Fully-Dynamic Submodular Cover with Bounded Recourse

Anupam Gupta
Computer Science Department
Carnegie Mellon University
Pittsburgh, USA
anupamg@cs.cmu.edu

Roie Levin
Computer Science Department
Carnegie Mellon University
Pittsburgh, USA
roiel@cs.cmu.edu

***Abstract*—In submodular covering problems, we are given a monotone, nonnegative submodular function $f : 2^{\mathcal{N}} \to \mathbb{R}_+$ and wish to find the min-cost set $S \subseteq \mathcal{N}$ such that $f(S) = f(\mathcal{N})$. When f is a coverage function, this captures SETCOVER as a special case. We introduce a general framework for solving such problems in a fully-dynamic setting where the function f changes over time, and only a bounded number of updates to the solution (a.k.a. recourse) is allowed. For concreteness, suppose a nonnegative monotone submodular integer-valued function g_t is added or removed from an active set $G^{(t)}$ at each time t. If $f^{(t)} = \sum_{g \in G^{(t)}} g$ is the sum of all active functions, we wish to maintain a competitive solution to SUBMODULARCOVER for $f^{(t)}$ as this active set changes, and with low recourse. For example, if each g_t is the (weighted) rank function of a matroid, we would be dynamically maintaining a low-cost common spanning set for a changing collection of matroids.**

We give an algorithm that maintains an $O(\log(f_{\max}/f_{\min}))$-competitive solution, where $f_{\max}, f_{\min}$ are the largest/smallest marginals of $f^{(t)}$. The algorithm guarantees a total recourse of $O(\log(c_{\max}/c_{\min}) \cdot \sum_{t \leq T} g_t(\mathcal{N}))$, where $c_{\max}, c_{\min}$ are the largest/smallest costs of elements in $\mathcal{N}$. This competitive ratio is best possible even in the offline setting, and the recourse bound is optimal up to the logarithmic factor. For monotone submodular functions that also have positive mixed third derivatives, we show an optimal recourse bound of $O(\sum_{t \leq T} g_t(\mathcal{N}))$. This structured class includes set-coverage functions, so our algorithm matches the known $O(\log n)$-competitiveness and $O(1)$ recourse guarantees for fully-dynamic SETCOVER. Our work simultaneously simplifies and unifies previous results, as well as generalizes to a significantly larger class of covering problems. Our key technique is a new potential function inspired by Tsallis entropy. We also extensively use the idea of *Mutual Coverage*, which generalizes the classic notion of mutual information.

***Keywords*-submodular optimization, online algorithms, dynamic algorithms**

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213. This research was done under the auspices of the Indo-US Virtual Networked Joint Center IUSSTF/JC-017/2017. Research supported in part by NSF awards CCF-1907820, CCF1955785, and CCF-2006953. Emails: `{anupamg,roiel}@cs.cmu.edu`.

The full version of this paper [30] is available at https://arxiv.org/abs/2009.00800

I. INTRODUCTION

In the SUBMODULARCOVER problem, we are given a monotone, nonnegative submodular function $f : 2^{\mathcal{N}} \to \mathbb{Z}_+$, as well as a linear cost function c, and we wish to find the min-cost set $S \subseteq \mathcal{N}$ such that $f(S) = f(\mathcal{N})$. This is a classical NP-hard problem: e.g., when f is a coverage function we capture the SETCOVER problem. Moreover, the greedy algorithm is known to be an $(1 + \ln f_{\max})$-approximation, where $f_{\max}$ is the maximum value of any single element [59]. This bound is tight assuming $\mathsf{P} \neq \mathsf{NP}$, even for the special case of SETCOVER [20], [18].

We consider this SUBMODULARCOVER problem in a fully-dynamic setting, where the notion of coverage changes over time. At each time, the underlying submodular function changes from $f^{(t)}$ to $f^{(t+1)}$, and the algorithm may have to change its solution from S_t to S_{t+1} to cover this new function. We do not want our solutions to change wildly if the function changes by small amounts. The goal of this work is to develop algorithms where this "churn" $|S_t \triangle S_{t+1}|$ is small (perhaps in an amortized sense), while maintaining the requirement that each solution S_t is a good approximate solution to the function $f^{(t)}$. The change $|S_t \triangle S_{t+1}|$ is often called *recourse* in the literature.

This problem has been posed and answered in the special case of SETCOVER—for clarity, from now on this paper we consider the equivalent HYPERGRAPHVERTEXCOVER (a.k.a. the HITTINGSET) problem. In this problem, hyperedges arrive to and depart from an active set over time, and we must maintain a small set of vertices that hit all active hyperedges. We know an algorithm that maintains an $O(\log n_t)$-approximation which has constant amortized recourse [27]; here n_t refers to the number of active hyperedges at time t. In other words, the total recourse—the total number of changes over T edge arrivals and departures—is only $O(T)$. The algorithm and analysis are based on a delicate token-based argument, which gives each hyperedge a constant number of tokens upon its arrival, and moves these tokens in a careful way between edges to account for the changes in the solution.

In the introduction we restrict to integer-valued functions for simplicity; all results extend to general nonnegative, monotone, submodular functions with suitable changes. See the technical sections for the full results and details.

2575-8454/20/$31.00 ©2020 IEEE
DOI 10.1109/FOCS46700.2020.00110

What do we do in the more general SUBMODULARCOVER case, where there is no notion of sets any more?

In this work we study the model where a submodular function is added or removed from the *active set* $G^{(t)}$ at each timestep: this defines the current submodular function $f^{(t)} := \sum_{g \in G_t} g$ as the sum of functions in this active set. The algorithm must maintain a subset $S_t \subseteq \mathcal{N}$ such that $f^{(t)}(S_t) = f^{(t)}(\mathcal{N})$ with cost $c(S_t)$ being within a small approximation factor of the optimal SUBMODULARCOVER for $f^{(t)}$, such that the total recourse $\sum_t |S_t \triangle S_{t+1}|$ remains small. To verify that this problem models dynamic HYPERGRAPHVERTEXCOVER, each arriving/departing edge A_t should correspond to a submodular function g_t taking on value 1 for any set S that hits A_t, and value zero otherwise (i.e., $g_t(S) = \mathbb{1}[S \cap A_t \neq \emptyset]$).

A. Our Results

Our main result is the following:

Theorem I.1 (Informal). *There is a deterministic algorithm that maintains an $e^2 \cdot (1 + \log f_{\max})$- competitive solution to Submodular Cover in the fully-dynamic setting where functions arrive/depart over time. This algorithm has total recourse:*

$$O\bigg(\sum_t g_t(\mathcal{N}) \ln\Big(\frac{c_{\max}}{c_{\min}}\Big)\bigg),$$

where $g_t(\mathcal{N})$ is the value of the function considered at time t, and $c_{\max}, c_{\min}$ are the maximum and minimum element costs.

Let us parse this result. Firstly, the approximation factor almost matches Wolsey's result up to the multiplicative factor of e^2; this is best possible in polynomial time unless $\mathsf{P} = \mathsf{NP}$ even in the offline setting. Secondly, the amortized recourse bound should be thought of as being logarithmic—indeed, specializing it to HITTINGSET where each $g_t(\mathcal{N}) = 1$, we get a total recourse of $O(T \log(c_{\max}/c_{\min}))$ over T timesteps, or an amortized recourse of $O(\log(c_{\max}/c_{\min}))$ per timestep. Hence this recourse bound is weaker by a log-of-cost-spread factor, while generalizing to all monotone submodular functions. Finally, since we are allowed to give richer functions g_t at each step, it is natural that the recourse scales with $\sum_t g_t(\mathcal{N})$, which is the total "volume" of these functions. In particular, this problem captures fully-dynamic HYPERGRAPHVERTEXCOVER where at each round a batch of k edges appears all at once (this is in contrast to the standard fully-dynamic model where hyperedges appear one at a time). In this case the algorithm may have to buy up to $k = g_t(\mathcal{N})/f_{\min}$ new vertices in general to maintain coverage.

We next show that for coverage functions (and hence for the HYPERGRAPHVERTEXCOVER problem), a variation on the algorithm from Theorem I.1 can remove the log-of-cost-spread factor in terms of recourse, at the cost of a slightly coarser competitive ratio. E.g., for HYPERGRAPHVERTEXCOVER the new competitive ratio corresponds to an $O(\log n)$ guarantee versus an $O(\log D_{\max})$ guarantee, where $D_{\max}$ being the largest degree of any vertex.

Theorem I.2 (Informal). *There is a deterministic algorithm that maintains an $O(\log f(\mathcal{N}))$- competitive solution to SUBMODULARCOVER in the fully-dynamic setting where functions arrive/depart over time, and each function is 3-increasing in addition to monotone and submodular. Furthermore, this algorithm has total recourse:*

$$O\bigg(\sum_t g_t(\mathcal{N})\bigg).$$

Indeed, this result holds not just for coverage functions, but for the broader class of 3-*increasing, monotone, submodular functions* [21], which are the functions we have been considering, with the additional property that have positive discrete mixed third-derivatives. At a high level, these are functions where the mutual coverage does not increase upon conditioning.

B. Techniques and Overview

The most widely known algorithm for SUBMODULARCOVER is the greedy algorithm; this repeatedly adds to the solution an element maximizing the ratio of its marginal coverage to its cost. It is natural to try to use the greedy algorithm in our dynamic setting; the main issue is that out-of-the-box, greedy may completely change its solution between time steps. In their result on recourse-bounded HYPERGRAPHVERTEXCOVER, [27] showed how to effectively imitate the greedy algorithm without sacrificing more than a small amount of recourse. A barrier to making greedy algorithms dynamic is their sequential nature, and hyperedge inserts/deletes can play havoc with this. So they give a local-search algorithm that skirts this sequential nature, while simultaneously retaining the key properties of the greedy algorithm. Unfortunately, their analysis hinges on delicately assigning edges to vertices. In the more general SUBMODULARCOVER setting, there are no edges to track, and this approach breaks down.

Our first insight is to return to the sequential nature of the greedy algorithm. Our algorithm maintains an ordering π on the elements of $\mathcal{N}$, which induces a solution to the SUBMODULARCOVER problem: we define the output of the algorithm as the prefix of elements that have non-zero marginal value. We maintain this permutation dynamically via local updates, and argue that only a small amount of recourse is necessary to ensure the solution is competitive. To bound the competitive ratio, we imagine that the permutation corresponds to the order in which an auxiliary offline algorithm selects elements, i.e. a *stack trace*. We show that our local updates maintain the invariant that this is the stack trace of an approximate greedy algorithm for the currently active set of functions. We hope that this general framework of doing local search *on the stack trace of an auxiliary algorithm* finds uses in other online/dynamic algorithms.

Our main technical contribution is to give a potential function to argue that our algorithm needs bounded recourse. The potential measures the *(generalized) entropy* of the *coverage vector* of the permutation π. This coverage vector is indexed by elements of the universe $\mathcal{N}$, and the value of each coordinate is the marginal coverage (according to permutation π) of the corresponding element. Entropy is often used as a potential function (notably, in recent developments in online algorithms) but in a qualitatively different way. In many if not all cases, these algorithms follow the maximum-entropy principle and seek *high entropy* solutions subject to feasibility constraints; the potential function then tracks the convergence to this goal. On the other hand, in our setting the cost function is the support size of the coverage vector, and minimizing this roughly corresponds to *low entropy*. We show entropy decreases sufficiently quickly with every change performed during the local search, and increases by a controlled amount with insertion/deletion of each function g, thus proving our recourse bounds.

We find our choice of entropy to be interesting for several reasons. For one, we use a suitably chosen *Tsallis entropy* [54], which is a general parametrized family of entropies, instead of the classical Shannon entropy. The latter also yields recourse bounds for our problem, but they are substantially weaker (see Appendix). Tsallis entropy has appeared in several recent algorithmic areas, for example as a regularizer in bandit settings [50], and as an approximation to Shannon entropy in streaming contexts [32]. Secondly, it is well known that for certain problems, minimizing an ℓ_1-objective is an effective proxy for minimizing sparsity [13]. In our dynamics, the ℓ_1 mass stays constant since total coverage does not change when elements are reordered. However, entropy is a good proxy for sparsity, in the sense that it decreases monotonically (and quickly!) as our algorithm moves within the ℓ_1 level set towards sparse vectors.

In Section II, we study the unit cost case to lay out our framework and highlight the main ideas. We show a $\log f_{\max}/f_{\min}$ competitive algorithm for fully-dynamic SUBMODULARCOVER with $O(\sum_t g_t(\mathcal{N})/f_{\min})$ total recourse. Then in Section III, we turn to general cost functions. We again show a $\log(f_{\max}/f_{\min})$ competitive algorithm, this time with $\log(c_{\max}/c_{\min}) \cdot \sum_t g_t(\mathcal{N})/f_{\min}$ recourse. The algorithm template is the same as before, but with a suitably generalized potential function and analysis. Here we also require a careful choice of the Tsallis entropy parameter, near (but not quite at) the limit point at which Tsallis entropy becomes Shannon entropy.

In the full version of this paper, we show how to remove the $\log(c_{\max}/c_{\min})$ dependence and achieve optimal recourse for a structured family of monotone, submodular functions: the class of 3-*increasing (monotone, submodular) set functions* [21]. These are set functions, all of whose discrete mixed third derivatives are nonnegative everywhere. Submodular functions in this class include *measure coverage functions*, which generalize set coverage functions, as well as entropies for distributions with positive interaction information (see, e.g., [37] for a discussion). Since this class includes SETCOVER, this recovers the optimal $O(1)$ recourse bound of [27]. For this result we use a more interesting generalization of the potential function from Section II that reweights the coverage of elements in the permutation non-uniformly as a function of their mutual coverage with other elements of the permutation.

In the appendix of the full version, we show how to get improved randomized algorithms when the functions g_t are assumed to be r-juntas. This is in analogy to approximation algorithms for SETCOVER under the frequency parametrization. We also show how to run an online "combiner" algorithm that gets the best of all worlds, with a competitive ratio of $O(\min(r, \log f_{\max}/f_{\min}))$. Finally, we demonstrate the generality of our framework by using it to recover known recourse bounded algorithms for fully-dynamic MINIMUMSPANNINGTREE and MINIMUMSTEINERTREE. These achieve $O(1)$ competitive ratios, and recourse bounds of $O(\log D)$ where D is the aspect ratio of the metric. Our proofs here are particularly simple and concise.

C. Related Work

Submodular Cover. While we introduced the problem for integer-valued functions, all results can be extended to real-valued settings by adding a dependence on $f_{\min}$, the smallest marginal value. Wolsey [59] showed that the greedy algorithm, repeatedly selecting the element maximizing marginal coverage to cost ratio, gives a $1 + \ln(f_{\max}/f_{\min})$ approximation for SUBMODULARCOVER; this guarantee is tight unless $\mathsf{P} = \mathsf{NP}$ even for the special case of SETCOVER [20], [18]. Fujito [22] gave a dual greedy algorithm that generalizes the F-approximation for SETCOVER [33] where F is the maximum-frequency of an element.

SUBMODULARCOVER has been used in many applications to resource allocation and placement problems, by showing that the coverage function is monotone and submodular, and then applying Wolsey's greedy algorithm as a black box. We automatically port these applications to the fully-dynamic setting where coverage requirements change with time. E.g., in selecting influential nodes to disseminate information in social networks [23], [44], [53], [36], exploration for robotics planning problems [40], [38], [6], placing sensors [60], [48], [62], [46], and other physical resource allocation objectives [61], [43], [55]. The networking community has been particularly interested in SUBMODULARCOVER recently, since SUBMODULARCOVER models network function placement tasks [2], [42], [39], [45], [15]. E.g., [45] want to place middleboxes in a network incrementally, and point out that avoiding closing extant boxes is a huge boon in practice.

The definition of *m-increasing functions* is due to Foldes and Hammer [21]. Bach [5] gave a characterization of the class of *measure coverage functions* (which we define later) in terms of its iterated discrete derivatives. This class generalizes coverage functions, and is contained in the class of 3-increasing functions. [35] give several additional examples of 3-increasing functions. Several papers [35], [17], [56], [57] have given algorithms specifically for 3-increasing submodular function optimization.

Online and Dynamic Algorithms. There is a still budding series of work on recourse-bounded algorithms. Besides [27] which is most directly related to our work, researchers have studied the Steiner Tree problem [34], [28], [25], [41], clustering [26], [16], matching [24], [14], [12], and scheduling [47], [58], [3], [49], [52], [19], [29].

A rich parallel line of work has studied how to minimize update time for problems in the dynamic or fully-dynamic setting. In [27], the authors give an $O(\log n)$ competitive and $O(F \log n)$ update time algorithm for fully-dynamic HYPERGRAPHVERTEXCOVER. An ongoing program of research for the frequency parametrization of HYPERGRAPHVERTEXCOVER [8], [7], [11], [1], [9], [10] has so far culminated in an $F(1+\epsilon)$ competitive algorithm with $\mathrm{poly}(F, \log c_{\max}/c_{\min}, 1/\epsilon)$ update time (where F is the frequency).

In recent work, [31] studied the problem of maintaining a feasible solution to SUBMODULARCOVER in a related online model. That setting is an insertion-only irrevocable analog of this work, where functions g_t may never leave the active set. Our results can be seen as an extension/improvement when recourse is allowed: not only can our algorithm handle the fully-dynamic case with insertions and deletions, but we improve the competitive ratio from $O(\log n \cdot \log f_{\max}/f_{\min})$ to $O(\log f_{\max}/f_{\min})$, which is best possible even in the offline case.

Our work is related to work on convex body chasing (e.g., [4], [51]) in spirit but not in techniques. For an online/dynamic covering problems, the set of feasible fractional solutions within distance α of the optimal solution at a given time step form a convex set: our goal is similarly to "chase" these convex bodies, while limiting the total movement traversed. The main difference is that we seek *absolute bounds* on the recourse, instead of recourse that is competitive with the optimal chasing solution. (We can give such bounds because our feasible regions are structured and not arbitrary convex bodies).

D. Notation and Preliminaries

A set function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ is *submodular* if $f(A \cap B) + f(A \cup B) \leq f(A) + f(B)$ for any $A, B \subseteq \mathcal{N}$. It is *monotone* if $f(A) \leq f(B)$ for all $A \subseteq B \subseteq \mathcal{N}$. We assume access to a *value oracle* for f that computes $f(T)$ given $T \subseteq \mathcal{N}$. The *contraction* of $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ onto $\mathcal{N} \setminus T$ is defined as $f_T(S) = f(S \mid T) := f(S \cup T) - f(T)$. If f is submodular then f_T is also submodular for any $T \subseteq \mathcal{N}$. We use the following notation.

$$f^{(t)}_{\max} := \max\{f^{(t)}(j) \mid j \in N\}, \tag{I.1}$$

$$f^{(t)}_{\min} := \min\{f^{(t)}(j \mid S) \mid j \in N, \; S \subseteq \mathcal{N},\; f^{(t)}(j \mid S) \neq 0\}. \tag{I.2}$$

$$f_{\max} := \max_t f^{(t)}_{\max}, \tag{I.3}$$

$$f_{\min} := \min_t f^{(t)}_{\min}. \tag{I.4}$$

Also we let $c_{\max}$ and $c_{\min}$ denote the largest and smallest costs of elements respectively.

Throughout this paper, we will use the convention that $1 : k$ denotes that range of indices from 1 to k.

Mutual Coverage. We will use the notion of mutual coverage defined in [31]. Independently, [35] defined and studied the same quantity under the slightly different name *submodular mutual information*.

Definition I.3 (Mutual Coverage)**.** *The* mutual coverage *and* conditional mutual coverage *with respect to a set function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ are defined as:*

$$\mathcal{I}_f(A; B) := f(A) + f(B) - f(A \cup B), \tag{I.5}$$

$$\mathcal{I}_f(A; B \mid C) := f_C(A) + f_C(B) - f_C(A \cup B). \tag{I.6}$$

We may think of $\mathcal{I}_f(A; B \mid C)$ intuitively as being the amount of coverage B "takes away" from the coverage of A (or vice-versa, since the definition is symmetric in A and B), given that C was already chosen. This generalizes the notion of *mutual information* from information theory: if $\mathcal{N}$ is a set of random variables, and $S \subseteq \mathcal{N}$, and if $f(S)$ denotes the joint entropy of the random variables in the set S, then $\mathcal{I}$ is the mutual information.

Fact I.4 (Chain Rule)**.** *Mutual coverage respects the identity:*

$$\mathcal{I}_f(A; B_1 \cup B_2 \mid C) = \mathcal{I}_f(A; B_1 \mid C) + \mathcal{I}_f(A; B_2 \mid C \cup B_1).$$

This neatly generalizes the chain rule for mutual information.

II. UNIT COST SUBMODULAR COVER

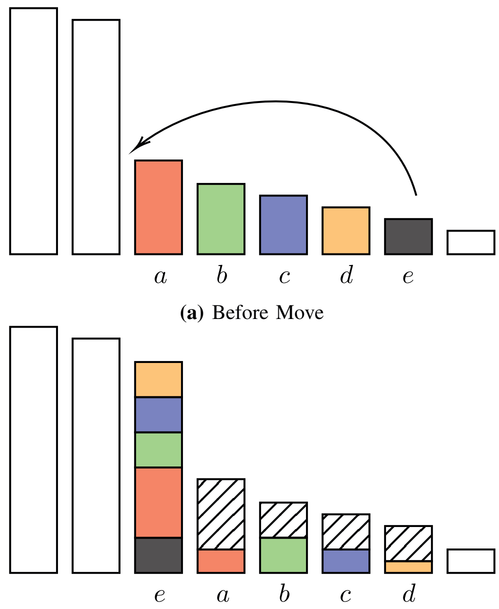
A. The Algorithm

We now present our first algorithm for unit-cost fully-dynamic SUBMODULARCOVER. We will show the following:

Theorem II.1. *For any $\gamma > e$, there is a deterministic algorithm that maintains a $\gamma(\log f^{(t)}_{\max}/f^{(t)}_{\min} + 1)$-competitive solution to unweighted* SUBMODULARCOVER *in the setting where functions arrive/depart over time. Furthermore, this algorithm has total recourse:*

$$2 \cdot \frac{e \ln \gamma}{\gamma - e \ln \gamma} \cdot \frac{\sum_t g_t(\mathcal{N})}{f_{\min}} = O\left(\frac{\sum_t g_t(\mathcal{N})}{f_{\min}}\right).$$

The algorithm and its analysis are particularly clean; we will build on these in the following sections. We begin by



(a) Before Move

(b) After Move

Figure 1: Illustration of a legal γ-move. Each rectangle represents the marginal coverage of an element of the permutation. The height of the item that moves must be at least γ times the height of anyone it cuts in line.

describing the algorithm. We maintain a permutation π of the elements in $\mathcal{N}$, and assign to each element its marginal coverage given what precedes it in the permutation. We write this marginal value assigned to element π_i as

$$\mathfrak{F}_\pi(\pi_i) := f(\pi_i \mid \pi_{1:i-1}). \tag{II.1}$$

We consider two kinds of local search moves:

1) **Swaps:** transform π to π' by moving an element at position i to position $i-1$ on the condition that $\mathfrak{F}(\pi_i) \geq \mathfrak{F}(\pi_{i-1})$. In words, this is a *bubble* operation (as in bubble-sort).
2) γ**-moves**: transform the permutation π to π' by moving an element u from a position q to some other position $p < q$ on the condition that for all $i \in \{p, \ldots, q-1\}$,

$$\mathfrak{F}_{\pi'}(\pi'_p) \geq \gamma \cdot \mathfrak{F}_\pi(\pi_i).$$

In words, when u moves ahead in line, it "steals" coverage from other elements along the way; we require that the amount covered by u *after the move* (which is given by $\mathfrak{F}_{\pi'}(\pi'_p)$ since u now sits at position p) to be at least a γ factor larger than the coverage *before the move* of any element that u jumps over. (See Figure 1.)

The dynamic algorithm is the following. When a new function $g^{(t)}$ arrives or departs, update the coverages $\mathfrak{F}_\pi$ of all the elements in the permutation. Subsequently, while there is a local search move available, perform the move. Output the prefix of π of all elements with non-zero coverage. This is summarized in Algorithm 1, with a setting of $\gamma > e$.

Algorithm 1 FULLYDYNAMICSUBMODULARCOVER

$\pi \leftarrow$ arbitrary initial permutation of elements $\mathcal{N}$.
for $t = 1, 2, \ldots, T$ **do**
  t^{th} function g_t arrives/departs.
  while there exists a legal γ-move or a swap for π **do**
    Perform the move, and update π.
  Output the collection of π_i such that $\mathfrak{F}_\pi(\pi_i) > 0$.

B. Bounding the Cost

Let us start by arguing that if the algorithm terminates, it must produce a competitive solution.

Lemma II.2. *Suppose no γ-moves are possible, then for every index i such that $\mathfrak{F}_\pi(\pi_i) > 0$, and for every index $i' > i$, the following holds. Let π' be the permutation where $\pi_{i'}$ is moved to position i. Then*

$$\mathfrak{F}_\pi(\pi_i) > \frac{\mathfrak{F}_{\pi'}(\pi_{i'})}{\gamma} \tag{II.2}$$

Proof: Suppose there are elements π_i and $\pi_{i'}$ such that condition (II.2) does not hold, i.e. $\mathfrak{F}_{\pi'}(\pi_j) \geq \gamma \cdot \mathfrak{F}_\pi(\pi_i)$. Since by assumption there are no swaps available, the permutation π is in non-increasing order of $\mathfrak{F}_\pi(\pi_i)$ values, so for all indices $j > i$ it also holds that $\mathfrak{F}_\pi(\pi_{i'}) \geq \gamma \cdot \mathfrak{F}_\pi(\pi_j)$. Hence moving i' from its current position to position i is a legal γ-move, which is a contradiction. ∎

Corollary II.3. *The output at every time step is $\gamma \cdot (\log f^{(t)}_{\max}/f^{(t)}_{\min} + 1)$-competitive.*

Proof: Lemma II.2 implies that the solution is equivalent to greedily selecting an element whose marginal coverage is within a factor of $1/\gamma$ of the largest marginal coverage currently available. Given this, the standard analyses of the greedy algorithm for SUBMODULARCOVER [59] imply that the solution is $\gamma \cdot (\log f^{(t)}_{\max}/f^{(t)}_{\min} + 1)$ competitive. ∎

C. Bounding the Recourse

We move on to showing that the algorithm must terminate with $O(g(\mathcal{N})/f_{\min})$ amortized recourse. For this analysis, we define a potential function parameterized by a number $\alpha \in (0, 1)$ to be fixed later:

$$\Phi_\alpha(f, \pi) := \sum_{i \in N} \left(\mathfrak{F}_\pi(\pi_i)\right)^\alpha.$$

As noted in the introduction, up to scaling and shifting, this is the Tsallis entropy with parameter α. We show several properties of this potential:

Properties of Φ_α:

I) Φ_α increases by at most $g_t(\mathcal{N}) \cdot (f_{\min})^{\alpha-1}$ with the addition of function g_t to the active set.
II) Φ_α does not increase with deletion of functions from the system.
III) Φ_α does not increase during swaps.
IV) For an appropriate range of γ, the potential Φ_α decreases by at least $(\gamma/(e \ln \gamma) - 1) \cdot (f_{\min})^\alpha = \Omega((f_{\min})^\alpha)$ with every γ-move.

Lemma II.4. *If $\alpha = (\ln \gamma)^{-1}$, then Φ_α respects properties I to IV.*

Proof: For brevity, define $h(z) := z^\alpha$. Since $\alpha \in (0,1)$, this function is concave and non-decreasing.

We start with property I. When a function g is added to the system, for some set of $i \in [n]$, it increases $k_i := \mathfrak{F}_\pi(\pi(i))$ by some amount Δ_i. Observe that $\sum_i \Delta_i = g(\mathcal{N})$. By the concavity of h:

$$\begin{aligned}\sum_i h(k_i + \Delta_i) - \sum_i h(k_i) &\leq \sum_i h(\Delta_i) \\ &\leq \frac{\sum_i \Delta_i}{f_{\min}} \cdot h(f_{\min}) \\ &= g(\mathcal{N}) \cdot (f_{\min})^{\alpha-1}.\end{aligned}$$

Property II follows since h is non-decreasing.

For property III, we wish to show that if u immediately precedes v in π and $\mathfrak{F}_\pi(u) \leq \mathfrak{F}_\pi(v)$, then swapping u and v does not increase the potential. Let $\widehat{\pi}$ denote the permutation after the swap. Note that $\mathfrak{F}_{\widehat{\pi}}(u) \leq \mathfrak{F}_\pi(v)$ and $\mathfrak{F}_{\widehat{\pi}}(v) \geq \mathfrak{F}_\pi(v)$, since u may only have lost some amount of coverage to v. Suppose this amount is k, i.e. $k = \mathfrak{F}_\pi(u) - \mathfrak{F}_{\widehat{\pi}}(u) = \mathfrak{F}_{\widehat{\pi}}(v) - \mathfrak{F}_\pi(v)$. Then:

$$\begin{aligned}&\Phi_\alpha(f, \widehat{\pi}) - \Phi_\alpha(f, \pi) \\ &= h(\mathfrak{F}_{\widehat{\pi}}(u)) + h(\mathfrak{F}_{\widehat{\pi}}(v)) - h(\mathfrak{F}_\pi(u)) - h(\mathfrak{F}_\pi(v)) \\ &= h(\mathfrak{F}_\pi(u) - k) + h(\mathfrak{F}_\pi(v) + k) - h(\mathfrak{F}_\pi(u)) - h(\mathfrak{F}_\pi(v))\end{aligned}$$

which is non-positive due to the concavity of h.

It remains to prove property IV. Suppose we perform a γ-move on a permutation π. Let u be the element moving to some position p from some position $q > p$, and let π' denote the permutation after the move. For convenience, also define:

$$\begin{aligned}v_i &:= \mathfrak{F}_\pi(\pi_i), \qquad \text{(the original coverage of the } i^{th} \text{ set)} \\ a_i &:= \mathcal{I}_f(\pi_i; u \mid \pi_{1:i-1}) = \mathfrak{F}_\pi(\pi_i) - \mathfrak{F}_{\pi'}(\pi_i). \\ &\qquad \text{(the loss in coverage of the } i^{th} \text{ set)}\end{aligned}$$

Note that for all $i \notin \{p, \ldots, q-1\}$, we necessarily have $a_i = 0$. Also note that $\mathfrak{F}_{\pi'}(u) = \sum_i a_i$, and by the definition of a γ-move, for any j we have $\sum_i a_i \geq \gamma \cdot v_j$. Then the change in potential is:

$$\begin{aligned}&\Phi_\alpha(f, \pi') - \Phi_\alpha(f, \pi) \\ &= \Big(\sum_i a_i\Big)^\alpha + \sum_i (v_i - a_i)^\alpha - \sum_i v_i^\alpha \\ &\leq \Big(\sum_i a_i\Big)^\alpha - \sum_i a_i \cdot \alpha \cdot v_i^{\alpha-1} \qquad \text{(II.3)} \\ &\leq \Big(\sum_i a_i\Big)^\alpha - \Big(\sum_i a_i\Big)^\alpha \cdot \alpha \cdot \gamma^{1-\alpha} \qquad \text{(II.4)} \\ &\leq -\Big(\frac{\gamma}{e \ln \gamma} - 1\Big)(f_{\min})^\alpha. \qquad \text{(II.5)}\end{aligned}$$

Above, (II.3) holds since h is concave and thus $h(v_i - a_i) - h(v_i) \leq \nabla h(v_i) \cdot (-a_i)$. Line (II.4) holds by the definition of a γ-move, since $\sum_i a_i \geq \gamma v_j$ for every $j \in \{p, \ldots, q-1\}$. Finally, (II.5) comes from our choice of $\alpha = (\ln \gamma)^{-1}$ and the fact that $\sum_i a_i \geq f_{\min}$. ■

Finally, we return to proving the main theorem.

Proof of Theorem II.1: By Lemma II.2, if Algorithm 1 (using Definition II.1 for $\mathfrak{F}_\pi$) terminates then it is $\gamma \cdot (\log f_{\max}^{(t)} / f_{\min}^{(t)} + 1)$-competitive.

By I to IV, the potential Φ_α increases by at most $g_t(\mathcal{N})(f_{\min})^{\alpha-1}$ for every function g_t inserted to the active set, decreases by $(f_{\min})^\alpha \cdot (\gamma/(e \ln \gamma) - 1)$ per γ-move, and otherwise does not increase. By inspection, $\Phi_\alpha \geq 0$. The number of elements e with $\mathfrak{F}_\pi(e) > 0$ grows by 1 only during γ-moves in which $\mathfrak{F}_\pi(e)$ was initially 0. Otherwise, this number never grows. We account for elements leaving the solution by paying recourse 2 upfront when they join the solution.

Hence, the number of changes to the solution is at most:

$$2 \cdot \frac{\sum_t g_t(\mathcal{N})}{(f_{\min})^{1-\alpha}} \cdot \frac{e \ln \gamma}{(f_{\min})^\alpha (\gamma - e \ln \gamma)} = O\left(\frac{\sum_t g_t(\mathcal{N})}{f_{\min}}\right).$$

■

Our algorithm gives a tunable tradeoff between approximation ratio and recourse depending on the choice of γ. Note that if we wish to optimize the competitive ratio, setting $\gamma = e(1+\delta)$ gives a recourse bound of

$$\left[\left(1 + \frac{\ln(1+\delta)}{\delta - \ln(1+\delta)}\right)\right] \frac{\sum_t g_t(\mathcal{N})}{f_{\min}} = O\left(\frac{1}{\delta^2}\right) \frac{\sum_t g_t(\mathcal{N})}{f_{\min}}$$

as δ approaches 0. For simplicity one can set $\gamma = e^2$ to get the bound in Theorem II.1.

III. General Cost Submodular Cover

A. The Algorithm

We now show the main result of our paper:

Theorem III.1. *There is a deterministic algorithm that maintains an $e^2 \cdot (\log f_{\max}^{(t)} / f_{\min}^{(t)} + 1)$- competitive solution to* SubmodularCover *in the setting where functions arrive/depart over time. Furthermore, this algorithm has amortized recourse:*

$$O\left(\frac{g(\mathcal{N})}{f_{\min}} \ln\left(\frac{c_{\max}}{c_{\min}}\right)\right)$$

per function arrival/departure.

Given the last section, our description of the new algorithm is very simple. We reuse Algorithm 1, except we redefine:

$$\mathfrak{F}_\pi(\pi_i) := \frac{f(\pi_i \mid \pi_{1:i-1})}{c(\pi_i)}.$$

We will specify the last free parameter γ shortly.

B. Bounding the Cost

To bound the competitive ratio, note that Eq. (II.2) did not use any particular properties of $\mathfrak{F}_\pi$, except that in the solution output by the algorithm, there are no γ-moves or swaps with respect to $\mathfrak{F}_\pi$ in permutation π. The analog of Corollary II.3 is nearly identical:

Corollary III.2. *The output at every time step is $\gamma \cdot (\log f^{(t)}_{\max}/f^{(t)}_{\min} + 1)$-competitive.*

Proof: Lemma II.2 implies that the solution is equivalent to greedily selecting an element with marginal coverage/cost ratio within a $1/\gamma$ factor of the largest marginal coverage/cost ratio currently available. The standard analyses of the greedy algorithm for SUBMODULARCOVER [59] imply that the solution is $\gamma \cdot (\log f^{(t)}_{\max}/f^{(t)}_{\min} + 1)$-competitive. ∎

C. Bounding the Recourse

To make our analysis modular, we will write a general potential function, parameterized by a function $h : \mathbb{R} \to \mathbb{R}$:

$$\Phi_h(f, \pi) := \sum_{i \in N} c(\pi_i) \cdot h\left(\mathfrak{F}_\pi(\pi_i)\right).$$

With foresight, we require several properties of h:

Properties of h:

i) h is monotone and concave.
ii) $h(0) = 0$.
iii) h satisfies $x \cdot h'(x/\gamma) \geq (1 + \epsilon_\gamma)h(x)$.
iv) h satisfies $y \cdot h(x/y)$ is monotone in y.

To bound the recourse, our goal will again be to show the following properties of our potential function Φ_h:

Properties of Φ_h:

I) Φ_h increases by at most
$$\frac{g_t(\mathcal{N})}{f_{\min}} \cdot c_{\max} \cdot h\left(\frac{f_{\min}}{c_{\max}}\right)$$
with the addition of function g_t to the active set.
II) Φ_h does not increase with deletion of functions from the system.
III) Φ_h does not increase during swaps.
IV) For an appropriate range of γ, the potential Φ_h decreases by at least
$$\epsilon_\gamma \cdot c_{\min} \cdot h\left(\frac{f_{\min}}{c_{\min}}\right)$$
with every γ-move.

Together, the statements imply a total recourse bound of:

$$\frac{\sum_t g_t(\mathcal{N})}{\epsilon_\gamma \cdot f_{\min}} \cdot \frac{c_{\max}}{c_{\min}} \cdot \frac{h(f_{\min}/c_{\max})}{h(f_{\min}/c_{\min})}$$

Lemma III.3. *If h respects properties i to iv then Φ_h respects properties I to IV.*

Proof: We start with property I. When a function g_t is added to the system, for some set of $i \in [n]$, it increases $k_i := \mathfrak{F}_\pi(\pi(i))$ by some amount Δ_i. Then the potential increase is:

$$\begin{aligned}
&\Phi_h(f^{(t)}, \pi) - \Phi_h(f^{(t-1)}, \pi) \\
&= \sum_{i \in [n]} c(\pi_i) \cdot h\left(\frac{k_i + \Delta_i}{c(\pi_i)}\right) - \sum_{i \in [n]} c(\pi_i) \cdot h\left(\frac{k_i}{c(\pi_i)}\right) \\
&\leq \sum_{i \in [n]} c(\pi_i) \cdot h\left(\frac{\Delta_i}{c(\pi_i)}\right) && \text{(III.1)} \\
&\leq \sum_{i \in [n]} c_{\max} \cdot h\left(\frac{\Delta_i}{c_{\max}}\right) && \text{(III.2)} \\
&\leq \frac{\sum_i \Delta_i}{f_{\min}} \cdot c_{\max} \cdot h\left(\frac{f_{\min}}{c_{\max}}\right) && \text{(III.3)} \\
&= \frac{g_t(\mathcal{N})}{f_{\min}} \cdot c_{\max} \cdot h\left(\frac{f_{\min}}{c_{\max}}\right).
\end{aligned}$$

Above step (III.1) is by properties i and ii, step (III.2) is by property iv and step (III.3) is by property i.

Property II follows since h is non-decreasing.

The proof of property III is similar to the one in Lemma II.4. Suppose u immediately precedes v in π but $\mathfrak{F}_\pi(u) \leq \mathfrak{F}_\pi(v)$, and let $\widehat{\pi}$ denote the permutation after the swap. We have that $\mathfrak{F}_{\widehat{\pi}}(u) \leq \mathfrak{F}_\pi(v)$ and $\mathfrak{F}_{\widehat{\pi}}(v) \geq \mathfrak{F}_\pi(v)$, since u may only have lost some amount of coverage to v. Suppose this amount is k, i.e. $k = \mathfrak{F}_\pi(u) - \mathfrak{F}_{\widehat{\pi}}(u) = \mathfrak{F}_{\widehat{\pi}}(v) - \mathfrak{F}_\pi(v)$. Then:

$$\begin{aligned}
&\Phi_h(f, \widehat{\pi}) - \Phi_h(f, \pi) \\
&= c(u) \cdot h\left(\mathfrak{F}_{\widehat{\pi}}(u)\right) + c(v) \cdot h(\mathfrak{F}_{\widehat{\pi}}(v)) \\
&\quad - c(u) \cdot h(\mathfrak{F}_\pi(u)) - c(v) \cdot h(\mathfrak{F}_\pi(v)) \\
&= c(u)\left(h\left(\mathfrak{F}_\pi(u) - \frac{k}{c(u)}\right) - h(\mathfrak{F}_\pi(u))\right) \\
&\quad + c(v)\left(h\left(\mathfrak{F}_\pi(v) + \frac{k}{c(v)}\right) - h(\mathfrak{F}_\pi(v))\right) \\
&\leq k \cdot (h'(\mathfrak{F}_\pi(v)) - h'(\mathfrak{F}_\pi(u)))
\end{aligned}$$

which is non-positive due to the concavity of h and the fact that $\mathfrak{F}_\pi(v) \geq \mathfrak{F}_\pi(u)$.

Finally the proof of property IV is also similar to the version in the last section. Suppose we perform a γ-move on a permutation π. Let u be the element moving to some position p from some position $q > p$, and let π' denote the permutation after the move. Recall the definitions of a_i and

v_i from Section II-C. Then:

$$\begin{aligned}
&\Phi_h(f,\pi') - \Phi_h(f,\pi) \\
&= c(u)\cdot h\left(\frac{\sum_{i\in[n]} a_i}{c(u)}\right) + \sum_{i\in[n]} c(\pi_i)\cdot h\left(\frac{v_i - a_i}{c(\pi_i)}\right) \\
&\quad - \sum_{i\in[n]} c(\pi_i)\cdot h\left(\frac{v_i}{c(\pi_i)}\right) \\
&\le c(u)\cdot h\left(\frac{\sum_{i\in[n]} a_i}{c(u)}\right) - \sum_{i\in[n]} a_i\cdot h'\left(\frac{v_i}{c(\pi_i)}\right) \qquad \text{(III.4)} \\
&\le c(u)\cdot h\left(\frac{\sum_{i\in[n]} a_i}{c(u)}\right) \\
&\quad - c(u)\sum_{i\in[n]} \frac{a_i}{c(u)}\cdot h'\left(\frac{\sum_{i\in[n]} a_i}{\gamma\cdot c(u)}\right) \qquad \text{(III.5)} \\
&\le c(u)\cdot h\left(\frac{\sum_{i\in[n]} a_i}{c(u)}\right) \\
&\quad - (1+\epsilon_\gamma)\cdot c(u)\cdot h\left(\frac{\sum_{i\in[n]} a_i}{c(u)}\right) \qquad \text{(III.6)} \\
&\le -\epsilon_\gamma\cdot c(u)\cdot h\left(\frac{\sum_{i\in[n]} a_i}{c(u)}\right) \\
&\le -\epsilon_\gamma\cdot c_{\min}\cdot h\left(\frac{f_{\min}}{c_{\min}}\right). \qquad \text{(III.7)}
\end{aligned}$$

Above step (III.4) uses the concavity of h. Step (III.5) uses the fact that moving u was a legal γ-move and thus $\sum_{i\in[n]} a_i/c(u) \ge \gamma v_i/c(\pi_i)$. Step (III.6) follows from property iii. Finally, (III.7) uses property iv again. ∎

With this setup, the proof of the main theorem boils down to identifying an appropriate function h, and a suitable choice of γ.

Proof of Theorem III.1: Recall the general recourse bound is

$$\frac{\sum_t g_t(\mathcal{N})}{\epsilon_\gamma\cdot f_{\min}}\cdot\frac{c_{\max}}{c_{\min}}\cdot\frac{h(f_{\min}/c_{\max})}{h(f_{\min}/c_{\min})}.$$

A good choice for h is $h(x) = x^{1-\delta}/(1-\delta)$ for $\delta = (\ln(c_{\max}/c_{\min})+1)^{-1}$, and with $\gamma = e^2$. Properties i to iv are easy to verify. To see that this implies the bound

$$O\left(\frac{\sum_t g_t(\mathcal{N})}{f_{\min}}\ln\left(\frac{c_{\max}}{c_{\min}}\right)\right),$$

note that $\gamma = e^2 \ge ((1+\delta)/(1-\delta))^{1/\delta}$, in which case $\epsilon_\gamma = \gamma^\delta(1-\delta) - 1 \ge \delta$. Finally, $(c_{\max}/c_{\min})^\delta = O(1)$. ∎

REFERENCES

[1] A. Abboud, R. Addanki, F. Grandoni, D. Panigrahi, and B. Saha. Dynamic set cover: improved algorithms and lower bounds. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 114–125, 2019.

[2] K. Andreev, C. Garrod, D. Golovin, B. Maggs, and A. Meyerson. Simultaneous source location. *ACM Trans. Algorithms*, 6(1):16:1–16:17, Dec. 2009.

[3] M. Andrews, M. X. Goemans, and L. Zhang. Improved bounds for on-line load balancing. *Algorithmica*, 23(4):278–301, 1999.

[4] C. Argue, A. Gupta, G. Guruganesh, and Z. Tang. Chasing convex bodies with linear competitive ratio. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1519–1524. SIAM, 2020.

[5] F. Bach. Learning with Submodular Functions: A Convex Optimization Perspective. *Foundations and TrendsÂő in Machine Learning*, 6(2-3):145–373, 2013.

[6] M. Beinhofer, J. Müller, A. Krause, and W. Burgard. Robust landmark selection for mobile robot navigation. In *IROS*, pages 3637–2643, 2013.

[7] S. Bhattacharya, D. Chakrabarty, and M. Henzinger. Deterministic fully dynamic approximate vertex cover and fractional matching in $O(1)$ amortized update time. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 86–98. Springer, 2017.

[8] S. Bhattacharya, M. Henzinger, and G. F. Italiano. Deterministic fully dynamic data structures for vertex cover and matching. *SIAM Journal on Computing*, 47(3):859–887, 2018.

[9] S. Bhattacharya, M. Henzinger, and D. Nanongkai. A New Deterministic Algorithm for Dynamic Set Cover. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 406–423. IEEE, 2019.

[10] S. Bhattacharya, M. Henzinger, D. Nanongkai, and X. Wu. An Improved Algorithm for Dynamic Set Cover. *arXiv preprint arXiv:2002.11171*, 2020.

[11] S. Bhattacharya and J. Kulkarni. Deterministically Maintaining a $(2+\epsilon)$-Approximate Minimum Vertex Cover in $O(1/\epsilon^2)$ Amortized Update Time. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1872–1885. SIAM, 2019.

[12] B. Bosek, D. Leniowski, P. Sankowski, and A. Zych. Online bipartite matching in offline time. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 384–393. IEEE, 2014.

[13] E. J. Candès, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

[14] K. Chaudhuri, C. Daskalakis, R. D. Kleinberg, and H. Lin. Online bipartite perfect matching with augmentations. In *IEEE INFOCOM 2009*, pages 1044–1052. IEEE, 2009.

[15] Y. Chen, J. Wu, and B. Ji. Virtual network function deployment in tree-structured networks. In *2018 IEEE 26th International Conference on Network Protocols (ICNP)*, pages 132–142. IEEE, 2018.

[16] V. Cohen-Addad, N. O. D. Hjuler, N. Parotsidis, D. Saulpic, and C. Schwiegelshohn. Fully Dynamic Consistent Facility Location. In *Advances in Neural Information Processing Systems*, pages 3250–3260, 2019.

[17] M. Corah and N. Michael. Distributed Submodular Maximization on Partition Matroids for Planning on Large Sensor Networks. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6792–6799, 2018.

[18] I. Dinur and D. Steurer. Analytical Approach to Parallel Repetition. In *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing*, STOC '14, pages 624–633, New York, NY, USA, 2014. ACM.

[19] L. Epstein and A. Levin. Robust algorithms for preemptive scheduling. *Algorithmica*, 69(1):26–57, 2014.

[20] U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[21] S. Foldes and P. L. Hammer. Submodularity, supermodularity, and higher-order monotonicities of pseudo-Boolean functions. *Mathematics of Operations Research*, 30(2):453–461, 2005.

[22] T. Fujito. Approximation algorithms for submodular set cover with applications. *IEICE Transactions on Information and Systems*, 83(3):480–487, 2000.

[23] A. Goyal, F. Bonchi, L. V. S. Lakshmanan, and S. Venkatasubramanian. On minimizing budget and time in influence propagation over social networks. *Social Network Analysis and Mining*, 3(2):179–192, Jun 2013.

[24] E. F. Grove, M.-Y. Kao, P. Krishnan, and J. S. Vitter. Online perfect matching and mobile computing. In *Workshop on Algorithms and Data Structures*, pages 194–205. Springer, 1995.

[25] A. Gu, A. Gupta, and A. Kumar. The power of deferral: maintaining a constant-competitive Steiner tree online. *SIAM Journal on Computing*, 45(1):1–28, 2016.

[26] X. Guo, J. Kulkarni, S. Li, and J. Xian. The Power of Recourse: Better Algorithms for Facility Location in Online and Dynamic Models. *arXiv preprint arXiv:2002.10658*, 2020.

[27] A. Gupta, R. Krishnaswamy, A. Kumar, and D. Panigrahi. Online and Dynamic Algorithms for Set Cover. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 537–550, New York, NY, USA, 2017. ACM.

[28] A. Gupta and A. Kumar. Online Steiner tree with deletions. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 455–467. SIAM, 2014.

[29] A. Gupta, A. Kumar, and C. Stein. Maintaining assignments online: Matching, scheduling, and flows. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 468–479. SIAM, 2014.

[30] A. Gupta and R. Levin. Fully-dynamic submodular cover with bounded recourse. *arXiv preprint arXiv:2009.00800*, 2020.

[31] A. Gupta and R. Levin. The Online Submodular Cover Problem. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1525–1537. SIAM, 2020.

[32] N. J. Harvey, J. Nelson, and K. Onak. Sketching and streaming entropy via approximation theory. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 489–498. IEEE, 2008.

[33] D. Hochbaum. Approximation Algorithms for the Set Covering and Vertex Cover Problems. *SIAM Journal on Computing*, 11(3):555–556, 1982.

[34] M. Imase and B. M. Waxman. Dynamic Steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4(3):369–384, 1991.

[35] R. Iyer, N. Khargoankar, J. Bilmes, and H. Asanani. Submodular combinatorial information measures with applications in machine learning. *arXiv preprint arXiv:2006.15412*, 2020.

[36] T. Izumi, T. Izumi, H. Ono, and K. Wada. Approximability and inapproximability of the minimum certificate dispersal problem. *Theoretical Computer Science*, 411(31):2773 – 2783, 2010.

[37] A. Jakulin. *Machine learning based on attribute interactions*. PhD thesis, Univerza v Ljubljani, 2005.

[38] S. Jorgensen, R. H. Chen, M. B. Milam, and M. Pavone. The risk-sensitive coverage problem: Multi-robot routing under uncertainty with service level and survival constraints. In *Decision and Control (CDC), 2017 IEEE 56th Annual Conference on*, pages 925–932. IEEE, 2017.

[39] G. Kortsarz and Z. Nutov. Approximating source location and star survivable network problems. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 203–218. Springer, 2015.

[40] A. Krause, H. B. McMahan, C. Guestrin, and A. Gupta. Robust submodular observation selection. *Journal of Machine Learning Research*, 9(Dec):2761–2801, 2008.

[41] J. Łącki, J. Oćwieja, M. Pilipczuk, P. Sankowski, and A. Zych. The power of dynamic distance oracles: Efficient dynamic algorithms for the steiner tree. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 11–20, 2015.

[42] S. Lee, M. Purohit, and B. Saha. Firewall placement in cloud data centers. In *Proceedings of the 4th Annual Symposium on Cloud Computing*, SOCC '13, pages 52:1–52:2, New York, NY, USA, 2013. ACM.

[43] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran. Mingen: Minimal generator set selection for small signal stability in power systems: A submodular framework. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 4122–4129, Dec 2016.

[44] G. Loukides and R. Gwadera. Limiting the diffusion of information by a selective pagerank-preserving approach. In *Data Science and Advanced Analytics (DSAA), 2016 IEEE International Conference on*, pages 90–99. IEEE, 2016.

[45] T. Lukovszki, M. Rost, and S. Schmid. Approximate and incremental network function placement. *Journal of Parallel and Distributed Computing*, 120:159 – 169, 2018.

[46] A. D. Mafuta and T. Walingo. Spatial relay node placement in wireless sensor networks. In *2016 IEEE 83rd Vehicular Technology Conference (VTC Spring)*, pages 1–5, May 2016.

[47] S. Phillips and J. Westbrook. Online load balancing and network flow. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 402–411, 1993.

[48] M. A. Rahimian and V. M. Preciado. Detection and isolation of failures in directed networks of lti systems. *IEEE Trans. Control of Network Systems*, 2(2):183–192, 2015.

[49] P. Sanders, N. Sivadasan, and M. Skutella. Online scheduling with bounded migration. *Mathematics of Operations Research*, 34(2):481–498, 2009.

[50] Y. Seldin and A. Slivkins. One Practical Algorithm for Both Stochastic and Adversarial Bandits. In *ICML*, pages 1287–1295, 2014.

[51] M. Sellke. Chasing convex bodies optimally. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1509–1518. SIAM, 2020.

[52] M. Skutella and J. Verschae. A robust PTAS for machine covering and packing. In *European Symposium on Algorithms*, pages 36–47. Springer, 2010.

[53] G. Tong, W. Wu, P. M. Pardalos, and D.-Z. Du. On positive-influence target-domination. *Optimization Letters*, 11(2):419–427, 2017.

[54] C. Tsallis. Possible generalization of Boltzmann-Gibbs statistics. *Journal of statistical physics*, 52(1-2):479–487, 1988.

[55] V. Tzoumas, M. A. Rahimian, G. J. Pappas, and A. Jadbabaie. Minimal actuator placement with bounds on control effort. *IEEE Transactions on Control of Network Systems*, 3(1):67–78, 2016.

[56] Z. Wang, B. Moran, X. Wang, and Q. Pan. An accelerated continuous greedy algorithm for maximizing strong submodular functions. *Journal of Combinatorial Optimization*, 30(4):1107–1124, 2015.

[57] Z. Wang, X. Wang, Y. Liang, and Q. Pan. Weapon target assignment leveraging strong submodularity. In *2013 IEEE International Conference on Information and Automation (ICIA)*, pages 74–79. IEEE, 2013.

[58] J. Westbrook. Load balancing for response time. *Journal of Algorithms*, 35(1):1–16, 2000.

[59] L. A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

[60] P. Wu, G. Chen, X. Zhu, and X. Wu. Minimizing receivers under link coverage model for device-free surveillance. *Computer Communications*, 63:53 – 64, 2015.

[61] Y. Yang, L. Chen, W. Dong, and W. Wang. Active base station set optimization for minimal energy consumption in green cellular networks. *IEEE Transactions on Vehicular Technology*, 64(11):5340–5349, Nov 2015.

[62] Z. Zheng, Y. Peng, F. Wu, S. Tang, and G. Chen. Trading data in the crowd: Profit-driven data acquisition for mobile crowd-sensing. *IEEE Journal on Selected Areas in Communications*, 35(2):486–501, 2017.

Appendix

We show that Shannon entropy also works as a potential, albeit with the weaker recourse bound of:

$$O\left(\frac{\sum_t g_t(\mathcal{N})}{f_{\min}} \ln\left(\frac{c_{\max}}{c_{\min}} \cdot \frac{f_{\max}}{f_{\min}}\right)\right).$$

Define the Shannon Entropy potential to be the expression:

$$\Phi_1(f,\pi) := \sum_{i \in N} \mathfrak{F}_\pi(\pi_i) \ln \frac{c(\pi_i)}{\mathfrak{F}_\pi(\pi_i)}.$$

In order to ensure that Φ_1 remains nonnegative and monotone in each $\mathfrak{F}_\pi(\pi_i)$, scale c by $1/c_{\min}$ and f by $1/(e \cdot f_{\max})$ such that all costs are greater than 1 and all coverages are less than $1/e$. We will account for this scaling at the end.

Note that Φ_1 is Φ_h from Section III with $h(x) = x\log(1/x)$. This h satisfies properties i, ii and iv but not iii.

Properties of Φ_1:

I) Φ_1 increases by at most $g_t(\mathcal{N}) \cdot \ln(c_{\max}/f_{\min})$ with the addition of function g_t to the active set.

II) Φ_1 does not increase with deletion of functions from the system.

III) Φ_1 does not increase during swaps.

IV) If $\gamma > e$, then Φ_1 decreases by at least $f_{\min}\ln(\gamma/e)$ with every γ-move.

The proofs that Φ_1 satisfies properties I to III follows directly from Lemma III.3, since these do not use property iii. It remains to show the last property.

Lemma A.1. *If $\gamma > e$, every γ-move decreases Φ_1 by at least $f_{\min} \cdot \ln(\gamma/e)$.*

Proof: Suppose we perform a γ-move on a permutation π. Let u be the element moving to some position p from some position $q > p$, and let π' denote the permutation after the move. For convenience, also define:

$$v_i := \mathfrak{F}_\pi(\pi_i), \qquad \text{(the original coverage of the } i^{th} \text{ set)}$$
$$a_i := \mathcal{I}_f(\pi_i; u \mid \pi_{1:i-1}) = \mathfrak{F}_\pi(\pi_i) - \mathfrak{F}_{\pi'}(\pi_i). \qquad \text{(the loss in coverage of the } i^{th} \text{ set)}$$

Then:

$$\begin{aligned}
&\Phi_1(f,\pi') - \Phi_1(f,\pi) \\
&= \sum_{i=1}^{n} (v_i - a_i) \ln \frac{c(\pi_i)}{v_i - a_i} + \sum_{i=1}^{n} a_i \ln \frac{c(u)}{\sum_{i=1}^{n} a_i} \\
&\quad - \sum_{i=1}^{n} v_i \ln \frac{c(\pi_i)}{v_i} \\
&\le - \sum_{i=1}^{n} a_i \ln \left(\frac{c(\pi_i)}{e \cdot v_i} \right) + \sum_{i=1}^{n} a_i \ln \frac{c(u)}{\sum_{i=1}^{n} a_i} \qquad \text{(A.1)} \\
&\le - \sum_{i=1}^{n} a_i \ln \left(\frac{\gamma}{e} \cdot \frac{c(u)}{\sum_i a_i} \right) + \sum_{i=1}^{n} a_i \ln \frac{c(u)}{\sum_{i=1}^{n} a_i} \qquad \text{(A.2)} \\
&= - \sum_{i=1}^{n} a_i \ln \left(\frac{\gamma}{e} \right) \\
&\le - f_{\min} \cdot \ln \left(\frac{\gamma}{e} \right).
\end{aligned}$$

Step (A.1) follows because, by concavity of the function $h(x) = x \log x$, we have $h(a+b) - h(a) \le b \cdot h'(a)$. Step (A.2) follows because u moving to position p is a γ-move, hence $\sum_j a_j / c(u) \ge \gamma \cdot v_i / c(\pi_i)$. ■

We now show the weaker recourse bound. By I, the potential Φ_h increases by at most $g_t(\mathcal{N}) \cdot \ln(c_{\max}/f_{\min})$ with the addition of function g_t to the active set. By IV, it decreases by $\Omega(f_{\min})$ with every move that costs recourse 1, and otherwise does not increase. Since we scaled costs by $1/c_{\min}$ and coverages by $1/(e \cdot f_{\max})$, this implies a recourse bound of:

$$O\left(\frac{\sum_t g_t(\mathcal{N})}{f_{\min}} \ln \left(\frac{c_{\max}}{c_{\min}} \cdot \frac{f_{\max}}{f_{\min}} \right) \right).$$