Online Submodular Maximization with Preemption

Niv Buchbinder[*]   Moran Feldman[†]   Roy Schwartz[‡]

January 26, 2015

Abstract

Submodular function maximization has been studied extensively in recent years under various constraints and models. The problem plays a major role in various disciplines. We study a natural online variant of this problem in which elements arrive one-by-one and the algorithm has to maintain a solution obeying certain constraints at all times. Upon arrival of an element, the algorithm has to decide whether to accept the element into its solution and may preempt previously chosen elements. The goal is to maximize a submodular function over the set of elements in the solution.

We study two special cases of this general problem and derive upper and lower bounds on the competitive ratio. Specifically, we design a $1/e$-competitive algorithm for the unconstrained case in which the algorithm may hold any subset of the elements, and constant competitive ratio algorithms for the case where the algorithm may hold at most k elements in its solution.

1 Introduction

Submodular function maximization has been studied extensively in recent years under various constraints and models. Submodular functions, which capture well the intuitive idea of diminishing returns, play a major role in various disciplines, including combinatorics, optimization, economics, information theory, operations research, algorithmic game theory and machine learning. Submodular maximization captures well known combinatorial optimization problems such as: Max-Cut [32, 36, 39, 40, 53], Max-DiCut [20, 32, 34], Generalized Assignment [11, 12, 24, 29], and Max-Facility-Location [2, 14, 15]. Additionally, one can find submodular maximization problems in many other settings. In machine learning, maximization of submodular functions has been used for document summarization [46, 47], sensor placement [41, 42, 44], and information gathering [43]. In algorithmic game theory, calculating market expansion [16] and computing core values of certain types of games [52] are two examples where the problem can be reduced to submodular maximization.

It is natural to consider online variants of submodular maximization, such as the following setting. There is an unknown ground set of elements $\mathcal{N} = \{u_1, u_2, \ldots, u_n\}$, a non-negative submodular function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ and perhaps a constraint determining the feasible sets of elements that may be chosen. The elements of $\mathcal{N}$ arrive one by one in an online fashion. Upon arrival, the online algorithm must decide whether to accept each revealed element into its solution and this decision is irrevocable. As in the offline case, the algorithm has access to the function f via a value oracle, but in the online case it may only query subsets of elements that have already been revealed. For a concrete example, consider a soccer team manager preparing his team for the 2014 World Cup and hiring k (= 11) players. Occasionally, the manager gets an additional application from a new potential player, and may recruit him to the team. The goal is to maximize the quality of the team. The quality of the team is indeed a (non-monotone) submodular function of the chosen players.

Although natural, it is not difficult to see that no algorithm has a constant competitive ratio for the above naïve model even with a simple cardinality constraint. Therefore, in order to obtain meaningful results, one must relax the model. One possible relaxation is to consider a random arrival model in which elements arrive at a random order [7, 26, 33]. This relaxation leads to "secretary type" algorithms. We propose here a different approach that still allows for an adversarial arrival. Here, we allow the algorithm to reject (preempt) previously accepted elements. Preemption appears as part of many online models (see, *e.g.*, [1, 17, 18, 30]). The use of preemption in our model is mathematically elegant and fits well with natural scenarios. For example, it is natural to assume that a team manager may release one of the players (while hiring another instead) if it benefits the quality of the team.

1.1 Our Results

We study online submodular maximization problems with two simple constraints on the feasible sets. The unconstrained case in which no restrictions are inflicted on the sets of elements that the algorithm may choose, and the cardinality constraint in which the online algorithm may hold at any time at most k elements. We provide positive results as well as hardness results. Our hardness results apply both to polynomial and non-polynomial time algorithms, but all our algorithms are polynomial (in some of them, a polynomial time implementation losses an ε in the competitive ratio).

The unconstrained case: The first special case we consider is, arguably, the most basic case in which there is no constraint on the sets of elements that may be chosen. In this case the problem is

trivial if the submodular function f is monotone[1] as the algorithm may take all elements. Therefore, we only consider non-monotone functions f. One simple candidate algorithm is an algorithm that selects each revealed element independently with probability $1/2$. Feige et al. [23] proved that this algorithm, which does not use preemption, is $1/4$-competitive. We prove that this is essentially optimal without using the additional power of preemption.

Theorem 1.1. *An online algorithm using no preemption cannot be $(1/4 + \varepsilon)$-competitive for any constant $\varepsilon > 0$, even if f is guaranteed to be a cut function of a weighted directed graph.*

On the positive side we prove that preemption is beneficial for the `unconstrained-model`.

Theorem 1.2. *There exists a $1/e$-competitive algorithm for the `unconstrained-model`, which can be implemented in polynomial time at the cost of an ε loss in the competitive ratio (for an arbitrary small constant $\varepsilon > 0$).*

A special case of the `unconstrained-model` that we study is the `dicut-model`. In this model every set is feasible and the objective function f is a cut function of a weighted directed graph $G = (V, A)$ having $\mathcal{N} \subseteq V$ (*i.e.*, a subset of the nodes of G form the ground set). We assume an algorithm for the `dicut-model` knows V (but not $\mathcal{N}$) and can query the weight (and existence) of every arc leaving a *revealed* node. Observe that the algorithm can in fact calculate the value of f for every set of revealed elements using this information and never needs to query f via the oracle.

The `dicut-model` can be viewed as an online model of the well-known Max-DiCut problem (see Section 1.2 for a discussion of another online model of Max-DiCut). Additionally, since the `dicut-model` is a special case of the `unconstrained-model` (with more power for the algorithm), it inherits the positive result given by Theorem 1.2. The next theorem gives a stronger result.

Theorem 1.3. *There exists a polynomial time 0.438-competitive algorithm for the `dicut-model`.*

Theorem 1.3 is proved by showing that an offline algorithm suggested by [21] can be implemented under the `dicut-model`. We complement Theorems 1.2 and 1.3 by the following theorem which gives hardness results for the `dicut-model` (and thus, also for the `unconstrained-model`).

Theorem 1.4. *No randomized (deterministic) algorithm for the `dicut-model` has a competitive ratio better than $4/5$ $(\frac{5-\sqrt{17}}{2} \approx 0.438)$.[2]*

Notice that for polynomial time algorithms a hardness result of $1/2$ proved by [23] for offline algorithms extends immediately to the `unconstrained-model`. For the `dicut-model` there exists a polynomial time 0.874-approximation offline algorithm [45], thus, polynomial time offline algorithms are strictly stronger than online algorithms in this model.

Cardinality constraint: The second case we consider is the `cardinality-model` in which a set is feasible if and only if its size is at most k, for some parameter $0 \leq k \leq n$. Our positive results for the `cardinality-model` are summarized by the following theorem.

Theorem 1.5. *There exists a randomized $(56/627 \approx 0.0893)$-competitive algorithm for the `cardinality-model`, which can be implemented in polynomial time at the cost of an ε loss in the competitive ratio (for an arbitrary small constant $\varepsilon > 0$). Moreover, if the objective function is restricted to be monotone, then there exists a polynomial time deterministic $(1/4)$-approximation algorithm for this model.*

---

[1] A set function $f : 2^{\mathcal{N}} \to \mathbb{R}$ is monotone if $A \subseteq B \subseteq \mathcal{N}$ implies $f(A) \leq f(B)$.
[2] Theorem 1.4 holds even if we allow the algorithm access to all arcs of G, including arcs leaving unrevealed elements.

	Monotone Objective		General Objective	
	Deterministic	Randomized	Deterministic	Randomized
Algorithm	$1/4$ [10, 54]	$1/4$ [10, 54]	−	$\frac{56}{627} \approx 0.0893$ (1.5)
Hardness	$1/2 + \varepsilon$ (1.6)	$3/4 + \varepsilon$ (1.6)	$\frac{5-\sqrt{17}}{2} \approx 0.438$ (1.4)	$1/2 + \varepsilon$ (1.6)
Polynomial time hardness	$1/2 + \varepsilon$ (1.6)	$1 - 1/e$ [50]	$\frac{5-\sqrt{17}}{2} \approx 0.438$ (1.4)	0.491 [31]

Table 1: Summary of the results for the `cardinality-model`. To the right of each result appears the number of the theorem (or references) proving it.

The second part of Theorem 1.5 follows from the works of Ashwinkumar [54] and Chakrabarti and Kale [10]. More specifically, these works present a streaming 1/4-competitive algorithm for maximizing a monotone submodular function subject to a cardinality constraint,[3] and this algorithm can be implemented also under the `cardinality-model`. We describe a different 1/4-competitive algorithm for monotone objectives under the `cardinality-model`, and then use additional ideas to prove the first part of Theorem 1.5.

It is interesting to note that both algorithms guaranteed by Theorem 1.5 can be implemented in the streaming model of Chakrabarti and Kale [10]. Thus, we give also the first algorithm for maximizing a general non-negative submodular function subject to a cardinality constraint under this streaming model.

On the negative side, notice that the `cardinality-model` generalizes the `unconstrained-model` (by setting $k = n$). Hence, both hardness results given by Theorem 1.4 extend to the `cardinality-model`. The following theorem gives a few additional hardness results for this model.

Theorem 1.6. *No algorithm for the `cardinality-model` is $(1/2+\varepsilon)$-competitive for any constant $\varepsilon > 0$. Moreover, even if the objective function is restricted to be monotone, no randomized (deterministic) algorithm for the `cardinality-model` is $(3/4+\varepsilon)$-competitive $((1/2+\varepsilon)$-competitive).*

Notice that polynomial time hardness results of 0.491 and $1 - 1/e$ for the `cardinality-model` and the `cardinality-model` with a monotone objective follow from [31] and [50], respectively. All the results for the `cardinality-model` are summarized in Table 1.

1.2 Related Work

The literature on submodular maximization problems is very large, and therefore, we mention below only a few of the most relevant works. The classical result of [51] states that the simple discrete greedy algorithm is a $(1 - 1/e)$-approximation algorithm for maximizing a monotone submodular function subject to a cardinality constraint. This result is known to be tight [50], even in the case where the objective function is a coverage function [19]. However, when one considers submodular objectives which are not monotone, less is known. An approximation of 0.309 was given by [56], which was later improved to 0.325 [31] using a simulated annealing technique. Extending the continuous greedy algorithm of [9] to general non-negative submodular objectives, [28] obtained an improved approximation of $1/e - o(1)$. Finally, [25] gave a fast $1/e$-approximation algorithm and a much slower $(1/e + 0.004)$-approximation algorithm, demonstrating that $1/e$ is not the right approximation ratio for the problem. On the hardness side, it is known that no polynomial time algorithm can have an approximation ratio better than 0.491 [31].

---

[3]In fact the algorithm of [54] and [10] is $(4m)^{-1}$-competitive for the more general constraint of m-matroids intersection.

For the problem of maximizing a non-negative submodular function subject to no constraints, the first approximation algorithm was presented by Feige et al. [23] who gave a 2/5-approximation. This was improved in [31] and [27] to 0.41 and 0.42, respectively. Finally, [8] described a 1/2-approximation linear time algorithm, matching the hardness result given by [23]. Huang and Borodin [37] study an online model of this problem where the algorithm can access both f and $\bar{f}$. This model is powerful enough to implement the randomized 1/2-approximation algorithm of [8], and therefore, [37] consider only deterministic algorithms. Azar et al. [3] consider the Submodular Max-SAT problem, and provide a 2/3-approximation algorithm for it that can be implemented also in a natural online model. We are not aware of any additional works on online models of submodular maximization.

For the problem of Max-DiCut, Goemans and Williamson [32] obtained 0.796-approximation using semi-definite programming. This was improved through a series of works [20, 45, 48] to 0.874. On the hardness side, a $(12/13 + \varepsilon)$-approximation algorithm will imply $\mathsf{P} = \mathsf{NP}$ [36]. Assuming the Unique Games Conjecture, the best possible approximation for Max-Cut is 0.878 [40, 49], and this hardness result holds also for Max-DiCut since the last generalizes Max-Cut.

Bar-Noy and Lampis [6] gave a (1/3)-competitive deterministic algorithm for an online model of Max-Cut where every revealed node is accompanied by its input and output degrees. For the case of a directed acyclic graph, they provide an improved deterministic algorithm with a competitive ratio of $2/3^{1.5} \approx 0.385$, which is optimal against an adaptive adversary. Huang and Borodin [37] notice that the (1/3)-competitive deterministic algorithm of [6] is in fact identical to the (1/3)-approximation deterministic algorithm of [8] for unconstrained submodular maximization. Using the same ideas, it is not difficult to show that the (1/2)-approximation randomized algorithm of [8] implies a (1/2)-competitive algorithm in this online model. Finally, Feige and Jozeph [21] consider oblivious algorithms for Max-DiCut—algorithms in which every node is selected into the cut independently with a probability depending solely on its input and output degrees. They show a 0.483-approximation oblivious algorithm, and prove that no oblivious algorithm has an approximation ratio of 0.4899.

Finally, the vast literature on buyback problems considers problems that are similar to our model, but assume a linear objective function. Many of these problems are non-trivial only when preemption has a cost, which is usually assumed to be either constant or linear in the value of the preempted element. The work from this literature most closely related to ours is the work of Babaioff et al. [4, 5] who considered a matroid constraint. For other buyback results see, *e.g.*, [13, 35, 38, 54, 55].

Paper Organization. Section 2 defines additional notation. Section 3 gives our results for the `unconstrained-model` and the `dicut-model`, except for Theorem 1.1 whose proof is deferred to Appendix A. Finally, Section 4 describes our results for the `cardinality-model`.

2 Preliminaries

We study the following online variant of the submodular maximization problem. There is an unknown ground set of elements $\mathcal{N} = \{u_1, u_2, \ldots, u_n\}$, a non-negative submodular function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ and and a down-monotone[4] collection of feasible sets $\mathcal{I} \subseteq 2^{\mathcal{N}}$. The objective of the instance is to find a feasible set maximizing f. The elements of $\mathcal{N}$ are revealed one by one. The algorithm creates $n + 1$ feasible solutions: $S_0, S_1, \ldots, S_n$ (each $S_i \in \mathcal{I}$). The solution S_0 is the empty set $\varnothing$. For every $1 \leq i \leq n$, S_i is the solution selected by the algorithm immediately after

---

[4]A collection $\mathcal{I}$ of sets is *down-monotone* if $A \subseteq B \subseteq \mathcal{N}$ and $B \in \mathcal{I}$ imply $A \in \mathcal{I}$.

element u_i is revealed and the algorithm can choose it to be any feasible subset of $S_{i-1} + u_i{}^5$. It is important to note that the algorithm does not know n (the size of the ground set) in advance, hence, the input might end after every element from the algorithm's point of view.

3 The `unconstrained-model` and the `dicut-model`

Our positive results for the `unconstrained-model` and the `dicut-model` (*i.e.*, Theorems 1.2 and 1.3) are proved in Section 3.1. The negative result for the `dicut-model` (Theorem 1.4), which applies also to the `unconstrained-model`, is proved in Section 3.2.

3.1 Algorithms for the `unconstrained-model` and the `dicut-model`

Before describing our algorithm for the `unconstrained-model`, we need some notation. For two vectors $x, y \in [0, 1]^{\mathcal{N}}$, we use $x \vee y$ and $x \wedge y$ to denote the coordinate-wise maximum and minimum, respectively, of x and y (formally, $(x \vee y)_u = \max\{x_u, y_u\}$ and $(x \wedge y)_u = \min\{x_u, y_u\}$). We abuse notation both in the description of the algorithm and in its analysis, and unify a set with its characteristic vector and an element with the singleton set containing it. The *multilinear* extension of a set function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ is a function $F : [0, 1]^{\mathcal{N}} \to \mathbb{R}^+$ defined by $F(x) = \mathbb{E}[f(\mathbb{R}(x))]$, where $\mathbb{R}(x)$ is a random set containing every element $u \in \mathcal{N}$ with probability x_u, independently. The multilinear extension is an important tool used in many previous works on submodular maximization (see, *e.g.*, [9, 28, 56]). We denote by $\partial_u F(x)$ the derivative of F at point x with respect to the coordinate corresponding to u. It can be checked that F is a multilinear function, and thus:

$$\partial_u F(x) = F(x \vee u) - F(x \wedge (\mathcal{N} - u)) \ .$$

Consider Algorithm 1. Recall that S_i is the solution that the algorithm produces after seeing element u_i.

Algorithm 1: Marginal Choice
---
1 foreach *element u_i revealed* **do**
2   Choose a uniformly random threshold $\theta_i \in [0, 1]$.
3   Let $\mathcal{N}_i \leftarrow \{u_1, u_2, \ldots, u_i\}$.
4   Let $S_i \leftarrow \{u_j \in \mathcal{N}_i \mid \partial_{u_j} F(\theta_j \cdot \mathcal{N}_i) \geq 0\}$.

Our first objective is to show that Algorithm 1 is an online algorithm according to the `unconstrained-model`.

Lemma 3.1. *For every* $1 \leq i \leq n$, $S_i \subseteq S_{i-1} + u_i$.

Proof. By definition, S_i contains only elements of $\mathcal{N}_i$. Fix an element $u_j \in \mathcal{N}_i - u_i$. Then:

$$u_j \in S_i \Rightarrow \partial_{u_j} F(\theta_j \cdot \mathcal{N}_i) \geq 0 \Rightarrow \partial_{u_j} F(\theta_j \cdot \mathcal{N}_{i-1}) \geq 0 \Rightarrow u_j \in S_{i-1} \ ,$$

where the second derivation follows from submodularity. $\qquad\square$

Next, we bound the competitive ratio of Algorithm 1. Fix an element $u_i \in \mathcal{N}$. By submodularity, $g(z) = \partial_{u_i} F(z \cdot \mathcal{N})$ is a continuous non-increasing function of z. Hence, by the intermediate value

---
[5]Given a set S and an element u, we use $S + u$ and $S - u$ as shorthands for $S \cup \{u\}$ and $S \setminus \{u\}$, respectively.

theorem, one of the following must hold: $g(z)$ is always positive in the range $[0, 1]$, $g(z)$ is always negative in the range $[0, 1]$ or $g(z)$ has at least one root $z_0 \in [0, 1]$. In the last case, the set $I_0 \subseteq [0, 1]$ of the roots of $g(z)$ is non-empty. Moreover, by the monotonicity and continuity of $g(z)$, I_0 is a closed range. Using these observations, we define a vector $y \in [0, 1]^{\mathcal{N}}$ as follows:

$$
y_{u_i} = \begin{cases} 0 & \text{if } \partial_{u_i} F(0 \cdot \mathcal{N}) < 0 \ , \\ 1 & \text{if } \partial_{u_i} F(1 \cdot \mathcal{N}) > 0 \ , \\ \max I_0 & \text{otherwise } . \end{cases}
$$

Remark: Notice that the case $\partial_{u_i} F(0 \cdot \mathcal{N}) < 0$ in the definition of y can happen only when $f(\varnothing) > 0$.

Observation 3.2. *Every element $u_i \in \mathcal{N}$ belongs to S_n with probability y_{u_i}, independently. Hence, $\mathbb{E}[f(S_n)] = F(y)$.*

Proof. An element $u_i \in \mathcal{N}$ belongs to S_n if and only if $\partial_{u_i} F(\theta_i \cdot \mathcal{N}) \geq 0$, which is equivalent to the condition $\theta_i \leq y_{u_i}$. Clearly, the last condition happens with probability y_{u_i}, and is independent for different elements. $\qquad\square$

The last observation implies that analyzing Algorithm 1 is equivalent to lower bounding $F(y)$.

Lemma 3.3. *For every $\lambda \in [0, 1]$, $F(y \wedge (\lambda \cdot \mathcal{N})) \geq F(\lambda \cdot \mathcal{N})$.*

Proof. Observe that:

$$
F(y \wedge (\lambda \cdot \mathcal{N})) = f(\varnothing) + \int_0^\lambda \frac{dF(y \wedge (z \cdot \mathcal{N}))}{dz} dz = f(\varnothing) + \int_0^\lambda \sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(y \wedge (z \cdot \mathcal{N})) dz \ ,
$$

where the second equality is due to the chain rule. By submoduliry and the observation that $\partial_u F(z \cdot \mathcal{N})$ is non-positive for every $z > y_u$, we get:

$$
\sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(y \wedge (z \cdot \mathcal{N})) \geq \sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(z \cdot \mathcal{N}) \geq \sum_{u \in \mathcal{N}} \partial_u F(z \cdot \mathcal{N}) \ .
$$

Combining the above equality and inequality, and using the chain rule again, gives:

$$
F(y \wedge (\lambda \cdot \mathcal{N})) \geq f(\varnothing) + \int_0^\lambda \left[\sum_{u \in \mathcal{N}} \partial_u F(z \cdot \mathcal{N}) \right] dz = f(\varnothing) + \int_0^\lambda \frac{dF(z \cdot \mathcal{N})}{dz} dz = F(\lambda \cdot \mathcal{N}) \ . \quad \square
$$

We also need a lemma proved by [28].

Lemma 3.4 (Lemma III.5 of [28]). *Consider a vector $x \in [0, 1]^{\mathcal{N}}$. Assuming $x_u \leq a$ for every $u \in \mathcal{N}$, then for every set $S \subseteq \mathcal{N}$, $F(x \vee S) \geq (1 - a) f(S)$.*

The following corollary follows from the last two lemmata.

Corollary 3.5. *$F(y) \geq e^{-1} \cdot F(OPT)$, where OPT is the subset of $\mathcal{N}$ maximizing $f(OPT)$.*

Proof. Fix two values $0 \leq a \leq b \leq 1$. By the chain rule:

$$F(y \wedge (b \cdot \mathcal{N})) - F(y \wedge (a \cdot \mathcal{N})) = \int_a^b \frac{dF(y \wedge (z \cdot \mathcal{N}))}{dz} dz \qquad (1)$$

$$= \int_a^b \sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(y \wedge (z \cdot \mathcal{N})) dz \geq \int_a^b \sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(z \cdot \mathcal{N}) dz \enspace ,$$

where the inequality follows from submodularity. We use two ways to lower bound the rightmost hand side of the above inequality. First, observe that $\partial_u F(z \cdot \mathcal{N}) \geq 0$ whenever $z < y_u$. Thus, $F(y \wedge (z \cdot \mathcal{N}))$ is a non-decreasing function of z. Additionally, $\partial_u F(z \cdot \mathcal{N}) \leq 0$ whenever $z > y_u$ and $\partial_u F(z \cdot \mathcal{N}) \geq 0$ whenever $z < y_u$. This allows us to derive the second lower bound, which holds whenever z is not equal to any coordinate of y (*i.e.*, for every value of z except for a finite set of values):

$$\sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(z \cdot \mathcal{N}) \geq \sum_{u \in OPT} \partial_u F(z \cdot \mathcal{N}) = \frac{\sum_{u \in OPT} [F(u \vee (z \cdot \mathcal{N})) - F(z \cdot \mathcal{N})]}{1 - z}$$

$$\geq \frac{F(OPT \vee (z \cdot \mathcal{N})) - F(z \cdot \mathcal{N})}{1 - z} \geq f(OPT) - \frac{F(y \wedge (z \cdot \mathcal{N}))}{1 - z} \enspace ,$$

where the last inequality follows from Lemmata 3.3 and 3.4. Plugging this lower bound and $a = 0$ into (1) gives:

$$F(y \wedge (b \cdot \mathcal{N})) \geq f(\varnothing) + \int_0^b \left[f(OPT) - \frac{F(y \wedge (z \cdot \mathcal{N}))}{1 - z} \right] dz \enspace .$$

The solution of this differential equation is:

$$F(y \wedge (b \cdot \mathcal{N})) \geq -(1 - b) \cdot \ln(1 - b) \cdot f(OPT) \enspace .$$

The rightmost hand side of the last inequality is maximized for $b = 1 - e^{-1}$. Recalling that $F(y \wedge (z \cdot \mathcal{N}))$ is a non-decreasing function of z, we get:

$$F(y) = F(y \wedge (1 \cdot \mathcal{N})) \geq F(y \wedge ((1 - e^{-1}) \cdot \mathcal{N})) \geq e^{-1} \cdot f(OPT) \enspace . \qquad \square$$

The last corollary completes the proof of the first part of Theorem 1.2. A naïve implementation of Algorithm 1 requires an exponential time to evaluate F. However, for every constant $c > 0$ it is possible to approximate, in polynomial time, $\partial_{u_j} F(\theta_j \cdot \mathcal{N}_i)$ with probability $1 - c\varepsilon i^{-2}$ up to an additive error of $c\varepsilon i^{-2} \cdot f(OPT)$ using sampling (see [9] for an example of a similar estimate).[6] For a small enough c, such an approximation is good enough to affect the competitive ratio of the algorithm by only an ε. To keep the algorithm within the `unconstrained-model`, the samples used to approximate $\partial_{u_j} F(\theta_j \cdot \mathcal{N}_i)$ for different values of i need to be correlated to guarantee that the approximation is a decreasing function of i. For more details, see Appendix B.

The rest of this section considers the `dicut-model` and is devoted for proving Theorem 1.3. Recall that in this model f is the cut function of some weighted directed graph $G = (V, A)$. Let $w_{\text{in}}(u)$ ($w_{\text{out}}(u)$) denote the total weight of the arcs entering (leaving) element u. Throughout this section we assume there are no arcs of G leaving nodes of $V \setminus \mathcal{N}$. Removing such arcs does not affect the value of any solution, and thus, our results hold also without the assumption. Feige and Jozeph [22] proved the following theorem (we rephrased it in our notation).

---

[6]It is not possible to decrease the error to something that depends on n because n is unknown to the algorithm when it calculates these estimations.

Theorem 3.6. *There exists a non-decreasing function* $h : [0,1] \rightarrow [0,1]$ *such that the vector* $y \in [0,1]^{\mathcal{N}}$ *defined by:*

$$y_u = h\left(\frac{w_{\text{out}}(u)}{w_{\text{in}}(u) + w_{\text{out}}(u)}\right)$$

obeys $F(y) \geq 0.483 \cdot f(OPT)$. *Moreover,* h *is independent of* G *and can be computed in constant time (assuming numbers comparison takes constant time).*

Notice that in the last theorem y_u is undefined when $w_{\text{in}}(u) = w_{\text{out}}(u) = 0$. The theorem holds regardless of the value used for y_u in this case because such an element u is isolated. We can now present our algorithm for the `dicut-model` which is depicted as Algorithm 2.

Algorithm 2: Degrees Ratio Choice

---
1 **foreach** *element* u_i *revealed* **do**
2 $\quad$ Choose a uniformly random threshold $\theta_i \in [0,1]$.
3 $\quad$ Initialize $S_i \leftarrow \varnothing$.
4 $\quad$ **for** $j = 1$ **to** i **do**
5 $\quad\quad$ Let $w_{\text{in}}(u_j, i)$ denote the total weight of arcs from *revealed* elements to u_j.
6 $\quad\quad$ **if** $w_{\text{in}}(u_j, i) + w_{\text{out}}(u_j) > 0$ **then** Let $y_{u_j}(i) \leftarrow h\left(\frac{w_{\text{out}}(u_j)}{w_{\text{in}}(u_j, i) + w_{\text{out}}(u_j)}\right)$. **else** Let $y_{u_j}(i) \leftarrow 1$.
7 $\quad\quad$ **if** $y_{u_j}(i) \geq \theta_j$ **then** Add u_j to S_i.

Observation 3.7. *The competitive ratio of Algorithm 2 is at least* 0.483.

Proof. Notice that the vector $y(n)$ defined by the algorithm is equal to the vector y defined by Theorem 3.6 (wherever the last is defined). Notice also that every element $u \in \mathcal{N}$ belongs to S_n with probability $y_u(n)$, independently. Hence, $\mathbb{E}[f(S_n)] = F(y)$. The observation now follows from Theorem 3.6. $\qquad\square$

To complete the proof of Theorem 1.3, we only need the following lemma which shows that Algorithm 2 is an online algorithm of the `dicut-model`.

Lemma 3.8. *For every* $1 \leq i \leq n$, $S_i \subseteq S_{i-1} + u_i$.

Proof. For $i = 1$ there is nothing to prove since $S_1 \subseteq \{u_1\}$. Thus, we assume from now on $i \geq 2$. By definition, S_i contains only elements of $\mathcal{N}_i$. Fix an element $u_j \in \mathcal{N}_i - u_i$. We have to show that $u_j \in S_{i-1}$. First, let us prove that:

$$y_{u_j}(i) \leq y_{u_j}(i-1) \ . \tag{2}$$

There are two cases to consider. If $w_{\text{in}}(u_j, i-1) + w_{\text{out}}(u_j) \leq 0$ then $y_{u_j}(i-1) = 1$, which proves (2) since $y_{u_j}(i) \in [0,1]$. Otherwise, since $w_{\text{in}}(u_j, i)$ is a non-decreasing function of i we get $w_{\text{in}}(u_j, i) + w_{\text{out}}(u_j) > 0$. Inequality (2) now follows since h is also non-decreasing. Using (2) we now get:

$$u_j \in S_i \Rightarrow y_{u_j}(i) \geq \theta_j \Rightarrow y_{u_j}(i-1) \geq \theta_j \Rightarrow u_j \in S_{i-1} \ . \qquad\square$$
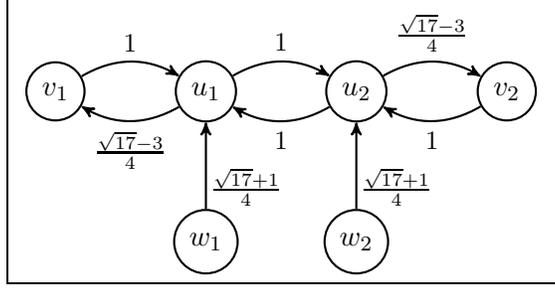
Figure 1: Graph for Lemma 3.9. The number beside each arc represents its weight.

3.2 Hardness results for the `dicut-model`

In this section we prove Theorem 1.4. The proof of the theorem is split between two lemmata. Lemma 3.9 proves the part of Theorem 1.4 referring to deterministic algorithms, and Lemma 3.10 proves the part referring to randomized algorithms. Both lemmata present an absolute graph $G = (V, A)$, and then fix an algorithm ALG and describe an adversary that reveals some of the nodes of V. The choice which nodes to reveal is done iteratively, *i.e.*, the adversary initially reveals some nodes and then can reveal additional nodes based on the decisions of ALG (or the probabilities of various decisions in case ALG is randomized). Formally, in the hard instance for ALG, the set $\mathcal{N}$ is the set of nodes revealed by the adversary, and the hard instance reveals these nodes in the same order they are revealed by the adversary.

Lemma 3.9. *No deterministic online algorithm has a competitive ratio better than $(5 - \sqrt{17})/2$ for the `dicut-model`.*

Proof. Consider the directed graph G describe in Figure 1, and fix a deterministic algorithm ALG for the `dicut-model`. We describe an adversary that reveals some of the nodes of G and forces ALG to be no better than $((5 - \sqrt{17})/2)$-competitive. For every pair of nodes x and y of G, we denote by $c(xy)$ the weight of the arc from x to y. Our adversary begins by revealing u_1 and u_2 and stops if ALG's solution at this point is either empty or equal to $\{u_1, u_2\}$. If ALG's solution is empty then ALG is not competitive at all. On the other hand, if ALG's solution is $\{u_1, u_2\}$ then its solution has a value of:

$$c(u_1 v_1) + c(u_2 v_2) = 2 \cdot \frac{\sqrt{17} - 3}{4} = \frac{\sqrt{17} - 3}{2} \ .$$

On the other hand, the optimal solution is $\{u_1\}$, whose value is:

$$c(u_1 v_1) + c(u_1 u_2) = \frac{\sqrt{17} - 3}{4} + 1 = \frac{\sqrt{17} + 1}{4} \ .$$

Thus, in this case, the competitive ratio of ALG is at most:

$$\frac{\sqrt{17} - 3}{2} : \frac{\sqrt{17} + 1}{4} = \frac{2\sqrt{17} - 6}{\sqrt{17} + 1} = \frac{5 - \sqrt{17}}{2} \ .$$

By the above discussion, the only interesting case is when ALG's solution contains exactly one of the elements u_1 or u_2. Notice that G is symmetric in the sense that switching every node having the index 1 with the corresponding node having the weight 2 does not change the graph. Hence, it is safe to assume ALG's solution is exactly $\{u_1\}$. The next step of the adversary is to reveal v_1. If
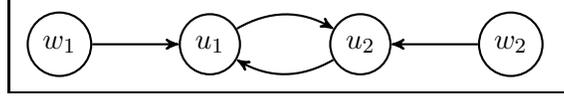
9

Figure 2: Graph for Lemma 3.10. All arcs have an identical weight assumed to be 1.

v_1 enters the solution of ALG, then, regardless of weather u_1 is kept in the solution, the value of ALG's solution is $c(u_1u_2) = c(v_1u_1) = 1$. On the other hand, the optimal solution at this point is $\{u_2, v_1\}$ whose value is:

$$c(u_2u_1) + c(u_2v_2) + c(v_1u_1) = 1 + \frac{\sqrt{17} - 3}{4} + 1 = \frac{\sqrt{17} + 5}{4} \ .$$

Hence, the competitive ratio of ALG is at most:

$$\frac{4}{\sqrt{17} + 5} = \frac{5 - \sqrt{17}}{2} \ .$$

We are left to handle the case in which v_1 does not enter ALG's solution. In this case, the adversary reveals also w_1. ALG's solution at this point must be a subset of $\{u_1, w_1\}$ and every such subset has a value of at most:

$$c(u_1v_1) + c(u_1u_2) = c(w_1u_1) = \frac{\sqrt{17} + 1}{4} \ .$$

On the other hand, the optimal solution at this point is $\{u_2, v_1, w_1\}$ whose value is:

$$c(u_2u_1) + c(u_2v_2) + c(v_1u_1) + c(w_1u_1) = 1 + \frac{\sqrt{17} - 3}{4} + 1 + \frac{\sqrt{17} + 1}{4} = \frac{\sqrt{17} + 3}{2} \ .$$

Hence, the competitive ratio of ALG is at most:

$$\frac{\sqrt{17} + 1}{4} : \frac{\sqrt{17} + 3}{2} = \frac{\sqrt{17} + 1}{2\sqrt{17} + 6} = \frac{7 - \sqrt{17}}{8} < \frac{5 - \sqrt{17}}{2} \ . \qquad \square$$

Lemma 3.10. *No randomized online algorithm has a competitive ratio better than 4/5 for the* `dicut-model`*.*

Proof. Consider the directed graph G describe in Figure 2, and fix an algorithm ALG for the `dicut-model`. We describe an adversary that reveals some of the nodes of G and forces ALG to be no better than 4/5-competitive. The adversary begins by revealing the nodes u_1 and u_2 of the graph. At this point, let p_1 be the probability that u_1 is alone in the solution of ALG and p_2 be the probability that u_2 is alone in this solution. If $p_1 + p_2 \leq 4/5$ then the adversary stops here, and the competitive ratio of ALG is no better than 4/5 since the optimal solution at this point has a value of 1 and the expected value of the ALG's solution is only $p_1 + p_2$. Thus, we may assume from now on $p_1 + p_2 > 4/5$.

By symmetry, we may also assume, without loss of generality, $p_2 \leq p_1$, which implies: $2p_1 \geq p_1 + p_2 > 4/5$. The next step of the adversary is revealing w_1. The optimal solution at this point is $\{u_2, w_1\}$, whose value is 2. Let analyze the best solution ALG can produce. With probability p_1, ALG must pick a subset of $\{u_1, w_1\}$, and thus, cannot end up with a solution of value better than 1. With the remaining probability, ALG produces a solution no better than the optimal solution, *i.e.*, its value is at most 2. Hence, the expected value of ALG's solution is at most:

$$p_1 + 2(1 - p_1) = 2 - p_1 < 2 - 2/5 = 2 \cdot (4/5) \ . \qquad \square$$

4 The `cardinality-model`

Our positive and negative results for the `cardinality-model` (*i.e.*, Theorems 1.5 and 1.6, respectively) are proved in Sections 4.1 and 4.2, respectively.

4.1 Algorithms for the `cardinality-model`

We first consider the special case of the `cardinality-model` where the objective function f is monotone (in addition to being non-negative and submodular). For this case it is possible to derive a 1/4-competitive algorithm from the works of Ashwinkumar [54] and Chakrabarti and Kale [10] on streaming algorithms. We describe an alternative algorithm which achieves the same competitive ratio and has an easier analysis. A variant of this algorithm is applied later in this section to the general `cardinality-model`.

Our alternative algorithm has a parameter $c > 0$ and is given as Algorithm 3. Intuitively the algorithm gets the first k revealed elements unconditionally. For every element u_i arriving after the first k elements, the algorithm finds the best swap replacing an element of the current solution with u_i. If this swap increases the value of the current solution by enough to pass a given threshold, then it is carried out and u_i enters the solution.

Algorithm 3: Greedy with Theshold(c)

1 Let $S_0 \leftarrow \varnothing$.
2 **foreach** *element u_i revealed* **do**
3   **if** $i \leq k$ **then**
4   Let $S_i \leftarrow S_{i-1} + u_i$.
5   **else**
6   Let u'_i be the element of S_{i-1} maximizing $f(S_{i-1} + u_i - u'_i)$.
7   **if** $f(S_{i-1} + u_i - u'_i) - f(S_{i-1}) \geq c \cdot f(S_{i-1})/k$ **then**
8   Let $S_i \leftarrow S_{i-1} + u_i - u'_i$.
9   **else**
10   Let $S_i \leftarrow S_{i-1}$.

Let us define some notation. For every $0 \leq i \leq n$, let $A_i = \cup_{j=1}^{i} S_j$. Informally, A_i is the set of all elements of $\{u_1, u_2, \dots, u_i\}$ originally accepted by Algorithm 3, regardless of whether they are preempted at a later stage or not. We also use $f(u \mid S)$ to denote the marginal contribution of an element u to a set S. Formally, $f(u \mid S) = f(S + u) - f(S)$. The following technical lemma is used later in the proof.

Lemma 4.1. *For every $k < i \leq n$, $f(S_{i-1} + u_i - u'_i) - f(S_{i-1}) \geq f(u_i \mid A_{i-1}) - f(S_{i-1})/k$.*

Proof. By the choice of u'_i and an averaging argument,

$$
\begin{aligned}
f(S_{i-1} + u_i - u'_i) - f(S_{i-1}) &\geq \frac{\sum_{u' \in S_{i-1}}[f(S_{i-1} + u_i - u') - f(S_{i-1})]}{k} \\
&\geq \frac{\sum_{u' \in S_{i-1}}[f(S_{i-1} + u_i) - f(S_{i-1})]}{k} + \frac{\sum_{u' \in S_{i-1}}[f(S_{i-1} - u') - f(S_{i-1})]}{k} ,
\end{aligned}
$$

where the second inequality follows from submodularity. The first term on the rightmost hand side is $f(u_i \mid S_{i-1})$, which can be lower bounded by $f(u_i \mid A_{i-1})$ because $S_{i-1} \subseteq A_{i-1}$. The second term

11

on the rightmost hand side can be lower bounded using the submodularity and non-negativity of f as follows:

$$\frac{\sum_{u' \in S_{i-1}}[f(S_{i-1} - u') - f(S_{i-1})]}{k} \geq \frac{f(\varnothing) - f(S_{i-1})}{k} \geq -\frac{f(S_{i-1})}{k} \ . \qquad \square$$

The following lemma relates S_i and A_i.

Lemma 4.2. *For every $0 \leq i \leq n$, $f(S_i) \geq \frac{c}{c+1} \cdot f(A_i)$.*

Proof. For $i = 0$, the lemma clearly holds since $f(A_0) = f(\varnothing) = f(S_0)$. Thus, it is enough to prove that for every $1 \leq i \leq n$, $f(S_i) - f(S_{i-1}) \geq \frac{c}{c+1} \cdot [f(A_i) - f(A_{i-1})]$. If the algorithm does not accept u_i then $S_i = S_{i-1}$ and $A_i = A_{i-1}$, and the claim is trivial. Hence, we only need to consider the case in which the algorithm accepts element u_i.

If $i \leq k$ then $f(S_i) - f(S_{i-1}) = f(u_i \mid S_{i-1}) \geq f(u_i \mid A_{i-1}) = f(A_i) - f(A_{i-1})$, where the inequality follows from submodularity since $S_{i-1} \subseteq A_{i-1}$. If $i > k$, then it is possible to lower bound $f(S_i) - f(S_{i-1}) = f(S_{i-1} + u_i - u_i') - f(S_{i-1})$ in two ways. First, a lower bound of $f(u_i \mid A_{i-1}) - f(S_{i-1})/k$ is given by Lemma 4.1. A second lower bound of $c \cdot f(S_{i-1})/k$ follows since the algorithm accepts u_i. Using both lower bounds, we get:

$$\begin{aligned}
f(S_i) - f(S_{i-1}) &\geq \ \max\{f(u_i \mid A_{i-1}) - f(S_{i-1})/k, c \cdot f(S_{i-1})/k\} \\
&\geq \ \frac{c \cdot [f(u_i \mid A_{i-1}) - f(S_{i-1})/k] + 1 \cdot [c \cdot f(S_{i-1})/k]}{c+1} \\
&\geq \ \frac{c}{c+1} \cdot f(u_i \mid A_{i-1}) = \frac{c}{c+1} \cdot [f(A_i) - f(A_{i-1})] \ . \qquad \square
\end{aligned}$$

We are now ready to prove the competitive ratio of Algorithm 3, and thus, also the second part of Theorem 1.5.

Corollary 4.3. *The competitive ratio of Algorithm 3 is at least $\frac{c}{(c+1)^2}$. Hence, for $c = 1$ the competitive ratio of Algorithm 3 is at least $1/4$.*

Proof. Let OPT be the optimal solution, and consider an element $u_i \in OPT \setminus A_i$. Since u_i was rejected by Algorithm 3, the following inequality must hold:

$$\frac{c \cdot f(S_{i-1})}{k} > f(S_{i-1} + u_i - u_i') - f(S_{i-1}) \geq f(u_i \mid A_{i-1}) - \frac{f(S_{i-1})}{k} \ ,$$

where the second inequality follows from Lemma 4.1. Rearranging yields:

$$f(u_i \mid A_{i-1}) < \frac{c+1}{k} \cdot f(S_{i-1}) \leq \frac{c+1}{k} \cdot f(S_n) \ ,$$

where the last inequality uses the monotonicity of $f(S_i)$ (as a function of i). In conclusion, by the submodularity and monotonicity of f and Lemma 4.2,

$$\begin{aligned}
f(OPT) &\leq f(A_n) + \sum_{u \in OPT \setminus A_n} f(u \mid A_n) < f(A_n) + \sum_{u \in OPT \setminus A_n} \left(\frac{c+1}{k} \cdot f(S_n)\right) \\
&\leq f(A_n) + (c+1) \cdot f(S_n) \leq \frac{(c+1)^2}{c} \cdot f(S_n) \ . \qquad \square
\end{aligned}$$

At this point we return the focus to the general `cardinality-model` allowing the objective f to be non-monotone. For the algorithm and analysis we present we need the notation described at the beginning of Section 3. More specifically, we use the multilinear extension F of f and the assumption that a set $S \subseteq \mathcal{N}$ represents also its characteristic vector. As a first step, consider a variant of Algorithm 3 modified in two ways:

- The value k is replaced with pk for an integer parameter $p \geq 1$.
- The algorithm uses an auxiliary function $g : 2^{\mathcal{N}} \to \mathbb{R}^+$ instead of the real objective function f. The auxiliary function g is defined as $g = F(p^{-1} \cdot S)$.

Notice that the modified algorithm does not produce a feasible solution. Still, we are interested in analyzing it for reasons that are explained later. The proofs of Lemmata 4.1 and 4.2 do not use the monotonicity of f. Hence, both lemmata still hold with pk and g taking the roles of k and f, respectively (notice that g is non-negative and submodular). The resulting lemmata after the replacements are given below.

Lemma 4.4. *For every $pk < i \leq n$, $g(S_{i-1} + u_i - u_i') - g(S_{i-1}) \geq g(u_i \mid A_{i-1}) - g(S_{i-1})/(pk)$.*

Lemma 4.5. *For every $0 \leq i \leq n$, $g(S_i) \geq \frac{c}{c+1} \cdot g(A_i)$.*

We also need the following lemma of [23].

Lemma 4.6 (Lemma 2.3 of [23]). *Let $f : 2^{\mathcal{N}} \to \mathbb{R}$ be submodular, $A, B \subseteq \mathcal{N}$ two (not necessarily disjoint) sets and $A(p), B(q)$ their independently sampled subsets, where each element of A appears in $A(p)$ with probability p and each element of B appears in $B(q)$ with probability q. Then*

$$\mathbb{E}[f(A(p) \cup B(q))] \geq (1-p)(1-q) \cdot f(\varnothing) + p(1-q) \cdot f(A) + (1-p)q \cdot f(B) + pq \cdot f(A \cup B) \ .$$

Using the above lemmata, it is possible to get a guarantee on $g(S_n)$.

Corollary 4.7. *Let OPT be the optimal solution. The modified Algorithm 3 guarantees:*

$$g(S_n) \geq \frac{cp^{-1}(1 - p^{-1})}{(c+1)(1 + p^{-1}c)} \cdot f(OPT) \ .$$

Hence, for $c = 7/4$ and $p = 3$, $g(S_n) \geq (56/627) \cdot f(OPT)$.

Proof. Consider an element $u_i \in OPT \setminus A_i$. Since u_i was rejected by Algorithm 3, the following inequality must hold:

$$\frac{c \cdot g(S_{i-1})}{pk} > g(S_{i-1} + u_i - u_i') - g(S_{i-1}) \geq g(u_i \mid A_{i-1}) - \frac{g(S_{i-1})}{pk} \ ,$$

where the second inequality follows from Lemma 4.4. Rearranging yields:

$$g(u_i \mid A_{i-1}) < \frac{c+1}{pk} \cdot g(S_{i-1}) \leq \frac{c+1}{pk} \cdot g(S_n) \ ,$$

where the last inequality uses the monotonicity of $g(S_i)$ (as a function of i) for $i \geq pk$. By the submodularity of g and Lemma 4.5, we now get:

$$g(A_n \cup OPT) \leq g(A_n) + \sum_{u \in OPT \setminus A_n} g(u_i \mid A_n) < g(A_n) + \sum_{u \in OPT \setminus A_n} \left(\frac{c+1}{pk} \cdot g(S_n) \right)$$

$$\leq g(A_n) + p^{-1}(c+1) \cdot g(S_n) \leq \frac{(c+1)(1 + p^{-1}c)}{c} \cdot g(S_n) \ .$$

On the other hand, by Lemma 4.6,

$$g(A_n \cup OPT) = F(p^{-1} \cdot OPT + p^{-1} \cdot (A_n \setminus OPT)) \geq p^{-1}(1 - p^{-1}) \cdot f(OPT) \ . \qquad \square$$

A naïve implementation of Algorithm 3 requires an exponential time to evaluate F. However, this can be fixed, at a loss of an arbitrary small constant $\varepsilon > 0$ in the guarantee of Corollary 4.7. See the above discussion about a polynomial time implementation of Algorithm 1 for the main ideas necessary for such an implementation. For more details, see Appendix C.

Consider a collection of vectors $\{y_i\}_{i=0}^n$ obtained from the sets $\{S_i\}_{i=0}^n$ via the following rule: $y_i = p^{-1} \cdot S_i$. The vectors $\{y_i\}_{i=0}^n$ form a *fractional* solution for the `cardinality-model` in the following sense. First, for every $0 \le i \le n$, the sum of the coordinates of y_i is at most k. Second, for every $1 \le i \le n$, $y_i \le y_{i-1} \vee u$, where the inequality is element-wise. To convert the vectors $\{y_i\}_{i=0}^n$ into an integral solution $\{\bar{S}_i\}_{i=0}^n$ for the `cardinality-model`, we need an online rounding method.

The rounding we suggest works as follows. For every $1 \le \ell \le k$, select a uniformly random value r_ℓ from the range $\{p(\ell-1)+1, p(\ell-1)+2, \ldots, p\ell\}$. Intuitively, the values r_ℓ specify the indexes of the elements from the set $\{u_j\}_{j=1}^{pk}$ that get into the rounded solution. Later elements can get into the rounded solution only if they "take the place" of one of these elements. More formally, we say that an element $u_j \in S_i$ uses place a if there exists a series of indexes: $j_1, j_2, \ldots, j_m$ such that:

$$j = j_1 \ , \quad u'_{j_i} = u_{j_{i-1}} \ \ (i.e.,\ u_{j_i} \text{ replaced } u_{j_{i-1}} \text{ in } S_{j_i}) \quad \text{and} \quad j_m = a \ .$$

The element $u_j \in S_i$ gets into $\bar{S}_i$ if and only if it uses a place a equal to r_ℓ for some $\ell \in \{1, 2, \ldots, k\}$.

Observation 4.8. *For every $0 \le i \le n$, at most one element of S_i uses every place $a \in \{1, 2, \ldots, pk\}$.*

Proof. The observation follows immediately from the definition when $i \le pk$ because for every $1 \le j \le i$, the element u_j uses place j. For larger values of i we prove the observation by induction. Assume the claim holds for $i - 1 \ge pk$, and let us prove it for i. Since u_i got into S_i, the place used by u_i is defined as the place used by u'_i. As $S_i = S_{i-1} + u_i - u'_i$, every place is used at most once also in S_i. $\square$

Corollary 4.9. *The sets $\{\bar{S}_i\}_{i=0}^n$ form a feasible online solution for the `cardinality-model`.*

Proof. Observation 4.8 implies immediately that $|\bar{S}_i| \le k$ since only k places make elements of S_i using them appear in $\bar{S}_i$. Observe also that the rule whether an element of S_i gets into $\{\bar{S}_i\}$ is independent of i. Hence, for every $1 \le i \le k$, $\bar{S}_i \subseteq \bar{S}_{i-1} + u_i$ simply because $S_i \subseteq S_{i-1} + u_i$. $\square$

Next, we want to show that $f(\bar{S}_n)$ is as good approximation for $f(OPT)$ as $g(S_n)$. This is done by the following lemma.

Lemma 4.10. $\mathbb{E}[f(\bar{S}_n)] \ge g(S_n)$.

Proof. For every $1 \le \ell \le k$, let R_ℓ be the set of elements of S_n using places from $\{p(\ell-1)+1, p(\ell-1)+2, p\ell\}$. By Observation 4.8, the size of R_ℓ is at most p. Let $\bar{R}_\ell$ be a set containing the single element of R_ℓ using the place r_ℓ, or the empty set if no element uses this place. By definition, $S_n = \bigcup_{\ell=1}^k \bar{R}_\ell$.

Recall also that given a vector $x \in [0,1]^{\mathcal{N}}$, $\mathtt{R}(x)$ is a random set containing every element $u \in R_i$ with probability x_u. Using this notation,

$$g(S_n) = \mathbb{E}[f(\mathtt{R}(p^{-1} \cdot S_n))] = \mathbb{E}\left[f\left(\bigcup_{\ell=1}^k \mathtt{R}(p^{-1} \cdot R_\ell) \right) \right] \ .$$

Hence, the lemma is equivalent to the inequality:

$$\mathbb{E}\left[f\left(\bigcup_{\ell=1}^{k}\bar{R}_{\ell}\right)\right] \geq \mathbb{E}\left[f\left(\bigcup_{\ell=1}^{k}\texttt{R}(p^{-1}\cdot R_{\ell})\right)\right] \ . \tag{3}$$

As a step toward proving Inequality (3), fix $1 \leq \ell \leq k$, and let D_{ℓ} be a random subset of $\mathcal{N} \setminus R_{\ell}$ with an arbitrary distribution. We use the notation $v_1, v_2, \ldots, v_{p'}$ to denote the elements of R_{ℓ} ($p' = |R_{\ell}| \leq p$), and $R_{\ell,i}$ to denote the set $\{v_1, v_2, \ldots, v_i\} \subseteq R_{\ell}$. By submodularity,

$$\mathbb{E}\left[f\left(D_{\ell} \cup \bar{R}_{\ell}\right)\right] = \mathbb{E}[f(D_{\ell})] + \frac{\sum_{i=1}^{p'}\mathbb{E}[f(v_i \mid D_{\ell})]}{p}$$

$$\geq \mathbb{E}[f(D_{\ell})] + \frac{\sum_{i=1}^{p'}\mathbb{E}[f(v_i \mid D_{\ell} \cup \texttt{R}(p^{-1}\cdot R_{\ell,i-1}))]}{p} = \mathbb{E}\left[f\left(D_{\ell} \cup \texttt{R}(p^{-1}\cdot R_{\ell})\right)\right] \ .$$

Inequality (3) follows by repeated applications of the above inequality, once for every $1 \leq \ell \leq k$. $\square$

The first part of Theorem 1.5 follows by combining the modified Algorithm 3 with the rounding method described above.

4.2 Hardness results for the `cardinality-model`

In this section we prove Theorem 1.6. The proof of the theorem is split between two lemmata. Lemma 4.11 proves the part of Theorem 1.6 referring to monotone objectives, and Lemma 4.12 proves the part referring to general objective functions.

Lemma 4.11. *No deterministic (randomized) online algorithm has a competitive ratio better than $1/2 + \varepsilon$ ($3/4 + \varepsilon$) for the* `cardinality-model`, *even if the objective function is restricted to be monotone.*

Proof. Let $k = \lceil (2\varepsilon)^{-1} \rceil$ and consider the ground set $\mathcal{N} = \{u_i\}_{i=1}^{k} \cup \{v_i\}_{i=1}^{k} \cup \{w\}$ and the monotone submodular function $f : 2^{\mathcal{N}} \to \mathbb{R}^{+}$ defined as follows:

$$f(S) = |S \cap \{u_i\}_{i=1}^{k}| + \min\{k, |S \cap \{v_i\}_{i=1}^{k}| + k \cdot |S \cap \{w\}|\} \ .$$

Intuitively, f gains a "point" for every element of the forms u_i and v_i. The element w gives k "points", but these are the same points of the elements of the form v_i, *i.e.*, v_i gives no "points" once w is in the solution.

Fix an algorithm ALG, and set the element w to be revealed after all the other elements of $\mathcal{N}$. Observe that as long as w is not revealed, there is no way for ALG to distinguish between the elements of $\{u_i\}_{i=1}^{k}$ and $\{v_i\}_{i=1}^{k}$. Thus, if ALG is deterministic, the set S_{2k} is determined solely based on the order in which the elements are revealed. By choosing the right order, one can guarantee that $S_{2k} \subseteq \{v_i\}_{i=1}^{k}$, which implies that S_{2k+1} is a subset of $\{v_i\}_{i=1}^{k} \cup \{w\}$, and thus, has a value of at most k. On the other hand, the optimal solution is $\{u_i\}_{i=1}^{k-1} \cup \{w\}$, and its value is $2k - 1$. Hence, the competitive ratio of ALG is at most:

$$\frac{k}{2k-1} \leq \frac{k+1}{2k} \leq \frac{1}{2} + \varepsilon \ .$$

If ALG is randomized, S_{2k} depends also on the random choices of ALG. However, by symmetry, it is still possible to choose a revelation order for the elements of $\{u_i\}_{i=1}^{k}$ and $\{v_i\}_{i=1}^{k}$ guaranteeing

that in expectation S_{2k} contains no more than $k/2$ elements of $\{u_i\}_{i=1}^k$. Thus, the expected value of $f(S_{2k+1})$ can be upper bounded by $3k/2$, and the competitive ratio of ALG is no more than:

$$\frac{3k/2}{2k-1} \leq \frac{3k/2+1}{2k} \leq \frac{3}{4} + \varepsilon \ . \qquad \qquad \square$$

Lemma 4.12. *No online algorithm has a competitive ratio better than $1/2 + \varepsilon$ for the* `cardinality-model`*.*

Proof. Intuitively, the proof of Lemma 4.12 provides a weaker bound for randomized algorithms because the number of elements of the form v_i in the ground set is quite small. In this proof we fix this problem. Let $k = \lceil \varepsilon^{-1} \rceil$ and $h = 2k \cdot \lceil \varepsilon^{-1} \rceil$. Consider the ground set $\mathcal{N} = \{u_i\}_{i=1}^k \cup \{v_i\}_{i=1}^h \cup \{w\}$ and the non-negative submodular function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ defined as follows:

$$f(S) = \begin{cases} |S| & \text{if } w \notin S \ , \\ k + |S \cap \{u_i\}_{i=1}^k| & \text{otherwise} \ . \end{cases}$$

Fix an algorithm ALG, and set the element w to be revealed after all the other elements of $\mathcal{N}$. Observe that as long as w is not revealed, there is no way for ALG to distinguish between the elements of $\{u_i\}_{i=1}^k$ and $\{v_i\}_{i=1}^h$. Hence, by symmetry, it is possible to choose a revelation order for these elements guaranteeing that in expectation S_{k+h} contains no more than $k^2/(k+h)$ elements of $\{u_i\}_{i=1}^k$. The final solution of ALG is a subset of $S_{k+h} + w$. Since the size of S_{k+h} is at most k, no element of $S_{k+h} + w$ has a negative marginal contribution, and thus, the value of ALG's solution can be upper bounded by $f(S_{k+h} + w)$. Hence, the total expected value ALG achieves is at most:

$$\mathbb{E}[f(S_{k+h} + w)] = k + \mathbb{E}[|S_{k+h} \cap \{u_i\}_{i=1}^k|] \leq k + \frac{k^2}{k+h} \ .$$

On the other hand, the optimal solution is $\{u_i\}_{i=1}^{k-1} \cup \{w\}$, and its value is $2k - 1$. Hence, the competitive ratio of ALG is at most:

$$\frac{k + \frac{k^2}{k+h}}{2k-1} \leq \frac{k+1}{2k} + \frac{k}{k+h} \leq \frac{1}{2} + \varepsilon \ . \qquad \qquad \square$$

References

[1] Ran Adler and Yossi Azar. Beating the logarithmic lower bound: Randomized preemptive disjoint paths and call control algorithms. In *SODA*, pages 1–10, 1999.

[2] A. A. Ageev and M. I. Sviridenko. An 0.828 approximation algorithm for the uncapacitated facility location problem. *Discrete Appl. Math.*, 93:149–156, July 1999.

[3] Yossi Azar, Iftah Gamzu, and Ran Roth. Submodular max-sat. In *ESA*, pages 323–334, 2011.

[4] Moshe Babaioff, Jason D. Hartline, and Robert D. Kleinberg. Online algorithms with buyback. In *Workshop on Ad Auctions*, 2008.

[5] Moshe Babaioff, Jason D. Hartline, and Robert D. Kleinberg. Selling ad campaigns: online algorithms with cancellations. In *EC*, pages 61–70, 2009.

[6] Amotz Bar-Noy and Michael Lampis. Online maximum directed cut. *Journal of Combinatorial Optimization*, 24(1):52–64, 2012.

[7] Mohammad Hossein Bateni, Mohammad Taghi Hajiaghayi, and Morteza Zadimoghaddam. Submodular secretary problem and extensions. In Maria Serna, Ronen Shaltiel, Klaus Jansen, and José Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, Lecture Notes in Computer Science, pages 39–52. Springer Berlin / Heidelberg, 2010.

[8] Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization, 2012. FOCS 2012.

[9] Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[10] Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: Matchings, matroids, and more. *CoRR*, abs/1309.2038, 2013.

[11] Chandra Chekuri and Sanjeev Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM J. Comput.*, 35(3):713–728, September 2005.

[12] Reuven Cohen, Liran Katzir, and Danny Raz. An efficient approximation for the generalized assignment problem. *Information Processing Letters*, 100(4):162–166, 2006.

[13] Florin Constantin, Jon Feldman, S. Muthukrishnan, and Martin Pál. An online mechanism for ad slot reservations with cancellations. In *SODA*, pages 1265–1274, 2009.

[14] G. Cornuejols, M. L. Fisher, and G. L. Nemhauser. Location of bank accounts to optimize float: an analytic study of exact and approximate algorithms. *Management Sciences*, 23:789–810, 1977.

[15] G. Cornuejols, M. L. Fisher, and G. L. Nemhauser. On the uncapacitated location problem. *Annals of Discrete Mathematics*, 1:163–177, 1977.

[16] Shaddin Dughmi, Tim Roughgarden, and Mukund Sundararajan. Revenue submodularity. In *EC*, pages 243–252, 2009.

[17] Matthias Englert and Matthias Westermann. Lower and upper bounds on fifo buffer management in QoS switches. *Algorithmica*, 53(4):523–548, 2009.

[18] Leah Epstein, Asaf Levin, Danny Segev, and Oren Weimann. Improved bounds for online preemptive matching. In *STACS*, pages 389–399, 2013.

[19] Uriel Feige. A threshold of $\ln n$ for approximating set cover. *J. ACM*, 45(4):634–652, 1998.

[20] Uriel Feige and Michel X. Goemans. Aproximating the value of two prover proof systems, with applications to max 2sat and max dicut. In *ISTCS*, pages 182–189, 1995.

[21] Uriel Feige and Shlomo Jozeph. Oblivious algorithms for the maximum directed cut problem. *CoRR*, abs/1010.0406, 2010.

[22] Uriel Feige and Shlomo Jozeph. Oblivious algorithms for the maximum directed cut problem. *Algorithmica*, pages 1–20, 2013.

[23] Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

[24] Uriel Feige and Jan Vondrák. Approximation algorithms for allocation problems: Improving the factor of $1 - 1/e$. In *FOCS*, pages 667–676, 2006.

[25] Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. Submodular maximization with cardinality constraints.

[26] Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. Improved competitive ratios for submodular secretary problems. In *APPROX*, pages 218–229, 2011.

[27] Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. Nonmonotone submodular maximization via a structural continuous greedy algorithm. In *ICALP*, pages 342–353, 2011.

[28] Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *FOCS*, 2011.

[29] Lisa Fleischer, Michel X. Goemans, Vahab S. Mirrokni, and Maxim Sviridenko. Tight approximation algorithms for maximum general assignment problems. In *SODA*, pages 611–620, 2006.

[30] Stanley P.Y. Fung. Online scheduling with preemption or non-completion penalties. *Journal of Scheduling*, 17(2):173–183, 2014.

[31] Shayan Oveis Gharan and Jan Vondrák. Submodular maximization by simulated annealing. In *SODA*, pages 1098–1117, 2011.

[32] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[33] Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. Constrained non-monotone submodular maximization: offline and secretary algorithms. In *WINE*, pages 246–257. Springer-Verlag, 2010.

[34] Eran Halperin and Uri Zwick. Combinatorial approximation algorithms for the maximum directed cut problem. In *SODA*, pages 1–7, 2001.

[35] Xin Han, Yasushi Kawase, and Kazuhisa Makino. Online unweighted knapsack problem with removal cost. *Algorithmica*, pages 1–16, 2013.

[36] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48:798–859, July 2001.

[37] Norman Huang and Allan Borodin. Bounds on double-sided myopic algorithms for unconstrained non-monotone submodular maximization. *CoRR*, abs/1312.2173, 2013.

[38] Kazuo Iwama and Shiro Taketomi. Removable online knapsack problems. In *ICALP*, pages 293–305, 2002.

[39] Richard M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[40] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? *SIAM J. Comput.*, 37:319–357, April 2007.

[41] Andreas Krause, AjitSingh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *J. Mach. Learn. Res.*, 9:235–284, January 2008.

[42] Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. In *UAI*, page 5, 2005.

[43] Andreas Krause and Carlos Guestrin. Near-optimal observation selection using submodular functions. In *AAAI*, pages 1650–1654. AAAI Press, 2007.

[44] Andreas Krause, Jure Leskovec, Carlos Guestrin, Jeanne VanBriesen, and Christos Faloutsos. Efficient sensor placement optimization for securing large water distribution networks. *Journal of Water Resources Planning and Management*, 134(6):516–526, November 2008.

[45] Michael Lewin, Dror Livnat, and Uri Zwick. Improved rounding techniques for the max 2-sat and max di-cut problems. In *IPCO*, pages 67–82, 2002.

[46] Hui Lin and Jeff Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *North American chapter of the Association for Computational Linguistics/Human Language Technology Conference (NAACL/HLT-2010)*, Los Angeles, CA, June 2010.

[47] Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *HLT*, pages 510–520, 2011.

[48] Shiro Matuura and Tomomi Matsui. 0.863-approximation algorithm for max dicut. In *APPROX*, pages 138–146, 2001.

[49] Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. In *FOCS*, pages 21–30, 2005.

[50] G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978.

[51] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions – I. *Mathematical Programming*, 14:265–294, 1978.

[52] Andreas S. Schulz and Nelson A. Uhan. Approximating the least core value and least core of cooperative games with supermodular costs. *Discrete Optimization*, 10(2):163–180, 2013.

[53] Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson. Gadgets, approximation, and linear programming. *SIAM J. Comput.*, 29:2074–2097, April 2000.

[54] Ashwinkumar Badanidiyuru Varadaraja. Buyback problem - approximate matroid intersection with cancellation costs. In Luca Aceto, Monika Henzinger, and Jiří Sgall, editors, *Automata, Languages and Programming*, volume 6755 of *Lecture Notes in Computer Science*, pages 379–390. Springer Berlin Heidelberg, 2011.

[55] Ashwinkumar Badanidiyuru Varadaraja and Robert Kleinberg. Randomized online algorithms for the buyback problem. In *WINE*, pages 529–536, 2009.

[56] Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM J. Comput.*, 42(1):265–304, 2013.

A Proof of Theorem 1.1

In this section we prove Theorem 1.1. Namely, we need to show that no algorithm using no preemption is $(1/4 + \varepsilon)$-competitive for any constant $\varepsilon > 0$ even when f is guaranteed to be a cut function of a weighted directed graph and there is no constraint on the set of elements that can be picked. Assume for the sake of contradiction that there exists such an algorithm ALG for some, fixed, $\varepsilon > 0$. We design an adversary that constructs an instance for ALG that leads to a contradiction. The adversary uses a ground set $\mathcal{N}$ with $n = n(\varepsilon)$ elements (where $n(\varepsilon)$ is function that depends only on ε and is described later).

Each time an element u_i is revealed the adversary uses the properties of ALG to choose the weighted set E_i of arcs that leaves u_i. The objective function f is defined as the weighted cut function of the graph $(\mathcal{N}, \bigcup_{i=1}^{n} E_i)$. Notice that any query of f made by ALG before u_i is revealed does not depend on E_i, and thus, the behavior of ALG up to the point when u_i is revealed can be described before $E_i, E_{i+1}, \dots, E_n$ are determined. An exact description of the adversary is given as Algorithm 4. For ease of notation, we denote by $\mathcal{N}_i$ the set $\{u_j \mid 1 \le j \le i\}$ for every $0 \le i \le n$.

Algorithm 4: Adversary for ALG

---

1 Let $m \leftarrow 1 + 4/\varepsilon$, $n_0 \leftarrow \lceil 8/\varepsilon \rceil$ and $n = \lceil e^{8/\varepsilon} \cdot n_0 \rceil$.

2 Let Mode $\leftarrow A$.

3 **for** $i = 1$ *to* n **do**

4   **if** Mode $= A$ **then**

5   Reveal element u_i with arcs of weight m^i going to all nodes revealed so far.

6   **else //** If Mode $= B$

7   Reveal element u_i with no exiting arcs.

8   **for** *every set* $S \subseteq V_{i-1}$ **do**

9   Let $A_{i,S}$ be the event that the algorithm accepted exactly the elements of S before u_i is revealed.

10   Let $p_{i,S}$ be the probability of $A_{i,S}$.

11   Let $q_{i,S}$ be the probability u_i is accepted given $A_{i,S}$.

12   **if** $i \ge n_0$ *and* $\sum_{S \subseteq N_{i-1}} \left[p_{i,S} \cdot q_{i,S} \cdot \left(1 - \frac{|S|}{i-1} \right) \right] < 1/4 + \varepsilon/4$ **then**

13   Mode $\leftarrow B$.

Observe that Algorithm 4 operates in two modes. In mode A every revealed element has an arc going to every previously revealed element. In mode B the revealed elements have no exiting arcs (*i.e.*, their output degrees are 0). The adversary starts in mode A and then switches permanently to mode B when some condition holds for the first time.

Lemma A.1. *The adversary given by Algorithm 4 never switches to mode B.*

Proof. Assume for the sake of contradiction that the adversary switches to mode B immediately after element u_i is revealed. Let us upper bound the weight of the cut C produced by ALG in this case. If ALG accepted a set $S \subseteq S_{i-1}$ of elements before u_i is revealed, then the expected number of arcs leaving u_i crossing C is $(i-1) \cdot q_{i,S} \cdot \left(1 - \frac{|S|}{i-1} \right)$. By the linearity of the expectation, the

expected number of arcs leaving u_i crossing C is:

$$(i-1) \cdot \sum_{S \subseteq \mathcal{N}_{i-1}} \left[p_{i,S} \cdot q_{i,S} \cdot \left(1 - \frac{|S|}{i-1} \right) \right] < (i-1) \cdot (1/4 + \varepsilon/4) .$$

The total weight of all arcs leaving elements other than u_i is:

$$\sum_{j=1}^{i-1} [(j-1) \cdot m^j] \leq (i-1) \cdot \sum_{j=1}^{i-1} m^j \leq (i-1) \cdot \sum_{j=0}^{\infty} m^{i-1-j} = (i-1) \cdot \frac{m^i}{m-1} = (i-1)m^i \cdot \frac{\varepsilon}{4} .$$

Hence, even if all the above arcs cross C, the total weight of C is still less than $(i-1)m^i \cdot (1/4 + \varepsilon/2)$. On the other hand, the optimal solution may accept u_i and no other elements, which results in a cut of weight $(i-1)m^i$. Hence, the competitive ratio of ALG cannot be $1/4 + \varepsilon$. $\qquad\square$

Let us denote $t_{i,S} = 1 - \frac{|S|}{i-1}$. Using this notation the previous lemma implies that for every $n_0 \leq i \leq n$:

$$\sum_{S \subseteq \mathcal{N}_{i-1}} (p_{i,S} \cdot q_{i,S} \cdot t_{i,S}) \geq 1/4 + \varepsilon/2 .$$

Consider the potential function $\Phi(i) = \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot t_{i,S}^2 \right)$.

Observation A.2. *For every $1 \leq i \leq n-1$,*

$$\Phi(i+1) - \Phi(i) = \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot \left[\frac{[(i-1)t_{i,S} + (1 - q_{i,S})]^2}{i^2} - t_{i,S}^2 \right] \right)$$

$$= \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot \frac{(1 - 2i)t_{i,S}^2 + 2(i-1)t_{i,S}(1 - q_{i,S}) + (1 - q_{i,S})^2}{i^2} \right) .$$

Proof. Let B_i be the expected set of elements accepted by ALG immediately after u_i is revealed. Notice that:

$$\Phi(i+1) = \mathbb{E} \left[\left(1 - \frac{|B_i|}{i} \right)^2 \right] = \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot \mathbb{E} \left[\left(1 - \frac{|B_i|}{i} \right)^2 \;\middle|\; A_{i,S} \right] \right) .$$

Observe also that $\mathbb{E}[|B_i| \mid A_{i,S}] = \mathbb{E}[|S| + q_{i,S} \mid A_{i,S}]$. Hence, by the linearity of the expectation.

$$\Phi(i+1) = \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot \mathbb{E} \left[\left(1 - \frac{|S| + q_{i,S}}{i} \right)^2 \;\middle|\; A_{i,S} \right] \right)$$

$$= \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot \left(1 - \frac{(1 - t_{i,S})(i-1) + q_{i,S}}{i} \right)^2 \right)$$

$$= \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot \left(\frac{t_{i,S}(i-1) + (1 - q_{i,S})}{i} \right)^2 \right) .$$

The observation follows by combining the above equality with the definition of $\Phi(i)$. $\qquad\square$

Using the above observation, we can now prove the following lemma.

Lemma A.3. *For every $n_0 \leq i \leq n-1$, $\Phi(i+1) - \Phi(i) \leq -\varepsilon/(4i)$.*

Proof. By Observation A.2,

$$\Phi(i+1) - \Phi(i) = \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot \frac{(1-2i)t_{i,S}^2 + 2(i-1)t_{i,S}(1-q_{i,S}) + (1-q_{i,S})^2}{i^2} \right)$$

$$\leq 2 \cdot \sum_{S \subseteq \mathcal{N}_{i-1}} \left(p_{i,S} \cdot \frac{-i \cdot t_{i,S}^2 + i \cdot t_{i,S} \cdot (1-q_{i,S}) + 1}{i^2} \right)$$

$$= \frac{2}{i} \cdot \sum_{S \subseteq \mathcal{N}_{i-1}} (p_{i,S} \cdot t_{i,S} \cdot (1-q_{i,S} - t_{i,S})) + \frac{2}{i^2} .$$

Notice that $t_{i,S} \cdot (1 - q_{i,S} - t_{i,S}) = t_{i,S} \cdot (1 - t_{i,S}) - t_{i,S} \cdot q_{i,S} \leq 1/4 - t_{i,S} \cdot q_{i,S}$. Plugging this inequality into the previous one yields:

$$\Phi(i+1) - \Phi(i) \leq \frac{2}{i} \cdot \sum_{S \subseteq \mathcal{N}_{i-1}} (p_{i,S} \cdot (1/4 - t_{i,S} \cdot q_{i,S})) + \frac{2}{i^2} = \frac{2}{i} \cdot \left[1/4 - \sum_{S \subseteq \mathcal{N}_{i-1}} p_{i,S} \cdot t_{i,S} \cdot q_{i,S} \right] + \frac{2}{i^2}$$

$$\leq \frac{2}{i} \cdot [1/4 - (1/4 + \varepsilon/4)] + \frac{2}{i^2} = -\frac{\varepsilon}{2i} + \frac{2}{i^2} \leq -\frac{\varepsilon}{2i} + \frac{\varepsilon}{4i} = -\frac{\varepsilon}{4i} ,$$

where the third inequality holds since $i \geq n_0 \geq 8/\varepsilon$. $\qquad\square$

Corollary A.4. $\Phi(n) \leq \Phi(n_0) - 2$.

Proof. By Lemma A.3,

$$\Phi(n) - \Phi(n_0) = \sum_{i=n_0}^{n-1} [\Phi(i+1) - \Phi(i)] \leq - \sum_{i=n_0}^{n-1} \frac{\varepsilon}{4i}$$

$$\leq -\frac{\varepsilon}{4} \cdot \int_{n_0}^{n} \frac{dx}{x} = -\frac{\varepsilon}{4} \cdot \ln(n/n_0) \leq -\frac{\varepsilon}{4} \cdot \ln(n_0 \cdot e^{8/\varepsilon}/n_0) = -2 . \qquad\square$$

Corollary A.4 leads to an immediate contradiction since Φ can only take values from the range $[0,1]$.

B   A Polynomial Time Implementation of Algorithm 1

In this section we prove the second part of Theorem 1.2 by giving a polynomial time implementation of Algorithm 1 having the same competitive ratio up to an arbitrary constant $\varepsilon > 0$. Our implementation is given as Algorithm 5. This algorithms uses the notation $T_\theta(x)$, where x is a vector and $\theta \in [0,1]$. This notation stands for $\{u \mid x_u \geq \theta\}$, *i.e.*, the set of all elements whose coordinate in x is at least θ.

Our first objective is to show that Algorithm 5 is an online algorithm according to the `unconstrained-model`.

Lemma B.1. *For every $1 \leq i \leq n$, $S_i \subseteq S_{i-1} + u_i$.*

Algorithm 5: Marginal Choice - Polynomial Time Implementation

---

1 **foreach** *element u_i revealed* **do**

2 $\quad$ Choose a uniformly random threshold $\theta_i \in [0, 1]$.

3 $\quad$ Let $\mathcal{N}_i \leftarrow \{u_1, u_2, \ldots, u_i\}$.

4 $\quad$ Let $\mathcal{V}_i$ be a collections of $\lceil 3072 \cdot \varepsilon^{-2} i^4 \ln[2\varepsilon^{-1}(4i)^2] \rceil$ random vectors from $[0,1]^{\mathcal{N}_i}$.

5 $\quad$ **for** $j = 1$ **to** $i - 1$ **do**

6 $\quad\quad$ **for** *every vector $v \in \mathcal{V}_j$* **do**

7 $\quad\quad\quad$ Extend v to be a vector of $[0,1]^{\mathcal{N}_i}$ by setting v_{u_i} to be an independent uniformly random value from $[0, 1]$.

8 $\quad$ **for** $j = 1$ **to** i **do**

9 $\quad\quad$ $m_j \leftarrow |\mathcal{V}_j|^{-1} \cdot \sum_{v \in \mathcal{V}_j} [f(T_{\theta_j}(v) + u_j) - f(T_{\theta_j}(v) - u_j)]$. /* $m_j \approx \partial_{u_j} F(\theta_j \cdot \mathcal{N}_i)$. */

10 $\quad$ Let $S_i \leftarrow \{u_j \in \mathcal{N}_i \mid m_j \geq 0\}$.

Proof. For $i = 1$ the claim is trivial since S_1 can contain only u_1. Thus, we assume from now on $i > 1$. For every $1 \leq j \leq i - 1$, let m_j and $\mathcal{V}_j$ denote the value m_j and the set $\mathcal{V}_j$ after iteration $i - 1$, and let m'_j and $\mathcal{V}'_j$ denote these entities after iteration i.

By definition, S_i contains only elements of $\mathcal{N}_i$. Fix an element $u_j \in \mathcal{N}_i - u_i$. Then:

$$u_j \in S_i \Rightarrow \frac{\sum_{v \in \mathcal{V}'_j} [f(T_{\theta_j}(v) + u_j) - f(T_{\theta_j}(v) - u_j)]}{|\mathcal{V}_j|} = m'_j \geq 0$$

$$\Rightarrow \frac{\sum_{v \in \mathcal{V}_j} [f(T_{\theta_j}(v) + u_j) - f(T_{\theta_j}(v) - u_j)]}{|\mathcal{V}_j|} = m_j \geq 0 \Rightarrow u_j \in S_{i-1} \;,$$

where the second derivation follows from submodularity since $\mathcal{V}'_j$ is produced from $\mathcal{V}_j$ by extending vectors. $\square$

To bound the loss in the competitive ratio of Algorithm 5, we need the following claims.

Observation B.2. *For every $1 \leq i \leq n$, each set in $\mathcal{V}_i$ is an independent sample of $[0,1]^{\mathcal{N}}$.*

Proof. Follows immediately by the way Algorithm 5 constructs $\mathcal{V}_i$ and the way this collection is updated when new elements arrive. $\square$

Lemma B.3. *Let $X_1, X_2, \ldots, X_\ell$ be independent random variables such that for each i, $X_i \in [-1, 1]$. Let $X = \frac{1}{\ell} \sum_{i=1}^{\ell} X_i$ and $\mu = \mathbb{E}[X]$. Then*

$$\Pr[X > \mu + \alpha] \leq e^{-\frac{\alpha^2 \ell}{12}} \qquad and \qquad \Pr[X < \mu - \alpha] \leq e^{-\frac{\alpha^2 \ell}{8}} \;,$$

for every $\alpha > 0$.

Proof. For every $1 \leq i \leq \ell$, let $Y_i = (1 + X_i)/2$. Additionally, let $Y = \sum_{i=1}^{\ell} Y_i$. Clearly, $Y_i \in [0, 1]$. Hence, by the Chernoff bound:

$$\Pr[X > \mu + \alpha] = \Pr[2Y/\ell - 1 > 2\mathbb{E}[Y]/\ell - 1 + \alpha] = \Pr\left[Y > \mathbb{E}[Y] + \frac{\alpha\ell}{2}\right]$$

$$\leq e^{-\frac{\mathbb{E}[Y](\alpha\ell/(2 \cdot \mathbb{E}[Y]))^2}{3}} = e^{-\frac{\alpha^2 \ell^2}{12 \cdot \mathbb{E}[Y]}} \leq e^{-\frac{\alpha^2 \ell}{12}} \;,$$

and

$$\Pr[X < \mu - \alpha] = \Pr[2Y/\ell - 1 < 2\mathbb{E}[Y]/\ell - 1 - \alpha] = \Pr\left[Y < \mathbb{E}[Y] - \frac{\alpha\ell}{2}\right]$$

$$\leq e^{-\frac{\mathbb{E}[Y](\alpha\ell/(2\cdot\mathbb{E}[Y]))^2}{2}} = e^{-\frac{\alpha^2\ell^2}{8\cdot\mathbb{E}[Y]}} \leq e^{-\frac{\alpha^2\ell}{8}} \ . \qquad \square$$

Let $m_i(z)$ be the value that m_i gets after the last iteration of Algorithm 5 if the random variable θ_i is z.

Lemma B.4. *For every* $1 \leq i \leq n$ *and* $z \in [0,1]$, $\Pr[|m_i(z) - \partial_{u_i}F(z \cdot \mathcal{N})| > \varepsilon(4i)^{-2} \cdot f(OPT)] \leq \varepsilon(4i)^{-2}$.

Proof. Observe that m_i is obtained by averaging $|\mathcal{V}_i|$ independent samples of $f(R(z \cdot \mathcal{N}) + u_i) - f(R(z\cdot\mathcal{N}) - u_i)$, and the expectation of the last expression is $\partial_{u_i}F(z\cdot\mathcal{N})$. Additionally, observe that every sample belongs to the range $[-f(OPT), f(OPT)]$ since $f(OPT) = \max_{S \subseteq \mathcal{N}} f(S)$. Hence, by Lemma B.3,

$$\Pr[|m_i(z) - \partial_{u_i}F(z \cdot \mathcal{N})| > \varepsilon(4i)^{-2} \cdot f(OPT)] \leq 2e^{\frac{-|\mathcal{V}_i|\cdot[\varepsilon(4i)^{-2}]^2}{12}} \leq 2e^{-\ln[2\varepsilon^{-1}(4i)^2]} = \varepsilon(4i)^{-2} \ . \quad \square$$

Fix the vectors $\mathcal{V} = (\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_n)$. For every element $u_i \in \mathcal{N}$, $m_i(z)$ is continuous non-increasing function of z by submodularity. Hence, by the intermediate value theorem, one of the following must hold: $m_i(z)$ is always positive in the range $[0,1]$, $m_i(z)$ is always negative in the range $[0,1]$ or $m_i(z)$ has at least one root $z_0 \in [0,1]$. In the last case, the set $I_0 \subseteq [0,1]$ of the roots of $m_i(z)$ is non-empty. Moreover, by the monotonicity and continuity of $m_i(z)$, I_0 is a closed range. Using these observations, we define a vector $y(\mathcal{V}) \in [0,1]^{\mathcal{N}}$ as follows:

$$y(\mathcal{V})_{u_i} = \begin{cases} 0 & \text{if } m_i(0) < 0 \ , \\ 1 & \text{if } m_i(1) > 0 \ , \\ \max I_0 & \text{otherwise} \ . \end{cases}$$

Let $\mathcal{E}_i$ be the event that $\partial_{u_i}F(z\cdot\mathcal{N}) \geq -\varepsilon(4i)^{-2}\cdot f(OPT)$ for every $z < y(\mathcal{V})_{u_i}$ and $\partial_{u_i}F(z\cdot\mathcal{N}) \leq \varepsilon(4i)^{-2} \cdot f(OPT)$ for every $z > y(\mathcal{V})_{u_i}$.

Lemma B.5. *For every* $1 \leq i \leq n$, *the event* $\mathcal{E}_i$ *holds with probability at least* $1 - 2\varepsilon(4i)^{-2}$.

Proof. Let us first bound the probability that $\partial_{u_i}F(z\cdot\mathcal{N}) \geq -\varepsilon(4i)^{-2}\cdot f(OPT)$ for every $z < y(\mathcal{V})_{u_i}$. If $\partial_{u_i}F(z \cdot \mathcal{N})$ is always at least $-\varepsilon(4i)^{-2} \cdot f(OPT)$, then there is nothing to prove. Otherwise, if $\partial_{u_i}F(z \cdot \mathcal{N})$ is always at most $-\varepsilon(4i)^{-2} \cdot f(OPT)$, then $\partial_{u_i}F(\varnothing) \leq 0$, which guarantees that $y(\mathcal{V})_{u_i} = 0$, and thus, makes the claim that we want to prove void. Hence, it is enough to consider the case in which $\partial_{u_i}F(z \cdot \mathcal{N})$ is at least $-\varepsilon(4i)^{-2} \cdot f(OPT)$ for some value of z and at most $-\varepsilon(4i)^{-2} \cdot f(OPT)$ for another value of z. In this case the intermediate value theorem and the continuity of $\partial_{u_i}F(z\cdot\mathcal{N})$ imply the existence of a value y_i' such that $\partial_{u_i}F(y_i'\cdot\mathcal{N}) = -\varepsilon(4i)^{-2}\cdot f(OPT)$. By Lemma B.4, $m_i(y_i') \leq 0$ with probability at least $1 - \varepsilon(4i)^{-2}$, which implies $y(\mathcal{V})_{u_i} \leq y_i'$, and thus:

$$\partial_{u_i}F(y(\mathcal{V})_{u_i} \cdot \mathcal{N}) \geq \partial_{u_i}F(y_i' \cdot \mathcal{N}) = -\varepsilon(4i)^{-2} \cdot f(OPT) \ .$$

Observe that the last inequality implies that $\partial_{u_i}F(z\cdot\mathcal{N}) \geq -\varepsilon(4i)^{-2}\cdot f(OPT)$ for every $z < y(\mathcal{V})_{u_i}$.

A similar argument shows that $\partial_{u_i}F(z \cdot \mathcal{N}) \leq \varepsilon(4i)^{-2} \cdot f(OPT)$ for every $z > y(\mathcal{V})_{u_i}$ with probability at least $1 - \varepsilon(4i)^{-2}$. The lemma now follow by the union bound. $\quad \square$

Let $\mathcal{E}$ be the event that $\mathcal{E}_i$ holds for every $1 \leq i \leq n$ at the same time.

24

Corollary B.6. *$\mathcal{E}$ happens with probability at least $1 - \varepsilon/2$.*

Proof. Combining Lemma B.5 and the union bound gives:

$$\Pr[\mathcal{E}] \geq 1 - \sum_{i=1}^{n} 2\varepsilon(4i)^{-2} \geq 1 - \frac{\varepsilon}{8} - \int_{1}^{n} 2\varepsilon(4x)^{-2}dx = 1 - \frac{\varepsilon}{8} + \left.\frac{\varepsilon}{8x}\right|_{1}^{n} \geq 1 - \frac{\varepsilon}{2} \ . \qquad \square$$

It is important to notice that the event $\mathcal{E}$ depends only on the vectors $\mathcal{V}$. Thus, fixing the vectors $\mathcal{V}$ does not affect the distribution of $\theta_1, \theta_2, \ldots, \theta_n$. Next, we analyze the competitive ratio of Algorithm 5 assuming $\mathcal{V}$ is fixed in a way that makes the event $\mathcal{E}$ hold. The next observation corresponds to Observation 3.2.

Observation B.7. *For every given vectors $\mathcal{V}$, every element $u_i \in \mathcal{N}$ belongs to S_n with probability $y(\mathcal{V})_{u_i}$, independently. Hence, $\mathbb{E}[f(S_n)] = F(y(\mathcal{V}))$.*

Proof. An element $u_i \in \mathcal{N}$ belongs to S_n if and only if $m_i(\theta_i) \geq 0$, which is equivalent to the condition $\theta_i \leq y(\mathcal{V})_{u_i}$ since m_i is a non-increasing function. Clearly, the last condition happens with probability $y(\mathcal{V})_{u_i}$, and is independent for different elements. $\qquad \square$

The last observation implies that analyzing Algorithm 5 is equivalent to lower bounding $F(y(\mathcal{V}))$. The next lemma and corollary correspond to Lemma 3.3 and Corollary 3.5, respectively.

Lemma B.8. *For every given vectors $\mathcal{V}$ for which the event $\mathcal{E}$ holds and for every $\lambda \in [0,1]$, $F(y(\mathcal{V}) \wedge (\lambda \cdot \mathcal{N})) \geq F(\lambda \cdot \mathcal{N}) - \varepsilon \cdot \sum_{i=1}^{n}(4i)^{-2} \cdot f(OPT)$.*

Proof. For ease of notation, we use y as a shorthand for $y(\mathcal{V})$ in this proof. Observe that:

$$F(y \wedge (\lambda \cdot \mathcal{N})) = f(\varnothing) + \int_{0}^{\lambda} \frac{dF(y \wedge (z \cdot \mathcal{N}))}{dz}dz = f(\varnothing) + \int_{0}^{\lambda} \sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(y \wedge (z \cdot \mathcal{N}))dz \ ,$$

where the second equality is due to the chain rule. By submodularity and the observation that $\mathcal{E}$ implies $\partial_{u_i} F(z \cdot \mathcal{N}) \leq \varepsilon(4i)^{-2} \cdot f(OPT)$ for every $z > y_{u_i}$, we get:

$$\sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(y \wedge (z \cdot \mathcal{N})) \geq \sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(z \cdot \mathcal{N}) \geq \sum_{u \in \mathcal{N}} \partial_u F(z \cdot \mathcal{N}) - \varepsilon \cdot \sum_{i=1}^{n}(4i)^{-2} \cdot f(OPT) \ .$$

Combining the above equality and inequality, and using the chain rule again, gives:

$$\begin{aligned}
F(y \wedge (\lambda \cdot \mathcal{N})) &\geq f(\varnothing) + \int_{0}^{\lambda} \left[\sum_{u \in \mathcal{N}} \partial_u F(z \cdot \mathcal{N})\right] dz - \varepsilon \cdot \sum_{i=1}^{n}(4i)^{-2} \cdot f(OPT) \\
&= f(\varnothing) + \int_{0}^{\lambda} \frac{dF(z \cdot \mathcal{N})}{dz}dz - \varepsilon \cdot \sum_{i=1}^{n}(4i)^{-2} \cdot f(OPT) \\
&= F(\lambda \cdot \mathcal{N}) - \varepsilon \cdot \sum_{i=1}^{n}(4i)^{-2} \cdot f(OPT) \ . \qquad \square
\end{aligned}$$

Corollary B.9. *For every given vectors $\mathcal{V}$ for which the event $\mathcal{E}$ holds, $F(y(\mathcal{V})) \geq (e^{-1} - \varepsilon/2) \cdot F(OPT)$.*

Proof. Again, we use y as a shorthand for $y(\mathcal{V})$ in this proof. Fix two values $0 \leq a \leq b \leq 1$. By the chain rule:

$$F(y \wedge (b \cdot \mathcal{N})) - F(y \wedge (a \cdot \mathcal{N})) = \int_a^b \frac{dF(y \wedge (z \cdot \mathcal{N}))}{dz} dz \tag{4}$$

$$= \int_a^b \sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(y \wedge (z \cdot \mathcal{N})) dz \geq \int_a^b \sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(z \cdot \mathcal{N}) dz \ ,$$

where the inequality follows from submodularity. We use two ways to lower bound the rightmost hand side of the above inequality. First, observe that $\partial_{u_i} F(z \cdot \mathcal{N}) \geq -\varepsilon(4i)^{-2} \cdot f(OPT)$ whenever $z < y_{u_i}$. Thus, $F(y \wedge (b \cdot \mathcal{N})) - F(y \wedge (a \cdot \mathcal{N})) \geq -\varepsilon(b - a) \cdot \sum_{i=1}^n (4i)^{-2} \cdot f(OPT)$. Additionally, $\partial_u F(z \cdot \mathcal{N}) \leq \varepsilon(4i)^{-2} \cdot f(OPT)$ whenever $z > y_u$ and $\partial_u F(z \cdot \mathcal{N}) \geq -\varepsilon(4i)^{-2} \cdot f(OPT)$ whenever $z < y_u$. This allows us to derive the second lower bound, which holds whenever z is not equal to any coordinate of y (*i.e.*, for every value of z except for a finite set of values):

$$\sum_{\substack{u \in \mathcal{N} \\ z \leq y_u}} \partial_u F(z \cdot \mathcal{N}) \geq \sum_{u \in OPT} \partial_u F(z \cdot \mathcal{N}) - \varepsilon \cdot \sum_{i=1}^n (4i)^{-2} \cdot f(OPT)$$

$$= \frac{\sum_{u \in OPT} [F(u \vee (z \cdot \mathcal{N})) - F(z \cdot \mathcal{N})]}{1 - z} - \varepsilon \cdot \sum_{i=1}^n (4i)^{-2} \cdot f(OPT)$$

$$\geq \frac{F(OPT \vee (z \cdot \mathcal{N})) - F(z \cdot \mathcal{N})}{1 - z} - \varepsilon \cdot \sum_{i=1}^n (4i)^{-2} \cdot f(OPT)$$

$$\geq f(OPT) - \frac{F(y \wedge (z \cdot \mathcal{N}))}{1 - z} - \varepsilon \cdot \left(1 + \frac{1}{1 - b}\right) \cdot \sum_{i=1}^n (4i)^{-2} \cdot f(OPT) \ ,$$

where the last inequality follows from Lemmata B.8 and 3.4. Plugging this lower bound and $a = 0$ into (4) gives:

$$F(y \wedge (b \cdot \mathcal{N}))$$

$$\geq f(\varnothing) + \int_0^b \left[f(OPT) - \frac{F(y \wedge (z \cdot \mathcal{N}))}{1 - z} - \varepsilon \cdot \left(1 + \frac{1}{1 - b}\right) \cdot \sum_{i=1}^n (4i)^{-2} \cdot f(OPT) \right] dz \ .$$

The solution of this differential equation is:

$$F(y \wedge (b \cdot \mathcal{N})) \geq -(1 - b) \cdot \ln(1 - b) \cdot \left(1 - \varepsilon \cdot \left(1 + \frac{1}{1 - b}\right) \cdot \sum_{i=1}^n (4i)^{-2}\right) \cdot f(OPT) \ .$$

Choosing $b = 1 - e^{-1}$, we get:

$$F(y) = F(y \wedge (1 \cdot \mathcal{N})) \geq F(y \wedge ((1 - e^{-1}) \cdot \mathcal{N})) - \varepsilon e^{-1} \cdot \sum_{i=1}^n (4i)^{-2} \cdot f(OPT)$$

$$\geq e^{-1} \left(1 - 4\varepsilon \cdot \sum_{i=1}^n (4i)^{-2}\right) \cdot f(OPT) - \varepsilon e^{-1} \cdot \sum_{i=1}^n (4i)^{-2} \cdot f(OPT)$$

$$\geq \left(e^{-1} - 4\varepsilon \cdot \sum_{i=1}^n (4i)^{-2}\right) \cdot f(OPT) \ .$$

The corollary now follows by observing that:

$$4 \cdot \sum_{i=1}^{n} (4i)^{-2} \geq \frac{1}{4} + 4 \cdot \int_{1}^{n} \frac{dx}{(4x)^2} = \frac{1}{4} - \frac{1}{4x}\Big|_{1}^{n} < \frac{1}{4} + \frac{1}{4} = \frac{1}{2} \ . \qquad \square$$

We are now ready to prove the competitive ratio of Algorithm 5, and complete the proof of Theorem 1.2.

Lemma B.10. *Algorithm 5 has a competitive ratio of at least $e^{-1} - \varepsilon$.*

Proof. By Observation B.7, the expected value of the set produced by Algorithm 5 is $\mathbb{E}[F(y(\mathcal{V}))]$. By the law of total expectation:

$$\mathbb{E}[F(y(\mathcal{V}))] \geq \Pr[\mathcal{E}] \cdot \mathbb{E}[F(y(\mathcal{V})) \mid \mathcal{E}] \geq (1 - \varepsilon/2) \cdot (e^{-1} - \varepsilon/2) \cdot f(OPT) \geq (e^{-1} - \varepsilon) \cdot f(OPT) \ ,$$

where the second inequality holds by Corollaries B.6 and B.9. $\qquad \square$

C   A Polynomial Time Implementation of Algorithm 3 for the Objective $g(S)$

In this section we present a polynomial time implementation of Algorithm 3 that can be used to get a set S obeying $g(S) \geq (56/627 - \varepsilon) \cdot f(OPT)$, for any constant $\varepsilon > 0$, and thus, together with the rounding described in Section 4, proves the second part of Theorem 1.5. The polynomial time implementation is given as Algorithm 6.

Algorithm 6: Greedy with Theshold for Non-monotone Functions - Polynomial Time Implementation(c, p)

---

1 Let $S_0 \leftarrow \varnothing$.
2 **foreach** *element u_i revealed* **do**
3   **if** $i \leq pk$ **then**
4   Let $S_i \leftarrow S_{i-1} + u_i$.
5   **else**
6   **for** *each $u \in S_{i-1}$* **do**
7   Approximate

$$\begin{aligned} m_{u,i} &= g(S_{i-1} + u_i - u_i') - (1 + c/(pk)) \cdot g(S_{i-1}) \\ &= \mathbb{E}[f(\mathtt{R}(p^{-1}(S_{i-1} + u_i - u_i'))) - (1 + c/(pk)) \cdot f(\mathtt{R}(p^{-1} \cdot S_{i-1}))] \end{aligned}$$

          using $\ell = \lceil 50000\varepsilon^{-2} i^6 (p+c)^2 \ln[2\varepsilon^{-1}pk(2i)^2] \rceil$ samples.

8   Let u_i' be the element of S_{i-1} maximizing $m_{u_i', i}$.
9   **if** $m_{u_i', i} \geq 0$ **then**
10   Let $S_i \leftarrow S_{i-1} + u_i - u_i'$.
11   **else**
12   Let $S_i \leftarrow S_{i-1}$.

Lemma C.1. *Every value $m_{u,i}$ calculated by Algorithm 6 obeys:*

$$\Pr[|m_{u,i} - g(S_{i-1} + u_i - u'_i) + (1 + c/(pk)) \cdot g(S_{i-1})| \le \varepsilon(4i)^{-3} \cdot f(OPT)] \le \varepsilon p^{-1} k^{-1}(2i)^{-2} \ .$$

Proof. The value $m_{u,i}$ is calculated by averaging ℓ samples. For every set S of size at most pk, submodularity implies $f(S) \le p \cdot f(OPT)$. Hence, each sample must belong to the range $[-(p + c/k) \cdot f(OPT), p \cdot f(OPT)]$. Thus, by Lemma B.3,

$$\Pr[|m_{u,i} - g(S_{i-1} + u_i - u'_i) + (1 + c/(pk)) \cdot g(S_{i-1})| \le \varepsilon(4i)^{-3} \cdot f(OPT)] \le 2e^{-\frac{\ell[\varepsilon(4i)^{-3}/(p+c)]^2}{12}}$$

$$\le 2e^{-\ln[2\varepsilon^{-1}pk(2i)^2]} = \varepsilon p^{-1} k^{-1}(2i)^{-2} \ . \qquad \square$$

Corollary C.2. *With probability at least $1 - \varepsilon/2$, all the values $m_{u,i}$ calculated by Algorithm 6 are accurate approximations up to an error of $\varepsilon(4i)^{-3} \cdot f(OPT)$.*

Proof. For every $1 \le i \le n$, Algorithm 6 approximates at most pk different $m_{u,i}$ values. Hence, by the union bound, the probability that all these approximations are correct up to an error of $\varepsilon(4i)^{-3} \cdot f(OPT)$ is at least:

$$1 - pk \cdot \sum_{i=1}^{n} (\varepsilon p^{-1} k^{-1} (2i)^{-2}) = 1 - \varepsilon \cdot \sum_{i=1}^{n} (2i)^{-2} \ge 1 - \frac{\varepsilon}{4} - \varepsilon \cdot \int_1^n \frac{dx}{4x} = 1 - \frac{\varepsilon}{4} + \frac{\varepsilon}{4x}\Big|_1^n > 1 - \frac{\varepsilon}{2} \ . \ \square$$

Let $\mathcal{E}$ be the event that Corollary C.2 holds. As in Section 4, for every $0 \le i \le n$ we define $A_i = \cup_{j=1}^i S_j$. The following two lemmata correspond to Lemmata 4.4 and 4.5.

Lemma C.3. *Assuming $\mathcal{E}$ holds, then for every $pk < i \le n$, $g(S_{i-1} + u_i - u'_i) - g(S_{i-1}) \ge g(u_i \mid A_{i-1}) - g(S_{i-1})/(pk) - 2\varepsilon(4i)^{-3} \cdot f(OPT)$.*

Proof. By the choice of u'_i and an averaging argument,

$$g(S_{i-1} + u_i - u'_i) - g(S_{i-1}) \ge m_{u'_i,i} + c/(pk) \cdot g(S_{i-1}) - \varepsilon(4i)^{-3} \cdot f(OPT)$$

$$\ge \frac{\sum_{u' \in S_{i-1}} [m_{u',i} + c/(pk) \cdot g(S_{i-1}) - \varepsilon(4i)^{-3} \cdot f(OPT)]}{pk}$$

$$\ge \frac{\sum_{u' \in S_{i-1}} [g(S_{i-1} + u_i - u') - g(S_{i-1}) - 2\varepsilon(4i)^{-3} \cdot f(OPT)]}{pk}$$

$$\ge \frac{\sum_{u' \in S_{i-1}} [g(S_{i-1} + u_i) - g(S_{i-1})]}{pk} + \frac{\sum_{u' \in S_{i-1}} [g(S_{i-1} - u') - g(S_{i-1})]}{pk} - 2\varepsilon(4i)^{-3} \cdot f(OPT) \ ,$$

where the last inequality follows from submodularity. The first term on the rightmost hand side is $g(u_i \mid S_{i-1})$, which can be lower bounded by $g(u_i \mid A_{i-1})$ because $S_{i-1} \subseteq A_{i-1}$. The second term on the rightmost hand side can be lower bounded using the submodularity and non-negativity of g as follows:

$$\frac{\sum_{u' \in S_{i-1}} [g(S_{i-1} - u') - g(S_{i-1})]}{pk} \ge \frac{g(\varnothing) - g(S_{i-1})}{pk} \ge -\frac{g(S_{i-1})}{pk} \ . \qquad \square$$

The following lemma relates S_i and A_i.

Lemma C.4. *Assuming $\mathcal{E}$ holds, then for every $0 \le i \le n$, $g(S_i) \ge \frac{c}{c+1} \cdot g(A_i) - 2\varepsilon \cdot \sum_{j=1}^{i} (4j)^{-3} \cdot f(OPT) \ge g(A_i) - \varepsilon/4 \cdot f(OPT)$.*

Proof. The second inequality holds since $\sum_{j=1}^{i}(4j)^{-3} = 0$ for $i = 0$ and for $i > 0$:

$$2 \cdot \sum_{j=1}^{i}(4j)^{-3} \leq \frac{1}{32} + \int_{1}^{i} \frac{dx}{32x^3} = \frac{1}{32} - \frac{1}{64x^2}\Big|_{1}^{i} < \frac{1}{32} + \frac{1}{64} < \frac{1}{4} \ .$$

Hence, in the rest of the proof we concentrate on proving the first inequality.

For $i = 0$, the inequality clearly holds since $g(A_0) = g(\varnothing) = g(S_0)$. Thus, it is enough to prove that for every $1 \leq i \leq n$, $g(S_i) - g(S_{i-1}) \geq \frac{c}{c+1} \cdot [g(A_i) - g(A_{i-1})] - 2\varepsilon(4i)^{-3} \cdot f(OPT)$. If the algorithm does not accept u_i then $S_i = S_{i-1}$ and $A_i = A_{i-1}$, and the claim is trivial. Hence, we only need to consider the case in which the algorithm accepts element u_i.

If $i \leq k$ then $g(S_i) - g(S_{i-1}) = g(u_i \mid S_{i-1}) \geq g(u_i \mid A_{i-1}) = g(A_i) - g(A_{i-1})$, where the inequality follows from submodularity since $S_{i-1} \subseteq A_{i-1}$. If $i > k$, then it is possible to lower bound $g(S_i) - g(S_{i-1}) = g(S_{i-1} + u_i - u_i') - g(S_{i-1})$ in two ways. First, a lower bound of $g(u_i \mid A_{i-1}) - g(S_{i-1})/k - 2\varepsilon(4i)^{-3} \cdot f(OPT)$ is given by Lemma 4.1. The second lower bound states that, since the algorithm accepts u_i,

$$g(S_{i-1} + u_i - u_i') - g(S_{i-1}) \geq m_{u_i',i} + c \cdot g(S_{i-1})/(pk) - \varepsilon(4i)^{-3} \cdot f(OPT)$$
$$\geq c \cdot g(S_{i-1})/(pk) - \varepsilon(4i)^{-3} \cdot f(OPT) \ .$$

Using both lower bounds, we get:

$$g(S_i) - g(S_{i-1}) \geq \max\left\{g(u_i \mid A_{i-1}) - \frac{g(S_{i-1})}{pk}, \frac{c \cdot g(S_{i-1})}{pk}\right\} - 2\varepsilon(4i)^{-3} \cdot f(OPT)$$
$$\geq \frac{c \cdot [g(u_i \mid A_{i-1}) - g(S_{i-1})/(pk)] + 1 \cdot [c \cdot g(S_{i-1})/(pk)]}{c+1} - 2\varepsilon(4i)^{-3} \cdot f(OPT)$$
$$\geq \frac{c}{c+1} \cdot g(u_i \mid A_{i-1}) - 2\varepsilon(4i)^{-3} \cdot f(OPT)$$
$$= \frac{c}{c+1} \cdot [g(A_i) - g(A_{i-1})] - 2\varepsilon(4i)^{-3} \cdot f(OPT) \ . \qquad \square$$

We also need to show that $g(S_i)$ is roughly monotone as a function of i.

Lemma C.5. *Assuming $\mathcal{E}$ holds, then for $pk < i \leq n$, $g(S_i) - g(S_{i-1}) \geq -\varepsilon(4i)^{-3} \cdot f(OPT)$.*

Proof. If $S_i = S_{i-1}$ then the claim is trivial. Otherwise, we must have $m_{u_i',i} \geq 0$, and thus,

$$g(S_i) - g(S_{i-1}) \geq \frac{c \cdot g(S_i)}{pk} - \varepsilon(4i)^{-3} \cdot f(OPT) \geq -\varepsilon(4i)^{-3} \cdot f(OPT) \ . \qquad \square$$

Corollary C.6. *Assuming $\mathcal{E}$ holds, then for $pk \leq i \leq n$, $g(S_i) \geq g(S_n) - \varepsilon(8i)^2 \cdot f(OPT)$.*

Proof. By Lemma C.5,

$$g(S_n) - g(S_i) \geq -\varepsilon \cdot \sum_{j=i+1}^{n}(4i)^{-3} \cdot f(OPT) \geq -\varepsilon \cdot \int_{i}^{n} \frac{dx}{64x^3} \cdot f(OPT)$$
$$= \varepsilon \cdot \frac{1}{128x^2}\Big|_{i}^{n} \cdot f(OPT) \geq -\frac{\varepsilon}{128i^2} \cdot f(OPT) \geq -\frac{\varepsilon}{(8i)^2} \cdot f(OPT) \ . \qquad \square$$

Using the above claims, it is possible to get a guarantee on $g(S_n)$ assuming $\mathcal{E}$ holds. The following corollary corresponds to Corollary 4.7.

29

Corollary C.7. *Assuming $\mathcal{E}$ holds, Algorithm 6 produces a set S_n such that:*

$$g(S_n) \geq \left(\frac{cp^{-1}(1 - p^{-1})}{(c + 1)(1 + p^{-1}c)} - \frac{\varepsilon}{2} \right) \cdot f(OPT) \ .$$

Proof. Consider an element $u_i \in OPT \setminus A_i$. Since u_i was rejected by Algorithm 6, we must have $m_{u'_i, i} < 0$, which implies:

$$\frac{c \cdot g(S_{i-1})}{pk} + \varepsilon(4i)^{-3} \cdot f(OPT) > g(S_{i-1} + u_i - u'_i) - g(S_{i-1})$$

$$\geq g(u_i \mid A_{i-1}) - \frac{g(S_{i-1})}{pk} - 2\varepsilon(4i)^{-3} \cdot f(OPT) \ ,$$

where the second inequality follows from Lemma C.3. Rearranging yields:

$$g(u_i \mid A_{i-1}) < \frac{c + 1}{pk} \cdot g(S_{i-1}) + 3\varepsilon(4i)^{-3} \cdot f(OPT) \leq \frac{c + 1}{pk} \cdot g(S_n) + 4\varepsilon(8i)^{-2} \cdot f(OPT) \ ,$$

where the last inequality uses Corollary C.6. By the submodularity of g and Lemma C.4, we now get:

$$g(A_n \cup OPT) \leq g(A_n) + \sum_{u \in OPT \setminus A_n} g(u_i \mid A_n)$$

$$< g(A_n) + \sum_{u \in OPT \setminus A_n} \left(\frac{c + 1}{pk} \cdot g(S_n) \right) + 4\varepsilon \cdot \sum_{i=1}^{n} (8i)^{-2} \cdot f(OPT)$$

$$\leq g(A_n) + p^{-1}(c + 1) \cdot g(S_n) + 4\varepsilon \cdot \sum_{i=1}^{n} (8i)^{-2} \cdot f(OPT)$$

$$\leq \frac{(c + 1)(1 + p^{-1}c)}{c} \cdot g(S_n) + \frac{c + 1}{c} \cdot \frac{\varepsilon}{4} \cdot f(OPT) + 4\varepsilon \cdot \sum_{i=1}^{n} (8i)^{-2} \cdot f(OPT) \ .$$

Observe now that:

$$4\varepsilon \cdot \sum_{i=1}^{n} (8i)^{-2} \cdot f(OPT) \leq \frac{\varepsilon}{16} + \int_1^n \frac{dx}{16x^2} = \frac{\varepsilon}{16} - \frac{1}{16x} \bigg|_1^n \leq \frac{\varepsilon}{16} + \frac{1}{16} < \frac{\varepsilon}{4} \ ,$$

and, by Lemma 4.6,

$$g(A_n \cup OPT) = F(p^{-1} \cdot OPT + p^{-1} \cdot (A_n \setminus OPT)) \geq p^{-1}(1 - p^{-1}) \cdot f(OPT) \ .$$

Combining the three last inequalities yields:

$$\left[p^{-1}(1 - p^{-1}) - \frac{c + 1}{c} \cdot \frac{\varepsilon}{2} \right] \cdot f(OPT) \leq \frac{(c + 1)(1 + p^{-1}c)}{c} \cdot g(S_n) \ .$$

The corollary now follow by dividing the last inequality by $\frac{(c+1)(1+p^{-1}c)}{c}$ and observing that $1 + p^{-1}c \geq 1$. $\square$

Corollary C.8. *Algorithm 6 produces a set S_n such that:*

$$\mathbb{E}[g(S_n)] \geq \left(\frac{cp^{-1}(1 - p^{-1})}{(c + 1)(1 + p^{-1}c)} - \varepsilon \right) \cdot f(OPT) \ .$$

Hence, for $c = 7/4$ and $p = 3$, $g(S_n) \geq (56/627 - \varepsilon) \cdot f(OPT)$.

Proof. By the law of total expectation,

$$\mathbb{E}[g(S_n)] \geq \Pr[\mathcal{E}] \cdot \mathbb{E}[g(S_n) \mid \mathcal{E}] \geq (1 - \varepsilon/2) \cdot \left(\frac{cp^{-1}(1 - p^{-1})}{(c+1)(1 + p^{-1}c)} - \frac{\varepsilon}{2} \right) \cdot f(OPT)$$
$$\geq \left(\frac{cp^{-1}(1 - p^{-1})}{(c+1)(1 + p^{-1}c)} - \varepsilon \right) \cdot f(OPT) \;,$$

where the second inequality holds by Corollaries C.2 and C.7. $\square$