Front. Electr. Electron. Eng. China 2008, 3(1): 34–39
DOI 10.1007/s11460-008-0006-6

RESEARCH ARTICLE

Hong LUO, Fangchun YANG, Yonghe LIU

A distributed routing algorithm for data aggregation in wireless sensor networks

Abstract Considering the impact of aggregation cost on the performance of aggregation routes in wireless sensor networks, an aggregation-decision-based distributed routing algorithm for data aggregation is proposed. When source nodes arrive or leave, the algorithm can calculate the aggregation benefit according to data correlation, aggregation cost and transmission cost. Then the algorithm will adaptively make aggregation and routing decisions based on aggregation benefit. Therefore, it can jointly optimize the aggregation and transmission costs and reduce the energy consumption for data gathering. This distributed algorithm makes all the decisions only relying on the local information. Hence, the routing maintenance cost is limited. Simulation results show that the energy consumption difference between this distributed online algorithm and the previous offline one is within 17% under any network conditions.

Keywords wireless sensor networks, data gathering, data aggregation, routing

1 Introduction

One important research issue in wireless sensor networks is how to reduce the energy consumption over the network [1,2]. In a densely deployed sensor network, the information gathered from neighboring nodes is highly correlated. Benefiting from this feature, data aggregation (or data fusion) strategy is widely used in such networks to reduce the transmission load in the network, decrease the energy consumption on each node along their routes, and hence prolong the network lifetime [1]. In literature, routing algorithms regarding where and when aggregation shall be performed along the routes have been explicitly or implicitly studied extensively [3–5]. However, the existing solutions focus on how to reduce transmission cost by using data aggregation, thereby omitting one important dimension in the optimization space, namely, the aggregation cost. In the future, more and more applications will tend to introduce complex information, such as image, multimedia information and encryption data, into sensor networks for special targets. Usually, aggregation algorithms for these kinds of data are complex and need more power for data aggregation processing. In some cases, the cost for data aggregation processing is on the same order as the transmission cost. Therefore, performing data aggregation at each node along the transmission paths may not be the most economical method. Adaptive fusion Steiner tree (AFST) proposed in Ref. [6] is a routing algorithm which can jointly take into account both the transmission and aggregation costs and adaptively adjust the aggregation points. However, the AFST is an offline solution that entails centralized aggregation decision and routing computation at the sink node and needs plenty of communication to disseminate the results to source nodes. Therefore, it cannot be readily implemented in a real network with numerous nodes and inconstant structure.

In order to solve the problems of aggregation decision and routing decision resulted from aggregation cost, we propose an aggregation-decision-based distributed routing algorithm, which jointly optimizes over both transmission and aggregation costs. When source nodes arrive or leave, the algorithm can determine the data gathering routes and the aggregation locations according to data correlation, aggregation cost and transmission cost. It can also be implemented by individual sensor nodes in a distributed fashion relying on local information only. Therefore, it is very practical when considering the dynamics of sensor networks.

Translated from *Journal of Beijing University of Posts and Telecommunications*, 2007, 30(1): 9–13 [译自: 北京邮电大学学报]

Hong LUO (✉), Fangchun YANG
Beijing Key Laboratory of Intelligent Telecommunication Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing 100876, China
E-mail: luoh@bupt.edu.cn

Yonghe LIU
Department of Computer Science and Engineering, University of Texas at Arlington, Arlington TX76019, USA

The remainder of this paper is organized as follows. In Sect. 2, we will describe the system model and formulate the aggregation routing problem. Section 3 describes the optimal access problem and the relative algorithm. Section 4 will present in detail the design of the proposed distributed online AFST algorithm. In Sect. 5 we experimentally study the performance of the online algorithm. Finally Sect. 6 concludes the paper.

2 System model

A sensor network is modeled as a graph $G = (V, E)$ where V denotes the node set and E denotes the edge set that represents the communication links between node-pairs. We assume a set $S \subset V$ of k nodes to be data sources of interests and the sensed data need to be reported periodically to a special sink node $t \in V$.

We define node weight $w(v)$ as the amount of information outgoing from v. An edge e is denoted by $e = (u, v)$, where u is the start node and v is the end node. The weight of edge e is equivalent to the weight of its start node, i.e., $w(e) = w(u)$. We abstract the unit transmission cost on edge e as $c(e)$ and the unit aggregation cost on it as $q(e)$. Thus the transmission cost $t(e)$ of edge e is given by

$$t(e) = w(e)c(e). \tag{1}$$

If node v is responsible for aggregating node u's data with its own, the weight of node v will change before and after aggregation. Let $\tilde{w}(v)$ and $w(v)$ denote the data amount of node v before and after aggregation, $x_{uv} \in \{0, 1\}$ denote whether aggregation occurs on edge $e = (u,v)$, and ρ_{uv} denote the correlation coefficient representing the data redundancy between nodes u and v. Then, we can summarize the aggregation function at node v as:

$$w(v) = \tilde{w}(v) + w(u)(1 - x_{uv}\rho_{uv}), \tag{2}$$

and the aggregation cost $f(e)$ of edge e is given by

$$f(e) = q(e)(w(u) + \tilde{w}(v)). \tag{3}$$

If one node has more than one child node, the aggregation process at the parent node (fusion point) is performed step by step (aggregating with nodes in turn). In the step by step aggregation method, the fusion point aggregates its own data with one input first, and next fuses the temporary aggregation result with another input. This process will be repeated until all the inputs are aggregated. The number of children nodes is called aggregation depth. Hence, the above Eqs. (2) and (3) are adequate in describing the aggregation process.

Mathematically, the solution of adaptive data aggregation and routing problem is to find a connected subgraph $G^* = (V^*, E^*)$, which includes all source nodes and the sink, so as to minimize the value of Eq. (4).

$$\sum_{e \in E_f^*} (f(e) + t(e)) + \sum_{e \in E_n^*} t(e), \tag{4}$$

E_f^* and E_n^* are two disjoint edge subsets. E_f^* is the edge set including all edges on which aggregation is performed and E_n^* is the set of edges on which aggregation is not performed. This is a NP-complete problem [3]. Although AFST-offline proposed in Ref. [6] cannot be readily implemented in real environments, it provides significant insight to the impact of aggregation cost and the construction of routing tree. Besides, it also provides comparison reference to the study of distributed online version. In this paper, we improve it to a distributed online algorithm (AFST-online) aiming at processing source nodes' arrival and departure.

AFST-online will solve two problems.

1) When a new source node joins the network, it must find the optimal access point in the existing tree to establish a connection. By using this connection, the new arrival node can access to the aggregation routing tree, and the structure of the original tree should not be changed. The access method used here can be either an aggregation method or a relay method. Aggregation method means that the data from the new arrival will be aggregated at the access point and the aggregated data will then be transmitted to the sink in the existing tree. Relay method means that the data from the new arrival will be transmitted directly to the sink along the shortest path without any aggregation.

2) When a source node leaves the network, its neighboring nodes must recognize this change immediately and remove the leaving one from their routing paths without affecting other parts of the aggregation routing tree. A simple way is to let all direct children of the leaving one reconnect to the existing tree accordingly.

Nonetheless, the key in designing the algorithm for joining and leaving is to find the optimal access point and suitable access method for a new comer. The criterion here is to confine the structure change to be local, so that the impact upon the remaining tree will be minimal, and at the same time to optimize the energy consumption of the new aggregation routing tree.

3 Problem formulation and algorithm of optimal access

Mathematically, the optimal access problem can be described as follows. Given the network graph $G = (V, E)$ and the existing data aggregation routing tree T for source set S ($S \subset T$), the objective is to construct a new

routing structure T' for the new source set $S' = S \cup \{u\}$ where u is the new arrival source, so as to make $S' \subset T'$ and $T \subset T'$.

Definition 1 Aggregation level: The aggregation level of an aggregation node is the aggregation depth of subtrees rooted at the aggregation node.

Definition 2 Aggregation benefit: When a new source node arrives, it can join in the existing tree by either relay method or aggregation method. Whatever method is used, delivering new information will bring in some additional cost. The difference of additional cost to the entire network between these two access methods is called aggregation benefit. When the benefit is positive, it means that performing aggregation can save more energy and the aggregation access method is preferred; otherwise, relaying new data directly to the sink is more energy efficient.

The optimal access algorithm (OA) consists of four steps as follows.

1) Set $T' = T$. Assign a level i, $1 \leqslant i \leqslant \lg(k+1)$, to the arrival node u with probability $1/2^i$.

2) From each source node in T with level higher than i and "unmatched" at level i, select the optimal aggregation point v for node u so that the total additional cost on the entire network is minimized. The total additional cost on path from u to sink t via v, denoted by $\Delta C(P_{ut})$, is given by

$$\begin{aligned}\Delta C(P_{ut}) = {} & q_{uv}(w(u)+\tilde{w}(v)) + c_{uv}w(u) \\ & + w(u)(1-\rho_{uv})g(v)\end{aligned} \tag{5}$$

Here, the first two parts of Eq. (5) are the aggregation and transmission costs on edge (u, v). $g(v) = \sum_{e \in P_{vt}} (q_e x_e + c_e)$ is the total cost of transmitting unit data from v to sink t in the existing aggregation tree and hence the last part of Eq. (5) is the additional cost on path (v, t) if u uses v as its access point to the existing tree.

3) If $\Delta C(P_{ut})$ is less than the transmission cost on the shortest path from u to t, choose v as the optimal aggregation point for u, and add edge (u, v) to T' as an aggregation edge.

4) If $\Delta C(P_{ut})$ is more than the transmission cost on the shortest path from u to t, add all edges on the shortest path from u to t (SP_{ut}) to T' as non-fusion edges.

Obviously, the basic idea of OA is to randomly assign a level to the arrival node in order to balance the aggregation cost among all nodes and then select an optimal aggregation point for it is based on the aggregation benefit. If no benefit exists, the arrival node will join in the existing tree adopting relay method.

4 Design of distributed online AFST algorithm

In wireless sensor networks, each node can only communicate and exchange information with neighboring nodes. Therefore, distributed algorithm should be implemented by individual sensor nodes to determine or verify its routes relying only on local information. In order to realize such aggregation and routing decisions easily, we divide the algorithm into two processes: first, the process of new source arrival and second, the process of existing source departure. In the process of new source arrival, we will discuss how the new source exchanges necessary information with its neighbors, finds the best access point based on such information, and joins in the existing tree. In the process of existing source departure, we will describe how all the neighboring nodes of the leaving one find the structure changes and reconstruct routes. Before getting into details of these two processes, we first present the data structure maintained at each node.

4.1 Data structure

The information about local routing structure stored at each source node includes the information about its parent or aggregation point, all children nodes whose data is aggregated at this point and itself. Parent information includes the parent's node ID (parentID). Children information consists of a list about all direct children (childList). Information about each child involves the child node ID (childID) and the additional weight this child brings in (childAddWeight). Self information includes the node ID (selfID), aggregation level (level), matched level list (matchedLevelList), current weight $w(\cdot)$ and unit cost to sink $g(\cdot)$.

Besides the information of routing structure, a source node should also maintain its location, the shortest path to the sink, and the information of all neighbors (neighborList). Neighbor information is composed of the neighboring node ID (neibID), its current weight $w(\cdot)$, unit cost to sink $g(\cdot)$, aggregation level (level), matched level list (matchedLevelList), unit transmission cost $c(\cdot)$ from the source, unit aggregation cost $q(\cdot)$ with the source, and data correlation coefficient $\rho(\cdot)$ with the source.

4.2 Process of new source arrival

When a new source arrives, it will calculate its best access point and trigger the process of routing construction. The details of this process are described as follows.

1) Create a neighbor list and the shortest path table. When a new source node u arrives, it shall first find its neighboring nodes by sending out a "hello" message and then query the local routing data of these neighbors. In the process of neighbor finding, it can estimate the unit transmission cost c_{uv} to its neighbors by received signal strength indicator (RSSI) measurement. Based on the location information, u can derive the shortest path to

the sink and estimate the correlation coefficients ρ_{uv} between itself and a neighbor v.

2) Randomly assign a level and create the candidate set of aggregation points. The new source u at random generates a level i for itself with the probability $1/2^i$, then creates the candidate set of aggregation points, denoted by F, from the neighboring sources with level higher than i and being "unmatched" at level i. Sink t is also involved in set F since sink is the last chance to join in the existing tree.

3) New source u selects the best aggregation point v from set F by executing the OA algorithm. As a result, the best aggregation point v may be either a neighbor of node u or the sink t.

4) If $v \neq t$, node u will join in the tree via node v and trigger the parameter update on the path in the tree from v to t according to the steps listed below.

a) Node u sets parentID = v, sets "matched" at level i in matchedLevelList, and modifies the unit cost to sink according to $g(u) = g(v) + c_{uv} + q_{uv}$.

b) Node u sends a "source join" message to node v with the information of u's level and u's additional weight $w_{\text{add}}(u) = w(u)(1 - \rho_{uv})$.

c) After receiving the "source join" message, node v resets its matchedLevelList, updates its current weight according to $w(v) += w_{\text{add}}(u)$, adds node u into its childList and sets the corresponding parameter childAddWeight $= w_{\text{add}}(u)$. Then, node v forwards this "source join" message along its path to sink t.

d) Each source node s on this path will update its current weight according to $w(s) += w_{\text{add}}(u)$ when receiving this message. If this message comes from child node p, node s should also modify the additional weight of child p according to childAddWeight$+= w_{\text{add}}(u)$. Then, node s continuatively forwards this message along its path till to sink t.

5) If the selected best point v is just the sink t, node u will use the shortest path to deliver its data to the sink without any aggregation. Therefore, new source u needs to set $g(u)$, which is its unit cost to the sink, as the unit transmission cost on its shortest path to the sink.

4.3 Process of existing source departure

Every node can find out whether its neighboring nodes leave the network by sending and acknowledging "hello" message. If a node detects that one of its neighbor node has left the network, it performs different departure processes according to its role in the existing tree.

1) When an aggregation point v finds that its child node u has left the network,

a) Node v sets "unmatched" at u's level i in its matchedLevelList, updates its current weight according to $w(v) -= w_{\text{add}}(u)$, where $w_{\text{add}}(u)$ is the additional weight that u brings in, and deletes node u from its childList.

b) Node v sends a "child departure" message with u's additional weight along the path from v to sink t.

c) Each node s on this path updates its current weight according to $w(s) -= w_{\text{add}}(u)$. If this message comes from child node p, node s should also modify the additional weight of child p according to childAddWeight$-= w_{\text{add}}(u)$. Then, node s forwards this message along its path till to sink t.

2) When a node s finds its parent node u has left,

a) Node s deletes node u from its neighborList first, and then executes the process of new source arrival detailed in the previous subsection to figure out its new parent node v. Subsequently, node s shall update its parentID as node v and send a "source join" message to its new parent, v, to update the routing information of every node on the path from v to t.

b) Node s calculates its unit cost amendment to the sink according to $\Delta g(s) = g_{\text{new}}(s) - g_{\text{old}}(s)$, and then sends a "change parent" message to all of its children with parameter $\Delta g(s)$.

c) Each child node p of s updates its unit cost to sink according to $g(p) += \Delta g(s)$, and forwards this "change parent" message to all of its children till to the leaf of the routing branch.

From the processes of source joining and leaving, we conclude that each node can adaptively adjust its route along with the changes of the network structure by only depending on the information of its neighbors and its own. To update the routing information of the affected nodes in the network, only three maintenance messages need to be transmitted along the affected paths. Therefore, this algorithm is easier to be implemented in a distributed fashion and the maintenance cost is limited.

5 Experimental studies

In this section, we present an extensive set of simulations to evaluate the performance of the proposed distributed online algorithm comparing with the offline AFST algorithm.

In our setup, up to 100 sensor nodes are uniformly distributed in a square of 50 m × 50 m. We assume that each node generates one 400-byte packet as original sensed data in each data gathering period and sends the data to the sink located at one corner. We also suppose that the maximal communication radius of a sensor is r_c. That is, if and only if two sensor nodes are within this range, there exists a communication link between them. When two nodes are within r_c, the unit transmission cost between them is give by $c(e) = \beta d^2 + \varepsilon$, where d denotes the distance between the two nodes, β represents the energy consumption on the transmit amplifier, and ε denotes energy consumption per bit on the transmitter circuit and receiver circuit. In our experiments, we set $\beta = 100$ pJ/bit·

m^{-2} and $\varepsilon = 100$ nJ/bit according to their typical values. By varying r_c, we can control the network connectivity and hence adjust the network topology and density. The correlation model employed here is an approximated spatial model. Let r_s denote the correlation range. If the distance between two nodes are longer than r_s, simply the correlation coefficient $\rho = 0$. Otherwise, it is given by $\rho = 1 - d/r_s$, where d denotes the distance between the nodes. Usually, two nodes located closer may bring in higher data correlation. By varying the correlation range r_s, we can control the average correlation coefficient of the network, and further control the average data reduction after data aggregation. For the aggregation cost, we assume that unit aggregation cost is appropriately constant and use ω to denote the average aggregation cost per bit at each node. By varying the unit aggregation cost ω, we can control the aggregation benefit among the network and then study the effect of aggregation decision.

5.1 Impact of network connectivity

By varying the communication range r_c, we can control the network connectivity and hence adjust the network topology and density. Figure 1 shows the result of comparison between AFST-online and AFST-offline. We can see that the difference of energy consumption between AFST-online and AFST-offline decreases along with increased network connectivity. Notably, the relative difference rapidly decreases from 17% to 5% when $r_c < 20$ m, which shows that the online algorithm actually benefits more from increased network connectivity and network density. The reason is that a new arrival can have more access candidates if the communication range is larger and hence the new arrival has more chances to find a more suitable access point. As a result, the performance of the online algorithm is close to the offline version. When $r_c > 20$ m, the difference range decreases slowly and becomes stable at about 5%. Regardlessly, the result shows that in a dense sensor network with high data correlation, the online algorithm is an excellent solution approximating the offline version.

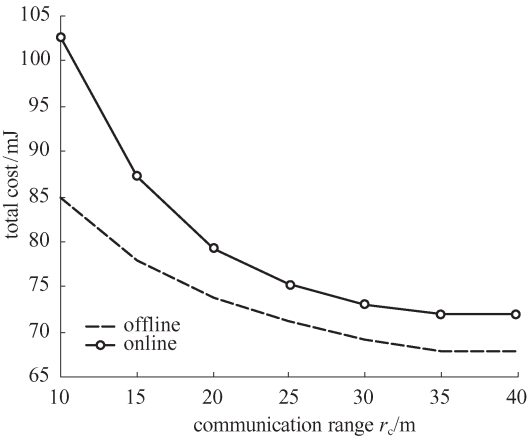

Fig. 1 Impact of network connection ($r_s = 50$ m, $\omega = 50$ nJ/bit)

5.2 Impact of data correlation

By varying the correlation range r_s, we can control the average correlation coefficient of the network, and further control the average data reduction after data fusion. Figure 2 illustrates the result when we increase the correlation range, r_s, from 0.5 m to 1024 m, which corresponds to the varying ρ from 0 to 1. When $\rho \to 0$ and $\rho \to 1$, the relative difference is very small, that is, ranging within 2%. The reason is that the online algorithm uses similar criteria to find the best access point for a new arrival as the offline version and the resulting trees are almost the same in such extremely cases. When $0 < \rho < 1$, since data correlation between nodes changes along with their distance, aggregation benefit can be positive only between closer nodes due to the existence of aggregation cost. In AFST-offline, sink node can calculate the perfect matching pairs and do aggregation decision depending on aggregation benefits among all nodes. On the contrary, in the online algorithm, an arrival may not find its perfect pair-node since source nodes join in the network randomly. Indeed, an arrival may choose to connect to the sink through the shortest path if all of its possible matching pair-nodes have not arrived yet. Therefore, the difference is larger as compared with both extreme cases: $\rho \to 0$ and $\rho \to 1$. However, during the whole range, the difference is consistently within 9%.

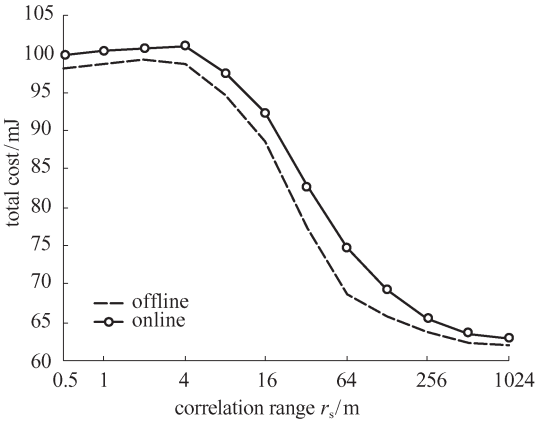

Fig. 2 Impact of data correlation ($r_c = 25$ m, $\omega = 50$ nJ/bit)

5.3 Impact of unit aggregation cost

By varying the unit aggregation cost ω, we can control the aggregation benefit among the network and study the effect of aggregation decision. From Fig. 3, we can see that the difference between the online and offline

algorithms decreases with the increase of the unit aggregation cost. It can be analyzed as follows, when the value of ω is small, the source node tends to employ aggregation access method since the aggregation benefit can be easily obtained. However, the new arrival may not find a perfect pair-node when $0 < \rho < 1$ as discussed in the previous subsection. Therefore, the AFST-online shows a 14% difference from the offline version. Along with the increase of unit aggregation cost, the aggregation benefit decreases. As a result, more and more source nodes tend to join in the network by using relay method. When ω is very high, both online and offline algorithms will result in the shortest path tree without performing any data aggregation. Hence, the performance difference decreases to be within 2%.

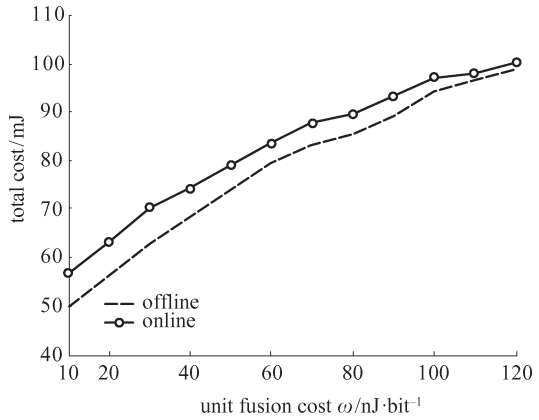

Fig. 3 Impact of unit aggregation cost ($r_c = 20$ m, $r_s = 50$ m)

6 Conclusions and future work

In this paper, we have proposed the AFST-online, a distributed online algorithm for enroute aggregation decision for gathering correlated data in sensor networks. The AFST-online optimizes over both the transmission and aggregation costs by adaptively adjusting its aggregation decisions based on data correlation, aggregation cost and transmission cost. Furthermore, the maintenance cost is very low since each node can calculate its best access point relying on local information only. Experimental results show that the energy consumption difference between this distributed online algorithm and the previous offline one is within 17% under any network conditions. In extreme cases, the difference is less than 2%. Therefore, we can conclude that this distributed online algorithm can hold the energy-efficient features of its offline ancestor and is practical in real network environments.

In the future, we will further study some more complex application environments, such as the case in which the unit aggregation cost is not constant or the unit transmission cost includes congestion cost, to make this algorithm more adaptive and practical.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant No. 60642005), the Foundation of China Scholarship and the Program for Changjiang Scholars and Innovative Research Team in University.

References

1. Krishnamachari B, Estrin D, Wicker S. The impact of data aggregation in wireless sensor networks. IEEE ICDCS'02, 2002: 575–578
2. Liu L F, Zou S H, Zhang L, et al. A density control algorithm based on probability coverage model in wireless sensor networks. Journal of Beijing Universities of Posts and Telecommunications, 2005, 28(4): 14–17 (in Chinese)
3. Cristescu R, Beferull-Lozano B, Vetterli M. On network correlated data gathering. IEEE Infocom'04, 2004, 4: 2571–2582
4. Goel A, Estrin D. Simultaneous optimization for concave costs: single sink aggregation or single source buy-at-bulk. ACM SODA'03, 2003: 499–505
5. Liu Y Y, Ji H, Yue G X. Routing protocol with optimal location of aggregation point in wireless sensor networks. The Journal of China Universities of Posts and Telecommunications, 2006, 13(1): 1–5
6. Luo H, Liu Y H, Das S K. Energy efficient routing with adaptive data fusion in sensor networks. ACM DIALM-POMC'05, 2005: 80–89