The Power of Local Search: Maximum Coverage over a Matroid

Yuval Filmus[1,2] **and Justin Ward**[1]

1 Department of Computer Science, University of Toronto
`{yuvalf,jward}@cs.toronto.edu`

2 Supported by NSERC

Abstract

We present an optimal, combinatorial $1 - 1/e$ approximation algorithm for Maximum Coverage over a matroid constraint, using non-oblivious local search. Calinescu, Chekuri, Pál and Vondrák have given an optimal $1 - 1/e$ approximation algorithm for the more general problem of monotone submodular maximization over a matroid constraint. The advantage of our algorithm is that it is entirely combinatorial, and in many circumstances also faster, as well as conceptually simpler.

Following previous work on satisfiability problems by Alimonti, as well as by Khanna, Motwani, Sudan and Vazirani, our local search algorithm is *non-oblivious*. That is, our algorithm uses an auxiliary linear objective function to evaluate solutions. This function gives more weight to elements covered multiple times. We show that the locality ratio of the resulting local search procedure is at least $1 - 1/e$. Our local search procedure only considers improvements of size 1. In contrast, we show that oblivious local search, guided only by the problem's objective function, achieves an approximation ratio of only $(n-1)/(2n-1-k)$ when improvements of size k are considered.

In general, our local search algorithm could take an exponential amount of time to converge to an *exact* local optimum. We address this situation by using a combination of *approximate* local search and the same partial enumeration techniques as Calinescu et al., resulting in a clean $(1 - 1/e)$-approximation algorithm running in polynomial time.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases approximation algorithms; maximum coverage; matroids; local search

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Maximum coverage [1,6–9,11,13,16] (also known as Max Cover) is a well-known combinatorial optimization problem related to Set Cover. Given a universe U, element weights $w\colon U \to \mathbb{R}_+$, a family $\mathcal{F} \subset 2^{2^U}$ of subsets of U, and a number n, the problem is to select n sets $S_i \in \mathcal{F}$ such that $w(S_1 \cup \cdots \cup S_n)$ is as large as possible.

Like many combinatorial optimization problems, maximum coverage is hard to solve exactly. A straightforward reduction from Set Cover shows that the decision version of maximum coverage with unit weights (deciding whether there are n sets that span at least m elements) is NP-complete, so the best we can hope for is an approximation algorithm.

One natural approach is the following heuristic. First pick the set S_1 of maximum weight. Then pick the set S_2 that maximizes $w(S_2 \setminus S_1)$, and so on. This approach leads to the well-known *greedy algorithm*, and yields an approximation ratio of $1 - 1/e \approx 0.632+$. Amazingly, in this setting the greedy algorithm is optimal. Feige [9] used the PCP theorem to

STACS 2012.
Editors: Christoph Dürr, Thomas Wilke; pp. 1–16
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

show that if there exists a polynomial-time algorithm that approximates maximum coverage within a ratio of $1 - 1/e + \epsilon$ for some $\epsilon > 0$ then $\mathsf{P} = \mathsf{NP}$.

The present paper deals with a generalized version of maximum coverage, in which we are given a matroid $\mathfrak{m}$ over $\mathcal{F}$, and the goal is to find a collection $S \subseteq \mathcal{F}$ that covers elements of maximum weight, subject to the constraint that $S \in \mathfrak{m}$. We call this problem *maximum coverage over a matroid constraint.* If the underlying matroid is the uniform matroid of rank n, we recover the original problem. Because every maximum coverage function is monotone submodular, this problem falls under the more general setting of maximizing monotone submodular functions subject to a matroid constraint. The greedy algorithm still applies in this general setting, but its approximation ratio is only $1/2$, even when the monotone submodular function is a maximum coverage function. Calinescu, Chekuri, Pál and Vondrák [5] developed a sophisticated algorithm for a general problem of monotone submodular maximization over a matroid constraint, which achieves the optimal approximation ratio of $1 - 1/e$. Their algorithm first finds a fractional solution using the *continuous greedy* algorithm, and then rounds it to an integral solution using *pipage rounding*.

1.1 Our contribution

We propose a simple algorithm for maximum coverage over a matroid constraint. Like the continuous greedy algorithm, our algorithm achieves the optimal approximation ratio of $1 - 1/e$. In contrast to the continuous greedy algorithm, however, our algorithm is *combinatorial*, in the sense that it only deals with integral solutions. Our approach is based on *non-oblivious local search*, a technique first proposed by Alimonti [2] and by Khanna, Motwani, Sudan and Vazirani [12].

In classical (or, *oblivious*) local search, the algorithm starts at an arbitrary solution, and proceeds by iteratively making small changes that improve the objective function, until no such improvement can be made. The *locality ratio* of a local search algorithm is $\min w(S)/w(O)$, where S is a solution that is locally-optimal with respect to the small changes considered by the algorithm, O is a global optimum, and w is the objective function for the problem. The locality ratio provides a natural, worst-case guarantee on the approximation performance of the local search algorithm.

In many cases, oblivious local search may have a very poor locality ratio, implying that a locally-optimal solution may be of significantly lower quality than the global optimum. For example, for our problem the locality ratio for an algorithm changing a single set at each step is $1/2$. As the locality ratio depends on the type and size of local changes considered, one approach for improving an algorithm's performance is simply to consider larger (but still constant-sized) neighborhoods. Unfortunately, in our case, this technique results in no asymptotic improvement in the locality ratio. Specifically, we show that the locality ratio of oblivious local search remains only $(n-1)/(2n-1-k)$ when k sets are exchanged. For constant k, this is only marginally better than the approximation ratio $1/2$ achievable using the greedy algorithm.

Non-oblivious local search adopts a more radical approach by altering the objective function used to guide the search. We proceed as before, but modify the local search algorithm to use an auxiliary objective function to decide whether the current solution is an improvement. By carefully choosing this auxiliary function, we can ensure that poor local optima with respect to the original objective function are no longer local optima.

In this paper, we present a non-oblivious local search algorithm for the problem of maximum coverage over a matroid constraint. Specifically, we construct an auxiliary objective function whose locality ratio (for single changes) is slightly larger than $1 - 1/e$, resulting in an algorithm whose approximation ratio is the best possible, assuming $\mathsf{P} \neq \mathsf{NP}$.

In general, local search algorithms could converge in exponentially many steps. We address this issue using approximate local search, a technique described systematically by Schulz, Orlin and Punnen [15]. Approximate local search allows us to bound the running time of the algorithm at a small cost in the resulting approximation ratio. By employing partial enumeration, as described by Calinescu et al., we can eliminate this small cost, achieving an approximation ratio of $1 - 1/e$.

1.2 Comparison with existing algorithms

Both our algorithm and the one by Calinescu et al. give the same approximation ratio. The continuous greedy algorithm works on a discretized version of a particular continuous relaxation of the problem to obtain a fractional solution to the problem. This solution must then be rounded to an integral solution using the pipage rounding technique, which employs a submodular minimization algorithm at each step. In contrast, our algorithm always maintains a current integral solution and requires only simple combinatorial operations that add and remove elements from this solution. Moreover, our algorithm is extremely simple, and can be described using only a few lines of pseudocode.

In most settings, our algorithm is also faster. For a fair comparison, we consider versions of both algorithms that achieve an approximation ratio of at least $1 - 1/e$. Furthermore, we analyze the algorithm of Calinescu et al. only in the special case of maximum coverage. In this setting, it is possible to calculate the continuous relaxation of the objective function exactly, rather than by random sampling, thus greatly improving the runtime of the continuous greedy algorithm. Denoting the rank of the matroid by n, the total number of sets by $s = |\mathcal{F}|$, the maximal size of a set by u, and the sum of the sizes of all sets by U, our algorithm runs in time $\tilde{O}(n^3 s^2 u)$, whereas the algorithm by Calinescu et al. runs in time $\tilde{O}(n^2 s^3 u + s^7)$. For all non-trivial instances of the problem, we must have $s > n$, and so our algorithm is indeed considerably faster, assuming that the size u of the largest set does not dominate both expressions. We stress that this is the case even after the continuous greedy algorithm has been optimized to compute the maximum coverage function directly, rather than by sampling as in the general analysis of Calinescu et al. [5].

Another relevant algorithm is described in an earlier paper by the same set of authors [4]. The algorithm presented by Calinescu et al. in that paper is more general than ours, but less general than their later paper. It applies to monotone submodular functions which are sums of weighted rank functions of matroids. The continuous greedy algorithm is replaced with a simple linear program. In the simplest case of a uniform matroid, the running time of this algorithm (using interior-point methods) is $\tilde{O}(U^{3.5} + s^7)$; more complicated matroids result in even worse running times. When the sets are large, this is considerably slower than our algorithm.

1.3 Future work

We believe that the approach outlined in this paper can be used to design approximation algorithms for similar problems. In particular, we are working on generalizing the algorithm to monotone submodular functions.

1.4 Paper organization

Section 2 provides a concise overview of existing work related to maximum coverage. In Section 3 we formally present the problem, discuss the limitations of oblivious local search, and present our non-oblivious local search algorithm. In Section 4 we give an analysis of the approximation ratio and runtime for our non-oblivious local search algorithm, and show how to attain a clean, polynomial time $(1 - 1/e)$-approximation by using partial enumeration techniques, as in [5].

For reasons of space, we defer some proofs to the appendix. In the appendix we also discuss the optimality of our auxiliary objective function, and analyze the standard oblivious local search in the special case of strongly base orderable matroids, showing that its performance can be much worse than our non-oblivious local search algorithm. Finally, we show that the difficulty of finding an exact local optimum is not an artifact of our oblivious non-oblivious objective function by presenting an instance for which even the oblivious local search algorithm requires an exponential number of improvements to find an exact local optimum.

1.5 Acknowledgments

We thank Anupam Gupta for suggesting the problem to us, and Allan Borodin and Roy Schwartz for helpful discussions.

2 Related work

Maximum coverage was first defined by Hochbaum and Pathria [11] in 1998. In fact, an even more general problem had been defined earlier by Cornuejols, Fisher and Nemhauser [8] in 1977, in the context of locating bank accounts. Both papers describe the greedy algorithm, and show that its approximation ratio is $1 - (1 - 1/n)^n \approx 1 - 1/e$.

Feige [9], in his seminal paper on the inapproximability of Set Cover, showed that unless $\mathsf{P} = \mathsf{NP}$, it is impossible to approximate maximum coverage to within $1 - 1/e + \epsilon$ for any $\epsilon > 0$. His proof uses PCP techniques, extending earlier work by Lund and Yannakakis [14].

Maximum coverage over a partition matroid was considered by Chekuri and Kumar [6] under the name *maximum coverage with group budget constraints*. In this variant, the family $\mathcal{F}$ is partitioned into subfamilies $\mathcal{F}_i$, and the solution must contain at most n_i sets from $\mathcal{F}_i$, and at most n overall. They analyze the performance of the greedy algorithm when the greedy choices are given by an approximate oracle.

Algorithms for maximum coverage with group budget constraints with the optimal approximation ratio $1-1/e$ were presented by Srinivasan [16] and by Ageev and Sviridenko [1]. Both algorithms first formulate the problem as an LP, and then round the solution. Srinivasan's approach involves sophisticated sampling. Ageev and Sviridenko's approach, *pipage rounding*, repeatedly simplifies the solution until it becomes integral, without decreasing its value. Both approaches work in more general settings.

Calinescu, Chekuri, Pál and Vondrák [5] combined a more sophisticated version of pipage rounding with the *continuous greedy* algorithm to obtain an optimal $1 - 1/e$ approximation algorithm for monotone submodular maximization over a matroid constraint. The continuous greedy algorithm is a steepest descent algorithm running in continuous time (in practice, a suitably discretized version is used), producing a fractional solution. This solution is rounded using pipage rounding.

Other generalizations of maximum coverage appear in the literature. In *budgeted maximum coverage*, each element is provided with a cost, and the sets chosen by the algorithm are restricted by the total cost of the elements they cover. Khuller, Moss and Naor [13] show that a greedy approach yields an optimal $1 - 1/e$ approximation algorithm. Cohen and Katzir [7] generalize this even further, and provide an optimal $1 - 1/e$ semi-greedy approximation algorithm.

3 Local search for maximum coverage over a matroid

We consider the problem of maximum coverage over a matroid. The inputs are

- A universe U.
- Value oracle access to a non-negative weight function $w\colon U \to \mathbb{R}_+$.
- A family $\mathcal{F}$ of subsets of U with $|\mathcal{F}| = s$, $\max_{A \in \mathcal{F}} |A| = u$.
- A matroid over $\mathcal{F}$ of rank n, given as an independence oracle.

Note that the matroid $\mathfrak{m}$ has as its ground set the *sets* $\mathcal{F}$, and so a member of $\mathfrak{m}$ is a collection of sets from $\mathcal{F}$. We call the members of $\mathfrak{m}$ *independent sets*. We extend w to a function over subsets A of U by letting $w(A) = \sum_{u \in A} w(u)$. The goal, then, is to find a collection of sets $S \subseteq \mathcal{F}$ that covers elements in U of maximal weight, subject to the constraint that $S \in \mathfrak{m}$:

$$\max_{S \in \mathfrak{m}} w\Big(\bigcup S\Big).$$

Recall that $\mathfrak{m}$ is a matroid over $\mathcal{F}$ if: (1) $\mathfrak{m} \neq \emptyset$, (2) $\mathfrak{m}$ is downward-closed (if $A \in \mathfrak{m}$ and $B \subset A$, then $B \in \mathfrak{m}$), and (3) for all $A, B \in \mathfrak{m}$ with $|A| > |B|$ we have $B \cup \{x\} \in \mathfrak{m}$ for some $x \in A \setminus B$. This last property guarantees that all maximal independent sets of the matroid have the same cardinality. These sets are called *bases*, and their common cardinality is called the *rank*, denoted in this paper by n.

Our starting point is the oblivious local search algorithm, shown in Algorithm 1. The algorithm starts from an arbitrary base S (obtained, e.g., by the standard greedy algorithm) and repeatedly attempts to improve the total weight of all elements covered by S by exchanging up to k sets in S with k sets not in S, maintaining the matroid constraint. We call a pair (A, B) a k-*exchange* for S if $A \subseteq S$ with $|A| \leq k$, $B \subseteq \mathcal{F} \setminus S$ with $|B| = |A|$. When no single k-exchange improves the weight of all elements covered by S, the algorithm returns S.

Algorithm 1 Oblivious k-LocalSearch

```
$S \leftarrow$ an arbitrary basis in $\mathfrak{m}$.
repeat
    $S_{\text{old}} \leftarrow S$ {Remember previous solution}
    Let $\mathcal{E}$ be the set of all valid $k$-exchanges for $S$
    $S \leftarrow \arg\max_{(A,B) \in \mathcal{E}} w\left(\bigcup (S \setminus A \cup B)\right)$
until $S = S_{\text{old}}$ {Repeat until no improvement is possible}
return $S$
```

As mentioned in the introduction, the locality ratio of Algorithm 1 is rather poor. Consider the following set system:

$$A_1 = \{x, \epsilon_A\}, \qquad A_2 = \{\epsilon_B\},$$
$$B_1 = \{y\}, \qquad B_2 = \{x\}.$$

The elements x, y have unit weight, and the elements ϵ_A, ϵ_B have some arbitrarily small weight $\epsilon > 0$. We consider the partition matroid whose independents sets contain at most one of $\{A_i, B_i\}$ for $i \in \{1, 2\}$. Under this matroid, the base $\{A_1, A_2\}$ is a local optimum for 1-exchanges, since replacing A_1 by B_1 or A_2 by B_2 both result in a net loss of ϵ. The global optimum is $\{B_1, B_2\}$, and the locality ratio is thus $(1 + 2\epsilon)/2$. Note that the base $\{A_1, A_2\}$ is also produced by the standard greedy algorithm applied to this instance. This shows that we cannot hope to beat the locality ratio by choosing a greedy starting solution.

We can generalize this example to show that oblivious k-local search has a locality ratio of at most $\frac{n-1}{2n-k-1}$ for all $k < n$. Let the universe U consist of $n-1$ elements $\{x_1, \dots, x_{n-1}\}$ and $n - k$ elements $\{y_1, \dots, y_{n-k}\}$, all of weight 1, and $n - 1$ elements $\{\epsilon_2, \dots, \epsilon_n\}$ of arbitrarily small weight $\epsilon > 0$. For each $1 \le i \le n$, there are two sets A_i and B_i, defined as follows:

$$A_i = \{\epsilon_i\} \text{ for } 1 \le i \le n-1, \qquad A_n = \{x_1, \dots, x_{n-1}\},$$
$$B_i = \{x_i\} \text{ for } 1 \le i \le n-1, \qquad B_n = \{y_1, \dots, y_{n-k}\}.$$

We consider the partition matroid whose independent sets contain at most one of $\{A_i, B_i\}$ for each $i \in [n]$. The globally optimal solution is the set $B = \{B_i\}_{1 \le i \le n}$, which covers elements of total weight $n - 1 + n - k = 2n - k - 1$. The set $A = \{A_i\}_{1 \le i \le n}$ is locally optimal under improvements of size k and covers elements of total weight only $(n - 1)(1 + \epsilon)$, and the locality ratio is thus $(n - 1)(1 + \epsilon)/(2n - k - 1)$. In order to see that it is, in fact, a local optimum, note that if we do not replace A_n with B_n, the remaining replacements can only decrease the value of the solution. Suppose, then, that we do replace A_n with B_n. This replacement reduces the total weight of the covered elements by $k - 1$. There are only $k - 1$ remaining exchanges, each of which can increase the total weight of the covered elements by less than 1. Again, we note that the solution A is also the solution produced by the greedy algorithm. In the special case of *strongly base orderable* matroids, the approximation ratio of oblivious k-local search is, in fact, exactly $\frac{n-1}{2n-k-1}$, as we show in Section A.3.

Let us examine the first instance above in more detail. Intuitively, the basis $\{A_1, B_2\}$ in our example is better than the basis $\{A_1, A_2\}$ since it is of almost equal value to $\{A_1, A_2\}$ but additionally covers element x twice. From a local search perspective, this is an advantage since it ensures that x will stay covered after the next exchange. In order to improve the performance of local search, we want to somehow give extra weight to solutions that offer flexibility in future exchanges, perhaps even at a slight loss in the objective function. Following this intuition, we employ a function which gives extra weight to elements that are covered multiple times as an auxiliary objective function. A similar idea appears in Khanna et al. [12], in the context of the maximum satisfiability problem, and even earlier in similar work by Alimonti [2]. There, the idea is to give extra weight to clauses which are satisfied by more than one variable, because these clauses will remain satisfied after flipping the next variable in the search procedure.

Thus, we seek to modify Algorithm 1 by replacing the oblivious objective function w with an auxiliary objective function f of the general form:

$$f(S) = \sum_{u \in U} \alpha_{h_u(S)} w(u),$$

where $h_u(S) = |\{A \in S : u \in A\}|$ is the number of sets in S that contain element u. By setting, for example $\alpha_i > \alpha_1$ for all $i > 1$, we can give extra value to solutions that cover some element more than once. Additionally, note that if we set $\alpha_0 = 0$ and $\alpha_i = 1$ for all $i > 0$, we recover the oblivious objective function. We now consider the problem of how to set the α_i. We want to balance two concerns. First, we want to allow the algorithm to potentially decrease the objective value of the current solution in the short-term, in exchange for future flexibility. However, in the long-term, we want the algorithm to give enough weight to the original objective that it produces a reasonable final solution.

There is no immediately compelling reason to assign any weight to elements that have not been covered at all, so let us set $\alpha_0 = 0$, and examine the above instance in terms of the remaining α_i. We have

$$f(\{A_1, A_2\}) = (1 + 2\epsilon)\alpha_1, \qquad f(\{A_1, B_2\}) = \alpha_2 + \epsilon\alpha_1,$$
$$f(\{B_1, A_2\}) = (1 + \epsilon)\alpha_1, \qquad f(\{B_1, B_2\}) = 2\alpha_1.$$

By setting $\epsilon = \frac{\alpha_2-\alpha_1}{\alpha_1}$, the solution $\{A_1, A_2\}$ will remain a local optimum, even for the non-oblivious potential function. The locality ratio (in terms of the *original* objective function) will remain $1/2 + \epsilon = 1/2 + \frac{\alpha_2-\alpha_1}{\alpha_1}$. If we set α_2 to be too close to α_1, there will not be much improvement in the locality ratio, and if $\alpha_2 < \alpha_1$, the locality ratio will *decrease*. This confirms our intuition that it is advantageous to give more weight to elements that are covered multiple times. Alternatively, if we set $\alpha_2 \geq 2\alpha_1$, then the solution $\{A_1, B_2\}$ will become a local optimum, and the locality ratio will become $1/2 + \epsilon/2$. This confirms our intuition that it is bad to give *too much* extra weight to elements covered multiple times.

By extending a similar analysis to arbitrary instances, we obtain the following values for α_i:

$$\alpha_0 = 0, \qquad \alpha_1 = 1 - \frac{1}{e^{[n]}}, \qquad \alpha_{i+1} = (i+1)\alpha_i - i\alpha_{i-1} - \frac{1}{e^{[n]}}, \tag{1}$$

where the constant $e^{[n]}$ is defined as

$$e^{[n]} = \sum_{l=0}^{n-1} \frac{1}{l!} + \frac{1}{(n-1)!(n-1)}.$$

This choice of α_i is optimal, as we show in Section A.2. We note that the sequence $e^{[i]}$, where $i \geq 2$, is decreasing and bounded below by e:

▶ **Lemma 1.** *For all $n \geq 2$ we have $e < e^{[n]}$ and $e^{[n]} > e^{[n+1]}$.*

Our coefficients α_i satisfy the following properties, which follow directly from their definition.

▶ **Lemma 2.** *Let $\beta_i = \alpha_{i+1} - \alpha_i$. Then, the β_i satisfy the recurrence relation*

$$\beta_0 = 1 - \frac{1}{e^{[n]}}, \quad \beta_i = i\beta_{i-1} - \frac{1}{e^{[n]}}.$$

▶ **Lemma 3.** *For all $i < n$, $\beta_i > 0$ and $\beta_{i+1} \leq \beta_i$.*

▶ **Lemma 4.** *There exists a universal constant C_0 such that for all $i \leq n$, $\alpha_i \leq C_0 \log i$.*

Our resulting local search algorithm for maximum coverage over a matroid is given in Algorithm 2. In addition to using the non-oblivious potential function f described above, we modify our algorithm to start from a greedy initial solution. This initial solution is a good starting point, and speeds up the convergence of the algorithm. Our algorithm takes as a parameter δ, which governs how much an improvement is required to improve the current solution to be accepted. We describe this aspect of the algorithm in more detail in the next section.

Algorithm 2 LocalSearch(δ)

```
{Greedy algorithm}
$S \leftarrow \varnothing$
for $i = 1 \to n$ do
    $S \leftarrow S + \underset{A \in \mathcal{F} : S+A \in \mathfrak{m}}{\arg\max} [f(S+A) - f(S)]$
end for
{Local search}
repeat
    $S_{\text{old}} = S$ {Remember previous solution}
    Let $\mathcal{E}$ be the set of all valid 1-exchanges for $S$
    $S \leftarrow \arg\max_{(A,B) \in \mathcal{E}} f((S \setminus A) \cup B)$
until $S \leq (1+\delta) S_{\text{old}}$ {Repeat until $\delta$-locally optimal}
return $S_{\text{old}}$
```

4 Analysis of the non-oblivious local search algorithm

Our analysis makes use of the notion of a δ*-approximate local optimum*. Formally, we say that a solution S is a δ-approximate local optimum if $(1+\delta)f(S) \geq f(S')$ for all solutions S' differing from S by a single set. Intuitively, replacing exact local optimality (as in Algorithm 1) with approximate local optimality limits the total number of improvements the algorithm can make, at a slight cost in the approximation factor. We consider some δ-approximate local optimum $S = \{S_1, \dots, S_n\}$ and some global optimum $O = \{O_1, \dots, O_n\}$. A classical result of Brualdi [3] shows that for any matroid $\mathfrak{m}$ we can renumber the sets of O so that for each i, $S_{-i}, O_i := (S \setminus \{S_i\}) \cup \{O_i\}$ is a base of $\mathfrak{m}$. We suppose that O has been renumbered so that this is the case, and consider the 1-exchanges that remove S_i from S and add O_i to the result. Local optimality implies that for each i, we have

$$(1+\delta)f(S) \geq f(S_{-i}, O_i).$$

Summing over all n such inequalities we obtain the inequality

$$n(1+\delta)f(S) \geq \sum_{i=1}^{n} f(S_{-i}, O_i). \tag{2}$$

The main difficulty of the analysis lies in the fact that inequality (2) is given in terms of the non-oblivious potential function f, but we wish to derive an approximation guarantee for the original objective function w. In order to put f and w on common ground, we make the following definitions.

For any two subsets $L, G \subset [n]$, we define $E_{L,G}$ to be the set of elements that belong to the sets S_i for $i \in L$, and O_j for $j \in G$, and no other sets in S and O. The sets $E_{L,G}$ thus form a partition of U. Then, for all integers $l, c, g \geq 0$ such that $1 \leq l + c + g \leq n$, we define

$$x_{l,c,g} = \sum_{\substack{L,G: \\ |L|=l+c, \\ |G|=g+c, \\ |L \cap G|=c}} w(E_{L,G}).$$

In words, $x_{l,c,g}$ is the total weight of elements that belong to exactly $l+c$ of the sets S_i, exactly $g+c$ of the sets O_j, exactly c of them sharing indices. We call the variables $x_{l,c,g}$ *symmetric variables.*

We can express all the quantities we are interested in using the symmetric variables $x_{l,g,c}$:

$$\begin{aligned} f(S) &= \sum_{l+c\geq 1} \alpha_{l+c} x_{l,c,g} = \sum_{l,c,g} \alpha_{l+c} x_{l,c,g} \text{ (since } \alpha_0 = 0\text{)}, \\ \sum_{i=1}^{n} f(S_{-i}, O_i) &= \sum_{l,c,g} (l\alpha_{l+c-1} + g\alpha_{l+c+1} + (n-l-g)\alpha_{l+c}) x_{l,c,g}, \\ w\left(\bigcup S\right) &= \sum_{l+c\geq 1} x_{l,c,g}, \\ w\left(\bigcup O\right) &= \sum_{g+c\geq 1} x_{l,c,g}. \end{aligned}$$

The only expression which is not immediate is the one for $\sum_{i=1}^{n} f(S_{-i}, O_i)$. We derive it as follows: consider an element $u \in E_{L,G}$ for some sets L, G. In S, the element u appears in $|L|$ sets. If $i \in L \cap G$ or $i \notin L \cup G$, it also appears in $|L|$ sets in (S_{-i}, O_i). Finally, if $i \in L \setminus G$, it appears in $|L| - 1$ sets in (S_{-i}, O_i). If $i \in G \setminus L$, it appears in $|L| + 1$ sets in (S_{-i}, O_i).

▶ **Theorem 5.** *If S is a δ-approximate local optimum, then $(1 + C_0\delta n \log n) w(\bigcup S) \geq (1 - 1/e^{[n]}) w(\bigcup O)$, for some universal constant C_0.*

Proof. Reformulating inequality (2) in terms of our symmetric notation and simplifying, we obtain

$$0 \leq \sum_{l,c,g} ((l + g + \delta n)\alpha_{l+c} - l\alpha_{l+c-1} - g\alpha_{l+c+1}) x_{l,c,g}. \tag{3}$$

Similarly, reformulating the statement of the theorem in terms of our symmetric notation, we obtain

$$0 \leq (1 + C_0\delta n \log n) \sum_{l+c\geq 1} x_{l,c,g} - \left(1 - \frac{1}{e^{[n]}}\right) \sum_{g+c\geq 1} x_{l,c,g}. \tag{4}$$

Since we have $x_{l,c,g} \geq 0$ for all l, c, g, in order to prove the lemma, it suffices to show that the coefficient of any term $x_{l,c,g}$ in (3) is at most its coefficient in (4). We consider three cases, and simplify expressions using the fact that $\alpha_0 = 0$. In the first, suppose that $g = c = 0$. We must show that for all $1 \leq l \leq n$,

$$(l + \delta n)\alpha_l - l\alpha_{l-1} \leq 1 + C_0\delta n \log n. \tag{5}$$

In the next case, suppose that $l = c = 0$. We must show that for all $1 \leq g \leq n$,

$$-g\alpha_1 \leq -\left(1 - \frac{1}{e^{[n]}}\right). \tag{6}$$

Finally, we must show for l, g, c such that $l + c \neq 0$, $g + c \neq 0$, and $1 \leq l + c + g \leq n$,

$$(l + g + \delta n)\alpha_{l+c} - l\alpha_{l+c-1} - g\alpha_{l+c+1} \leq \frac{1}{e^{[n]}} + \delta C_0 n \log n. \tag{7}$$

We now verify each of these inequalities, using the properties of α_i stated in Lemmas 2, 3, and 4. For (5), we have

$$\begin{aligned} (l + \delta n)\alpha_l - l\alpha_{l-1} \;&=\; l\beta_{l-1} + \delta n\alpha_l \;=\; \beta_l + \frac{1}{e^{[n]}} + \delta n\alpha_l \\ &\leq \beta_0 + \frac{1}{e^{[n]}} + \delta n\alpha_l \;=\; 1 + \delta n\alpha_l \;\leq\; 1 + C_0\delta n \log n. \end{aligned}$$

Inequality (6) follows directly from the fact that $g \geq 1$ and $\alpha_1 = 1 - \frac{1}{e^{[n]}}$. For inequality (7), we consider two cases. First, suppose $g = 0$ and so $c \geq 1$. Then, we have

$$\begin{aligned}(l+\delta n)\alpha_{l+c} - l\alpha_{l+c-1} \leq l\beta_l + \delta n\alpha_{l+c} &= \beta_{l+1} + \frac{1}{e^{[n]}} - \beta_l + \delta n\alpha_{l+c} \\ &\leq \frac{1}{e^{[n]}} + \delta n\alpha_{l+c} \leq \frac{1}{e^{[n]}} + C_0\delta n\log n.\end{aligned}$$

If $g \geq 1$, then we have

$$\begin{aligned}(l+g+\delta n)\alpha_{l+c} - l\alpha_{l+c-1} - g\alpha_{l+c+1} &= l\beta_{l+c-1} - g\beta_{l+c} + \delta n\alpha_{l+c} \\ \leq l\beta_{l+c-1} - \beta_{l+c} + \delta n\alpha_{l+c} &= \frac{1}{e^{[n]}} - c\beta_{l+c} + \delta n\alpha_{l+c} \leq \frac{1}{e^{[n]}} + C_0\delta n\log n.\end{aligned}$$

This completes the proof of Theorem 5. ◀

We obtain the following corollary by setting

$$\delta = \frac{\epsilon}{C_0 n\log n\left(1 - \frac{1}{e^{[n]}} - \epsilon\right)} = O\left(\frac{\epsilon}{n\log n}\right):$$

▶ **Corollary 6.** *Algorithm* LocalSearch$(C_1\epsilon/(n\log n))$ *is a* $(1 - 1/e^{[n]} - \epsilon)$*-approximation algorithm, for some universal constant* C_1 *and all* $\epsilon > 0$.

Since $e > e^{[n]}$, the same proof shows that if we replace $e^{[n]}$ by e in the definition of the sequence α_i, then the resulting approximation ratio is $1 - 1/e - \epsilon$.

Now, we turn to the run-time of Algorithm 2. First, we note that by keeping track of how many times each element of U is covered by the current solution, we can compute the change in f due to adding or removing a set from the solution using only $O(u)$ value oracle calls. The initial greedy phase takes time $O(nsu)$. Each iteration of the local search phase requires ns calls to the independence oracle for $\mathfrak{m}$ and $O(nsu)$ calls to the value oracle for w. Thus, each iteration can be completed in time $O(nsu)$. We now bound the total number of improvements that the algorithm can make.

▶ **Lemma 7.** *For any* $\delta > 0$, *Algorithm* LocalSearch(δ) *makes at most* $O(\delta^{-1})$ *improvements.*

Proof. Let $\hat{S}$ be the solution produced by the initial greedy phase of LocalSearch(δ), and let $\hat{O} = \arg\max_{S\in\mathfrak{m}} f(S)$. Then, LocalSearch$(\delta)$ makes at most $\log_{1+\delta}(f(\hat{O})/f(\hat{S}))$ improvements. Because the sequence of coefficients α_i is increasing and concave, for any $S \subseteq T \subset \mathcal{F}$ and $A \in \mathcal{F}\setminus T$ we must have $0 \leq f(T+A) - f(T) \leq f(S+A) - f(S)$, and so f is monotone submodular. Thus, the classical result of Fischer, Nemhauser, and Wolsey [10] implies that the greedy algorithm is a 2-approximation for maximizing f, so $2f(\hat{S}) \geq f(\hat{O})$. Algorithm LocalSearch$(\delta)$ can therefore make at most $\log_{1+\delta} 2 = \frac{\log 2}{\log(1+\delta)} = O(\delta^{-1})$ improvements. ◀

By setting $\epsilon = 1/e^{[n]} - 1/e$, we would obtain a clean $(1 - 1/e)$-approximation algorithm. However, since $1 - 1/e^{[n]}$ is very close to $1 - 1/e$, the resulting δ would be superpolynomial in n, and so we would not obtain a polynomial-time algorithm. Instead, we use a partial enumeration technique described by Khuller et al. [13]. Effectively, we try to “guess" a single set in the optimal solution, and then run LocalSearch on a contracted instance containing this set. We then iterate over all possible guesses.

▶ **Definition 8.** Let $\mathcal{I} = \langle U, w, \mathcal{F}, \mathfrak{m}\rangle$ be an instance of maximum coverage, where U is the universe, w is the weight function, $\mathcal{F}$ is a family of subsets of U, and $\mathfrak{m}$ is a matroid defined over $\mathcal{F}$.

Let $A \in \mathcal{F}$. The *contracted* instance $\mathcal{I}|_A = \langle U|_A, w|_A, \mathcal{F}|_A, \mathfrak{m}|_A \rangle$ is defined as follows:

$$U|_A = U,$$
$$w|_A = u \mapsto \begin{cases} w(u) & u \notin A, \\ 0 & u \in A, \end{cases}$$
$$\mathcal{F}|_A = \mathcal{F} - A,$$
$$\mathfrak{m}|_A = \mathfrak{m}/A = \{S \subset \mathcal{F} : S + A \in \mathfrak{m}\}.$$

Note that the definition of $w|_A$ directly implies that

$$w|_A \left(\bigcup S\right) = w\left(\bigcup S \setminus A\right) = w\left(\bigcup S \cup A\right) - w(A). \tag{8}$$

We can now formulate the new algorithm. Algorithm Approx simply runs LocalSearch(δ) on the instance $\mathcal{I}|_A$ for each $A \in \mathcal{F}$, and returns the best resulting solution of the original instance $\mathcal{I}$.

Algorithm 3 Approx(δ)

```
for $A \in \mathcal{F}$ do
    Let $S_A$ be the result of running LocalSearch($\delta$) on instance $\mathcal{I}|_A$
end for
$A \leftarrow \arg\max_{A \in \mathcal{F}} w(\bigcup S_A \cup A)$
return $S_A + A$
```

▶ **Theorem 9.** *If* LocalSearch(δ) *has an approximation ratio of* θ *on matroids of rank* $n-1$ *then* Approx(δ) *has an approximation ratio of* $1/n + (1-1/n)\theta$ *on matroids of rank* n.

Proof. Let O be some optimal solution, and $A \in O$ be a set of maximum weight in O. Submodularity implies that $w(O) \le \sum_{B \in O} w(B) \le nw(A)$, and so $w(A) \ge w(O)/n$. Since $O - A \in \mathfrak{m}|_A$, we have $w|_A(\bigcup S_A) \ge \theta \cdot w|_A(\bigcup O \setminus A)$. From identity (8) we then have

$$\begin{aligned} w\left(\bigcup S_A \cup A\right) &= w(A) + w|_A\left(\bigcup S_A\right) \\ &\ge w(A) + \theta \cdot w|_A\left(\bigcup O \setminus A\right) \\ &= (1-\theta)w(A) + \theta \cdot w\left(\bigcup O\right) \\ &\ge \left(\frac{1-\theta}{n} + \theta\right) w\left(\bigcup O\right). \end{aligned}$$ ◀

▶ **Corollary 10.** *For some universal constant* C_2 *and all* $n \ge 3$*, Algorithm* Approx($C_2/(n^2 \log n)$) *has an approximation ratio of at least* $1 - 1/e$.

Proof. Let $C_2 = 3C_1$, where C_1 is the constant defined in Corollary 6. The corollary implies that LocalSearch($C_2/(n^2 \log n)$) has an approximation ratio of $1 - 1/e^{[n-1]} - 1/3n$. Theorem 9 implies that the approximation ratio of Approx($C_2/(n^2 \log n)$) is

$$\frac{1}{n} + \left(1 - \frac{1}{n}\right)\left(1 - \frac{1}{e^{[n-1]}} - \frac{1}{3n}\right) \qquad \ge 1 - \frac{1}{e^{[n-1]}} + \frac{\frac{1}{e^{[n-1]}} - \frac{1}{3}}{n} \ge 1 - \frac{1}{e},$$

using the fact that $e^{[n-1]} \le e^{[2]} = 3$. ◀

Our final algorithm Approx($C_2/(n^2 \log n)$) makes s calls to LocalSearch. Each of these calls makes at most $O(n^2 \log n)$ improvements, each taking time $O(nsu)$. The final runtime is thus $\tilde{O}(n^3 s^2 u)$.

References

1 Alexander A. Ageev and Maxim I. Sviridenko. Pipage rounding: a new method of constructing algorithms with proven performance guarantee. *J. of Comb. Opt.*, 8:17–30, 2004.

2 Paola Alimonti. New local search approximation techniques for maximum generalized satisfiability problems. In *Algorithms and Complexity*, volume 778 of *LNCS*, pages 40–53. Springer, 1994.

3 Richard A. Brualdi. Comments on bases in dependence structures. *Bull. Austral. Math. Soc.*, 1:161–167, 1969.

4 Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. In *IPCO*, 2007.

5 Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. In *STOC*, 2008.

6 Chandra Chekuri and Amit Kumar. Maximum coverage problem with group budget constraints and applications. In *APPROX*, pages 72–83, 2004.

7 Reuven Cohen and Liran Katzir. The generalized maximum coverage problem. *Inf. Proc. Lett.*, 108(1):15–22, 2008.

8 Gerard Cornuejols, Marshall L. Fisher, and George L. Nemhauser. Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management Science*, 23(8):789–810, April 1977.

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

10 M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions—II. In *Polyhedral Combinatorics*, volume 8 of *Mathematical Programming Studies*, pages 73–87. Springer Berlin Heidelberg, 1978.

11 Dorit S. Hochbaum and Anu Pathria. Analysis of the greedy approach in covering problems. *Naval Research Quarterly*, 45:615–627, 1998.

12 Sanjeev Khanna, Rajeev Motwani, Madhu Sudan, and Umesh V. Vazirani. On syntactic versus computational views of approximability. *SIAM J. Comp.*, 28:164–191, 1998.

13 Samir Khuller, Anna Moss, and Joseph (Seffi) Naor. The budgeted maximum coverage problem. *Inf. Process. Lett.*, 70:39–45, April 1999.

14 Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41:960–981, September 1994.

15 Andreas S. Schulz, James B. Orlin, and Abraham P. Punnen. Approximate local search in combinatorial optimization. *SIAM J. Comp.*, pages 587–596, 2004.

16 Aravind Srinivasan. Distributions on level-sets with applications to approximation algorithms. In *FOCS*, pages 588–599, 2001.

A Appendix

A.1 Proofs Omitted From Section 3

▶ **Lemma 1.** *For all $n \geq 2$ we have $e < e^{[n]}$ and $e^{[n]} > e^{[n+1]}$.*

Proof. For the first claim, note that

$$e^{[n]} - e = \sum_{k=0}^{n-1} \frac{1}{k!} + \frac{1}{(n-1)!(n-1)} - \sum_{k=0}^{\infty} \frac{1}{k!} = \frac{1}{(n-1)!(n-1)} - \sum_{k=n}^{\infty} \frac{1}{k!}$$
$$> \frac{1}{(n-1)!(n-1)} - \frac{1}{n!} \sum_{k=0}^{\infty} \frac{1}{n^k} = \frac{1}{(n-1)!(n-1)} - \frac{1}{n!}\frac{n}{n-1} = 0.$$

For the next claim,

$$e^{[n+1]} - e^{[n]} = \frac{1}{n!} + \frac{1}{n!n} - \frac{1}{(n-1)!(n-1)} = \frac{n(n-1)+n-1-n^2}{n!n(n-1)} = \frac{-1}{n!n(n-1)}. \quad ◀$$

▶ **Lemma 2.** *Let $\beta_i = \alpha_{i+1} - \alpha_i$. Then, the β_i satisfy the recurrence relation*

$$\beta_0 = 1 - \frac{1}{e^{[n]}}, \quad \beta_i = i\beta_{i-1} - \frac{1}{e^{[n]}}. \tag{9}$$

Furthermore, they are in fact equal to

$$\beta_i = i!\left(1 - \frac{1}{e^{[n]}}\sum_{k=0}^{i}\frac{1}{k!}\right). \tag{10}$$

Proof. Clearly $\beta_0 = \alpha_1 - \alpha_0 = 1 - \frac{1}{e^{[n]}}$. In general, we have

$$\beta_i = \alpha_{i+1} - \alpha_i = i(\alpha_i - \alpha_{i-1}) - \frac{1}{e^{[n]}} = i\beta_{i-1} - \frac{1}{e^{[n]}}.$$

Using this recurrence relation, we can prove formula (10) by induction. It is immediate in the case $i = 0$. For $i > 0$, we have

$$\begin{aligned}\beta_i = i\beta_{i-1} - \frac{1}{e^{[n]}} &= i(i-1)!\left(1 - \frac{1}{e^{[n]}}\sum_{k=0}^{i-1}\frac{1}{k!}\right) - i!\frac{1}{i!e^{[n]}} \\ &= i!\left(1 - \frac{1}{e^{[n]}}\sum_{k=0}^{i}\frac{1}{k!}\right). \quad ◀\end{aligned}$$

▶ **Lemma 3.** *For all $i < n$, $\beta_i > 0$ and $\beta_{i+1} \le \beta_i$.*

Proof. The inequality $\beta_i > 0$ follows directly from (10) and the fact that $\sum_{k=0}^{i}\frac{1}{k!} < e^{[n]}$ for all $i < n$. For the second claim, if $i = 0$, then we have $\beta_1 = \beta_0 - \frac{1}{e^{[n]}} < \beta_0$. If $i \ge 1$, we note that:

$$\beta_{i+1} - \beta_i = i\beta_i - \frac{1}{e^{[n]}} = i \cdot i!\left(1 - \frac{1}{e^{[n]}}\sum_{k=0}^{i}\frac{1}{k!}\right) - \frac{1}{e^{[n]}} \le i \cdot i!\left(1 - \frac{1}{e^{[i+1]}}\sum_{k=0}^{i}\frac{1}{k!}\right) - \frac{1}{e^{[i+1]}},$$

where the last line follows from the fact that $e^{[n]} \le e^{[i+1]}$ for all $1 \le i \le n-1$. Now, set $a = \frac{1}{i \cdot i!}$ and $b = \sum_{k=0}^{i}\frac{1}{k!}$. We have $e^{[i+1]} = a + b$, and the final expression above is equal to:

$$\frac{1}{a} - \frac{1}{a}\cdot\frac{b}{a+b} - \frac{1}{a+b} = \frac{a+b-b-a}{a(a+b)} = 0. \quad ◀$$

▶ **Lemma 4.** *There exists a universal constant C_0 such that for all $i \le n$, $\alpha_i \le C_0 \log i$.*

Proof. The recurrence (9) and Lemma 3 imply that:

$$\beta_{i-1} = \frac{\beta_i + \frac{1}{e^{[n]}}}{i} \le \frac{\beta_0 + \frac{1}{e^{[n]}}}{i} = \frac{1}{i}.$$

We obtain the bound on α_i by summing:

$$\alpha_i = \sum_{k=0}^{i-1}\beta_k \le \sum_{k=1}^{i}\frac{1}{k} = \log i + O(1). \quad ◀$$

A.2 Optimality of f

The analysis of LocalSearch depends intimately on the exact values of the constants α_i used to define the function f. Theorem 5 shows that the locality ratio of LocalSearch(0) is $1 - 1/e^{[n]}$. We proceed to show that our choice of values for the constants α_i is optimal.

▶ **Theorem 11.** *The locality ratio of* LocalSearch(0) *is at most* $1 - 1/e^{[n]}$ *for* any *setting of the* α_i.

Proof. Consider the following instances of maximum coverage over a matroid, given in terms of the symmetric variables $x_{l,c,g}$:

$$\begin{aligned} x^{(l)}_{l,0,1} &= \alpha_1, \quad x^{(l)}_{0,0,1} = (l+1)\alpha_l - l\alpha_{l-1} - \alpha_{l+1}, \quad \text{for } 1 \le l \le n-1, \\ x^{(n)}_{n-1,1,0} &= \alpha_1, \quad x^{(n)}_{0,0,1} = (n-1)\alpha_n - (n-1)\alpha_{n-1}. \end{aligned}$$

The values of $x^{(l)}_{0,0,1}$ are chosen to maximize $w(\bigcup O)$ while maintaining local optimality, according to inequality (3). The locality ratio in each instance is given by $\frac{\alpha_1}{\alpha_1 + x^{(l)}_{0,0,1}}$. Let θ be the minimum of all these expressions, an upper bound on the locality ratio of the algorithm. Without loss of generality, assume that $\alpha_1 = \theta$. We deduce from each line an inequality $x^{(l)}_{0,0,1} \le 1 - \theta$. In particular,

$$-\alpha_1 + \sum_{l=1}^{n-1} \frac{x^{(l)}_{0,0,1}}{l!} + \frac{x^{(n)}_{0,0,1}}{(n-1)(n-1)!} \le -\theta + (1-\theta)\left[\sum_{l=1}^{n-1} \frac{1}{l!} + \frac{1}{(n-1)(n-1)!}\right] = (1-\theta)(e^{[n]}-1) - \theta.$$

The left-hand side is a linear combination of the α_i in which (as we shall show) the total coefficient of each α_i is zero. We deduce that $\theta \le 1 - 1/e^{[n]}$.

Put $x^{(0)}_{0,0,1} = -\alpha_1$. Each α_i for $1 \le i \le n-2$ appears in three expressions $x^{(j)}_{0,0,1}$ for $j = i-1, i, i+1$, with total coefficient

$$-\frac{1}{(i-1)!} + \frac{i+1}{i!} - \frac{i+1}{(i+1)!} = \frac{-i + i + 1 - 1}{i!} = 0.$$

The coefficients of α_{n-1} and α_n are, respectively,

$$\begin{aligned} -\frac{1}{(n-2)!} + \frac{n}{(n-1)!} - \frac{n-1}{(n-1)(n-1)!} &= \frac{-(n-1)^2 + n(n-1) - (n-1)}{(n-1)(n-1)!} = 0, \\ -\frac{1}{(n-1)!} + \frac{(n-1)}{(n-1)(n-1)!} &= \frac{(n-1)-(n-1)}{(n-1)(n-1)!} = 0. \end{aligned}$$ ◀

A.3 Oblivious local search on SBO matroids

Algorithm 1, oblivious k-local search, is far inferior than our algorithm for maximum coverage over a matroid constraint. In Section 3 we presented an instance on which its approximation ratio is only $(n-1)/(2n-1-k)$. This approximation ratio is tight for *strongly base orderable matroids*. These matroids satisfy the additional property that for any two bases S, O, there is a permutation $\pi\colon S \to O$ such that for each $T \subset S$, the set $S \setminus T \cup \{\pi(A) : A \in T\}$ is a basis.

▶ **Theorem 12.** *Algorithm 1 has an approximation ratio of* $(n-1)/(2n-1-k)$ *on strongly base orderable matroids.*

Proof. Let $S = \{S_1, \dots, S_n\}$ be the local optimum given by the algorithm, and let $O = \{O_1, \dots, O_n\}$ be some global optimum. The fact that the matroid is strongly base orderable ensures that we can label the sets in such a way that for each set of indices I, the matroid contains $S_{-I} \cup O_I = \{S_i : i \notin I\} \cup \{O_i : i \in I\}$. Local optimality implies that $w(\bigcup S) \geq w(\bigcup S_{-I} \cup \bigcup O_I)$ whenever $|I| \leq k$. Summing this over all sets I of cardinality k, we obtain

$$\binom{n}{k} w\left(\bigcup S\right) \geq \sum_{I \subset [n] \colon |I|=k} w\left(\bigcup S_{-I} \cup \bigcup O_I\right).$$

We prove below the following crucial inequality:

$$\sum_{I \subset [n] \colon |I|=k} w\left(\bigcup S_{-I} \cup \bigcup O_I\right) \geq \binom{n-2}{k} w\left(\bigcup S\right) + \binom{n-1}{k-1} w\left(\bigcup O\right). \tag{11}$$

Given this inequality, we deduce that

$$w\left(\bigcup S\right) \geq \frac{\binom{n-1}{k-1}}{\binom{n}{k} - \binom{n-2}{k}} w\left(\bigcup O\right) = \frac{n-1}{2n-1-k} w\left(\bigcup O\right).$$

We prove (11) by counting the number of times an element $x \in \bigcup S \cup \bigcup O$ appears on both sides. Looking at the right-hand side, it is clear that it is enough to consider the case where x belongs to at most one set S_i and at most one set O_j. Denote the weight of x in the left-hand side by L, and its weight in the right-hand side by R. Our goal is to show that in all cases, $L \geq R$.

If x belongs only to a set S_i then

$$L = \binom{n-1}{k} \geq \binom{n-2}{k} = R.$$

If x belongs only to a set O_j then

$$L = \binom{n-1}{k-1} = R.$$

If x belongs to both S_i and O_j for $i \neq j$ then (using Pascal's identity)

$$L = \binom{n}{k} - \binom{n-2}{k-1} = \binom{n-2}{k} + \binom{n-1}{k-1} = R.$$

Finally, if x belongs to both S_i and O_i then

$$L = \binom{n}{k} \geq \binom{n-2}{k} + \binom{n-1}{k-1} = R.$$ ◀

A.4 Convergence of oblivious local search

Algorithm LocalSearch stops at an approximate local optimum. This guarantees that its running time is polynomial. Such a modification is necessary since there are instances for which the local search algorithm could run through an exponential sequence of local improvements. It is an intriguing direction for future research to analyze how well different modifications, such as the *best improvement* heuristic, are able to bound the running time.

Here, we present an instance which has bad behavior for oblivious 1-local search. Let $s \geq 1$ be an integer, and consider the following instance $\mathcal{I}_s$ of maximum coverage over a matroid. The sets are:

$$X_l = \{x_l\}, \quad Y_l = \{y_l\} \cup \{z_r : r < l\}, \quad Z_l = \{z_l\}, \quad 1 \leq l \leq s.$$

The matroid is defined by allowing at most one set among $\{X_l, Y_l, Z_l\}$ for each l. The weights are

$$w(x_l) = 1, \quad w(y_l) = 2, \quad w(z_l) = 3 \cdot 2^{l-1}.$$

▶ **Theorem 13.** *Let $s \geq 1$ be an integer. There is a sequence σ_s of length $\ell_s = 3 \cdot 2^s - s - 2$ of bases of $\mathcal{I}_s$, starting at $\{X_l : 1 \leq l \leq s\}$ and terminating at $\{Z_l : 1 \leq l \leq s\}$, whose weight increases by 1 at each step.*

Proof. The proof is by induction. For $s = 1$ the sequence is $\sigma_1 = \{X_1\}, \{Y_1\}, \{Z_1\}$, of length $\ell_1 = 3$.

Suppose the claim is true for s. We prove it for $s + 1$. Denote the individual elements of σ_s by $\sigma_s(1), \ldots, \sigma_s(\ell_s)$. The new sequence is defined as follows:

$$\begin{aligned}
&\sigma_{s+1}(t) = \sigma_s(t) \cup \{X_{s+1}\}, && 1 \leq t \leq \ell_s,\\
&\sigma_{s+1}(\ell_s + t) = \{X_l : 1 \leq l < t\} \cup \{Z_l : t \leq l \leq s\} \cup \{Y_{s+1}\}, && 1 \leq t \leq s + 1,\\
&\sigma_{s+1}(\ell_s + s + 1 + t) = \sigma_s(t) \cup \{Z_{s+1}\}, && 1 \leq t \leq \ell_s.
\end{aligned}$$

The weight increases by 1 at each step during the initial and final parts by induction. In the middle part, since $Z_l \subset Y_{s+1}$, when Z_l is changed to X_l the weight increases by $w(x_l) = 1$. When moving from the initial part to the middle part, the weight increases by $w(y_{s+1}) - w(x_{s+1}) = 1$. Finally, when moving from the middle part to the final part, the weight increases by

$$w(Z_{s+1}) - w(Y_{s+1}) = w(z_{s+1}) - \sum_{l=1}^{s} w(z_s) - w(y_s) = 3\left(2^s - \sum_{t=1}^{s} 2^{t-1}\right) - 2 = 1.$$

The length of the sequence is

$$\ell_{s+1} = 2\ell_s + s + 1 = 2(3 \cdot 2^s - s - 2) + s + 1 = 3 \cdot 2^{s+1} - s - 3.$$ ◀