Monitoring Quality Maximization through Fair Rate Allocation in Harvesting Sensor Networks

Weifa Liang, *Senior Member*, *IEEE*, Xiaojiang Ren, *Student Member*, *IEEE*, Xiaohua Jia, *Fellow*, *IEEE*, and Xu Xu

Abstract—In this paper, we consider an energy harvesting sensor network where sensors are powered by reusable energy such as solar energy, wind energy, and so on, from their surroundings. We first formulate a novel monitoring quality maximization problem that aims to maximize the quality, rather than the quantity, of collected data, by incorporating spatial data correlation among sensors. An optimization framework consisting of dynamic rate weight assignment, fair data rate allocation, and flow routing for the problem is proposed. To fairly allocate sensors with optimal data rates and efficiently route sensing data to the sink, we then introduce a weighted, fair data rate allocation and flow routing problem, subject to energy budgets of sensors. Unlike the most existing work that formulated the similar problem as a linear programming (LP) and solved the LP, we develop fast approximation algorithms with provable approximation ratios through exploiting the combinatorial property of the problem. A distributed implementation of the proposed algorithm is also developed. The key ingredients in the design of algorithms include a dynamic rate weight assignment and a reduction technique to reduce the problem to a special maximum weighted concurrent flow problem, where all source nodes share the common destination. We finally conduct extensive experiments by simulation to evaluate the performance of the proposed algorithm. The experimental results demonstrate that the proposed algorithm is very promising, and the solution to the weighted, fair data rate allocation and flow routing problem is fractional of the optimum.

Index Terms—Energy harvesting sensor networks, monitoring quality maximization, fair rate allocation optimization, time-varying energy replenishment, maximum weighted concurrent flow problem, approximation algorithms, combinatorial optimization problem

1 Introduction

In conventional wireless sensor networks, sensors are typically powered by batteries and, thus, have limited life time [2]. A promising approach to achieve "perpetual operations" for sensor networks is to harvest various energy from its surrounding environments such as solar energy, wind energy, electromagnetic waves energy, thermal energy, salinity gradients energy, vibration energy, and so on [15], [9], [25], [17]. The characteristics of these energy harvesters is essentially different from that of conventional battery-powered counterparts. This opens up a new dimension to the design of algorithms and routing protocols for energy harvesting sensor networks. The random nature of harvested energy requires that the data rate, transmission power control, and (data) flow routing scheduling are optimized accordingly [18].

The time-varying profile of replenishment rates of renewable energy sources poses challenges on assigning optimal data rates to source nodes to maximize the quality of collected data. On the one hand, if the data rate at a node is set too high, the node will run out of its energy quickly. Consequently, this will compromise the sensing coverage of the node as well as network connectivity. On the other hand, if the data rate at a node is set too low while its energy replenishment rate is high, the volume of data generated at the planned data rate is small, thereby affecting the quality of monitoring. This also loses the opportunity of utilizing the extra energy charged during this period and the extra charged energy will be wasted. Thus, to maximize the data quality of a harvesting sensor network, it is vital to assign each source node a fair, optimal data rate, and make sure all generated data will be routed to the sink efficiently.

1.1 Related Work

It is desirable to maximize the sum of rates from all the nodes in the network, subject to energy constraint on each sensor node. Typically, the problem is formulated as a linear programming problem with an objective to maximize the sum of data rates or the network lifetime with the flow balance and energy constraints at each node. However, as mentioned by Hou et al. [13], although such a solution can maximize the sum of the rates of all nodes, it suffers a severe bias in rate allocation among the nodes. In particular, nodes consuming the least amounts of energy on the data routing are allocated with much more data rates than the other nodes in the network. As a result, the data collection behavior of the entire network only favors the nodes near to the sink, while other remote nodes will be unfavorably penalized with much smaller, even zero data rates. Consequently, the quality of monitoring of the network will be compromised because not sensing data from all sensors has been collected. Thus, it is needed to provide fair

- *W. Liang, X. Ren, and X. Xu are with the Research School of Computer Science, The Australian National University, Building 108 (CS&IT Building), North Road, Canberra, ACT 0200, Australia. E-mail: wliang@cs.anu.edu.au, {richard.rxj, grace.xu}@anu.edu.au.*
- *X. Jia is with Research Lab for Mobile Ad-hoc and Sensor Networks, Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong. E-mail: csjia@cityu.edu.hk.*

Manuscript received 2 July 2012; revised 8 Oct. 2012; accepted 25 Nov. 2012; published online 16 May 2013.
Recommended for acceptance by M. Thai.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2012-07-0613.
Digital Object Identifier no. 10.1109/TPDS.2013.136.

data rate allocation to all sensors to ensure the quality of monitoring of the network.

In conventional wireless sensor networks, to provide a fair, optimal data rate allocation for each node under the specified network lifetime constraint, the fair data rate allocation and flow routing problem was extensively studied [30], [6], [13], [28]. For example, Hou et al. [13] coined the problem as the *lexicographic max-min rate allocation problem*, and proposed a linear programming solution by exploiting the parametric analysis technique and the duality of the problem. Although their algorithm does deliver a polynomial-time heuristic solution, the computational complexity is $O(n^8m)$, which is quite high, where n and m are the number of nodes and links in the network. Chen et al. [6] studied the same problem and provided a linear programming solution, too. Wang and Kar [30] addressed the rate control problem with the objective of achieving proportional fairness among the end-to-end sessions. Su et al. [28] formulated the rate allocation problem as a network utility maximization problem and expressed the utility function as a nonconvex function. They transformed the problem into an approximate, convex optimization problem, and decomposed the problem into a rate control and scheduling subproblems, using duality theory.

In energy harvesting sensor networks, Fan et al. [10], [18], [19] studied the lexicographic max-min rate allocation problem and provided a distributed algorithm for the problem in special network topologies including tree or directed acyclic graph networks. Liu et al. [18] considered the same problem by providing a linear programming solution, too. Consider the time-varying nature of energy replenishment in harvesting sensor networks, they presented a heuristic to adaptively adjust data rates and showed that this dynamic adjustment strategy is optimal in long-term [18]. Zhang et al. [32] recently considered the data rate allocation problem in tree networks for optimizing a concave utility function. They devised an optimal solution and a distributed implementation of the proposed algorithm.

Most existing studies on fair data rate allocation aim to maximize the rate sum without incorporating spatial and temporal data correlations among sensors. Such a rate allocation may suffer the following deficiencies. For these higher rate neighboring sensors whose sensing data are highly correlated, lots of identical data from them will be routed to the sink, while for those lower rate sensors whose readings are not correlated, less data from them will be routed to the sink. Consequently, the data received at the sink from different subregions will be unbalanced, and the quality of monitoring of the network will be compromised.

As sensors usually are densely and randomly deployed, their sensing data are likely spatially temporally correlated. Particularly for those sensors that are near each other, their readings have high similarities. In this paper, we focus on the monitoring quality (or data quality) by incorporating spatial data correlations when performing fair rate allocation and flow routing. We aim to achieve proportional fairness of requested rates. Unlike previous work by formulating the data rate allocation and flow routing problem as a linear program problem (LP) and solving

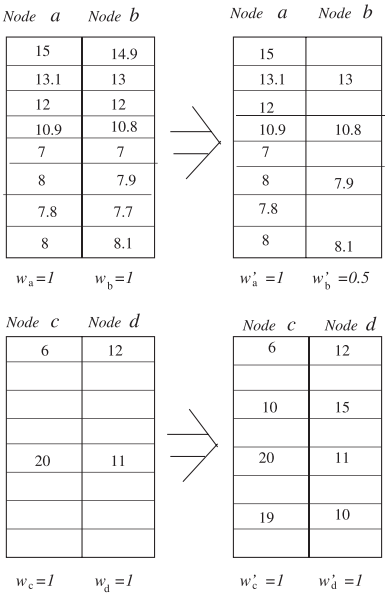


Fig. 1. A motivation example.

the LP, we focus on developing fast approximate solutions by exploiting the combinatorial property of the problem, as an exact solution delivered by the LP usually is computationally expensive and the worst of all may not be applicable to real sensor networks due to the time varying nature of harvesting energy source profiles. Since the uncertainty of energy replenishment, there is no way to predict the exact amount of energy that a sensor node will be recharged for a given time period. Thus, in most application scenarios, to maintain the "neutral operation" of the network, it is assumed that the energy budget of each sensor in the given time period is roughly equal to the amount of energy generated in that period, based on the past information of harvested energy at the node [16]. Therefore, there is no need to find an exact solution to the optimal data rate allocation to each node based on this estimated energy budget, instead, an approximate solution suffices. Furthermore, finding an approximate solution takes much less time than finding an optimal solution. In this paper, we will focus on exploiting the combinatorial property of the problem by developing a fast, approximate solution to the problem. Motivated by a scenario illustrated in Fig. 1, we assume that the data rates of nodes a and b are 8 packets per second, while those of nodes c and d are 2 packets per second, assuming that each packet contains a single reading only. We further assume that nodes a and b are near each other and their sensing readings are highly correlated, while nodes c and d are relatively far away from each other and their readings are not correlated. To incorporate spatial data correlation among the sensors into fair data rate allocation and flow routing, we introduce the rate weight concept. Denote by w_v the rate weight of a source node v. In traditional data rate allocation, all source nodes have equal weights, i.e., $w_a = w_b = w_c = w_d = 1$. When incorporating the spatial data correlation into the rate allocation with an objective to maximize the monitoring quality, we may set $w'_a = 1$, $w'_b = 0.5$, $w'_c = 1$ and $w'_d = 1$. The rate weights of nodes c and d with $w'_c = w'_d = 1$ imply that

their sensing readings are independent of each other. Now, if we reduce the rate of node b to 4 packets/s, we are still able to monitor the readings of node b with great confidence, because its readings can be approximately represented by the readings of node a. Once the network resource (e.g., the energy used for routing) originally occupied by node b is now released back to the network, nodes c and d can take this opportunity to utilize the energy by increasing their own data rates to 4 packets/s. Thus, the network can better monitor the readings of nodes c and d, thereby improving the monitoring quality in their vicinities and the entire network.

1.2 Contributions

The main contributions of this paper are as follows: A novel monitoring quality maximization problem in harvesting sensor networks is formulated that focused on the data quality, not the data quantity, by incorporating spatial sensing data correlation among sensors into consideration. A dynamic rate weight concept is introduced. To fairly allocate sensors optimal data rates and efficiently route the sensing data to the sink, a weighted, fair data rate allocation, and flow routing problem is introduced. Unlike most existing work that formulated the similar problem (lexicographic rate maximization problem) as a linear programming (LP) and solved the LP, we exploit the combinatorial property of the problem by developing fast approximation algorithms with provable approximation ratios. Distributed implementations of the proposed algorithms are also devised. To evaluate the performance of the proposed algorithms, extensive experiments by simulation are conducted. The experimental results demonstrate that the proposed algorithms are promising, and their solutions are fractional of the optimum.

To the best of our knowledge, this is the first time that the spatial data correlation among sensors has been incorporated into the data rate allocation in energy harvesting sensor networks, a novel optimization framework aiming to optimize the quality, not the quantity (the volume) of collected data is formulated. A dynamic rate weight concept is introduced and a very first fast approximation algorithm for the fair data rate allocation and flow routing problem is devised, which may have interest by itself and be applicable to other optimization problems beyond energy harvesting sensor networks.

1.3 Paper Organization

The remainder of this paper is organized as follows: We first introduce the system model, notions, and problem definitions in Section 2. We then briefly describe Garg and Könemann's flow framework for the maximum concurrent flow problem in Section 3. We third devise a novel algorithm for the monitoring quality maximization problem in which a key subroutine, an approximation algorithm for the weighted, fair rate allocation and flow routing problem, is devised based on Garg and Könemann's flow framework in Section 4. Furthermore, we devise a faster approximation algorithm for the weighted, fair rate allocation and flow routing problem in Section 5. A distributed implementation of the new algorithm is developed in Section 6. We finally conduct extensive experiments by simulation to evaluate the performance of the proposed algorithm in Section 7, and we conclude in Section 8.

2 Preliminaries

In this section, we first introduce the system model. We then define the monitoring quality of the network, data correlation, data quality, and rate weight concept, the monitoring quality maximization problem, and the weighted, fair data rate allocation and flow routing problem precisely.

2.1 System Model

Consider a wireless sensor network $G = (V, E)$ consisting of $n = |V|$ stationary sensor nodes and a sink (a base station), deployed for periodic environmental monitoring, where each sensor is powered by reusable energy like solar energy and has identical transmission range. There is an edge in E between two sensors or a sensor and the base station if they are within the transmission range of each other, where $m = |E|$. Time is slotted into equal time slots. At each time slot, a single sampling reading or a packet can be generated or transmitted. *An interval* consists of a fixed number of consecutive time slots. We assume that data rate allocation and flow routing are scheduled interval by interval, in response to dynamic changes of spatial data correlation and energy replenishment rates of sensor nodes. The scheduling proceeds in the beginning of each interval, based on the energy budget of each sensor in the interval. Denote by τ the number of time slots within an interval, where the value of τ usually is determined by the energy recharging and consuming rates of sensors.

In this paper, we follow a wide-adopted assumption that the amount of harvested energy at a future time period is uncontrollable but predictable based on the source type and harvesting history [16], [18]. We assume that in the beginning of each interval t, the energy budget $B_v(t)$ of sensor v for that interval is instantaneously available, and the energy replenishment rate of v is much slower than its energy consumption rate [18]. To estimate the energy budget $B_v(t)$ of node v in interval t, there are many efficient approaches to achieve that [16], [22], [24]. For example, one such approach is that the energy budget of sensor v in interval t is estimated based on the weighted sum of its available energy in previous l intervals, i.e., $B_v(t) = a_1 \cdot B_v(t-1) + a_2 \cdot B_v(t-2) + \cdots + a_l \cdot B_v(t-l)$ if $B_v(t) \leq CB_v$, $B_v(t) = CB_v$ otherwise, where a_i is a constant with $0 < a_i < 1$ and $\sum_{i=1}^{l} a_i = 1$, and CB_v is the battery capacity of node v. We also assume that E^{TX} and E^{RX} are the amounts of energy consumed by transmitting and receiving a single bit of data, respectively. Following the similar assumption as in [10], we assume that no constraints are imposed on the link bandwidth because we consider environmental monitoring. In such application scenarios, sensors usually sense samples every 1-5 minutes and the communication bandwidth is not a major issue [3]. Furthermore, to support long-period, continuous monitoring service, we assume that sensors should not consume more energy than they can collect to achieve perpetual operations [16]. The data rate of a sensor in a given period, thus, is determined by the amount of energy it can collect in that period.

2.2 Data Correlation and Rate Weight

To maximize the quality of monitoring, we assign each source node a *rate weight* between 0 and 1 that represents a certain degree of spatial correlation of readings between the node and its neighbors. This rate weight will be used in the next interval when performing the rate allocation to this node and its neighbors. In the following, we detail how to collect spatial data correlation information in the current interval. That is, at each time slot whenever there is data transmission from a source node v, all its neighbors can receive the packet no matter whether the receiver is an intended receiver. To represent this spatial data correlations among sensors, an undirected graph, the *data correlation graph* $G(t) = (V, E, c_t)$ in interval t is identical to the communication network graph $G(V, E)$, its edge weight function $c_t(\cdot)$ is defined as follows:

Given two neighboring nodes u and v with data rates r_v and r_u in interval t, we assume $r_v \geq r_u$. For the case where $r_v \leq r_u$, the discussion is similar, omitted. Let $r_v(1)$, $r_v(2), \ldots, r_v(\tau)$ and $r_u(1), r_u(2), \ldots, r_u(\tau)$ be the sequences of sensing reading values of v and u within interval t if there were a reading per time slot, *the distance* $d_t(u, v)$ between u and v is defined as follows: Assume that $r_v(i_1), r_v(i_2), \ldots, r_v(i_{r_v})$ are the sampling readings of node v within interval t, then for each missing sampling reading $r_v(i)$ with $i \in \{1, \ldots, \tau\} \setminus \{i_1, i_2, \ldots, i_{r_v}\}$ but $i_{j-1} \leq i < i_j$, the recent reading of v will be used as the estimate of the missing reading at time slot i, i.e., $\hat{r}_v(i) = r_v(i_{j-1})$. The distance between u and v in interval t then is

$$d_t(u, v) = \sum_{j=1}^{\tau} \sigma\left(\left|\frac{r_v(j) - r_u(j)}{\max\{r_v(j), r_u(j)\}}\right|\right), \tag{1}$$

where ς is a given similarity threshold with $0 \leq \varsigma < 1$, $\sigma(|a - b|) = 1$ if $|a - b| \leq \varsigma$; otherwise $\sigma(|a - b|) = 0$.

In the rest of the paper, we assume that the data correlation graph $G = (V, E, c_t)$ in interval t has been constructed, where $c_t(u, v) = \frac{d_t(u,v)}{\tau} \leq 1$ for each edge $(u, v) \in E$. We will calculate data correlation based on historical data correlation information, using similar techniques for energy-harvesting predictions. The calculated data correlation information will be used for fair rate allocation at the current time slot. We here use a simple data correlation model as an illustrative example. Let r_v be the rate of node $v \in V$ that is evenly distributed among the τ time slots in each time interval t. The rest $\tau - r_v$ time slots without data samplings are referred to as "default value time slots." The r_v sampling readings of node v can be written as a vector $V_v(t)$ in which each component $V_v(t, j)$ is either "—" or the actual sampling value at each slot j, $1 \leq j \leq \tau$. We now "smooth" vector $V_v(t)$ to another vector $V'_v(t)$ as follows: For each default value, we use the previous reading data of node v to replace the default value.

2.3 Data Quality

We aim to maximize the quality of collected data (monitoring quality), by incorporating the spatiotemporal data correlation into fair rate allocation and routing, where the quality of collected data at time slot t is defined as follows: Given a node $v \in V$, its contribution to the network-wide data quality can be represented by a nonnegative, concave utility function $u_t(v)$:

$$u_t(v) = \sum_{u \in N_c(v)} g(r_v(t), r_u(t)), \tag{2}$$

where the value of $u_t(v)$ will be jointly determined by a function g whose parameters are the rate r_v of v, the rates r_u of its correlated master neighbors $u \in N(v)$, and the common sensing data time slots between r_u and r_v in the interval. Here, $N_c(v) \subseteq N(v)$ is the set of correlated neighbors of v. The utility function $u_t(v)$ is a monotonic increasing function whose marginal utility decreases with the increase of its rate, $0 \leq u_t(v) \leq 1$, which means that when the assigned data rate to v approaches its maximum requested data rate R_v, its gain becomes smaller.

In this paper, we adopt the following utility function. Each node has at most one neighbor to be chosen as its correlated partner at time slot t. For this special assignment, assume that node v is the *master node* $w_v = 1$ while node u is the *slave node* of v with $w_u < 1$, then the (sampling data) vector of u at time slot t can be further smoothed by the data from the vector of v due to their highly spatial data correlation. Notice that the contribution of slave u to the data quality of the network is determined not only by its own readings but also by its master readings, the readings of v at time slot t. As we assumed that the sampling of a node is evenly performed within the τ slots and let $r_{u,v}$ ($r_{u,v} \leq \min\{r_u, r_v\}$) be the number of time slots at which both u and v have sampling readings, then the utility rate of the slave node u is $\mu_u = \min\{1, \frac{r_u + r_v - r_{u,v}}{R_u}\}$ while the utility rate of its master node v is $\mu_v = r_v/R_v$, and for a stand-alone node v (not pairing with any other node), its utility rate is $\mu_v = r_v/R_v$, where the utility function can be any concave function, for example, it can be defined as follows:

$$u_t(v) = f(\mu_v) = 1 - (1 - \mu_v)^a, \tag{3}$$

where function $f(x)$ is a monotonic increasing function with $0 \leq x \leq 1$ and $a > 1$ is a constant. Generally, for a given node v, the larger the utility value $u_t(v)$ is, the bigger the contribution made by the node to the higher the data quality of the entire network will be. Thus, maximizing the quality of collected data is equivalent to maximizing the sum of utilities of all nodes, i.e., $\sum_{v \in V} u_t(v)$.

2.4 Problem Definitions

Given a harvesting sensor network $G(V, E)$ with a sink and a data correlation confidence threshold θ with $0 < \theta \leq 1$, *the monitoring quality maximization problem* in $G(V, E)$ in interval $t + 1$ is to assign each sensor node $v \in V$ a data rate $r_v(t + 1)$ and all generated data by the node will be routed to the sink such that the monitoring quality of the network in interval $t + 1$, $\sum_{v \in V} u_t(v)$, is maximized, subject to the energy budget constraint $B_v(t + 1)$ on each sensor $v \in V$ and the maximum data rate of v being bounded by a given value R_v.

To solve the above optimization problem is to assign each source node a rate weight based on the data correlation graph in the previous interval, and to solve *the weighted, fair data rate allocation and flow routing problem*, which is defined

as follows: Given a harvesting sensor network $G(V, E)$, assume that the maximum data rate of each sensor $v \in V$ is bounded by R_v in interval $t+1$, the rate weight w_v of v has been calculated from the data collection graph $G(V, E, c_t)$ with $0 \leq w_v \leq 1$, the problem is to allocate each sensor node $v \in V$ a *weighted, fair data rate* $r_v \leq R_v$ such that the sum of allocated rates is maximized, subject to the energy budget $B_v(t+1)$ of each sensor $v \in V$ in interval $t+1$, where the *fairness* on the weighted data rate allocation is that each source node will be allocated with the same proportional of its request rate (the maximum data rate). In other words, the objective is to maximize the value of λ such that $\sum_{v \in V} \lambda(w_v \cdot R_v)$ is maximized.

2.5 Approximation Algorithm

Given an optimization problem, denote by $Approx$ and OPT the costs of the approximate solution delivered by an approximation algorithm $\mathcal{A}$, and the optimal solution of the problem, the approximation ratio of algorithm $\mathcal{A}$ is ρ if $\frac{Approx}{OPT} \leq \rho$ when it is a minimization problem, or $\frac{Approx}{OPT} \geq \rho$ when it is a maximization problem.

3 The Maximum Weighted Concurrent Flow Problem

The *multicommodity flow problem* is defined on a directed network $G(V, E, c)$ with capacities $c : E \rightarrow R^+$ and k source-sink terminal pairs (s_i, t_i), $1 \leq i \leq k$, the problem is to find flows f_i from s_i to t_i that satisfy node conservation constraints and meet some objective function criteria so that the sum of flows on each edge $e \in E$ does not exceed its capacity $c(e)$. Let $|f_i|$ be the amount of flow sent from s_i to t_i. The *maximum concurrent flow problem* is to find a largest λ such that there is a multicommodity flow which routes $\lambda \cdot d_i$ units of commodity i from s_i to t_i, assuming that there are d_i demands of commodity i for all i with $1 \leq i \leq k$.

3.1 Garg and Könemann's Framework

Garg and Könemann's [12] flow framework for the maximum concurrent flow problem is as follows.

Let $\mathcal{P}_i$ denote the set of paths in $G(V, E, c)$ from s_i to t_i and let $\mathcal{P} = \cup_{i=1}^{k} \mathcal{P}_i$. Variable f_p represents the flow on path $p \in \mathcal{P}$. The linear programming formulation for the maximum concurrent flow problem is then

$$\begin{aligned} \textbf{LP}: \quad \max \quad & \lambda \\ \text{s.t.} \quad & \sum_{e \in p} f_p \leq c(e) && \forall e \in E, \\ & \sum_{p \in \mathcal{P}_i} f_p \geq \lambda \cdot d_i && \forall 1 \leq i \leq k, \\ & f_p \geq 0 && \forall p \in \mathcal{P}, \\ & \lambda \geq 0. \end{aligned}$$

The dual linear programming of the LP is to assign a length $l(e)$ to each edge $e \in E$ and a nonnegative variable z_i to each commodity i so that the length of the shortest path from s_i to t_i is at least z_i for each commodity i with $1 \leq i \leq k$, and the sum of the product of commodity variables and demands is at least 1. The dual of the above LP is as follows:

$$\begin{aligned} \textbf{DP}: \quad \min \quad & D(l) = \sum_{e \in E} l(e)c(e) \\ \text{s.t.} \quad & \sum_{e \in p} l(e) \geq z_i && \forall p \in \mathcal{P}_i, \\ & \sum_{i=1}^{k} d_i \cdot z_i \geq 1, \\ & l(e) \geq 0 && \forall e \in E, \\ & z_i \geq 0 && \forall 1 \leq i \leq k. \end{aligned}$$

Garg and Könemann's approximation algorithm for the DP proceeds in *phases*. Initially, $l(e) = \frac{\delta}{c(e)}$, $z_i = \min_{p \in \mathcal{P}_i}\{l(p)\}$ where $l(p) = \sum_{e \in p} l(e)$ and δ is a constant to be defined later. In each phase, there are k *iterations*. In iteration i, it routes d_i units of flow for commodity i from s_i to t_i, which can be further divided into *steps*. In each step, a shortest path p from s_i to t_i is found, using the current edge length function $l(\cdot)$. Assume that $c(e')$ is the bottleneck capacity of path p, then the minimum c_{min} between $c(e')$ and the remaining demands d_i' of commodity i is sent along path p. The dual variable l is updated accordingly, and z_i is then set equal to the length of the new shortest path from s_i to t_i. The algorithm terminates when the dual objective function value $D(l)$ is at least one, i.e., $D(l) = \sum_{e \in E} l(e) \cdot c(e) \geq 1$. The final flow f delivered may violate some of the edge capacities. To obtain a feasible solution, we need to scale f by the maximum congestion incurred on edges. Since we only augment the flow along the paths with lengths smaller than one, and the length update rule ensures that the length of edges is exponential in their congestion, we conclude that the maximum congestion on any edge will not be very large. Let β be the optimal objective value of $D(l)$. When $\delta = (\frac{1-\epsilon}{|E|})^{1/\epsilon}$, Garg and Könemann showed the following lemmas.

Lemma 1 (see [12]). *If $\beta \geq 1$, Garg and Könemann's algorithm terminates after at most $T = 1 + \frac{\beta}{\epsilon}\log_{1+\epsilon}\frac{|E|}{1+\epsilon}$ phases.*

Lemma 2 (see [12]). *After $(T-1)$ phases, $(T-1) \cdot d_i$ units of commodity i have been routed. Scaling the final flow by $\log_{1+\epsilon}\frac{1}{\delta}$ yields a feasible primal solution of value $\lambda = \frac{T-1}{\log_{1+\epsilon} 1/\delta}$, where ϵ is a constant with $0 < \epsilon \leq 1/3$.*

Lemma 3 (see [12]). *If $\beta \geq 1$, then $\frac{|f|}{\log_{1+\epsilon}\frac{1}{\delta}} \geq (1-3\epsilon) \cdot OPT$, where OPT is the value of the optimal flow, where ϵ is a constant with $0 < \epsilon \leq 1/3$.*

In case $\beta < 1$, they proposed an approach to scale the value of β to at least one by adopting a popular technique in [26]. We state this in the following theorem.

Theorem 1 (see [12]). *There is an approximation algorithm for the maximum concurrent flow problem in $G(V, E)$, which delivers a solution with approximation ratio of $(1-3\epsilon)$. The algorithm takes $O(\epsilon^{-2}(k \log m + m) \log m \cdot (m + n \log n))$ time, where $n = |V|$, $m = |E|$, and ϵ is a constant with $0 < \epsilon \leq 1/3$.*

3.2 Maximum Weighted Concurrent Flow Problem

We now generalize the maximum concurrent flow problem to the *maximum weighted concurrent flow problem*. The latter is to find flows for the k pairs such that the weighted ratio of net demands is maximized, assuming that each commodity i is associated with a positive weight w_i. In other words, let λ_i be the ratio of the delivered demands D_i of commodity i,

to demands d_i, then $\lambda_i = \frac{D_i}{d_i}$, the *weighted ratio* of commodity i is $w_i \cdot \lambda_i$. The optimization objective of the maximum weighted concurrent flow problem, thus, is to maximize $\lambda = \min\{w_i \cdot \lambda_i \mid 1 \leq i \leq k\}$.

Notice that the maximum weighted concurrent flow problem where each commodity has demands d_i with weight w_i can be reduced to another maximum concurrent flow problem where each commodity has demands $d'_i = w_i \cdot d_i$. To obtain a linear program formulation for the maximum weighted concurrent flow problem, we can simply change the constraint of the LP: $\sum_{p \in \mathcal{P}_i} f_p \geq \lambda \cdot d_i$ to $\sum_{p \in \mathcal{P}_i} f_p \geq \lambda \cdot (w_i d_i)$. The dual of this modified LP is the DP(2), where the constraint $\sum_{i=1}^{k} d_i \cdot z_i$ now becomes $\sum_{i=1}^{k} (w_i d_i) \cdot z_i$.

4 Maximizing Monitoring Quality

In this section, we provide an optimization framework for the monitoring quality maximization problem, which consists of assigning source nodes rate weights, and devising an approximation algorithm for the weighted, fair data rate allocation and flow routing problem.

4.1 Rate Weight Assignment

We assign each source node a weight between 0 and 1. We assume that each source node can either "stand alone," which means that its sampling readings are independent of its neighbors, or become a partner of one of its neighbors. For the latter, we further assume that the source node has "at most" one partner only for the convenience of discussion. It is easy to extend this single partner to multiple partners, omitted. For each pair of partner nodes, due to the high correlation between their sampling readings, one of the nodes will become *the master* and its rate weight is set to 1; another will become *the slave* of the master and its rate weight is set to a value strictly less than 1. For each stand-alone source node, its rate weight is set to 1.

Given the data correlation graph $G(V, E, c_t)$ in interval t, we proceed the computation of partner pairs and rate weight assignment in interval $t+1$ as follows:

We first construct a subgraph $G' = (V, E')$ of $G(V, E, c_t)$, where $E' \subseteq E$. For each node $v \in V$, let (v, v_u) be the edge incident to v with the maximum cost $c_t(v, v_u)$ among its incident edges of v. If $c_t(v, v_u) \geq \theta$, then $(v, u_v) \in E'$, where θ is a given *confidence threshold*.

We then find a maximum matching $M_{G'}$ in G', where the two endpoints of each matched edge become a pair of partners. We finally assign each source node a rate weight. For each source node v that is not an endpoint of any edge in $M_{G'}$, assign it a rate weight $w_v = 1$. For each edge $(u, v) \in M_{G'}$, if the energy budget $B_v(t+1)$ of node v is no less than $B_u(t+1)$ of node u, v will become the *master* of u while u will become the *slave* of v. The rate weight assigned to v is $w_v = 1$. For slave nodes, there are two different rate weight assignment strategies: one is the *uniform weight assignment* that assigns all slaves an identical weight $0 \leq w < 1$. Another is the *variable weight assignment* that assigns each slave node u a weight of $w_u = 1 - c_t(u, v)$, where v is the master of u. It can be seen that the larger the value of $c_t(u, v)$, the higher the reading similarity between u and v, and the smaller the weight w_u will be.

4.2 Algorithm for Monitoring Quality Maximization Problem

Having assigned the rate weight to each source node, we now allocate each source node a weighted, fair data rate and route the sensing data by the node to the sink such that the sum of all data rates in interval $t+1$ is maximized. We reduce the weighted, fair data rate allocation and flow routing problem in $G(V, E, B(t+1))$ to the maximum weighted concurrent flow problem in graph $G_1(V_1, E_1, C_t)$ whose definition will be given later, where for each sensor node in V there is a corresponding *source node* in G_1. Each source node $v' \in V_1$ has a commodity of demands $d_v = R_v/\tau$, where R_v is the maximum data rate of node v in interval $t+1$. Algorithm for the monitoring quality maximization problem, `Monitor_Quality_Max`, is described in `Algorithm` 1.

Algorithm 1. Monitor_ Quality_Max in interval $t+1$.

Input: The data correlation graph $G = (V, E, c_t)$ in interval t, the confidence threshold θ, the energy budget $B_v(t+1)$ and the maximum data rate R_v for each $v \in V$ in interval $t+1$, and an accuracy parameter $\epsilon > 0$.

Output: The rate allocation r_v of each node $v \in V$ in interval $t+1$ such that $\sum_{v \in V} r_v$ is maximized.

1: Construct a subgraph G' from the data correlation graph $G(V, E, c_t)$ with confidence threshold θ;
2: Find a maximum matching $M_{G'}$ in G', using an algorithm due to Micali and Vazirani [21];
3: Assign each slave node a fractional weight for each matched edge in $M_{G'}$ and every other node with weight 1; relay $M_{G'}$ to the sink;
4: Construct the auxiliary graph $G_1(V_1, E_1, C_t)$;
5: Find a maximum weighted concurrent flow in G_1 from all source nodes to the sink by calling either `Algorithm` 2 (to be introduced), or a faster approximation algorithm, `Algorithm` 3, in Section 5;
6: Assign each source node $v \in V$ a data rate r_v in interval $t+1$.

4.3 Algorithm for the Weighted, Fair Data Rate Allocation and Flow Routing Problem

In this section, we devise an approximation algorithm, `Algorithm` 2, for the weighted, fair data rate allocation and flow routing problem in $G(V, E, B(t+1))$, by reducing it to the maximum weighted concurrent flow problem in $G_1(V_1, E_1, C_t)$. To apply Garg and Könemann's flow framework for the maximum weighted concurrent flow problem in a flow network, it requires that the network is a directed network with edge capacities. However, the harvesting sensor network $G(V, E, B(t+1))$ is an undirected network with node capacities, where the energy budget $B_v(t+1)$ for each node v in interval $t+1$ is its capacity in the interval, there is not any specified capacity on any edge.

Algorithm 2. Weighted_Fair_Data_Rate.

Input: the flow network $G_1(V_1, E_1, C_t)$ derived from $G(V, E, B(t+1))$, the rate weight w_v, the energy budget $B_v(t+1)$ in interval $t+1$ and the maximum data rate R_v of each $v \in V$, and the accuracy parameter ϵ.

Output: the rate allocation r_v for each $v \in V$ in interval $t+1$ such that $\sum_{v \in V} w_v r_v$ is maximized.

for each node $v' \in V_1$ **do**
  $r_v \leftarrow 0$; $d_v \leftarrow \frac{w_v \cdot R_v}{\tau}$
  /* r_v is the data rate and d_v is the demands of source node v */
end for;
$D \leftarrow 0$ /* $D(l) = \sum_{e \in E_1} l(e) \cdot C_t(e)$* /;
for each edge $e \in E_1$ **do**
  $\delta \leftarrow (\frac{1-\epsilon}{|E_1|})^{1/\epsilon}$; $l(e) \leftarrow \frac{\delta}{C_t(e)}$; $D \leftarrow D + l(e) \cdot C_t(e)$;
end for;
/* the value of D determines whether the following iteration executes */
while $D < 1$ **do**
  for each node $v' \in V_1$ **do**
    $d'_v \leftarrow d_v$;
    while $(d'_v \neq 0)$ & $(D < 1)$ **do**
      Find a shortest path p_v in G_1 from v' to the sink w.r.t. the current length function $l()$;
      Find the bottleneck capacity $C_t(e')$ in p_v with $e' \in p_v$, $r_v(p_v) \leftarrow \frac{C_t(e')}{E^{TX}+E^{RX}}$;
      /* calculate the amount of the increased data rate of v */
      $\Delta r_v \leftarrow \min\{r_v(p_v), d'_v\}$;
      for each edge $e \in p_v$ **do**
        $l(e) \leftarrow l(e)(1 + \epsilon \cdot \frac{\Delta r_v (E^{TX}+E^{RX})}{C_t(e)})$;
        $D \leftarrow D + \epsilon \cdot \frac{\Delta r_v (E^{TX}+E^{RX})}{C_t(e)}$;
        $r_{v'} \leftarrow r_{v'} + \Delta r_v$;
        $d'_v \leftarrow d'_v - \Delta r_v$
      end for;
    end while;
  end for;
end while;
for each node $v \in V$ **do**
  the rate $r_v \leftarrow \tau \cdot r_v / \log_{1+\epsilon} \frac{(1+\epsilon)}{\delta}$; /* flow scaling */
end for

We now transform the maximum weighted concurrent flow problem in $G(V, E, B(t+1))$ with node capacities to the maximum weighted concurrent flow problem in another directed network $G_1 = (V_1, E_1, C_t)$ with edge capacities, using a traditional transformation approach [1]. That is, for each node $v \in V$ except the sink, there are two corresponding nodes v' and v'' in V_1 and v' is a *source node*. Each source node $v' \in V_1$ has demands $w_v \cdot d_v$ ($=w_v \cdot \frac{R_v}{\tau}$) and the energy capacity of v' is $\frac{B_v(t+1)}{\tau}$ at each time slot among the τ time slots in interval $t+1$. There is a directed edge $\langle v', v'' \rangle$ in E_1 from v' to v'' whose capacity is a fraction of the energy budget $B_v(t+1)$ of node v, i.e., $C_t(v', v'') = B_v(t+1)/\tau$. For each edge $(v, u) \in E$, there are two edges $\langle v'', u' \rangle$ and $\langle u'', v' \rangle$ with capacities $C_t(v'', u') = B_v(t+1)/\tau$ and $C_t(u'', v') = B_u(t+1)/\tau$, respectively, because the energy budget for each outgoing edge of a node is upper bounded by its energy budget. The detailed algorithm for the weighted, fair data rate allocation and flow routing problem, `Weighted_Fair_Data_Rate`, is presented in `Algorithm` 2.

In the following, we first investigate the maximum concurrent flow property of network G_1, followed by analyzing the time complexity and the approximation ratio of `Algorithm` 2.

Lemma 4. *Given the constructed network $G_1(V_1, E_1, C_t)$, if a flow routing protocol based on Garg and Könemann's flow framework is adopted to route the sensing data of all source nodes to the sink, the following claims hold: 1) Each edge $\langle v', v'' \rangle$ is saturated no later than any edge $\langle v'', u' \rangle$ for all $v' \in V_1$. 2) Once all edges $\langle v', v'' \rangle$ are saturated for all $v' \in V_1$, `Algorithm` 2 will terminate no matter whether the other edges in G_1 are saturated or not. 3) The types of edges $\langle v', v'' \rangle$ are the first n edges to be saturated.*

Proof. Case 1. Let p be a directed path from a source node x' to the sink and $e_1 = \langle v'', u' \rangle$ an edge on p. Then, edge $e_2 = \langle v', v'' \rangle$ must be on p too, because it is the only entry edge to node v'' by the construction of G_1. In addition, the capacities of both e_1 and e_2 are $B_v(t+1)$. Now, if e_1 is saturated through the flow augmentation, e_2 must be saturated, too. This implies that e_2 is saturated is no later than that of e_1. Case 2. Note that all directed paths from each source node v' to the sink must include edge $\langle v', v'' \rangle$. When each edge $\langle v', v'' \rangle$ for all $v' \in V_1$ is saturated, there will no longer have any directed path from a source node to the sink any more, this implies that no further flow augmentations from source nodes to the sink will be possible, the algorithm terminates. Case 3. Following the similar discussion as the one in Case 2, once edge $\langle v', v'' \rangle$ is saturated, all the directed edges $\langle v'', u' \rangle$ starting from node v'' will not be in any directed path from any other source node to the sink, because edge $\langle v', v'' \rangle$ is the only predecessor of the edge $\langle v'', u' \rangle$ on any such directed path. Meanwhile, edge $\langle v'', u' \rangle$ is one of the successors of edge $\langle v', v'' \rangle$ in any such path. □

Theorem 2. *There is an approximation algorithm `Algorithm` 2 for the weighted, fair data rate allocation and flow routing problem in a harvesting sensor network $G(V, E, B(t+1))$ in interval $t+1$. The algorithm takes $O(\epsilon^{-2}(m^2 \log^2 n + mn \log^3 n))$ time, and delivers an approximate solution of no less than $(1 - 3\epsilon)$ times of the optimum for any constant ϵ with $0 < \epsilon \leq 1/3$, where $n = |V|$ and $m = |E|$.*

Proof. Following Lemma 1, Lemma 2, Lemma 3 and Theorem 1, the claims are straightforward, omitted. □

5 Shortest Path Tree-Based Algorithm

In this section, we devise a faster approximation algorithm for the weighted, fair data rate allocation and flow routing problem. Since all source nodes share the common destination node—the sink, there is a shortest path tree rooted at the sink *w.r.t.* the edge length function. Instead of routing a fractional flow along a single shortest path from a source node to the sink in each step within each iteration by Garg and Könemanns' framework, we can route *the same proportion* of the demands of each commodity along the shortest path from each source node to the sink simultaneously such that 1) the sum of flows is maximized; and 2) the ratio of net demands for all source nodes is identical, subject to the bottleneck capacity of the edges in the shortest path tree. Consequently, the number of iterations needed to

saturate all edges in the network can be significantly reduced, the algorithm will run much faster.

5.1 Algorithm

The proposed algorithm constructs shortest path trees iteratively. Within each iteration i with $i \geq 1$, a shortest path tree T_i spanning all source nodes of G_1 is constructed, the maximum proportion of the remaining demands from all source nodes will be routed to the sink such that at least one tree edge is saturated. Then, update the length of each edge in the tree, where the length of an edge will be determined by the amount of flows running through the edge. The algorithm terminates when $D(l) = \sum_{e \in E_1} l(e)C_t(e) \geq 1$.

Consider the shortest path tree T_i. Clearly, T_i contains all edges $\langle v', v'' \rangle$ in G_1, otherwise, at least one source node v does not have any demands left for transmission and the algorithm terminates. Therefore, in the rest we only consider these tree edges $\langle v', v'' \rangle$ by neglecting the other tree edges $\langle v'', u' \rangle$, because the former will be saturated before the latter by Lemma 4.

For each tree edge $e = \langle v', v'' \rangle$ (or $e = \langle v'', u' \rangle$), denoted by $DS(e)$ the set of descendant source nodes in the subtree rooted at v' of T_i including v' itself (for the sake of simplicity, we abuse the notation and use v to represent v' if no ambiguity arises). For each source node v, there is a unique path p_v^i in T_i from v' to the root, and the flow routed along p_v^i is in a proportion of the remaining demands of source node v.

Let f_v^i be the total data flow (the primal objective function value) of source node $v \in V$ routed and f_i the total data flow routed from iteration 1 to iteration i, then $f_i = \sum_{v \in v} f_v^i$. For any two source nodes u and v, the rate λ_i of their flows in iteration i is equal, i.e.,

$$\lambda_i = \frac{f_v^i - f_v^{i-1}}{d'_v} = \frac{f_u^i - f_u^{i-1}}{d'_u}, \tag{4}$$

where d'_v is the amount of remaining demands at source node v in iteration $(i-1)$. Initially $d'_v = d_v$. The amount of data flow routed through tree T_i is the sum of flows, i.e., $f(T_i) = \sum_{v \in V}(f_v^i - f_v^{i-1}) = \lambda_i \cdot (\sum_{v \in V} d'_v)$. To maximize $f(T_i)$ is equivalent to maximize λ_i. In the following, we show how to find a flow to maximize the value of λ_i.

Lemma 5. *Given the shortest path tree T_i in iteration i, for each edge $e = \langle v', v'' \rangle$ in T_i, compute the maximum rate $\lambda(e)$ of descendant source nodes in the subtree of T_i rooted at v' that statures edge e,*

$$\lambda(e) = \frac{C_t(e)}{\sum_{v \in DS(e)} d'_v (E^{TX} + E^{RX})}.$$

If we route the same rate of demands from each source node such that at least one of the tree edges is saturated by the flows, then the maximum rate λ_i is $\lambda_i = \min\{\lambda(e) \mid \forall e \in T_i\}$.

Proof. We show the claim by contradiction. Assume that the maximum rate in T_i is λ'_i with $\lambda'_i > \lambda_i$. Let e and e' be the two edges in T_i such that $\lambda_i = \lambda(e)$ and $\lambda'_i = \lambda(e')$. We distinguish the relationship between e and e' into the following three cases.

Case 1. e' and e are in the same directed path of T_i and e is an ancestor edge of e', then $DS(e') \subset DS(e)$. If all the source nodes in $DS(e)$ also adopt the maximum rate λ'_i to route their fractional demands, then the amount energy consumed at e is

$$\begin{aligned}
&\sum_{v \in DS(e)} (\lambda'_i d'_v)(E^{TX} + E^{RX}) \\
&= \sum_{v \in DS(e) - DS(e')} (\lambda'_i d'_v)(E^{TX} + E^{RX}) \\
&\quad + \sum_{u \in DS(e')} (\lambda'_i d'_u)(E^{TX} + E^{RX}) \\
&> \sum_{v \in DS(e) - DS(e')} (\lambda_i d'_v)(E^{TX} + E^{RX}) \\
&\quad + \sum_{u \in DS(e')} (\lambda_i d'_u)(E^{TX} + E^{RX}) \\
&= \sum_{v \in DS(e)} (\lambda_i d'_v)(E^{TX} + E^{RX}) = C_t(e),
\end{aligned}$$

which is strictly larger than the energy capacity of edge e, leading to a contradiction.

Case 2. e' and e are in the same directed path of T_i and e' is an ancestor edge of e', the proof is similar to Case one, omitted.

Case 3. There is no relationship between edge e and edge e', i.e., they are on two disjoint paths to the sink. Now, if all source nodes in $DS(e)$ adopt the maximum rate λ'_i, then the energy consumption at edge e is $\sum_{v \in DS(e)}(\lambda'_i d'_v)(E^{TX} + E^{RX}) > \sum_{v \in DS(e)}(\lambda_i d'_v)(E^{TX} + E^{RX}) = C_t(e)$, which is larger than the energy capacity of edge e. This leads to a contradiction. □

The detailed algorithm `Shortest_Path_Tree` is presented in `Algorithm` 3.

Algorithm 3. Shortest_Path_Tree (G_1).

Input: flow network $G_1(V_1, E_1, C_t)$ derived from $G(V, E, B(t+1))$, the rate weight w_v, the energy budget $B_v(t+1)$ in interval $t+1$ and the maximum data rate R_v for each $v \in V$, and the accuracy parameter ϵ.

Output: the rate allocation r_v for each $v \in V$ in interval $t+1$ such that $\sum_{v \in V} w_v r_v$ is maximized.

1: **for** each node $v' \in V_1$ **do**
2: $r_v \leftarrow 0$; $d'_v \leftarrow \frac{w_v \cdot R_v}{\tau}$
3: **end for**;
4: $f \leftarrow 0$; $D \leftarrow 0$; /* $D(l) = \sum_{e \in E_1} l(e) \cdot C_t(e)$ */
5: **for** each edge $e \in E_1$ **do**
6: $\delta \leftarrow \frac{1+\epsilon}{((1+\epsilon)|E_1|)^{1/\epsilon}}$; $l(e) \leftarrow \frac{\delta}{C_t(e)}$; $D \leftarrow D + l(e) \cdot C_t(e)$
7: **end for**;
8: $i \leftarrow 0$; /* the number of iterations */
9: **while** $(D < 1)$ and $(d'_v \neq 0)$ **do**
10: $i \leftarrow i + 1$;
11: Find a shortest path tree T_i in G_1 rooted at the sink and spanning all source nodes;
12: **for** each edge $e = \langle v', v'' \rangle \in T_i$ **do**
13: Compute $DS(e)$, the set of descendant nodes in the subtree rooted at v';
14: $\lambda(e) \leftarrow \frac{C_t(e)}{(E^{TX}+E^{RX}) \cdot \sum_{v \in DS(e)} d'_v}$;

/* the maximum rate for sources through e */
end for;
$\lambda_i \leftarrow 1$; /* the possible maximum ratio */
for each edge $e = \langle v', v'' \rangle \in T_i$ **do**
if $\lambda_i > \lambda(e)$ **then**
$\lambda_i \leftarrow \lambda(e)$
end if;
end for;
for each edge $e \in T_i$ **do**
$l(e) \leftarrow l(e)\left(1 + \epsilon \frac{(E^{TX}+E^{RX})\cdot\sum_{u\in DS(e)}(\lambda_i \cdot d'_u)}{C_t(e)}\right)$;
$D \leftarrow D + \epsilon \frac{(E^{TX}+E^{RX})\cdot\sum_{u\in DS(e)}(\lambda_i \cdot d'_u)}{C_t(e)}$
end for;
for each node $v \in V$ **do**
$r_v \leftarrow r_v + \lambda_i \cdot d'_v$; $d'_v \leftarrow d'_v - \lambda_i \cdot d'_v$;
end for;
$f \leftarrow f + \sum_{v\in V} \lambda_i \cdot d'_v$;
end while;
for each node $v \in V$ **do**
return $r_v \leftarrow \frac{r_v \cdot \tau}{\log_{1+\epsilon}\frac{1+\epsilon}{\delta}}$
end for.

5.2 Algorithm Analysis

We now investigate the computational complexity of `Algorithm 3` and analyze its approximation ratio.

Let e' be an edge in T_i such that $\lambda_i = \lambda(e')$. Edge e' is referred to as the *bottleneck rate edge* of T_i. Clearly, the bottleneck rate edge will be saturated in iteration i and the total amount of flows routed in iteration i is $f(T_i) = \sum_{v\in V} \lambda_i d'_v$, where d'_v is the remaining demands of source node $v \in V$. We have the following lemma.

Lemma 6. *`Algorithm 3` has the following properties. 1) The number of iterations is upper bounded by $n \log_{1+\epsilon}\frac{1+\epsilon}{\delta}$; and 2) the maximum amount of flow going through any edge $e \in E_1$ is bounded by $C_t(e)\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$, where δ is a constant to be determined later.*

Proof. Case 1. Given any edge e in $G_1(V_1, E_1)$, let $l_i(e)$ be the length of edge e in iteration i. The initial length of e is $l_0(e) = \frac{\delta}{C_t(e)}$, where δ is a constant. Within each iteration the length of e is updated, where

$$l_i(e) = l_{i-1}(e)\left(1 + \epsilon \frac{\sum_{v\in DS(e)}(\lambda_i d'_v)(E^{TX}+E^{RX})}{C_t(e)}\right)$$

if e is an edge in T_i *w.r.t.* l_{i-1}; $l_i(e) = l_{i-1}(e)$ otherwise. Since $\sum_{v\in DS(e)}(\lambda_i d'_v)(E^{TX}+E^{RX}) \le \min\{C_t(e') \mid \forall e' \in T_i\}$ by `Algorithm 3`,

$$\frac{\sum_{v\in DS(e)}(\lambda_i d'_v)(E^{TX}+E^{RX})}{C_t(e)} \le 1.$$

We, thus, have

$$l_i(e) = l_{i-1}(e)\left(1 + \epsilon \frac{\sum_{v\in DS(e)}(\lambda_i d_v)(E^{TX}+E^{RX})}{C_t(e)}\right) \le l_{i-1}(e)(1+\epsilon) \le l_0(e)(1+\epsilon)^i,$$

since $(1+\epsilon)^a \le 1 + \epsilon \cdot a$ if $0 < a \le 1$. We claim that the final length of e is no greater than $\frac{1+\epsilon}{C_t(e)}$ when the algorithm stops. The termination condition of the algorithm is $D(l_{i_0}) = \sum_{e\in E_1} l_{i_0}(e)\cdot C_t(e) \ge 1$ in iteration i_0, while $D(l_{i_0}) \ge l_{i_0}(e)\cdot C_t(e) = \frac{1+\epsilon}{C_t(e)}\cdot C_t(e) = 1+\epsilon \ge 1$. The rest is to find the largest i_0 such that $l_{i_0}(e) \le \frac{1+\epsilon}{C_t(e)}$. Thus, the number of flow augmentations on e is bounded by $i_0 \le \log_{1+\epsilon}\frac{1+\epsilon}{\delta}$.

Intuitively, `Algorithm 3` stops when all the edges in G_1 are saturated, the number of iterations, thus, is bounded by $|E_1|\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$. However, as G_1 is a special network, one important fact about it is that once all edges $\langle v', v'' \rangle$ are saturated, the algorithm terminates no matter whether or not the other edges are saturated. Furthermore, by Lemma 4, edges $\langle v', v'' \rangle$ are the first n edges to be saturated. Thus, the maximum number of iterations of `Algorithm 3` is $n\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$.

Case 2. It is obvious that the total amount of flows routed on an edge e cannot exceed $C_t(e)\cdot i_0 = C_t(e)\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$. In other words, a feasible solution to the maximum concurrent flow problem can be obtained by scaling the final flow by a factor of $\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$. □

Theorem 3. *There is an approximation algorithm `Algorithm 3` for the weighted, fair data allocation and flow routing problem in a harvesting sensor network $G(V, E, B(t+1))$ in interval $t+1$, which takes $O(\epsilon^{-2}n^3\log n)$ time and delivers an approximate solution with the approximation ratio $(1-2\epsilon)$ for any constant ϵ with $0 < \epsilon \le 1/2$, where $n = |V|$ and $m = |E|$.*

Proof. The construction of network G_1 takes $O(|E_1| + |V_1|) = O(m_1 + n_1) = O(m+n)$ time, where $m_1 = |E_1| = 2|E| + |V| = 2m + n$ and $n_1 = |V_1| = 2|V| = 2n$. The initialization of the algorithm takes $O(m)$ time. The number of iterations in `Algorithm 3` is bounded by $n\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$ by Lemma 6. Within iteration i, it first finds a shortest path tree T_i which takes $O(m_1 + n_1\log n_1) = O(m + n\log n)$ time, using Dijkstra's algorithm. It then performs the fractional, optimal data rate calculation to all source nodes, which takes $O(n^2)$ time. Specifically, for each node in T_i, it takes $O(n)$ time to find all descendant source nodes of the node as the number of nodes in T_i is $O(n)$. Meanwhile, the calculation of $\lambda(e)$ for each edge $e \in E(T_i)$ in T_i takes $O(n)$ time. Thus, it takes $O(n|E(T_i)|) = O(n^2)$ time to find all $\lambda(e)$ and λ_i. The running time of `Algorithm 3`, thus, is

$$\begin{aligned}
&O\left(n\log_{1+\epsilon}\frac{1+\epsilon}{\delta}(n^2+m+n\log n) + \sqrt{n}m\right)\\
&= O\left(n^3\log_{1+\epsilon}\frac{1+\epsilon}{\delta}\right)\\
&= O\left(n^3\log_{1+\epsilon}(1+\epsilon)\Big/\frac{1+\epsilon}{((1+\epsilon)m_1)^{1/\epsilon}}\right)\\
&= O\left(\frac{n^3}{\epsilon}\cdot(1+\log_{1+\epsilon}m_1)\right) = O\left(\frac{n^3}{\epsilon}\cdot\left(1+\frac{\log m_1}{\log 1+\epsilon}\right)\right)\\
&= O\left(\frac{n^3}{\epsilon}\cdot\left(1+\frac{\log m_1}{\epsilon}\right)\right)\text{since } \log(1+\epsilon) \ge \epsilon \text{ when } \epsilon < 1\\
&= O\left(\frac{n^3}{\epsilon} + \frac{n^3\log(2m+n)}{\epsilon^2}\right) = O\left(\frac{n^3\log n}{\epsilon^2}\right).
\end{aligned}$$

The rest is to analyze its approximation ratio, by adopting the similar proof due to Garg and Könemann [12]. Recall that $D(l_i) = \sum_{e \in E_1} l_i(e) C_t(e)$ is the dual objective function value in iteration i. Let p_v^i be the shortest path in tree T_i from a source node v' to the sink with respect to the length function l_{i-1} in iteration i. Let $\alpha_v(l_{i-1}) = \sum_{e \in p_v^i} l_{i-1}(e)$ be the length of p_v^i and $\alpha(l_{i-1})$ the length sum of shortest paths from all source nodes to the sink, then $\alpha(l_{i-1}) = \sum_{v \in V} \alpha_v(l_{i-1})$. Denote $\alpha(l_{i-1})$ by $\alpha(i-1)$ and $D(l_i)$ by $D(i)$. Let $\beta = \min\{\frac{D(i)}{\alpha(i)} \mid 0 \le i \le i_0\}$ be the optimal dual objective value, where i_0 will be determined later, then $\beta \le \frac{D(i-1)}{\alpha(i-1)}$ for all i. For every iteration $i \ge 1$,

$$
\begin{aligned}
D(i) &= \sum_e l_i(e) C_t(e) \\
&= \sum_e l_{i-1}(e) C_t(e) + \epsilon \sum_{v \in V} \sum_{e \in p_v^i} l_{i-1}(e) \cdot (\lambda_i d'_v)(E^{TX} + E^{RX}) \\
&= \sum_e l_{i-1}(e) C_t(e) + \epsilon \sum_{v \in V} l_{i-1}\big(p_v^i\big) \cdot (\lambda_i d'_v)(E^{TX} + E^{RX}) \\
&= \sum_e l_{i-1}(e) C_t(e) + \epsilon \sum_{v \in V} \alpha_v(i-1) \cdot (\lambda_i d'_v)(E^{TX} + E^{RX}) \\
&= D(i-1) + \epsilon \sum_{v \in V} \alpha_v(i-1) \cdot \big(f_v^i - f_v^{i-1}\big)(E^{TX} + E^{RX}) \\
&\le D(i-1) + \epsilon \sum_{v \in V} \alpha_v(i-1) \\
&\quad \cdot \sum_{v \in V} \big(f_v^i - f_v^{i-1}\big)(E^{TX} + E^{RX}), \\
&\text{since } \sum (a_i b_i) \le \Big(\sum a_i\Big) \cdot \Big(\sum b_i\Big), \\
&\le D(i-1) + \epsilon \alpha(i-1) \cdot \sum_{v \in V} \big(f_v^i - f_v^{i-1}\big)(E^{TX} + E^{RX}) \\
&\le D(i-1) + \epsilon \frac{D(i-1)}{\beta} \cdot \sum_{v \in V} \big(f_v^i - f_v^{i-1}\big)(E^{TX} + E^{RX}), \\
&\text{since } \beta \le \frac{D(i-1)}{\alpha(i-1)}, \\
&= D(i-1)\Big(1 + \frac{\epsilon}{\beta} \cdot \sum_{v \in V} \big(f_v^i - f_v^{i-1}\big)(E^{TX} + E^{RX})\Big) \\
&\le D(i-1) e^{\frac{\epsilon}{\beta} \sum_{v \in V} (f_v^i - f_v^{i-1})(E^{TX} + E^{RX})} \\
&\text{since } 1 + x \le e^x \quad \text{for} \quad x \ge 0 \\
&= D(0) e^{\frac{\epsilon \cdot f_i (E^{TX} + E^{RX})}{\beta}} \text{ since } f_i = \sum_{v \in V} f_v^i \quad \text{and} \quad f_v^0 = 0.
\end{aligned}
\tag{5}
$$

While $D(0) = \sum_{e \in E_1} l_0(e) \cdot C_t(e) = \sum_{e \in E_1} \frac{\delta}{C_t(e)} \cdot C_t(e) = |E_1| \cdot \delta = m_1 \delta$, Algorithm 3 stops when $D(i_0) \ge 1$. We, thus, have

$$
1 \le D(i_0) \le D(0) \cdot e^{\frac{\epsilon \cdot f_{i_0}(E^{TX} + E^{RX})}{\beta}} = (m_1 \delta) \cdot e^{\frac{\epsilon \cdot f_{i_0}(E^{TX} + E^{RX})}{\beta}}.
$$

This implies

$$
\frac{\beta}{f_{i_0}(E^{TX} + E^{RX})} \le \frac{\epsilon}{\ln \frac{1}{m_1 \delta}}. \tag{6}
$$

Let γ be the ratio of the dual and primal solutions, then $\gamma = \frac{\beta}{f_{i_0}(E^{TX} + E^{RX})} \log_{1+\epsilon} \frac{1+\epsilon}{\delta}$. By substituting the bound on $\frac{\beta}{f_{i_0}(E^{TX} + E^{RX})}$ from Inequality (6), we have

$$
\begin{aligned}
\gamma &= \frac{\beta}{f_{i_0}(E^{TX} + E^{RX})} \log_{1+\epsilon} \frac{1+\epsilon}{\delta} \le \frac{\epsilon \log_{1+\epsilon} \frac{1+\epsilon}{\delta}}{\ln \frac{1}{m_1 \delta}} \\
&= \frac{\epsilon}{\ln(1+\epsilon)} \cdot \left(\frac{\ln \frac{1+\epsilon}{\delta}}{\ln \frac{1}{m_1 \delta}}\right) \le \frac{\epsilon}{(1-\epsilon)\ln(1+\epsilon)} \\
&\le \frac{\epsilon}{(1-\epsilon)(\epsilon - \epsilon^2/2)} \le \frac{1}{(1-\epsilon)^2} \le \frac{1}{1-2\epsilon},
\end{aligned}
$$

due to the fact that $\frac{\ln \frac{1+\epsilon}{\delta}}{\ln \frac{1}{m_1 \delta}} = \frac{1}{1-\epsilon}$ when $\delta = \frac{1+\epsilon}{((1+\epsilon)m_1)^{1/\epsilon}}$. The theorem then follows. □

6 Distributed Implementation

In this section, we provide a distributed implementation of the proposed centralized algorithm Algorithm 1 in interval $t+1$. In a distributed environment, each node has only the knowledge of its neighbors and, thus, can only communicate with its neighbors. The distributed implementation of Algorithm 1 where Algorithm 3 is employed as it subroutine is described as follows:

First, to calculate the rate weight w_v of each source node $v \in V$, based on the data correlation graph $G(V, E, c_t)$ generated in interval t, instead of finding a maximum matching $M_{G'}$ in the centralized algorithm 1, a maximal matching in $G'(V, E')$ is found, using a distributed algorithm by Sanghavi et al. [27], [4], where G' is a subgraph of G. The detailed description of the Sanghavi et al.'s algorithm proceeds as follows:

Given a positive integer $K > 1$, each source node in network G' randomly decides to be a *seed* with a given probability p. Each seed chooses an *intended size* for its augmentation, uniformly from the set $\{1, 2, \ldots, K\}$. All seeds are the *active nodes*. Each seed then aims to find an augmented path within $2K+1$ iterations. Consequently, there are a number of augmented paths with lengths no more than $2K+1$, the algorithm finally terminates and returns a maximal matching that is derived from the augmented paths. They showed that the cardinality ratio of the maximal matching to the maximum matching is $\frac{K}{1+K}$ with a certain probability [27]. Note that $G(V, E, c_t)$ has identical topology as the harvesting sensor network $G(V, E)$. Thus, the implementation of the distributed algorithm by Sanghavi et al. [27], [4] in the sensor network is straightforward, omitted.

Second, assuming that a maximal matching in G' has been found, the rate weight assignment to each source node then can be performed in constant time, as each matched edge in the matching corresponds to a pair of a master and a slave. The maximal matching obtained can be sent to the sink, using any routing tree structure. For example, a minimum spanning tree rooted at the sink delivered by a distributed algorithm due to Gallager et al. [11] can be used for such a purpose.

Third, to perform the weighted, fair data rate allocation to each source node, the weighted fair rate allocation and routing problem can be reduced into a maximum weighted concurrent flow problem in network $G_1(V_1, E_1, C_t)$, where $C_t(v)$ is the energy budget of sensor node $v \in V$ in interval $t+1$. G_1 is embedded into the physical network $G(V, E)$ as follows: Each source node $v \in V$ in G accommodates two corresponding nodes v' and v'' in G_1 as well as the edge $\langle v', v'' \rangle \in E_1$. The physical link between two neighboring

nodes u and v in G can be treated at two directed edges $\langle v'', u' \rangle$ and $\langle u'', v' \rangle$ in G_1. Since G_1 is a directed network with edge capacities and the rate weight of each source node, `Algorithm 3` is then applied to G_1 which delivers a feasible solution to the maximum weighted concurrent flow problem. Notice that `Algorithm 3` essentially calls Dijkstra's algorithm iteratively. Thus, its efficient distributed implementation is determined by the distributed implementation of Dijkstra's algorithm, and there are many efficient distributed algorithms for the latter (e.g., [5]).

Having the shortest path tree T_i rooted at the sink in iteration i, finally, it assigns each source node an identical fractional data rate such that the flow is maximized, subject to the bottleneck rate edge in the tree. This can be implemented in $O(n)$ time with $O(n)$ messages, because finding the value of λ_i takes $O(n)$ time and $O(n)$ messages, using the tree topology T_i. The tree root broadcasts λ_i to all source nodes through the tree edges. Each source node v sets its data rate as $\lambda_i \cdot d'_v$ in iteration i. The lengths of the edges in T_i are then updated. Consequently, the distributed implementation `Algorithm 1` is stated by the following theorem.

Theorem 4. *Given an energy harvesting sensor network $G(V, E)$ with $|V| = n$ and $|E| = m$, there is a distributed implementation of `Algorithm 1` for the monitoring quality maximization problem in G where `Algorithm 3` is employed as its subroutine. `Algorithm 1` takes $O(n \log_{1+\epsilon} \frac{1+\epsilon}{\delta} \cdot (T_{SP} + O(n))$ time and $O(n \log_{1+\epsilon} \frac{1+\epsilon}{\delta} \cdot (M_{SP} + O(n))$ messages, where T_{SP} and M_{SP} are the time and message complexity of distributed Dijkstra's algorithm in G, ϵ is a constant with $0 \leq \epsilon \leq 1/2$ and $\delta = \frac{1+\epsilon}{((1+\epsilon)\cdot(2m+n))^{1/\epsilon}}$.*

Proof. We analyze the computational and message complexities of the distributed implementation of `Algorithm 1` as follows:

Finding a maximal matching takes $O(1)$ time and $O(m)$ messages. Since K is a fixed integer, it takes $O(1)$ time and $O(m)$ messages in each iteration to augment paths starting from seeds. Assigning a rate weight to each source node takes $O(1)$ time and $O(n)$ messages. Relaying the found maximal matching to the sink takes $O(n)$ time and $O(m + n \log n)$ messages as it takes $O(n)$ time and $O(m + n \log n)$ time to build a minimum spanning tree in G' rooted at the sink [11], the tree will serve as the routing tree. Embedding G' into the physical network G takes $O(1)$ time and $O(m)$ messages. Assuming that it takes $O(T_{SP})$ time and requires $O(M_{SP})$ messages to implement Dijkstra's algorithm in the distributed network G, where `Algorithm 3` invokes Dijkstra's algorithm at most $n \log_{1+\epsilon} \frac{1+\epsilon}{\delta}$ times. Thus, the distributed implementation of `Algorithm 1` takes

$$O\left((n + T_{SP}) \cdot n \log_{1+\epsilon} \frac{1+\epsilon}{\delta}\right) = O\left(n \log_{1+\epsilon} \frac{1+\epsilon}{\delta} \cdot (T_{SP} + n)\right)$$

time and requires

$$O\left((m + n \log n + (M_{SP} + n)) \cdot n \log_{1+\epsilon} \frac{1+\epsilon}{\delta}\right) = O\left(n \log_{1+\epsilon} \frac{1+\epsilon}{\delta} \cdot (M_{SP} + n)\right)$$

messages, where $\delta = \frac{1+\epsilon}{((1+\epsilon)(2m+n))^{1/\epsilon}}$. □

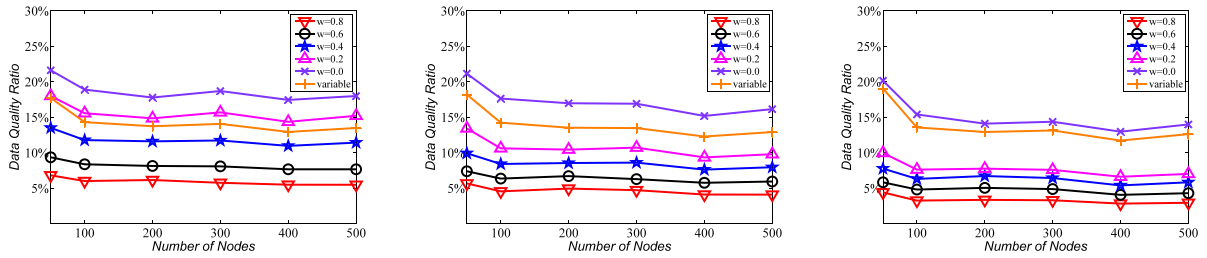
7 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithms and investigate the impact of different parameters on the performance.

7.1 Simulation Environment

We consider a wireless sensor network consisting of 50 to 500 sensors randomly deployed in a 100×100 square meters region. The base station is also randomly located. In all our experiments, we adopt the energy consumption parameters of real sensors MICA2 motes [8], where $E^{TX} = 14.4 \times 10^{-6}$ Joules/bit and $E^{RX} = 5.76 \times 10^{-6}$ Joules/bit. We assume that the maximum data rate of each sensor R_{max} is randomly drawn from the set $\{60, 80, 100\}$ packets per interval, and each interval are divided into $\tau = 100$ equal time slots. Each sensor is powered by a solar panel. The solar power harvesting profile is obtained from the Baseline Measurement System at the National Renewable Energy Laboratory [23]. For example, we here take the solar energy harvesting as an example [10]. The total amount of energy collected from a $37\ \mathrm{mm} \times 33\ \mathrm{mm}$ solar cell over a 48-hour period is $655.15\ \mathrm{mWh}$ in a sunny day and $313.70\ \mathrm{mWh}$ in a partly cloudy day. We further assume the energy budget (the amount of recharged energy) of each node in an interval is a random value ranged in $[6.53, 13.65]\ \mathrm{mJ/s}$. The accuracy parameter ϵ is 0.1 and the power a in the utility function is 2 in the default setting. Each value in figures is the mean of the results by applying each mentioned algorithm to 15 different network topologies of the same size. The running time is obtained on a desktop with Intel $Core$ i7 CPU and 4-GB RAM.

The sensing data in sensors are generated as follows: Each sensor node is assigned a sensing data sequence for 10 consecutive intervals, and within each interval a sensor is assigned 100 sensing data. The rate weights of source nodes in interval $i+1$ is calculated based on the data correlation information in interval i, $1 \leq i < 10$. The sensing data here are the real temperature sensing data collected from 54 sensor nodes between February 28, 2004 and April 5, 2004, by the Intel Berkeley Research Lab [14], where a sensor generates a reading every 31 seconds. Now, we map these real sensing data to the sensor nodes in a sensor network as follows: Considering the network consisting of 50 sensor nodes, the mapping between the real sensing data and these nodes is as follows: We map the sensing data sequence by a real sensor (e.g., sensor A) between 1:00 am and 23:00 pm on February 28, 2004 to sensor node 1 for this period, and map the sensing data sequence by another sensor B in this period to sensor node 2, and so on. In the end, the sensing data sequences of the 50 sensor nodes have been assigned. For a network size larger than 54, the mapping is done similarly. That is, we map the sensing data sequences generated in different time periods to different sensor nodes. For example, take a 100 sensor node network, we use the sensing data sequences collected by the first 50 sensors on February 28, 2004 as the sensing data sequences of the first 50 corresponding sensor nodes, and the sensing data sequences collected by these 50 sensors on March 2, 2004 as the sensing data sequences of the rest 50 sensor nodes.



(a) $\theta = 0.4$ (b) $\theta = 0.6$ (c) $\theta = 0.8$

Fig. 2. Impact of different θ and w on the data quality by algorithm 1 in which algorithm 3 is employed when $\epsilon = 0.1$.

7.2 Performance Evaluation of the Proposed Algorithm on the Quality of Monitoring

We first evaluate the performance of `Algorithm 1` in which `Algorithm 3` is employed as its subroutine. Fig. 2 plots the performance curves, by varying the confidence threshold θ from 0.4 to 0.8, the uniform weight w from 0 to 0.8, or a variable weight $1 - c_t(u, v)$, where $c_t(u, v)$ is the data correlation confidence between node v and its master u at interval t. In comparison with the unweighted case, where each source node has an identical weight of 1, Fig. 2 clearly indicates that the weighted data rate allocation improves the data quality of the network significantly in comparison with the unweighted rate allocation. When the uniform weight for each slave node is set to zero or assigned a variable weight, the proposed algorithm achieves a much better performance, as the resources (bandwidth and energy) previously occupied by the slave nodes now is now fully reallocated to the master and stand-alone nodes, particularly for those low rate nodes. Consequently, the low rate nodes can substantially improve their data rates. Specifically, define *the data quality ratio* as the ratio of the data quality delivered by the proposed algorithm based on the weighted rate assignment to the data quality delivered by the same algorithm based on the unweighted rate assignment. Then, when θ is fixed at 0.4, the data quality ratio is at least 18, 15, 11, 8, and 6 percent by varying the uniform rate weight w from 0 to 0.8 with step 0.2. Similarly, when θ is fixed at 0.6, the data quality ratio by the proposed algorithm is at least 16, 10, 8, 6, and 5 percent by varying w from 0 to 0.8. When θ is fixed at 0.8, the data quality ratio is at least 13, 7, 6, 5, and 3 percent by varying w from 0 to 0.8 with step 0.2. The data quality ratio is around 13 percent when the variable rate weight is used. In summary, with the increase on both the confidence threshold, the data quality ratio becomes deteriorating, no matter whether it is a uniform weight or a variable weight assignment. For a given confidence threshold, the smaller the uniform weight w, the higher the data quality delivered, because more resources released from the slave nodes will be utilized by their master and the other stand-alone nodes. Fig. 2 also shows that when the uniform weight w is fixed, the larger the confidence threshold θ, the less improvement the data quality, because fewer number of nodes in the network will become the slave nodes when θ is quite large. However, when $w = 0$, the data quality obtained is at the same level regardless of varying the confidence threshold. The reason behind is that the utility values of most master nodes (e.g., a master node u, its utility value r_u/R_u) are already very high (above 0.8), their utility gain becomes insignificant with any further increase on the data rates.

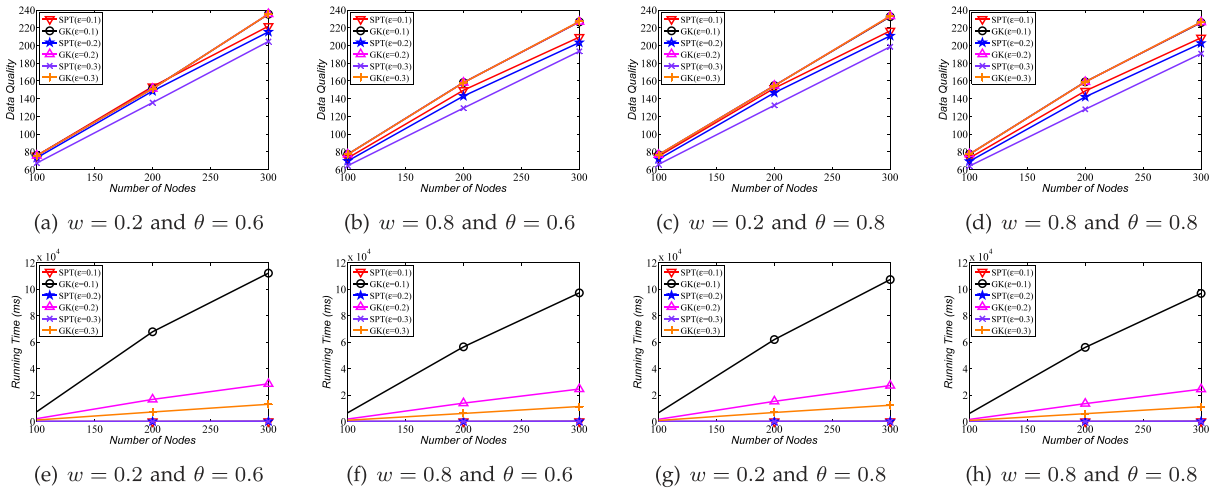
7.3 Impact of Different Data Rate Allocation Algorithms on the Quality

We then study the impact of the two subroutines, `Algorithm 2`, referred to as `GK`'s algorithm, and `Algorithm 3`, referred to as the `SPT`-based algorithm, on the data quality delivered by `Algorithm 1` under different parameter settings.

Fig. 3 plots the performance of `Algorithm 1` by employing two different subroutines `Algorithm 2` and `Algorithm 3`, respectively. In terms of the data quality, it can be seen from this figure that although both subroutines deliver almost identical results, the running time of `Algorithm 3` is substantially less than that of `Algorithm 2`. To be more specific, the running time of `Algorithm 3` is no more than 1 percent of that of `Algorithm 2`. Note that the dominant running time of `Algorithm 1` is the subroutine `Algorithm 2` or the subroutine `Algorithm 3`. Specifically, assume that ϵ is set to 0.1. When w and θ are fixed at 0.2 and 0.6, respectively, the data quality of `Algorithm 3` is no less than 95 percent of that of `Algorithm 2`. When w and θ are fixed at 0.8 and 0.6, respectively, the data quality of `Algorithm 3` is no less than 93 percent of that of `Algorithm 2`. When w and θ are fixed at 0.2 and 0.8, respectively, the data quality of `Algorithm 3` is no less than 94 percent of that of `Algorithm 2`. When both w and θ are fixed at 0.8, the data quality of `Algorithm 3` is no less than 93 percent of that of `Algorithm 2`. Fig. 3 implies that the smaller the accuracy parameter ϵ, the bigger the running time gap between the two subroutines. Meanwhile, it is noted that from Fig. 3 when fixing both θ and ϵ but varying the uniform weight w, the smaller the uniform weight is, the higher the data quality will be.

8 Conclusion

In this paper, we have studied the monitoring quality maximization problem in harvesting sensor networks by incorporating spatial data correlation into consideration. We first formulated the problem as a novel optimization problem, namely, the monitoring quality maximization problem. The key to approach this problem is to solve the weighted, fair data rate allocation and flow routing problem, for which we devised a fast approximation algorithm with a provable approximation ratio. Also, a



(a) $w = 0.2$ and $\theta = 0.6$ (b) $w = 0.8$ and $\theta = 0.6$ (c) $w = 0.2$ and $\theta = 0.8$ (d) $w = 0.8$ and $\theta = 0.8$

(e) $w = 0.2$ and $\theta = 0.6$ (f) $w = 0.8$ and $\theta = 0.6$ (g) $w = 0.2$ and $\theta = 0.8$ (h) $w = 0.8$ and $\theta = 0.8$

Fig. 3. The data quality delivered and the running time between algorithms SPT and GK.

distributed implementation of the proposed algorithm is provided, too. The core ingredients in the design of algorithm include dynamic data rate weight assignment and a reduction technique to reduce the problem to a special maximum weighted concurrent flow problem. We finally conducted extensive experiments by simulation to evaluate the performance of the proposed algorithm. We also investigated the impact of various parameters on the performance of algorithms. The experimental results demonstrate that the proposed algorithm for the monitoring quality maximization is promising. Particularly the approximate solution to the weighted, fair data rate allocation and flow routing problem is the very first approximation algorithm, which may have independent of interest by itself and can be applicable to solve other optimization problems beyond wireless harvesting sensor networks.

ACKNOWLEDGMENTS

It is acknowledged that Weifa Liang's research was funded by the Australian Research Council Discovery grant DP120102627. Xiaohua Jia's research was partially supported by the Natural Science Foundation of China under Grant No. 61173137. They would like to thank anonymous referees for their helpful comments.

REFERENCES

[1] R.K. Ahuja, T.L. Magnaati, and J.B. Orlin, *Network Flows: Theory, Algorithms, and Applications.* Prentice-Hall, Inc., 1993.

[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks,* vol. 38, pp. 393-422, 2002.

[3] M. Alan, C. David, P. Joseph, S. Robert, and A. John, "Wireless Sensor Networks for Habitat Monitoring," *Proc. First ACM Int'l Workshop Wireless Sensor Networks and Applications (WSNA),* pp. 88-97, 2002.

[4] L.X. Bui, S. Sanghavi, and R. Srikant, "Distributed Link Scheduling with Constant Overhead," *IEEE/ACM Trans. Networking,* vol. 17, no. 5, pp. 1467-1480, Oct. 2009.

[5] K.M. Chandy and J. Misra, "Distributed Computation on Graphs: Shortest Path Algorithms." *Comm. ACM,* vol. 25, pp. 833-837, 1982.

[6] S. Chen, Y. Fang, and Y. Xia, "Lexicographic Maxmin Fairness for Data Collection in Wireless Sensor Networks," *IEEE Trans. Mobile Computing,* vol. 6, no. 7, pp. 762-776, July 2007.

[7] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms,* third ed. MIT Press, 2009.

[8] Crossbow Inc, "MPR-Mote Processor Radio Board Users Manual," 2003.

[9] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Witehouse, and D. Culler, "Trio: Enabling Sustainable and Scalable Outdoor Wireless Sensor Network Deployments," *Proc. Fifth Int'l Conf. Information Processing in Sensor Networks (IPSN),* pp. 407-415, 2006.

[10] K.-W. Fan, Z.-Z. Zheng, and P. Sinha, "Steady and Fair Rate Allocation for Rechargeable Sensors in Perpetual Sensor Networks," *Proc. Sixth ACM Conf. Embedded Network Sensor Systems (SenSys),* pp. 239-252, 2008.

[11] R.G. Gallager, A. Humblet, and P.M. Spira, "A Distributed Algorithm for Minimum-Weight Spanning Trees." *ACM Trans. Programming Languages and Systems,* vol. 5, pp. 66-77, 1983.

[12] N. Garg and J. Könemann, "Faster and Simpler Algorithms for Multicommodity Flow and Other Fractional Packing Problems," *Proc. 39th Ann. Symp. Foundation Computer Sciences (FOCS '98),* pp. 300-309, 1998.

[13] Y.T. Hou, Y. Shi, and H.D. Sherali, "Rate Allocation and Network Lifetime Problems for Wireless Sensor Networks," *IEEE/ACM Trans Networking,* vol. 16, no. 2, pp. 321-334, Apr. 2008.

[14] "Intel Berkeley Research Lab," http://db.csail.mit.edu/labdata/labdata.html, 2013.

[15] X. Jiang, J. Polastre, and D. Culler, "Perpetual Environmentally Powered Sensor Networks," *Proc. Fourth Int'l Symp. Information Processing in Sensor Networks (IPSN),* pp. 463-468, 2005.

[16] A. Kansal, J. Hsu, S. Zahedi, and M.B. Srivastava, "Power Management in Energy Harvesting Sensor Networks," *ACM Trans. Embedded Computing Systems,* vol. 6, article 32, Sept. 2007.

[17] K. Lin, J. Yu, J. Hsu, S. Zahedi, D. Lee, J. Friedman, A. Kasal, V. Raghunathan, and M. Srivastava, "Heliomote: Enabling Long-Lived Sensor Networks Through Solar Energy Harvesting," *Proc. Third Int'l Conf. Embedded Networked Sensor Systems (SenSys),* p. 309, 2005.

[18] R.-S. Liu, P. Sinha, and C.E. Koksal, "Joint Energy Management and Resource Allocation in Rechargeable Sensor Networks," *Proc. IEEE INFOCOM '10,* 2010.

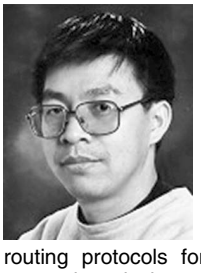
[19] R.-S. Liu, K.-W. Fan, Z. Zheng, and P. Sinha, "Perpetual and Fair Data Collection for Environmental Energy Harvesting Sensor Networks," *IEEE/ACM Trans. Networking,* vol. 19, no. 4, pp. 947-960, Aug. 2011.

[20] J. Luo and J.P. Hubaux, "Joint Sink Mobility and Routing to Maximize the Lifetime of Wireless Sensor Networks: The Case of Constrained Mobility," *IEEE/ACM Trans. Networking,* vol. 18, no. 3, pp. 871-884, June 2010.

[21] S. Micali and V.V. Vazirani, "An $O(\sqrt{|V|} \cdot E)$ Algorithm for Finding Maximum Matching in General Graphs," *Proc. IEEE 21st Symp. Foundations of Computer Science,* pp. 17-27, 1980.

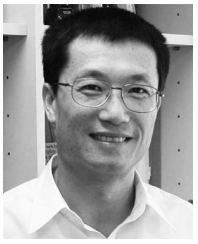
[22] C. Moser, J. Chen, and L. Thiele, "Reward Maximization for Embedded Systems with Renewable Energies," *Proc. IEEE 14th Int'l Conf. Embedded and Real-Time Computing Systems and Applications,* 2008.

[23] "National Renewable Energy Laboratory," http://www.nrel.gov, 2013.
[24] D.K. Noh, L. Wang, Y. Yang, H.K. Le, and T. Abdelzaher, "Minimum Variance Energy Allocation for Solar-Powered Sensor Systems," *Proc. Fifth Int'l Conf. Distributed Computing in Sensor Systems (DOCSS '09)*, 2009.
[25] C. Park and P. Chou, "AmbiMax: Autonomous Energy Harvesting Platform for Multi-Supply Wireless Sensor Nodes," *Proc. IEEE Third Ann. Comm. Soc. Sensor and Ad Hoc Comm. Networks (SECON)*, pp. 168-177, 2006.
[26] S. Plotkin, D. Shmoys, and E. Tardos, "Fast Approximation Algorithms For Fractional Packing and Covering Problem," *Math. Operations Research*, vol. 20, pp. 257-301, 1995.
[27] S. Sanghavi, L. Bui, and R. Srikant, "Distributed Link Scheduling with Constant Overhead," *Proc. ACM SIGMETRICS Int'l Conf. Measurement and Modeling Computer Systems (SIGMETRICS '07)*, pp. 313-324, 2007.
[28] L. Su, Y. Gao, Y. Yang, and G. Cao, "Towards Optimal Rate Allocation for Data Aggregation in Wireless Sensor Networks," *Proc. ACM MobiHoc '11*, 2011.
[29] C.M. Vigorito, D. Ganesan, and A.G. Barto, "Adaptive Control Duty Cycling in Energy-Harvesting Wireless Sensor Networks," *Proc. IEEE Conf. Sensor and Ad Hoc Comm. Networks (SECON) '07*, 2007.
[30] X. Wang and K. Kar, "Cross-Layer Control for End-to-End Proportional Fairness in Wireless Networks with Random Access," *Proc. ACM MobiHoc '05*, pp. 157-168, 2005.
[31] Y. Xue, Y. Cui, and K. Nahrstedt, "Maximizing Lifetime for Data Aggregation in Wireless Sensor Networks," *Mobile Networks and Applications*, vol. 10, pp. 853-864, 2005.
[32] B. Zhang, R. Simon, and H. Aydin, "Maximal Utility Rate Allocation for Energy Harvesting Wireless Sensor Networks," *Proc. 14th ACM Int'l Conf. Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '11)*, 2011.



Weifa Liang (M'99-SM'01) received the BSc degree from Wuhan University, China in 1984, the ME degree from the University of Science and Technology of China in 1989, and the PhD degree from the Australian National University in 1998, and all in computer science. He is currently an associate professor in the Research School of Computer Science at the Australian National University. His research interests include design and analysis of energy-efficient routing protocols for wireless ad hoc and sensor networks, cloud computing, design and analysis of parallel and distributed algorithms, approximation algorithms, and graph theory. He is a senior member of the IEEE.

Xiaojiang Ren received the BE and ME degrees from the Huazhong University of Science and Technology in China in 2004 and 2007, respectively, and is currently working toward the PhD degree in the Research School of Computer Science at the Australian National University. His research interests include wireless sensor networks, routing protocol design for wireless networks, and optimization problems. He is a student member of the IEEE.



Xiaohua Jia received the BSc and MEng degrees in 1984 and 1987, respectively, from the University of Science and Technology of China, and DSc in 1991 in information science from the University of Tokyo. He is currently a chair professor with Department of Computer Science at City University of Hong Kong. His research interests include cloud computing and distributed systems, computer networks, wireless sensor networks and mobile wireless networks. He is an editor of *IEEE Transactions on Parallel and Distributed Systems* (2006-2009), *Wireless Networks*, *Journal of World Wide Web*, *Journal of Combinatorial Optimization*, and so on. He is the general chair of ACM MobiHoc 2008, TPC co-chair of IEEE MASS 2009, area-chair of IEEE INFOCOM 2010, TPC co-chair of IEEE GlobeCom 2010 Ad Hoc and Sensor Networking Symposium, and Panel co-chair of IEEE INFOCOM 2011. He is a fellow of the IEEE.

Xu Xu received the BS degree from China Agricultural University in 2006, the ME degree from the Institute of Computing Technology, Chinese Academy of Sciences in 2009, both in computer science, and is currently working toward the PhD degree in the Research School of Computer Science at the Australian National University. Her research interests include sink mobility in wireless sensor networks, routing protocol design for wireless ad hoc and sensor networks, data gathering in wireless sensor networks, and optimization problems.