Data Redundancy Aware Topology Control in Wireless Sensor Networks *

Hao Han
State Key Laboratory for Novel Software Technology
Nanjing University, China
hanhao@dislab.nju.edu.cn

Tao Gu
Institute for Infocomm Research
Singapore
tgu@i2r.a-star.edu.sg

Sanglu Lu
State Key Laboratory for Novel Software Technology
Nanjing University, China
sanglu@nju.edu.cn

Abstract

Prolonging the lifetime of wireless sensor networks remains the most challenging design requirement in unattended wireless sensor network applications. Studies show that considerable energy can be saved by turning certain nodes into inactive mode. In this paper, we propose a novel topology control algorithm which takes into account of both data redundancy and network connectivity. Compared to other previous work, the algorithm inactivates a set of redundant nodes that can be delegated by others in the premise of network connectivity without losing useful information of an environment. We conduct simulations and our results show that our algorithm can reduce node redundancy by 40% maximum in high data redundancy scenarios.

1. Introduction and Motivation

Recent advances in wireless communications and electronics have enabled the rapid development of large-scale wireless sensor networks for applications in various areas, such as environment monitoring and military surveillance, etc. Saving energy is the most important requirement in the design of any wireless sensor network protocol or application as most of sensor nodes are battery operated in an unattended manner. Replacing batteries is not only costly, but also impossible in many situations such as hard-to-reach places. Previous studies show that, in wireless sensor networks, communication still spends much energy which is greater than computation and storage cost by nearly two orders of magnitude. Furthermore, sending and receiving data are both energy-consuming (in the same order of magnitude) [6]. Therefore, turning some nodes' radio into sleep mode at given time is an efficient technique to save energy and extend the network lifetime. However, which nodes should be turned off (or "inactive")? The node selection process needs to be carried out based on certain metrics. Previous work showed that coverage and connectivity are two important metrics [4]. Node communication redundancy [2] is also an important metric, which indicates that nodes sharing a high percentage of nodes in their neighbor sets can be turned off without loss of connectivity. However, none of them concerns about data redundant. Actually, those inactive nodes may provide the information of an environment which should not be ignored. The selection of which node can go to sleep should consider data distribution.

We have two observations in wireless sensor networks applications:

- Observation 1: Wireless sensor networks are high-density networks. The number of operational sensor nodes in a region usually is much higher than the need in most applications because of random deployment, and limitations of existing optimal sensor placement techniques. Without certain nodes, the network can still retain connectivity.
- Observation 2: As a result of Observation 1, there exists a large number of redundant data in wireless sensor networks, especially in high-density regions. Nodes in the vicinity often produce very similar data, and hence inactivating some of these nodes will not lose any useful information.

Motivated by the above two observations, in this paper, we propose a novel topology control algorithm which aim to eliminate unnecessary nodes according to data redundancy and retain network connectivity.

The most related work done by Y. Kotidis [5], with a similar approach but a different objective, demonstrates how to construct a small set of representative nodes in the network to provide quick approximate to snapshot queries. In that paper, the authors select certain nodes to delegate others according to data distribution, which is similar to our approach, but the main differences are: (1) we find long-term redundancy instead of temporary similarity; (2) our objective is to simplify topology, therefore we need to consider network connectivity as we turn off redundant nodes. In [5], the delegated nodes do not go to sleep, and they only ignore

* This work is partially supported by the National High-Tech Research and Development Program of China (863) under Grant No. 2006AA01Z199; the National Natural Science Foundation of China under Grant No.60402027; Jiangsu Natural Science Foundation under Grant No. BK2005411.

user queries; and (3) in our algorithm, the nodes can delegate others beyond multi-hops as long as they are redundant not only neighbors.

In summary, the contributions of this paper focus on: (1) combining both data redundancy and network connectivity in wireless sensor networks for topology control; and (2) a novel energy-efficient distributed node reduction algorithm to eliminate redundant nodes while maintaining four-connectivity.

2. System Model

Before we discuss our algorithm in detail, we first describe the system model and our assumptions as follows: Let n senor nodes be randomly distributed over a fixed two dimensional geographic area, and the network is modeled as an undirected graph, $G = (V, E)$, where $V(|V| = n)$ represents the set of senor nodes and E denotes the set of edges where an edge exists between two nodes if and only if the distance is not beyond the maximal radio range. In such a model, each node, $v_i \in V$, knows the coordinates (which can be obtained through an internal device like GPS or any other locating systems), and has a unique ID (which can be computed by coordinates). Using this system model, our problem is then formulated as how to find the subset of $V'(V' \subseteq V)$, where $\forall v_i \in V - V' \; \exists v_j \in V'$, that v_i can be delegated by v_j, and the subgraph $G' = (V', E')$ retains connectivity.

3. Algorithm Overview

Our algorithm consists of four steps. In **Step 1**, we divide the network into many virtual grids, and each node performs computation locally based on its coordinators to decide which cell it belong to. The side length of square cell, r, must satisfy the relation (1), where R is the maximum of radio range of a sensor node:

$$r^2 + (2r)^2 \leqslant R^2 \Rightarrow r \leqslant \frac{R}{\sqrt{5}} \tag{1}$$

The above relation ensures the nodes in adjoining cells can communicate with each other directly. If guaranteed that there is at least one node in each cell after node selection, the network can be at least four-connective, which possesses good fault-tolerance. Even some nodes fails in such a network, we can still achieve network connectivity.

In **Step 2**, to reduce large energy consumption by broadcasting raw data, we build a regression model to capture the feature of data distribution in each node. For example, if a degree-two polynomial model: $f(t) = w_0 + w_1 t + w_2 t^2$ which measures an attribute (e.g, temperature) fits actual values well at any time, we only need to transmit the coefficients $\overline{w}$ instead of large amount of data. Formally, assuming a node collects a set of values, $f(t_1), f(t_2), ..., f(t_m)$ within the learning time T, and given a set of basic functions $H = (h_1, h_2, ..., h_k)$ $(e.g, 1, t, t^2)$, we would like to match the function value to the real value, that is, to find the best

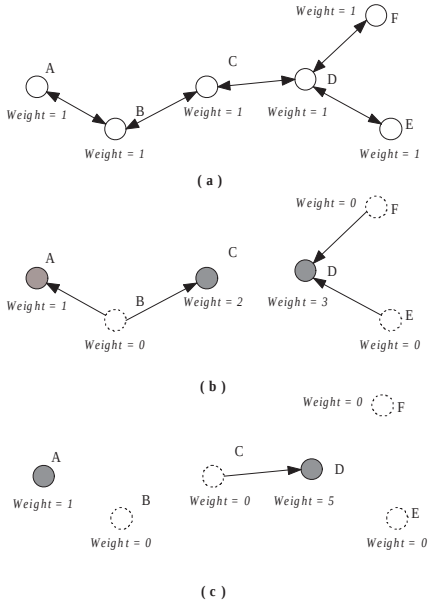

Figure 1. An example of **Step 3** in the algorithm

basic function coefficients, $\overline{w^*} = (w_1, ..., w_k)$, such that:

$$\overline{w^*} = \arg\min_{w} \|\widehat{f} - f\| \tag{2}$$

where parameters $\overline{w^*}$ minimize the root mean squared error, and $\widehat{f}(t) = \sum_i w_i h_i(t)$, $(t \in [t_1, t_m])$. In practice, we can use linear regression (detail algorithm can be found in [3]) to calculate $\overline{w^*}$.

After learning data distribution, we move on to the kernel **Step 3**: discovery data redundancy and inactivate unnecessary nodes. In this step, each node v_i maintains a list: $Redu\_list_i$ which contains all nearby nodes having very similar data distribution, and a $weight_i$ which indicates how many nodes it can delegate. It should be noticed that the weight is very important in ensuring the accuracy of user queries. Without such a weight, the result may be totally different, especially in an aggregation function like AVERAGE etc. Initially, $Redu\_list_i = \{v_i\}$ and $weight = 1$. Then v_i broadcasts $\overline{w^*}$ with its remaining energy and ID. The node v_j that receives this message compares it with own coefficients. If the relative error does not exceed the threshold, it adds v_i to $Redu\_list_j$. Upon receiving all these messages from neighbors or exceeding the maximal waiting time, v_j begins to determine whether to go to sleep. In practice, we must avoid the case that two nodes decide to turn off at the same time when they thought each other as the delegator. Our algorithm uses remaining energy and unique ID to break ties (any other tie-breaking policy would work as long as both nodes can make a consistent decision). Concretely, the nodes who find that they are not the only node in cells and have the lowest energy (if equal, use ID instead) in local $Redu\_lists$, will send their requests to the highest nodes for delegation. On receiving the ACK messages back (otherwise they go

on sending the request to the second highest), they will go to sleep mode and inform all the nodes in their lists. Then the nodes which receive this message will delete them from their $Redu\_lists$, and the delegators also need to increase their weights besides that.

To illustrate **Step 3** of our algorithm, we describe an simple example here. Considering six nodes having similar data distribution as shown in Fig.1, we assume that: (1) each node can only directly communicate with adjoining nodes; (2) all of them can be turned off because there exist other nodes not drawn in their cell; (3) and the energy level from high to low is $D \geqslant C > A > B \geqslant E \geqslant F$. After broadcasting the probe messages, the $Redu\_lists$ of every nodes change to:

$$Redu\_list_A = \{A, B\} \quad Redu\_list_B = \{A, B, C\}$$
$$Redu\_list_C = \{B, C, D\} \quad Redu\_list_D = \{C, D, E, F\}$$
$$Redu\_list_E = \{D, E\} \quad Redu\_list_F = \{D, F\}$$

Then node B, E, and F send individual delegation messages to C and D with their $weights$. Upon receiving the messages ACK, they inform neighbors, and then turn to sleep mode. Now only node A, C, D are left (see Fig. 1(b)), whose $Redu\_lists$ are:

$$Redu\_list_A = \{A\} \quad Redu\_list_C = \{C, D\}$$
$$Redu\_list_D = \{C, D\}$$

Followed by the same procedure, only A and D remain at last (in Fig. 1(c)).

Through three steps aforementioned, large amount of redundant nodes can be turned off. In **Step** 4, we use time-sharing between active and inactive nodes to further prolong the network lifetime. In this step, MAX_SLEEP_TIME is set for each node. On exceeding this threshold, sleeped nodes will become active again, and a new round of selection is initiated. This time, they are very likely to delegate others because they conserved much energy during their sleep.

4. Evaluation

We develop a wireless sensor network simulator based on peerSim[1] to evaluate the performance of our algorithm. The parameters are as follows: the size of monitored region is 400*400 m; radio range is 50 m; the number of measured attributes is 1; sample ratio is 1 tuple per 30 second; relative error of $\overline{w^*}$ is 0.01; and learning time T is 1 hour. In addition, similar to other evaluation methods in wireless sensor network, we do not consider the link-errors among nodes.

Since the algorithm is sensitive to the actual data distribution, we evaluate the percentage of inactive nodes under three different data distributions: **EQUAL** (all nodes produce the same value), **RANDOM** and **GAUSSIAN** (two dimensional gaussian distribution generated by function *peaks* in MATLAB). Fig. 2 shows that: (1) the performance under EQUAL distribution is the best, next is GAUSSIAN, and the worst is RANDOM. It is caused by different data redundancy; (2) the more the number of nodes in the region is, the better the performance of our algorithm achieves. The reason is that each node has more neighbors to select. The above experimental result suggests that our topology control algorithm will perform better in the scenarios with high data redundancy or dense node deployment.

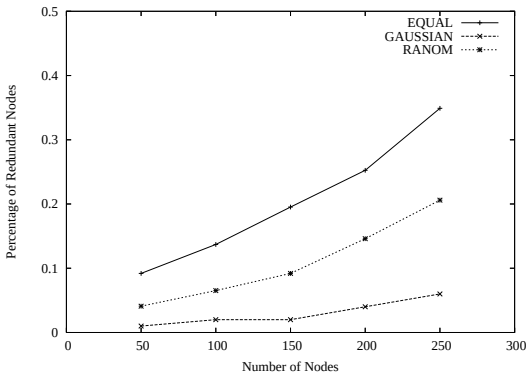

Figure 2. Performance Comparison with Different Data Distributions

5. Conclusions and Future work

This paper proposes a distributed energy saving scheme considering both data redundancy and network connectivity in wireless sensor networks. Our preliminary results demonstrate that our algorithm can reduce node redundancy by 40% maximum. In future work, we plan to adjust the length of learning time dynamically instead of giving a fixed time in advance. We are also interested in the case how to maintain the k-connectivity without knowing the node location.

References

[1] *http://peersim.sourceforge.net/.*

[2] S. Al-Omari and W. Shi. Redundancy-aware topology management in wireless sensor networks. In *CollaborateCom*, pages 1–10, 2006.

[3] C. Guestrin, P. Bodík, R. Thibaux, M. A. Paskin, and S. Madden. Distributed regression: an efficient framework for modeling sensor network data. In *IPSN*, pages 1–10, 2004.

[4] R. Iyengar, K. Kar, and S. Banerjee. Low-coordination topologies for redundancy in sensor networks. In *MobiHoc*, pages 332–342, 2005.

[5] Y. Kotidis. Snapshot queries: Towards data-centric sensor networks. In *ICDE*, pages 131–142, 2005.

[6] G. Mathur, P. Desnoyers, and etc. Ultra-low power data storage for sensor networks. In *IPSN*, pages 374–381, 2006.