Routing Correlated Data with Fusion Cost in Wireless Sensor Networks

Hong Luo, *Member*, *IEEE*, Yonghe Liu, *Member*, *IEEE*, and Sajal K. Das, *Member*, *IEEE*

Abstract—In this paper, we propose a routing algorithm called *Minimum Fusion Steiner Tree (MFST)* for energy efficient data gathering with aggregation (fusion) in wireless sensor networks. Different from existing schemes, MFST not only optimizes over the data transmission cost, but also incorporates the cost for data fusion, which can be significant for emerging sensor networks with vectorial data and/or security requirements. By employing a randomized algorithm that allows fusion points to be chosen according to the nodes' data amounts, MFST achieves an approximation ratio of $\frac{5}{4}\log(k+1)$, where k denotes the number of source nodes, to the optimal solution for extremely general system setups, provided that fusion cost and data aggregation are nondecreasing against the total input data. Consequently, in contrast to algorithms that only excel in full or nonaggregation scenarios without considering fusion cost, MFST can thrive in a wide range of applications.

Index Terms—Wireless sensor networks, data fusion, routing, randomized algorithm, approximation.

1 Introduction

Wireless sensor networks have attracted a plethora of research efforts due to their vast potential applications [1], [2]. In particular, an extensive set of research work has been devoted to providing energy efficient routing algorithms for data gathering [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18]. While a class of shortest path tree (SPT)-based routing strategies has been developed in [3], [4], [9] assuming statistically independent information, the more realistic case of correlated data has also been considered in [5], [6], [7], [8], [10], [11], [12], [13], [15], [16], [17], [18]. By exploring data correlation and employing in-network processing, redundancy among sensed data can be curtailed and, hence, the network load can be reduced [6]. The objective of the sensor routing algorithm is then to jointly explore the data structure and network topology to provide the optimal strategy for data gathering.

Routing with data aggregation can be generally classified into two categories: routing-driven and aggregation-driven. In *routing-driven* algorithms [5], [6], [7], [10], [11], [13], data is routed through shortest paths to the sink, with aggregation taking place opportunistically when data flows encounter. In *aggregation-driven* routing algorithms [12], [15], [16], routing paths are heavily dependent on data correlation in order to fully benefit from information reduction resulted from data aggregation. In this paper, we will use "aggregation" and "fusion" interchangeably, denoting the data reduction process on intermediate sensor nodes.

Regardless of the techniques employed, existing strategies miss one key dimension in the optimization space for routing correlated data, namely, the *data aggregation cost*. The cost for data aggregation may be negligible for certain types of networks. For example, sensor networks monitoring field temperature may use simple average, max, or min functions, which essentially cost nothing. However, other networks may require complex operations for data fusion. One example is hop-by-hop secure networks, where encryption and decryption at intermediate nodes will significantly augment fusion cost even though the fusion function itself may be simple. It has been shown in [19] that energy consumption of a beamforming algorithm for acoustic signal fusion is on the same order of that for data transmission. Moreover, in our own experimental study described in [20], we found that a typical aggregation function for vectorial data, such as image fusion, costs tens of nanojoules (nJ) per bit, which is on the same order as the communication cost reported in the literature [19].

In this paper, we include fusion cost as another dimension to the space of routing optimization for correlated data. Differing from transmission cost, which depends on the output of the fusion function, fusion cost is mainly determined by the input of the fusion function. Therefore, in addition to transmission cost, fusion cost can significantly affect routing decisions when involving data aggregation. For example, a high fusion cost may deter a node from employing multihop transmission strategy, especially when the data amount cannot be significantly reduced. At the same time, various pairing options among nodes and, hence, different fusion costs may ultimately affect the optimal routing topology. Therefore, an optimal routing algorithm needs to *jointly* optimize over the transmission and fusion costs in order to minimize the total energy consumption. Since this problem is NP-complete [15], our objective is to design an approximation algorithm.

1.1 Related Work

Routing with data aggregation targets at jointly exploring the data structure and network topology to reduce energy

- *H. Luo is with the College of Computer Science and Technology, Beijing University of Posts and Telecommunications, 100876, China. E-mail: luoh@bupt.edu.cn.*
- *Y. Liu and S.K. Das are with the Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, TX 76019. E-mail: {yonghe, das}@cse.uta.edu.*

Manuscript received 16 Apr. 2005; revised 23 Oct. 2005; accepted 28 Jan. 2006; published online 15 Sept. 2006.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-0103-0405.

consumption for data gathering in resource limited sensor networks.

If the complete knowledge of all source correlations is available in advance at each source, theoretically the best approach is to use distributed source coding typified by Slepian-Wolf coding [21]. In this technique, compression is done at original sources in a distributed manner to achieve the minimum entropy and, hence, avoid the need for data aggregation on the intermediate nodes. In [15], an optimal rate allocation algorithm is proposed for nodes in the network and SPT is employed as the routing scheme. However, implementation of distributed source coding in a practical setting is still an open problem and likely to incur significant additional cost because of the aforementioned assumption.

Routing-driven algorithms emphasize source compression at each individual node and aggregation occurs opportunistically when routes intersect. In [11], the directed diffusion scheme was proposed, where sensors create gradients of information in their respective neighborhoods. If the gradients match the broadcast interests from the sink, information routes back to the sink are formed and data is aggregated at the intersections. To improve path sharing for more energy savings, a greedy incremental tree (GIT) is described in [10] to adjust aggregation points on the routes. LEACH [5] is a cluster-based protocol in which sensors directly send raw data to cluster heads where data fusion is performed. Aggregated data is then delivered to the sink through multihop path. In PEGASIS [18], sensors form chains along which a node transmits and receives from a nearby neighbor. Data aggregation is then performed while data moves from node to node. In [13], [14], the sensor collaboration issue in target tracking is addressed, where sensors in a target area collaborate among themselves to aggregate data, and one of them generates a data report to the sink. This scheme focuses on dynamic tree expanding/pruning and tree reconfiguration when the target moves. The basic routing structure in target area is simple SPT. In [15], it has been proved that the minimum-energy data gathering problem is NP-complete by applying reduction to the set-cover problem and claimed that the optimal result is between SPT and a traveling salesman path. A common feature in these protocols is that data correlation is not exploited explicitly.

When designing aggregation-driven algorithms, various assumptions have been made on the model regarding data aggregation. In the single-input aggregation model, fusion of one node's information depends only on the information of one other node and the encoded data can not be recoded. This strategy best fits asynchronous sensor networks. Under this model, an optimal algorithm MEGA for foreign-coding and an approximation algorithm LEGA for self-coding are proposed in [16]. In MEGA, each node sends raw data to its encoding point using a directed minimum spanning tree (MST) and encoded data is then transmitted to the sink through SPT. On the other hand, LEGA uses a shallow light tree (SLT) [22], [23] as the data gathering topology and achieves a $2(1+\sqrt{2})$ approximation ratio for self-coding.

In a multi-input aggregation model, the amount of aggregated information sent to the sink from one node depends on the structure of the subtree rooted at that node. In this model, each node can theoretically obtain the joint entropy of its subtree to receive the maximal aggregation ratio. One strategy is that aggregation is performed at a node only if all input information from its child nodes is available in order to exploit the correlation among them. Based on this model, a hierarchical matching algorithm is proposed in [12], resulting in an aggregation tree with a logarithmic approximation ratio to the optimal for all concave aggregation functions. However, in this model, aggregation depends only on the number of nodes in the subtree rooted at the aggregation node, regardless of the correlation among the data.

1.2 Our Contributions

In this paper, we employ a general aggregation model, where data aggregation may potentially occur at any point along a route. In particular, aggregated data may be fused again. Mathematically, the model only requires that the output data amount of the fusion function is not less than any of its inputs and not more than the summation of all inputs. From this point of view, the model is a generalization of the multi-input model. Moreover, our model does not depend on any specific relations among information supplied by sensors nor on specific correlation models.

We define the minimum energy routing problem constrained not only by transmission cost, but also by fusion cost, since fusion cost can be comparable to transmission cost in certain sensor systems due to either data characteristics or encryption/decryption overhead. Consequently, we formulate the problem as a combinatorial optimization problem. As the problem is NP-complete, by proposing a new metric combining both fusion and transmission costs, we design *Minimum Fusion Steiner Tree (MFST)*, a randomized algorithm with a provable approximation ratio of $\frac{5}{4}\log(k+1)$ to the optimal, where k denotes the number of source nodes. While our technique is rooted in [12], [24], the problem and approach are significantly different. On one hand, we allow the costs of fusion and transmission to be link dependent and model it as a function of the amount of data. On the other hand, there is no data aggregation in [24] nor any fusion cost in [12], while our model incorporates a general aggregation model to describe data reduction.

Our model is quite unrestricted. It accounts for per link transmission cost, general nonconvex fusion cost as a function of input streams, and a broad range of data aggregation models. An extensive set of simulations show that MFST performs well under various system setups. Unlike MST and SPT algorithms that can only perform well under certain extreme situations such as full or nonaggregation of data without considering fusion cost, MFST adapts well to varying sensor correlations, fusion costs, and network topologies.

The remainder of this paper is organized as follows: In Section 2, we describe the system model and formulate the routing problem. Section 3 details the randomized approximation algorithm, followed by the analysis in Section 4. Section 5 provides analytic comparison between MFST and other algorithms, while Section 6 studies the performance of MFST through extensive simulations. Finally, Section 7 concludes the paper.

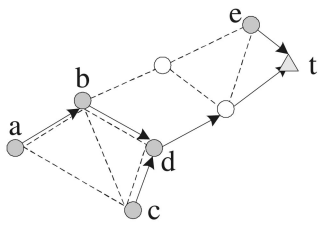


Fig. 1. An example of a data gathering tree.

2 System Model and Problem Formulation

2.1 Network Model

We model a sensor network as a graph $G = (V, E)$, where V denotes the set of sensors (nodes) and E the set of edges representing the communication links between pairs of sensors. We assume that a set $S \subset V$ of k nodes are data sources of interests and the sensed data needs to be gathered at a special sink node $t \in V$, where it is further processed. Our focus is given to energy-efficient gathering of the information from the source nodes to the sink. Fig. 1 illustrates an example of the data gathering process, where gray circles represent sensor nodes generating source data, dashed lines are possible communication links among the nodes, and the solid lines compose a possible routing tree for data gathering.

Intuitively, two components of the network will determine the energy consumption of a routing strategy, namely, the information amount of the source nodes and the transmission cost on each link. For convenience, we denote the amount of information of a node to be its *weight*. Formally, for a node $v \in S$, the node weight $w(v) : S \rightarrow \Re^+$ denotes the amount of information outgoing from v, where $\Re^+$ denotes the set of positive real numbers. In addition, an edge $e \in E$ is denoted as $e = (u, v)$, where u is the start node and v is the end node. The weight of edge e is equivalent to the weight of its starting node, i.e., $w(e) = w(u)$. Associated with edge e is the transmission cost, which is defined as $t(e) : E \rightarrow \Re^+$, denoting the cost for transmitting $w(e)$ data from u to v.

As mentioned earlier, along the routing path, data from multiple nodes can be aggregated in order to reduce the network load. For example, data from node a can be aggregated with that of node b, which will in turn forward the aggregated data to d. We assume that data aggregation can potentially take place at any intermediate node along the route: An intermediate node can explore the redundancy among multiple child-nodes' data and aggregate all into one compressed data stream.

In this paper, we also capture the cost of aggregating data in the network. Specifically, on edge $e = (u, v)$, we define fusion cost $f(e) : E \rightarrow \Re^+$, denoting energy consumption for the fusion process at node v.[1] Therefore, the weight of a leaf node in the routing tree, such as node a in Fig. 1, is the same as its original amount of information, whereas the weight of an intermediate node, such as node d, is the total amount of information of the subtree rooted at that intermediate node *after* data fusion. Since data fusion is performed by intermediate nodes to aggregate their own data with that of their children, in order to avoid confusion, we use $\widetilde{w}(\cdot)$ to denote the temporary weight of a node *before data fusion* and use $w(\cdot)$ to denote the weight of a node after data fusion.

In the following, we will further detail and formally define data aggregation, transmission cost, and fusion cost.

2.2 Correlation and Data Aggregation

Key to a sensor data routing protocol is the data reduction ratio after data aggregation. Unfortunately, this ratio is heavily dependent on the application scenarios. For example, in a sensor network detecting the maximum temperature in a field, each node only sends out one temperature value packet after data aggregation. On the other hand, in a video sensor network monitoring an area, images collected by different sensor nodes may offer redundancy due to overlapping fields of view. However, even with data aggregation, information is likely to increase.

To accommodate a variety of applications, we do not constrain ourselves to any particular model of data aggregation. The only assumption we make is that if the data of nodes u and v is fused at v, the resulting amount of data is not less than either of the component data. In other words, we assume

$$w(v) \geq \max\{w(u), \widetilde{w}(v)\}. \quad (1)$$

And, evidently, we shall have $w(v) \leq w(u) + \widetilde{w}(v)$. Otherwise, aggregation shall not be performed at all and the problem becomes trivial.

In this paper, we assume that the aggregation process for multiple inputs at a particular point is performed step by step (fusing with nodes in turn) and, hence, the above formula is adequate in characterizing the fusion process. The justification of this assumption lies in the resource limitation of sensor nodes. Storing multiple inputs and fusing them at once may be difficult for sensors as it requires large memory and additional processing power. Second, data reported from different sensors cannot arrive at the same time, due to either the shared wireless medium or various intermediate nodes and processing. Therefore, fusing existing data with the newly received data when it arrives is a natural solution. In other words, in a step-by-step fusion manner, the fusion point first aggregates its own data with one input and next fuses the aggregation result with another input. This process will be repeated until all the inputs are aggregated. For example, in Fig. 1, node d fuses data from node c with its original data and saves it as its temporary data; then, node d will aggregate it again with the data from node b and send the final result along its path to sink t.

2.3 Transmission and Fusion Costs

The transmission cost over an edge e depends on two factors: the unit cost of the link for transmitting data from u to v and the amount of data to be transmitted. The latter factor is simply $w(e)$. In practice, cost per unit data depends not only on the Euclidian distance between the two nodes and the physical layer technology employed, but also on the various networking overhead. However, to simplify our model, we abstract the unit cost as $c(e)$ and, thus, the transmission cost $t(e)$ is

$$t(e) = w(e)c(e). \quad (2)$$

1. The fusion cost is defined on the edge instead of the node for notational convenience.

Notice that $c(e)$ is link-dependent and, hence, can accommodate various conditions per link, for example, different distances between nodes and local congestion situations.

The fusion cost over an edge e depends on the amount of data to be fused as well as the algorithms utilized. In this paper, fusion cost is expressed by a general function $q(\cdot)$, such that the cost for fusing the data of nodes u and v at node v is given as

$$f(e) = q\Big(w(u), \widetilde{w}(v)\Big). \tag{3}$$

We require $q(x, y)$ to possess the following properties: 1) it is symmetrical of x and y, 2) $q(x, y) \geq 0$ and equality is true iff $x \cdot y = 0$, 3) $q(x, y)$ is monotonically nondecreasing of x and y, and 4) $q(x, y)$ is nonconvex when either x or y is fixed. All these properties can be naturally justified. For example, more energy is required for fusing larger amount of data and thus justifies 3). Similarly, with the increase of data, the margin of fusion cost of unit data will decrease as overhead will be averaged down. This justifies 4).

Although both transmission and fusion costs are link-based, we remark that they cannot be simply combined together and, hence, rely on existing techniques solely based on the transmission cost to solve this problem. The reason is that the fusion cost on an edge is determined by the inputs of the fusion function. The inputs include both the incoming data from other nodes and the data produced by the fusion point itself. On the contrary, the transmission cost on an edge is only determined by the weight of the start point of the edge. In other words, for a fusion point, the transmission cost is only determined by the output of the fusion function. More evidently, this can be seen from (2) and (3).

2.4 Problem Formulation

Given the source node set S and sink t, our objective is to design a routing algorithm that minimizes the energy consumption when delivering data from all source nodes in S to the sink t. Mathematically, the goal is to find a connected subgraph $G^* = (V^*, E^*) \subseteq G$, which contains all sources ($S \subset V^*$) and the sink ($t \in V^*$), such that the following sum is minimized:

$$\sum_{e \in E^*} \Big(f(e) + t(e)\Big). \tag{4}$$

Different from existing work, the objective function includes both transmission and fusion costs. In particular, as discussed above, the transmission cost and fusion cost are link-dependent, which can account for general application scenarios.

As each node in the network will aggregate all inputs with its own data to form one outgoing aggregated packet, the solution to the above problem evidently is in the form of a Steiner tree rooted at sink t. Therefore, our objective next is to find a routing Steiner tree that is the solution to (4), which minimizes the total energy consumption.

3 MFST Algorithm Design

It has been shown that, even if only the transmission cost is considered, the problem defined in the last section is NP-complete [15]. Therefore, heuristic algorithms have been designed in the literature for finding approximations to the minimum transmission cost tree [15], [16]. Since fusion cost is also incorporated into our design, the resultant combinatorial problem is also NP-complete. In this section, we design a randomized approximation algorithm that is bounded within a $\frac{5}{4}\log(k+1)$ ratio to the optimal solution, where k denotes the number of source nodes. As our focus is given to the joint minimization of both transmission and fusion costs, we term our solution *Minimum Fusion Steiner Tree (MFST)*. To the best of our knowledge, this is the first attempt that concurrently optimizes both transmission and fusion costs in designing routing algorithms for gathering correlated data in wireless sensor networks.

3.1 Minimum Fusion Steiner Tree

In MFST, we first pair up source nodes (or a source with the sink) based on the metric defined below and then randomly select a fusion node from the node-pair. The weight of the nonfusion node will be transferred to the fusion node, paying appropriate transmission and fusion costs on that edge. Subsequently, the nonfusion node will be eliminated and the fusion node with aggregated weight will be grouped as a new set of sources. We then repeat this process on the new set until the sink is the only remaining node. In this paper, we term each such process a "stage" of the algorithm. The detailed algorithm is presented below.

MFST ALGORITHM:

1) Initialize stage index $i = 0$, $S_0 = S \cup \{t\}$, and $E^* = \emptyset$. Let $w_0(v)$ for any $v \in S$ equal to its original weight and let $w_0(t) = 0$, where t is the sink.
2) Given S_i for stage i, for every pair of nonsink nodes $(u, v) \in S_i$:
   - Find the minimum cost path (u, v) in G according to the metric

$$M(e) = q(w_i(u), w_i(v)) + \alpha(w_i(u), w_i(v))c(e), \tag{5}$$

     where $\alpha(w_i(u), w_i(v)) = \frac{w_i(u)w_i(v)(w_i(u)+w_i(v))}{w_i^2(u)+w_i^2(v)}$.
   - Define $K_i(u, v)$ to be the distance under metric $M(e)$ of this path.
3) For every nonsink node $u \in S_i$:
   - Find the minimum cost path (u, t) in G according to the metric

$$M(e) = q\Big(w_i(u), w_i(t)\Big) + w_i(u)c(e). \tag{6}$$

   - Define $K_i(u, t)$ to be the distance under metric $M(e)$ of this path.
4) Find minimum-cost perfect matching[2] between nodes in S_i. Let $(u_{i,j}, v_{i,j})$ denote the jth matched pair in S_i, where $1 \leq j \leq |S_i|/2$. If there is only one nonsink node left after matching, match it to itself without any cost, and consider it as the last "single-node pair" in S_i.

2. Minimum-cost perfect matching is a matching that guarantees the total cost (distance) for all pairs under $M(e)$ is minimized. Papadimitriou and Steiglitz [25] provide polynomial-time algorithms for this problem. An example solution is to divide all nodes into different connected subgraphs whose edges are determined by selecting the nearest neighbor for each vertex, construct an Euler tour for each subgraph and reduce it to a Hamiltonian, and then select the best matching out of all matchings on each Hamiltonian.

5) For each matched pair (u, v), add those edges that are on the path defining $K_i(u, v)$ to the set E^*.
6) For each pair of nonsink matched nodes (u, v), choose u to be the fusion node with probability

$$P(u = fusion\ node) = \frac{w_i^2(u)}{w_i^2(u)+w_i^2(v)}. \quad (7)$$

Otherwise, v will be the fusion node. For pair (u, t), choose t to be the fusion node.
7) Transport the weight of a nonfusion node to its corresponding fusion node. According to (1), the weight of the fusion node satisfies

$$w_{i+1}(fusion\ node) \geq \max\{w_i(u), w_i(v)\}. \quad (8)$$

8) Remove all nonfusion nodes from S_i; then, the remaining fusion nodes induce S_{i+1}.
9) If S_{i+1} contains only the sink, we return $G^* = (V^*, E^*)$, where E^* is the set of edges constructed and V^* includes the source nodes and the sink. Otherwise, the matching process increment from Step 2 can be executed again.

One of the key design components in the algorithm is metric $M(e)$ for edge $e = (u, v)$ as defined in (5). This metric is composed of two parts, the fusion cost and the transmission cost on edge e. As transmission cost is dependent on the data amount and different choices of fusion point (u or v) will lead to different amounts of information to be transmitted, we employ

$$\alpha(w_i(u), w_i(v)) = \frac{w_i(u)w_i(v)(w_i(u) + w_i(v))}{w_i^2(u) + w_i^2(v)}$$

as the expected weight for transmission to evaluate the transmission cost. As we will show later, this new metric will allow the algorithm to jointly optimize over the transmission and fusion costs in order to minimize the total energy consumption.

Notice that the size of the set S_i is reduced to half after each stage of the algorithm. Therefore, the process terminates after $\log(k + 1)$ stages. Furthermore, since the fusion node is randomly selected according to a probability based on the node weights, the fusion process is randomly distributed among all sensor nodes. To utilize this property, the algorithm can be rerun periodically to generate a new realization of the tree. As a by-product, the algorithm can then balance fusion costs among sensor nodes naturally and, hence, prevent certain node's battery power from being exhausted due to heavy fusion in a short time.

4 Analysis of MFST

In this section, we prove that the approximation ratio of MFST to the optimal solution is $\frac{5}{4}\log(k + 1)$, where k is the number of source nodes. Let T^* denote the optimal solution tree. The optimal cost (minimum-energy consumption) on T^* is defined as $C^* = \sum_{e \in T^*}(f(e) + t(e))$. We measure MFST's performance against C^*. Since MFST is randomized, we analyze its expected performance.

In each stage, the algorithm incurs transmission and fusion costs on the set S_i, which is the source set in the $(i + 1)$th stage, for merging nodes. Let G_i denote the total expected cost of the ith stage. The expected cost of the algorithm is then the summation of the expected costs of all stages. Define C_i^* as the cost of the optimal routing tree for S_i. Obviously, $C_0^* = C^*$. Our approach for proof is to first upper bound the expected cost of the optimal routing algorithm, C_i^*, for S_i in Lemma 1. Then, in Lemmas 3 and 4, we prove that G_{i+1} is in turn bounded by $(\frac{5}{4})C_i^*$. Combining these lemmas, the desired result is derived in Theorem 1.

Before proceeding further, we first introduce the following assumption needed in the analysis. In the data gathering tree, a link may reside on multiple routes for different sources. If nodes v and u are physically in proximity, the probability of a link residing on the route of u to the sink and the probability of it residing on the route of v to the sink are equal. This assumption can be intuitively justified for sensor networks with dense deployment and also where the sink is not deployed in the monitored environment together with the sensor nodes. Given this assumption, we can obtain the following lemma:

Lemma 1. *For each stage i, $i \geq 1$, the expected cost*

$$\mathbf{E}[C_i^*] \leq C_{i-1}^*.$$

Proof. Let $w_i(v)$ denote the weight of source node v in stage i. Let $(u_{i,j}, v_{i,j})$ represent the jth matching pair constructed in the $(i + 1)$th stage of the MFST algorithm. For a given i, construct a sequence $D_{i,j}^*$, for $0 \leq j \leq \lceil |S_i|/2 \rceil$, where $D_{i,j}^*$ is the cost of the optimal solution for the residual problem[3] after the jth random fusion node selection during the $(i + 1)$th stage. Note that the last pair may be the special "pair" with only one node. By definition, we have $D_{i,0}^* = C_i^*$ and $D_{i,\lceil |S_i|/2 \rceil}^* = C_{i+1}^* = D_{i+1,0}^*$.

In order to prove $E[C_i^*] \leq C_{i-1}^*$, we first prove that the sequence $D_{i,j}^*$ is supermartingale for a fixed value of i. That is, for all $0 \leq i < \log(k + 1)$ and $0 \leq j < \lceil |S_i|/2 \rceil$, $\mathbf{E}[D_{i,j+1}^*] \leq D_{i,j}^*$.

Let $T_{i,j}^*$ denote the optimal tree for the residual problem after the jth random fusion node selection during the $(i + 1)$th stage, where $0 \leq j < \lceil |S_i|/2 \rceil$. For an edge e in tree $T_{i,j}^*$, let $w(e)$ denote the total data routed through e. After the $(j + 1)$th fusion node selection, let $w'(e)$ be the total data through this edge for the new residual problem on tree $T_{i,j}^*$. Notice that the optimal tree $T_{i,j+1}^*$ for the new residual problem might be quite different from $T_{i,j}^*$. For a node pair $(u_{i,j+1}, v_{i,j+1})$, there are three cases before fusion node selection:

1. Edge e lies on the paths from both $u_{i,j+1}$ and $v_{i,j+1}$ to the sink in $T_{i,j}^*$.
2. Edge e lies on neither of the paths.
3. Edge e lies only on the path of $u_{i,j+1}$ or $v_{i,j+1}$, but not on both.

In the first two cases, $w'(e) \equiv w(e)$ regardless of which node is chosen as the fusion node. In the last case, let p be the probability of selecting u as a fusion node; then, the probability of selecting v is $(1 - p)$. After the selection,

3. For simplification in this proof, we use the *residual problem* to represent the routing problem for the remaining source nodes after current fusion node selections in one stage.

$w'(e)$ will increase by $\triangle w(v)$ with probability p or increase by $\triangle w(u)$ with probability $(1-p)$, depending on which node is chosen, as extra data will be routed through it. Similarly, $w'(e)$ will decrease by $\triangle w(v)$ with probability p or decrease by $\triangle w(u)$ with probability $(1-p)$ when data, routed through it earlier, changes path. As $u_{i,j+1}$ and $v_{i,j+1}$ are nodes to be paired together, they shall be within proximity of each other, as compared with other nodes remaining in S_i. Otherwise, the high transmission cost will factor in and deter the fusion. Using the aforementioned assumption, the probability of an edge being on either path to the sink shall be equal. Given this condition, it is easy to show that the expected value of $w'(e)$ in the third case is also $w(e)$. For the special "single node pair" $u_{i,j+1}$, only the first two cases are possible before fusion node selection. Therefore, by exhausting all cases, we have $E[w'(e)] = w(e)$.

Let $D_{i,j+1}$ denote the cost of the tree $T^*_{i,j}$ for the residual problem after the $(j+1)$th fusion node selection. In other words, $D_{i,j+1}$ is the cost of $T^*_{i,j}$ with new weight on each edge. Since $D^*_{i,j+1}$ is the cost of the optimal tree $T^*_{i,j+1}$ for the same set of nodes, it must be less than $D_{i,j+1}$. Formally, we have

$$D^*_{i,j+1} \le D_{i,j+1} = \sum_{e \in T^*_{i,j}} \Big(q\Big(w'(e), \widetilde{w}_p(e)\Big) + w'(e)c(e)\Big).$$

If $e = (u, v)$ and v is parent of u in $T^*_{i,j}$, then $\widetilde{w}_p(e)$ is the weight of v before the fusion on e occurs. Since function $q(x, y)$ is nonconvex when either x or y is fixed, based on Jensen's inequality [26], we have $\mathbf{E}\big[q(x, y)\big] \le q(\mathbf{E}[x], y)$ if y is fixed. Substituting y and x with $\widetilde{w}_p(e)$ and $w'(e)$, respectively, we obtain the following inequality on the expected value of $D^*_{i,j+1}$:

$$\begin{aligned}
&\mathbf{E}\Big[D^*_{i,j+1}\Big] \\
&\quad \le \mathbf{E}\left[\sum_{e \in T^*_{i,j}} \Big(q\Big(w'(e), \widetilde{w}_p(e)\Big) + w'(e)c(e)\Big)\right] \\
&\quad \le \sum_{e \in T^*_{i,j}} \Big(q\Big(\mathbf{E}[w'(e)], \widetilde{w}_p(e)\Big) + \mathbf{E}[w'(e)]c(e)\Big) \\
&\quad = \sum_{e \in T^*_{i,j}} \Big(q\Big(w(e), \widetilde{w}_p(e)\Big) + w(e)c(e)\Big) \\
&\quad = D^*_{i,j}.
\end{aligned}$$

Consequently, we obtain $E[C^*_i] \le C^*_{i-1}$ as

$$C^*_i = D^*_{i,0} = D^*_{i-1,\lceil |S_i|/2 \rceil}.$$

□

Lemma 2. *Given a tree $T = (V, E)$ and a set of nodes $S \subseteq V$, there exists a perfect matching of the nodes in S that uses each edge of T at most once.*

Proof. We will prove this lemma by induction on the number of edges in the tree. If the tree has only one node, the result is trivially true since there is no edge. For a tree with more than one node, suppose $v \in V$ is the deepest leaf of this tree. If $v \notin S$, we can remove v and the edge connecting it to its parent from the tree to produce a smaller tree, T'. We inductively produce a perfect matching of the nodes in S on T' and use the same matching for T. If $v \in S$, we instead consider v's parent node, $par(v)$.

If $par(v)$ has an even number of children, we match every pair of sibling nodes with edges via their parent. Every edge connected to these children is used only once. On the other hand, if $par(v)$ has an odd number of children, we match every pair of sibling nodes with edges via their parent and match the last child with the parent. Every edge connected to these children is also used only once. We then remove all matched nodes and their edges from the tree T to produce a smaller tree T'.

Notice that if $par(v)$ has an even number of children and it belongs to S, it remains in T'; if $par(v)$ has an even number of children and it does not belong to S, it will remain out of T'. If $par(v)$ has an odd number of children and it belongs to S, it remains out of T'; if $par(v)$ has an odd number of children and it does not belong to S, it will remain in T' (and also in S) on behalf of node u that is matched with $par(v)$. The reason for $par(v)$ to remain in T' is that u shall obtain its real matching pair via $par(v)$ in the future in this case.

Then, we inductively match the rest of S on T' until all nodes are matched or only the root (sink) is left. In this process, the desired matching is produced and each edge in T is used at most once. □

Lemma 3. *Let K_i be the total distance of matchings in stage $i+1$. Then, the expected cost of that stage, denoted by G_{i+1}, is the same as K_i.*

Proof. The objective of the $(i+1)$th stage is to find the perfect matching in S_i and match them. The cost of the process consists of the total cost of transferring weight of matched nodes from nonfusion nodes to their fusion nodes and the total cost of fusing data at fusion nodes. Suppose we match u and v with weight $w_i(u)$ and $w_i(v)$. The fusion cost $q(w_i(u), w_i(v))$ is independent of which node is chosen to be the fusion node since the fusion cost is only determined by the fusion function itself and its inputs. However, the transmission cost is different when using different fusion nodes. If we select v as the fusion node, we need to transport u's data to v. This introduces a transmission cost of $w_i(u)c(e)$. On the other hand, if we select u as the fusion node, the transmission cost will be $w_i(v)c(e)$. Let $G_{i+1}(u, v)$ denote the cost of matching u and v, and G_{i+1} denote the total cost in the $(i+1)$th stage. The expected value of $G_{i+1}(u, v)$ is given by

$$\begin{aligned}
&\mathbf{E}[G_{i+1}(e)]\Big|_{e=(u,v)} \\
&\quad = P(fuse\ at\ u)\Big(f_i(e) + t_i(u \leftarrow v)\Big) \\
&\qquad + P(fuse\ at\ v)\Big(f_i(e) + t_i(u \rightarrow v)\Big) \\
&\quad = \frac{w_i^2(u)}{w_i^2(u) + w_i^2(v)} \Big(q\Big(w_i(u), w_i(v)\Big) + w_i(v)c(e)\Big) \\
&\qquad + \frac{w_i^2(v)}{w_i^2(u) + w_i^2(v)} \Big(q\Big(w_i(u), w_i(v)\Big) + w_i(u)c(e)\Big) \\
&\quad = q\Big(w_i(u), w_i(v)\Big) + \alpha\Big(w_i(u), w_i(v)\Big)c(e),
\end{aligned}$$

where

$$\alpha(w_i(u), w_i(v)) = \frac{w_i(u)w_i(v)(w_i(u) + w_i(v))}{w_i^2(u) + w_i^2(v)}.$$

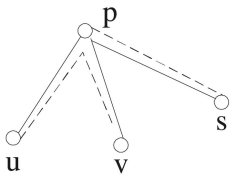


Fig. 2. An example of perfect matching using edges in T_i^*. Here, the dashed lines connect the matching pairs and the solid lines are the available communication links.

Notice that this expected cost is exactly $K_i(u,v)$, defined in (5). It follows that the expected cost of the $(i+1)$th stage is equal to the total K_i-distance of the matchings found. Let X_{i+1} denote the set of matched edges in $(i+1)$th stage, thus

$$\begin{aligned}\mathbf{E}[G_{i+1}] =& \mathbf{E}\left[\sum_{e\in X_{i+1}} G_{i+1}(e)\right] \le \sum_{e\in X_{i+1}} \mathbf{E}[G_{i+1}(e)] \\ =& \sum_{e\in X_{i+1}} K_i(e) = K_i.\end{aligned}$$

□

Next, we examine the relationship between K_i and C_i^*.

Lemma 4. *The total distance of matchings in the* $(i+1)$th *stage satisfies* $K_i \le (\frac{5}{4})C_i^*$.

Proof. Let T_i^* denote the optimal tree for S_i. By matching the nodes in S_i in the proper way described in Lemma 2, we can get a perfectly matching X_{i+1}, which guarantees that we use only edges in T_i^* and use no edge more than once. In T_i^*, let $w_{T_i^*}(\cdot)$ denote the total node weight in this optimal tree; only leaf nodes have $w_{T_i^*}(u) = w_i(u)$. Intermediate nodes satisfy $w_{T_i^*}(u) \ge w_i(u)$ due to data aggregation. To illustrate the remaining proof, we will use Fig. 2 as an example. This figure is a subtree of T_i^* with four nodes, where p is the parent, u, v, and s are children, and solid lines are the edges. According to the matching scheme described in Lemma 2, we can get two matches: (s,p) as the parent-child matching and (u,v) as the sibling-sibling matching. Below, we enumerate these two different kinds of matchings and analyze their fusion cost and transmission cost individually.

For all sibling-sibling matching (u,v),

$$K_i(u,v) = q\Big(w_i(u), w_i(v)\Big) + \alpha\Big(w_i(u), w_i(v)\Big)c(u,v) \tag{9}$$

includes two parts. The first part of $K_i(u,v)$ is the fusion cost $f_i(u,v) = q(w_i(u), w_i(v))$. We bound it by the fusion cost of edges (u,p) and (v,p) at p in T_i^* as

$$\begin{aligned}&q\Big(w_i(u), w_i(v)\Big) \\ &= q\Big(w_i(u), 0\Big) + q\Big(w_i(u), w_i(v)\Big) \\ &\le q\Big(w_i(u), w_i(p)\Big) + q\Big(w_i(v), \max(w_i(u), w_i(p))\Big) \\ &\le q\Big(w_i(u), w_i(p)\Big) + q\Big(w_i(v), \widetilde{w}_i(p)\Big) \\ &\le q\Big(w_{T_i^*}(u), \widetilde{w}_{T_i^*}(p)\Big) + q\Big(w_{T_i^*}(v), \widetilde{\widetilde{w}}_{T_i^*}(p)\Big).\end{aligned} \tag{10}$$

Here, $w_i(p)$ is the information amount at p before the fusion of u and p, whereas $\widetilde{w}_i(p)$ denotes the information amount at p after the fusion of u and p and before the fusion of v and p. Similarly, $\widetilde{w}_{T_i^*}(p)$ and $\widetilde{\widetilde{w}}_{T_i^*}(p)$, respectively, denote the information amount at p before the fusion of u and p and after the fusion of u and p but before the fusion of v and p in T_i^*. The first line in (10) is the fusion cost in $K_i(u,v)$ and the last line is the fusion cost of (u,p) and (v,p) in T_i^*.

The second part of $K_i(u,v)$ is the expectation of transmission cost

$$t_i(u,v) = \frac{w_i(u)w_i(v)(w_i(u)+w_i(v))}{w_i^2(u)+w_i^2(v)} \cdot c(u,v).$$

We will bound it by the transmission cost of u and v to p in T_i^*. Since $w_i^2(u) + (2w_i(u) - w_i(v))^2 \ge 0$, we have

$$\begin{aligned}&\frac{w_i(u)w_i(v)(w_i(u)+w_i(v))}{w_i^2(u)+w_i^2(v)} \le \frac{5}{4}w_i(u), \\ \text{and } &\frac{w_i(u)w_i(v)(w_i(u)+w_i(v))}{w_i^2(u)+w_i^2(v)} \le \frac{5}{4}w_i(v).\end{aligned} \tag{11}$$

As (u,v) are matched together using edges (u,p) and (v,p), we have $c(u,v) = c(e_{up}) + c(e_{vp})$. Therefore,

$$\begin{aligned}&\frac{w_i(u)w_i(v)(w_i(u)+w_i(v))}{w_i^2(u)+w_i^2(v)} \cdot c(u,v) \\ &\le \frac{5}{4}\min\Big(w_i(u), w_i(v)\Big) \cdot \Big(c(e_{up}) + c(e_{vp})\Big) \\ &\le \frac{5}{4}\Big(w_i(u)c(e_{up}) + w_i(v)c(e_{vp})\Big) \\ &\le \frac{5}{4}\Big(w_{T_i^*}(u)c(e_{up}) + w_{T_i^*}(v)c(e_{vp})\Big).\end{aligned} \tag{12}$$

The first line in (12) is the transmission cost in $K_i(u,v)$, and the last line is $\frac{5}{4}$ times the transmission cost of (u,p) and (v,p) in T_i^*.

For all parent-child matching, like (s,p), $K_i(s,p)$ again includes two parts. We bound them in the same way as we did for the sibling-sibling matching. Toward this end, we conclude that for any node pair (u,v) in S_i, the total distance $K_i(u,v)$ is no more than $\frac{5}{4}$ times the cost of merging u and v in the optimal tree T_i^*. Therefore,

$$\begin{aligned}K_i &= \sum_{e=(u,v)\in X_{i+1}} K_i(u,v) \\ &\le \sum_{e=(u,v)\in T_i^*} \frac{5}{4}\Big(q\Big(w_{T_i^*}(u), \widetilde{w}_{T_i^*}(v)\Big) + w_{T_i^*}(u)c(e)\Big) \\ &= \Big(\frac{5}{4}\Big)C_i^*.\end{aligned} \tag{13}$$

□

The above lemmas lead to the following theorem:

Theorem 1. *The approximation ratio of MFST is no more than* $\frac{5}{4}\log(k+1)$ *to the optimal.*

Proof. The expected cost of MFST is equal to the sum of the expected costs of all stages. This yields

$$\mathbf{E}[G] = \mathbf{E}\left[\sum_{i=1}^{\log(k+1)} G_i\right] \le \sum_{i=1}^{\log(k+1)} \mathbf{E}[K_{i-1}].$$

Using Lemmas 1 and 4, we conclude

$$\mathbf{E}[G] \le \sum_{i=1}^{\log(k+1)} \frac{5}{4}\mathbf{E}\left[C^*_{i-1}\right] \le \left(\frac{5}{4}\right) \sum_{i=1}^{\log(k+1)} C^*_0 = \frac{5}{4}\log(k+1)C^*.$$

□

We remark that, for the simplified case analyzed in [12], MFST can achieve the same approximation ratio. There, the authors assume that 1) each node has the same amount of original information and 2) the amount of information after fusion is just a function of the number of incoming nodes. Under these assumptions, for any node u, v in S_i, we have $w_i(u) = w_i(v)$. Therefore,

$$\frac{w_i(u)w_i(v)(w_i(u) + w_i(v))}{w_i^2(u) + w_i^2(v)} = w_i(u).$$

In other words, (13) can be improved to $K_i \le C_i^*$. As a result, the approximation ratio can be improved to $\log(k+1)$ as derived in [12]. Although in this sense MFST is a generalization of the algorithm described in [12], the generalized assumptions and introduction of fusion cost involve significantly different design and proof of MFST.

5 Comparison with Other Algorithms

In this section, we perform a comparison between MFST and other algorithms such as SPT, MST, and SLT. Recall that SLT is a routing algorithm proposed in [23], targeted at simultaneously approximating both MST and SPT for a given node. SLT is used in [16] as an approximation solution to solve the aggregation tree problem. From the comparison, we will conclude that MFST can better approximate the optimal solution with different correlation coefficients.

5.1 Scenario

Consider a sensor network where nodes are deployed as an $N \times N$ square grid, where only N nodes in the left column are sources. The sink is located at the rightmost bottom corner. We assume that each source generates unit data I_0 that is to be gathered at the sink. Data packets will be aggregated when they encounter on their paths to the sink. The fusion cost at the sink is naturally ignored from the total routing cost since a sink usually has abundant energy.

Nodes in the grid can only communicate with their neighbors. The cost for transmitting one bit of data between neighboring nodes is assumed to be c_0. Let q_0 be the cost for fusing two source data packets of I_0. For fusion to be meaningful, the fusion cost q_0 shall be smaller than the transmission cost $c_0 I_0$. Otherwise, intermediate nodes will prefer forwarding data directly instead of doing fusion for energy saving.

Under this setup, we compare four routing schemes, namely SPT, MST, SLT, and MFST. We consider two extreme scenarios to demonstrate their performance differences:

- In the first scenario, the gathered data are identical for every sensor. In other words, the data aggregation ratio among sensors is 100 percent.
- In the second scenario, there is no redundancy among the information gathered by different sensors, i.e., data aggregation ratio is 0.

5.2 When Data Aggregation Ratio is 100 Percent

In this scenario, the routes established by four algorithms are depicted in Fig. 3. Since it reaches the highest aggregation ratio, at each intermediate routing node, two completely redundant data packets I_0 are aggregated without increasing the data amount, resulting in another I_0 packet. In this case, it is easy to verify that MST is the optimal solution while SPT is the worst one. In SPT, the distance from each source node to the sink is $(N-1)$ hops. In MST, the farthest source is $2(N-1)$ hops from the sink. Since $2(N-1) < (1+\sqrt{2})(N-1)$, according to [23], SLT will degrade into MST for this scenario. Since MFST is a randomized algorithm, we will only analyze its best-case and worst-case performance.

In the following, we will examine the cost of MST (SLT), SPT, and MFST for this network. For the sake of simplicity, we assume $I_0 = 1$.

The cost for MST, C_{MST}, can be derived as

$$C_{MST} = \sum_{i=1}^{N-1}(c_0 + q_0) + \sum_{i=1}^{N-1} c_0 = (2c_0 + q_0)(N-1). \quad (14)$$

The cost for SPT is

$$\begin{aligned} C_{SPT} &= \sum_{i=1}^{N-1} ic_0 + \sum_{i=1}^{N-1} c_0 + \sum_{i=1}^{N-2} q_0 \\ &= c_0\frac{N(N-1)}{2} + c_0(N-1) + q_0(N-2). \end{aligned} \quad (15)$$

For MFST, since data aggregation ratio is 100 percent, matching between adjacent nodes is perfect. When $N = 2^n$, all sources connect to one node which in turn will connect to the sink via the shortest path as shown in Fig. 3c. Hence, the cost is the same as MST, which is the optimal. When $N = 2^n - 1$, the perfect matching algorithm will divide the sources into n clusters with node numbers $2^0, 2^1, \ldots, 2^{n-1}$, respectively. Nodes within each cluster will be connected to a center, which is connected to the sink via the shortest path as shown in Fig. 3d. Since the largest number of shortest paths are employed as compared with the cases when $2^{n-1} \le N \le 2^n - 1$, it is indeed the worst case for MFST. Because MFST is a randomization algorithm, different centers in each cluster will induce different paths and consequently have different total costs. However, we can examine the worst-case scenario for the randomization when the center is selected as the farthest node to the sink in each cluster. In this worst realization,

$$\begin{aligned} C^{Worst}_{MFST} &= \sum_{i=1}^{n-1}\left(2^i - 1\right)(c_0 + q_0) \\ &\quad + \sum_{i=1}^{n-1}\left(2^{i+1} - 2\right)c_0 + \left(\sum_{i=1}^{N-1} c_0 + \sum_{i=1}^{n-2} q_0\right). \end{aligned} \quad (16)$$

The first component of (16) represents the fusion and transmission costs in each cluster, the second component summarizes the transmission costs from center nodes to the

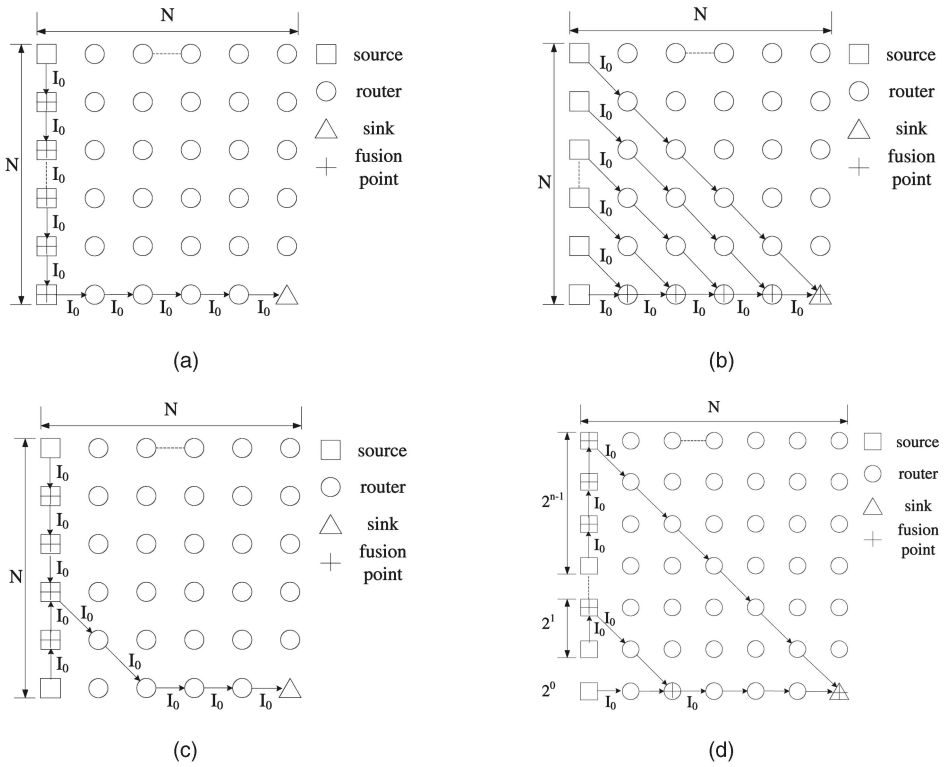


Fig. 3. A data aggregation tree for MST, SPT, and MFST when the data aggregation ratio is 100 percent. (a) MST (SLT). (b) SPT. (c) MFST best case. (d) MFST worst case.

fusion points on the bottom line, and the third component captures the fusion and transmission costs on the bottom line. Since $N = 2^n - 1$, the above equation can be simplified as

$$C_{MFST}^{Worst} = (4N - 3n - 1)c_0 + (N - 2)q_0. \quad (17)$$

Comparing the worst-case cost of MFST in (17) with that of MST, we have

$$\frac{C_{MFST}^{Worst}}{C_{MST}} = \frac{(4N - 3n - 1)c_0 + (N - 2)q_0}{2c_0(N - 1) + q_0(N - 1)} < 2.$$

Evidently, provided that N exceeds a certain threshold, even the worst case MFST always outperforms SPT, and with increasing N, their difference is unbounded as demonstrated in the equation below:

$$C_{SPT} - C_{MFST}^{Worst} = \frac{c_0}{2}(N^2 - 7N + 6\log(N + 1)) \geq 0 \quad (\text{when } N \geq 3).$$

If the fusion cost at the sink is not ignored, the same conclusions can still be drawn by following a similar analysis. We remark that MFST can actually approximate to the optimal solution within a factor of 2 in this case. Simultaneously, it is always better than SPT, and their difference is unbounded.

5.3 When Data Aggregation Ratio is 0

Since there is no redundancy, the amount of data will not be reduced at each fusion point. In this case, SPT is the optimal solution and MST is the worst solution. As explained before, SLT is the same as MST in this network. MFST, derived by our randomized approximation algorithm, lies between them. Similarly to the previous section, we can conclude that MFST also outperforms MST (and, hence, SLT) in this extreme scenario. The cost difference between MFST and MST is unbounded. When $N > 4$, the approximation ratio of MFST to SPT, i.e., the optimal solution, is less than 3.

The above analysis concludes that MFST can trade off MST (SLT) and SPT in different scenarios while SPT and MST (SLT) can only excel in certain extreme cases. Indeed, the data aggregation ratio is usually between 0 and 1. In the next section, we will give extensive simulation results to illustrate the outperforming of MFST under more general system setups.

6 Simulation Study

In this section, we present an extensive set of simulations to evaluate the performance of our proposed routing algorithm. For sensor nodes randomly deployed in a 2D field, we compare the performance of MFST with other routing algorithms based on SPT, MST, and SLT. The impact of network connectivity, correlation coefficient, and unit fusion cost on different algorithms are carefully studied.

Concurring with our design goal and analysis of the MFST, our key finding of the experiments is that MFST can adapt itself to a wide range of data correlation among sensor nodes and fusion costs. While other algorithms may achieve better performance in some extreme cases, they suffer from varying conditions and, hence, perform poorly in general scenarios.

6.1 Simulation Environment

We consider 100 sensors uniformly distributed in a square region of size $50\ \text{m} \times 50\ \text{m}$. We assume that each node

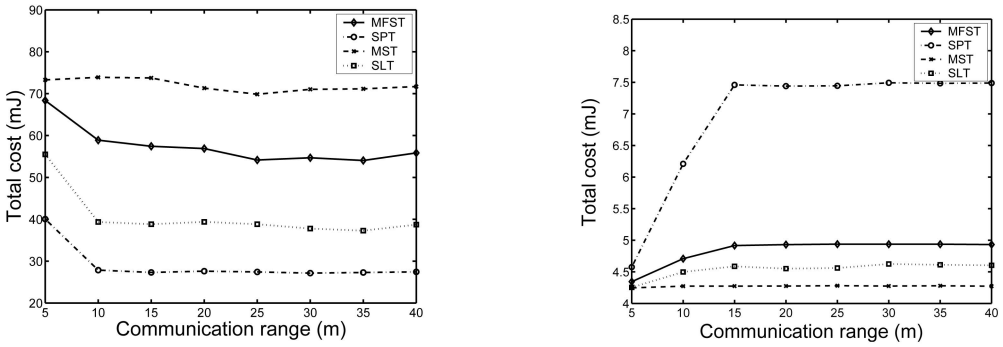


Fig. 4. Total cost as a function of network connectivity (Fusion cost is zero, $r_c = 5 \sim 40\ m$). (a) $r_s = 0.1\ m$ to simulate $\rho \to 0$. (b) $r_s = 4{,}000\ m$ to simulate $\rho \to 1$.

produces one unit data (400 bytes) and sends it to the sink located at the bottom-right corner. All sensors act as both sources and routers. We also performed a set of experiments with different numbers of sensors and different sizes of fields; the results are similar and omitted here.

We assume the maximal communication radius is r_c, i.e., if and only if two sensor nodes are within r_c, there exists a communication link between them and, hence, an edge in graph G. By varying r_c, we can control the network connectivity and, hence, the topology of the network. We instantiate unit transmission cost on each edge, $c(e)$, using the first order radio model presented in [5]. The transmission cost for sending I amount of information from one node to another node d distance away is given by $I(\beta d^{\gamma} + \varepsilon)$ when $d < r_c$. We set $\gamma = 2$ and $\beta = 100\ pJ/bit/m^2$ to calculate the energy consumption on the transmit amplifier. Here, ε denotes energy consumption per bit on the transmitter and receiver circuit. The typical value of ε is from 10 to $100\ nJ/bit$ [19] and is set to $40\ nJ/bit$ in our simulation.

To possibly accommodate a wide range of scenarios, we abstract data redundancy among two sensor nodes using a single value ρ, termed *correlation coefficient*. ρ will determine the amount of data reduction due to aggregation. Given the correlation coefficient between node u and v, if their parent node fuses their data together, we assume that the weight of the parent node equals

$$\max(w(u), w(v)) + \min(w(u), w(v))(1 - \rho(u, v)),$$

where $w(u)$ and $w(v)$ are the weights of u and v *before* data fusion.

The correlation model employed here is an approximated spatial model, where the correlation coefficient decreases with the distance between two nodes provided that they are within a correlation range r_s. If two nodes are more than r_s distance apart, the correlation coefficient is simply 0. Otherwise, the correlation coefficient is $\rho = 1 - d/r_s$, where d denotes the distance between the nodes. By varying the correlation range r_s, we can control the average correlation coefficient of the network.

In order to distinguish the correlation between data originated from two nodes and that among aggregated data, we use a "forgetting" factor on the correlation coefficient among aggregated data. For example, the correlation between aggregated information at two parent nodes is only a fraction of their own data correlation calculated according to their distance. Throughout the simulation, we use a factor of 0.8. A set of other values are also studied, which lead to similar results and are omitted here.

For fusion cost, in the simulation, we assume that $q(x, y) = \omega \cdot (x + y)$, where ω denotes the fusion cost of unit data. In other words, fusion cost is linear with the total amount of data to be fused.

6.2 Impact of Network Connectivity

Since r_c denotes the transmission range of a node, by varying r_c, we can control the connectivity of the network. Naturally, different connectivity (node degrees) will affect the behavior of different routing algorithms.

6.2.1 Without Fusion Cost

In this set of experiments, we first disregard fusion cost. Notice that MFST is an algorithm designed with fusion cost. By disregarding fusion cost, we can validate its performance in a scenario that actually favors those that dedicate their optimization solely to transmission cost. Surprisingly, our results show that MFST has comparable performance with SLT while outperforming MST and SPT in varying scenarios.

Fig. 4 summarizes the results. Two extreme cases are studied. In both cases, r_c is varied from 5 m to 40 m, denoted by the x-axis. In the first case shown in Fig. 4a, r_s is set to $0.1\ m$; in the second case shown in Fig. 4b, r_s is set to $4{,}000\ m$. According to the correlation model $\rho = 1 - d/r_s$, when $d < r_s$, a very small r_s essentially eliminates the correlation among sensors ($\rho \to 0$), while an extremely large r_s makes the sensed data completely redundant ($\rho \to 1$). Our simulation results correspond to those described in [15]. In a weakly correlated network, SPT is the optimal solution while MST is the worst. On the contrary, in a strongly correlated network, MST is the optimal solution and SPT is the worst. Similarly to SLT, MFST can balance SPT and MST and has comparable performance with SLT even though balancing SPT and MST is not the main objective of MFST.

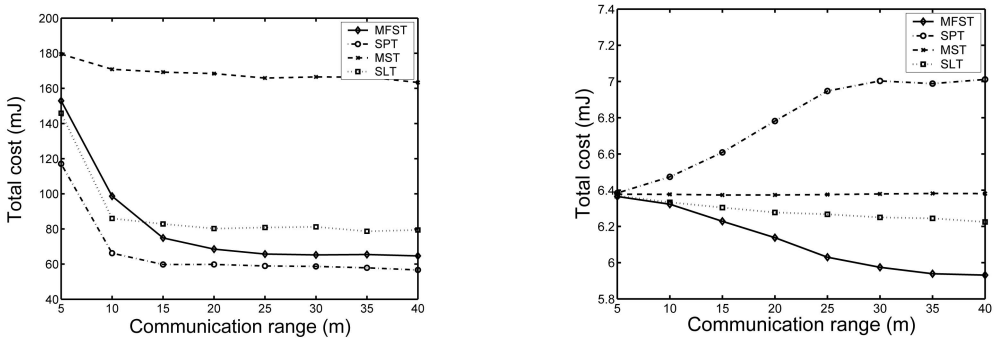


Fig. 5. Total cost as a function of network connectivity ($\omega = 15\ nJ/bit,\ r_c = 5 \sim 40\ m$). (a) $r_s = 0.1\ m$ to simulate $\rho \to 0$. (b) $r_s = 4{,}000\ m$ to simulate $\rho \to 1$.

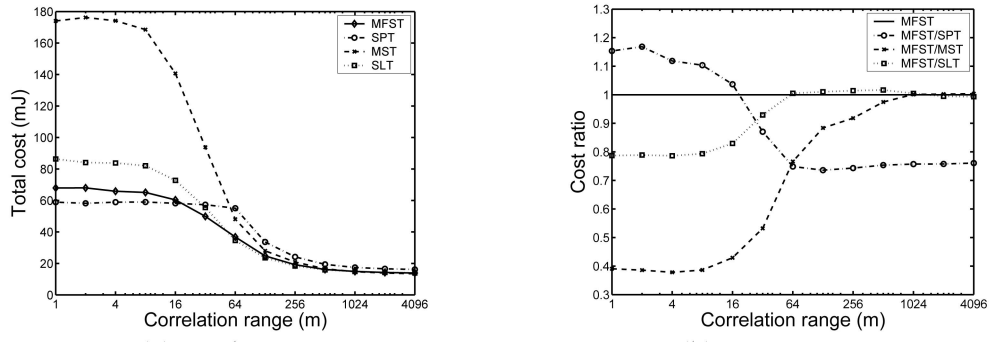


Fig. 6. (a) Total cost and (b) cost ratio as a function of correlation coefficient ($r_c = 30\ m$, $\omega = 15\ nJ/bit$, $r_s = 1 \sim 4{,}000\ m$).

6.2.2 With Fusion Cost

In this set of simulations, we include fusion cost and study its impact on the performance of routing algorithms. We set ω, the fusion cost for unit data, to be $15\ nJ/bit$. Again, the cases for $r_s = 0.1\ m$ and $r_s = 4{,}000\ m$ are studied and the results are depicted in Fig. 5.

Compared with the results shown in Fig. 4, as $\rho \to 0$ (illustrated in Fig. 5a), the performance of MFST is the closest one to the optimal solution SPT. The approximation ratio is below 1.5 through the range of communication radius. Notably, with the increase of communication radius, the approximation ratio gets smaller. This can be explained as follows: In a network with poor correlation, nodes shall send data directly to the routing nodes near the sink instead of relaying information through multiple hops, as fusion at each hop is not efficient in reducing the data amount. As MFST explicitly considers fusion cost, this phenomenon can be captured and exploited. On the contrary, SLT results in a fixed routing structure according to network topology and a fixed approximation ratio to MST and SPT and, hence, cannot adapt to the change of data correlation. Therefore, when ρ approaches zero, SLT cannot recognize the advantage of transmitting over direct links and results in poor performance. When $\rho \to 1$ as illustrated in Fig. 5b, MFST performs better than all other algorithms. This is due to the waste of energy for fusion at every node in MST and the waste of transmission energy in SPT for using shortest paths with long hop distance. In contrast, MFST and SLT can balance between data aggregation and direct transmission and thus produce better performance. Since SLT gets the benefit implicitly and MFST explicitly targets this balance, we observe that the cost of MFST decreases faster than SLT. Longer transmission range and thus better network connectivity of the network is in favor of MFST as it can employ more direct shortest paths to prevent unnecessary fusion cost at each node.

6.3 Impact of Correlation Coefficient

Next, we fix the transmission range of the sensor nodes and study the impact of correlation coefficient on the routing performance. Here, we set r_c to be 30 m and the unit fusion cost ω is set to be $15\ nJ/bit$. We increase r_s from 1 to $4{,}000\ m$, which corresponds to varying ρ from 0 to 1. Fig. 6a illustrates the total costs of the four algorithms.

The costs of all algorithms decrease with the increase of ρ, the correlation coefficient. This exemplifies that data aggregation in sensor networks can greatly benefit the routing performance by reducing redundancy among correlated data. When ρ is small, SPT performs well. However, it does not benefit from the increase of ρ as the total cost only incurs a slight drop. Although both MFST and SLT are more balanced than MST and SPT, we observe that MFST performs much better than SLT, especially when $r_s < 64$ m. The main reason is that MFST recalculates nodes' weights in every stage to get perfect matching and, thus, can adapt to the correlation among nodes.

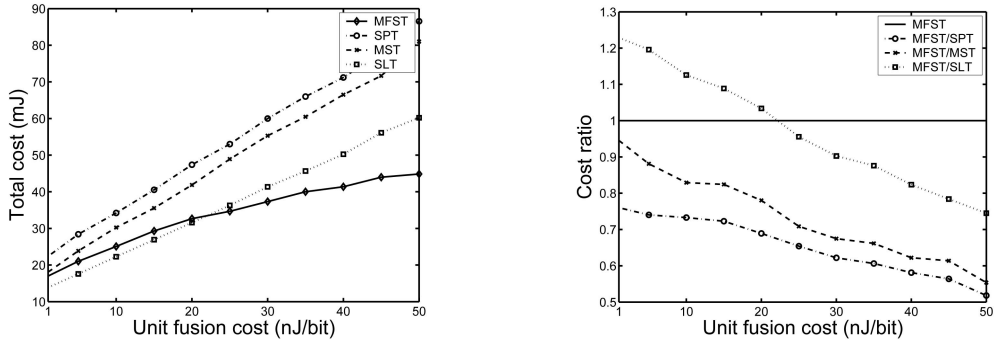


Fig. 7. (a) Total cost and (b) cost ratio as a function of unit fusion cost ($r_c = 30\ m$, $r_s = 20\ m$, $\omega = 1 \sim 50\ nJ/bit$).

Fig. 6b shows the cost ratio of MFST to other algorithms. As we can see, MFST achieves the optimal trade-off over the entire range of correlations. It can save nearly 20 percent of energy compared with SLT, when ρ is small, while retaining almost the same performance as SLT when ρ is large. On the other hand, MFST can save more than 60 percent energy compared with MST, when ρ is small, and maintaining comparable performance when ρ is large. Finally, compared with SPT, MFST can save about 25 percent of energy when ρ is large at the cost of spending slightly more energy (less than 15 percent) when ρ is small. As the correlation among nodes often vary from application to application, from node to node, and even from time to time, only a general algorithm such as MFST optimized for a wide range of the value of ρ can accommodate versatile scenarios.

6.4 Impact of Unit Fusion Cost

Since MFST includes fusion cost in the routing constraint, it will evidently outperform other algorithms with the increase of fusion cost. In this set of experiments, we study the performance gain of MFST as compared with other algorithms when the unit fusion cost is increased.

Fig. 7 illustrates the results when ω is increased from 1 nJ/bit to 50 nJ/bit. The total cost of SPT, MST, and SLT exhibits linearity with ω as shown in Fig. 7a. However, MFST shows logarithmic increase with ω. The reason is that SPT, MST, and SLT generate routes only based on network topology and do not take fusion cost into account. Therefore, the resulting routing trees are fixed and, hence, the total cost will increase linearly with ω. Since MFST explicitly exploits the fusion cost when optimizing routes, it can best adjust to the change of fusion cost. Fig. 7b clearly shows that with increasing ω, MFST can continually distant itself from others.

As described in Section 2, the fusion cost per unit data may vary widely from network to network. As an example, a temperature surveillance sensor network has little fusion cost to calculate the max, min, or average temperature. On the other hand, a wireless video sensor network may incur significant fusion cost when performing image fusion. Our experiments show that MFST can adapt well to a wide range of fusion costs and, hence, is applicable to a variety of applications.

7 Conclusion

In this paper, we propose a randomized algorithm, termed *Minimum Fusion Steiner Tree (MFST)*, for routing correlated data in sensor networks. MFST incorporates the missing dimension of fusion cost into the problem formulation and guarantees an approximation ratio of $\frac{5}{4}\log(k+1)$ to the optimal solution. Analytical and experimental results show that MFST adapts well to varying network conditions including network topology, fusion cost, and the degree of correlation. Therefore, MFST provides a feasible general routing scheme for wireless sensor networks facing various applications, unpredictable environments, and time evolving reconfigurations.

As an ongoing effort, we are designing an online algorithm based on MFST that can be executed in a distributed manner by sensor nodes. At the same time, we are investigating the robustness of the proposed algorithm and possible enhancements.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments that helped them improve the technical quality of the paper. This work is partially supported by US National Science Foundation grants IIS-0121297 and IIS-0326505. The work was done while H. Luo was a visiting scholar with the Department of Computer Science and Engineering at the University of Texas at Arlington.

References

[1] C. Chong and S. Kumar, "Sensor Networks: Evolution, Opportunities, and Challenges," *Proc. IEEE,* vol. 91, no. 8, pp. 1247-1256, Aug. 2003.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Comm. Magazine,* vol. 40, no. 8, pp. 102-114, Aug. 2002.

[3] W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive Protocol for Information Dissemination in Wireless Sensor Networks," *Proc. ACM MobiCom,* pp. 174-185, Aug. 1999.

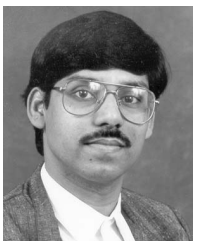
[4] A.A. Ahmed, H. Shi, and Y. Shang, "A Survey on Network Protocols for Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Information Technology: Research and Education,* Aug. 2003.

[5] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," *Proc. 33rd Ann. Hawaii Int'l Conf. System Sciences,* Jan. 2000.

[6] B. Krishnamachari, D. Estrin, and S. Wicker, "Impact of Data Aggregation in Wireless Sensor Networks," *Proc. 22nd Int'l Conf. Distributed Computing Systems,* pp. 575-578, July 2002.
[7] A. Scaglione and S.D. Servetto, "On the Interdependence of Routing and Data Compression in Multi-Hop Sensor Networks," *Proc. ACM MobiCom,* Sept. 2002.
[8] S. Pattem, B. Krishnamachari, and R. Govindan, "The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks," *Proc. Third Int'l Symp. Information Processing in Sensor Networks,* Apr. 2004.
[9] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang, "A Two-Tier Data Dissemination Model for Large-Scale Wireless Sensor Networks," *Proc. ACM MobiCom,* pp. 148-159, Sept. 2002.
[10] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of Network Density on Data Aggregation in Wireless Sensor Networks," *Proc. 22nd Int'l Conf. Distributed Computing Systems,* July 2002.
[11] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed Diffusion for Wireless Sensor Networking," *IEEE/ACM Trans. Networking,* vol. 11, no. 1, pp. 2-16, Feb. 2003.
[12] A. Goel and D. Estrin, "Simultaneous Optimization for Concave Costs: Single Sink Aggregation or Single Source Buy-at-Bulk," *Proc. ACM-SIAM Symp. Discrete Algorithms,* Jan. 2003.
[13] W. Zhang and G. Cao, "Optimizing Tree Reconfiguration for Mobile Target Tracking in Sensor Networks," *Proc. IEEE INFOCOM,* Mar. 2004.
[14] W. Zhang and G. Cao, "DCTC: Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks," *IEEE Trans. Wireless Comm.,* vol. 3, no. 5, pp. 1689-1701, July 2004.
[15] R. Cristescu, B. Beferull-Lozano, and M. Vetterli, "On Network Correlated Data Gathering," *Proc. IEEE INFOCOM,* Mar. 2004.
[16] P.V. Rickenbach and R. Wattenhofer, "Gathering Correlated Data in Sensor Networks," *Proc. ACM Workshop Foundations of Mobile Computing,* Oct. 2004.
[17] Y. Yu, B. Krishnamachari, and V. Prasanna, "Energy-Latency Tradeoff for Data Gathering in Wireless Sensor Networks," *Proc. IEEE INFOCOM,* Mar. 2004.
[18] S. Lindsey and C.S. Raghavendra, "PEGASIS: Power-Efficient Gathering in Sensor Information Systems," *Proc. IEEE Aerospace Conf.,* Mar. 2002.
[19] A. Wang, W.B. Heinzelman, A. Sinha, and A.P. Chandrakasan, "Energy-Scalable Protocols for Battery-Operated Microsensor Networks," *J. VLSI Signal Processing,* vol. 29, no. 3, pp. 223-237, Nov. 2001.
[20] H. Luo, J. Luo, S.K. Das, and Y. Liu, *Energy Efficient Routing with Adaptive Data Fusion in Sensor Networks,* technical report, Computer Science and Eng. Dept., Univ. of Texas at Arlington, Mar. 2005.
[21] S.S. Pradhan and K. Ramchandran, "Distributed Source Coding Using Syndromes (DISCUS): Design and Construction," *IEEE Trans. Information Theory,* vol. 49, no. 3, pp. 626-643, Mar. 2003.
[22] A. Goel and K. Munagala, "Balancing Steiner Trees and Shortest Path Trees Online," *Proc. 11th ACM-SIAM Symp. Discrete Algorithms,* Jan. 2000.
[23] B. Raghavachari, S. Khuller, and N. Young, "Balancing Minimum Spanning and Shortest Path Trees," *Proc. Fourth ACM-SIAM Symp. Discrete Algorithms,* Jan. 1993.
[24] A. Meyerson, K. Munagala, and S. Plotkin, "Cost-Distance: Two Metric Network Design," *Proc. 41st Ann. Symp. Foundations of Computer Science,* Nov. 2000.
[25] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity,* pp. 247-266, 1998.
[26] R. Motwani and P. Raghavan, *Randomized Algorithms,* pp. 148-159, 1995.

Hong Luo received the BS and MS degrees from the Beijing University of Posts and Telecommunications in 1990 and 1993, respectively. She is an associate professor with the College of Computer Science and Technology at Beijing University of Posts and Telecommunications. Her research interests are wireless networking, sensor networks, and communication software. She is a member of the IEEE.

Yonghe Liu received the BS and MS degrees from Tsinghua University in 1998 and 1999, respectively, and the PhD degree from Rice University in 2004. He is an assistant professor at the Department of Computer Science and Engineering, the University of Texas at Arlington. His research interests are wireless networking, sensor networks, security, and system integration. He is a member of the IEEE.



Sajal K. Das is a professor of computer science and engineering and also the founding director of the Center for Research in Wireless Mobility and Networking (CReWMaN) at The University of Texas at Arlington (UTA). His current research interests include sensor networks, resource and mobility management in wireless networks, mobile and pervasive computing, wireless multimedia and QoS provisioning, mobile Internet architectures and protocols, grid computing and applied game theory. He has published more than 350 research papers in these areas, holds four US patents in wireless internet and mobile networks. He received Best Paper Awards at ACM MobiCom '99, ICOIN '02, ACM MSwiM '00 and ACM/IEEE PADS '97. He is also a recipient of UTA's Outstanding Faculty Research Award in Computer Science (2001 and 2003), College of Engineering Research Excellence Award (2003), and University Award for Distinguished record of Research (2005). He serves as the editor-in-chief of the *Pervasive and Mobile Computing* journal and associate editor of the *IEEE Transactions on Mobile Computing*, *ACM/SPringer Wireless Networks*, and the *IEEE Transactions on Parallel and Distributed Systems*. He has served as general or program chair and TPC member of numerous IEEE and ACM conferences. He is the vice chair of IEEE TCCC and TCPP Executive Committees. He is a member of the IEEE.