# Crowd-Sourced Storage-Assisted Demand-Response in Microgrids

Mohammad H. Hajiesmaili
Johns Hopkins University
hajiesmaili@jhu.edu

Minghua Chen
The Chinese University of Hong Kong
minghua@ie.cuhk.edu.hk

Enrique Mallada
Johns Hopkins University
mallada@jhu.edu

Chi-Kin Chau
Masdar Institute of Science and Technology
ckchau@masdar.ac.ae

## ABSTRACT

This paper studies the problem of utilizing heterogeneous energy storage systems, including electric vehicles and residential batteries, to perform demand-response in microgrids. The objective is to minimize the operational cost while fulfilling the demand-response requirement. The design space is to select and schedule a subset of available storage devices that are heterogeneous in operating cost, capacity, and availability in time. Designing a performance-optimized solution, however, is challenging due to the combinatorial nature of the problem with mixed packing and covering constraints, and the essential need for online solution design in practical scenarios where both demand-response requirement and the profile of user-owned storage systems arrive online. We tackle these challenges and design several online algorithms, by leveraging a recent theoretical computer science technique which uses a problem-specific exponential potential function to solve online mixed packing and covering problems. We show that the fractional version of the algorithm achieves a logarithmic bi-criteria competitive ratio. Empirical trace-driven experiments demonstrate that our algorithms perform much better than the theoretical bounds and achieve close-to-optimal performance.

## CCS CONCEPTS

**•Hardware → Smart grid; •Theory of computation →** *Online algorithms;* Scheduling algorithms;

## KEYWORDS

Microgrid, crowd-sourced storage-assited demand response, competitive online algorithm design, scheduling

**ACM Reference format:**
Mohammad H. Hajiesmaili, Minghua Chen, Enrique Mallada, and Chi-Kin Chau. 2017. Crowd-Sourced Storage-Assisted Demand-Response in Microgrids. In *Proceedings of e-Energy '17, Shatin, Hong Kong, May 16-19, 2017,* 13 pages.
DOI: http://dx.doi.org/10.1145/3077839.3077841


*e-Energy '17, Shatin, Hong Kong*

DOI: http://dx.doi.org/10.1145/3077839.3077841

## 1 INTRODUCTION

Microgrid is a small-scale power system that, by leveraging renewable sources, operates autonomously to match the demand and the supply of a local community [20]. It represents a promising paradigm to address the economic, reliability, and environmental concerns encountered by today's power grids [12, 23]. As reported in [15], the global capacity of microgrids will expand by more than 5 times in 2015-2024, from 1.4GW to nearly 7.6GW.

Demand-response in smart (micro) grid [31] is regarded as a potential solution for real-time balancing between supply and demand to boost the reliability of the grid and reduce the operational costs. In the literature, at least three different approaches towards demand response in smart grid have been proposed: (i) demand reduction through shifting the flexible demand, e.g., in data centers, in temporal domain [22]; (ii) reducing the net grid demand by intelligent scheduling of local generation units [10, 28]; (iii) reducing the net demand by active participation of user-owned crowd-sourced energy storage systems and reducing their charging demand or even discharging them and releasing energy back to the microgrid [31].

In this paper, we focus on the third approach of "crowd-sourced storage-assisted demand-response" in microgrid. In this approach, hundreds of batteries of electric vehicles and smart homes equipped with residential storages [6], residing in the microgrid, with huge aggregated capacity can actively participate in microgrid demand response through reducing their charging demand or even discharging and selling back the electricity to the microgrid, e.g., through *vehicle-to-grid* scheme [16] for EVs. These storage devices can be charged during off-peak periods, and be discharged during on-peak periods. In this way, not only the microgrid can reduce its electricity bill[1], but also, the customers can earn money through participating in this scheme.

In this paper, we consider a scenario in which microgrid operator orchard heterogeneous energy storage sources, such as EVs (through V2G technology [16]) and residential batteries, to establish demand response through storage crowd-sourcing paradigm. We assume the microgrid has a MicroGrid Central Controller (MGCC) [24], through which the microgrid operator can coordinate among the distributed resources.[2]

[1]Since the residual demand of microgrid, i.e., total demand subtracted by local supply, must be fulfilled by the main grid, hence, the microgrid get charged by the main grid.
[2]Typically in microgrid, there are two main strategies for demand response: distributed demand response, which is distributed by using a PI controller at each distributed generators, or centralized demand response, which is manually mastered by the MGCC [24].

After receiving all the information of the available sources, MGCC selects a subset of energy storage systems and sets their output levels, by either reducing their charging rate or even discharging, so as to (i) fulfill the supply shortage of microgrid, for reliable operation of microgrid, and (ii) minimize total cost of involving the chosen energy storage systems, for economical operation of the microgrid.

**Challenges.** It turns out that achieving the above objectives is a formidable task since it requires to solve a joint Source Selection and Scheduling Problem (S3P hereafter), which is uniquely challenging to solve because of two critical challenges:

The first challenge originates from the heterogeneity of energy storage systems, in terms of cost, capacity, and availability in time. Theoretically speaking, heterogeneity in the S3P results in a combinatorial problem with both packing constraints, i.e., capacity constraint of the sources, and covering constraints, i.e., supply shortage of the microgrid. The 0/1 decision space in source selection part and continues variable in scheduling part, together with mixed linear packing and covering constraints make the problem as one of the mixed packing and covering combinatorial problems which is difficult to tackle in general. A careful investigation shows that a simplified version of the S3P can be re-expressed as the splittable version of Capacitated Facility Location Problem [21] (CFLP hereafter), which is known to be as a fundamental theoretical computer science problem [27]. The S3P is more challenging to solve than the CFLP, as it not only inherits all the difficulty of the CFLP as a mixed packing and covering combinatorial problem, but also involves a non-trivial "topological" constraint caused by the heterogeneous availability in time of storage systems.

The second challenge lies in the essential need for online solution design. For power demand response, the supply shortage as well as the availability of energy storage systems are revealed in a slot-by-slot fashion. The decision in the current slot depends on input of future slots and it is challenging to make optimal online scheduling decisions without knowing future input. It turns out that without knowing future input, even finding a feasible online solution for S3P is non-trivial, and violation of either packing or covering constraints is inevitable. In this paper, we follow a competitive online algorithm design to jointly minimize both cost and capacity violation. Then, we aim to analyze the performance of the algorithm using bi-criteria $(\alpha, \beta)$-competitive ratio analysis. In context of our problem, a bi-criteria $(\alpha, \beta)$-competitive online algorithm produces a solution of cost at most $\alpha$ times of the offline optimum, while violating the capacity constraints by no more that a $\beta$ factor.

We note that demand response with energy storage management has been studied in literature using other technical approach such as Lyapunov optimization [13] and Markov decision process [29]. These approaches relies on the underlying stochastic process of the input parameters. In competitive design approach, however, there is no assumptions on the underlying stochastic processes of the unknown input parameters.

**Contribution.** In this paper, we tackle S3P in online scenario and make the following contributions.

Although the centralized demand response requires additional communication infrastructure, it allows the microgrid owners to economically schedule different resources to participate in the program and thus minimize the operational cost.

$\triangleright$ We first focus on designing an online fractional algorithm for the linear-relaxed version of the S3P. Recall that even linear version of the problem is still difficult in online scenario, because the input to the time-coupled linear problem are not known in advance. By adapting the recently proposed framework for online mixed packing and covering problems [2], we propose an online fractional algorithm called OnFrc. In the OnFrc at each slot, we obtain a fractional solution for the S3P by constructing a potential function that is linear in cost and exponential in violating the capacity constraint of the storage sources. We demonstrate that the OnFrc is a bi-criteria $O(n \log n, \log n)$-competitive online algorithm, where $n$ is the number of sources.

$\triangleright$ Then, by a randomized rounding algorithm called OnInt, we obtain an integral solution for the S3P. In addition, we provide several other heuristics to improve the performance of our algorithms mainly to minimize violating the capacity constraint of the sources.

$\triangleright$ By extensive experiments using real-world data traces, we investigate the performance of both online fractional and integral algorithms. Note that although the proposed algorithms are logarithmic competitive, this is a worst-case bound and our results signify that they perform much better under practical settings. In particular, for a set of representative scenarios (in which the number of sources varies from 50 to 150), the average empirical online cost ratios, i.e., the cost obtained by our online algorithms over the offline optimum is 1.7.

The rest of the paper is organized as follows. In Sec. 2, we introduce the system model and formulate the problem. The online solution is explained in Sec. 3. The results of trace-driven experiments are given in Sec. 4. We review the literature in Sec. 5. Finally, the paper is concluded in Sec. 6.

## 2 PROBLEM FORMULATION

### 2.1 System Model

We assume that the system is time-slotted, where each time slot $t \in \mathcal{T}, (T \triangleq |\mathcal{T}|)$ has a fixed length (e.g., 1 hour) that is set by the MGCC. We assume that at each slot $t$, the microgrid has a shortage $d_t \geq 0$ in supply. In the microgrid with high penetration of renewable, it is highly difficult to predict total net supply of renewables in advance. Thereby, we assume that at the beginning of each slot only the value of $d_t$ for the incoming slot is known. Beyond that, we have no assumptions on the exact or stochastic modeling of $d_t$. By summarizing the key notations in Table 1, we proceed to introduce the properties of energy storage systems.

*2.1.1 Energy Storage Systems (ESS).* Let $\mathcal{I}, (n \triangleq |\mathcal{I}|)$, be the set of ESS[3] in the microgrid that are available to contribute in demand response scheme. By ESS we mean any type of devices like EVs, residential batteries, on-site storages for data centers, etc., that can be connected to the microgrid and participate in demand response by either reducing their charging rate or discharging back to the microgrid. The sources are heterogeneous in terms of (1) availability over time horizon, (2) demand-response energy capacity, and (3) operating cost.

[3] In this paper, we use ESS and source interchangeably, since ESS are the sources that can cover the energy shortage in demand response scheme for the microgrid.

**Table 1: Summary of key notations**

| Notation | Description |
|---|---|
| $\mathcal{I}$ | The set of sources (energy storage systems), ($n \triangleq \|\mathcal{I}\|$) |
| $\mathcal{T}$ | The set of time slots, ($T \triangleq \|\mathcal{T}\|$) |
| $c_i$ | Total available energy of source $i$ |
| $k_i$ | Maximum discharge rate of source $i$ at each slot |
| $f_i$ | Fixed-cost of source $i$ |
| $u_i$ | Unit-cost of source $i$ |
| $\mathcal{T}_i$ | $\mathcal{T}_i = [a_i, b_i]$, available interval of source $i$, where $a_i \le b_i \in [1, T]$ are arrival and departure slots |
| $d_t$ | Supply shortage at $t$ |
| $x_i$ | **Opt. variable**, 1: source $i$ is selected, 0, otherwise |
| $y_i(t)$ | **Opt. variable**, amount of supply shortage that is covered by source $i$ at slot $t$ |

**Available interval:** Source $i$ is available in interval $\mathcal{T}_i \subseteq \mathcal{T}$, which is $\mathcal{T}_i = [a_i, b_i]$, where $a_i$ is the arrival slot and $b_i$ is its departure slot. This captures the availability of sources, e.g., EVs are available in different intervals in the parking lots, or residential batteries are available during the intervals that their own usage is low. In our model, we also assume sources arrive online, i.e., at the beginning of each slot, only the full information of available sources is known, and we have no exact or stochastic information of the sources that their arrival is in the future slots.

**Capacity:** We assume that without participating in demand response scheme, each source $i$ is charged at the maximum charging rate $\kappa_i$ during its available interval $\mathcal{T}_i$. Hence, its total state of the charge at departure time slot would be $\mathrm{SoC}_i^{\mathrm{a}} + \kappa_i|\mathcal{T}_i|$[4], where $\mathrm{SoC}_i^{\mathrm{a}}$ is the state of charge at arrival of source $i$.

By participating in demand response, the charging rate of the sources could be decreased to meet the supply shortage of the microgrid. In addition, even discharging is allowed in some particular slots in which the supply shortage is large.

The total aggregate energy that is deducted from each source during its available window is limited and is set by each source separately, based on the total capacity of the battery and user preferences. More specifically, let $\mathrm{SoC}_i^{\mathrm{d}}$ be the minimum requirement of the state of charge at departure for source $i$. Now, we get $c_i$ as total available energy of source $i$ to participate in demand response as follows:

$$c_i = \mathrm{SoC}_i^{\mathrm{a}} + \kappa_i|\mathcal{T}_i| - \mathrm{SoC}_i^{\mathrm{d}}, \tag{1}$$

where we assume that $\mathrm{SoC}_i^{\mathrm{d}} < \mathrm{SoC}_i^{\mathrm{a}} + \kappa_i|\mathcal{T}_i|$, hence $c_i > 0$. This means that source $i$ has a positive amount of energy to contribute in demand response.

Each source $i$ has a single-slot capacity $k_i$ that is the aggregation of the charging and discharging rates of source $i$. The parameter $k_i$ captures the maximum amount of the energy that source $i$ can contribute at each slot. Hereafter, for brevity and with a phrase abuse, we call $k_i$ as the maximum discharge rate of source $i$, while in fact it is the aggregation of the charging and discharging rates of the source.

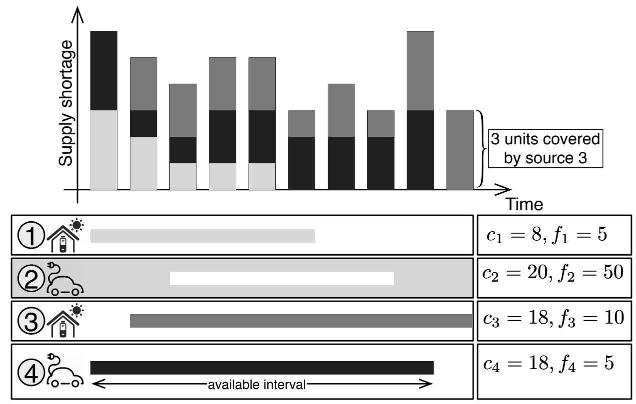



**Figure 1: An illustration of the system model and source selection and scheduling problem. A simple example with $T = 10$ and 4 different sources with different costs, capacities, and availabilities. For simplicity, we assume equal unit costs for all sources. The problem is to *select* and *schedule* the deduction amount of selected sources such that the supply shortage is fulfilled at each slot. In this example, the optimal solution is to select sources 1, 3, and 4. A feasible scheduling is also depicted by color-coded shading of the shortage bars. Source 2 is not selected since it is too expensive and by proper scheduling all shortages could be covered by the other sources. See Sec. 5 for the importance of scheduling in this problem.**

**Cost:** Cost model of source $i$ consists of two parts: (i) a fixed (start-up) cost $f_i$, which is fixed value regardless of the amount of energy that is solicited, and (ii) a unit cost $u_i$ which must be multiplied by the volume of energy that is contributed by source $i$ during its available interval to compute the volume cost. We assume that the sources always declare their true cost values. Extension of the framework into a truthful mechanism, in which truth-telling is the dominant strategy of the sources is part of the future study.

**Illustrative example:** In Fig. 1, we use a simple example with four heterogeneous sources to clarify the design space of the problem. The properties of the sources are mentioned in the figure. We note that the available interval of the sources are different. On the other hand, at each slot, the microgrid encounters different amount in supply shortage, as shown in the bar plot of Fig. 1. By using each source, the microgrid is charged a fixed cost regardless of how much energy is covered by the source. Hence, the first design space in the cost minimization problem is to select minimum number of sources. Furthermore, since the supply shortages and the availability of sources are different at different slots, the second design space is to schedule the sources, i.e., set the amount of deduction in their charging, so as to compensate for the supply shortage. In Fig. 1, an optimal solution (in which sources 1, 3, and 4 with total

[4] We assume that the storage capacity of source $i$ is large enough such that $\mathrm{SoC}_i^{\mathrm{a}} + \kappa_i|\mathcal{T}_i|$ is less than the storage capacity.

cost of 20 are selected) along with the feasible scheduling (which is color-coded in the bar plot) is shown.

We finally note that joint consideration of source selection and scheduling makes the problem difficult. There are some studies in the literature [30, 31] which assume that the scheduling is fixed, i.e., in the context of our problem the deduction amount is constant and fixed at each slot, and solve the source selection problem (also known as winner determination problem in the context of auction design [31]) in online manner. However, this simplification leads to sub-optimal solution and reduces the feasibility region of the problem. For detailed discussion, we refer to Sec. 5.

## 2.2 Problem Formulation

Given the set of heterogeneous sources, the objective of MGCC is to use the potentials of the available sources by *selecting* a subset of them such that by a proper *scheduling*, (i) its supply shortage during time horizon is covered, and (ii) at the same time total cost is minimized. Hence, the underlying optimization problem is a joint *source selection and scheduling problem* (S3P) that is formulated as

$$\begin{aligned}
\text{S3P : min} \quad & \sum_{i \in \mathcal{I}} \Big( f_i x_i + u_i \sum_{t \in \mathcal{T}_i} y_i(t) \Big) \\
\text{s.t.} \quad & \sum_{t \in \mathcal{T}_i} y_i(t) \le c_i x_i, \quad \forall i \in \mathcal{I}, && \text{(2a)} \\
& y_i(t) \le k_i x_i, \quad \forall i \in \mathcal{I}, t \in \mathcal{T}_i, && \text{(2b)} \\
& \sum_{i \in \mathcal{I} : t \in \mathcal{T}_i} y_i(t) \ge d_t, \quad \forall t \in \mathcal{T}, && \text{(2c)} \\
\text{vars.} \quad & x_i \in \{0, 1\}, \quad \forall i \in \mathcal{I}, \\
& y_i(t) \ge 0, \quad \forall i \in \mathcal{I}, t \in \mathcal{T}_i,
\end{aligned}$$

where optimization variables are $x_i$ and $y_i(t)$. $x_i = 1$, if source $i$ is selected, 0, otherwise. In addition, $y_i(t)$ denotes the amount of energy that is covered by source $i$ at time $t$, by decreasing its charging or discharging. Constraint (2a) is the long-term capacity (packing) constraint of the sources. Constraint (2b) is about single-slot capacity of sources that says provided that source $i$ is selected, the maximum amount of decrease in demand at each slot is limited to $k_i$. Constraint (2c) is the covering constraint that guarantees that total acquired energy by the chosen sources covers the shortage at each slot. Finally, note that the S3P is a mixed-integer linear program which is difficult to solve, in general, even in offline setting.

Theorem 2.1. *Problem S3P is NP-complete.*

Proof. By setting $T = 1$, $u_i = 0, i \in \mathcal{I}$, the problem is the well-known minimum knapsack problem [3] which is one of the original NP-complete problems [17]. □

We highlight that a simplified version of the S3P can be translated to the capacitated facility location problem (CFLP) [27]. In the CFLP two sets of *facilities* and *clients* are given. Each facility has an *opening* cost and *capacity*. There is an *assignment* cost of assigning each client to each facility. The problem asks us to open a subset of facilities and assign the clients to these facilities, while ensuring that the capacity constraint of all facilities is respected. The objective is then to minimize the aggregated fixed and assignment costs. In particular, by contemplating sources as the facilities and time slots as the clients, and neglecting the short-term capacity constraint (2b), i.e., $k_i = c_i, \forall i \in \mathcal{I}$, and assuming complete availability of sources, i.e., $\mathcal{T}_i = \mathcal{T}, \forall i \in \mathcal{I}$, the S3P is the *splittable* version of the CFLP where a client can be partially assigned to multiple facilities.

By this mapping, the simplified version S3P inherits all the difficulties of the splittable CFLP, i.e., combinatorial nature due to 0/1 selection variable, and mixed packing and covering constraints in a single problem. In addition, the general S3P comes with two additional unique challenges: (i) single-slot capacity constraint (2b), which could be translated into the maximum assignment of each facility to clients in the CFLP context, and (ii) interval availability that could be translated into a "topological" constraint that each facility can be served just for a subset of clients. Putting together these issues, most of the previous results for the CFLP [1, 18, 21] cannot be directly applied to solve the S3P problem.

On the other hand, the S3P requires online solution design. The online inputs in our problem are two-fold. First, supply shortage $d_t$ is revealed in slot-by-slot fashion, which is natural in microgrid due to high uncertainty in the renewable output. Second, the sources arrive online. Recall that each source $i$ has interval availability $\mathcal{T}_i = [a_i, b_i]$, where $a_i$ is the arrival time slot, hence, its data is revealed at $t = a_i$. In this way, all the characteristics (cost, capacity, and departure time) of *available* sources are given to the MGCC at the beginning of each time slot. In terms of underlying optimization problem, both packing and covering constraints arrive online. Now, we proceed to design an online solution for the S3P.

# 3 ONLINE SOLUTION

Since the S3P encounters mixed packing and covering constraints, in online scenario it is inevitable that either packing or covering constraint is violated. In demand response, however, it is critical that the shortage is fulfilled by the chosen sources. Hence, in our online algorithm design, we force the covering constraint to be respected, and as a result, violation of capacity constraints of the sources is permitted. As such, MGCC aims to minimize the capacity violation of selected sources, in addition to total cost minimization.

As characterized in Eq. (1), the capacity of each source $i$ is related to the desired state of the charge $\mathrm{SoC}_i^{\mathrm{d}}$ at departure. Capacity violation in this context means the state of the charge is lower than the desired value. One can imagine several ways to compensate for this violation. The first solution is to make additional higher payment to the sources beyond their capacity. Another solution is to acquire the energy beyond the capacity from the main grid, or on-site back-up generator owned by the microgrid, which are usually much more expensive than using the crowd-sourced storage systems. Finally, note that in Sec. 3.3, we propose heuristics to minimize the capacity violation as much as possible.

In this section, we devise three algorithms in order. *First,* at each slot, we obtain a fractional algorithm called OnFrc for the S3P without single-slot capacity constraint (2b) (Sec. 3.1). The OnFrc is built upon a recently proposed framework for online mixed packing and covering problems [2]. *Second*, through a randomized rounding algorithm called OnInt we obtain an integer solution (Sec. 3.2). In Sec. 3.3, we propose several heuristics to improve the performance of both OnFrc and OnInt. Finally in Sec. 3.4, we extend the OnInt to the case that respects the single-slot capacity constraint (2b), as well.

## 3.1 Online Competitive Fractional Algorithm

*First*, in Sec. 3.1.1 we reformulate the relaxed linear problem and perform some scaling procedures such that the new problem is equivalent to the (relaxed) original S3P (without single-slot capacity constraint (2b)) and is more convenient for our analysis. *Second*, in Sec. 3.1.2, we propose the fractional algorithm OnFrc that fractionally chooses and schedules the available sources at the beginning of each time slot. We also analyze the bi-criteria competitive ratio of the proposed algorithm and find bounds for both the cost ratio and packing constraint violation.

*3.1.1 Linear Relaxation Problem Formulation.* We assume that the number of time horizon ($T$) is known to the MGCC in advance and also the optimal offline cost OPT is given.[5] Without loss of generality, we assume that $f_i \leq \text{OPT}, \forall i \in \mathcal{I}$, otherwise, we exclude the sources with fixed cost greater than OPT. Now, we introduce $\hat{f}_i$ as the scaled fixed-cost of source $i$ as

$$\hat{f}_i = \max\left\{\frac{f_i n}{\text{OPT}}, 1\right\}, \tag{3}$$

and $\hat{u}_i(t)$ as the *time-dependent* normalized unit cost of the source $i$ at time $t$ as

$$\hat{u}_i(t) = \frac{u_i d_t n}{\text{OPT}}.$$

Finally, we define $d_i(t) = d_t/c_i$. Note that by multiplying fixed and unit cost parameters by $n/\text{OPT}$, the optimal value of the problem changes to $n$. In addition setting the minimum fixed cost of the sources to 1 will increase the optimal cost to at most $2n$. The goal of these scaling procedures is to facilitate our competitive analysis.

Now, for the optimization variables, let $x_i \geq 0$ be the relaxed integer source selection variable. Moreover, $z_i(t) \in [0, 1]$ is the portion of supply shortage $d_t$ that is fulfilled by source $i$, provided that $t \in \mathcal{T}_i$. Indeed, $y_i(t) = d_t z_i(t)$, where $y_i(t)$ is the scheduling variable in the S3P in Sec. 2. With these modifications and definitions, we formulate the following linear-relaxed problem:

$$\begin{aligned}
\text{S3P-LP}: \min \quad & \sum_{i\in\mathcal{I}} \hat{f}_i x_i + \sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}_i} \hat{u}_i(t) z_i(t) \\
\text{s.t.} \quad & \sum_{t\in\mathcal{T}_i} d_i(t) z_i(t) \leq x_i, \quad \forall i \in \mathcal{I}, && \text{(4a)} \\
& \sum_{i\in\mathcal{I}: t\in\mathcal{T}_i} z_i(t) \geq 1, \quad \forall t \in \mathcal{T}, && \text{(4b)} \\
\text{vars.} \quad & x_i \geq 0, \quad \forall i \in \mathcal{I}, \\
& z_i(t) \geq 0 \quad \forall i \in \mathcal{I}, t \in \mathcal{T}_i.
\end{aligned}$$

We note that in the S3P-LP in addition to linear relaxation, the single-slot capacity constrain (2b) is also neglected. In Sec. 3.4, we explain how to modify the algorithms to consider this constraint.

*3.1.2 Online Fractional Algorithm.* **Virtual cost of sources.** Our online competitive algorithm accomplishes source selection and scheduling by providing an ascending ordering among the available sources at each slot. In this way, the main aim is to construct a metric to be used for sorting of the sources. The sorting metric $v_i(t)$ which we refer to it as *virtual cost* of source $i$ at time $t$ is defined as follows.

**Algorithm 1:** OnFrc- Online Fractional Algorithm, at time $t$

1 **Initialization**
2 $\mathcal{I}_t \leftarrow$ an ordering of sources that are available in $t$ in ascending order of $v_i(t)$ in Eq. (5)
3 $\mathcal{P}_t \leftarrow$ the maximal subset of $\mathcal{I}_t$ such that $\sum_{i\in\mathcal{P}_t} x_i < 1$
4 $l_t \leftarrow$ the first source in $\mathcal{I}_t$ that is not in $\mathcal{P}_t$
5 **while** $\sum_{i\in\mathcal{P}_t\cup\{l_t\}: t\in\mathcal{T}_i} z_i(t) < 1$ **do**
6     **foreach** $i \in \mathcal{P}_t \cup \{l_t\}$ **do**
7         **if** $i \in \mathcal{P}_t$ ***OR*** $(i = l_t$ ***AND*** $x_i < 1)$ **then**
8             $x_i \leftarrow \min\left\{x_i + x_i/\hat{f}_i T, 1\right\}$
9             $z_i(t) \leftarrow \min\left\{2x_i, \frac{x_i - 1/n}{d_i(t) + \hat{u}_i(t)/\hat{f}_i}\right\}$
10         **end**
11         **if** $i = l_t$ ***AND*** $x_i = 1$ **then**
12             $z_i(t) \leftarrow z_i(t) + 1/v_i(t)T$
13         **end**
14     **end**
15 **end**

*Definition 3.1.* The virtual cost of source $i$ at time $t$ is defined as:

$$v_i(t) = \begin{cases} \hat{f}_i \theta^{\gamma_i(t)-1} d_i(t) + \hat{u}_i(t), & \text{if } x_i = 1, \\ \hat{f}_i d_i(t) + \hat{u}_i(t), & \text{otherwise}, \end{cases} \tag{5}$$

where $\theta > 1$ is a constant factor and $\gamma_i(t) = \sum_{\tau\in\mathcal{T}_i: \tau\leq t} d_i(\tau) z_i(\tau)$ is the current congestion level of source $i$.

By this definition, when source $i$ is not fully chosen ($x_i < 1$), the cost is linear in both normalized fixed and unit cost. On the other hand, if source $i$ is already fully chosen ($x_i = 1$) in the fractional solution due to the scheduling in the previous time slots, the virtual cost is linear in fixed cost, however exponential in the congestion level $\gamma_i(t)$. In this way, there is an exponential penalty in capacity violation of the sources, i.e., respecting capacity constraint is more important than cost minimization. Parameter $\theta > 1$ can be interpreted as the design parameter to provide trade-off between total cost against capacity violation, i.e., the larger the value of $\theta$, the higher the importance of congestion level in virtual cost. In Sec. 4, we investigate the impact of $\theta$ on the performance of online algorithms using experiments.

**Online fractional algorithm.** We now propose OnFrc that runs at the beginning of each slot and its goal is to cover the shortage $d_t$. Note that at $t = 1$ we initialize $x_i = 1/n$, so we get $\sum_{i\in\mathcal{I}} x_i = 1$ at initialization.

The detailed description of the proposed online algorithm OnFrc that summarized in Alg. 1 is as follows. First, we sort the available sources at slot $t$ in non-decreasing order of their virtual cost (Line 2). Then, by picking the most cost effective ones with aggregated selection variable less than 1, we construct set $\mathcal{P}_t$ (Line 3). Additionally, let $l_t$ be the first source that is not in $\mathcal{P}_t$, but in $\mathcal{I}_t$. Note that since by initialization we get $\sum_{i\in\mathcal{I}} x_i = 1$, $l_t$ is always non-empty. In last step, we cover demand $d_t$ in an iterative procedure (Lines 5-15) by a scheduling among the sources in $\mathcal{P}_t \cup \{l_t\}$. The scheduling is as

[5] These assumptions are reasonable since $T$ is fixed usually. The optimal offline cost also can be estimated based on historical data. Nevertheless, the algorithm can be extended to the case that the optimal offline cost is not known at the expense of adding a multiplicative logarithmic order in competitive ratio [2].

follows. If the current source is not fully chosen (either in previous time slots or in the previous iterations of the current time slots), it *sets* its $z_i(t)$ according to Line 9. Otherwise, it *increases* $z_i(t)$ by $1/v_i(t)T$ (Line 12). This scheduling procedure continues until $d_t$ is covered. The following theorem characterizes the competitive ratio of the OnFrc.

THEOREM 3.2. *Given* $1 < \theta < 1.5$*, OnFrc generates a fractional schedule that is* $O(n \log n, \log n)$*-competitive, where n is the number of sources.*

See Appendix A for the proof. This means that the cost of OnFrc is at most $O(n \log n)$ times than the offline optimum, while the packing violation is no more than $O(\log n)$.

## 3.2 Randomized Rounding Algorithm

In this section, we devise an online randomized rounding algorithm called OnInt to obtain an integral solution from the fractional algorithm OnFrc to the original problem.

The proposed randomized rounding algorithm OnInt is summarized in Alg. 2. Our randomized rounding algorithm must produce integral values for $x_i$s at any time slot $t$. First, we notice that due to executing the rounding algorithm in the previous slots, some values of $x_i$ are already set to 1, i.e., the corresponding sources are chosen. Hence, our algorithm first construct the chosen available sources $\mathcal{I}_t^{\text{sel}}$ at the previous time slots (Line 5). Then, in the main body of the algorithm (Lines 7-15), among the set of sources that are fractionally chosen ($x_i < 1$), and actively participate in the current time slot ($x_i(t) > x_i(t-1)$), it randomly set them to 1 with probability $x_i(t)$. Due to rounding, some sources that already contribute in the OnFrc to cover a portion of supply shortage might be rounded to 0. We accumulate the covering amount of those sources in Line 12 and then in Lines 16-18, equally schedule this amount between all the fully chosen sources.

We note that the OnInt comes with an additional performance loss (for both integral cost and capacity violation) due to rounding fractional solution to integral solution. Our experiments shows that this performance gap is not significant. We refer to Sec. 4 for details. Analytical study of the worst-case bound due to randomized rounding is part of the future work.

Finally, we remark that the algorithms OnFrc and OnInt are obtained by neglecting the single-slot capacity constraint (2b). We extend the OnInt in Sec. 3.4 and propose another algorithm to respect this constraint. The overall idea is to restrict $y_i(t)$ to at most the maximum discharge rate $k_i$ for all selected sources. By doing so, we encounter an additional shortage in the covering constraint due to respecting the maximum discharge rate limit. We compensate for this shortage by adding more sources to the set of selected sources.

## 3.3 Heuristics to Improve the Algorithms

In this section, we propose several other heuristics to improve the performance of both fractional algorithm OnFrc and integral algorithm OnInt in practice. Our focus in this section is to mainly tailor the algorithms to be more conservative in violating the capacity constraint. The heuristic in Sec. 3.3.1 modifies the OnFrc, and the heuristics in Sec.3.3.2 modifies the OnInt.

**Algorithm 2:** OnInt- Online Randomized Rounding Algorithm, at time $t$

1 **Initialization**
2 $p_i \leftarrow \text{RAND}(0,1), \forall i \in \mathcal{I}$ `// A random number picked uniformly from [0,1]`
3 $x_i(t) \leftarrow$ the value of $x_i$ by executing the OnFrc at $t$
4 $x_i(t-1) \leftarrow$ the value of $x_i$ by executing the OnFrc at $t-1$
5 $\mathcal{I}_t^{\text{sel}} \leftarrow \{i \in \mathcal{I} : x_i = 1, t \in \mathcal{T}_i\}$ `// The set of available sources that are used in the previous time slots.`
6 $y^{\text{nc}} \leftarrow 0$`// the uncovered demand by rounding to 0`
7 **foreach** $i \in \mathcal{I}$ **do**
8     **if** $x_i < 1$ *AND* $x_i(t) > p_i$ *AND* $x_i(t) > x_i(t-1)$ **then**
9         $x_i \leftarrow 1$
10         $\mathcal{I}_t^{\text{sel}} \leftarrow \mathcal{I}_t^{\text{sel}} \cup \{i\}$
11     **else**
12         $y^{\text{nc}} \leftarrow y^{\text{nc}} + d_t z_i(t)$
13         $z_i(t) \leftarrow 0$
14     **end**
15 **end**
`// scheduling the uncovered shortage due to rounding`
16 **foreach** $i \in \mathcal{I}_t^{sel}$ **do**
17     $y_i(t) \leftarrow y_i(t) + y^{\text{nc}}/|\mathcal{I}_t^{\text{sel}}|$
18 **end**

*3.3.1 Heuristics Applied to the* OnFrc . In this heuristic, we change the update equation for $z_i(t)$ in Line 9 of the OnFrc to

$$z_i(t) = \min\left\{2x_i, \frac{x_i - 1/n}{d_i(t) + \hat{u}_i(t)/\hat{f}_i}, \left[\frac{c_i - \sum_{\tau=a_i}^{t-1} d(\tau) z_i(\tau)}{d(t)}\right]^+\right\}.$$

As compared to the update equation in Line 9 of the OnFrc, the update equation above has an additional third term to enforce respecting capacity constraint of each source. Similarly, we update the update equation in Line 12 of the OnFrc. In addition, by the end of running the OnFrc at slot $t$, we set $x_i = 1$, for sources that their total congestion exceeds the capacity, i.e., $\{x_i = 1 : c_i \leq \sum_{\tau=a_i}^{t} y_i(t)\}$. In this way, for the forthcoming slots, we calculate the virtual cost of these sources according to the first line in Eq. (5), in which there is an exponential penalty for further usage of these sources. Last but not the least, we break the update equations in the body of for loop in Lines 6-14 of the OnFrc immediately after fulfilling the supply shortage, without going over the remaining sources. In this way, we prevent over-coverage of the supply shortage.

*3.3.2 Heuristics Applied to the* OnInt . In this section, we propose three additional heuristics for the OnInt.

**(1) Time-aware Randomized Rounding.** The main intuition behind this heuristic is to select sources more aggressively at the initial slots, and as time goes ahead, select the new sources more conservatively. More specifically, at the initial slots, usually the number of selected sources are not large enough to cover the demand that may lead to higher capacity violation. Hence, we tune our rounding to pick the sources more aggressively. On the other

**Algorithm 3:** kOnInt- Online Integral Algorithm with Maximum Discharge Rate, at time $t$

1 **Initialization**
2 $\mathcal{I}_t^{\mathsf{sel}} \leftarrow$ the set of selected sources by executing OnInt at slot $t$
3 $\mathcal{I}_t^{\mathsf{unsel}} \leftarrow \left\{ i \in \mathcal{I} : i \notin \mathcal{I}_t^{\mathsf{sel}} \text{ and } t \in \mathcal{T}_i \right\}$
4 $y^{\mathsf{bk}} \leftarrow 0$`// the aggregated demand beyond the maximum discharge rate`
5 **foreach** $i \in \mathcal{I}_t^{\mathsf{sel}}$ **do**
6 $\quad y^{\mathsf{bk}} \leftarrow y^{\mathsf{bk}} + [y_i(t) - k_i]^+$
7 $\quad y_i(t) \leftarrow \min \left\{ k_i, y_i(t) \right\}$
8 **end**
9 sort the sources in $\mathcal{I}_t^{\mathsf{unsel}}$ in descending order of $x_i$ values obtained by the OnFrc at slot $t$
10 **while** $y^{\mathsf{bk}} > 0$ **do**
11 $\quad i \leftarrow$ the next source in sorted set of $\mathcal{I}_t^{\mathsf{unsel}}$
12 $\quad y_i(t) \leftarrow \min \left\{ k_i, c_i - \sum_{\tau=a_i}^{t-1} y_i(\tau), y^{\mathsf{bk}} \right\}$
13 $\quad y^{\mathsf{bk}} \leftarrow y^{\mathsf{bk}} - y_i(t)$
14 $\quad x_i \leftarrow 1$
15 $\quad \mathcal{I}_t^{\mathsf{sel}} \leftarrow \mathcal{I}_t^{\mathsf{sel}} \cup \{i\}$
16 **end**

hand, when enough number of slots have been passed, potentially sufficient number of sources are already selected in the previous slots, hence, we select new sources more conservatively. Toward this, we modify the random generation in Line 2 of the OnInt as an increasing function of time, e.g., $p_i = \beta(t \times \mathsf{RAND}(0, 1))/T$, where $\beta$ is a constant factor. In this way, the sources are selected in Line 8 more conservatively with large values of $t$.

**(2) Congestion-aware Source Selection.** Another heuristic runs after the source selection phase in Lines 7-15 of the OnInt. The high-level idea is to measure the current congestion of the selected sources and if a pre-determined portion of them are fully congested, select a new source. In our experiments, at each slot we select a new source given that at least 25% of selected sources are fully-congested. The new selected source is the one with the largest $x_i$ in the OnFrc that (i) already is not chosen, and (ii) is available at the current slot.

**(3) Water Filling-based Uncovered Demand Scheduling.** The last heuristic changes the re-scheduling approach of the OnInt in Lines 16-18. The high-level idea is to fulfill the uncovered demand $y^{\mathsf{nc}}$ following a water-filling approach. Toward this, we sort the selected sources according to their unused capacity, and start covering $y^{\mathsf{nc}}$ with the one with the highest unused capacity, and proceed to the others accordingly.

## 3.4 Extension to the Maximum Discharge Rate

In this section, we extend the OnInt to consider the case that the maximum discharge rate is limited, i.e., $y_i(t) \leq k_i, \forall i, t$. In other words, in this setting the single-slot capacity constraint in Eq. (2b) of the S3P is taken into account. The procedure of the algorithm is summarized in Algorithm 3 and called the kOnInt. The algorithm works in two successive steps.

In the first step as listed in Lines 5-8, we add the surplus amount, i.e., $[y_i(t) - k_i]^+$, to $y^{\mathsf{bk}}$ which contains the aggregate amount of shortage that is beyond the maximum discharge rate of the selected sources. Note that $[a]^+ = \max\{0, a\}$. Then, we project the deduction amount of the selected sources to at most $k_i$, i.e., $y_i(t) = \min \left\{ k_i, y_i(t) \right\}$.

In the second step, we select more sources in the set of sources that are available at slot $t$ and are not selected by executing the OnInt ($\mathcal{I}_t^{\mathsf{unsel}}$ in Line 3). Our selection criteria is to sort the unselected available sources in descending order of their highest fractional values $x_i$ in algorithm the OnFrc (as shown in Line 9). Recall that according to the OnFrc, the higher the value of $x_i$, the lower the cost of the corresponding source. Then, in an iterative procedure, we select the sources in the ordered set one-by-one and set the $y_i(t)$ values according to Line 12, which is the minimum amount between the maximum discharge rate and the residual capacity of the current selected source, and the remaining uncovered shortage. We proceed this procedure until the uncovered shortage is fulfilled. We finally note that in the kOnInt, we assume that the sufficient sources to cover the supply shortage.

In Sec. 4.2.5, we investigate the performance of the kOnInt as a function of the ratio $\rho_i = k_i/c_i$, where lower value of $\rho_i$ corresponds that the maximum discharge rate restricts the scheduling, while $\rho_i = 1$ relaxes the maximum discharge rate, i.e., it is possible to consume all the capacity of any selected sources at single slot.

# 4 PERFORMANCE EVALUATIONS

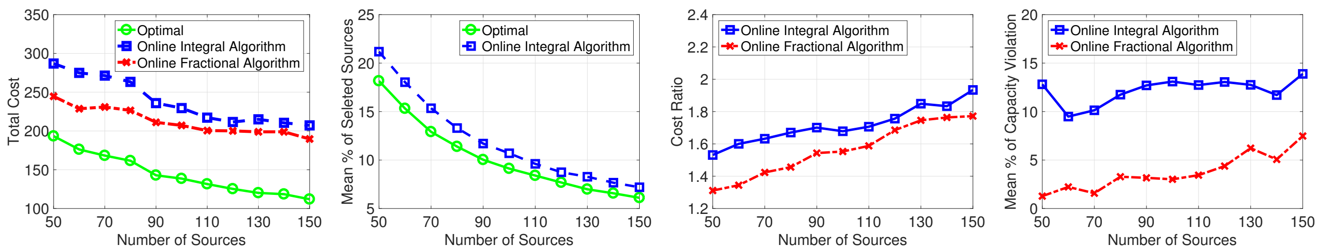
In this section, we use real-world date traces to evaluate the performance of the online fractional and integral algorithms OnFrc and OnInt and the extended algorithm kOnInt in different scenarios.

## 4.1 Experimental Setup and Overview

*4.1.1 Parameter Settings and Data Traces.* The electricity data are obtained from [7] which is the demand of a college in California. To inject the uncertainty in demand, the renewable energy supply is injected by a wind power trace from [14] which is the output of a wind station in California with installed capacity of 12MW. Finally, we assume that on average %10 of the demand is regarded as supply shortage in each slot.

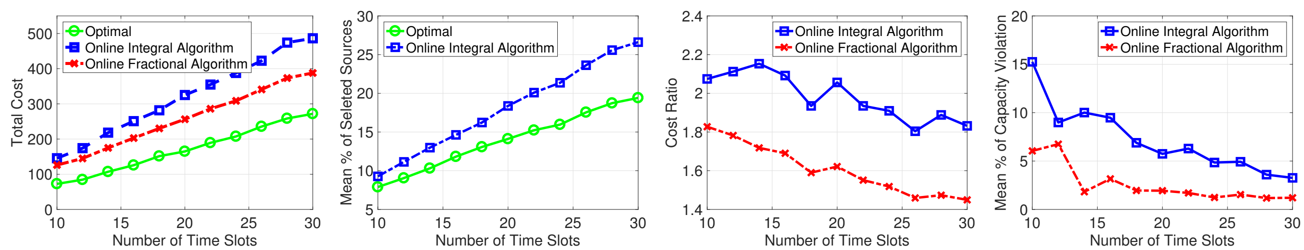
Unless otherwise specified, the unit cost for each source follows a uniform distribution over [\$0, \$1]. The fixed cost is chosen in order of ×20 of the unit costs, which is roughly around 1/3 of the volume cost. The available capacity $c_i$ is randomly generated in [10, 70]kWh which includes typical sizes for Tesla EV [25] and SolarCity Powerwall batteries (10kWh for approximately 1 bedroom home, and 70kWh for approximately 6 bedrooms home). We set $T = 12$ and the length of each slot to 1 hour, and randomly generate the available interval $\mathcal{T}_j$ for sources. The value of congestion parameter $\theta$ in the OnFrc is set to 1.2.

*4.1.2 Performance Metrics.* We compare the result of both online fractional (OnFrc) and integral algorithms (OnInt) to the offline optimum which is calculated by Gurobi solver [9]. The difference between the result of the OnFrc and the OnInt is the integrality gap, i.e., the performance loss due to the randomized rounding to integral solution.




(a) (b) (c) (d)

**Figure 2: Impact of number of sources**




(a) (b) (c) (d)

**Figure 3: Impact of number of slots**

In our experiments, we report four metrics: (i) total cost, we report this value for offline optimum, the OnFrc, and the OnInt; (ii) average percentage of capacity violation for the OnFrc and OnInt; note that offline optimum finds the feasible solution without capacity violation so in corresponding figures, there is no capacity violation for the offline optimum. (iii) empirical cost ratio which is defined as the ratio between obtained cost of our algorithms over the optimum. This metric is reported for both OnFrc and OnInt; and finally (iv) the average percentage of selected sources for offline optimum and the OnInt. Note that this measure is not reported for the OnFrc because it partially selects the sources.

*4.1.3 Experimental Scenarios.* We report the above performance metrics under different scenarios. In particular, we consider four different scenarios and investigate the impact of (i) the number of sources; (ii) the number of time slots; (iii) the capacity of sources; and finally (iv) the congestion parameter $\theta$ in OnFrc algorithm. Note that each data point of the figures demonstrates average value of 100 runs, each of which is a different randomly generated scenario.

## 4.2 Results of the Online Algorithm

*4.2.1 Impacts of the Number of Sources.* In this experiment, we change the number of users from 50 to 150 with step 10. The results are shown in Fig. 2 and we report the following observations:

(i) As the number of sources increases, total cost (Fig. 2(a)) and average percentage of selected sources decreases (Fig. 2(b)). Both are reasonable since with the increase in the number of sources, there is more freedom to pick cost-effective sources. In addition, since the supply shortage is fixed in this scenario, the average percentage of selected sources decreases.

(ii) The average cost ratios for the fractional algorithm OnFrc and the integral algorithm OnInt are 1.56 and 1.71, respectively (Fig. 2(c)) which demonstrate sound performance of our algorithms. In addition, the performance loss in terms of total cost due to randomized rounding is 9.6%, on average.

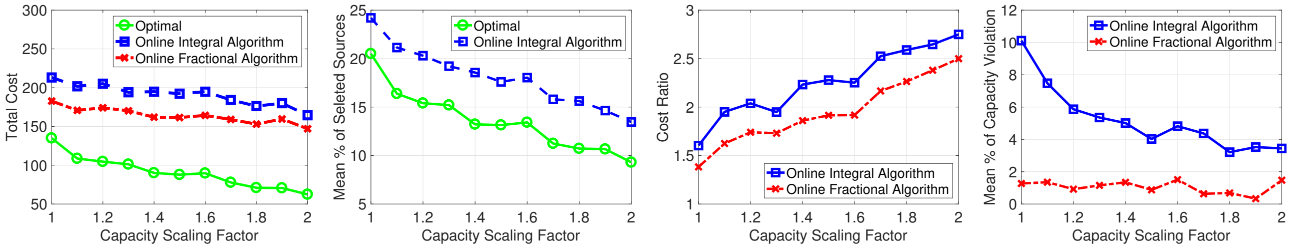
(iii)Another observation is that as the number of sources increases the cost ratio also increases (from 1.53 when $n = 50$ to 1.93 when $n = 150$ in OnInt). This justifies the competitive ratio analysis since it increases as the number of sources increases. Finally, we note that the obtained empirical cost ratios demonstrate that our algorithms can achieve much better results than those obtained in theoretical analysis.

(iv) The average capacity violation in OnFrc and OnInt are 3.7% and 12.2%, respectively (Fig. 2(d)). The latter number says that on average OnInt goes 12.2% beyond the announced capacity of selected sources. As mentioned in Sec. 3, this capacity violation could be compensated by using external sources such as on-site microgrid generation or acquiring the energy from main grid. Another approach, perhaps, is to intelligently scale down the capacities of the sources, such that the violation amount in scaled-down version of the problem is roughly equal to the amount that is reserved due to scaling down the capacity.

*4.2.2 Impacts of the Number of Time Slots.* In this scenario, the number of time slots changes from 10 to 30 with step 2. The results are shown in Fig. 3 and we report the following observations:

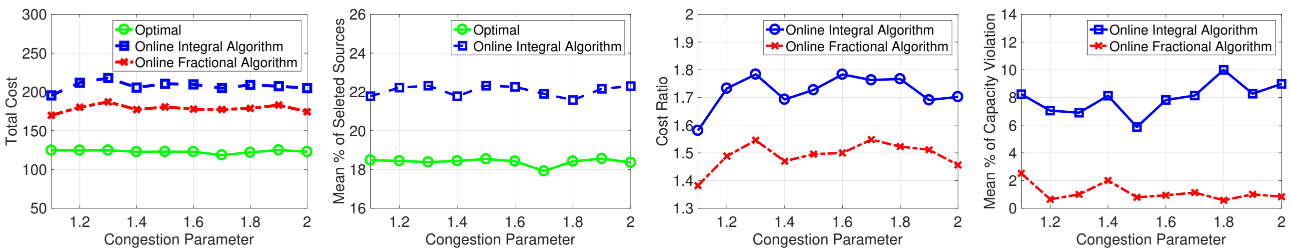
(i) As shown in Figs. 3(a) and 3(b), as the number of time slots increases, total cost and the average percentage of selected sources increase for the offline optimum and also for our online algorithms. This is reasonable since with fixed number of sources as the number of slots increases, more demand must be covered and hence total cost and the number of selected sources increase.

(ii) The second interesting observation is shown in Fig. 3(c), where as the number of slot increases, the cost ratio decreases (from 2.11 when $T = 10$ to 1.83 when $T = 30$ for OnInt). The




(a) (b) (c) (d)

**Figure 4: Impact of capacity scaling**




(a) (b) (c) (d)

**Figure 5: Impact of congestion parameter $\theta$**

justification is that as the number of time slot increases the unit (volume) cost dominates the fixed cost, so, the cost ratio decreases. Recall that fixed-cost and time availability are two main issues that make the online decision making important and not-trivial. And, as time horizon increases, it means that the importance of fixed cost becomes lower, which lead to lower cost ratio.

(iii) The last observation in Fig. 3(d) shows that with the increase in the number of slots, the capacity violation decreases, on average (from 15.25% to 3.26%). The reason is that with the increase in slots, more sources are selected (see Fig. 3(b)), hence, there is more room to schedule the sources without capacity violation.

*4.2.3 Impacts of the Capacity of Sources.* In this experiment, we scale the capacity of the sources from ×1 to ×2 of the original values, and plot the results in Fig. 4. We report the following observations:

(i) As shown in Fig. 4(a), as the capacity scales, total cost decreases since each source can cover more supply shortage. Apparently, the percentage of selected sources also decreases with the same reason (Fig. 4(b)).

(ii) As the results show in Fig. 4(c), the empirical cost ratio, increases with scaling up the capacity, which is counter-intuitive. This is due to the fact that when we increase the capacity of the sources, optimal offline have more flexibility in scheduling to reduce total cost. This observation, indeed, calls for further investigation on providing more intelligent scheduling policies for the online scenarios.

(iii) As expected in Fig. 4(d), the capacity violation reduces significantly for OnInt with scaling the capacity (from 10.1% to 3.4%), since there are enough capacity to fulfill the shortage by the selected set of sources.

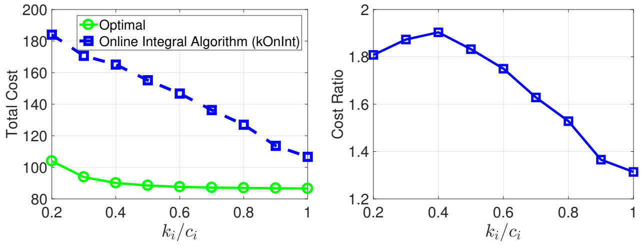
*4.2.4 Impacts of the Congestion Parameter $\theta$.* An important parameter in online algorithm OnFrc is the congestion parameter $\theta$ which is used in Eq. (5) to incorporate the importance of capacity violation on the online algorithm. In general, the higher the value of $\theta$, the more the importance of respecting the capacity constraint. Toward investigating the impacts of $\theta$, we change the value of $\theta$ from 1.1 to 2 with step 0.1. The results are shown in Fig. 5.

(i) The results in Figs. 5(a) and 5(b) demonstrate almost smooth values for total cost and percentage of selected sources, which are reasonable due to no change on the parameters that can potentially impact these values.

(ii) The results in Fig. 5(c) demonstrate a gentle increase in cost ratios for both OnFrc and OnInt. Ideally, we would expect increase in cost ratio when the congestion parameter $\theta$ increases, since by doing so, we lean toward respecting violation as compared to reducing the cost. However, the performance of OnFrc is already promising in respecting the capacity constraints (the capacity violation of OnFrc is always less than 2% in this scenario as shown in Fig. 5(d)). Hence, the increase in cost ratio is not significant (from 1.38 to 1.51 in OnFrc and from 1.58 to 1.76 in OnInt).

(iii) On the other hand, results for the online fractional algorithm OnFrc in Fig. 5(d) demonstrate that capacity violation drops with increase in congestion parameter (from 2.52% to 0.57%). The capacity violation behavior for OnInt is ad-hoc in Fig. 5(d), which mainly demonstrate that the capacity violation due to fulfilling the uncovered demand of rounding procedure (as shown is Lines 16-18 of OnInt algorithm) dominates the capacity violation due to OnFrc.

*4.2.5 The Performance of the Algorithm kOnInt .* In this section, we evaluate the algorithm proposed in Sec. 3.4 that takes into account the maximum discharge rate of each source at each slot. Toward this, we change the ratio $\rho_i = k_i/c_i$ as an indicator on how much the maximum discharge rate is restrictive. For example, $\rho_i = 0.1$ mean that at each slot, we can use at most 0.1 of the




(a) (b)

**Figure 6: Impact of the ratio $k_i/c_i$**

capacity of source $i$. On the other extreme, $\rho_i = 1$ means that there is no maximum discharge limit and it is possible to use the entire capacity in one slot. In this experiment, we change the value of $\rho_i$ form 0.1 to 1 with step 0.1, and the results are shown in Fig. 6.

The first observation in Fig. 6(a) is that as $\rho_i$ increases, total cost of both optimal solution (from 104 to 87) and the kOnInt (from 184 to 106) decreases. The reason is that with the increase in $\rho_i$ there is more flexibility in scheduling that leads to lower cost. The second observation in Fig. 6(b) demonstrate that the empirical cost ratio decreases as $\rho_i$ increases. The reason is that with lower values of $\rho_i$, the scheduling is more complicated and the optimal can find it, while our algorithm kOnInt is not able to find the proper scheduling, and instead selects more sources which increases the cost.

## 5 RELATED WORK

S3P is mainly related to CFLP [21] and fixed-charge transportation problem [19]. While all challenges in these two problems exist in our problem, there are some unique challenges raised by special constraint in our problem. There are some existing offline results for CFLP [18, 21] and fixed-charge transportation problem [26], but, none of them provide constant approximation ratios for the general case. Recently, An *et al.* in [1] proposed a 288-approximation offline algorithm for CFLP that is, to the best of our knowledge, the first constant factor algorithm for CFLP based on LP relaxation. The result in [1] cannot be directly applied to S3P because of the unique challenges in our problem, mainly interval availability. In addition, the result in [1] works for the offline scenarios, while the problem of study in this paper emphasizes online algorithm design.

The second category of related problems in the literature is called interval cover [4, 5, 8], which is the time-expanded version of the minimum knapsack *without capacity constraints, and unit cost taking into account.* The capacity constraint in S3P turns the problem into a mixed packing and covering one which is fundamentally different and more challenging than the interval cover as a covering problem [2]. The most promising result of interval cover problem has been presented in [4], that is a 4-approximation offline algorithm.

The last category is the problem that can be contemplated as the S3P with a fixed scheduling as input. This problem has appeared in different application scenarios such as device-to-device load balancing in cellular networks [11] and client-assisted cloud storage systems [30]. The setting considers a time-decoupled problem where at each slot a sub-problem must be solved by just selecting the winning sources. In this way, the design space by proper scheduling is overlooked. This apparently leads to a sub-optimal scenario. In addition, equal scheduling may lead to infeasible cases for scheduling. For example in the simple scenario of Fig. 1, the only available source at the last slot is source 3. The total capacity of this source is 18 and since this source is available at 9 slots, by fixed scheduling, there would be 18/9 = 2 units available for each slot. On the other hand, at the last slot in which source 3 is the only available source, the amount of supply shortage is 3, hence, it is not possible to cover this shortage with fixed scheduling of 2 units at each slot. This example shows that scheduling is important role in our problem.

We also note that several related problems with different solution approaches have been studied in the literature. We refer to [13] with the Lyapunov optimization and [29] with Markov decision process as examples. Overall, these approaches relies on the stochastic process of the inputs. Instead, in this paper, we follow competitive design approach, in which there is no assumptions on the stochastic process of the unknown inputs. As in competitive analysis there is no assumptions on the stochastic modeling of future input, the online algorithm tries to compete against an adversarial input. In competitive design the goal is to compete against the adversarial input, hence, the competitive algorithm might be conservative, and it cannot provide satisfactory results in practical scenarios in some cases. On the other hand, stochastic optimization approaches rely on the distribution of the input sequence. However, learning the potentially time-varying distribution in real inputs can be a formidable task. It is worth noting that not only our online algorithm in this paper can guarantee a bounded performance, but also, our experimental results demonstrate sound results of our algorithm.

Finally, we note that our basic online fractional algorithm OnFrc leverages the ideas in [2]. More specifically, we tailor the algorithm in [2] to our problem which is different from the one in [2] mainly because of different availability of sources. Our integral algorithm OnInt is entirely different from the approach in [2] and also as explained in Sec. 3.3, we propose several heuristics to improve the performance of our algorithms in practice.

## 6 CONCLUSION AND FUTURE WORK

In this paper, we advocated the idea of using the potentials of existing energy storage systems in a microgrid to accomplish crowdsources storage-assisted demand response. We formulate the joint problem of source selection and scheduling with the goal of minimizing the cost, while respecting mixed packing and covering constraints. We devised online algorithms that are built upon a recent results for online problems with mixed packing and covering constraints. We also analyzed the analytical performance of our online algorithms in bi-criteria competitive approach. The trace-driven experimental results demonstrated that the proposed algorithms work well in practice. Obtained analytical and experimental results open a number of further research directions. First, an interesting line is to study the problem through mechanism design to see whether the proposed algorithms are dominant strategy incentive compatible or not. Second, providing lower bound for fractional algorithm to characterize the fundamental price-of-uncertainty of the problem and characterizing the integrality gap due to randomized rounding are two important open questions.

## ACKNOWLEDGMENT

The work presented in this paper was supported in part by National Basic Research Program of China (Project No. 2013CB336700) and the University Grants Committee of the Hong Kong Special Administrative Region, China (Theme-based Research Scheme Project No. T23-407/13-N and Collaborative Research Fund No. C7036-15G).

## REFERENCES

[1] Hyung-Chan An, Monika Singh, and Ola Svensson. 2014. LP-Based Algorithms for Capacitated Facility Location. In *Proc. of IEEE FOCS*. 256–265.
[2] Yossi Azar, Umang Bhaskar, Lisa Fleischer, and Debmalya Panigrahi. 2013. Online mixed packing and covering. In *Proc. of ACM SODA*. 85–100.
[3] Tim Carnes and David Shmoys. 2008. Primal-dual schema for capacitated covering problems. In *Integer Programming and Combinatorial Optimization*. Springer, 288–302.
[4] Venkatesan T Chakaravarthy, Amit Kumar, Sambuddha Roy, and Yogish Sabharwal. 2011. Resource allocation for covering time varying demands. In *Algorithms–ESA*. Springer, 543–554.
[5] Venkatesan T Chakaravarthy, Gyana R Parija, Sambuddha Roy, Yogish Sabharwal, and Amit Kumar. 2011. Minimum cost resource allocation for meeting job requirements. In *Proc. of IEEE IPDPS*. 14–23.
[6] Solar City. 2017. Powerwall. http://www.solarcity.com/residential/powerwall. (2017).
[7] California Energy Commission. 2006. California Commercial End-Use Survey. http://capabilities.itron.com/CeusWeb/. (2006).
[8] Pankaj Dayama, Balakrishnan Narayanaswamy, Dinesh Garg, and Y Narahari. 2015. Truthful Interval Cover Mechanisms for Crowdsourcing Applications. In *Proc. of ACM AAMAS*. 1091–1099.
[9] Gurobi. 2016. LP Solver. http://www.gurobi.com/. (2016).
[10] Mohammad H. Hajiesmaili, Chi-Kin Chau, Minghua Chen, and Longbo Huang. 2016. Online microgrid energy generation scheduling revisited: The benefits of randomization and interval prediction. In *Proc. of ACM e-Energy*. ACM, 1–11.
[11] Mohammad H. Hajiesmaili, Lei Deng, Minghua Chen, and Zongpeng Li. 2017. Incentivizing Device-to-Device Load Balancing for Cellular Networks: An Online Auction Design. *IEEE Journal on Selected Areas in Communications* 35, 2 (2017), 265–279.
[12] AD Hawkes and MA Leach. 2009. Modelling high level system design and unit commitment for a microgrid. *Applied energy* 86, 7 (2009), 1253–1265.
[13] Longbo Huang, Jean Walrand, and Kannan Ramchandran. 2012. Optimal demand response with energy storage management. In *Proc. of IEEE SmartGridComm*. 61–66.
[14] Internet. 2016. National Renewable Energy Laboratory. http://wind.nrel.gov. (2016).
[15] Internet. 2016. Navigant Research. https://www.navigantresearch.com/research/market-data-microgrids. (2016).
[16] Internet. 2016. Vehicle to grid project. http://www.udel.edu/V2G/. (2016).
[17] Richard M Karp. 1972. *Reducibility among combinatorial problems*. Springer.
[18] Stavros G Kolliopoulos and Yannis Moysoglou. 2015. Integrality gaps for strengthened linear relaxations of capacitated facility location. *Mathematical Programming* (2015), 1–43.
[19] Krzysztof Kowalski and Benjamin Lev. 2008. On step fixed-charge transportation problem. *Omega* 36, 5 (2008), 913–917.
[20] Robert H Lasseter and Paolo Paigi. 2004. Microgrid: a conceptual solution. In *Proc. of IEEE PESC*, Vol. 6. 4285–4290.
[21] Retsef Levi, David B Shmoys, and Chaitanya Swamy. 2004. LP-based approximation algorithms for capacitated facility location. In *Integer Programming and Combinatorial Optimization*. Springer, 206–218.
[22] Zhenhua Liu, Adam Wierman, Yuan Chen, Benjamin Razon, and Niangjun Chen. 2013. Data center demand response: Avoiding the coincident peak via workload shifting and local generation. *Performance Evaluation* 70, 10 (2013), 770–791.
[23] Lian Lu, Jinlong Tu, Chi-Kin Chau, Minghua Chen, and Xiaojun Lin. 2013. Online Energy Generation Scheduling for Microgrids with Intermittent Energy Sources and Co-generation. In *Proc. of ACM SIGMETRICS*. 53–66.
[24] J.A. Peas Lopes, C.L. Moreira, and A.G. Madureira. 2006. Defining control strategies for MicroGrids islanded operation. *IEEE Trans. Power Systems* 21, 2 (2006), 916–924. DOI: http://dx.doi.org/10.1109/TPWRS.2006.873018
[25] Tesla. 2016. Tesla motors. https://www.teslamotors.com/. (2016).
[26] Mathieu Van Vyve. 2013. Fixed-charge transportation on a path: optimization, LP formulations and separation. *Mathematical Programming* 142, 1-2 (2013), 371–395.
[27] David P Williamson and David B Shmoys. 2011. *The design of approximation algorithms*. Cambridge University Press.
[28] Ying Zhang, Mohammad H. Hajiesmaili, and Minghua Chen. 2015. Peak-aware online economic dispatching for microgrids. In *Proc. of ACM e-Energy*. 27–36.
[29] Yu Zhang and Mihaela van der Schaar. 2014. Structure-aware stochastic storage management in smart grids. *IEEE Journal of Selected Topics in Signal Processing* 8, 6 (2014), 1098–1110.
[30] Jian Zhao, Xiaowen Chu, Hai Liu, Yiu-Wing Leung, and Zongpeng Li. 2015. Online procurement auctions for resource pooling in client-assisted cloud storage systems. In *Proc. of IEEE INFOCOM*. 576–584.
[31] Ruiting Zhou, Zongpeng Li, Chuan Wu, and Minghua Chen. 2015. Demand response in smart grids: A randomized auction approach. *IEEE Journal on Selected Areas in Communications* 33, 12 (2015), 2540–2553.

# A APPENDIX

## A.1 Proof of Theorem 3.2

The proof is adapted from [2] with several modifications based on special properties of the problem studied in this paper. The first step is to define a potential function for each source based on its cost, capacity characteristics, and congestion level. Toward this, at first we define the potential function $\phi_i$ for source $i$ as

$$\phi_i(t) = \begin{cases} \hat{f}_i \theta^{\gamma_i(t)-1} + \sum_{\tau=a_i}^{t-1} \hat{u}_i(\tau) z_i(\tau), & \text{if } x_i = 1, \\ \hat{f}_i x_i + \sum_{\tau=a_i}^{t-1} \hat{u}_i(\tau) z_i(\tau), & \text{otherwise.} \end{cases} \tag{6}$$

During the execution of the algorithm as congestion on each source increases, its potential function also changes. In addition, let $\phi(t) = \sum_{i \in \mathcal{I}} \phi_i(t)$ as the aggregated potential function. To prove Theorem 3.2 is suffices to show that $\phi(T) = O(n \log n)$ [2]. We obtain this bound by achieving a bound on any single iteration in OnFrc in Lemma A.1. Then, we use the results in Lemma A.1 and prove the bound on the final potential function value after executing OnFrc algorithm.

Lemma A.1. *If $1 < \theta < 3/2$, then the increment of $\phi$ in a single iteration of OnFrc is at most $4/T$.*

Proof. First, for notation convenience, let us define $\mathcal{P}'_t = \mathcal{P}_t \cup \{l_t\}$, as the set of sources that actively contribute in covering $d_t$ in fractional solution in OnFrc algorithm. Hereafter, for notational convenience we drop index $t$ from the potential function and use $\phi_i$ and $\phi$ instead of $\phi_i(t)$ and $\phi(t)$. To prove we consider two cases:

**case 1: when $x_{l_t} < 1$.**

Since $x_i < 1$ for all $i \in \mathcal{P}'_t$ (by definition in Lines 3-4 of OnFrc algorithm) the second term in Eq. (6) is active. Hence, The increase in potential function denoted as $\delta\phi$ is given by:

$$\delta\phi \leq \sum_{i \in \mathcal{P}'_t} \left( \hat{f}_i \frac{x_i}{\hat{f}_i T} + \hat{u}_i(t) \left( \frac{x_i / \hat{f}_i T}{d_i(t) + \hat{u}_i(t)/\hat{f}_i} \right) \right). \tag{7}$$

Knowing the fact that $d_i(t) \geq 0$ we have

$$\begin{aligned} \delta\phi &\leq \sum_{i \in \mathcal{P}'_t} \left( \hat{f}_i + \frac{\hat{u}_i(t)}{\hat{u}_i(t)/\hat{f}_i} \right) \left( \frac{x_i}{\hat{f}_i T} \right) \\ &= \sum_{i \in \mathcal{P}'_t} \frac{2x_i}{T} \leq \frac{4}{T}, \end{aligned} \tag{8}$$

where the last inequality is because $\sum_{i \in \mathcal{P}'_t} x_i \leq 2$.

**case 2: when $x_{l_t} = 1$.**

By taking a similar approach as the previous case, we know that that total increase in potential function for the sources in $\mathcal{P}_t$ is $\leq 2/T$; the procedure is exactly like the previous case, just in the last step because $\sum_{i \in \mathcal{P}_t} x_i < 1$, total increase is $\leq 2/T$. Now, the goal is to show that total increase due to the last source $l_t$ is $\leq 2/T$, hence, total increase would be $\leq 4/T$.

The increase in potential for last source $l_t$ follows from the first term in Eq (6). So, it is required to calculate the increase in congestion level which is $\delta\gamma_{l_t}(t) = d_{l_t}(t)/v_{l_t}(t)T$. Now, the increase in potential function $\delta\phi_{l_t}$ is given as

$$\delta\phi_{l_t} = \hat{f}_{l_t}\left(\theta^{\gamma_{l_t}(t-1)-1+\delta\gamma_{l_t}(t)} - \theta^{\gamma_{l_t}(t-1)-1}\right) + \frac{\hat{u}_{l_t}(t)}{v_{l_t}(t)T}.$$

On the other hand, we have

$$\delta\gamma_{l_t}(t) = \frac{d_{l_t}(t)}{v_{l_t}(t)T},$$

then, we by a simple rearrangement we get

$$\begin{aligned}\delta\phi_{l_t} &= \hat{f}_{l_t}\theta^{\gamma_{l_t}(t-1)-1}\left(\theta^{\frac{d_{l_t}(t)}{v_{l_t}(t)T}} - 1\right) + \frac{\hat{u}_{l_t}(t)}{v_{l_t}(t)T} \\ &= \alpha\left(\theta^{1/\beta T} - 1\right) + \frac{\hat{u}_{l_t}(t)}{\beta T d_{l_t}(t)},\end{aligned} \tag{9}$$

where

$$\alpha = \hat{f}_{l_t}\theta^{\gamma_{l_t}(t-1)-1},$$
$$\beta = \frac{v_{l_t}(t)}{d_{l_t}(t)} = \alpha + \frac{\hat{u}_{l_t}(t)}{d_{l_t}(t)},$$
$$v_{l_t}(t) = \hat{f}_{l_t}\theta^{\gamma_{l_t}(t-1)-1}d_{l_t}(t) + \hat{u}_{l_t}(t),$$

where the last equality is from the first term in Eq. (5). Now, given the following inequality, $(1+x)^{1/y} < e^{x/y} < 1 + 2x/y, y \geq x > 0$, we get

$$\begin{aligned}\delta\phi_{l_t} &< \frac{\alpha 2(\theta-1)}{\beta T} + \frac{\hat{u}_{l_t}(t)}{\beta T d_{l_t}(t)} \\ &< \left(\alpha + \frac{\hat{u}_{l_t}(t)}{d_{l_t}(t)}\right)\frac{1}{\beta T} = \frac{1}{T} < \frac{2}{T}.\end{aligned} \tag{10}$$

The proof requires $1 < \theta < 3/2$. □

Now, we proceed to obtain a bound on total potential function $\phi$. The goal is to achieve bounds on total increase on the value of potential function on all possible scenarios that might be happened in execution of OnFrc algorithm.

Let us denote $\mathcal{I}^\star$ as the optimal selected sources and $\mathcal{I}_t^\star$ as the set of sources that contribute in covering the demand in time slot $t$ in the optimal offline solution. In addition, let $i' \in \mathcal{I}_t^\star$ be the most expensive source in terms of virtual cost that contributed in covering the demand $d_t$. We find bounds on total increase on the value of potential function by categorizing the fractional solution by OnFrc as compared to the optimal offline solution as follows:

**category 1:** $i' \in \mathcal{P}_t$.

In this case, in any iteration of OnFrc the value of $x_{i'}$ increases to $x_{i'}\left(1 + \frac{1}{\hat{f}_{i'}T}\right)$. In addition, $x_i$ is initialized to $1/n$ for all sources. Consequently, there are at most $\sum_{i\in\mathcal{I}^\star}\hat{f}_i T\log n$ iterations in this case. From Lemma A.1 the bound in increase of potential function in each iteration is $4/T$. With the scaling procedure in Sec. 3.1.1 we know that $i \leq \sum_{i\in\mathcal{I}^\star}\hat{f}_i \leq 2n$. Then, the total increase in potential function in this case is $O(n\log n)$.

**category 2:** $i' \notin \mathcal{P}_t$ and $x_{i'} < 1$.

First, we find a bound on total number of iterations. For any time slot $t$, OnFrc will stop if the demand $d_t$ is covered, i.e., $\sum_{i\in\mathcal{I}:t\in\mathcal{T}_i} z_i(t) \geq 1$. Our goal is to show that after at most $2v_{i'}(t)T$ steps the covering constraint is satisfied. This is equivalent to show that in each iteration the increase in covered demand is at least $1/2v_{i'}(t)T$. Since $i' \notin \mathcal{P}_t$, the virtual cost of $i'$ is greater than all the items in $\mathcal{P}'_t$, i.e., $v_i(t) \leq v_{i'}(t), \forall i \in \mathcal{P}'_t$.

Now, lets consider the case when $x_i = 1, i = l_t$ in Line 12 of OnFrc algorithm. Indeed, the total increase in covered demand is at least $1/v_i(t)T \geq 1/v_{i'}(t)T \geq 1/2v_{i'}(t)T$. Hence, the proof is completed for this case.

We turn our focus into Line 9. In what follows we show that in any iteration, the total increment in covering demand is at least $1/2v_{i'}(t)T$ except for the last iteration. We note that in the case of Line 9, the increment in covering demand is the minimum of two values. Lets distinguish the sources based on this increment, so, let $\mathcal{P}_t^1 \subseteq \mathcal{P}'_t$ be the sources that their increment in covering variable is the first term, i.e., $\mathcal{P}_t^1 = \left\{i : i \in \mathcal{P}'_t, 2x_i \leq \frac{(x_i - 1/n)}{d_i(t) + \hat{u}_i(t)/\hat{f}_i}\right\}$. Similarly, we define $\mathcal{P}_t^2 = \mathcal{P}'_t \backslash \mathcal{P}_t^1$. If $\sum_{i\in\mathcal{P}_t^1} x_i \geq 1/2$, total increment is $\sum_{i\in\mathcal{P}_t^1} 2x_i \geq 1$, then the demand is covered, thereby the current step is the last iteration. Otherwise, the total increment by sources in $\mathcal{P}_t^2$ could be expressed by

$$\begin{aligned}\sum_{i\in\mathcal{P}_t^2}\frac{x_i}{(\hat{f}_i d_i(t) + \hat{u}_i(t))T} &= \sum_{i\in\mathcal{P}_t^2}\frac{x_i}{v_i(t)T} \\ \geq \frac{1}{v_{i'}(t)T}\sum_{i\in\mathcal{P}_2(t)} x_i &> \frac{1}{2v_{i'}(t)T}\end{aligned}$$

The last inequality is because we have $\sum_{i\in\mathcal{P}_t^2} x_i > 1/2$. Then, the lower bound on the minimum cover in each iteration is $1/(2v_{i'}(t)T)$.

The total increase in potential for all iterations is at most the increase in each iteration ($\leq 4/T$, by Lemma A.1) multiplied by the total number of iterations ($\leq 2v_{i'}(t)T$, proved above), i.e., the total increase in potential is at most $\sum_{t\in\mathcal{T}} 4/T \times 2v_{i'}(t)T = 8\sum_{t\in\mathcal{T}} v_{i'}(t)$, where $v_{i'}(t) = \hat{f}_{i'}(t)d_{i'}(t) + \hat{u}_{i'}(t)$ which is the second term in Eq. (5) since in this category $x_{i'} < 1$. Finally, we require to bound the total increase for all time slots, that is $\sum_{t\in\mathcal{T}} v_{i'}(t) = O(n)$ because of scaling procedure in Sec. 3.1.1.

**category 3:** $i'$ is fully chosen, i.e., $x_{i'} = 1$.

In this case we show that total increase in potential function after all iterations is bounded by $n + 8\sum_{i\in\mathcal{I}_A}\hat{f}_i\theta^{\Gamma_i - 1}$, where $\Gamma_i$ is the final congestion level of source $i$ by executing OnFrc after all time slots and $\mathcal{I}_A$ is the set of sources that are fully selected by OnFrc algorithm. Since $\gamma_i(t) \leq \Gamma_i$, we get

$$v_{i'}(t) = \hat{f}_{i'}\theta^{\gamma_{i'}(t)-1}d_{i'}(t) + \hat{u}_{i'}(t) \leq \hat{f}_{i'}\theta^{\Gamma_{i'}-1}d_{i'}(t) + \hat{u}_{i'}(t)$$

Then, the total increase in potential function $\Delta\phi$ is bounded by the maximum increase in each iteration ($\leq 4/T$, by Lemma A.1) multiplied by the total number of iterations ($\leq 2v_{i'}(t)T$)

$$\begin{aligned}
\Delta\phi &\leq 8\sum_{t\in\mathcal{T}} v_{i'}(t) \\
&\leq 8\sum_{t\in\mathcal{T}} \hat{f}_{i'}\theta^{\Gamma_{i'}-1} d_{i'}(t) + \hat{u}_{i'}(t) \\
&< n + 8\sum_{i\in\mathcal{I}^\star\cap\mathcal{I}_A} \hat{f}_i\theta^{\Gamma_i-1}\sum_{t:i'_t=i} d_i(t) \\
&\leq n + 8\sum_{i\in\mathcal{I}_A} \hat{f}_i\theta^{\Gamma_i-1}
\end{aligned}$$

Finally, note that after initialization of $x_i = 1/n$, $\phi \leq n$, because $\hat{f}_i \leq n$, then $\hat{f}_i x_i = \hat{f}_i/n \leq 1$. Indeed, the initial value of potential function is equal to $\sum_{i\in\mathcal{I}} \phi_i \leq n$. The value of potential function is at most the aggregation of its initial value and total increase during the iterations of the fractional algorithm, that is $\phi \leq O(n\log n) + 8\phi + n$. Hence, $\phi = O(n\log n)$.