

Max-min Fair Rate Allocation and Routing in Energy Harvesting Networks: Algorithmic Analysis

Jelena Marašević¹, Cliff Stein², Gil Zussman¹

¹Department of Electrical Engineering,

²Department of Industrial Engineering and Operations Research

Columbia University

{jelena@ee, cliff@ieor, gil@ee}.columbia.edu

Abstract

This paper considers max-min fair rate allocation and routing in energy harvesting networks where fairness is required among *both the nodes and the time slots*. Unlike most previous work on fairness, we focus on *multihop topologies* and consider *different routing methods*. We assume a predictable energy profile and focus on the design of efficient and optimal algorithms that can serve as benchmarks for distributed and approximate algorithms. We first develop an algorithm that obtains a max-min fair rate assignment for any given (time-variable or time-invariable) *unsplittable routing* or a *routing tree*. For *time-invariable unsplittable routing*, we also develop an algorithm that finds routes that maximize the minimum rate assigned to any node in any slot. For *fractional routing*, we study the joint routing and rate assignment problem. We develop an algorithm for the *time-invariable* case with constant rates. We show that the *time-variable* case is at least as hard as the 2-commodity feasible flow problem and design an FPTAS to combat the high running time. Finally, we show that finding an unsplittable routing or a routing tree that provides lexicographically maximum rate assignment (i.e., that is the best in the max-min fairness terms) is NP-hard, even for a time horizon of a single slot. Our analysis provides insights into the problem structure and can be applied to other related fairness problems.

Keywords: Energy Harvesting, Energy Adaptive Networking, Sensor networks, Routing, Fairness

1 Introduction

Recent advances in the development of ultra-low-power transceivers and energy harvesting devices (e.g., solar cells) will enable self-sustainable and perpetual wireless networks [11, 14, 15]. In contrast to legacy wireless sensor networks, where the available energy only decreases as the nodes sense and forward data, in energy harvesting networks the available energy can also increase through a replenishment process. This results in significantly more complex variations of the available energy, which pose challenges in the design of resource allocation and routing algorithms.

The problems of resource allocation, scheduling, and routing in energy harvesting networks have received considerable attention [2, 4, 9, 12, 13, 16–18, 23, 24, 29, 32, 34]. Most existing work considers simple networks consisting of a single node or a link [2, 4, 13, 16, 29, 34]. Moreover, fair rate assignment has not been thoroughly studied, and most of the work either focuses on maximizing the total (or average) throughput [2, 4, 9, 12, 18, 23, 27, 29, 32, 34], or considers fairness either *only over nodes* [24] or *only over time* [13, 16]. An exception is [17], which requires fairness over both the nodes and the time, but is limited to *two nodes*.

In this paper, we study the max-min fair rate assignment and routing problems, requiring fairness over both nodes and time slots, and with the goal of designing optimal and efficient algorithms.

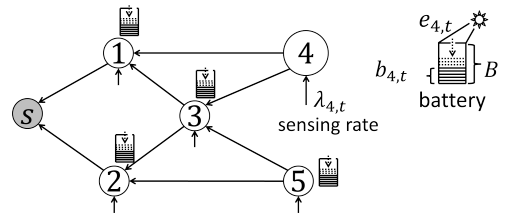


Figure 1: A simple energy harvesting network: the nodes sense the environment and forward the data to a sink s . Each node has a battery of capacity B . At time t a node i 's battery level is $b_{i,t}$, it harvests $e_{i,t}$ units of energy, and senses at data rate $\lambda_{i,t}$.

Following [9, 13, 16, 17, 23, 24], we assume that the harvested energy is known for each node over a finite time horizon. Such a setting corresponds to a highly-predictable energy profile, and can also be used as a benchmark for evaluating algorithms designed for unpredictable energy profiles. We consider an energy harvesting sensor network with a single sink node, and network connectivity modeled by a directed graph (Fig. 1). Each node senses some data from its surrounding (e.g., air pressure, temperature, radiation level), and sends it to the sink. The nodes spend their energy on sensing, sending, and receiving data.

1.1 Fairness Motivation

Two natural conditions that a network should satisfy are:

- (i) balanced data acquisition across the entire network, and
- (ii) persistent operation (i.e., even when the environmental energy is not available for harvesting).

The condition (i) is commonly reinforced by requiring fairness of the sensing rates over network nodes. We note that in the network model we consider, due to these different energy costs for sending, sensing, and receiving data, throughput maximization can be inherently unfair even in the static case. Consider a simple network with two energy harvesting nodes a and b and a sink s illustrated in Fig. 2. Assume that a has one unit of energy available, and b has two units of energy. Let c_{st} denote the joint cost of sensing and sending a unit flow, c_{rt} denote the joint cost for receiving and sending a unit flow. Let λ_a and λ_b denote the sensing rates assigned to the nodes a and b , respectively. Suppose that the objective is to maximize $\lambda_a + \lambda_b$. If $c_{st} = 1$, $c_{rt} = 2$, then in the optimal solution $\lambda_a = 1$ and $\lambda_b = 0$. Conversely, if $c_{st} = 2$, $c_{rt} = 1$, then in the optimal solution $\lambda_a = 0$ and $\lambda_b = 1$. This example easily extends to more general degenerate cases in which throughput-maximum solution assigns non-zero sensing rates only to one part of the network, whereas the remaining nodes do not send any data to the sink.

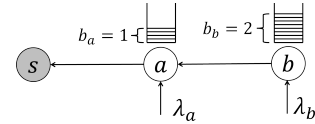


Figure 2: An example of a network in which throughput maximization can result in a very unfair rate allocation among the nodes.

One approach to achieving (ii) is by assigning constant sensing rates to the nodes. However, this approach can result in underutilization of the available energy. As a simple example, consider a node that harvests outdoor light energy over a 24-hour time horizon. If the battery capacity is small, then the sensing rate must be low to prevent battery depletion during the nighttime. However, during the daytime, when the harvesting rates are high, a low sensing rate prevents full utilization of the energy that can be harvested. Therefore, it is advantageous to vary the sensing rates over time. However, fairness must be required over time slots to prevent the rate assignment algorithm from assigning high rates during periods of high energy availability, and zero rates when no energy is available for harvesting.

1.2 Routing Types

We consider different routing types, which are illustrated in Fig. 3.

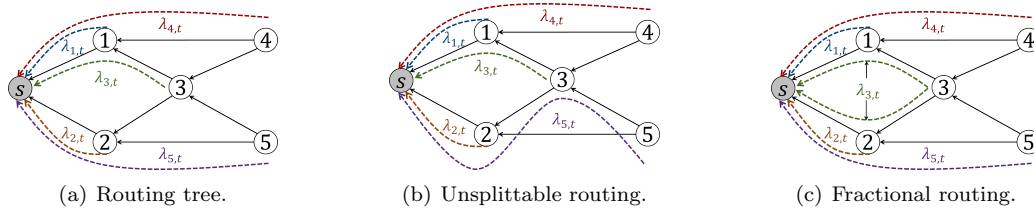


Figure 3: Routing types: (a) a routing tree, (b) unsplittable routing: each node sends its data over one path, (c) fractional routing: nodes can send their data over multiple paths. Paths are represented by dashed lines.

Each routing type incurs different trade-offs between the supported sensing rates¹ and the required amount of control information. Routing types with higher number of active links require more control

¹A metric of performance can be the minimum sensing rate that gets assigned to any node in any time slot.

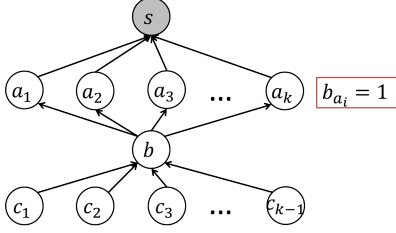


Figure 4: A network example in which unsplitable routing provides minimum sensing rate that is $\Omega(n)$ higher than for any routing tree. Assume $c_{st} = c_{rt} = 1$ and $T = 1$. Available energies at all the nodes a_i , $i \in \{1, \dots, k\}$ are equal to 1, as shown in the box next to the nodes. Other nodes have energies that are high enough so that they are not constraining. In any routing tree, b has some a_i as its parent, so $\lambda_{a_i} = \lambda_b = \lambda_{c_1} = \dots = \lambda_{c_{k-1}} = 1/(k+1)$ and $\lambda_j = 1$ for $j \neq i$. In an unsplitable routing with paths $p_{a_i} = \{a_i, s\}$, $p_{c_i} = \{c_i, b, a_i, s\}$, and $p_b = \{b, a_k, s\}$, all the rates are equal to $1/2$. As $k = \Theta(n)$, the minimum rate improves by $\Omega((k+1)/2) = \Omega(n)$.

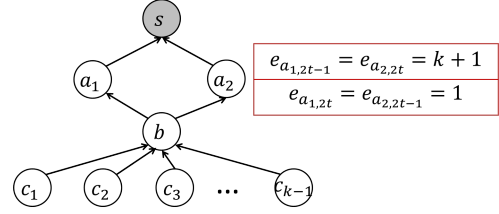


Figure 5: A network example in which time-variable routing solution provides minimum sensing rate that is $\Omega(n)$ higher than for any time-invariable routing. The batteries of a_1 and a_2 are initially empty, and the battery capacity at all the nodes is $B = 1$. Harvested energies over time slots for nodes a_1 and a_2 are shown in the box next to them. Other nodes are assumed not to be energy constraining. In any time-invariable routing, at least one of a_1, a_2 has $\Omega(k) = \Omega(n)$ descendants, forcing its rate to the value of $1/\Omega(n)$ in the slots in which the harvested energy is equal to 1. In a routing in which b sends the data only through a_1 in odd slots and only through a_2 in even slots: $\lambda_b = \lambda_{c_1} = \dots = \lambda_{c_{k-1}} = 1$.

information to be exchanged between neighboring nodes (e.g., to maintain synchronization), and complicate the transmission and/or sleep-wake scheduling implementation. Moreover, energy consumed by the control messages can affect achievable rates significantly, due to limited energy budget, as confirmed via experiments in [15]. Below we outline the main characteristics of the routing types we consider.

Routing Tree—the simplest form of routing, in which every node sends all of the data it senses and receives to a single neighboring (parent) node. It requires minimum number of active links, yielding minimum energy consumption due to control messages. However, in general, it provides the lowest sensing rates (see more details below).

Unsplittable Routing—a single-path routing, in which every node sends all of its sensed data over a single path to the sink (a routing tree is a special case of the unsplittable routing, in which all the paths incoming into node i outgo via the same edge). There are simple cases in which unsplittable routing provides a rate assignment with the minimum sensing rate $\Omega(n)$ times higher than in a routing tree, where n is the number of nodes (see Fig. 4 for an example). However, in general, it has higher number of active links than the routing tree, yielding higher energy consumption for control information.

Fractional Routing—a multi-path routing, in which each node can split its data over multiple paths to the sink (unsplittable routing is a special case of fractional routing in which every node has one path to the sink). It is the most general routing that subsumes both routing trees and unsplittable routings, and, therefore, provides the best sensing rates. However, it utilizes the highest number of links, yielding the highest energy consumption due to control messages.

Time-invariable vs Time-variable Routing—A routing is *time-invariable* if every node uses the same (set of) path(s) in each time slot to send its data to the sink. If the paths change over time, the routing is *time-variable*.² While there are cases in which time-variable routing provides a rate assignment with the minimum sensing rate $\Omega(n)$ times higher than in the time-invariable case (see, e.g., Fig. 5), it requires substantial control information exchange for routing reconfigurations, yielding high energy consumption.

1.3 Our Contributions

For the *unsplittable routing* and *routing tree*, we design a fully-combinatorial algorithm that solves the *max-min fair rate assignment* problem, both in the time-variable and time-invariable settings, when the routing is provided at the input. We then turn to *fractional routing*, considering two settings: time-variable and

²Whether the rates are constant or time-variable is independent of whether the routing is time-variable or not.

time-invariable. We demonstrate that in the *time-variable* setting verifying whether a given rate assignment is feasible is at least as hard as solving a feasible 2-commodity flow. This result implies that, to our current knowledge, it is unlikely that max-min fair time-variable fractional routing³ can be solved without the use of linear programming. To combat the high running time induced by the linear programming, we develop a fully polynomial time approximation scheme (FPTAS). For the *time-invariable setting*, we provide a fully-combinatorial algorithm that determines a max-min fair routing with constant rates.

Our rate assignment algorithms rely on the well-known water-filling framework, which is described in Section 4. It is important to note that water-filling is a framework—not an algorithm—and therefore it does not specify how to solve the maximization nor fixing of the rates steps (see Section 4). Even though general algorithms for implementing water-filling such as e.g., [8, 24, 31] can be adapted to solve the rate assignment problems studied in this paper, their implementation would involve solving $O(n^2T^2)$ linear programs (LPs) with $O(mT)$ variables and $O(nT)$ constraints, thus resulting in an unacceptably high running time. Moreover, such algorithms do not provide insights into the problem structure. Our algorithms are devised relying on the problem structure, and in most cases do not use linear programming. The only exception is the algorithm for time-variable fractional routing (Section 6), which solves $O(nT)$ LPs, and thus provides at least $O(nT)$ -fold improvement as compared to an adaptation of [8, 24, 31]. Furthermore, each LP in our solution searches over much smaller space (in fact, only within the ϵ -region of the starting point, for any ϵ -approximation)

We show that *determining* an unsplittable routing or a routing tree that supports lexicographically maximum rate assignment is NP-hard even for a single time slot. On the other hand, as a positive result, we develop an algorithm that determines a time-invariable unsplittable routing that maximizes the minimum sensing rate assigned to any node in any time slot.

The considered problems generalize classical max-min fair routing problems that have been studied outside the area of energy harvesting networks, such as: max-min fair fractional routing [28], max-min fair unsplittable routing [21], and bottleneck routing [3]. In contrast to the problems studied in [3, 21, 28], our model allows different costs for flow generation and forwarding, and has time-variable node capacities determined by the available energies at the nodes. We note that studying networks with node capacities is as general as studying networks with capacitated edges, since there are standard methods for transforming one of these two problems into another (see, e.g., [1]). Therefore, we believe that the results can find applications in other related areas.

1.4 Organization of the paper

The rest of the paper is organized as follows. Section 2 provides the model and problem formulations, which are placed in the context of related work in Section 3. Section 4 describes the connection between max-min fairness and lexicographic maximization. Section 5 considers rate assignment in unsplittable routing, while Sections 6 and 7 study fractional routing and rate assignment in time-variable and time-invariable settings, respectively. Section 8 provides hardness results for determining unsplittable routing or a routing tree. Section 9 provides conclusions and outlines possible future directions.

2 Model and Problem Formulation

We consider a network that consists of n energy harvesting nodes and one sink node (see Fig. 1). The sink is the central point at which all the sensed data is collected, and is assumed not to be energy constrained. In the rest of the paper, the term “sink” will be used for the sink node. The connectivity between the nodes is modeled by a directed graph $G = (V, E)$, where $|V| = n + 1$ (n nodes and the sink), and $|E| = m$. We assume without loss of generality that every node has a directed path to the sink, because otherwise it can be removed from the graph. The main notation is summarized in Table 1.

Each node is equipped with a rechargeable battery of finite capacity B . The time horizon is T time slots. The duration of a time slot is assumed to be much longer than the duration of a single data packet, but short enough so that the rate of energy harvesting does not change during a slot. For example, if outdoor light energy is harvested, one time slot can be at the order of a minute. In a time slot t , a node i harvests $e_{i,t}$

³We refer to a routing as max-min fair if it provides a lexicographically maximum rate assignment. The notions of max-min fairness and lexicographical ordering of vectors are defined in Section 4.

Table 1: Nomenclature.

| | | |
|-----------|------------------|--|
| inputs | n | Number of energy harvesting nodes |
| | T | Time horizon |
| | i | Node index, $i \in \{1, 2, \dots, n\}$ |
| | t | Time index, $t \in \{1, \dots, T\}$ |
| | B | Battery capacity |
| | $e_{i,t}$ | Harvested energy at node i in time slot t |
| | c_s | Energy spent for sensing a unit flow |
| | c_{tx} | Energy spent for transmitting a unit flow |
| variables | c_{rx} | Energy spent for receiving a unit flow |
| | $\lambda_{i,t}$ | Sensing rate of node i in time slot t |
| | $f_{ij,t}$ | Flow on link (i, j) in time slot t |
| | $b_{i,t}$ | Battery level at node i at the beginning of time slot t |
| notation | c_{st} | Energy spent for jointly sensing and transmitting a unit flow: $c_{st} = c_s + c_{tx}$ |
| | c_{rt} | Energy spent for jointly receiving and transmitting a unit flow: $c_{rt} = c_{rx} + c_{tx}$ |
| | $f_{i,t}^\Sigma$ | Total flow entering node i in time slot t : $f_{i,t}^\Sigma = \sum_{j:(j,i) \in E} f_{ji,t}$ |

units of energy. The battery level of a node i at the beginning of a time slot t is $b_{i,t}$. We follow a predictable energy profile [9, 13, 16, 17, 23, 24], and assume that all the values of harvested energy $e_{i,t}$, $i \in \{1, \dots, n\}$, $t \in \{1, \dots, T\}$, battery capacity B , and all the initial battery levels $b_{i,1}$, $i \in \{1, \dots, n\}$ are known and finite.

A node i in slot t senses data at rate $\lambda_{i,t}$. A node forwards all the data it senses and receives towards the sink. The flow on a link (i, j) in slot t is denoted by $f_{ij,t}$. Each node spends c_s energy units to sense a unit flow, and c_{tx} , respectively c_{rx} , energy units to transmit, respectively receive, a unit flow.

The feasible region \mathcal{R} for the sensing rates and flows is determined by the following set of linear⁴ constraints:

$$\begin{aligned} \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} : \\ \sum_{(j,i) \in E} f_{ji,t} + \lambda_{i,t} = \sum_{(i,j) \in E} f_{ij,t} \end{aligned} \quad (1)$$

$$b_{i,t+1} = \min\{B, \quad b_{i,t} + e_{i,t} - (c_{rt} f_{i,t}^\Sigma + c_{st} \lambda_{i,t})\} \quad (2)$$

$$b_{i,t+1} \geq 0, \lambda_{i,t} \geq 0, f_{ij,t} \geq 0, \forall (i, j) \in E \quad (3)$$

where $f_{i,t}^\Sigma \equiv \sum_{(j,i) \in E} f_{ji,t}$, $c_{st} \equiv c_s + c_{tx}$, and $c_{rt} \equiv c_{rx} + c_{tx}$. Eq. (1) is a classical flow conservation constraint, while (2) models battery evolution over time slots.

Similar to the definition of max-min fairness in [3], define a rate assignment $\{\lambda_{i,t}\}$, $i \in \{1, \dots, n\}$, $t \in \{1, \dots, T\}$, to be max-min fair if no $\lambda_{i,t}$ can be increased without either losing feasibility or decreasing some other rate $\lambda_{j,\tau} \leq \lambda_{i,t}$. Max-min fairness is closely related to lexicographic maximization, as will be explained in Section 4.

In some of the problems, the routing is provided at the input as a set of paths $\mathcal{P} = \{p_{i,t}\}$, for $i \in \{1, \dots, n\}$, $t \in \{1, \dots, T\}$. In such a case, \mathcal{R} should be interpreted with respect to \mathcal{P} , instead with respect to the input graph G .

2.1 Considered problems

We examine different routing types, in time-variable and time-invariable settings, as described in the introduction. For the unsplittable routing and routing trees, we examine the problems of *determining a rate assignment* and *determining a routing* separately, as described below. Table 1 summarizes inputs and variables.

⁴Note that we treat eq. (2) as a linear constraint, since the problems we are solving are maximizing $\lambda_{i,t}$'s (under the max-min fairness criterion), and (2) can be replaced by $b_{i,t+1} \leq B$ and $b_{i,t+1} \leq b_{i,t} + e_{i,t} - (c_{rt} f_{i,t}^\Sigma + c_{st} \lambda_{i,t})$ while leading to the same solution for $\{\lambda_{i,t}\}$.

P-UNSPLITTABLE-RATES: For a given time-variable unsplittable routing $\mathcal{P} = \{p_{i,t}\}$, determine the max-min fair assignment of the rates $\{\lambda_{i,t}\}$. Note that this setting subsumes time-invariable unsplittable routing, time-invariable routing tree, and time-variable routing tree.

P-UNSPLITTABLE-FIND: Associate with each (time-invariable or time-variable) unsplittable routing \mathcal{P} , a set of sensing rates $\{\lambda_{i,t}^{\mathcal{P}}\}$ that optimally solves P-UNSPLITTABLE-RATES. Determine an unsplittable routing \mathcal{P} that provides the lexicographically maximum assignment of rates $\{\lambda_{i,t}^{\mathcal{P}}\}$.

P-TREE-FIND: Let \mathcal{T} denote a (time-invariable or time-variable) routing tree on the input graph G . Associate with each \mathcal{T} a set of sensing rates $\{\lambda_{i,t}^{\mathcal{T}}\}$ that optimally solves P-UNSPLITTABLE-RATES. Determine \mathcal{T} that provides the lexicographically maximum assignment of rates $\{\lambda_{i,t}^{\mathcal{T}}\}$.

For the fractional routing, we study the following two variants of max-min fair routing, where the routing and the rate assignment are determined jointly.

P-FRACTIONAL: Determine a *time-variable* fractional routing that provides the max-min fair rate assignment $\{\lambda_{i,t}\}$, together with the max-min fair rate assignment $\{\lambda_{i,t}\}$, considering all the (time-variable, fractional) routings.

P-FIXED-FRACTIONAL: Determine a *time-invariable* fractional routing that provides the max-min fair time-invariable rate assignment $\{\lambda_{i,t}\} = \{\lambda_i\}$, together with the max-min fair rate assignment $\{\lambda_i\}$, considering all the (time-invariable, fractional) routings with rates that are constant over time.

3 Related Work

Energy-harvesting Networks. Rate assignment in energy harvesting networks in the case of a single node or a link was studied in [2, 4, 9, 13, 16, 29, 34]. In [9], the solution for one node is extended to the network setting by (1) formulating a convex optimization problem, and solving it using the Lagrange duality theory, and (2) using a heuristic algorithm. Both of these two solutions focus on maximizing the sum network throughput, and do not consider fairness.

Resource allocation and scheduling for network-wide scenarios using the Lyapunov optimization technique was studied in [12, 18, 27, 32]. While the work in [12, 18, 27, 32] can support unpredictable energy profiles, it focuses on the (sum-utility of) time-average rates, which is, in general, time-unfair. The design of online algorithms for resource allocation and routing was studied in [10, 23].

Max-min time-fair rate assignment for a single node or a link was studied in [13, 16], while max-min fair energy allocation for single-hop and two-hop scenarios was studied in [17]. Similar to our work, [17] requires fairness over both the nodes and the time slots, but considers only two energy harvesting nodes. The work on max-min fairness in network-wide scenarios [24] is explained in more detail below.

Sensor Networks. Problems P-FRACTIONAL, P-FIXED-FRACTIONAL, and P-UNSPLITTABLE-RATES are related to the maximum lifetime routing problems (see, e.g., [6, 26] and the follow-up work) in the following sense. In our model, maximization of the minimum sensing rate is equivalent to the network lifetime maximization in sensor networks, *but only if the system is observed for $T = 1$* . Namely, the nodes have the initial energy, and no harvesting happens over time.

Determining a maximum lifetime tree in sensor networks as in [5] is a special case of P-TREE-FIND. We extend the NP-hardness result from [5] and provide a lower bound of $\Omega(\log n)$ for the approximation ratio (for both [5] and P-TREE-FIND), where n is the number of nodes in the network.

Max-min Fair Unsplittable Routing. Rate assignment in unsplittable routing was studied extensively (see [3, 7] and references therein). P-UNSPLITTABLE-RATES reduces to the problem studied in [3, 7] for $c_{st} = c_{rt}$ and $T = 1$. In the energy harvesting network setting, this problem has been studied in [24], for rates that are constant over time and a time-invariable routing tree. We consider a more general case than in [24], where the rates are time-variable, fairness is required over both network nodes and time slots, and the routing can be time-variable and given in a form of an unsplittable routing or a routing tree.

Determining a max-min fair unsplittable routing as studied in [21] is a special case of P-UNSPLITTABLE-FIND for $c_{st} = c_{rt}$ and $T = 1$, and the NP-hardness results from [21] implies the NP-hardness of P-UNSPLITTABLE-FIND.

Max-min Fair Fractional Routing. Max-min fair fractional routing was first studied in [28]. The algorithm from [28] relies on the property that the total values of a max-min fair flow and max flow are

equal, which does not hold even in simple instances of P-FIXED-FRACTIONAL and P-FRACTIONAL. P-FIXED-FRACTIONAL and P-FRACTIONAL reduce to the problem of [28] for $T = 1$ and $c_{\text{st}} = c_{\text{rt}}$.

Max-min fair fractional routing in energy harvesting networks has been considered in [24]. The distributed algorithm from [24] solves P-FIXED-FRACTIONAL, but only as a heuristic. We provide a combinatorial algorithm that solves P-FIXED-FRACTIONAL optimally in a centralized manner.

A general linear programming framework for max-min fair routing was provided in [31], and extended to the setting of sensor and energy harvesting networks in [8] and [24], respectively. This framework, when applied to P-FRACTIONAL, is highly inefficient. P-FRACTIONAL reduces to [8] for $T = 1$, and to [24] when the rates are constant over time.

4 Max-min Fairness and Lexicographic Maximization

Recall that a rate assignment $\{\lambda_{i,t}\}$, $i \in \{1, \dots, n\}$, $t \in \{1, \dots, T\}$, is max-min fair if no rate $\lambda_{i,t}$ can be increased without either losing feasibility or decreasing some other rate $\lambda_{j,\tau} \leq \lambda_{i,t}$. Closely related to the max-min fairness is the notion of lexicographic maximization. The lexicographic ordering of vectors, with the relational operators denoted by $\stackrel{\text{lex}}{=}$, $\stackrel{\text{lex}}{>}$, and $\stackrel{\text{lex}}{<}$, is defined as follows:

Definition 4.1. Let u and v be two vectors of the same length l , and let u_s and v_s denote the vectors obtained from u and v respectively by sorting their elements in the non-decreasing order. Then: (i) $u \stackrel{\text{lex}}{=} v$ if $u_s = v_s$ element-wise; (ii) $u \stackrel{\text{lex}}{>} v$ if there exists $j \in \{1, 2, \dots, l\}$, such that $u_s(j) > v_s(j)$, and $u_s(1) = v_s(1), \dots, u_s(j-1) = v_s(j-1)$ if $j > 1$; (iii) $u \stackrel{\text{lex}}{<} v$ if not $u \stackrel{\text{lex}}{=} v$ nor $u \stackrel{\text{lex}}{>} v$.

It has been proved in [31] that a max-min fair allocation vector exists on any convex and compact set. The results from [33] state that in a given optimization problem whenever a max-min fair vector exists, it is unique and equal to the lexicographically maximum one.

In the problems P-UNSPLITTABLE-FIND, P-FRACTIONAL, and P-FIXED-FRACTIONAL the feasible region \mathcal{R} is determined by linear constraints (1)-(3), and it is therefore convex. As we are assuming that all the input values $B, e_{i,t}$, and $b_{i,1}$ are finite, it follows that the feasible region is also bounded, and therefore compact. Therefore, for the aforementioned problems, lexicographic maximization produces the max-min fair assignment of the sensing rates $\{\lambda_{i,t}\}$.

Lexicographic maximization can be implemented using the water-filling framework (see, e.g., [3]):

Algorithm 1 WATER-FILLING-FRAMEWORK(G, b, e)

- 1: Set $\lambda_{i,t} = 0 \forall i, t$, and mark all the rates as not fixed.
 - 2: Increase all the rates $\lambda_{i,t}$ that are not fixed by the same maximum amount, subject to the constraints from \mathcal{R} .
 - 3: Fix all the $\lambda_{i,t}$'s that cannot be further increased.
 - 4: If all the rates are fixed, terminate. Else, go to step 2.
-

To solve the problems P-FRACTIONAL, P-FIXED-FRACTIONAL, and P-UNSPLITTABLE-RATES, the challenge is to implement the Steps 2 and 3 efficiently, which we study in the following sections. We will refer to the algorithm that implements Step 2 as MAXIMIZING-THE-RATES, and to the algorithm that implements Step 3 as FIXING-THE-RATES.

Note: A rate $\lambda_{i,t}$ can in general get fixed in any iteration of the WATER-FILLING-FRAMEWORK; there is no rule that relates an iteration k to a node i or a time slot t .

5 Rates in Unsplittable Routing

This section studies P-UNSPLITTABLE-RATES, the problem of rate assignment for an unsplittable routing provided at the input. The analysis applies to any time-invariable or time-variable unsplittable routing or a routing tree.

We assume that the routing over time $t \in \{1, \dots, T\}$ is provided as a set of routing paths $\mathcal{P} = \{p_{i,t}\}$ from a node i to the sink s , for each node $i \in V \setminus \{s\}$. We say that a node j is a descendant of a node i in a time slot t if $i \in p_{j,t}$.⁵

As observed in Section 4, to design an efficient rate assignment algorithm, we need to implement the Steps 2 and 3 of WATER-FILLING-FRAMEWORK efficiently.

Before describing the algorithms in detail, we need to introduce some notation. Let $F_{i,t}^k = 1$ if the rate $\lambda_{i,t}$ is not fixed at the beginning of the k^{th} iteration of WATER-FILLING-FRAMEWORK, $F_{i,t}^k = 0$ otherwise. Initially, $F_{i,t}^1 = 1, \forall i, t$. If a rate $\lambda_{i,t}$ is not fixed, we will say that it is “active”. We will denote by $D_{i,t}^k$ the number of active descendants of the node i in the time slot t , where $D_{i,t}^1 = |\{j : i \in p_{j,t} \setminus \{j\}\}|$. Notice that $D_{i,t}^k = \sum_{j:i \in p_{j,t} \setminus \{j\}} F_{j,t}^k$. Finally, let $\lambda_{i,t}^k$ denote the value of $\lambda_{i,t}$ in the k^{th} iteration of WATER-FILLING-FRAMEWORK, and let $\lambda_{i,t}^0 = 0, \forall i, t$. Under this notation, the rates can be expressed as $\lambda_{i,t}^k = \sum_{l=1}^k F_{i,t}^l \lambda^l$, where λ^l denotes the common amount by which all the active rates get increased in the l^{th} iteration.

5.1 Maximizing the Rates

Maximization of the common rate λ^k in k^{th} iteration of WATER-FILLING-FRAMEWORK can be formulated as follows:

$$\begin{aligned} \max \quad & \lambda^k \\ \text{s.t.} \quad & \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} : \\ & \lambda_{i,t}^k = \lambda_{i,t}^{k-1} + F_{i,t}^k \cdot \lambda^k \\ & f_{i,t}^\Sigma + \lambda_{i,t}^k = \sum_{(i,j) \in E} f_{ij,t} \\ & b_{i,t+1} = \min\{B, b_{i,t} + e_{i,t} - (c_{\text{rt}} f_{i,t}^\Sigma + c_{\text{st}} \lambda_{i,t}^k)\} \\ & b_{i,t} \geq 0, \lambda^k \geq 0, f_{ij,t} \geq 0, \forall (i,j) \in E \end{aligned}$$

As in each slot t every node i sends all the flow it senses over a single path, we can compute the total inflow into a node i as the sum of the flows coming from i 's descendants:

$$\begin{aligned} f_{i,t}^\Sigma &= \sum_{j:i \in p_{j,t} \setminus \{j\}} \sum_{l=1}^k F_{j,t}^l \cdot \lambda^l = \sum_{l=1}^k \lambda^l \sum_{j:i \in p_{j,t} \setminus \{j\}} F_{j,t}^l \\ &= \sum_{l=1}^k D_{i,t}^l \cdot \lambda^l \end{aligned}$$

Denoting the battery levels in the iteration k as $b_{i,t}^k$, the problem can now be written more compactly as:

$$\begin{aligned} \max \quad & \lambda^k \\ \text{s.t.} \quad & \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} : \\ & b_{i,t+1}^k = \min\{B, b_{i,t}^k + e_{i,t} - \sum_{l=1}^k \lambda^l (c_{\text{rt}} D_{i,t}^l + c_{\text{st}} F_{i,t}^l)\} \\ & b_{i,t}^k \geq 0, \lambda^k \geq 0, \end{aligned}$$

where $\forall i \forall k : b_{i,1}^k = b_{i,1}$.

Instead of using all of the λ^l 's from previous iterations in the expression for $b_{i,t+1}^k$, we can define the battery drop in the iteration k , for node i and time slot t as: $\Delta b_{i,t}^k = \sum_{l=1}^k \lambda^l (c_{\text{rt}} D_{i,t}^l + c_{\text{st}} F_{i,t}^l)$ and only keep track of the battery drops from the previous iteration. The intuition is as follows: to determine the battery levels in all the time slots, we only need to know the initial battery level and how much energy

⁵Notice that this is consistent with the definition of a descendant in a routing tree.

$(\Delta b_{i,t})$ is spent per time slot. Setting $\Delta b_{i,t}^0 = 0$, the problem can be written as:

$$\begin{aligned}
& \mathbf{max} \quad \lambda^k \\
& \mathbf{s.t.} \quad \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} : \\
& \quad \Delta b_{i,t}^k = \Delta b_{i,t}^{k-1} + \lambda^k (c_{rt} D_{i,t}^k + c_{st} F_{i,t}^k) \\
& \quad b_{i,t+1}^k = \min\{B, b_{i,t}^k + e_{i,t} - \Delta b_{i,t}^k\} \\
& \quad b_{i,t}^k \geq 0, \lambda^k \geq 0
\end{aligned}$$

Writing the problem for each node independently, we can solve the following subproblem:

$$\mathbf{max} \quad \lambda_i^k \tag{4}$$

$$\mathbf{s.t.} \quad \forall t \in \{1, \dots, T\} :$$

$$\Delta b_{i,t}^k = \Delta b_{i,t}^{k-1} + \lambda_i^k (c_{rt} D_{i,t}^k + c_{st} F_{i,t}^k) \tag{5}$$

$$b_{i,t+1}^k = \min\{B, b_{i,t}^k + e_{i,t} - \Delta b_{i,t}^k\} \tag{6}$$

$$b_{i,t}^k \geq 0, \lambda_i^k \geq 0 \tag{7}$$

for each i with $\sum_{i,t} F_{i,t}^k > 0$, and determine $\lambda^k = \min_i \lambda_i^k$. Notice that we can bound each λ_i^k by the interval $[0, \lambda_{\max,i}^k]$, where $\lambda_{\max,i}^k$ is the rate for which node i spends all its available energy in the first slot τ in which its rate is not fixed:

$$\lambda_{\max,i}^k = \frac{b_{i,\tau}^{k-1} + e_{i,\tau}}{c_{rt} D_{i,\tau}^k + c_{st}}, \quad \tau = \min\{t : F_{i,t}^k = 1\}.$$

The subproblem of determining λ_i^k can now be solved by performing a binary search in the interval $[0, \lambda_{\max,i}^k]$.

Let δ denote the precision of the input variables. Note that however small, δ can usually be expressed as a constant.

Lemma 5.1. MAXIMIZING-THE-RATES in P-UNSPLITTABLE-FIND can be implemented in time

$$O\left(T \sum_i \log(\lambda_{\max,i}^k / \delta)\right) = O\left(nT \log\left(B + \max_{i,t} e_{i,t} / (\delta c_{st})\right)\right).$$

5.2 Fixing the rates

Recall that the elements of the matrix F^k are such that $F_{i,t}^k = 0$ if the rate $\lambda_{i,t}$ is fixed for the iteration k , and $F_{i,t}^k = 1$ otherwise. At the end of iteration $k \geq 1$, let $F^{k+1} = F^k$, and consider the following set of rules for fixing the rates:

- (F1) For all (i, t) such that $b_{i,t+1}^k = 0$ set $F_{i,t}^{k+1} = 0$.
- (F2) For all (i, t) such that $b_{i,t+1}^k = 0$ determine the longest sequence $(i, t), (i, t-1), (i, t-2), \dots, (i, \tau), \tau \geq 1$, with the property that $b_{i,s}^k + e_{i,s} - \Delta b_{i,s}^k \leq B \forall s \in \{t, t-1, \dots, \tau\}$, and set $F_{i,s}^{k+1} = 0 \forall s$.
- (F3) For all (i, t) for which the rules (F1) and (F2) have set $F_{i,t}^{k+1} = 0$, and for all j such that $i \in p_{j,t}$, set $F_{j,t}^{k+1} = 0$.

We will need to prove that these rules are necessary and sufficient for fixing the rates. Here, “necessary” means that no rate that gets fixed at the end of iteration k can get increased in iteration $k+1$ without violating at least one of the constraints. “Sufficient” means that all the rates $\lambda_{i,t}$ with $F_{i,t}^{k+1} = 1$ can be increased by a positive amount in iteration $k+1$ without violating feasibility.

Lemma 5.2. (Necessity) No rate fixed by the rules (F1), (F2) and (F3) can be increased in the next iteration without violating feasibility constraints.

Proof. We will prove the lemma by induction.

The base case. Consider the first iteration and observe the pairs (i, t) for which $F_{i,t}^1 = 0$.

Suppose that $b_{i,t+1}^1 = 0$. The first iteration starts with all the rates being active, so we get from the constraint (6):

$$\begin{aligned} b_{i,t+1}^1 &= \min \left\{ B, b_{i,t}^1 + e_{i,t} - \left(c_{\text{rt}} \sum_{i \in p_{j,t} \setminus \{j\}} \lambda_{j,t}^1 + c_{\text{st}} \lambda_{i,t}^1 \right) \right\} \\ &= b_{i,t}^1 + e_{i,t} - \left(c_{\text{rt}} \sum_{i \in p_{j,t} \setminus \{j\}} \lambda_{j,t}^1 + c_{\text{st}} \lambda_{i,t}^1 \right) = 0 \end{aligned} \quad (8)$$

as $B > 0$, where the third line comes from all the rates being equal in the first iteration and the fact that all the i 's descendants must send their flow through i .

As every iteration only increases the rates, if we allow $\lambda_{i,t}$ to be increased in the next iteration, then (from (8)) we would get $b_{i,t+1} < 0$, which is a contradiction. Alternatively, if we increase $\lambda_{i,t}^1$ at the expense of decreasing some $\lambda_{j,t}^1$, $i \in p_{j,t} \setminus \{j\}$, to keep $b_{i,t+1} \geq 0$, then the solution is not max-min fair, as $\lambda_{j,t}^1 = \lambda_{i,t}^1 = \lambda^1$. This proves the necessity of the rule (F1). By the same observation, if we increase the rate $\lambda_{j,t}^1$ of any of the node i 's descendants j at time t , we will necessarily get $b_{i,t+1} < 0$ (or would need to sacrifice the max-min fairness). This proves the rule (F3) for all the descendants of node i , such that $F_{i,t}^2$ is set to 0 by the rule (F1).

Now let $(i, t), (i, t-1), (i, t-2), \dots, (i, \tau), \tau \geq 1$, be the longest sequence with the property that: $b_{i,t} = 0$ and $b_{i,s}^1 + e_{i,s} - \Delta b_{i,s}^1 \leq B \forall s \in \{t, t-1, \dots, \tau\}$. Observe that when this is the case, we have:

$$\begin{aligned} \forall s \in \{\tau, \tau+1, \dots, t-2, t-1\} : \\ b_{i,s+1}^1 &= \min \{ B, b_{i,s}^1 + e_{i,s} - \Delta b_{i,s}^1 \} = b_{i,s}^1 + e_{i,s} - \Delta b_{i,s}^1 \\ &= b_{i,s}^1 + e_{i,s} - \left(c_{\text{rt}} \sum_{j: i \in p_{j,t} \setminus \{j\}} \lambda_{j,s}^1 + c_{\text{st}} \lambda_{i,s}^1 \right) \end{aligned}$$

This gives a recursive relation, so $b_{i,t+1}$ can also be written as:

$$b_{i,t+1}^1 = b_{i,\tau}^1 + \sum_{s=\tau}^t e_{i,s} - c_{\text{rt}} \sum_{s=\tau}^t \sum_{j: i \in p_{j,t} \setminus \{j\}} \lambda_{j,s}^1 - c_{\text{st}} \sum_{s=\tau}^t \lambda_{i,s}^1.$$

If we increase $\lambda_{i,s}$ or $\lambda_{j,s}$, for any j, s such that $i \in p_{j,s} \setminus \{j\}$, $s \in \{\tau, \tau+1, \dots, t-2, t-1\}$, then either $b_{i,t+1}$ becomes negative, or we sacrifice the max-min fairness, as all the rates are equal to λ^1 in the first iteration. This proves both the rule (F2) and completes the proof for the necessity of the rule (F3).

The inductive step. Suppose that all the rules are necessary for the iterations $1, 2, \dots, k-1$, and consider the iteration k .

Observe that:

- (o1) $\lambda_{j,t} \leq \lambda_{i,t}, \forall j : i \in p_{j,t}$, as all the rates, until they are fixed, get increased by the same amount in each iteration, and once a rate gets fixed for some (i, t) , by the rule (F3), it gets fixed for all the node i 's descendants in the same time slot. Notice that the inequality is strict only if $\lambda_{j,t}$ got fixed before $\lambda_{i,t}$; otherwise these two rates get fixed to the same value.
- (o2) Once fixed, a rate never becomes active again.
- (o3) If a rate $\lambda_{i,t}$ gets fixed in iteration k , then $\lambda_{i,t} = \lambda_{i,t}^k = \sum_{p=1}^k \lambda^p = \lambda_{i,t}^l, \forall l > k$.

Suppose that $b_{i,t+1}^k = 0$ for some $i \in \{1, \dots, n\}, t \in \{1, \dots, T\}$. If $F_{i,t}^k = 0$, then by the inductive hypothesis $\lambda_{i,t}$ cannot be further increased in any of the iterations $k, k+1, \dots$. Assume $F_{i,t}^k = 1$. Then:

$$\begin{aligned} b_{i,t+1}^k &= \min \left\{ B, b_{i,t}^k + e_{i,t} - \left(c_{\text{rt}} \sum_{j: i \in p_{j,t} \setminus \{j\}} \lambda_{j,t}^k + c_{\text{st}} \lambda_{i,t}^k \right) \right\} \\ &= b_{i,t}^k + e_{i,t} - \left(c_{\text{rt}} \sum_{j: i \in p_{j,t} \setminus \{j\}} \lambda_{j,t}^k + c_{\text{st}} \lambda_{i,t}^k \right) = 0 \end{aligned}$$

By the observation (o1), $\lambda_{j,t}^k \leq \lambda_{i,t}^k$, $\forall j$ such that $i \in p_{j,t} \setminus \{j\}$, where the inequality holds with equality if $F_{j,t}^k = 0$. Therefore, if we increase $\lambda_{i,t}$ in some of the future iterations, either $b_{i,t+1} < 0$, or we need to decrease some $\lambda_{j,t} \leq \lambda_{i,t}$, violating the max-min fairness condition. This proves the necessity of the rule (F1). For the rule (F3), as for all (j,t) with $F_{j,t}^k = 1$, $i \in p_{j,t} \setminus \{j\}$, we have $\lambda_{j,t} = \lambda_{i,t}$, none of the i 's descendants can further increase its rate in the slot t .

Now for (i,t) such that $b_{i,t+1}^k = 0$, let $(i,t), (i,t-1), (i,t-2), \dots, (i,\tau), \tau \geq 1$, be the longest sequence with the property that: $b_{i,s}^k + e_{i,s} - \Delta b_{i,s}^k \leq B \forall s \in \{t, t-1, \dots, \tau\}$. Similarly as for the base case:

$$\begin{aligned} \forall s \in \{\tau, \tau+1, \dots, t-2, t-1\} : \\ b_{i,s+1}^k &= \min \{B, b_{i,s}^k + e_{i,s} - \Delta b_{i,s}^k\} \\ &= b_{i,s}^k + e_{i,s} - \left(c_{\text{rt}} \sum_{j: i \in p_{j,t} \setminus \{j\}} \lambda_{j,s}^k + c_{\text{st}} \lambda_{i,s}^k \right) \end{aligned}$$

and we get that:

$$b_{i,t+1}^k = b_{i,\tau}^k + \sum_{s=\tau}^t e_{i,s} - c_{\text{rt}} \sum_{s=\tau}^t \sum_{j: i \in p_{j,t} \setminus \{j\}} \lambda_{j,s}^k - c_{\text{st}} \sum_{s=\tau}^t \lambda_{i,s}^k. \quad (9)$$

If any of the rates appearing in (9), was fixed in some previous iteration, then it cannot be further increased by the inductive hypothesis. By the observation (o1), all the rates that are active are equal, and all the rates that are fixed are strictly lower than the active rates. Therefore, by increasing any of the active rates from (9), we either violate battery nonnegativity constraint or the max-min fairness condition. Therefore, rule (F2) holds, and rule (F3) holds for all the descendants of nodes whose rates got fixed by the rule (F2), in the corresponding time slots. \square

Lemma 5.3. (*Sufficiency*) If $F_{i,t}^{k+1} = 1$, then $\lambda_{i,t}$ can be further increased by a positive amount in the iteration $k+1$, $\forall i \in \{1, \dots, n\}$, $\forall t \in \{1, \dots, T\}$.

Proof. Suppose that $F_{i,t}^{k+1} = 1$. Notice that by increasing $\lambda_{i,t}$ by some $\Delta \lambda_{i,t}$ node i spends additional $\Delta b_{i,t} = c_{\text{st}} \Delta \lambda_{i,t}$ energy *only in the time slot t* . As $F_{i,t}^{k+1} = 1$, by the rules (F1) and (F2), either $b_{i,t'} > 0 \forall t' > t$, or there is a time slot $t'' > t$ such that $b_{i,t''}^k + e_{i,t''} - \Delta b_{i,t''}^k > B$ and $t'' < t'''$, where $t''' = \arg \min \{\tau > t : b_{i,\tau} = 0\}$.

If $b_{i,t'} > 0 \forall t' > t$, then the node i can spend $\Delta b_{i,t} = \min_{t+1 \leq t' \leq T+1} b_{i,t'}^k$ energy, and keep $b_{i,t'} \geq 0, \forall t'$, which follows from the battery evolution equation (6).

If there is a slot $t''' > t$ in which $b_{i,t'''}^k = 0$, then let t'' be the minimum time slot between t and t''' , such that $b_{i,t''}^k + e_{i,t''} - \Delta b_{i,t''}^k > B$. Decreasing the battery level at t'' by $(b_{i,t''}^k + e_{i,t''} - \Delta b_{i,t''}^k) - B$ does not influence any other battery levels, as in either case $b_{i,t''+1} = B$. As all the battery levels are positive in all the time slots between t and t'' , i can spend at least $\min\{(b_{i,t''}^k + e_{i,t''} - \Delta b_{i,t''}^k) - B, \min_{t+1 \leq t' \leq t''} b_{i,t'}^k\}$ energy at the time t and have $b_{i,t'} \geq 0 \forall t'$.

By the rule (F3), $\forall j$ such that $j \in p_{i,t}$ we have that $b_{j,t} > 0$, and, furthermore, if $\exists t''' > t$ with $b_{j,t'''} = 0$ then $\exists t'' \in \{t, t'''\}$ such that $b_{i,t''}^k + e_{i,t''} - \Delta b_{i,t''}^k > B$. By the same observations as for the node i , each $j \in p_{i,t}$ can spend some extra energy $\Delta b_{j,t} > 0$ in the time slot t and keep all the battery levels nonnegative. In other words, there is a directed path from the node i to the sink on which every node can spend some extra energy in time slot t and keep its battery levels nonnegative. Therefore, if we keep all other rates fixed, the rate $\lambda_{i,t}$ can be increased by $\Delta \lambda_{i,t} = \min\{\Delta b_{i,t}/c_{\text{st}}, \min_{j \in p_{i,t}} \Delta b_{j,t}/c_{\text{rt}}\} > 0$.

As each active rate $\lambda_{i,t}$ can (alone) get increased in the iteration $k+1$ by some $\Delta \lambda_{i,t} > 0$, it follows that all the active rates can be increased simultaneously by at least $\min_{i,t} \Delta \lambda_{i,t} / (T(c_{\text{st}} + nc_{\text{rt}})) > 0$. \square

Theorem 5.4. Fixing rules (F1), (F2) and (F3) provide necessary and sufficient conditions for fixing the sensing rates in WATER-FILLING-FRAMEWORK.

Proof. Follows directly from Lemmas 5.2 and 5.3. \square

Lemma 5.5. The total running time of FIXING-THE-RATES in P-UNSPLITTABLE-FIND is $O(mT)$.

Proof. Rules (F1) and (F2) can be implemented for each node independently in time $O(T)$ by examining the battery levels from slot $T + 1$ to slot 2.

For the rule (F3), in each time slot $t \in \{1, \dots, T\}$ enqueue all the nodes i whose rates got fixed in time slot t by either of the rules (F1), (F2) and perform a breadth-first search. Fix the rates of all the nodes discovered by a breadth-first search. This gives $O(m)$ time per slot, for a total time of $O(mT)$. Combining with the time for rules (F1) and (F2), the result follows. \square

Combining Lemmas 5.1 and 5.5, we can compute the total running time of WATER-FILLING-Framework for P-UNSPLITTABLE-FIND, as stated in the following lemma.

Lemma 5.6. WATER-FILLING-Framework with Steps 2 MAXIMIZING-THE-RATES and 3 FIXING-THE-RATES implemented as described in Section 5 runs in time:

$$O(nT(mT + nT \log(B + \max_{i,t} e_{i,t}/(\delta c_{st}))))).$$

Proof. To bound the running time of the overall algorithm that performs lexicographic maximization, we need to first bound the number of iterations that the algorithm performs. As in each iteration at least one sensing rate $\lambda_{i,t}$, $i \in \{1, \dots, n\}$, $t \in \{1, \dots, T\}$, gets fixed, and once fixed remains fixed, the total number of iterations is $O(nT)$. The running time of each iteration is determined by the running times of the steps 2 (MAXIMIZING-THE-RATES) and 3 (FIXING-THE-RATES) of the WATER-FILLING-Framework. MAXIMIZING-THE-RATES runs in $O\left(nT \log\left(\frac{B + \max_{i,t} e_{i,t}}{\delta c_{st}}\right)\right)$ (Lemma 5.1), whereas FIXING-THE-RATES runs in $O(mT)$ time (Lemma 5.5). Therefore, the total running time is: $O(nmT^2 + n^2T^2 \log(B + \max_{i,t} e_{i,t}/(\delta c_{st})))$. \square

6 Fractional Routing

The feasible region \mathcal{R} for the rates and flows in P-Fractional can be described by the following constraints:

$$\begin{aligned} \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} : \\ f_{i,t}^\Sigma + \lambda_{i,t} &= \sum_{(i,j) \in E} f_{ij,t} \\ b_{i,t+1} &= \min\{B, b_{i,t} + e_{i,t} - (c_{rt} f_{i,t}^\Sigma + c_{st} \lambda_{i,t})\} \\ b_{i,t} &\geq 0, \lambda_{i,t} \geq 0, f_{ij,t} \geq 0, \forall (i,j) \in E, \end{aligned}$$

where $f_{i,t}^\Sigma \equiv \sum_{(j,i) \in E} f_{ji,t}$.

Observe that we can avoid computing the values of battery levels $b_{i,t+1}$, and instead explicitly write the non-negativity constraints for each of the terms inside the min. Reordering the terms, we get the following formulation:

$$\begin{aligned} \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} : \\ f_{i,t}^\Sigma + \lambda_{i,t} &= \sum_{(i,j) \in E} f_{ij,t} \end{aligned} \tag{10}$$

$$\sum_{\tau=1}^t (c_{rt} f_{i,\tau}^\Sigma + c_{st} \lambda_{i,\tau}) \leq b_{i,1} + \sum_{\tau=1}^t e_{i,\tau} \tag{11}$$

$$\sum_{\tau=s}^t (c_{rt} f_{i,\tau}^\Sigma + c_{st} \lambda_{i,\tau}) \leq B + \sum_{\tau=s}^t e_{i,\tau}, \quad 2 \leq s \leq t \tag{12}$$

$$\lambda_{i,t} \geq 0, f_{ij,t} \geq 0, \forall (i,j) \in E \tag{13}$$

In the k^{th} iteration of WATER-FILLING-Framework we have that $\lambda_{i,t}^k = \lambda_{i,t}^{k-1} + F_{i,t}^k \cdot \lambda^k = \sum_{l=1}^k F_{i,t}^l \cdot \lambda^l$, where $\lambda_{i,t}^0 = 0$. Let:

$$u_{i,t}^b = b_{i,1} + \sum_{\tau=1}^t (e_{i,\tau} - c_{st} \lambda_{i,\tau}^{k-1}), \quad u_{i,t,s}^B = B + \sum_{\tau=s}^t (e_{i,\tau} - c_{st} \lambda_{i,\tau}^{k-1})$$

Since in the iteration k all $\lambda_{i,t}^{k-1}$'s are constants, the rate maximization subproblem can be written as:

$$\mathbf{max} \quad \lambda^k \tag{14}$$

$$\mathbf{s.t.} \quad \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} :$$

$$-f_{i,t}^\Sigma - F_{i,t}^k \cdot \lambda^k + \sum_{(i,j) \in E} f_{ij,t} = \lambda_{i,t}^{k-1} \tag{15}$$

$$\sum_{\tau=1}^t (c_{rt} f_{i,\tau}^\Sigma + F_{i,\tau}^k \cdot c_{st} \lambda^k) \leq u_{i,t}^b \tag{16}$$

$$\sum_{\tau=s}^t (c_{rt} f_{i,\tau}^\Sigma + F_{i,\tau}^k \cdot c_{st} \lambda^k) \leq u_{i,t,s}^B, \quad 2 \leq s \leq t \tag{17}$$

$$\lambda^k \geq 0, f_{ij,t} \geq 0, \forall (i,j) \in E \tag{18}$$

Notice that in this formulation all the variables are on the left-hand side of the constraints, whereas all the right-hand sides are constant.

6.1 Relation to Multi-commodity Flow

Let $T = 2$, and observe the constraints in (10)–(13). We claim that verifying whether any set of sensing rates $\lambda_{i,t}$ is feasible is at least as hard as solving a 2-commodity feasible flow problem with capacitated nodes and a single sink. To prove the claim, we first rewrite the constraints in (10)–(13) as:

$$\begin{aligned} \sum_{(j,i) \in E} f_{ji,t} + \lambda_{i,t} &= \sum_{(i,j) \in E} f_{ij,t}, & t \in \{1, 2\} \\ c_{rt} \sum_{(j,i) \in E} f_{ji,1} &\leq b_{i,1} + e_{i,1} - c_{st} \lambda_{i,1} \\ c_{rt} \sum_{\tau=1}^2 \sum_{(j,i) \in E} f_{ji,\tau} &\leq b_{i,1} + \sum_{\tau=1}^2 (e_{i,\tau} - c_{st} \lambda_{i,\tau}) \\ c_{rt} \sum_{(j,i) \in E} f_{ji,2} &\leq B + e_{i,2} - c_{st} \lambda_{i,2} \\ \lambda_{i,t} \geq 0, f_{ij,t} \geq 0, &\forall i \in \{1, \dots, n\}, (i,j) \in E, t \in \{1, 2\} \end{aligned}$$

Suppose that we are given any 2-commodity flow problem with capacitated nodes, and let:

- $\lambda_{i,t}$ denote the supply of commodity t at node i ;
- $u_{i,t}$ denote the per-commodity capacity constraint at node i for commodity t ;
- u_i denote the bundle capacity constraint at node i .

Choose values of $c_s, c_{rt}, B, b_{i,1}, b_{i,2}, e_{i,1}, e_{i,2}$ so that the following equalities are satisfied:

$$\begin{aligned} u_{i,1} &= \frac{1}{c_{rt}} \cdot (b_{i,1} + e_{i,1} - c_{st} \lambda_{i,1}) \\ u_{i,2} &= \frac{1}{c_{rt}} \cdot (B + e_{i,2} - c_{st} \lambda_{i,2}) \\ u_i &= \frac{1}{c_{rt}} (b_{i,1} + \sum_{\tau=1}^2 (e_{i,\tau} - c_{st} \lambda_{i,\tau})) \end{aligned}$$

Then feasibility of the given 2-commodity flow problem is equivalent to the feasibility of (10)–(13). Therefore, any 2-commodity feasible flow problem can be stated as an equivalent problem of verifying feasibility of sensing rates $\lambda_{i,t}$ in an energy harvesting network for $T = 2$.

For $T > 2$, (11) and (12) are general packing constraints. If a flow graph G_t in time slot t is observed as a flow of a commodity indexed by t , then for each node i the constraints (11) and (12) define capacity constraints for every sequence of consecutive commodities $s, s+1, \dots, t, 1 \leq s \leq t \leq T$.

Therefore, to our current knowledge, it is unlikely that the general rate assignment problem can be solved exactly in polynomial time without the use of linear programming, as there have not been any combinatorial algorithms that solve feasible 2-commodity flow exactly.

6.2 Fractional Packing Approach

The fractional packing problem is defined as follows [30]:

PACKING: *Given a convex set P for which $Ax \geq 0 \forall x \in P$, is there a vector x such that $Ax \leq b$?* Here, A is a $p \times q$ matrix, and x is a q -length vector.

A given vector x is an ϵ -approximate solution to the PACKING problem if $x \in P$ and $Ax \leq (1 + \epsilon)b$. Alternatively, scaling all the constraints by $\frac{1}{1+\epsilon}$, we obtain a solution $x' = \frac{1}{1+\epsilon}x \in (\frac{1}{1+\epsilon}x_{\text{OPT}}, x_{\text{OPT}}] \subset ((1 - \epsilon)x_{\text{OPT}}, x_{\text{OPT}}]$, for $\epsilon < 1$, where x_{OPT} is an optimal solution to the packing problem. The algorithm in [30] either provides an ϵ -approximate solution to the PACKING problem, or it proves that no such solution exists. It's running time depends on:

- The running time required to solve $\min\{cx : x \in P\}$, where $c = y^T A$, y is a given p -length vector, and $(\cdot)^T$ denotes the transpose of a vector.
- The width of P relative to $Ax \leq b$, which is defined by $\rho = \max_i \max_{x \in P} \frac{a_i x}{b_i}$, where a_i is the i^{th} row of A , and b_i is the i^{th} element of b .

For a given error parameter $\epsilon > 0$, a feasible solution to the problem $\min\{\beta : Ax \leq \beta b, x \in P\}$, its dual solution y , and $C_P(y) = \min\{cx : c = y^T A, x \in P\}$, [30] defines the following relaxed optimality conditions:

$$(1 - \epsilon)\beta y^T b \leq y^T Ax \tag{P1}$$

$$y^T Ax - C_P(y) \leq \epsilon(y^T Ax + \beta y^T b) \tag{P2}$$

The packing algorithm [30] is implemented through subsequent calls to the procedure IMPROVE-PACKING:

Algorithm 2 IMPROVE-PACKING(x, ϵ) [30]

- 1: Initialize $\beta_0 = \max_i a_i x / b_i$; $\alpha = 4\beta_0^{-1}\epsilon^{-1} \ln(2p\epsilon^{-1})$; $\sigma = \epsilon / (4\alpha\rho)$.
 - 2: **while** $\max_i a_i x / b_i \geq \beta_0/2$ and x, y do not satisfy (P2) **do**
 - 3: For each $i = 1, 2, \dots, p$: set $y_i = (1/b_i)e^{\alpha a_i x / b_i}$.
 - 4: Find a min-cost point $\tilde{x} \in P$ for costs $c = y^T A$.
 - 5: Update $x = (1 - \sigma)x + \sigma\tilde{x}$.
 - 6: **return** x .
-

The running time of the ϵ -approximation algorithm provided in [30], for $\epsilon \in (0, 1]$, equals $O(\epsilon^{-2}\rho \log(m\epsilon^{-1}))$ multiplied by the time needed to solve $\min\{cx : c = y^T A, x \in P\}$ and compute Ax (Theorem 2.5 in [30]).

6.2.1 Maximizing the Rates as Fractional Packing

We demonstrated at the beginning of this section that for the k^{th} iteration MAXIMIZE-THE-RATES can be stated as (14)-(18). Observe the constraints (16) and (17). Since λ^k , $f_{ij,t}$ and all the right-hand sides in (16) and (17) are nonnegative, (16) and (17) imply the following inequalities:

$$\begin{aligned} \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} : \\ F_{i,\theta}^k \cdot c_{\text{st}} \lambda^k &\leq u_{i,t}^b, & 1 \leq \theta \leq t \\ F_{i,\theta}^k \cdot c_{\text{st}} \lambda^k &\leq u_{i,t,s}^B, & 2 \leq s \leq t, s \leq \theta \leq t \\ c_{\text{rt}} \sum_{(j,i) \in E} f_{ji,\theta} &\leq u_{i,t}^b - c_{\text{st}} \sum_{\tau=1}^t F_{i,\tau}^k \lambda^k, & 1 \leq \theta \leq t \\ c_{\text{rt}} \sum_{(j,i) \in E} f_{ji,\theta} &\leq u_{i,t,s}^B - c_{\text{st}} \sum_{\tau=s}^t F_{i,\tau}^k \lambda^k, & 2 \leq s \leq t, s \leq \theta \leq t \end{aligned}$$

Therefore, we can yield an upper bound λ_{max}^k for λ^k :

$$\begin{aligned} \lambda^k &\leq \lambda_{\text{max}}^k \equiv \\ \frac{1}{c_{\text{st}}} \min_{i,t,s \geq 2} \{ &u_{i,t}^b : \sum_{\tau=1}^t F_{i,\tau}^k > 0, u_{i,t,s}^B : \sum_{\tau=s}^t F_{i,\tau}^k > 0 \} \end{aligned} \tag{19}$$

For a fixed λ^k , the flow entering a node i at time slot t can be bounded as:

$$\begin{aligned} \sum_{(j,i) \in E} f_{ji,t} &\leq u_{i,t} \equiv \\ \frac{1}{c_{rt}} \min_{\substack{i, t_1 \geq t \\ s \geq 2}} \{ &u_{i,t_1}^b - c_{st} \sum_{\tau=1}^{t_1} F_{i,\tau}^k \lambda^k, u_{i,t,s}^B - c_{st} \sum_{\tau=s}^{t_1} F_{i,\tau}^k \lambda^k \} \end{aligned} \quad (20)$$

We choose to keep only the flows $f_{ij,t}$ as variables in the PACKING problem. Given a $\lambda^k \in [0, \lambda_{\max}^k]$, we define the convex set P^6 via the following set of constraints:

$$\begin{aligned} \forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} : \\ - \sum_{(j,i) \in E} f_{ji,t} + \sum_{(i,j) \in E} f_{ij,t} &= \lambda_{i,t}^{k-1} + F_{i,t}^k \cdot \lambda^k \end{aligned} \quad (21)$$

$$\sum_{(j,i) \in E} f_{ji,t} \leq u_{i,t} \quad (22)$$

$$f_{ij,t} \geq 0, \quad \forall (i,j) \in E \quad (23)$$

Proposition 6.1. *For P described by (21) – (23) and a given vector y , problem $\min\{cf : c = y^T Af, f \in P\}$ reduces to T min-cost flow problems.*

Proof. Constraint (21) is a standard flow balance constraint at a node i in a time slot t , whereas constraint (22) corresponds to a node capacity constraint at the time t , given by (20). As there is no interdependence of flows over time slots, we get that the problem can be decomposed into subproblems corresponding to individual time slots. Therefore, to solve the problem $\min\{cf : c = y^T Af, f \in P\}$ for a given vector y , it suffices to solve T min-cost flow problems, one for each time slot $t \in \{1, 2, \dots, T\}$. \square

The remaining packing constraints of the form $Ax \leq b$ are given by (16) and (17), where $x \equiv f$.

Proposition 6.2. $Ax \geq 0 \forall f \in P$.

Proof. As $f_{ij,t} \geq 0 \forall (i,j) \in E, t \in \{1, \dots, T\}$, and all the coefficients multiplying $f_{ij,t}$'s in (16) and (17) are nonnegative, the result follows immediately. \square

Lemma 6.3. *One iteration of IMPROVE-PACKING for P-FRACTIONAL can be implemented in time*

$$O(nT^2 + T \cdot MCF(n, m)),$$

where $MCF(n, m)$ denotes the running time of a min-cost flow algorithm on a graph with n nodes and m edges.

Proof. Since the flows over edges appear in the packing constraints only as the sum-terms of the total incoming flow of a node i in a time slot t , we can use the total incoming flow $f_{i,t}^\Sigma = \sum_{(j,i) \in E} f_{ji,t}$ for each (i, t) as variables. Reordering the terms, the packing constraints can be stated as:

$$\sum_{\tau=1}^t f_{i,\tau}^\Sigma \leq \frac{1}{c_{rt}} (u_{i,t}^b - c_{st} \sum_{\tau=1}^t F_{i,\tau}^k \lambda^k), \quad 1 \leq t \leq T \quad (24)$$

$$\sum_{\tau=s}^t f_{i,\tau}^\Sigma \leq \frac{1}{c_{rt}} (u_{i,t,s}^B - c_{st} \sum_{\tau=s}^t F_{i,\tau}^k \lambda^k), \quad 2 \leq s \leq t, \quad 2 \leq t \leq T \quad (25)$$

With this formulation on hand, the matrix A of the packing constraints $Af^\Sigma \leq b$ is a 0 – 1 matrix that can be decomposed into blocks of triangular matrices. To see this, first notice that for each node i constraints given by (24) correspond to a lower-triangular 0-1 matrix of size T . Each sequence of constraints of type (25) for fixed i and fixed $s \in \{2, \dots, T\}$, and $t \in \{s, s+1, \dots, T\}$ corresponds to a lower-triangular 0-1 matrix of size $T - s + 1$. This special structure of the packing constraints matrix allows an efficient computation of the dual vector y and the corresponding cost vector c . Moreover, as constraints (24, 25) can be decomposed

⁶ P is determined by linear equalities and inequalities, which implies that it is convex.

into independent blocks of constraints of the type $A_i f_i^\Sigma \leq b_i$ for nodes $i \in \{1, \dots, n\}$, the dual vector y and the corresponding cost vector c can be decomposed into vectors y_i, c_i for $i \in \{1, \dots, n\}$. Cost $c_{i,t}$ can be interpreted as the cost of sending 1 unit of flow through node i in time slot t .

Observe the block of constraints $A_i f_i^\Sigma \leq b_i$ corresponding to the node i . The structure of A_i is as follows:

$$\begin{aligned}
& T \begin{Bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{Bmatrix} \\
& T-1 \begin{Bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & 1 & \cdots & 1 & 1 \end{Bmatrix} \\
& \vdots \\
& 2 \begin{Bmatrix} 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{Bmatrix} \\
& 1 \begin{Bmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \end{Bmatrix}
\end{aligned}$$

As A_i can be decomposed into blocks of triangular matrices, each $y_{i,j}$ in the IMPROVE-PACKING procedure can be computed in constant time, yielding $O\left(\frac{T(T-1)}{2}\right) = O(T^2)$ time for computing y_i . This special structure of A_i also allows a fast computation of the cost vector c_i . Observe that each $c_{i,t}, t \in \{1, \dots, T\}$ can be computed by summing $O(T)$ terms. For example, $c_{i,1} = \sum_{j=1}^T y_{i,j}$, $c_{i,2} = c_{i,1} - y_{i,1} + \sum_{j=T+1}^{2T-1} y_{i,j}$, $c_{i,3} = c_{i,2} - y_{i,2} - y_{i,T+1} + \sum_{j=2T}^{3T-2} y_{i,j}$, etc. Therefore, computing the costs for node i takes $O(T^2)$ time. This further implies that one iteration of IMPROVE-PACKING takes $O(nT^2 + T \cdot MCF(n, m))$ time, where $MCF(n, m)$ denotes the running time of a min-cost flow algorithm on a graph with n nodes and m edges. \square

Lemma 6.4. *Width ρ of P relative to the packing constraints (16) and (17) is $O(T)$.*

Proof. As $u_{i,t}$ is determined by the tightest constraint in which $\sum_{(j,i) \in E} f_{ji,t} \equiv f_{i,t}^\Sigma$ appears, we have that in every constraint given by (24, 25):

$$\begin{aligned}
f_{i,\theta}^\Sigma &\leq \frac{1}{c_{\text{rt}}} (u_{i,t}^b - c_{\text{st}} \sum_{\tau=1}^t F_{i,\tau}^k \lambda^k), & 1 \leq \theta \leq t \\
f_{i,\theta}^\Sigma &\leq \frac{1}{c_{\text{rt}}} (u_{i,t,s}^B - c_{\text{st}} \sum_{\tau=s}^t F_{i,\tau}^k \lambda^k), & 2 \leq s \leq t, s \leq \theta \leq t
\end{aligned}$$

As the sum of $f_{i,j,\theta}^\Sigma$ over θ in any constraint from (24, 25) can include at most T terms, it follows that $\rho \leq \frac{T \cdot b_i}{b_i} = T$. \square

Lemma 6.5. *MAXIMIZING-THE-RATES that uses packing algorithm from [30] can be implemented in time: $\tilde{O}(T^2 \epsilon^{-2} \cdot (nT + MCF(n, m)))$, where \tilde{O} -notation ignores poly-log terms.*

Proof. We have from (19) that $\lambda^k \in [0, \lambda_{\max}^k]$, therefore, we can perform a binary search to find the maximum λ^k for which both $\min\{y^T A f | f \in P\}$ is feasible and PACKING outputs an ϵ -approximate solution. Multiplying the running time of the binary search by the running time of the packing algorithm [30], the total running time becomes:

$$\begin{aligned} & O\left(\log\left(\frac{\lambda_{\max}^k}{\delta}\right) \epsilon^{-2} \rho \log(m\epsilon^{-1}) (nT^2 + T \cdot MCF(n, m))\right) \\ &= \tilde{O}\left(\frac{T^2}{\epsilon^2} \cdot (nT + MCF(n, m))\right). \end{aligned}$$

□

6.2.2 Fixing the Rates

As MAXIMIZING-THE-RATES described in previous subsection outputs an ϵ -approximate solution in each iteration, the objective of the algorithm is not to output a max-min fair solution anymore, but an ϵ -approximation. We consider the following notion of approximation, as in [21]:

Definition 6.6. *For a problem of lexicographic maximization, say that a feasible solution given as a vector v is an element-wise ϵ -approximate solution, if for vectors v and v_{OPT} sorted in nondecreasing order $v \geq (1 - \epsilon)v_{OPT}$ component-wise, where v_{OPT} is an optimal solution to the given lexicographic maximization problem.*

Let Δ be the smallest real number that can be represented in a computer, and consider the algorithm that implements FIXING-THE-RATES as stated below.

Algorithm 3 FIXING-THE-RATES

- 1: Solve the following linear program:
 - 2: **max** $\sum_{i=1}^n F_{i,t}^k \lambda_{i,t}^k$
 - 3: **s.t.** $\forall i \in \{1, \dots, n\}, t \in \{1, \dots, T\} :$
 - 4: $\lambda_{i,t}^k \geq \lambda_{i,t}^{k-1} + F_{i,t}^k \cdot \lambda^k$
 - 5: $\lambda_{i,t}^k \leq \lambda_{i,t}^{k-1} + F_{i,t}^k \cdot (\epsilon \lambda_{i,t}^{k-1} + (1 + \epsilon)\lambda^k + \Delta)$
 - 6: $f_{i,t}^\Sigma + \lambda_{i,t}^k = \sum_{(i,j) \in E} f_{ij,t}$
 - 7: $b_{i,t+1} = \min \left\{ B, b_{i,t} + e_{i,t} - \left(c_{rt} f_{i,t}^\Sigma + c_{st} \lambda_{i,t}^k \right) \right\}$
 - 8: $b_{i,t} \geq 0, \quad \lambda_{i,t}^k \geq 0, \quad f_{ij,t} \geq 0$
 - 9: Let $F_{i,t}^{k+1} = F_{i,t}^k, \forall i, t.$
 - 10: If $\lambda_{i,t}^k < (1 + \epsilon)(\lambda_{i,t}^{k-1} + F_{i,t}^k \cdot \lambda^k) + \Delta$, set $F_{i,t}^{k+1} = 0.$
-

Assume that FIXING-THE-RATES does not change any of the rates, but only determines what rates should be fixed in the next iteration, i.e., it only makes (global) changes to $F_{i,t}^{k+1}$. Then:

Lemma 6.7. *If the Steps 2 and 3 in the WATER-FILLING-FRAMEWORK are implemented as MAXIMIZING-THE-RATES and FIXING-THE-RATES from this section, then the solution output by the algorithm is an element-wise ϵ -approximate solution to the lexicographic maximization of $\lambda_{i,t} \in \mathcal{R}$.*

Proof. The proof is by induction.

The base case. Observe the first iteration of the algorithm. After rate maximization, $\forall i, t : \lambda_{i,t} = \lambda^1 \geq \frac{1}{1 + \epsilon} \lambda_{OPT}^1$ and $F_{i,t}^1 = 1.$

Observe that in the output of the linear program of FIXING-THE-RATES, all the rates must belong to the interval $[\lambda^1, (1 + \epsilon)\lambda^1 + \Delta]$. Choose any (i, t) with $\lambda_{i,t}^1 < (1 + \epsilon)(\lambda_{i,t}^{k-1} + F_{i,t}^1 \cdot \lambda^1) + \Delta = (1 + \epsilon)\lambda^1 + \Delta$. There must be at least one such rate, otherwise the rate maximization did not return an ϵ -approximate solution. As $\sum_{i=1}^n F_{i,t}^1 \lambda_{i,t}^1 = \sum_{i=1}^n \lambda_{i,t}^1$ is maximum, if $\lambda_{i,t}^1$ is increased, then at least one other rate needs to be decreased to maintain feasibility. To get a lexicographically greater solution $\lambda_{i,t}^1$ can only be increased by lowering the rates with the value greater than $\lambda_{i,t}^1$. Denote by $S_{i,t}^1$ the set of all the rates $\lambda_{j,\tau}^1$ such that $\lambda_{j,\tau}^1 > \lambda_{i,t}^1$. In the lexicographically maximum solution, the highest value to which $\lambda_{i,t}^1$ can be increased is

at most $\frac{1}{|S_{i,t}^1|} \left(\lambda_{i,t}^1 + \sum_{\lambda_{j,\tau} \in S_{i,t}^1} \lambda_{j,\tau}^1 \right) < (1 + \epsilon)\lambda^1 + \Delta$, which implies $\lambda_{i,t,\max} \leq (1 + \epsilon)\lambda^1$. Therefore, if $\lambda_{i,t}$ is fixed to the value of λ^1 , it is guaranteed to be in the ϵ -range of its optimal value.

Now consider all the (i, t) 's with $\lambda_{i,t}^1 = (1 + \epsilon)\lambda^1 + \Delta$. As all the rates that get fixed are fixed to a value $\lambda_{i,t} = \lambda^1 \leq \lambda_{i,t}^1$, it follows that in the next iteration all the rates that did not get fixed can be increased by at least $\epsilon\lambda^1 + \Delta$, which FIXING-THE-RATES properly determines.

The inductive step. Suppose that up to iteration $k \geq 2$ all the rates that get fixed are in the ϵ -optimal range, and observe the iteration k . All the rates that got fixed prior to iteration k satisfy:

$$\begin{aligned} \lambda_{i,t}^k &\geq \lambda_{i,t}^{k-1} + F_{i,t}^k \cdot \lambda^k = \lambda_{i,t}^{k-1}, \text{ and} \\ \lambda_{i,t}^k &\leq \lambda_{i,t}^{k-1} + F_{i,t}^k \cdot (\epsilon\lambda_{i,t}^{k-1} + (1 + \epsilon)\lambda^k + \Delta) = \lambda_{i,t}^{k-1} \end{aligned}$$

and, therefore, they remain fixed for the next iteration, as $\lambda_{i,t}^k = \lambda_{i,t}^{k-1} < (1 + \epsilon)\lambda_{i,t}^{k-1}$.

Now consider all the (i, t) 's with $F_{i,t}^k = 1$. We have that:

$$\begin{aligned} \lambda_{i,t}^k &\geq \lambda^{k-1} + 1 \cdot \lambda^k = \sum_{l=1}^k \lambda^l \\ \lambda_{i,t}^k &\leq (1 + \epsilon) (\lambda^{k-1} + 1 \cdot \lambda^k) + \Delta = (1 + \epsilon) \sum_{l=1}^k \lambda^l + \Delta \end{aligned}$$

Similarly as in the base case, if $\lambda_{i,t}^k < (1 + \epsilon) \sum_{l=1}^k \lambda^l + \Delta$, let $S_{i,t}^k = \{\lambda_{j,\tau}^k : \lambda_{j,\tau}^k > \lambda_{i,t}^k\}$. There must be at least one such (i, t) , otherwise the rate maximization did not output an ϵ -approximate solution. In any lexicographically greater solution:

$$\begin{aligned} \lambda_{i,t,\max}^k &\leq \frac{1}{|S_{i,t}^k|} \left(\lambda_{i,t}^k + \sum_{\lambda_{j,\tau}^k \in S_{i,t}^k} \lambda_{j,\tau}^k \right) \\ &< (1 + \epsilon) \sum_{l=1}^k \lambda^l + \Delta, \end{aligned}$$

which implies $\lambda_{i,t,\max}^k \leq (1 + \epsilon) \sum_{l=1}^k \lambda^l$. Therefore, if we fix $\lambda_{i,t}$ to the value $\sum_{l=1}^k \lambda^l$, it is guaranteed to be at least as high as $(1 - \epsilon)$ times the value it gets in the lexicographically maximum solution.

Finally, all the (i, t) 's with $\lambda_{i,t}^k = (1 + \epsilon) \sum_{l=1}^k \lambda^l + \Delta$ can simultaneously increase their rates by at least $\epsilon \sum_{l=1}^k \lambda^l + \Delta$ in the next iteration, so it should be $F_{i,t}^{k+1} = 1$, which agrees with FIXING-THE-RATES. \square

Lemma 6.8. *An FPTAS for P-FRACTIONAL can be implemented in time:*

$$\tilde{O}(nT(T^2\epsilon^{-2} \cdot (nT + MCF(n, m) + LP(mT, nT))),$$

where $LP(mT, nT)$ denotes the running time of a linear program with mT variables and nT constraints, and $MCF(n, m)$ denotes the running time of a min-cost flow algorithm run on a graph with n nodes and m edges.

Proof. It was demonstrated in the proof of Lemma 6.7 that in every iteration at least one rate gets fixed. Therefore, there can be at most $O(nT)$ iterations. From Lemma 6.5, MAXIMIZING-THE-RATES can be implemented in time $\tilde{O}(T^2\epsilon^{-2} \cdot (nT + MCF(n, m)))$. The time required for running FIXING-THE-RATES is $LP(mT, nT)$, where $LP(mT, nT)$ denotes the running time of a linear program with mT variables and nT constraints. \square

Note: A linear programming framework as in [8, 24, 31] when applied to P-FRACTIONAL would yield a running time equal to $O(n^2T^2 \cdot LP(mT, nT))$. As the running time of an iteration in our approach is dominated by $LP(mT, nT)$, the improvement in running time is at least $O(nT)$ -fold, at the expense of providing an ϵ -approximation.

7 Fixed Fractional Routing

Suppose that we want to solve lexicographic maximization of the rates keeping both the routing and the rates constant over time. Observe that, as both the routing and the rates do not change over time, the energy consumption per time slot of each node i is also constant over time and equal to $\Delta b_i = c_{\text{st}}\lambda_i + c_{\text{rt}} \sum_{(j,i) \in E} f_{ji}$.

Proposition 7.1. *Maximum constant energy consumption Δb_i can be determined in time $O(T \log(\frac{b_{i,1} + e_{i,1}}{\delta}))$ for each node $i \in V \setminus \{s\}$, for the total time of $O(nT \log(\frac{b_{i,1} + e_{i,1}}{\delta}))$.*

Proof. Since the battery evolution can be stated as:

$$b_{i,t+1} = \min \{B, \quad b_{i,t} + e_{i,t} - \Delta b_i\},$$

maximum Δb_i for which $b_{i,t+1} \geq 0 \forall t \in \{1, \dots, T\}$ can be determined via a binary search from the interval $[0, b_{i,1} + e_{i,1}]$, for each node i . \square

Similarly as in previous sections, let $F_i^k = 0$ if the rate i is fixed at the beginning of iteration k , and $F_i^k = 1$ if it is not. Initially: $F_i^1 = 1, \forall i$. Rate maximization can then be implemented as follows:

Algorithm 4 MAXIMIZING-THE-RATES(G, F^k, b, e, k)

- 1: $\lambda_{\max}^k = \frac{1}{c_{\text{st}}} \min_i \{\Delta b_i - c_{\text{st}}\lambda_i^{k-1} : F_i^k = 1\}$
 - 2: **repeat** for $\lambda^k \in [0, \lambda_{\max}^k]$, via binary search
 - 3: Set the supply of node i to $d_i = \lambda^{k-1} + F_i^k \lambda^k$, capacity of node i to $u_i = \frac{1}{c_{\text{rt}}}(\Delta b_i - c_{\text{st}}\lambda^k)$, for each i
 - 4: Set the demand of the sink to $\sum_i d_i$
 - 5: Solve feasible flow problem on G
 - 6: **until** λ^k takes maximum value for which the flow problem is feasible on G
-

The remaining part of the algorithm is to determine which rates should be fixed at the end of iteration k . We note that in each iteration k , the maximization of the rates produces a flow f in the graph G^k with the supply rates λ_i^k . Instead of having capacitated nodes, we can modify the input graph by a standard procedure of splitting each node i into two nodes i' and i'' , and assigning the capacity of i to the edge (i', i'') . This allows us to obtain a residual graph $G^{r,k}$ for the given flow. We claim the following:

Lemma 7.2. *The rate λ_i of a node $i \in G$ can be further increased in the iteration $k+1$ if and only if there is a directed path from i to the sink node in $G^{r,k}$.*

Proof. First, observe that the only capacitated edges in G^k are those corresponding to the nodes that were split. The capacity of an edge (i', i'') corresponds to the maximum per-slot energy the node i can spend without violating the battery non-negativity constraint. If an edge (i', i'') has residual capacity of $u_{(i', i'')}^r > 0$, then the node i can spend additional $c_{\text{rt}} u_{(i', i'')}^r$ amount of energy keeping the battery level non-negative in all the time slots. If (i', i'') has no residual capacity ($u_{(i', i'')}^r = 0$), then the battery level of node i reaches zero in at least one time slot, and increasing the energy consumption per time slot leads to $b_{i,t} < 0$ for some t , which is infeasible.

(\Leftarrow) Suppose that the residual graph contains no directed path from the node i to the sink. By the flow augmentation theorem [1], the flow from the node i cannot be increased even when the flows from all the remaining nodes are kept constant. As the capacities correspond to the battery levels at the nodes, sending more flow from i causes at least one node's battery level to become negative.

(\Rightarrow) Suppose that there is a directed path from i to the sink, and let $u_i^r > 0$ denote the minimum residual capacity of the edges (split nodes) on that path. Then each node on the path can spend at least $c_{\text{rt}} u_i^r$ amount of energy maintaining feasibility. Let U denote the set of all the nodes that have a directed path to the sink in $G^{r,k}$. Then increasing the rate of each node $i \in U$ by $\Delta \lambda = \frac{\min_i u_i^r c_{\text{rt}}}{c_{\text{st}} + n c_{\text{rt}}} > 0$ and augmenting the flows of $i \in U$ over their augmenting paths in $G^{r,k}$ each node on any augmenting path spends at most $\min_i u_i^r c_{\text{rt}}$ amount of energy, which is at most equal to the energy the node is allowed to spend maintaining feasibility. \square

Lemma 7.3. WATER-FILLING-FRAMEWORK for P-FIXED-FRACTIONAL can be implemented in time

$$O(n \log(\frac{b_{i,1} + e_{i,1}}{\delta})(T + MF(n, m))),$$

where $MF(n, m)$ denotes the running time of a max-flow algorithm for a graph with n nodes and m edges.

Proof. From Proposition 7.1, determining the values of Δb_i for $i \in V \setminus \{s\}$ can be implemented in time $O(nT \log(\frac{b_{i,1} + e_{i,1}}{\delta}))$.

Running time of an iteration of WATER-FILLING-FRAMEWORK is determined by the running times of MAXIMIZING-THE-RATES and FIXING-THE-RATES. Each iteration of the binary search in MAXIMIZING-THE-RATES constructs and solves a feasible flow problem, which is dominated by the time required for running a max-flow algorithm that solves feasible flow problem on the graph G . Therefore, MAXIMIZING-THE-RATES can be implemented in time $O(\log(\frac{b_{i,1} + e_{i,1}}{\delta})MF(n, m))$, where $MF(n, m)$ denotes the running time of a max-flow algorithm.

FIXING-THE-RATES constructs a residual graph $G^{r,k}$ and runs a breadth-first search on this graph, which can be implemented in time $O(n + m)$ ($= O(MF(n, m))$ for all the existing max-flow algorithms).

Every iteration of WATER-FILLING-FRAMEWORK fixes at least one of the rates λ_i , $i \in V \setminus \{s\}$, which implies that there can be at most n iterations.

Therefore, the total running time is

$$O(n \log(\frac{b_{i,1} + e_{i,1}}{\delta})(T + MF(n, m))).$$

□

8 Determining a Routing

In this section we demonstrate that solving P-UNSPLITTABLE-FIND and P-TREE-FIND is NP-hard for both problems. Moreover, we show that it is NP-hard to obtain an approximation ratio better than $\Omega(\log n)$ for P-TREE-FIND. For P-UNSPLITTABLE-FIND, we design an efficient combinatorial algorithm for a relaxed version of this problem—it determines a time-invariable unsplittable routing that maximizes the minimum rate.

8.1 Unsplittable Routing

Lemma 8.1. P-UNSPLITTABLE-FIND is NP-hard.

Proof. The proof of NP-hardness for P-UNSPLITTABLE-FIND is a simple extension of the proof of NP-hardness for max-min fair unsplittable routing provided in [21]. We use the same reduction as in [21], derived from the non-uniform load balancing problem [22]. From [21, 22], the following problem is NP-hard: P-NON-UNIFORM-LOAD-BALANCING: Let $J = \{J_1, \dots, J_k\}$ be a set of jobs, and $M = \{M_1, \dots, M_n\}$ be a set of machines. Each job J_i has a time requirement $r_i \in \{1/2, 1\}$, and the sum of all the job requirements is equal to n : $\sum_{i=1}^k r_i = n$. Each job $J_i \in J$ can be run only on a subset of the machines $S_i \subset M$. Is there an assignment of jobs to machines, such that the sum requirement of jobs assigned to each machine M_j equals 1?

For a given instance of P-NON-UNIFORM-LOAD-BALANCING we construct an instance of P-UNSPLITTABLE-FIND as follows (Fig. 6). Let $T = 1$, and $c_{st} = c_{rt} = 1$. Create a node J_i for each job $J_i \in J$, a node M_j for each machine $M_j \in M$, and add an edge (J_i, M_j) if $M_j \in S_i$. Connect all the nodes $M_j \in M$ to the sink. Let available energies at the nodes be $b_{J_i} = r_i$, $b_{M_j} = 2$.

Suppose that the instance of P-NON-UNIFORM-LOAD-BALANCING is a "yes" instance, i.e., there is an assignment of jobs to machines such that the sum requirement of jobs assigned to each machine equals 1. Observe the following rate assignment: $\lambda^* = \{\lambda_{J_i} = r_i, \lambda_{M_j} = 1\}$. This rate assignment is feasible only for the unsplittable routing in which M_j 's descendants are the jobs assigned to M_j in the solution for P-NON-UNIFORM-LOAD-BALANCING. Moreover, as in this rate assignment all the nodes spend all their available energies and since $\sum_{i=1}^k b_{J_i} = \sum_{i=1}^k r_i = n$, it is not hard to see that this is the lexicographically

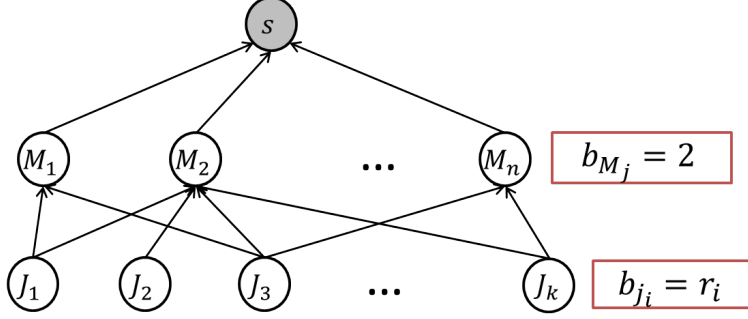


Figure 6: A reduction from P-NON-UNIFORM-LOAD-BALANCING for proving NP-hardness of P-UNSPLITTABLE-FIND. Jobs are represented by nodes J_i , machines by nodes M_j , and there is an edge from J_i to M_j if job J_i can be executed on machine M_j . Each job J_i has time requirement $r_i \in \{1/2, 1\}$, and $\sum_{i=1}^k J_i = n$. Available energies at the nodes are shown in the boxes next to the nodes. If at the optimum of P-UNSPLITTABLE-FIND $\lambda_{J_i} = r_i$ and $\lambda_{M_j} = 1$, then there is an assignment of jobs to the machines such that the sum requirement of jobs assigned to each machine equals 1.

maximum rate assignment that can be achieved for any instance of P-NON-UNIFORM-LOAD-BALANCING. If the instance of P-NON-UNIFORM-LOAD-BALANCING is a "no" instance, then P-UNSPLITTABLE-FIND at the optimum necessarily produces a rate assignment that is lexicographically smaller than λ^* .

Therefore, if P-UNSPLITTABLE-FIND can be solved in polynomial time, then P-NON-UNIFORM-LOAD-BALANCING can also be solved in polynomial time. \square

As the proof of Lemma 8.1 is constructed for $T = 1$, it follows that P-UNSPLITTABLE-FIND is NP-hard for general T , in either time-variable or time-invariable setting.

On the other hand, determining a time-invariable unsplittable routing that guarantees the maximum value of the minimum sensing rate over all time-invariable unsplittable routings is solvable in polynomial time, and we provide a combinatorial algorithm that solves it below.

We first observe that in any time-invariable unsplittable routing, if all the nodes are assigned the same sensing rate λ , then every node i spends a fixed amount of energy Δb_i per time slot equal to the energy spent for sensing and sending own flow and for forwarding the flow coming from the descendant nodes: $\Delta b_i = \lambda (c_{st} + c_{rt} D_{i,t})$.

The next property we use follows from the integrality of the max flow problem with integral capacities (see, e.g., [1]). This property was stated as a theorem in [20] for single-source unsplittable flows, and we repeat it here for the equivalent single-sink unsplittable flow problem:

Theorem 8.2. [20] *Let $G = (N, E)$ be a given graph with the predetermined sink node s . If the supplies of all the nodes in the network are from the set $\{0, \lambda\}$, $\lambda > 0$, and the capacities of all the edges/nodes are integral multiples of λ , then: if there is a fractional flow of value f , there is an unsplittable flow of value at least f . Moreover, this unsplittable flow can be found in polynomial time.*

Note: For the setting of Theorem 8.2, any augmenting-path or push-relabel based max flow algorithm produces a flow that is unsplittable, as a consequence of the integrality of the solution produced by these algorithms. We will assume that the used max-flow algorithm has this property.

The last property we need is that our problem can be formulated in the setting of Theorem 8.2. We observe that for a given sensing rate λ , each node spends $c_{st}\lambda$ units of energy for sensing, whereas the remaining energy can be used for routing the flow originating at other nodes. Therefore, for a given λ , we can set the supply of each node i to λ , set its capacity to $u_i = (\Delta b_i - c_{st}\lambda)/c_{rt}$ (making sure that $\Delta b_i - c_{st}\lambda \geq 0$), and observe the problem as the feasible flow problem. For any feasible *unsplittable* flow solution with all the supplies equal to λ , we have that flow through every edge/node equals the sum flow of all the routing paths that contain that edge/node. As every path carries a flow of value λ , the flow through every edge/node is an integral multiple of λ . Therefore, to verify whether it is feasible to have a sensing rate of λ at each node, it is enough to down-round all the nodes' capacities to the nearest multiple of λ : $u_i = \lambda \cdot \lfloor (\Delta b_i - c_{st}\lambda)/(c_{rt}\lambda) \rfloor$, and apply the Theorem 8.2.

An easy upper bound for λ is $\lambda_{\max} = \min_i \Delta b_i / c_{\text{st}}$, which follows from the battery nonnegativity constraint. The algorithm becomes clear now:

Algorithm 5 MAXMIN-UNSPLITTABLE-ROUTING(G, b, e)

- 1: Perform a binary search for $\lambda \in [0, \lambda_{\max}]$.
 - 2: For each λ chosen by the binary search set node supplies to λ and node capacities to $u_i = \lambda \cdot \lfloor (\Delta b_i - c_{\text{st}}\lambda) / (c_{\text{rt}}\lambda) \rfloor$. Solve feasible flow problem.
 - 3: Return the maximum feasible λ .
-

Lemma 8.3. *The running time of MAXMIN-UNSPLITTABLE-ROUTING is $O(\log(\min_i(b_{i,1} + e_{i,1})/(c_{\text{st}}\delta)))(MF(n+1, m))$, where $MF(n, m)$ is the running time of a max-flow algorithm on an input graph with n nodes and m edges.*

8.2 Routing Tree

If it was possible to find the (either time variable or time-invariable) max-min fair routing tree in polynomial time for any time horizon T , then the same result would follow for $T = 1$. It follows that if P-TREE-FIND NP-hard for $T = 1$, it is also NP-hard for any $T > 1$. Therefore, we restrict our attention to $T = 1$.

Assume w.l.o.g. $e_{i,1} = 0 \forall i \in V \setminus \{s\}$. Let \mathcal{T} denote a routing tree on the given graph G , and $D_i^{\mathcal{T}}$ denote the number of descendants of a node i in the routing tree \mathcal{T} . Maximization of the common rate $\lambda_i = \lambda$ over all routing trees can be stated as:

$$\max_{\mathcal{T}} \min_{i \in N} b_i / (c_{\text{st}} + c_{\text{rt}} D_i^{\mathcal{T}}) \quad (26)$$

This problem is equivalent to maximizing the network lifetime for $\lambda_i = 1 \forall i \in V \setminus \{s\}$ as studied in [5]. This problem, which we call P-MAXIMUM-LIFETIME-TREE, was proved to be NP-hard in [5] using a reduction from the SET-COVER problem [19]. The instance used in [5] for showing the NP-hardness of the problem has the property that the equivalent problem of finding a tree with the lexicographically maximum rate assignment, P-TREE-FIND, is such that at the optimum $\lambda_1 = \lambda_2 = \dots = \lambda_n = \lambda$. Therefore, P-TREE-FIND is also NP-hard.

We will strengthen the hardness result here and show that the lower bound on the approximation ratio for the P-TREE-FIND problem is $\Omega(\log n)$. Notice that because we are using the instance for which at the optimum $\lambda_i = \lambda \forall i$, the meaning of the approximation ratio is clear. In general, the optimal routing tree can have a rate assignment with distinct values of the rates, in which case we would need to consider an approximation to a vector $\{\lambda_i\}_{i \in \{1, \dots, n\}}$. However, we note that for any reasonable definition of approximation (e.g., element-wise or prefix-sum as in [21]) our result for the lower bound is still valid. As for the instance we use P-TREE-FIND is equivalent to the P-MAXIMUM-LIFETIME-TREE problem, the lower bound applies to both problems.

We extend the reduction from the SET-COVER problem used in [5] to prove the lower bound on the approximation ratio. In the SET-COVER problem, we are given elements $1, 2, \dots, n^*$ and sets $S_1, S_2, \dots, S_m \subset \{1, 2, \dots, n^*\}$. The goal is to determine the minimum number of sets from S_1, \dots, S_m that cover all the elements $\{1, \dots, n^*\}$. Alternatively, the problem can be recast as a decision problem that determines whether there is a set cover of size k or not. Then the minimum set cover can be determined by finding the smallest k for which the answer is "yes".

Suppose that there exists an approximation algorithm that solves P-TREE-FIND (or P-MAXIMUM-LIFETIME-TREE) with the approximation ratio r . For a given instance of SET-COVER, construct an instance of P-TREE-FIND as in Fig. 7 and denote it by G . This reduction is similar to the reduction used in [5], with modifications being made by adding line-topology graphs, and by modifying the node capacities appropriately to limit the size of the solution to the corresponding SET-COVER problem. Let l^x denote a directed graph with line topology of size x . Assume that all the nodes in any l^x have capacities that are non-constraining. By the same observations as in the proof of NP-completeness of P-MAXIMUM-LIFETIME-TREE [5], if there is a routing tree that achieves $\lambda = 1$, then there is a set cover of size k for the given input instance of SET-COVER.

Now observe a solution that an approximation algorithm with the ratio r would produce, that is, an algorithm for which $\frac{1}{r} \leq \lambda \leq 1$ when $\lambda_{\text{OPT}} = 1$.

Table 2: Running times of the algorithms for the WATER-FILLING-FRAMEWORK implementation.

| | MAXIMIZING-THE-RATES | FIXING-THE-RATES | Total |
|---------------------|---|------------------|--|
| P-UNSPLITTABLE-FIND | $O(nT \log(\frac{B + \max_{i,t} e_{i,t}}{\delta c_{\text{st}}}))$ | $O(mT)$ | $O(nT(nT \log(\frac{B + \max_{i,t} e_{i,t}}{\delta c_{\text{st}}}) + mT))$ |
| P-FIXED-FRACTIONAL | $O(n \log(\frac{b_{i,1} + e_{i,1}}{\delta})(T + MF(n, m)))$ | $O(m)$ | $O(n \log(\frac{b_{i,1} + e_{i,1}}{\delta})(T + MF(n, m)))$ |
| P-FRACTIONAL | $\tilde{O}(T^2 \epsilon^{-2} \cdot (nT + MCF(n, m)))$ | $LP(mT, nT)$ | $\tilde{O}(nT(T^2 \epsilon^{-2} \cdot (nT + MCF(n, m) + LP(mT, nT)))$ |

(verifying whether $k = m$ is a set cover is trivial) and find a $3r$ -approximation for the minimum set cover, which is stated in the following lemma.

Lemma 8.6. *If there is a polynomial-time r -approximation algorithm for P-TREE-FIND, then there is a polynomial-time $3r$ -approximation algorithm for SET-COVER.*

Proof. Suppose that there was an algorithm that solves P-TREE-FIND in polynomial time with some approximation ratio r . For a given instance of SET-COVER construct an instance of P-TREE-FIND as in Fig. 7. Solve (approximately) P-TREE-FIND for $k \in \{1, \dots, m-1\}$. In all the solutions, it must be $\lambda \leq 1$. Let k_m denote the minimum $k \in \{1, \dots, m-1\}$ for which $\lambda \geq \frac{1}{r}$. Then the minimum set cover size for the input instance of SET-COVER is $k^* \geq k_m$, otherwise there would be some other $k'_m < k_m$ for which $\lambda \geq \frac{1}{r}$. From Lemmas 8.4 and 8.5, the solution to the constructed instance of P-TREE-FIND corresponds to a set cover of size $p \leq 3r \cdot k_m$ for the input instance. But this implies $p \leq 3r \cdot k^*$, and, therefore, the algorithm provides a $3r$ -approximation to the SET-COVER. \square

Theorem 8.7. *The lower bound on the approximation ratio of P-TREE-FIND is $\Omega(\log n)$.*

Proof. The lower bound on the approximation ratio of SET-COVER was shown to be $\Omega(\log n)$ in [25].

The proof for the lower bound on the approximation ratio given in [25] was derived assuming a polynomial relation between n^* and m . Therefore, the lower bound of $\Omega(\log n^*)$ holds for $m = n^{*c^*}$, where $c^* \in \mathbb{R}$ is some positive constant. Assume that $n^* \geq 3$. The graph given for an instance of SET-COVER (as in Fig. 7) contains $n = 2rn^* + mrn^* + 3 \leq rn^{*c'}$ nodes, for some other constant $c' > 1$. Therefore: $n^* \geq \sqrt[c']{\frac{n}{r}}$. As $r \geq \frac{1}{3}c \log n^*$, it follows that:

$$\begin{aligned}
 r &\geq \frac{1}{3}c \log \sqrt[c']{\frac{n}{r}} = \frac{c}{3c'}(\log n - \log r) \\
 \Leftrightarrow \frac{c}{3c'} \log r + r &\geq \frac{c}{3c'} \log n \Rightarrow r \geq c'' \log n,
 \end{aligned}$$

for some $c'' \in \mathbb{R}$. \square

9 Conclusions and Future Work

This paper presents a comprehensive algorithmic study of the max-min fair rate assignment and routing problems in energy harvesting networks with predictable energy profile. We develop algorithms for the WATER-FILLING-FRAMEWORK implementation under various routing types. The running times of the developed algorithms are summarized in Table 2. The algorithms provide important insights into the structure of the problems, and can serve as benchmarks for evaluating distributed and approximate algorithms possibly designed for unpredictable energy profiles.

The results reveal interesting trade-offs between different routing types. While we provide an efficient algorithm that solves the rate assignment problem in any time variable or time-invariable unsplittable routing or a routing tree, we also show that determining a routing with the lexicographically maximum rate assignment for any of these settings is NP-hard. On the positive side, we are able to construct a combinatorial

algorithm that determines a time-invariable unsplittable routing which maximizes the minimum sensing rate assigned to any node in any time slot.

Fractional time-variable routing provides the best rate assignment (in terms of lexicographical maximization), and both the routing and the rate assignment are determined jointly by one algorithm. However, as demonstrated in Section 6, the problem is unlikely to be solved optimally without the use of linear programming, incurring a high running time. While we provide an FPTAS for this problem, reducing the algorithm running time by a factor of $O(nT)$ (as compared to the framework of [8, 24, 31]), the proposed algorithm still requires solving $O(nT)$ linear programs.

If fractional routing is restricted to be time-invariable and with constant rates, the problem can be solved by a combinatorial algorithm, which we provide in Section 7. However, as discussed in the introduction, constant sensing rates often result in the underutilization of the available energy.

There are several directions for future work. For example, extending the model to incorporate the energy consumption due to the control messages exchange would provide a more realistic setting. Moreover, designing algorithms for unpredictable energy profiles that can be implemented in an online and/or distributed manner is of high practical significance.

10 Acknowledgements

This research was supported in part by NSF grants CCF-1349602, CCF-09-64497, and CNS-10-54856. The authors are grateful to Prof. Mihalis Yannakakis for useful discussions.

References

- [1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.
- [2] B. Bacinoglu and E. Uysal-Biyikoglu. Finite-horizon Online Transmission Rate and Power Adaptation on a Communication Link with Markovian Energy Harvesting. *CoRR*, abs/1305.4558, 2013.
- [3] D. Bertsekas and R. Gallager. *Data networks (2nd ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1992.
- [4] P. Blasco, D. Gunduz, and M. Dohler. A learning theoretic approach to energy harvesting communication system optimization. In *Proc. IEEE Globecom Workshops'12*, 2012.
- [5] C. Buragohain, D. Agrawal, and S. Suri. Power aware routing for sensor databases. In *Proc. IEEE INFOCOM'05*, 2005.
- [6] J.-H. Chang and L. Tassiulas. Maximum lifetime routing in wireless sensor networks. *IEEE/ACM Trans. Netw.*, 12(4):609–619, 2004.
- [7] A. Charny, D. D. Clark, and R. Jain. Congestion control with explicit rate indication. In *Proc. IEEE ICC'95*, 1995.
- [8] S. Chen, Y. Fang, and Y. Xia. Lexicographic maxmin fairness for data collection in wireless sensor networks. *IEEE Trans. Mobile Comput.*, 6(7):762–776, July 2007.
- [9] S. Chen, P. Sinha, N. Shroff, and C. Joo. Finite-horizon energy allocation and routing scheme in rechargeable sensor networks. In *Proc. IEEE INFOCOM'11*, 2011.
- [10] S. Chen, P. Sinha, N. Shroff, and C. Joo. A simple asymptotically optimal energy allocation and routing scheme in rechargeable sensor networks. In *Proc. IEEE INFOCOM'12*, 2012.
- [11] S. DeBruin, B. Campbell, and P. Dutta. Monjolo: an energy-harvesting energy meter architecture. In *ACM SenSys'13*, 2013.
- [12] M. Gatzianas, L. Georgiadis, and L. Tassiulas. Control of wireless networks with rechargeable batteries. *IEEE Trans. Wireless Commun.*, 9(2):581–593, 2010.

- [13] M. Gorlatova, A. Bernstein, and G. Zussman. Performance evaluation of resource allocation policies for energy harvesting devices. In *Proc. WiOpt'11*, 2011.
- [14] M. Gorlatova, P. Kinget, I. Kymissis, D. Rubenstein, X. Wang, and G. Zussman. Challenge: ultra-low-power energy-harvesting active networked tags (EnHANTs). In *Proc. ACM MobiCom'09*, 2009.
- [15] M. Gorlatova, R. Margolies, J. Sarik, G. Stanje, J. Zhu, B. Vignaham, M. Szczodrak, L. Carloni, P. Kinget, I. Kymissis, and G. Zussman. Energy harvesting active networked tags (EnHANTs): Prototyping and experimentation. Technical Report 2012-07-27, Columbia University, July 2012.
- [16] M. Gorlatova, A. Wallwater, and G. Zussman. Networking low-power energy harvesting devices: Measurements and algorithms. *IEEE Trans. Mobile Comput.*, 12(9):1853–1865, 2013.
- [17] B. Gurakan, O. Ozel, J. Yang, and S. Ulukus. Energy cooperation in energy harvesting two-way communications. In *Proc. IEEE ICC'13*, 2013.
- [18] L. Huang and M. Neely. Utility optimal scheduling in energy-harvesting networks. *IEEE/ACM Trans. Netw.*, 21(4):1117–1130, 2013.
- [19] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.
- [20] J. Kleinberg. Single-source unsplittable flow. In *Proc. IEEE FOCS'96*, 1996.
- [21] J. Kleinberg, Y. Rabani, and E. Tardos. Fairness in routing and load balancing. In *Proc. IEEE FOCS'99*, 1999.
- [22] J. Lenstra, D. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical programming*, 46(1-3):259–271, 1990.
- [23] L. Lin, N. Shroff, and R. Srikant. Asymptotically optimal energy-aware routing for multihop wireless networks with renewable energy sources. *IEEE/ACM Trans. Netw.*, 15(5):1021–1034, 2007.
- [24] 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. Netw.*, 19(4):947–960, Aug. 2011.
- [25] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41(5):960–981, Sept. 1994.
- [26] R. Madan and S. Lall. Distributed algorithms for maximum lifetime routing in wireless sensor networks. *IEEE Trans. Wireless Commun.*, 5(8):2185–2193, 2006.
- [27] Z. Mao, C. Koksall, and N. Shroff. Near optimal power and rate control of multi-hop sensor networks with energy replenishment: Basic limitations with finite energy and data storage. *IEEE Trans. Autom. Control*, 57(4):815–829, 2012.
- [28] N. Megiddo. Optimal flows in networks with multiple sources and sinks. *Mathematical Programming*, 7(1):97–107, 1974.
- [29] O. Ozel, K. Tutuncuoglu, J. Yang, S. Ulukus, and A. Yener. Transmission with energy harvesting nodes in fading wireless channels: Optimal policies. *IEEE J. Sel. Areas Commun.*, 29(8):1732–1743, 2011.
- [30] S. Plotkin, D. Shmoys, and E. Tardos. Fast approximation algorithms for fractional packing and covering problems. *Math. of O.R.*, 20(2):257–301, 1995.
- [31] B. Radunović and J.-Y. L. Boudec. A unified framework for max-min and min-max fairness with applications. *IEEE/ACM Trans. Netw.*, 15(5):1073–1083, Oct. 2007.
- [32] S. Sarkar, M. Khouzani, and K. Kar. Optimal routing and scheduling in multihop wireless renewable energy networks. *IEEE Trans. Autom. Control*, 58(7):1792–1798, 2013.

- [33] S. Sarkar and L. Tassiulas. Fair allocation of discrete bandwidth layers in multicast networks. In *Proc. IEEE INFOCOM'00*, 2000.
- [34] R. Srivastava and C. Koksai. Basic performance limits and tradeoffs in energy-harvesting sensor nodes with finite data and energy storage. *IEEE/ACM Trans. Netw.*, 21(4):1049–1062, 2013.