General Bounds for Incremental Maximization[*]

Aaron Bernstein[1], Yann Disser[2], and Martin Groß[3]

1   **TU Berlin, Berlin, Germany**
    bernstei@gmail.com
2   **TU Darmstadt, Darmstadt, Germany**[†]
    disser@mathematik.tu-darmstadt.de
3   **University of Waterloo, Waterloo, Canada**[‡]
    mgrob@uwaterloo.ca

─── **Abstract** ───

We propose a theoretical framework to capture incremental solutions to cardinality constrained maximization problems. The defining characteristic of our framework is that the cardinality/support of the solution is bounded by a value $k \in \mathbb{N}$ that grows over time, and we allow the solution to be extended one element at a time. We investigate the best-possible competitive ratio of such an incremental solution, i.e., the worst ratio over all k between the incremental solution after k steps and an optimum solution of cardinality k. We define a large class of problems that contains many important cardinality constrained maximization problems like maximum matching, knapsack, and packing/covering problems. We provide a general 2.618-competitive incremental algorithm for this class of problems, and show that no algorithm can have competitive ratio below 2.18 in general.

In the second part of the paper, we focus on the inherently incremental greedy algorithm that increases the objective value as much as possible in each step. This algorithm is known to be 1.58-competitive for submodular objective functions, but it has unbounded competitive ratio for the class of incremental problems mentioned above. We define a relaxed submodularity condition for the objective function, capturing problems like maximum (weighted) (b-)matching and a variant of the maximum flow problem. We show that the greedy algorithm has competitive ratio (exactly) 2.313 for the class of problems that satisfy this relaxed submodularity condition.

Note that our upper bounds on the competitive ratios translate to approximation ratios for the underlying cardinality constrained problems.

1   Introduction

Practical solutions to optimization problems are often inherently *incremental* in the sense that they evolve historically instead of being established in a one-shot fashion. This is especially true when solutions are expensive and need time and repeated investments to be implemented, for example when optimizing the layout of logistics and other infrastructures.

Figure 1 Example showing that the *s-t*-flow problem does not always admit good incremental solutions, where $\varepsilon > 0$ is arbitrarily small.

In this paper, we propose a theoretical framework to capture *incremental maximization* problems in some generality.

We describe an incremental problem by a set U containing the possible elements of a solution, and an objective function $f\colon 2^U \to \mathbb{R}^+$ that assigns to each solution $S \subseteq U$ some non-negative value $f(S)$. We consider problems of the form

$$\max\ f(S) \tag{1}$$
$$\text{s.t.}\ |S| \le k$$
$$S \subseteq U,$$

where $k \in \mathbb{N}$ grows over time.

An incremental solution $\vec{S}$ is given by an order $\{s_1, s_2, \dots\} := U$ in which the elements of U are to be added to the solution over time. A good incremental solution needs to provide a good solution after k steps, for every k, compared to an optimum solution $S_k^\star$ with k elements, where we let $S_k^\star \in \arg\max_{S \subseteq U, |S|=k} f(S)$ and $f_k^\star := f(S_k^\star)$. Formally, we measure the quality of an incremental solution by its *competitive ratio*. For $\vec{S}_k := \{s_1, \dots, s_k\} \subseteq U$ being the first k elements of $\vec{S}$, we say that $\vec{S}$ is (strictly) ρ-competitive if

$$\max_{k\in\{1,\dots,|U|\}} \frac{f_k^\star}{f(\vec{S}_k)} \le \rho.$$

An algorithm is called ρ-competitive if it always produces a ρ-competitive solution, and its *competitive ratio* is the infimum over all $\rho \ge 1$ such that it is ρ-competitive. Notice that we do not require the algorithm to run in polynomial time.

While all cardinality constrained optimization problems can be viewed in an incremental setting, clearly not all such problems admit good incremental solutions. For example, consider a cardinality constrained formulation of the classical maximum *s-t*-flow problem: For a given graph $G = (V, E)$, two vertices $s, t \in V$ and capacities $u\colon E \to \mathbb{R}^+$, we ask for a subset $E' \subseteq E$ of cardinality $k \in \mathbb{N}$ such that the maximum flow in the subgraph (V, E') is maximized. The example in Figure 1 shows that we cannot hope for an incremental solution that is simultaneously close to optimal for cardinalities 1 and 2.

In order to derive general bounds on the competitive ratio of incremental problems, we need to restrict the class of objective functions f that we consider. Intuitively, the unbounded competitive ratio in the flow example comes from the fact that we have to invest in the *s-t*-path of capacity 1 as soon as possible, but this path only yields its payoff once it is completed after two steps.

In order to prevent this and similar behaviors, we require f to be monotone (i.e., $f(S) \le f(T)$ if $S \subseteq T$) and sub-additive (i.e., $f(S) + f(T) \ge f(S \cup T)$). Many important optimization problems satisfy these weak conditions, and we give a short list of examples below. We will see that all these (and many more) problems admit incremental solutions with a bounded competitive ratio. More specifically, we develop a general 2.618-competitive

incremental algorithm that can be applied to a broad class of problems, including all problems mentioned below. We illustrate in detail how to apply our model to obtain an incremental variant of the matching problem, and then list incremental versions of other important problems that are obtained analogously.

- MAXIMUM WEIGHTED MATCHING: Consider a graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}_{\geq 0}$. If we think of edges as *potential* connections and edge weights as *potential* payoffs, then it is not enough to find the final matching because we cannot construct the edges all at once: the goal is to find a sequence of edges that achieves a high pay-off in the short, the medium, and the long term. In terms of our formal framework, we add edges to a set S one at a time with $U = E$ and $f(S)$ is the maximum weight of a matching $M \subseteq S$. In order to be ρ-competitive, we need that, after k steps for every k, our solution S of cardinality k is no worse than a factor of ρ away from the optimum solution of cardinality k, i.e., $f(S) \geq f(S_k^\star)/\rho$.
  This model captures the setting where the infrastructure (e.g. the matching, the knapsack, the covering, or the flow) must be built up over time. The online model would be too restrictive in this setting because here we know our options in advance. Note that, as we add more edges, the set of edges S only needs to contain a large matching M, but does not have to be a matching itself; The matching M can change to an arbitrary subset of S from one cardinality to the next and does not have to stay consistent. This ensures that $f(S)$ is monotonically increasing, and is in keeping with the infrastructures setting where the potential regret present in the online model does not apply: building more infrastructure can only help, since once it is built, we can change how it is used. Accordingly, in all the problems below the set S does not have to be a valid solution to the cardinality constrained problem at hand, but rather needs to *contain* a good solution as a subset. The objective $f(S)$ is consistently defined to be the value of the best solution that is a subset of S. Notice that this approach can easily be generalized to MAXIMUM b-MATCHING.
- SET PACKING: Given a set of weighted sets $\mathcal{X}$ we ask for an incremental subset $S \subseteq \mathcal{X}$ where $f(S)$ is the maximum weight of mutually disjoint subsets in S. This problem captures many well-known problems such as MAXIMUM HYPERGRAPH MATCHING and MAXIMUM INDEPENDENT SET.
- MAXIMUM COVERAGE: Given a set of weighted sets $\mathcal{X} \subseteq 2^U$ over an universe of elements U, we ask for an incremental subset $S \subseteq \mathcal{X}$, where $f(S)$ is the weight of elements in $\bigcup_{X \in S} X$. This problem captures maximization versions of clustering and location problems. We can include opening costs $c : \mathcal{X} \to \mathbb{R}_{\geq 0}$ by letting $f(S)$ be the maximum over all subsets $S' \subseteq S$ of the number (or weight) of the sets in S' minus their opening costs.
- KNAPSACK: Given a set X of items, associated sizes $s : X \to \mathbb{R}_{\geq 0}$ and values $v : X \to \mathbb{R}_{\geq 0}$, and a knapsack of capacity 1, we ask for an incremental subset $S \subseteq X$, where $f(S)$ is the largest value $\sum_{x \in S'} v(x)$ of any subset $S' \subseteq S$ with $\sum_{x \in S'} s(x) \leq 1$. This problem can be generalized to MULTI-DIMENSIONAL KNAPSACK by letting item sizes be vectors and letting the knapsack have a capacity in every dimension.
- DISJOINT PATHS: Given a graph $G = (V, E)$, a set of pairs $\mathcal{X} \subseteq V^2$ with weights $w : \mathcal{X} \to \mathbb{R}_{\geq 0}$, we ask for an incremental subset $S \subseteq \mathcal{X}$, where $f(S)$ is the maximum weight of a subset $S' \subseteq S$, such that G contains mutually disjoint paths between every pair in S'.
- MAXIMUM BRIDGE-FLOW: We argued above that the maximum s-t-flow problem is not amenable to the incremental setting because it does not pay off to build paths partially.

To overcome this, we consider a natural restriction of the flow problem where most edges
are freely available to be used, and only the edges of a directed s-t-cut need to be built
incrementally. If the directed cut has no backward edges, every s-t-path contains exactly
one edge that needs to be built, and we never have to invest multiple steps to establish a
single path. This problem captures logistical problems where links need to be established
between two clusters, like when bridges need to be built across a river, cables across an
ocean, or when warehouses need to be opened in a supplier-warehouse-consumer network.
Formally, given a directed graph $G = (V, E)$ with capacities $u \colon E \to \mathbb{R}$, vertices $s, t \in V$,
and a directed s-t-cut $C \subseteq E$ induced by the partition (U, W) of V such that the directed
cut induced by (W, U) is empty, we ask for an incremental subset $S \subseteq C$ where $f(S)$ is
the value of a maximum flow in the subgraph $(V, E \setminus (C \setminus S))$.

It is easy to verify that all the problems mentioned above (and many more) indeed have a
monotone and sub-additive objective function. In addition, each one of these problems satisfies
the following property: For every $S \subseteq U$, there exists $s \in S$ with $f(S \setminus \{s\}) \geq f(S) - f(S)/|S|$.
We call this property the *accountability* property – to our knowledge, it has not been named
before. Intuitively, this property ensures that the value of a set $S \subseteq U$ is the sum of individual
contributions of its elements, and there cannot be additional value that emerges only when
certain elements of U combine. While it is easy to formulate artificial problems that have
monotonicity and sub-additivity but no accountability, we were not able to identify any
natural problems of this kind. This justifies to add accountability to the list of properties
that we require of incremental problems.

▶ **Definition 1.** Given a set of elements U, and a function $f \colon 2^U \to \mathbb{R}$, we say that the
function f is *incremental* if it satisfies the following properties for every $S, T \subseteq U$:
1. (monotonicity): $S \subseteq T \Rightarrow f(S) \leq f(T)$,
2. (sub-additivity): $f(S) + f(T) \geq f(S \cup T)$,
3. (accountability): $\exists s \in S \colon f(S \setminus \{s\}) \geq f(S) - f(S)/|S|$.
We say that a cardinality constrained problem with increasing cardinality (eq. (1)) is
incremental if its objective function is incremental.

Observe that a ρ-competitive incremental algorithm immediately yields a ρ-approximation
algorithm for the underlying cardinality constrained problem, with the caveat that the
resulting approximation algorithm might not be efficient since we make no demands on the
runtime of the incremental algorithm. The converse is rarely the case since approximation
algorithms usually do not construct their solution in incremental fashion. A prominent
exception are *greedy* algorithms that are inherently incremental in the sense that they pick
elements one-by-one such that each pick increases the objective by the maximum amount
possible. This type of a greedy algorithm has been studied as an approximation algorithm
for many cardinality constrained problems, and approximation ratios translate immediately
to competitive ratios for the incremental version of the corresponding problem. In particular,
the greedy algorithm is known to have competitive ratio (exactly) $\frac{e}{e-1} \approx 1.58$ if the objective
function f is monotone and submodular [27]. Note, however, that of all the incremental
problems listed above, only MAXIMUM COVERAGE (without opening costs) has a submodular
objective function. It is also known that if we relax the submodularity requirement and
allow f to be the minimum of two monotone (sub-)modular functions, the greedy algorithm
can be arbitrarily bad [21]. We provide a different relaxation of submodularity that captures
MAXIMUM (WEIGHTED) (b-)MATCHING and MAXIMUM BRIDGE-FLOW, and where the
greedy algorithm has a bounded competitive/approximation ratio.

Our Results. As our first result, we show that every incremental problem admits a bounded competitive ratio.

▶ **Theorem 2.** *Every incremental problem admits a $(1 + \varphi)$-competitive algorithm, where φ is the golden ratio and $(1 + \varphi) \approx 2.618$. No general deterministic algorithm for this class of problems has a competitive ratio of 2.18 or better.*

Again, note that we make no guarantees regarding the running time of our incremental algorithm. In fact, our algorithm relies on the ability to compute the optimum of the underlying cardinality constrained problem for increasing cardinalities. If we can provide an efficient approximation of this optimum, we get an efficient incremental algorithm in the following sense.

▶ **Corollary 3.** *If there is a polynomial time α-approximation algorithm for a cardinality constrained problem with incremental objective function, then we can design a polynomial time $\alpha(1 + \varphi)$-competitive incremental algorithm.*

We also analyze the approximation/competitive ratio of the greedy algorithm. We observe that for many incremental problems like KNAPSACK, MAXIMUM INDEPENDENT SET, and DISJOINT PATHS, the greedy algorithm has an unbounded competitive ratio. On the other hand, we define a relaxation of submodularity called α-*augmentable* under which the greedy algorithm has a bounded competitive ratio. In particular, this relaxation captures our cardinality constrained versions of MAXIMUM (WEIGHTED) (b-)MATCHING and MAXIMUM BRIDGE-FLOW, where the incremental set S need not be feasible but only contain a good feasible subset. We get the following result, where the tight lower bound for $\alpha = 2$ is obtained for MAXIMUM BRIDGE-FLOW. Notice that for $\alpha = 1$, we obtain the $\frac{e}{e-1} \approx 1.58$ bound that is known for submodular functions. For $\alpha = 2$, the bound is $\frac{e^2}{e^2-1} \approx 2.313$.

▶ **Theorem 4.** *For every cardinality constrained problem with an α-augmentable objective (defined below), the greedy algorithm has approximation/competitive ratio $\alpha \frac{e^\alpha}{e^\alpha - 1}$. This bound is tight for the greedy algorithm on problems with 2-augmentable objectives, which includes* MAXIMUM (WEIGHTED) (b-)MATCHING *and* MAXIMUM BRIDGE-FLOW.

We emphasize that the families of instances we construct to obtain the lower bounds in Theorems 2 and 4 require the number of elements to tend to infinity, since it takes time for incremental solutions to sufficiently fall behind the optimum solution.

Related Work. Most work on incremental settings has focused on cardinality constrained *minimization* problems. A prominent exception is the robust matching problem, introduced by Hassin and Rubinstein [16]. This problem asks for a weighted matching M with the property that, for every value k, the total weight of the $\min(k, |M|)$ heaviest edges of M comes close to the weight of a maximum weight matching of cardinality k. Note that this differs from our definition of incremental matchings in that the robust matching problem demands that the "incremental" solution consists of a matching, while we allow any edge set that contains a heavy matching as a subset. Since their model is more strict, all of the following competitive ratios carry over to our setting. Note that, in contrast to our setting, the objective function of the robust matching problem is submodular, and hence the greedy algorithm has competitive ratio at most $\frac{e}{e-1} \approx 1.58$ [27]. Hassin and Rubinstein [16] gave an improved, deterministic algorithm that achieves competitive ratio $\sqrt{2} \approx 1.414$. They also give a tight example for the $\sqrt{2}$ ratio, which also works in our incremental setting. Fujita et al. [11] extended this result to matroid intersection, and Kakimura and Makino [18] showed

that every independence system allows for a $\sqrt{\mu}$-competitive solution, with μ being the extendibility of the system. Matuschke at al. [24] describe a randomized algorithm for this problem that, under the assumption that the adversary does not know the outcome of the randomness, has competitive ratio $\ln(4) \approx 1.386$.

A variant of the knapsack problem with a similar notion of robustness was proposed by Kakimura et al. [19]. In this problem a knapsack solution needs to be computed, such that, for every k, the value of the k most valuable items in the knapsack compares well with the optimum solution using k items, for every k. Kakimura et al. [19] restrict themselves to polynomial time algorithms and show that under this restriction a bounded competitive ratio is possible only if the rank quotient of the knapsack system is bounded. In contrast, our results show that if we do not restrict the running time and if we only require our solution to *contain* a good packing with k items for every k, then we can be $(1 + \varphi)$-competitive using our generic algorithm, even for generalizations like MULTI-DIMENSIONAL KNAPSACK. If we restrict the running time and use the well-known PTAS for the knapsack problem [17, 22], we still get a $(1 + \varphi)(1 + \varepsilon)$-competitive algorithm. Megow and Mestre [25] and Disser et al. [7] considered another variant of the knapsack problem that asks for an order in which to pack the items that works well for every knapsack capacity. Kobayashi and Takizawa [20] study randomized strategies for cardinality robustness in the knapsack problem.

Hartline and Sharp [15] considered an incremental variant of the maximum flow problem where capacities increase over time. This is in contrast to our framework where the cardinality of the solution increases.

Incremental solutions for cardinality constrained minimization problems have been studied extensively, in particular for clustering [3, 6], k-median [4, 10, 26], minimium spanning tree [1, 12], and facility location [13]. An important result in this domain is the incremental framework given by Lin et al. [23]. This general framework allows to devise algorithms for every incremental minimization problem for which a suitable augmentation subroutine can be formulated. Lin et al. [23] used their framework to match or improve many of the known specialized bounds for the problems above and to derive new bounds for covering problems. In contrast to their result, our incremental framework allows for a general algorithm that works out-of-the-box for a broad class of incremental maximization problems and yields a constant (relatively small) competitive ratio.

Abstractly, incremental problems can be seen as optimization problems under uncertainty. Various approaches to handling uncertain input data have been proposed, ranging from robust and stochastic optimization to streaming and exploration. On this level, incremental problems can be seen as a special case of online optimization problems, i.e., problems where the input data arrives over time (see [2, 9]). Whereas online optimization in general assumes adversarial input, incremental problems restrict the freedom of the adversary to deciding when to stop, i.e., the adversary may choose the cardinality k while all other data is fixed and known to the algorithm. Online problems with such a "non-adaptive" adversary have been studied in other contexts [5, 8, 14]. Note that online problems demand irrevocable decisions in every time step – a requirement that may be overly restrictive in many settings where solutions develop over a long time period. In contrast, our incremental model only requires a growing solution "infrastructure" and allows the actual solution to change arbitrarily over time within this infrastructure.

2 A competitive algorithm for incremental problems

In this section, we show the second part of Theorem 2, i.e., we give an incremental algorithm that is $(1 + \varphi \approx 2.618)$-competitive for all incremental problems. For convenience, we define

the *density* δ_S of a set $S \subseteq U$ via $\delta_S := f(S)/|S|$, and we let $\delta_k^\star := \delta_{S_k^\star}$ denote the optimum density for cardinality k. Our algorithm relies on the following two observations that follow from the accountability of the objective function.

▶ **Lemma 5.** *In every incremental problem and for every cardinality k, there is an ordering $\vec{S}_k^\star := \{s_1^\star, s_2^\star, \ldots, s_k^\star\} := S_k^\star$, such that $\delta_{\{s_1^\star, \ldots, s_i^\star\}} \geq \delta_{\{s_1^\star, \ldots, s_{i+1}^\star\}}$ for all $i \in \{1, \ldots, k-1\}$. We say that $\vec{S}_k^\star$ is a* greedy order *of $S_k^\star$.*

Proof. By accountability of f, there is an element $s_k^\star \in S_k^\star$ for which

$$\delta_{S_k^\star \setminus \{s_k^\star\}} = \frac{f(S_k^\star \setminus \{s_k^\star\})}{k-1} \geq \frac{f(S_k^\star)}{k} = \delta_{S_k^\star}.$$

We can repeat this argument for $s_{k-1}^\star \in S_k^\star \setminus \{s_k^\star\}$, $s_{k-2}^\star \in S_k^\star \setminus \{s_k^\star, s_{k-1}^\star\}$, etc. to obtain the desired ordering $\vec{S}_k^\star$. ◀

▶ **Lemma 6.** *In every incremental problem and for every $1 \leq k' \leq k$ we have $\delta_{k'}^\star \geq \delta_k^\star$.*

Proof. Fix any cardinality $k > 1$. By accountability of the objective function f, there is an element $s^\star \in S_k^\star$ with

$$\delta_k^\star = \frac{f(S_k^\star)}{k} \leq \frac{f(S_k^\star \setminus \{s^\star\})}{k-1} \leq \frac{S_{k-1}^\star}{k-1} = \delta_{k-1}^\star.$$

It follows that $\delta_k^\star$ is monotonically decreasing in k. ◀

Now, we define $k_0 := 1$ and $k_i := \lceil (1+\varphi)k_{i-1} \rceil$ for all positive integers i. Our algorithm operates in phases $i \in \{0, 1, \ldots\}$. In each phase i, we add the elements of the optimum solution $S_{k_i}^\star$ of cardinality k_i to our incremental solution in greedy order (Lemma 5). Note that we allow the algorithm to add elements multiple times (without effect) in order to not complicate the analysis needlessly (of course we would only improve the algorithm by skipping over duplicates). In the following, we denote by t_i the number of steps (possibly without effect) until the end of phase i, i.e., we let $t_0 := k_0$ and $t_i := t_{i-1} + k_i$.

▶ **Lemma 7.** *For every phase $i \in \{0, 1, \ldots\}$, we have $t_i \leq \varphi k_i$.*

Proof. We use induction over i, with the case $i = 0$ being trivial, since $t_0 = k_0$. Now assume that $t_{i-1} \leq \varphi k_{i-1}$ for some $i \geq 1$. Using the property $\frac{\varphi}{\varphi+1} = \varphi - 1$ of the golden ratio, we get

$$t_i = t_{i-1} + k_i \leq \varphi k_{i-1} + k_i \leq \frac{\varphi}{\varphi+1} k_i + k_i = \varphi k_i. \qquad ◀$$

Finally, we show the solution $\vec{S}$ computed by our algorithm is $(1+\varphi)$-competitive.

▶ **Theorem 8.** *For every cardinality k, we have $f(\vec{S}_k) \geq f_k^\star/(1+\varphi)$.*

Proof. We use induction over k. The claim is true for $k = t_0 = 1$, since $\vec{S}_1 = S_1^\star$ by definition of the algorithm. For the inductive step, we prove that if the claim is true for $k = t_{i-1}$, then it remains true for all $k \in \{t_{i-1}+1, \ldots, t_i\}$. Recall that $k_i = \lceil (1+\varphi)k_{i-1} \rceil$. By Lemma 7, we have

$$t_{i-1} \leq \varphi k_{i-1} < k_i < t_{i-1} + k_i = t_i,$$

and we can therefore distinguish the following cases.

Case 1: $t_{i-1} < k < k_i$.   Since $k > t_{i-1}$, our algorithm has already completed phase $i-1$ and added all elements of $S^\star_{k_{i-1}}$, so we have $f(\vec{S}_k) \geq f^\star_{k_{i-1}}$. Because k is an integer and $k < k_i = \lceil(1+\varphi)k_{i-1}\rceil$, we have that $k < (1+\varphi)k_{i-1}$. By Lemma 6, we thus have

$$f^\star_k = \delta^\star_k \cdot k < \delta^\star_{k_{i-1}} \cdot (1+\varphi)k_{i-1} = (1+\varphi)f^\star_{k_{i-1}} \leq (1+\varphi)f(\vec{S}_k).$$

Case 2: $k_i \leq k \leq t_i$.   At time k, our algorithm has already completed the first $k - t_{i-1}$ elements of $S^\star_k$. Since the algorithm adds the elements of $S^\star_{k_i}$ in greedy order, we have $f(\vec{S}_k) \geq (k - t_{i-1})\delta^\star_{k_i}$. On the other hand, since $k \geq k_i$, by Lemma 6 we have $f^\star_k = k \cdot \delta^\star_k \leq k \cdot \delta^\star_{k_i}$. In order to complete the proof, it is thus sufficient to show that $k \leq (1+\varphi)(k - t_{i-1})$. To see this, let $k = k_i + k'$ for some non-negative integer k'. Because t_{i-1} is integral, Lemma 7 implies $t_{i-1} \leq \lfloor\varphi k_{i-1}\rfloor$. Since φ is irrational and k_{i-1} is integral, φk_{i-1} cannot be integral, thus

$$k - t_{i-1} = k' + k_i - t_{i-1} \geq k' + \lceil(1+\varphi)k_{i-1}\rceil - \lfloor\varphi k_{i-1}\rfloor = k' + k_{i-1} + 1.$$

This completes the proof, since

$$(1+\varphi)(k - t_{i-1}) \geq (1+\varphi)(k' + k_{i-1} + 1) > k' + (1+\varphi)k_{i-1} + 1 \geq k' + k_i = k. \qquad \blacktriangleleft$$

Corollary 3 follows if we replace $S^\star_{k_i}$ by an α-approximate solution for cardinality k_i.

3 Lower bound on the best-possible competitive ratio

In this section, we show the second part of Theorem 2, i.e., we give a lower bound on the best-possible competitive ratio for the maximization of incremental problems. For this purpose, we define the REGION CHOOSING problem. In this problem, we are given N disjoint sets $R_1, \ldots, R_N$, called *regions*, with region R_i containing i elements with a value of $\delta(i)$ each. We say that $\delta(i)$ is the *density* of region R_i. The total value of all elements in the region R_i is $v(i) := i \cdot \delta(i)$ for all $i \in \{1, \ldots, N\}$.

The objective is to compute an incremental solution $S \subseteq U := \bigcup_{i=1}^N R_i$ such that the maximum value of the items from a single region in S is large. Formally, the objective function is given by $f(S) := \max_{i \in \{1, \ldots, N\}} |R_i \cap S| \cdot v(i)$.

▶ **Observation 9.** REGION CHOOSING *is an incremental problem.*[1]

For our lower bound, we set $\delta(i) := i^{\beta-1}$ for some $\beta \in (0, 1)$ that we will choose later. For this choice of β, we have $\delta(i) < \delta(j)$ and $v(i) > v(j)$ for $0 \leq j < i \leq N$. Also, for $N \to \infty$ we have $\lim_{i\to\infty} v(i) = \infty$. We call instances of the REGION CHOOSING problem in this form β-*decreasing*. Observe that in every β-decreasing instance the optimum solution of cardinality $i \leq N$ is to take all i elements from region R_i. This solution has value $f^\star_i = i^\beta$.

In order to impose a lower bound on the best-possible competitive ratio for β-decreasing instances, we need some insights into the structure of incremental solutions with an optimal competitive ratio. First, consider a solution that picks only $i' < i$ elements from region R_i. In this case, we could have picked i' elements from region $R_{i'}$ instead – this would only improve the solution, since densities are decreasing. Secondly, if we take i elements from region R_i, it is always beneficial to take them in an uninterrupted sequence before taking any elements from a region R_j with $j > i$: Our objective depends only on the region with the

---

[1] This and all other missing proofs are deferred to the full version of this paper.

most value, therefore it never helps to take elements from different regions in an alternating fashion. This leads us the following observation.

▶ **Observation 10.** *For every β-decreasing instance of* REGION CHOOSING *there is an incremental solution with optimal competitive ratio of the following structure: For $k_0 < k_1 < \cdots < k_m \in \mathbb{N}$ with $m \in \mathbb{N}$, it takes k_0 elements from region R_{k_0}, followed by k_1 elements from R_{k_1}, and so on, until finally k_m elements from region R_{k_m} are chosen.*

Thus, we can describe an algorithm for the region-choosing problem by an increasing sequence of region indices $k_0, \ldots, k_m$. Note that, in order to have a bounded competitive ratio if $N \to \infty$, we must have $m \to \infty$, since $\lim_{i \to \infty} v(i) \to \infty$. We are interested in a cardinality for which an incremental solution given by $k_0, \ldots, k_m$ has a bad competitive ratio. We define

$$\alpha_i := \frac{1}{k_i} \sum_{j=0}^{i} k_j \qquad \text{for all } i \in \{0, \ldots, m\}.$$

Observe that $\alpha_i > 1$ for all $i \in \{1, \ldots, m\}$. We know that the value of the optimum solution for cardinality $\alpha_i k_i$ is $v(\alpha_i k_i) = (\alpha_i k_i)^\beta$, whereas the incremental solution only achieves a value of $v(k_i) = (k_i)^\beta$. This allows us to derive the following necessary condition on the α_i-values of ρ-competitive solutions.

▶ **Observation 11.** *If an incremental solution defined by a sequence $k_0, \ldots, k_m$ is ρ-competitive for some $\rho \geq 1$, we must have*

$$\rho \geq \frac{v(\alpha_i k_i)}{v(k_i)} = \left(\frac{\alpha_i k_i}{k_i} \right)^\beta = \alpha_i^\beta \iff \alpha_i \leq \rho^{\frac{1}{\beta}} \qquad \text{for all } i \in \{0, \ldots, m\}. \tag{2}$$

We will exclude a certain range of values of ρ by showing that we can find a $\beta \in (0, 1)$ such that, for a sufficiently large number of regions N, necessary condition (2) is violated. We do this by showing that, for some $i^\star \in \mathbb{N}$ and some fixed $\varepsilon > 0$, we have $\alpha_{i+1} - \alpha_i > \varepsilon$ for all $i \geq i^\star$, i.e., as i goes to ∞, condition (2) must eventually be violated. The following definition relates a value of $\beta \in (0, 1)$ to a lower bound on the competitive ratio ρ for β-decreasing instances.

▶ **Definition 12.** A pair (ρ, β) with $\rho \geq 1$ and $\beta \in (0, 1)$ is *problematic* if there is $\varepsilon > 0$ such that for all $x \in (1, \rho^{1/\beta}]$ it holds that $h_{\rho, \beta}(x) < 0$, where

$$h_{\rho, \beta}(x) := (\rho^{\frac{1}{\beta}} + \varepsilon - x)^{\frac{1}{1-\beta}} - \frac{x}{x - 1 + \varepsilon}.$$

We show that problematic pairs indeed have the intended property.

▶ **Lemma 13.** *If (ρ, β) is a problematic pair, then ρ is a strict lower bound on the competitive ratio of incremental solutions for β-decreasing instances of* REGION CHOOSING.

All that remains is to specify a problematic pair in order to obtain a lower bound via Lemma 13. It is easy to verify that $(2.18, 0.86)$ is a problematic pair. Note that the resulting bound of 2.18 can slightly be increased to larger values below 2.19.

▶ **Theorem 14.** *There is no 2.18-competitive incremental* REGION CHOOSING *algorithm.*

4   The greedy algorithm for a subclass of incremental problems

In this section, we analyze the greedy algorithm that computes an incremental solution $\vec{S}$ with $\vec{S}_k = \vec{S}_{k-1} \cup \{s_k\}$, where $s_k \in \arg\max_{s \in U \setminus \vec{S}_{k-1}} f(\vec{S}_{k-1} \cup \{s_k\})$ and $\vec{S}_0 = \emptyset$. This algorithm is well-known to have competitive ratio $\frac{e}{e-1} \approx 1.58$ if the objective function f is monotone and submodular [27]. Note that every monotone and submodular function is incremental. On the other hand, in general, the greedy algorithm does not have a bounded competitive ratio for incremental problems.

▶ **Observation 15.** *The greedy algorithm has an unbounded competitive ratio for many incremental problems, e.g.,* Knapsack*,* Weighted Independent Set*, and* Disjoint Paths*.*

We will now define a subclass of incremental problems where the competitive ratio of greedy can be bounded. Observe that submodularity of a function $f \colon 2^U \to \mathbb{R}_{\geq 0}$ implies that, for every $S \neq T \subseteq U$, there exists an element $t \in T \setminus S$ with $f(S \cup \{t\}) - f(S) \geq (f(S \cup T) - f(S))/|T \setminus S|$. Accordingly, we can define the following relaxation of submodularity.

▶ **Definition 16.** *We say that $f \colon 2^U \to \mathbb{R}_{\geq 0}$ is α-augmentable for an $\alpha > 0$, if for every $S, T \subseteq U$ with $T \setminus S \neq \emptyset$ there exists an element $t \in T \setminus S$ with*

$$f(S \cup \{t\}) - f(S) \geq \frac{f(S \cup T) - \alpha f(S)}{|T|}. \tag{3}$$

Thus, if f is α-augmentable, we can improve a greedy solution if its value is more than a factor of α away from the value of an optimal solution. This definition is meaningful in the sense that it induces an interesting subclass of incremental problems.

▶ **Lemma 17.** *The objective functions of* Maximum (Weighted) (b-)Matching *and* Maximum Bridge-Flow *are 2-augmentable, but not submodular.*

We now show the first part of Theorem 4.

▶ **Theorem 18.** *If the objective function of an incremental problem is α-augmentable, the greedy algorithm is $\alpha\frac{e^\alpha}{e^\alpha-1}$-competitive.*

Proof. Let $\vec{S}_i$ be our greedy incremental solution after i elements have been added. Let us focus on an arbitrary cardinality $k > \alpha$, and say that for this cardinality we have $f_k^\star = \alpha f(\vec{S}_k) + \beta$, for some $\beta > 0$. For cardinalities $k \leq \alpha$, note that $f_1^\star = f(\vec{S}_1)$ and that $f_k^\star \leq k f_1^\star \leq \alpha f_1^\star \leq \alpha f(\vec{S}_k)$ due to sub-additivity. We will show that we must have $\beta \leq \alpha f(\vec{S}_k)/(e^\alpha - 1)$, which proves the theorem for cardinalities $k > \alpha$.

First, let us define $p_k := f(\vec{S}_k) - f(\vec{S}_{k-1})$ to be the additional value obtained by adding the k-th element to our greedy solution $\vec{S}_k$. We claim that

$$p_i \geq \frac{\beta + \alpha \sum_{j=i+1}^k p_j}{k - \alpha} \quad \text{for any positive integer } i < k, \text{ and} \quad p_k \geq \frac{\beta}{k - \alpha}. \tag{4}$$

To prove (4), we apply (3) for $S = \vec{S}_{i-1}$ and $T = S_k^\star$, which guarantees the existence of a $t \in S_k^\star \setminus \vec{S}_{i-1}$ such that

$$f(\vec{S}_{i-1} \cup \{t\}) - f(\vec{S}_{i-1}) \geq \frac{f(\vec{S}_{i-1} \cup S_k^\star) - \alpha f(\vec{S}_{i-1})}{k} \geq \frac{f_k^\star - \alpha f(\vec{S}_{i-1})}{k} \geq \frac{\beta + \alpha \sum_{j=i}^k p_j}{k}.$$

The last inequality holds since we assume $f_k^\star = \alpha f(\vec{S}_k) + \beta$ and for any $i \leq k$ we know that $f(\vec{S}_k) - f(\vec{S}_{i-1}) = \sum_{j=i}^k p_j$. Since we construct $\vec{S}_i$ greedily, we have $f(\vec{S}_i) \geq f(\vec{S}_{i-1} \cup \{t\})$

and thus $p_i \geq (\beta + \alpha \sum_{i \leq j \leq k} p_j)/k$. Rearranging to isolate p_i we get the bounds in (4). Next, we claim that

$$p_i \geq \frac{\beta}{k - \alpha} \cdot \left(\frac{k}{k - \alpha} \right)^{k-i} \qquad \text{for all } i \leq k. \tag{5}$$

We prove this by induction for decreasing values of i. The induction base for $i = k$ follows directly from (4). For the induction step, we assume that the formula holds for all p_j with $j \in \{i+1, \ldots, k\}$. By (4) and the inductive hypothesis, we have

$$(k - \alpha) \cdot p_i \geq \beta + \sum_{j=i+1}^{k} \alpha p_j \geq \beta + \alpha \sum_{j=i+1}^{k} \frac{\beta}{k - \alpha} \cdot \left(\frac{k}{k - \alpha} \right)^{k-j}$$

$$= \beta + \frac{\alpha \beta}{k - \alpha} \sum_{j=0}^{k-i-1} \left(\frac{k}{k - \alpha} \right)^j = \beta + \frac{\alpha \beta}{k - \alpha} \cdot \frac{(\frac{k}{k-\alpha})^{k-i} - 1}{\frac{k}{k-\alpha} - 1} = \beta \cdot \left(\frac{k}{k - \alpha} \right)^{k-i}, \tag{6}$$

which shows that the formula also holds for p_i, and thus completes the induction step.

We are now ready to prove the theorem. Recall that we assumed $f_k^\star = \alpha f(\vec{S}_k) + \beta$ and want to show that $\beta < \alpha f(\vec{S}_k)/(e^\alpha - 1)$. We have

$$f(\vec{S}_k) = \sum_{i=1}^{k} p_i \overset{(6)}{\geq} \frac{\beta}{k - \alpha} \sum_{i=1}^{k} \left(\frac{k}{k - \alpha} \right)^{k-i} = \frac{\beta}{k - \alpha} \sum_{i=0}^{k-1} \left(\frac{k}{k - \alpha} \right)^i = \frac{\beta}{k - \alpha} \frac{(\frac{k}{k-\alpha})^k - 1}{\frac{k}{k-\alpha} - 1}$$

which can be rearranged to

$$\beta \leq \frac{(k - \alpha)\left(\frac{k}{k-\alpha} - 1 \right) f(\vec{S}_k)}{(\frac{k}{k-\alpha})^k - 1} = \frac{\alpha f(\vec{S}_k)}{(\frac{k}{k-\alpha})^k - 1} \leq \frac{\alpha f(\vec{S}_k)}{e^\alpha - 1}$$

The last inequality holds since for any $x > 0$ we have $(1+1/x)^{x+1} \geq e$, and thus for $x = k/\alpha - 1$ it follows that $e \leq \left(1 + \frac{1}{\frac{k}{\alpha} - 1} \right)^{\frac{k}{\alpha}} = \left(1 + \frac{\alpha}{k - \alpha} \right)^{\frac{k}{\alpha}} = \left(\frac{k}{k-\alpha} \right)^{\frac{k}{\alpha}}$ and thus $(\frac{k}{k-\alpha})^k \geq e^\alpha$. ◀

4.1 Lower bound

We now show the second part of Theorem 4, i.e., we show a matching lower bound on the competitive ratio of the greedy algorithm. We show this by constructing the following family of instances for the Maximum Bridge-Flow problem. For $k \in \mathbb{N}$, we define a graph $G_k = (V_k, E_k)$ with designated nodes s and t by

$$V_k := \{s, t\} \cup \{v_i^1, v_i^4 \mid i = 1, \ldots, 2k\} \cup \{v_i^2, v_i^3 \mid i = 1, \ldots, 4k\},$$

$$E_k := E_k^1 \cup E_k^\infty \cup \bigcup_{i=1}^{2k} E_{k,i} \cup \bigcup_{i=1}^{2k} E_{k,i}',$$

$$E_k^1 := \{(s, v_i^2), (v_i^3, t) \mid i = 1, \ldots, k\},$$

$$E_k^\infty := \{(s, v_{3k+i}^2), (v_i^2, v_i^3), (v_{3k+i}^2, v_{3k+i}^3), (v_i^3, t) \mid i = 1, \ldots, k\},$$

$$E_{k,i} := \{(s, v_i^1), (v_i^1, v_{k+i}^2), (v_{k+i}^2, v_{k+i}^3), (v_{k+i}^3, v_i^4), (v_i^4, t)\} \quad \text{for all } i = 1, \ldots, 2k,$$

$$E_{k,i}' := \{(v_i^1, v_j^2), (v_{3k+j}^3, v_i^4) \mid j = 1, \ldots, k\} \quad \text{for all } i = 1, \ldots, 2k.$$

The edge capacities $u_k \colon E_k \to \mathbb{R}_{\geq 0}$ are given by $u_k(e) = (\frac{k}{k-1})^{2k+1-i}$ for $e \in E_{k,i}$, by $u_k(e) = \frac{1}{k}(\frac{k}{k-1})^{2k+1-i}$ for $e \in E_{k,i}'$, by $u_k(e) = 1$ for $e \in E_k^1$, and by $u_k(e) = \infty$ for $e \in E_k^\infty$.

For every G_k, we choose a directed s-t-cut $C_k := \left\{ (v_i^2, v_i^3) \mid i = 1, \ldots, 4k \right\}$. Without loss of generality, we will assume in the following that we can resolve all ties in the greedy algorithm to our preference. This can be done formally by adding some very small offsets to the edge weights, but we omit this for clarity. Now consider how the greedy algorithm operates on graph G_k.

▶ **Lemma 19.** *In step $j \in \{1, \ldots, 2k\}$, the greedy algorithm picks edge (v_{k+j}^2, v_{k+j}^3).*

With this, we are ready to show the following result, which, together with Lemma 17, implies the second part of Theorem 4.

▶ **Theorem 20.** *The greedy algorithm has competitive ratio at least $\frac{2e^2}{e^2-1} \approx 2.313$ for* MAXIMUM BRIDGE-FLOW.

Proof. By Lemma 19, the greedy algorithm picks the edges $(v_{k+1}^2, v_{k+1}^3), \ldots, (v_{3k}^2, v_{3k}^3)$ in the first $2k$ steps. Thus, after step $2k$, greedy can send an s-t-flow of value

$$\sum_{i=1}^{2k} \left(\frac{k}{k-1} \right)^i = \left(\frac{\left(\frac{k}{k-1} \right)^{2k+1} - 1}{\frac{k}{k-1} - 1} - 1 \right) = (k-1) \left(\frac{k}{k-1} \right)^{2k+1} - k .$$

On the other hand, the solution of size $2k$ consisting of the edges $(v_1^2, v_1^3), \ldots, (v_k^2, v_k^3)$, and $(v_{3k+1}^2, v_{3k+1}^3), \ldots, (v_{4k}^2, v_{4k}^3)$ results in an (optimal) flow value of

$$2k + 2 \sum_{i=1}^{2k} \left(\frac{k}{k-1} \right)^i = 2(k-1) \left(\frac{k}{k-1} \right)^{2k+1} .$$

This corresponds to a competitive ratio of

$$\frac{2(k-1) \left(\frac{k}{k-1} \right)^{2k+1}}{(k-1) \left(\frac{k}{k-1} \right)^{2k+1} - k} = \frac{2 \left(\frac{k}{k-1} \right)^{2k}}{\left(\frac{k}{k-1} \right)^{2k} - 1} = \frac{2 \left(\frac{k}{k-1} \right)^{2(k-1)+2}}{\left(\frac{k}{k-1} \right)^{2(k-1)+2} - 1} .$$

Substituting $x := k-1$ and using the identity $\lim_{x \to \infty} (1 + 1/x)^x = e$, we get the lower bound on the competitive ratio of the greedy algorithm claimed in Theorem 4 in the limit:

$$\lim_{x \to \infty} \frac{2 \left(\frac{x+1}{x} \right)^{2x} \left(\frac{x+1}{x} \right)^2}{2 \left(\frac{x+1}{x} \right)^{2x} \left(\frac{x+1}{x} \right)^2 - 1} = \lim_{x \to \infty} \frac{2e^2 \left(\frac{x+1}{x} \right)^2}{2e^2 \left(\frac{x+1}{x} \right)^2 - 1} = \frac{2e^2}{e^2 - 1} . \qquad ◀$$

5 Conclusion

We have defined a formal framework that captures a large class of incremental problems and allows for incremental solutions with bounded competitive ratio. We also defined a new and meaningful subclass consisting of problems with α-augmentable objective functions for which the greedy algorithm has a bounded competitive ratio. Hopefully our results can inspire future work on incremental problems from a perspective of competitive analysis.

Obvious extensions of our results would be to close the gap between our bounds of 2.618 and 2.18 for the best-possible competitive ratio of incremental algorithms. In particular, it would be interesting whether or not the bound of 2.313 for the greedy algorithm in the 2-augmentable setting can be beaten by some other incremental algorithm already in the general setting. Also, the α-augmentable and strictly submodular settings may allow for

better incremental algorithms than the greedy algorithm. Another question is whether abandoning the accountability condition yields an interesting class of problems. Finally, it may be possible to generalize our framework to problems with a continuously growing budget and a cost associated with each element, instead of a growing cardinality constraint.

References

1 Avrim Blum, Prasad Chalasani, Don Coppersmith, Bill Pulleyblank, Prabhakar Raghavan, and Madhu Sudan. The minimum latency problem. In *Proceedings of the 26th Annual ACM Symposium on Theory of computing (STOC)*, pages 163–171, 1994.

2 A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, Cambridge, UK, 1998.

3 Moses Charikar, Chandra Chekuri, Tomas Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. *SIAM Journal on Computing*, 33(6):1417–1440, 2004.

4 Marek Chrobak, Claire Kenyon, John Noga, and Neal E. Young. Incremental medians via online bidding. *Algorithmica*, 50(4):455–478, 2007.

5 Varsha Dani and Thomas P. Hayes. Robbing the bandit: less regret in online geometric optimization against an adaptive adversary. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 937–943, 2006.

6 Sanjoy Dasgupta and Philip M. Long. Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, 70(4):555–569, 2005.

7 Yann Disser, Max Klimm, Nicole Megow, and Sebastian Stiller. Packing a Knapsack of Unknown Capacity. *SIAM Journal on Discrete Mathematics*, to appear.

8 U. Faigle, W. Kern, and G. Turan. On the performance of on-line algorithms for partition problems. *Acta Cybernetica*, 9(2):107–119, 1989.

9 A. Fiat and G. J. Woeginger, editors. *Online Algorithms: The State of the Art*. Springer, Berlin, 1998.

10 Dimitris Fotakis. Incremental algorithms for facility location and k-median. *Theoretical Computer Science*, 361(2–3):275–313, 2006. `doi:10.1016/j.tcs.2006.05.015`.

11 Ryo Fujita, Yusuke Kobayashi, and Kazuhisa Makino. Robust matchings and matroid intersections. *SIAM Journal on Discrete Mathematics*, 27(3):1234–1256, 2013. `doi:10.1137/100808800`.

12 Michel Goemans and Jon Kleinberg. An improved approximation ratio for the minimum latency problem. *Mathematical Programming*, 82(1–2):111–124, 1998.

13 C. Greg Plaxton. Approximation algorithms for hierarchical location problems. *Journal of Computer and System Sciences*, 72(3):425–443, 2006.

14 Magnús M. Halldórsson and Hadas Shachnai. Return of the boss problem: Competing online against a non-adaptive adversary. In *Proceedings of the 5th International Conference on Fun with Algorithms (FUN)*, pages 237–248, 2010.

15 Jeff Hartline and Alexa Sharp. Incremental flow. *Networks*, 50(1):77–85, 2007.

16 Refael Hassin and Shlomi Rubinstein. Robust matchings. *SIAM Journal on Discrete Mathematics*, 15(4):530–537, 2002. `doi:10.1137/S0895480198332156`.

17 Oscar H. Ibarra and Chul E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM*, 22(4):463–468, 1975.

18 Naonori Kakimura and Kazuhisa Makino. Robust independence systems. *SIAM Journal on Discrete Mathematics*, 27(3):1257–1273, 2013. `doi:10.1137/120899480`.

19 Naonori Kakimura, Kazuhisa Makino, and Kento Seimi. Computing knapsack solutions with cardinality robustness. *Japan Journal of Industrial and Applied Mathematics*, 29(3):469–483, 2012. `doi:10.1007/s13160-012-0075-z`.

20 Yusuke Kobayashi and Kenjiro Takazawa. Randomized strategies for cardinality robustness in the knapsack problem. *Theoretical Computer Science*, pages 1–10, 2016. `doi:10.1016/j.tcs.2016.12.019`.

21 A. Krause, H. B. McMahan, C. Guestrin, and A. Gupta. Robust submodular observation selection. *Journal of Machine Learning Research*, 9:2761–2801, 2008.

22 Eugene L. Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4(4):339–356, 1979.

23 Guolong Lin, Chandrashekhar Nagarajan, Rajmohan Rajaraman, and David P. Williamson. A general approach for incremental approximation and hierarchical clustering. *SIAM Journal on Computing*, 39(8):3633–3669, 2010. `doi:10.1137/070698257`.

24 Jannik Matuschke, Martin Skutella, and José A. Soto. Robust randomized matchings. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1904–1915, 2014.

25 Nicole Megow and Julian Mestre. Instance-sensitive robustness guarantees for sequencing with unknown packing and covering constraints. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science (ITCS)*, pages 495–504, 2013.

26 Ramgopal R. Mettu and C. Greg Plaxton. The online median problem. *SIAM Journal on Computing*, 32(3):816–832, 2003. `doi:10.1137/S0097539701383443`.

27 G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions – I. *Mathematical Programming*, 14(1):265–294, 1978.