each time a node with the maximum capability is included into the current tree. Thus, the nodes are added one by one until all the nodes are included in the tree. The motivation behind this algorithm is that adding the node with the maximum capability innately balances the node residual energy $E_r(v)$ and the actual energy consumption for transmitting a message to its parent $E_c(v, p(v))$ (as the definition of the node capability), so that the network lifetime is dramatically prolonged. Specifically, we denote by T the current tree and V_T the set of nodes included in T so far. Initially, T only includes the base station, i.e. $V_T = \{s\}$. Algorithm LmNC repeatedly picks up a node v ($v \in V - V_T$) with maximum capability to u ($u \in V_T$) and adds it into T with u as its parent. The algorithm continues until $V - V_T = \emptyset$. The detailed algorithm is given below.

Algorithm Lifetime_Efficient_Network_Configuration (G, E_r)

/ G is the current sensor network and E_r is an array of the residual energy of the nodes */*

begin

1. $V_T \leftarrow \{s\}$; */* add the base station into the tree */*
2. $Q \leftarrow V - V_T$; */* the set of nodes which is not in the tree*/*
3. while $Q \neq \emptyset$ do
4. $C_{\max} \leftarrow 0$; */* the maximal capability of nodes in the tree */*
5. for each $v \in Q$ and $u \in V_T$ do
6. compute $C(v, u)$;
7. if $C_{\max} < C(v, u)$
 then $C_{\max} \leftarrow C(v, u)$;
 $added_node \leftarrow v$;
 $temp_parent \leftarrow u$;
8. $p(added_node) \leftarrow temp_parent$;
 / set the parent for the node with maximum capability */*
9. $V_T \leftarrow V_T \cup \{added_node\}$;
 / add node with maximum capability into tree */*
10. $Q \leftarrow Q - \{added_node\}$;

end.

Note that, although there have been several algorithms for LmRTP considering the residual energy of nodes during the construction of the routing tree (including [1]), they failed to consider the actual transmission energy consumption from a node to its parent. This can be illustrated by the following example.

Assume that there are a partially built routing tree and a number of nodes to be added into the current tree. Further assume that node v_i has the maximum residual energy among the nodes out of the tree, while the distance between v_i and its parent is much longer than that between another node v_j and its parent. Now, if node v_i is added into the tree, it will be prone to die in the further tree construction because of the enormous transmission energy consumption from v_i to its parent. Therefore, although v_i has more residual energy than v_j at the moment, the maximum number of the messages transmitted by v_i to its parent is less than that by v_j . We thus conclude that the lifetime of node v_i is shorter than that of node v_j .

4 Improved Algorithm LmGaNc

Although algorithm LmNC manifests the significant improvement on the network lifetime for LmRTP, the total energy consumption for the entire network is hardly considered, because the node with maximum capability may be far away from its parent, and the excess transmission energy consumption by the node will be triggered. In this section we present an improved algorithm **Lifetime-maximized Group-aware Network Configuration (LmGaNc)** which allows group aggregation to reduce the energy consumption, and pursue the further improvements by incorporating the conditional clauses into consideration.

4.1 Algorithm Description

Since group aggregation is able to combine the messages from the same group into one message, clustering the nodes of the same group into a routing path will reduce the energy consumption and maximize the network lifetime, because the messages drawn from these nodes will contain fewer groups. With this idea, Sharaf *et al* provided a heuristic algorithm **GaNc** (in [17]) to construct an energy-efficient routing tree. Further incorporating this idea into algorithm LmNC, an improved algorithm **LmGaNc** is proposed as follows.

Algorithm Lifetime_Efficient_Network_Configuration (G, E_r)

/ G is the current sensor network and E_r is an array of the residual energy of the nodes */*

begin

1. $V_T \leftarrow \{s\}$; */* add the base station into the tree */*
2. $Q \leftarrow V - V_T$; */* the set of nodes which is not in the tree*/*
3. while $Q \neq \emptyset$ do
4. $C_{\max} \leftarrow 0$; */* the maximal capability of nodes in the tree */*
5. for each $v \in Q$ and $u \in V_T$ do
6. compute $C(v, u)$;
7. if $C_{\max} < C(v, u)$
 then $C_{\max} \leftarrow C(v, u)$;
 $added_node \leftarrow v$;
 $temp_parent \leftarrow u$;
8. $p(added_node) \leftarrow temp_parent$;

```

/* set the parent for the node with maximum capability */
9.    $d_{min} \leftarrow \infty$ ; /* minimum distance to choose */
10.  for each  $u' \in V_T$  and  $u' \neq temp\_parent$  do
11.    if  $group\_id(u') = group\_id(added\_node)$  and
         $d_{added\_node,u'} \leq df * d_{added\_node,temp\_parent}$  and
         $d_{added\_node,u'} < d_{min}$ 
        then  $p(added\_node) \leftarrow u'$ ;
12.   $V_T \leftarrow V_T \cup \{added\_node\}$ ;
    /* add node with maximum capability into tree */
13.   $Q \leftarrow Q - \{added\_node\}$ ;
end.

```

Algorithm LmGaNC is similar to algorithm LmNC. The difference is that, during the construction of the routing tree, a child with the maximum capability chosen by LmNC will keep checking whether there is a node in the same group as itself in the current tree in terms of LmGaNC (we call this node a better parent). If yes, the child is allowed to switch to this better parent. If there are more than one better parent to choose from, the closest one will be chosen. Notice that choosing a better parent far away from a child will cause the extra transmission energy consumption. Under this circumstance, a concept of *distance factor* (df) is employed, which is the upper bound of the distance between a child and its selected parent. For example, if $df=1.5$, then we only consider the parent whose distance to the child is at most $df * d_{v,u} = 1.5d_{v,u}$, where $d_{v,u}$ is the distance between a child v and its current parent u .

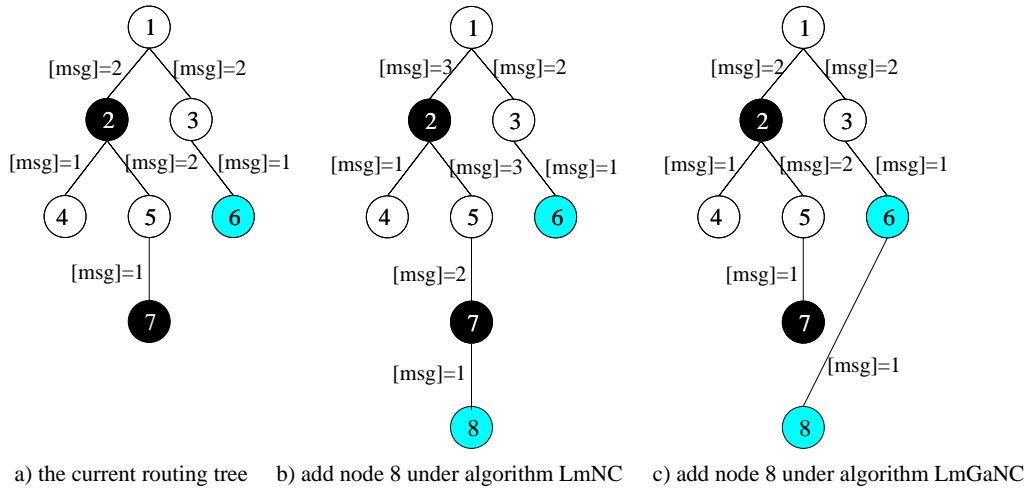


Figure 3: Benefit of algorithm LmGaNC

Energy reduction brought by the improved algorithm **LmGaNc** is demonstrated by the following example. In Figure 3, we have a partially built routing tree with node 1 as the tree root (See Fig. 3(a)). Assume that black nodes 2 and 7 belong to Group 1, shaded node 6 belongs to Group 2, and the rest of nodes belong to Group 3. The numbers of the messages are as shown in the figure (depending on the number of various groups in the subtree). Now, under algorithm **LmNC**, shaded node 8 (in Group 2) has the maximum capability to its parent node 7 (See Fig. 3(b)). In order to forward one message originally from node 8, all the nodes in the path from node 8 to the root, except the root node 1 with constant energy supply, have to consume extra energy for this transmission, because node 8 is in a different group. While under the improved algorithm **LmGaNc**, node 8 can switch to a better parent which is in the same group as node 8 (Group 2). Therefore, node 8 will choose node 6 as its new parent, assuming without violation of the distance factor. As a result, none of the nodes, except node 8 itself, needs to forward one extra message for node 8, so that significant energy savings can be achieved.

Recall that a node capability is the maximum number of descendants that the node can have by its current residual energy without data aggregation. More generally speaking, the capability of a node actually indicates the maximum number of the messages that it can forward for its descendants. Here, after applying group aggregation, a node capability indicates the number of its descendants in different groups (excluding the group of the node itself) rather than the total number of its descendants, because the messages from the descendants in the same group can be merged into one message only.

4.2 Further Improvement

In this subsection, we consider the impact of *where* and *having* conditional clauses on the aggregate query results. Our plan aims to further achieve the energy efficiency of algorithm **LmGaNc**.

4.2.1 Effect of *Where* Clause

For more complicated queries, it is very common to use some conditions or thresholds to filter out the unwanted query results. For example, to answer the query “what is the

average temperature in each room at level 5?”, we need to use “GROUP BY Room_no.” to partition the query results into a set of room groups and calculate the average temperature for each room group. However, to exclude the nodes which are not at level 5, we normally employ the *where* clause originated from the SQL language to further reduce the energy consumption. So, in the above case, the condition clause “WHERE Level_no. = 5” will be imposed on the query. Before each sensor node in the routing tree transmits its sensor data to its parent, it will check whether its sensor data matches the condition in the query specification. If not, then the node just transmits a bit of notification information with value 0 to its parent, rather than its original data, so that its parent will not keep waiting for the data from this child. This is especially true under the aggregation schema in Cougar [19], where each node holds a waiting list for its children and will not transmit its data to its parent until it hears from all the nodes on the waiting list. Since the size of the transmitted data is shrunk into only 1 bit given the mismatch, the transmission energy consumption will be further reduced. However, even if a node matches the query condition, whether its residual energy can afford the message transmission is still questionable. One possible solution for this is that the node checks whether it has sufficient residual energy to complete the transmission. If not, it will send a bit of notification information with value 1 instead of its original data to its parent to indicate the insufficiency of its residual energy.

4.3 Effect of *Having* Clause

Note that *where* clause can be used in both aggregate and the non-aggregate queries. If it is imposed on an aggregate query, it will be considered upon all the sensor data locally before the group aggregation. While *having* clause in our query model can only be used when group aggregation is compulsory for the query and it filters out the unneeded groups from the final aggregate results. Due to this fact, at the most of time *having* clause is centrally used at the base station to exclude the undesired groups after the group aggregation. For example, predicates AVG, COUNT and SUM can not be sent down to the network, because they can only be measured upon the final aggregate results, rather than the local inaccurate aggregate values.

However, for some special cases, *having* conditions can also be pushed down into the sensor network to reduce the number of transmitted messages before group aggregation. For example, to check if the air-conditioning system works normally, if we

pose the query “which room has its minimum temperature above 20 Celsius degrees?”, *having* clause will be of the form “HAVING MIN(Temperature)>20”. In this case, if the condition is pushed down into the routing tree, then the local sensor data with temperature lower than 20 does not need to be transmitted up along the tree, and meanwhile the whole group this node belongs to will be evicted. This can be easily illustrated by “MIN(Temperature) \leq Local(Temperature) \leq 20”, which means the minimum temperature of this group is never larger than 20 Celsius degrees and the whole group does not satisfy the clause “ HAVING MIN(Temperature) > 20”. So, when a node detects its local message does not satisfy the condition as above, it can cancel its message transmission and notify other nodes from its group in the network to suppress their transmission, too. More generally, this query plan can only be used when considering the *having* conditions like “MIN(attribute)>Threshold” or “MAX(attribute)<Threshold”. However, for the *having* condition like “MIN(attribute)<Threshold” or “MAX(attribute)>Threshold”, the transmission of the local message which is out of the condition can still be suppressed, as this useless message will not be shown up in the final result after applying the *having* condition. But the message transmission of the other nodes from the same group should not be affected, because the local value is larger than x does not mean that the future minimum value of the whole group is larger than x .

5 Simulations

This section evaluates the performance of the proposed algorithms LmNC and LmGaNc against the existing algorithms including MST (Minimum Spanning Tree), SPT (Shortest Path Tree) and GaNC. The experimental metrics are the network lifetime and the total network energy consumption and energy consumption per each query, based on different numbers of groups, various distance factors, and with and without *where* and *having* condition clauses.

We assumed that network topologies are randomly generated by the *NS-2* network simulator with the nodes distributed in a 100×100 m^2 region and each sensor node is initially equipped with 10^5 μ -Joules energy. Two nodes are only connected when they are within the transmission range of each other. For each aggregate query, we randomly assign an integer ‘Group_id’ i ($i \in [1, n]$, n is the number of groups) for each node and construct a routing tree to process the query. The network lifetime refers

to the maximum number of queries that the network can process before the first node failure, and the total energy consumption refers to the total network transmission cost during the entire network lifetime. We take the average of the experimental results from 30 distinct network topologies for each network size.

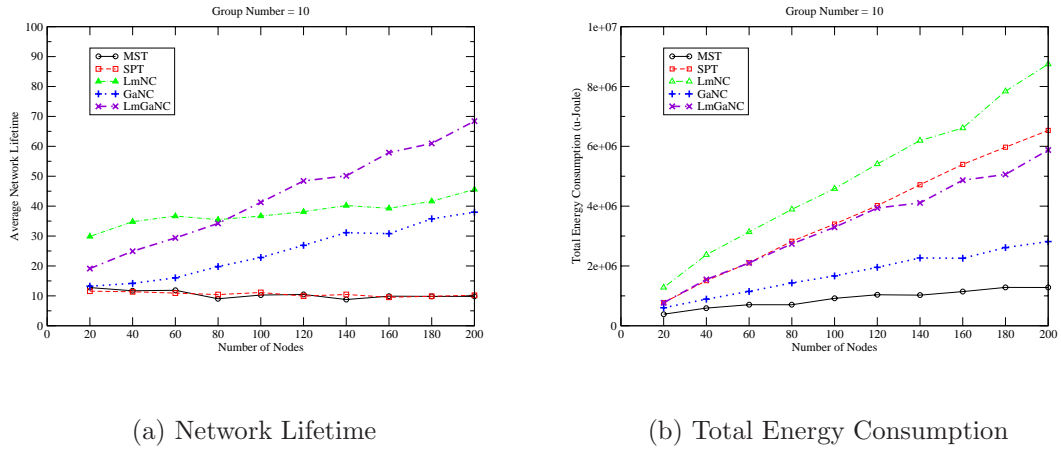
5.1 Performance of the Two Proposed Algorithms

Before we proceed, we reproduce an existing heuristic algorithm called GaNC [17] for the concerned problem, which will be used as the benchmark. Algorithm GaNC with the group aggregation concept is derived from a simple First-Heard-From (FHF) protocol where the nodes always select the first node from which they hear as their parents after the query specification is broadcast over the network. The main difference between GaNC and FHF is that the child under GaNC can change to a better parent in the same group within the given distance factor.

The simulation results in Figure 4(a) show that the network lifetimes delivered by algorithms LmNC and LmGaNC significantly outperform the ones delivered by MST, SPT and GaNC. Figure 4(b) indicates that algorithm LmGaNC gracefully balances the energy consumption of LmNC. Note that the total energy consumption delivered by our algorithms in Figure 4(b) actually involves longer network lifetime, which means more rounds of query processing. For clarification, we allocate the energy consumption into each query processing in Figure 4(c), which clearly shows that LmGaNC brings the average energy consumption on each query under control. We ignore the effect of the distance factor at this stage and set $df = 100$.

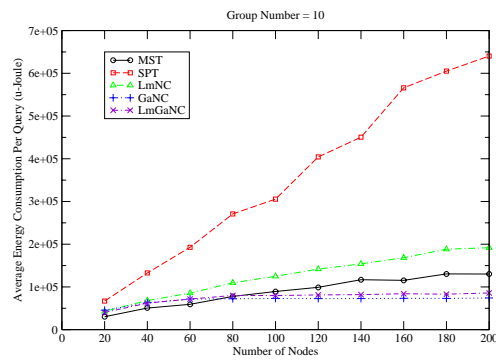
5.2 Sensitivity to The Number of Groups

Figure 5 indicates, when the number of groups is decreased from 10 to 5, both of the two algorithms manifest their lifetime improvements. The reasons behind is as follows. On one hand, fewer groups mean that more sensor nodes will be in the same group, and thus the possibility of message suppression under the group aggregation will be enhanced. On the other hand, fewer groups make a child node have more chances to switch to a better parent in the same group, so less transmission energy will be consumed. The distance factor here is also set to be 100.



(a) Network Lifetime

(b) Total Energy Consumption



(c) Average Energy Consumption Per Query

Figure 4: Performance comparison among various algorithms

5.3 Sensitivity to The Distance Factor

As discussed earlier, the distance factor is introduced to limit the maximum distance that is acceptable when a child node switches to a better parent in the same group. It avoids unnecessary energy dissipation resulting from this switching. As such, the smaller the distance factor is, the less the energy dissipation will be, therefore, the longer the network can endure.

Figure 6 shows that when distance factor is decreased from 3 to 1.5, algorithm LmGaNC exhibits its sensitivity immediately and the network lifetime is significantly prolonged, while GaNC reacts much more rigidly.

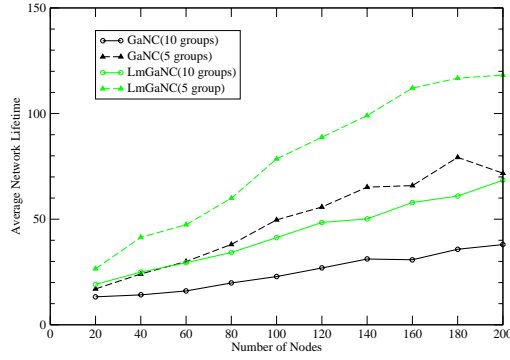


Figure 5: *LmGaNC* vs *GaNC* in Different Numbers of Groups

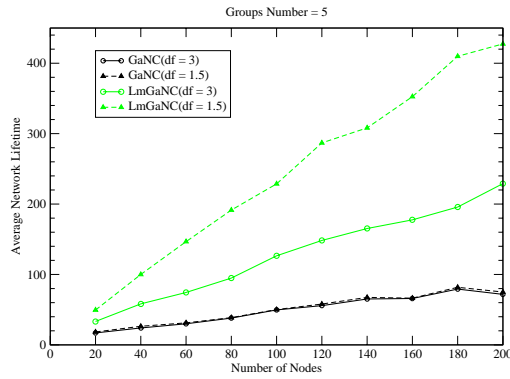
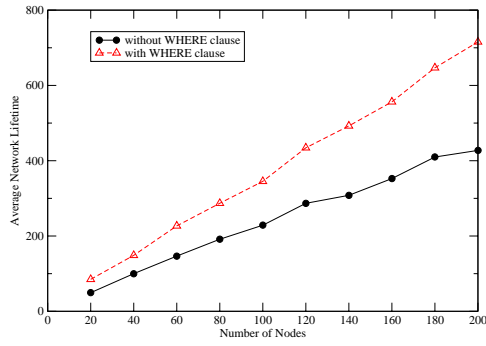


Figure 6: *LmGaNC* vs *GaNC* in Different Distance Factors

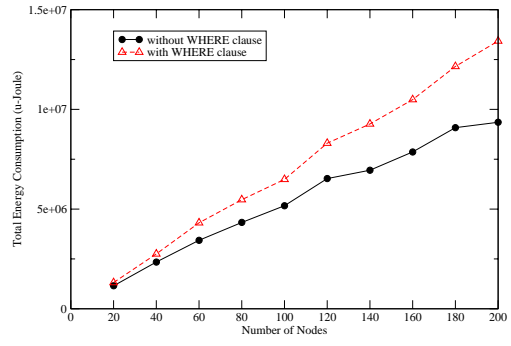
5.4 Sensitivity to The *Where* Condition Clause

The experiments here aim to further reduce the energy consumption of evaluating an aggregate query through allowing the nodes that mismatch the *where* condition to send a 1-bit notification to their parents instead of the sensor data. Figure 7 illustrates the effects of the *where* clause in an aggregate query on both the network lifetime and the energy consumption, when group number = 5 and distance factor = 1.5.

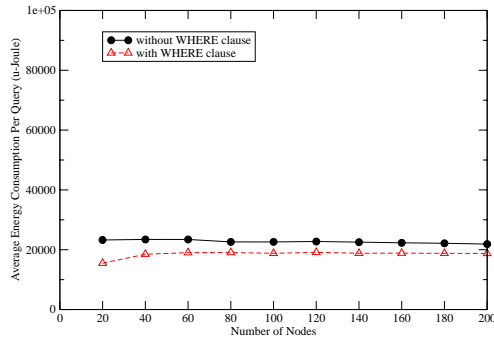
The experimental results show that the network lifetime increases by more than 50% while the total energy consumption only goes up by around 25% under *LmGaNC*, which indicates that the average energy consumption on each query is actually reduced by employing *where* clause as shown in Figure 7(c).



(a) Network Lifetime



(b) Total Energy Consumption

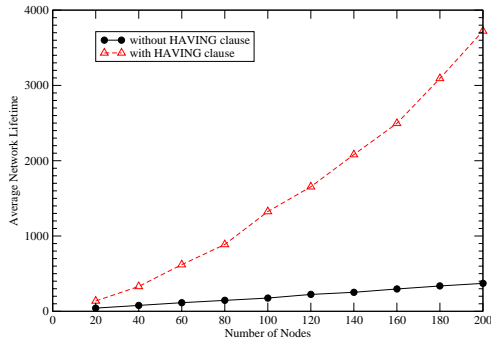


(c) Average Energy Consumption Per Query

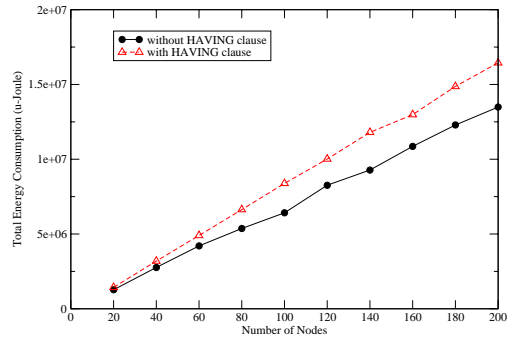
Figure 7: Performance of LmGaNc with *Where* Clause

5.5 Sensitivity to The *Having* Condition Clause

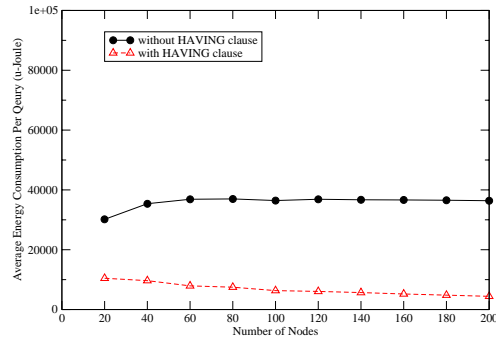
Most *having* clauses do not contribute to the energy savings during the aggregate query evaluation, except for MAX and MIN predicates. For the conditions like “MIN(attribute) < Threshold” and “MAX(attribute) > Threshold”, the local mismatched messages can be suppressed as *where* clause. While for the conditions like “MIN(attribute) > Threshold” and “MAX(attribute) < Threshold”, the massive energy savings can be achieved by flooding these predicates into the network, because not only the local undesired messages but also the messages from the whole group can be suppressed from the transmission. If we assume the selectivity rate of local messages is 50%, there are 10 groups and the distance factor is set to be 1.5, then the considerable improvement on the network lifetime and the energy consumption will be as shown in Figure 8.



(a) Network Lifetime



(b) Total Energy Consumption



(c) Average Energy Consumption Per Query

Figure 8: Performance of LmGaNc with *Having* Clause

6 Related Work

To improve the energy efficiency and prolong the network lifetime, several existing protocols for various routing problems have been proposed in both ad hoc networks and sensor networks [1, 2, 4, 7, 8, 9, 11, 18]. For example, in ad hoc networks Chang and Tassiulas [1, 2] realized a group of unicast requests by discouraging the participation of low energy nodes. Kang and Poovendran [9] provided a globally optimal solution for broadcasting through a graph theoretic approach. While in sensor networks, Heinzelman *et al* [4] initialized the study of data gathering by proposing a clustering protocol LEACH, in which nodes are grouped into a number of clusters. Within a cluster, a node is chosen as the cluster head which is used to gather and aggregate the data from the other members and forward the aggregated result to the base station directly.

Lindsey and Raghavendra [11] provided an improved protocol PEGASIS using a chain concept, where all the nodes in network form a chain and one of the nodes is chosen as the chain head in turn to report the aggregated results to the base station. Tan and Kórpeoğlu [18] provided a protocol PEDAP for the data gathering problem, which constructs a minimum spanning tree (MST) rooted at the base station to limit the total energy consumption. Kalpakis *et al* [8] considered a generic data gathering problem with an objective to maximize the network lifetime, for which they proposed an integer program solution and a heuristic solution.

This paper provides the evaluation of an aggregate query in a sensor network with an objective to prolong the network lifetime. The pervasive way to do this is to apply the in-network aggregation, as discussed above, to proceed the query evaluation. This has been presented in [12, 20], where the information-directed routing is proposed to minimize the transmission energy consumption. In addition, query semantics for efficient data routing has also been considered in [15] to save transmission energy, in which a semantic routing tree (SRT) is used to exclude the nodes that the query does not apply to. Furthermore, group aggregation has been incorporated into the routing algorithm GaNC in [17], where the sensor nodes in the same group are clustered along the same routing path with the goal of reducing the size of transmitted data. However, an obvious indiscretion in some of the routing protocols, such as MST and GaNC, is that a node is chosen to be added into tree without taking into account its residual energy during the construction of the routing tree. As a result, the nodes closer to the root of the routing tree will exhaust their energy rapidly due to the fact that they serve as relay nodes and forward the messages for their descendants in the tree. Thus, the network lifetime is shortened.

7 Conclusions

In this paper we considered the aggregate query evaluation in a sensor network database with the aim to prolong the network lifetime. Based on the node capability concept, we first proposed a heuristic algorithm to prolong the network lifetime, then presented the improved algorithm by incorporating group aggregation to reduce the energy consumption. We finally conducted experiments to evaluate the performance of the proposed algorithms against those of the existing ones. The experimental results showed that the proposed algorithms outperform the existing algorithms significantly.

Acknowledgment. It is acknowledged that the work by the authors was partially supported by a research grant from the Faculty of Engineering and Information Technology at the Australian National University.

References

- [1] J-H Chang and L. Tassiulas. Energy conserving routing in wireless ad hoc networks. *Proc. of INFOCOM'00*, IEEE, 2000.
- [2] J-H Chang and L. Tassiulas. Fast approximate algorithms for maximum lifetime routing in wireless ad hoc networks. *IFIP-TC6/European Commission Int'l Conf.*, Lecture Notes in Computer Science, Vol. 1815, pp. 702–713, Springer, 2000.
- [3] J. Gehrke and S. Madden. Query processing in sensor networks. *ComSoc'04*, IEEE, 2004.
- [4] W. R. Heinzelman, A. Chandrakasan and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. *Proc. of 33th Hawaii International Conference on System Sciences*, IEEE, 2000.
- [5] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin and D. Ganesan. Building efficient wireless sensor networks with low-level naming. *Proc. of 18th Symposium on Operating Systems Principles (SOSP'01)*, ACM, 2001.
- [6] J. Hill, R. Szewczyk, A. Woo, S. Hollar, and D. C. K. Pister. System architecture directions for networked sensors. *Proc. of ASPLOS*, ACM, 2000.
- [7] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. *Proc. of 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, IEEE, 2002.
- [8] K. Kalpakis, K. Dasgupta and P. Namjoshi. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computer Networks*, Vol. 42, pp. 697–716, 2003.
- [9] I. Kang and R. Poovendran. Maximizing static network lifetime of wireless broadcast ad hoc networks. *Proc. of ICC'03*, IEEE, 2003.

- [10] W. Liang. Constructing minimum-energy broadcast trees in wireless ad hoc networks. *Proc. of MOBILHOC'02*, ACM, 2002.
- [11] S. Lindsey and C. S. Raghavendra. PEGASIS: Power-efficient gathering in sensor information systems. *Proc. of Aerospace Conference*, IEEE, pp. 1125–1130, 2002.
- [12] J. Liu, F. Zhao, and D. Petrovic. Information-directed routing in ad hoc sensor networks. *Proc. of 2nd international conference on wireless sensor networks and applications*, ACM, 2003.
- [13] A. Manjeshwar and D. P. Agrawal. APTEEN: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. *Proc. of IPDPS'02*, IEEE, 2002.
- [14] S. Madden, M. J. Franklin, J. Hellerstein and W. Hong. TAG: a tiny aggregation service for ad-hoc sensor networks. *Proc. of 5th symposium on Operating systems design and implementation*, ACM, 2002.
- [15] S. Madden, M. J. Franklin, J. Hellerstein and W. Hong. The design of an acquisitional query processor for sensor networks. *Proc. of SIGMOD'03*, ACM, 2003.
- [16] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TinyDB: An acquisitional query processing system for sensor networks. *ACM Transaction on Database Systems*, Vol. 30, pp. 122-173, 2005.
- [17] M. A. Sharaf, J. Beaver, A. Labrinidis and P. K. Chrysanthis. Balancing energy efficiency and quality of aggregate data in sensor networks. *J. VLDB*, Springer, 2004.
- [18] H. Ó. Tan and İ. Kórpeoğlu. Power efficient data gathering and aggregation in wireless sensor networks. *ACM SIGMOD Record*, Vol. 32, No. 4, pp. 66–71, 2003.
- [19] Y. Yao and J. Gehrke. Query processing in sensor networks. *Proc. of 1st Biennial Conf. Innovative Data Systems Research (CIDR'03)*, ACM, 2003.
- [20] F. Zhao, J. Shin, and J. Reich. Information-driven dynamic sensor collaboration. *IEEE Signal Processing Magazine*, Vol. 19, 2002.