Ad Hoc & Sensor Wireless Networks, Vol. 3, pp. 55–75
Reprints available directly from the publisher
Photocopying permitted by license only

Energy-Efficient Aggregate Query Evaluation in Wireless Sensor Networks

ZHUOYUAN TU, WEIFA LIANG

Department of Computer Science, Australian National University,
Canberra, ACT 0200, Australia.
E-mail: zytu,wliang@cs.anu.edu.au

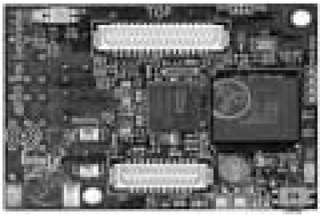
Received: November 11, 2005. In Final Form: February 28, 2006.

Wireless sensor networks, consisting of sensors equipped with energy-limited batteries, have been receiving significant attention and are increasingly being deployed for environmental monitoring and surveillance. Data generated by the sensors usually 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 aggregate query evaluation problem in wireless sensor networks with an objective to prolong the network lifetime. To do so, we first introduce a node capability concept, which balances the residual energy and the consumed energy at each node, and propose a heuristic algorithm based on this concept. We then present an improved algorithm by incorporating group aggregation and conditional clauses into consideration. We finally evaluate the performance of the proposed algorithms against the existing algorithms through experimental simulations. The experimental results show that the proposed algorithms significantly outperform the existing ones in terms of network lifetime.

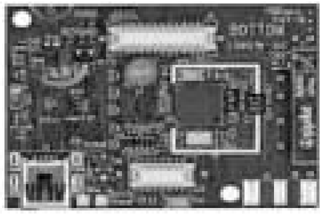
Keywords: sensor network, energy optimization, aggregate query, network lifetime, group aggregation, sensor database.

1 INTRODUCTION

In recent years, inevitable research trends have 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. These new sensors are differentiated from the traditional wireless devices in that their autonomous, tether-less nature makes users free



(a) The Top Side



(b) The Bottom Side

FIGURE 1
Intel mote 2.

from the concerns like configuration of network routes, recharging of batteries and tuning of parameters, etc [15]. An example of such small sensor devices is the *Intel mote 2* based on the modular design of the original Berkeley mote, consisting of an *ARM7* core processor, a *Bluetooth* radio, *RAM*, *FLASH* memory, a small battery pack and various *I/O* ports (see Figure 1), which runs on the top of the `TinyOS` operating system [6]. Such motes are small in physical size and can also be connected to *PC*s, *PDA*s, the *WWW*, and existing wired networks, given the gateway and interface products.

Due to commercial maturing on both hardware and operating systems[1], it is expected that the mote-based sensor networks will be widely deployed for the various applications in the near future including intelligent civilian surveillance, fine-grained natural habitat monitoring, and military object recognition. Recent advances in microelectronical technologies empower these sensor devices to monitor information at previously unobtainable resolutions [3]. For example, using these sensors, biologists are able to obtain the ambient conditions for endangered plants and animals every few seconds [16]. Security guards can detect the subtle temperature variation in storage warehouses to detect fire. Medicine manufacturers can maintain the strict requirements on the environmental parameters during product manufacturing. These inexpensive sensors gracefully exhibit their blatant monitoring superiority, particularly 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 sensors are capable of locating each other and routing data regardless of the network topology, thereby allowing the network topology to change as sensors move [13].

Wireless sensor networks whose sensors are powered by energy-limited batteries, however, have severe resource constraints on communication and

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

computation. First, the typical communication distances for low power wireless radios used in sensors 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 frequent packet dropping. 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 512*KB* storage. This restricts the complexity of data processing and the size of intermediate results stored on a sensor node [20]. Third, sensor nodes are powered by energy-limited batteries. The current commercial motes are usually equipped with a pair of *AA* batteries, and the lifetime range of them is from days to months or even years. For example, if a 2200-*mAh* pair of batteries is used naively on the 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, individual nodes even can achieve the lifetime in one year. In summary, the energy conservation issue is of paramount importance in a wireless sensor network, because a node failure, due to its battery exhaustion, can paralyze the entire network. Therefore, the *network lifetime* of a wireless sensor network is defined as the time of the first node failure [1].

1.1 Sensed Data and Queries

To meet various monitoring requirements, a large number of sensors are distributed to monitor the physical phenomenon. Each sensor node operates as an autonomous data source. The generated 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 to meet users requirements. Therefore, a sensor network can be treated as a distributed database system, where the data generated by each sensor node can be viewed as a horizontal fragment of a relational table, as shown in

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
Fragment of a relational table produced by Sensor no. 255.

Figure 2. Here we assume that during each time interval, a sensor node only produces a tuple. An attribute within a tuple is either the information about the sensor node itself (e.g., its identification or location) or the sensed data by the node (e.g., the temperature in the vicinity of the node). Note that in some complicated sensor network systems, multiple sensing functions need to be used for extensive information collection. For example, there are three different sensing functions employed in a sensor to monitor the temperature, light and humidity. 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 sensing functions. For simplicity, in this paper we only consider the sensor network database with one relational schema.

Having the above sensor database, how to interact with the database is an issue for each of its users. One popular way is to use *declarative queries*. Instead of inventing a new query language to express such queries, a SQL-style query language is adopted. That is, users simply specify their interested data from the network by posing SQL-like queries over the 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, which will be used to broadcast user queries throughout the network and collect the query results back to users. The semantics of the above query template is much 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 locations, detailed data processing and resulting data gathering, thus becomes a prevalent interaction approach between users and the sensor network.

1.2 In-Network Aggregation

To respond to an aggregate query, the network can process in either centralized or in-network manner. In the centralized implementation, all the data generated by sensor nodes are transmitted to the base station directly, and the interested data 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 communication for data transmission. By virtue of the autonomous, full-fledged computation ability of sensor nodes, it is possible to filter or aggregate data locally. Since the energy consumption in sensor networks is dominated by the wireless communication, the application will benefit in terms of energy efficiency by aggregating the data inside the network rather than simply transmitting them to the base station. Such *in-network aggregation* can dramatically decrease the number of massages transmitted through pushing the partial computation tasks from the base station into the intermediate nodes, so has been used as a paradigm in energy-constrained wireless sensor networks.

To process an aggregate query, in-network aggregation consists of *distribution* and *collection* phases [13]. In the distribution phase, a tree rooted at the base station and spanning all the sensor nodes is constructed. 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 aggregate result to its parent. The aggregate result will be relayed to the base station ultimately. Within a data aggregation session, each node in the routing tree will fall into a group, according to the distinct value of a list of Group-By attributes in the aggregate query. Messages from different nodes are merged into one message at each relay node in the tree if they are in the same group [18]. For example, if we issue a query "the average temperature in each building" for the sensor database that we mentioned above, each sensor node will first generate its own sensed data and collect data from its descendants in the routing 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 a single 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 significantly decrease.

1.3 Our Contributions

To prolong the network lifetime when dealing with aggregate query evaluation in wireless sensor networks, in this paper we first propose a heuristic algorithm for the problem of concern by introducing the node capability concept to balance the residual energy and the consumed energy at each node. We then present an improved algorithm for the problem by allowing group aggregation to reduce the energy consumption further. We finally conduct experiments by simulation. The experimental results show that the proposed algorithms outperform the existing ones in terms of performance metrics.

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 by taking group aggregation, `WHERE` and `HAVING` conditional clauses into consideration. Section 5 conducts extensive experiments to evaluate the performance of the proposed algorithms. Section 6 reviews related works, and 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 a region of interest. Each sensor periodically produces sensed 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. In other words, the network can be modeled as a directed graph $M = (N, A)$, where N is the set of sensor 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 data from u to v is modeled to be $md^{\alpha}_{v,u}$, 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 a sequence of aggregate queries, issued one by one at the base station, the problem is to evaluate each of the queries against the sensor network database by constructing a spanning tree rooted at the base station for the query with an objective to maximize the network lifetime. In other words, we aim to find an energy-efficient routing tree for each of the aggregate queries such that maximum number of queries can be answered before the first node failure in the network. 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 `Lifetime-maximized Network Configuration (LmNC)` for `LmRTP` based on the 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 $E_c(v, p(v)) = md^{\alpha}_{v,p(v)}$, 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 aggregate query. Assume that the length of data sensed by each node is identical, (i.e., m-bit), then the capability of node v to $p(v)$ is defined as

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

If node v in the routing tree has k descendants, then the energy consumption at v to forward all the messages (its own generated message and messages received from its descendants) to its parent $p(v)$ will be $(k+1)md^{\alpha}_{v,p(v)}$, given that there is no aggregation at v. If v will exhaust its residual energy after this transmission, then $E_r(v) = (k+1)md^{\alpha}_{v,p(v)}$. From Eq. (1), it can be seen that $k = E_r(v)/md^{\alpha}_{v,p(v)} - 1 = C(v, p(v))$. So, if there is no aggregation at v, the node capability of v, $E_c(v, p(v))$, actually indicates the maximum number of descendants that it can support by its current residual energy. In the following we propose a heuristic algorithm for the problem based on this concept. We assume that the algorithm will execute, once an aggregate query enters into the system.

3.2 Algorithm Description

The basic idea behind algorithm `LmNC` is as follows. Since a node with larger capability can have more descendants in the routing tree than others, it should be placed closer to the tree root to prolong the network lifetime. Based on this idea, a heuristic algorithm `LmNC` is proposed. The algorithm proceeds as follows. Initially, the routing tree only contains the base station s only. Each time a node with maximum capability is added to the routing tree. Thus, the nodes are added one by one until all of them are included into the tree. The motivation behind this algorithm is that adding the node with 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 routing 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 description of the proposed algorithm is given below.

Algorithm LmNC (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 are not yet 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 $C_{\max} \leftarrow C(v, u)$;
                $added\_node \leftarrow v$;
                $temp\_parent \leftarrow u$;
        endif;
    endfor;
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. $Q \leftarrow Q - \{added\_node\}$;
endwhile;

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 the one in [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 is a partially built routing tree and a number of nodes are to be added to the tree. Further assume that node v_i has the maximum residual energy among those nodes that are 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 next tree construction because of the enormous transmission energy consumption from v_i to its parent. Therefore, in spite of that v_i has more residual energy than v_j at the moment, the maximum number of the messages transmitted by v_i is less than that by v_j. It thus derives 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 of 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 further. We also pursue the further improvement by taking into account conditional clauses during the design of the heuristic algorithm.

4.1 Algorithm Description

Since group aggregation is able to combine the messages from the same group into one message, clustering the nodes in 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 [18]) to construct an energy-efficient routing tree for aggregate queries. Following the same spirit, we incorporate this heuristic into algorithm `LmNC`, an improved algorithm `LmGaNC` is then proposed as follows.

Algorithm LmGaNC (G, E_r) /* G is the current sensor network and E_r is an array of the residual energy of nodes */
begin

1. $V_T \leftarrow \{s\}$; /* add the base station into the tree */
2. $Q \leftarrow V - V_T$; /* the set of nodes that are not in the tree yet*/
3. while $Q \neq \emptyset$ do
4. $C_{\max} \leftarrow 0$; /* the maximal node capability 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 $C_{\max} \leftarrow C(v, u)$;
                $added\_node \leftarrow v$;
                $temp\_parent \leftarrow u$;
        endif;
    endfor;
8. $p(added\_node) \leftarrow temp\_parent$;
    /* set the parent for the node with maximum capability */
9. $d_{min} \leftarrow \infty$; /* the minimum distance from the node to its parent */
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 $(d_{min} > d_{added\_node,u'})$
              then $d_{min} \leftarrow d_{added\_node,u'}$;

$p(added\_node) \leftarrow u'$;
endif;
endfor;
12. $V_T \leftarrow V_T \cup \{added\_node\}$;
/* add the node with maximum capability into the tree */
13. $Q \leftarrow Q - \{added\_node\}$;
endwhile;

end.

Algorithm `LmGaNC` is essentially similar to algorithm `LmNC`, the only difference is that during the construction of the routing tree, a child node with maximum capability chosen by `LmNC` will check whether there is another node in the routing tree that is in the same group with the child node, and this node is referred to the *better parent* of the child node. If yes, the child node is allowed to switch to this better parent. In case 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 node will cause the extra transmission energy consumption. To avoid this, the *distance factor* (df) concept here is employed, which is the upper bound of the distance between a child node and its chosen parent. For example, if df =1.5, then only those potential parents whose distances to the child node are at most $df * d_{v,u} = 1.5d_{v,u}$, where $d_{v,u}$ is the distance between a child node v and its current parent u.

We use the following example (see Figure 3 to illustrate energy reduction brought by algorithm `LmGaNC`. We assume that there is a partially built routing tree with node 1 as the tree root (see Fig. 3a). We further assume that black nodes 2 and 7 are in Group 1, the shaded node 6 is in Group 2, and the rest of the nodes are in Group 3. The number of messages is shown in the figure (depending on the number of various groups in the subtree). Now, following algorithm `LmNC`, we add a new node to the tree. We assume that a shaded node 8 is in Group 2 and 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 other nodes in the path from node 8 to the root except the root node itself have to consume extra energy for the transfer of this message, because node 8 is in a group different from the other nodes in the path. However, apply the improved algorithm `LmGaNC` to build the routing tree, node 8 can switch to a better parent that is in the same group as itself (Group 2). Therefore, node 8 will choose node 6 as its new parent, assuming without violation of the distance factor (see Fig. 3(c)). 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 defined as the maximum number of descendants that the node can have by its current residual energy without

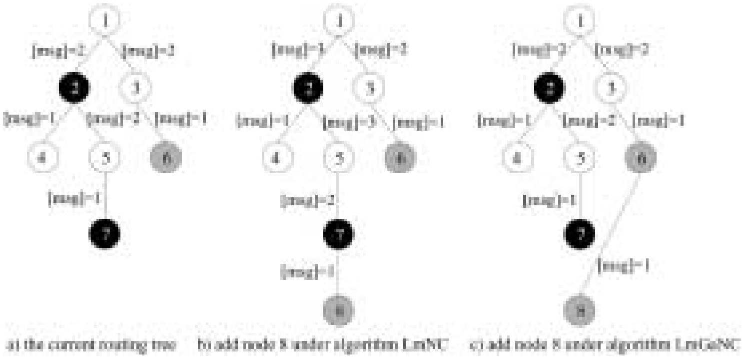


FIGURE 3
Illustration of energy savings brought by algorithm LmGaNC.

data aggregation. In other words, the capability of a node in the routing tree actually indicates the maximum number of messages that it can forward for its descendants. In algorithm LmGaNC, a node capability indicates the number of descendants that are in different groups from the node, rather than the total number of its descendants, because the descendents messages from the same group will be merged into one message after group aggregation.

4.2 Further Improvement

In the following we show how to further achieve energy efficiency delivered by algorithm LmGaNC through the consideration of the impact of WHERE and HAVING conditional clauses on aggregate queries.

4.2.1 Effect of WHERE Clause

To evaluate a complicated query, it is very common to use some conditions or thresholds to filter out the unwanted immediate results as early as possible during the query evaluation. 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 divide the query results into a set of room groups and calculate the average temperature for each room group. However, to exclude the room nodes that are not at level 5, we employ the WHERE clause originated from the SQL language to further reduce the energy consumption. 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 conditional specifications. If it is mismatched, 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 [20], where each node holds a waiting list for its children and will not transmit its aggregated result to its parent until it hears from all its children. Since the size of the message is shrunk into only 1 bit given the mismatch, the transmission energy consumption will be further reduced.

However, even if a node matches the conditional specifications in the query, whether its residual energy can afford the message transmission is still in doubt. 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 to its parent to indicate its insufficient residual energy.

4.3 Effect of HAVING Clause

Note that the WHERE clause can be used in both aggregate and non-aggregate queries. If it is imposed on an aggregate query, it will be considered upon all the sensed data locally before group aggregation. While the 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 the HAVING clause is centrally used at the base station to exclude the undesired groups after 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. For some special cases, however, the 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 degree Celsius?, the 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 degree Celsius and the whole group does not satisfy the clause HAVING MIN(Temperature) $>$ 20. Thus, when a node detects its local data does not satisfy the above condition, it can cancel its message transmission and notify other nodes in its group to suppress their transmissions.

5 SIMULATIONS

In this section we evaluate the performance of the proposed algorithms LmNC and LmGaNC against existing algorithms including MST (Minimum Spanning Tree), SPT (Shortest Path Tree) and GaNC. The experimental metrics are the network lifetime, the total network energy consumption, and energy consumption per each query, based on different number 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. The sensor nodes are distributed within a $100 \times 100\ m^2$ region. Each sensor node is initially equipped with 10^5 *μ-Joules* energy. Two nodes are only connected when their residual energy can support their transmissions to each other directly. For each aggregate query, we randomly assign an integer $\texttt{Group\_id} = i$ for each node, $1 \leq i \leq n$, where n is the number of groups in the query. A routing tree for each query is constructed 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. The value of network lifetime or energy consumption in each figure is the average of the network lifetime and energy consumption from 30 distinct network topologies for each network size.

5.1 Performance of the Two Proposed Algorithms

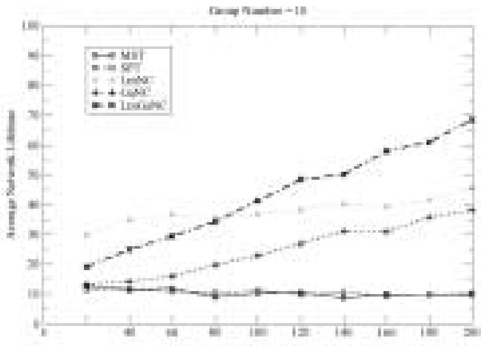
We reproduce an existing heuristic algorithm called `GaNC` [18] for the concerned problem, which will be used as the benchmark. Algorithm `GaNC` with the group aggregation concept is derived from a simple `First-Heard-From (FHF)` protocol, where the nodes always select the first node from which they 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` is able to switch to a better parent in the same group as itself, assume that the distance factor is met.

If we ignore the effect of the distance factor by setting $df = 100$, 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`. Fig. 4(b) indicates that the total energy consumption delivered by algorithm `LmGaNC` is strictly less than that of algorithm `LmNC`. Fig. 4(c) clearly shows that the average energy consumption for each query by algorithm `LmGaNC` is significantly less than that by algorithm `LmNC`, which means more queries can be answered if algorithm `LmGaNC` is applied, given the same amount of energy over the network.

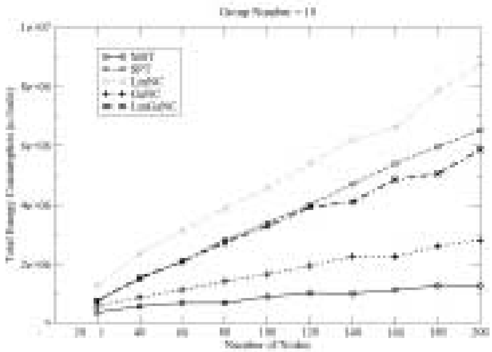
5.2 Sensitivity to the Number of Groups

Figure 5 indicates, when the number of groups is decreased from 10 to 5, both algorithms manifest their lifetime improvements. The reason behind is as follows.

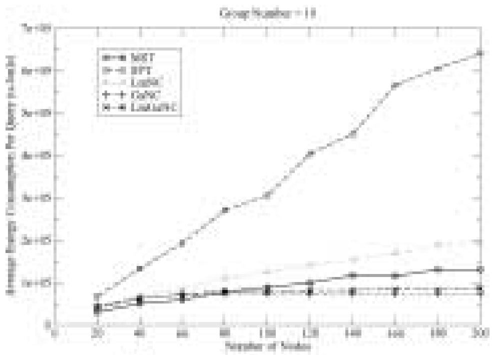
On one hand, fewer groups mean that more sensor nodes will be in the same group, and thus the possibility of message suppression under group aggregation will be enhanced. On the other hand, fewer groups make a



(a) Network Liftime



(b) Total Energy Consumption



(c) Average Energy Consumption Per Query

FIGURE 4
Performance comparison among various algorithms.

child node have more chances to switch to a better parent in its group, so less transmission energy will be consumed. The distance factor here is also set to be 100.

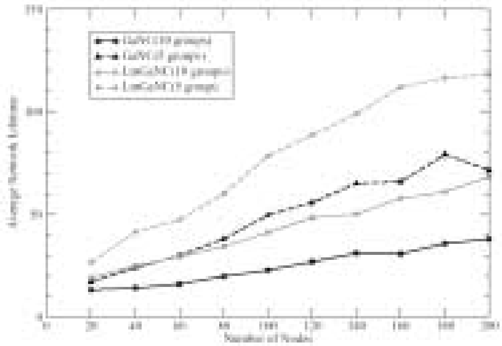


FIGURE 5
LmGaNC vs GaNC in different number of groups.

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 its 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 implies that when distance factor is decreased from 3 to 1.5, algorithm `LmGaNC` exhibits its sensitivity to the distance factor immediately and the network lifetime is significantly prolonged, while the response of algorithm `GaNC` to this is very insensitive.

5.4 Sensitivity to the `WHERE` Condition Clause

The experiments here are aimed 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 instead of the sensor data

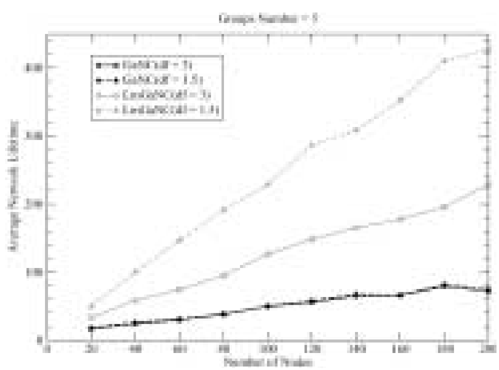
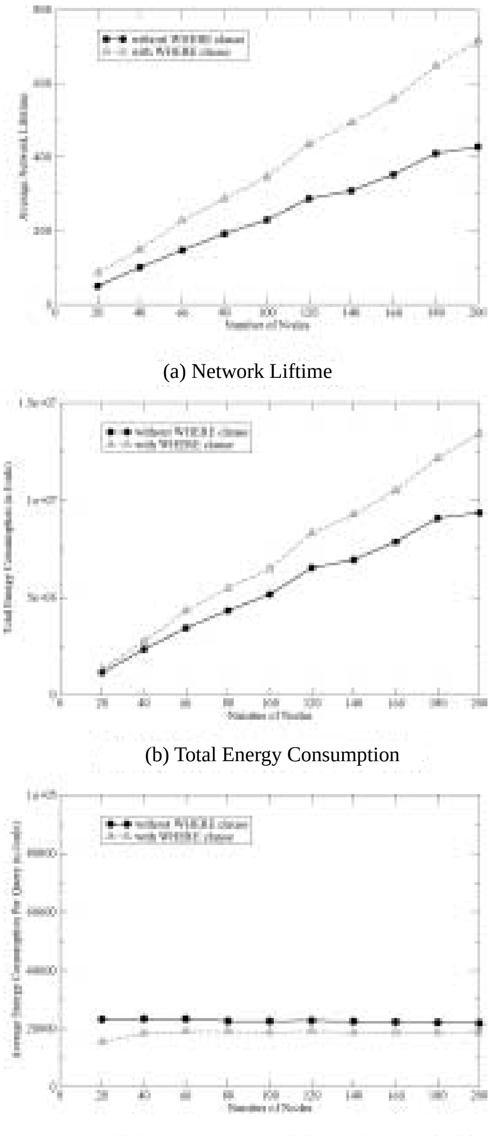


FIGURE 6
LmGaNC vs GaNC in different distance factors.



(a) Network Liftime

(b) Total Energy Consumption

(c) Average Energy Consumption Per Query

FIGURE 7
Performance of `LmGaNC` with `WHERE` clause.

to their parents. 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.

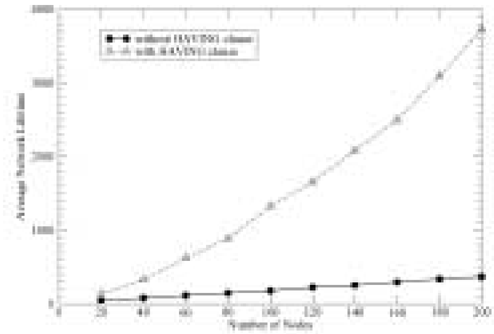
The experimental results show that the network lifetime increases by more than 50% while the total energy consumption only goes up by around 25% under algorithm LmGaNC, which implies that the average energy consumption per query is actually reduced through the employment of the WHERE clause as shown in Fig. 7(c).

5.5 Sensitivity to the HAVING Condition Clause

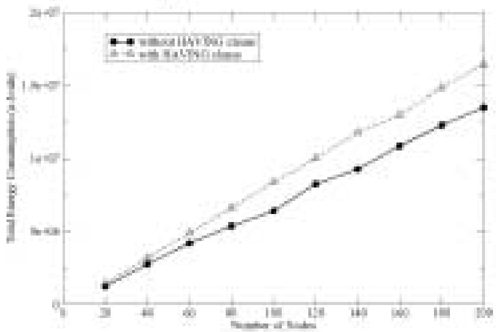
Most HAVING clauses in an aggregate query do not contribute to energy savings during the query evaluation, except for MAX and MIN predicates. For the conditions like MIN(attribute) $<$ Threshold and MAX(attribute)$>$Threshold, the mismatched messages can be suppressed locally as the WHERE clause. While 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. 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 improvements on the network lifetime and the energy consumption are shown in Figure 8.

6 RELATED WORK

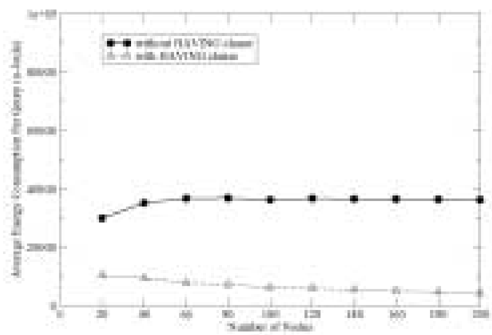
To prolong the network lifetime through improving the energy efficiency of network operation, several protocols for various routing problems in both ad hoc networks and sensor networks have been proposed recently [1, 2, 4, 7, 8, 9, 11, 19]. For example, in ad hoc networks Chang and Tassiulas [1, 2] realized 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 the nodes in a sensor network are grouped into a number of clusters. Within a cluster, a chosen node as the cluster head will be 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 [19] provided a protocol PEDAP for the data gathering problem, which constructs a minimum spanning tree (MST) rooted at the base station to limit the



(a) Network Liftime



(b) Total Energy Consumption



(c) Average Energy Consumption Per Query

FIGURE 8
Performance of `LmGaNC` with `HAVING` clause.

total energy consumption. Kalpakis *et al* [8] considered a generic data gathering problem and proposed an integer program solution and a heuristic solution.

This paper provides the evaluation of a sequence of aggregate queries in a wireless sensor network with an objective to maximize the network lifetime. The pervasive way to evaluate an aggregate query is to apply in-network aggregation, which has been addressed in [12, 21], 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 [14] to save transmission energy further. In this case, 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 [18], where the sensor nodes in the same group are clustered along the same routing path with the goal to reduce the size of transmitted data, thereby saving energy. However, an obvious indiscretion in some of the routing protocols like `MST` and `GaNC`, is that a chosen node is added to the routing tree without the consideration of its residual energy during the construction of the routing tree. As a result, the nodes near to the tree root will exhaust their energies rapidly, since they serve as relay nodes to relay messages for their descendants. Thus, the network lifetime will be shortened.

7 CONCLUSIONS AND FUTURE WORK

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 for the problem, then presented an improved algorithm by incorporating group aggregation and conditional clauses to further reduce the energy consumption. We finally conducted experiments to evaluate the performance of proposed algorithms against those of existing ones. The experimental results showed that the proposed algorithms outperform the existing algorithms significantly. In contract to the existing algorithm, although the proposed algorithms promise to prolong the network lifetime significantly, these algorithms are centralized algorithms and the costs associated with them are a longer running time and more energy consumption to construct the routing tree. As for future work, we intend to explore the possibility of its distributed implementation to make the proposed protocols practical.

8 ACKNOWLEDGMENT

We would like to thank the anonymous referees for their constructive comments and suggestions which have helped improve the quality and presentation of the paper. Meanwhile, we would like to acknowledge that this work was partially supported by a research grant from the Faculty of Engineering and Information Technology at the Australian National University.

REFERENCES

[1] Chang, J-H. and Tassiulas, L. (2000). Energy conserving routing in wireless ad hoc networks. *Proc. of INFOCOM'00*, IEEE.

[2] Chang, J-H. and Tassiulas, L. (2000). Fast approximate algorithms for maximum lifetime routing in wireless ad hoc networks. *IFIP-TC6/European Commission Int'l Conf.*, Lecture Notes in Computer Science, *1815*, 702–713, Springer.

[3] Gehrke, J. and Madden, S. (2004). Query processing in sensor networks. *ComSoc'04*, IEEE.

[4] Heinzelman, W. R., Chandrakasan, A. and Balakrishnan, H. (2000). Energy-efficient communication protocol for wireless microsensor networks. *Proc. of 33th Hawaii International Conference on System Sciences*, IEEE.

[5] Heidemann, J., Silva, F., Intanagonwiwat, C., Govindan, R., Estrin, D. and Ganesan., D. (2001). Building efficient wireless sensor networks with low-level naming. *Proc. of 18th Symposium on Operating Systems Principles (SOSP'01)*, ACM.

[6] Hill, J., Szewczyk, R., Woo, A., Hollar, S. and Pister, D. C. K. (2000). System architecture directions for networked sensors. *Proc. of ASPLOS*, ACM.

[7] Intanagonwiwat, C., Estrin, D., Govindan, R. and Heidemann, J. (2002). Impact of network density on data aggregation in wireless sensor networks. *Proc. of 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, IEEE.

[8] Kalpakis, K., Dasgupta, K. and Namjoshi, P. (2003). Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computer Networks*, *42*, 697–716.

[9] Kang, I. and Poovendran, R. (2003). Maximizing static network lifetime of wireless broadcast ad hoc networks. *Proc. of ICC'03*, IEEE.

[10] Liang, W. (2002). Constructing minimum-energy broadcast trees in wireless ad hoc networks. *Proc. of MOBILHOC'02*, ACM.

[11] Lindsey, S. and Raghavendra, C. S. (2002). PEGASIS: Power-efficient gathering in sensor information systems. *Proc. of Aerospace Conference*, IEEE, 1125–1130.

[12] Liu, J., Zhao, F. and Petrovic, D. (2003). Information-directed routing in ad hoc sensor networks. *Proc. of 2nd international conference on wireless sensor networks and applications*, ACM.

[13] Madden, S., Franklin, M. J., Hellerstein, J. and Hong, W. (2002). TAG: a tiny aggregation service for ad-hoc sensor networks. *Proc. of 5th symposium on Operating systems design and implementation*, ACM.

[14] Madden, S., Franklin, M. J., Hellerstein, J. and Hong, W. (2003). The design of an acquisitional query processor for sensor networks. *Proc. of SIGMOD'03*, ACM.

[15] Madden, S., Franklin, M. J., Hellerstein, J. M. and Hong, W. (2005). TinyDB: An acquisitional query processing system for sensor networks. *ACM Transaction on Database Systems*, *30*, 122–173.

[16] Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D. and Anderson, J. (2002). Wireless sensor networks for habitat monitoring. *Proc. 1st ACM International Workshop on Wireless Sensor Networks and Applications*, ACM, 88–97.

[17] Manjeshwar, A. and Agrawal, D. P. (2002). APTEEN: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. *Proc. of IPDPS'02*, IEEE.

[18] Sharaf, M. A., Beaver, J., Labrinidis, A. and Chrysanthis., P. K. (2004). Balancing energy efficiency and quality of aggregate data in sensor networks. *J. VLDB*, Springer.

[19] Tan, H. Ó. and Kórpeoğlu, İ. (2003). Power efficient data gathering and aggregation in wireless sensor networks. *ACM SIGMOD Record*, *32*, 4, 66–71.

[20] Yao, Y. and Gehrke, J. (2003). Query processing in sensor networks. *Proc. of 1st Biennial Conf. Innovative Data Systems Research (CIDR'03)*, ACM.

[21] Zhao, F., Shin, J. and Reich, J. (2002). Information-driven dynamic sensor collaboration. *IEEE Signal Processing Magazine*, *19*.