A $(k+3)/2$-approximation algorithm for monotone submodular k-set packing and general k-exchange systems

Justin Ward

Department of Computer Science, University of Toronto
Toronto, Canada
jward@cs.toronto.edu

Abstract

We consider the monotone submodular k-set packing problem in the context of the more general problem of maximizing a monotone submodular function in a k-exchange system. These systems, introduced by Feldman et al. [9], generalize the matroid k-parity problem in a wide class of matroids and capture many other combinatorial optimization problems. We give a deterministic, non-oblivious local search algorithm that attains an approximation ratio of $(k+3)/2+\epsilon$ for the problem of maximizing a monotone submodular function in a k-exchange system, improving on the best known result of $k+\epsilon$, and answering an open question posed by Feldman et al.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases k-set packing, k-exchange systems, submodular maximization, local search, approximation algorithms

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

1 Introduction

In the general k-set packing problem, we are given a collection $\mathcal{G}$ of sets, each with at most k elements, and an objective function $f : 2^{\mathcal{G}} \to \mathbb{R}_+$ assigning each subset of $\mathcal{G}$ a value and seek a collection $S \subseteq \mathcal{G}$ of pairwise-disjoint sets maximizing f. In the special case that $f(A) = |A|$, we obtain the *unweighted k-set packing problem*. Similarly, if f is linear function, so that $f(A) = \sum_{e \in A} w(e)$ for some weight function $w : \mathcal{G} \to \mathbb{R}_+$ we obtain the *weighted k-set packing problem*. In this paper we consider the case in which f may be any monotone submodular function.

For unweighted k-set packing, Hurkens and Schrijver [15] and Halldórsson [13] independently obtained a $k/2+\epsilon$ approximation via a simple local search algorithm. Using similar techniques, Arkin and Hassin [1] obtained a $k-1+\epsilon$ approximation for weighted k-set packing, and showed that this result is tight for their simple local search algorithm. Chandra and Halldórsson [5] showed that a more sophisticated local search algorithm, which starts with a greedy solution and always chooses the best possible local improvement at each stage, attains an approximation ratio of $2(k+1)/3+\epsilon$. This was improved further by Berman [2], who gave a *non-oblivious* local search algorithm yielding a $(k+1)/2+\epsilon$ approximation for weighted k-set packing. Non-oblivious local search [16] is a variant of local search in which an auxiliary objective function, rather than the problem's given objective, is used to evaluate solutions. In the case of Berman, the local search procedure repeatedly seeks to improve the sum of the *squares* of the weights in the current solution, rather than the sum of the weights.

Many of the above local search algorithms for k-set packing yield the same approximations for the more general problem of finding maximum independent sets in $(k+1)$-claw free graphs.

STACS 2012.
Editors: Christoph Dürr, Thomas Wilke; pp. 1–12

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Local search techniques have also proved useful for other generalizations of k-set packing, including variants of the matroid k-parity problem [18, 20]. Motivated by the similarities between these problems, Feldman et al. [9] introduced the class of *k-exchange* systems, which captures many problems amenable to approximation by local search algorithms. These systems are formulated in the general language of independence systems, which we now briefly review.

An independence system is specified by a ground set $\mathcal{G}$, and a hereditary (i.e. non-empty and downward-closed) family $\mathcal{I}$ of subsets of $\mathcal{G}$. The sets in $\mathcal{I}$ are called *independent sets*, and the inclusion-wise maximal sets in $\mathcal{I}$ are called *bases*. Given an independence system $(\mathcal{G}, \mathcal{I})$ and a function $f : 2^{\mathcal{G}} \to \mathbb{R}_+$, we shall consider the problem of finding an independent set $S \in \mathcal{I}$ that maximizes f.

The class of k-exchange systems satisfy the following additional property:

▶ **Definition 1** (k-exchange system [9])**.** An independence system $(\mathcal{G}, \mathcal{I})$ is a *k-exchange system* if, for all A and B in $\mathcal{I}$, there exists a multiset $Y = \{Y_e \subseteq B \setminus A \mid e \in A \setminus B\}$, containing a subset Y_e of $B \setminus A$ for each element $e \in A \setminus B$, that satisfies:

(K1) $|Y_e| \leq k$ for each $x \in A$.
(K2) Every $x \in B \setminus A$ appears in at most k sets of Y.
(K3) For all $C \subseteq A \setminus B$, $(B \setminus (\bigcup_{e \in C} Y_e)) \cup C \in \mathcal{I}$.

We call the set Y_e in Definition 1 the *neighborhood* of e in B. For convenience, we extend the collection Y in Definition 1 by including the set $Y_x = \{x\}$ for each element $x \in A \cap B$. It is easy to verify that the resulting collection still satisfies conditions (K1)–(K3).

The 1-exchange systems are precisely the class of strongly base orderable matroids described by Brualdi [3]. This class is quite large and includes all gammoids, and hence all transversal and partition matroids. For $k > 1$, the class of k-exchange systems may be viewed as a common generalization of the matroid k-parity problem in strongly base orderable matroids and the independent set problem in $(k+1)$-claw free graphs. Feldman et al. showed that k-exchange systems encompass a wide variety of combinatorial optimization problems, including k-set packing, intersection of k strongly base orderable matroids, b-matching (here $k = 2$), and asymmetric traveling salesperson (here $k = 3$).

Our results hold for any k-exchange system, and so we present them in the general language of Definition 1. However, the reader may find it helpful to think in terms of a concrete problem, such as the k-set packing problem. In that case, the ground set $\mathcal{G}$ is the given collection of sets, and a sub-collection of sets $S \subseteq \mathcal{G}$ is independent if and only if all the sets in S are disjoint. Given A and B as in Definition 1, Y_e is the set of all sets in B that contain any element contained by the set $e \in A$ (i.e. the set of all sets in B that are not disjoint from e). Then, property (K3) is immediate, and (K1) and (K2) follow directly from the fact that each set in $\mathcal{G}$ contains at most k elements.

1.1 Related Work

Recently, the problem of maximizing submodular functions subject to various constraints has attracted much attention. We focus here primarily on results pertaining to matroid constraints and related independence systems.

In the case of an arbitrary single matroid constraint, Calinescu et al. have attained a $e/(e-1)$ approximation for monotone submodular maximization via the *continuous greedy algorithm*. This result is tight, provided that $P \neq NP$ [6]. In the case of $k \geq 2$ simultaneous matroid constraints, an early result of Fisher, Nemhauser, and Wolsey [10] shows that

the standard greedy algorithm attains a $k+1$ approximation for monotone submodular maximization. Fischer et al. state further that the result can be generalized to k-systems (a full proof appears in Calinescu et al. [4]). More recently, Lee, Sviridenko, and Vondràk [19] have improved this result to give a $k+\epsilon$ approximation for monotone submodular maximization over $k \geq 2$ arbitrary matroid constraints via a simple, oblivious local search algorithm. Feldman et al. [9] used a similar analysis to show that oblivious local search attains a $k+\epsilon$ approximation for the class of k-exchange systems (here, again, $k \geq 2$). For the more general class of k-systems, Gupta et al. [12] give a $(1+\beta)(k+2+1/k)$ approximation, where β is the best known approximation ratio for unconstrained non-monotone submodular maximization.

In the case of unconstrained non-monotone submodular maximization, Feige, Mirrokni, and Vondrák [7] gave a randomized 2.5 approximation, which was iteratively improved by Gharan and Vondrák [11] and then Feldman, Naor, and Shwartz [8] to ≈ 2.38. For non-monotone maximization subject to k matroid constraints, Lee, Sviridenko, and Vondrák [17] gave a $k+2+1/k+\epsilon$ approximation, and later improved [19] this to a $k+1+1/(k-1)+\epsilon$ approximation. Again, the latter result is obtained by a standard local search algorithm. Feldman et al. [9] apply similar techniques to yield a $k+1+1/(k-1)+\epsilon$ approximation for non-monotone submodular maximization the general class of k-exchange systems.

1.2 Our Contribution

In the restricted case of a linear objective function, Feldman et al. [9] gave a non-oblivious local search algorithm inspired by Berman's algorithm [2] for $(k+1)$-claw free graphs. They showed that the resulting algorithm is a $(k+1)/2+\epsilon$ approximation for linear maximization in any k-exchange system. Here we consider a question posed in [9]: namely, whether a similar technique can be applied to the case of monotone submodular maximization in k-exchange systems. In this paper, we answer this question affirmatively, giving a non-oblivious local search algorithm for monotone submodular maximization in a k-exchange system. As in [9], the k-exchange property is used only in the analysis of our algorithm. Our algorithm attains an approximation ratio of $\frac{k+3}{2}+\epsilon$. For $k>3$, this improves upon the $k+\epsilon$ approximation obtained by the oblivious local search algorithm presented in [9]. Additionally, we note that our algorithm runs in time polynomial in ϵ^{-1}, while the $k+\epsilon$ approximation algorithm of [9] requires time exponential in ϵ^{-1}.

As a consequence of our general result, we obtain an improved approximation guarantee of $\frac{k+3}{2}$ for a variety of monotone submodular maximization problems (some of which are generalizations of one another) including: k-set packing, independent sets in $(k+1)$-claw free graphs, k-dimensional matching, intersection of k strongly base orderable matroids, and matroid k-parity in a strongly base orderable matroid. In all cases, the best previous result was $k+\epsilon$.

2 A First Attempt at the Submodular Case

Before presenting our algorithm, we describe some of the difficulties that arise when attempting to adapt the non-oblivious local search algorithm of [2] and [9] to the submodular case. Our hope is that this will provide some intuition for our algorithm, which we present in the next section.

We recall that a function $f: 2^{\mathcal{G}} \to \mathbb{R}_+$ is submodular if $f(A)+f(B) \geq f(A \cup B)+f(A \cap B)$ for all $A, B \subseteq \mathcal{G}$. Equivalently, f is submodular if for all $S \subseteq T$ and all $x \notin T$, $f(S+x)-f(S) \geq f(T+x)-f(T)$. In other words, submodular functions are characterized by decreasing

marginal gains. We say that a submodular function f is monotone if it additionally satisfies $f(S) \leq f(T)$ for all $S \subseteq T$.

The non-oblivious algorithm of [9] for the linear case is shown in Algorithm 1. It repeatedly searches for a k-replacement (A, B) that improves the non-oblivious potential function w^2. Formally, we call the pair of sets (A, B), where $B \subseteq S$ and $A \subseteq \mathcal{G} \setminus (S \setminus B)$ a *k-replacement* if $|A| \leq k$, $|B| \leq k^2 - k + 1$ and $(S \setminus B) \cup A \in \mathcal{I}$. If $w(e) = f(\{e\})$ is the weight assigned to an element e, then the non-oblivious potential function used by Algorithm 1 is given by $w^2(S) = \sum_{e \in S} w(e)^2$. That is, our non oblivious potential function $w^2(S)$ is simply the sum of the *squared* weights of the elements of S.[1] We use the fact that $w^2(S) > w^2((S \setminus B) \cup A)$ if and only if $w^2(A) > w^2(B)$, to slightly simplify the search for an improvement.

Algorithm 1: Non-Oblivious Local Search for Linear Objective Functions

Input:
- Ground set $\mathcal{G}$
- Membership oracle for $\mathcal{I} \subseteq 2^{\mathcal{G}}$
- Value oracle for monotone submodular function $f : 2^{\mathcal{G}} \to \mathbb{R}_+$
- Approximation parameter $\epsilon \in (0, 1)$

Let $S_{init} = \{\arg\max_{e \in \mathcal{G}} w(e)\}$;
Let $\alpha = w(S_{init})\epsilon/n$;
Round all weights $w(e)$ down to integer multiples of α;
$S \leftarrow S_{init}$;
$S_{old} \leftarrow S$;
repeat
  foreach k-replacement (A, B) **do**
    if $w^2(A) > w^2(B)$ **then**
      $S_{old} \leftarrow S$;
      $S \leftarrow (S \setminus B) \cup A$;
      break;
until $S_{old} = S$;
return S;

In the monotone submodular case, we can no longer necessarily represent f as a sum of weights. However, borrowing some intuition from the greedy algorithm, we might decide to replace each weight $w(e)$ in the potential function w with the marginal gain/loss associated with e. That is, at the start of each iteration of the local search algorithm, we assign each element $e \in \mathcal{G}$ weight $w(e) = f(S + e) - f(S - e)$, where S is the algorithm's current solution, then proceed as before. Note that $w(e)$ is simply the marginal gain attained by adding e to S (in the case that $e \notin S$) or the marginal loss suffered by removing e from S (in the case that $e \in S$). We define the non-oblivious potential function w^2 in terms of the resulting weight function w as before.

Unfortunately, the resulting algorithm may fail to terminate, as the following small example shows. We consider a simple, unweighted coverage function on the universe $U =$

[1] To ensure polynomial-time convergence, Algorithm 1 first round the weights down to integer multiples of a suitable small value α, related to the approximation parameter ϵ. The algorithm then converges in time polynomial in ϵ^{-1} and n, at a loss of only $(1 - \epsilon)^{-1}$ in the approximation factor.

$\{a, b, c, x, y, z\}$. Let:

$$S_1 = \{a, b\} \qquad S_3 = \{x, y\}$$
$$S_2 = \{a, c\} \qquad S_4 = \{x, z\}$$

Our ground set $\mathcal{G}$ is then $\{1, 2, 3, 4\}$ and our objective function $f(A) = \left|\bigcup_{i \in A} S_i\right|$ for all $A \subseteq \mathcal{G}$. We consider the 2-exchange system with only 2 bases: $P = \{1, 2\}$ and $Q = \{3, 4\}$. For current solution $S = P$ we have $w(1) = w(2) = 1$ and $w(3) = w(4) = 2$. Since $w^2(\{1, 2\}) = 2 < 8 = w^2(\{3, 4\})$, the 2-replacement $(\{3, 4\}, \{1, 2\})$ is applied, and the current solution becomes Q. In the next iteration, we have $S = Q$, and $w(1) = w(2) = 2$ and $w(3) = w(4) = 1$, so the 2-replacement $(\{1, 2\}, \{3, 4\})$ is applied by the algorithm. This returns us to the solution to P, where the process repeats indefinitely.

3 The New Algorithm

Intuitively, the problem with this initial approach is that the weight function used at each step of the algorithm depends on the current solution S (since all marginals are taken with respect to S). Hence, it may be the case that a k-replacement (A, B) results in an improvement with respect to the current solution's potential function, but in fact results in a *decreased* potential value in the next iteration after the weights have been updated. Surprisingly, we can solve the problem by introducing *even more* variation in the potential function. Specifically, we allow the algorithm to use a different weight function not only for each current solution S, but also for *each k-replacement (A, B) that is considered*. We give the full algorithm at the end of this section and a detailed analysis in the next.

First, we describe the general approach we use to generate the weights used in our potential function. Rather than calculating all marginal gains with respect to S, we consider elements in some order and assign each element a weight corresponding to its marginal gain with respect to those elements that precede it. By carefully updating both the current solution and its order each time we apply a local improvement, we ensure that the algorithm converges to a local optimum.

The algorithm stores the current solution S as an ordered sequence $s_1, s_2, \ldots, s_{|S|}$. At each iteration of the local search, before searching for an improving k-replacement, it assigns a weight $w(s_i)$ to each $s_i \in S$, as follows. Let $S_i = \{s_j \in S : j \leq i\}$ be the set containing the first i elements of S. Then, the weight function w assigning weights to the elements of S is given by

$$w(s_i) = f(S_{i-1} + s_i) - f(S_{i-1}) = f(S_i) - f(S_{i-1})$$

for all $s_i \in S$. Note that our weight function satisfies

$$\sum_{s_i \in S} w(s_i) = \sum_{i=1}^{|S|} [f(S_i) - f(S_{i-1})] = f(S) - f(\emptyset) \leq f(S) \ . \tag{1}$$

In order to evaluate a k-replacement (A, B), we also need to assign weights to the elements in $A \subseteq \mathcal{G} \setminus (S \setminus B)$. We use a different weight function for each k-replacement (A, B), obtained as follows. We order A according to an arbitrary ordering $\prec$ on $\mathcal{G}$ and a_i be the ith element of A and $A_i = \{a_j \in A : j \leq i\}$. Then, the weight function $w_{(A,B)}$ assigning weights to the elements of A is given by

$$w_{(A,B)}(a_i) = f((S \setminus B) \cup A_{i-1} + a_i) - f((S \setminus B) \cup A_{i-1}) = f((S \setminus B) \cup A_i) - f((S \setminus B) \cup A_{i-1})$$

for all $a_i \in A$. Note that for every k-replacement (A, B),

$$\sum_{x \in A} w_{(A,B)}(a_i) \geq \sum_{i=1}^{|A|} [f((S \setminus B) \cup A_i) - f((S \setminus B) \cup A_{i-1})]$$
$$= f((S \setminus B) \cup A) - f(S \setminus B) \geq f(S \cup A) - f(S) \ , \quad (2)$$

where the last inequality follows from the submodularity of f. Note that since the function f is *monotone* submodular, all of the weights w and $w_{(A,B)}$ that we consider will be non-negative. This fact plays a crucial role in our analysis. Furthermore, the weights w assigned to elements in S remain fixed for all k-replacements considered in a single phase of the algorithm. Finally, we note that although both w and $w_{(A,B)}$ depend on the current solution S, we omit this dependence from our notation, instead stating explicitly when there is a chance of confusion which solution's weight function we are considering.

Our final algorithm appears in Algorithm 2. We start from an initial solution $S_{init} = \arg\max_{e \in \mathcal{G}} f(\{e\})$, consisting of the single element of largest value. When applying a k-replacement (A, B), the algorithm updates the ordered solution S in a fashion that ensures all of the elements of $S \setminus B$ precede those of A, while all elements of $S \setminus B$ and, respectively, A occur in the same relative order. As we shall see in the next section, this guarantees that the algorithm will converge to a local optimum. As in the linear case, we use the sum of squared weights $w^2(B) = \sum_{b \in B} w(b)^2$ and $w^2_{(A,B)}(A) = \sum_{a \in A} w_{(A,B)}(a)^2$ to guide the search.

To ensure polynomial-time convergence, we round all of our weights down to the nearest integer multiple of α, which depends on the parameter ϵ. This ensures that every improvement improves the current solution by an additive factor of at least α^2. Because of this rounding factor, we must actually work with the following analogs of (1) and (2):

$$\sum_{x \in S} w(x) \leq \sum_{i=1}^{|S|} [f(S_i) - f(S_{i-1})] = f(S) - f(\emptyset) \leq f(S) \quad (3)$$

$$\sum_{x \in A} w_{(A,B)}(x) \geq \sum_{i=1}^{|A|} [f((S \setminus B) \cup A_i) - f((S \setminus B) \cup A_{i-1}) - \alpha]$$
$$= f((S \setminus B) \cup A) - f(S \setminus B) - |A|\alpha \geq f(S \cup A) - f(S) - |A|\alpha \quad (4)$$

4 Analysis of Algorithm 2

We now analyze the approximation and runtime performance of Algorithm 2. We consider the worst-case ratio, or *locality gap*, $f(O)/f(S)$ where S is any locally optimal solution (with respect to Algorithm 2's potential function) and O is a globally optimal solution. We shall need the following technical lemma, which is a direct consequence of Lemma 1.1 in [19]. We give a proof here for the sake of completeness.

▶ **Lemma 2.** *Let f be a submodular function on $\mathcal{G}$, Let $T, S \subseteq \mathcal{G}$, and $\{T_i\}_{i=1}^t$ be a partition of T. Then,*

$$\sum_{i=1}^{t} [f(S \cup T_i) - f(S)] \geq f(S \cup T) - f(S)$$

Algorithm 2: Non-Oblivious Local Search

Input:
- Ground set $\mathcal{G}$
- Membership oracle for $\mathcal{I} \subseteq 2^{\mathcal{G}}$
- Value oracle for monotone submodular function $f : 2^{\mathcal{G}} \to \mathbb{R}_+$
- Approximation parameter $\epsilon \in (0, 1)$

Fix an arbitrary ordering $\prec$ on the elements of $\mathcal{G}$
Let $S_{init} = \arg\max_{e \in \mathcal{G}} f(\{e\})$
Let $\delta = \left(1 + \frac{k+3}{2\epsilon}\right)^{-1}$ and $\alpha = f(S_{init})\delta/n$
$S \leftarrow S_{init}$ and $S_{old} \leftarrow S$
repeat
  $X \leftarrow \emptyset$
  for $i = 1$ **to** $|S|$ **do**
    $w(s_i) \leftarrow \lfloor (f(X + s_i) - f(X))/\alpha \rfloor \alpha$
    $X \leftarrow X + s_i$
  foreach k-replacement (A, B) **do**
    Let a_i be the ith element in A according to $\prec$
    $X \leftarrow S \setminus B$
    for $i = 1$ **to** $|A|$ **do**
      $w_{(A,B)}(a_i) \leftarrow \lfloor (f(X + a_i) - f(X))/\alpha \rfloor \alpha$
      $X \leftarrow X + a_i$
    if $w^2_{(A,B)}(A) > w^2(B)$ **then**
      $S_{old} \leftarrow S$
      Delete all elements in B from S
      Append the elements of A to the end of S, in the order given by $\prec$.
      break
until $S_{old} = S$
return S

Proof. Define $A_0 = S$ and then $B_i = T_i \setminus S$ and $A_i = B_i \cup A_{i-1}$, for all $1 \leq i \leq t$. Then, since $S \subseteq A_{i-1}$ for all $1 \leq i \leq t$, submodularity of f implies

$$f(T_i \cup S) - f(S) = f(B_i \cup S) - f(S) \leq f(B_i \cup A_{i-1}) - f(A_{i-1}) = f(A_i) - f(A_{i-1})$$

for all $1 \leq i \leq t$. Now, we have

$$\sum_{i=1}^{t} [f(S \cup T_i) - f(S)] \geq \sum_{i=1}^{t} [f(A_i) - f(A_{i-1})] = f(A_t) - f(A_0) = f(S \cup T) - f(S) \; . \quad \blacktriangleleft$$

4.1 Approximation Ratio of Algorithm 2

We now consider the approximation ratio of Algorithm 2. A general outline of the proof is as follows: we consider only a particular set of k-replacements (A, B) and derive (in Lemma 3) a relationship between the non-oblivious potential functions w^2 and $w^2_{(A,B)}$ and the weight functions w and $w_{(A,B)}$ for these k-replacements. We use this relationship to derive (in Lemma 4) a lower bound on the weight $w(x)$ of each element x in a locally optimal solution. Finally, in Theorem 5 we combine these lower bounds to obtain a bound on the locality gap of Algorithm 2.

For the rest of this subsection, we consider an arbitrary instance $(\mathcal{G}, \mathcal{I}, f)$ and suppose that S is a locally optimal solution returned by the algorithm on this instance, while O is a global optimum for this instance. Because S is locally optimal, we must have $w^2_{(A,B)}(A) \leq w^2(B)$ for every k-replacement (A, B), where w and each $w_{(A,B)}$ are the weight functions determined by S.

We now describe the set of k-replacements used in our analysis. We have $S, O \in \mathcal{I}$ for the k-exchange system $(\mathcal{G}, \mathcal{I})$. Thus, there must be a collection Y assigning each e of O a neighborhood $Y_e \subseteq S$, satisfying the conditions of Definition 1. For each $x \in S$, let P_x be the set of all elements $e \in O$ for which: (1) $x \in Y_e$ and (2) for all $z \in Y_e$, $w(z) \leq w(x)$. That is, P_x is the set of all elements of O which have x as their heaviest neighbor in S. Note that the construction of P_x depends on the weights w assigned to elements in S being fixed throughout each iteration and independent of the particular improvement under consideration.

We define $N_x = \bigcup_{e \in P_x} Y_e$, and consider (P_x, N_x). From property (K2) of Y we have $|P_x| \leq k$. Similarly, from property (K1) and the fact that all elements $e \in P_x$ have as a common neighbor $x \in Y_e$ we have $|N_x| \leq 1 + k(k-1) = k^2 - k + 1$. Finally, from property (K3) we have $(S \setminus N_x) \cup P_x \in \mathcal{I}$. Thus, (P_x, N_x) is a valid k-replacement for all of our sets P_x, $x \in S$. Furthermore, $\{P_x\}_{x \in S}$ is a partition of O, and by the definition of P_x, we have $w(x) \geq w(z)$ for all $z \in N_x$. Again, this depends on the weights of elements in S being the same for all k-replacements considered by the algorithm during a given phase.

The following extension of a theorem from [2], relates the squared weight potentials w^2 and $w^2_{(P_x,N_x)}$ to the weight functions w and $w_{(P_x,N_x)}$ for each of our k-replacements (P_x, N_x).

▶ **Lemma 3.** *For all $x \in S$, and $e \in P_x$,*

$$w_{(P_x,N_x)}(e)^2 - w^2(Y_e - x) \geq w(x) \cdot \big(2w_{(P_x,N_x)}(e) - w(Y_e)\big) \ .$$

Proof. First, we note that

$$0 \leq (w(x) - w_{(P_x,N_x)}(e))^2 = w(x)^2 - 2w(x) \cdot w_{(P_x,N_x)}(e) + w_{(P_x,N_x)}(e)^2 \ . \tag{5}$$

Additionally, since $e \in P_x$, every element z in Y_e has weight at most $w(x)$, and so

$$w^2(Y_e - x) = \sum_{z \in Y_e - x} w(z)^2 \ \leq \ w(x) \sum_{z \in Y_e - x} w(z) \ = \ w(x) \cdot w(Y_e - x) \ . \tag{6}$$

Adding (5) and (6) then rearranging terms using $w(x) \cdot w(Y_e - x) + w(x)^2 = w(x) \cdot w(Y_e)$ gives the desired result. ◀

We now prove the following lemma, which uses the local optimality of S to obtain a lower bound on the weight $w(x)$ of each element $x \in S$.

▶ **Lemma 4.** *For each $x \in S$, $w(x) \geq \sum_{e \in P_x} \big[2w_{(P_x,N_x)}(e) - w(Y_e)\big]$.*

Proof. Because S is locally optimal with respect to k-replacements, including in particular (P_x, N_x), we must have

$$w^2_{(P_x,N_x)}(P_x) \leq w^2(N_x) \ . \tag{7}$$

First, we consider the case $w(x) = 0$. Recall that all the weights produced by the algorithm are non-negative, and $w(x)$ is the largest weight in N_x. Thus, $w(e) = 0$ for all $e \in N_x$ and $w^2(N_x) = 0$. Moreover, (7) implies that $w^2_{(P_x,N_x)}(P_x) = 0$ as well, and so $w_{(P_x,N_x)}(e) = 0$ for all $e \in P_x$. The claim then follows.

Now, suppose that $w(x) \neq 0$, and so $w(x) > 0$. From (7), together with the fact that $x \in Y_e$ for all $e \in P_x$, we have:

$$w^2_{(P_x,N_x)}(P_x) \leq w^2(N_x) \leq w(x)^2 + \sum_{e \in P_x} w^2(Y_e - x) \ . \tag{8}$$

Rearranging (8) using $w^2_{(P_x,N_x)}(P_x) = \sum_{e \in P_x} w_{(P_x,N_x)}(e)^2$ we obtain:

$$\sum_{e \in P_x} \left[w_{(P_x,N_x)}(e)^2 - w^2(Y_e - x) \right] \leq w(x)^2 \ . \tag{9}$$

Applying Lemma 3 to each term on the left of (9) we have:

$$\sum_{e \in P_x} w(x) \cdot \left[2w_{(P_x,N_x)}(e) - w(Y_e) \right] \leq w(x)^2 \ . \tag{10}$$

Dividing by $w(x)$ (recall that $w(x) \neq 0$) then yields

$$\sum_{e \in P_x} \left[2w_{(P_x,N_x)}(e) - w(Y_e) \right] \leq w(x) \ . \qquad \blacktriangleleft$$

We now prove our main result, which gives an upper bound on the locality gap of Algorithm 2.

▶ **Theorem 5.** $\left(\frac{k+3}{2} + \epsilon\right) f(S) \geq f(O)$

Proof. Lemma 4 gives us one inequality for each $x \in S$. We now add all $|S|$ inequalities to obtain

$$\sum_{x \in S} \sum_{e \in P_x} \left[2w_{(P_x,N_x)}(e) - w(Y_e) \right] \leq \sum_{x \in S} w(x) \ . \tag{11}$$

We have $\sum_{x \in S} w(x) \leq f(S)$ by (3). Additionally, from (4), $f(S \cup P_x) - f(S) - |P_x|\alpha \leq \sum_{e \in P_x} w_{(P_x,N_x)}(e)$ for every P_x. Thus, (11) implies

$$2 \sum_{x \in S} \left[f(S \cup P_x) - f(S) - |P_x|\alpha \right] - \sum_{x \in S} \sum_{e \in P_x} w(Y_e) \leq f(S) \ . \tag{12}$$

Since P is a partition of O, (12) is equivalent to

$$2 \sum_{x \in S} \left[f(S \cup P_x) - f(S) \right] - 2|O|\alpha - \sum_{e \in O} w(Y_e) \leq f(S) \ . \tag{13}$$

We have $w(x) \geq 0$ for all $x \in S$, and there are at most k distinct e for which $x \in Y_e$, by property (K2) of Y. Thus

$$\sum_{e \in O} w(Y_e) \leq k \sum_{x \in S} w(x) \leq k f(S) \ ,$$

by (3). Combining this with (13), we obtain

$$2 \sum_{x \in S} \left[f(S \cup P_x) - f(S) \right] - 2|O|\alpha - k f(S) \leq f(S) \ . \tag{14}$$

Using again the fact that P is a partition of O, we can apply Lemma 2 to the remaining sum on the left of 14, yielding

$$2 \left[f(S \cup O) - f(S) \right] - 2|O|\alpha - k f(S) \leq f(S) \ ,$$

which simplifies to

$$f(S \cup O) - |O|\alpha \leq \frac{k+3}{2} f(S) \ . \tag{15}$$

From the definition of α and the optimality of O, we have

$$|O|\alpha \leq n\alpha = \delta f(S_{init}) \leq \delta f(O) \ .$$

Finally, since f is monotone, we have $f(O) \leq f(S \cup O)$. Thus, (15) implies

$$(1-\delta) f(O) \leq \frac{k+3}{2} f(S) \ ,$$

which is equivalent to $f(O) \leq \left(\frac{k+3}{2} + \epsilon\right) f(S)$ by the definition of δ. ◀

4.2 Runtime of Algorithm 2

Now, we consider the runtime of Algorithm 2. Each iteration requires time $O(n)$ to compute the weights for S, plus time to evaluate all potential k-replacements. There are $O(n^{k+k(k-1)+1}) = O(n^{k^2+1})$ such k-replacements (A, B), and each one can be evaluated in time $O(k^2)$, including the computation of the weights $w_{(A,B)}$. Thus, the total runtime of Algorithm 2 is $O(Ik^2 n^{k^2+1})$, where I is the number of improvements it makes. The main difficulty remaining in our analysis is showing that Algorithm 2 constantly improves some global quantity, and so I is bounded. Here, we show that although the weights w assigned to elements of the current solution S change at each iteration, the non-oblivious potential $w^2(S)$, is monotonically increasing.

▶ **Lemma 6.** *Suppose that Algorithm 2 applies a k-replacement (A, B) to solution S to obtain a new solution T. Let w_S be the weight function w determined by solution S and w_T be the weight function w determined by solution T. Then, $w_T^2(T) \geq w_S^2(S) + \alpha^2$.*

Proof. We first show that $w_S(s_i) \leq w_T(s_i)$ for each element $s_i \in S \setminus B$ and $w_{(A,B)}(a_i) \leq w_T(a_i)$ for any element $a_i \in A$. In the first case, let S_i (respectively, T_i) be the set of all elements in S (respectively T) that come before s_i and A_i be the set of all elements of A that come before a_i (in the ordering $\prec$). When the algorithm updates the solution S, it places all of A after $S \setminus B$, removes all elements of B from S, and leaves all elements of $S \setminus B$ in the same relative order. Thus, $T_i \subseteq S_i$. It follows directly from the submodularity of f that

$$w_S(x) = \left\lfloor \frac{f(S_i + s_i) - f(S_i)}{\alpha} \right\rfloor \alpha \leq \left\lfloor \frac{f(T_i + s_i) - f(T_i)}{\alpha} \right\rfloor \alpha = w_T(x) \ .$$

Let $w_{(A,B)}$ be the weight function for k-exchange (A, B) and current solution S. We now show that $w_{(A,B)}(a_i) \leq w_T(a_i)$ for each element $a_i \in A$. In this case, we let A_i be the set of all elements of A that come before a_i (in the ordering $\prec$) and T_i be the set of all elements of T that come before a_i after applying (A, B) to S. When the algorithm updates the solution S, it places all elements of A after all of $S \setminus B$, removes all elements of B from S, and leaves all elements of A in the same relative order. Thus, $T_i \subseteq (S \setminus B) \cup A_i$ and so from the submodularity of f

$$w_{(A,B)}(a_i) = \left\lfloor \frac{f((S \setminus B) \cup A_i + a_i) - f((S \setminus B) \cup A_i)}{\alpha} \right\rfloor \alpha \leq \left\lfloor \frac{f(T_i + a_i) - f(T_i)}{\alpha} \right\rfloor \alpha = w_T(a_i) \ .$$

Finally, since Algorithm 2 applied (A, B) to S, we must have $w^2_{(A,B)}(A) > w^2_S(B)$, and since all weights are integer multiples of α, we must in fact have $w^2_{(A,B)}(A) \geq w^2_S(B) + \alpha^2$. From this inequality, together with the above bounds on w_S and $w_{(A,B)}$, we have

$$w^2_S(S) = \sum_{x \in S \setminus B} w_S(x)^2 + \sum_{x \in B} w_S(x)^2 \leq \sum_{x \in S \setminus B} w_S(x)^2 + \sum_{y \in A} w_{(A,B)}(y)^2 + \alpha^2$$
$$\leq \sum_{x \in S \setminus B} w_T(x)^2 + \sum_{y \in A} w_T(y)^2 + \alpha^2 = w^2_T(T) + \alpha^2 \ . \quad \blacktriangleleft$$

▶ **Theorem 7.** *For any value $\epsilon \in (0, 1)$, Algorithm 2 makes at most $O(n^3 \epsilon^{-2})$ improvements.*

Proof. Because f is submodular, for any element e and any set $T \subseteq \mathcal{G}$, we have $f(T + e) - f(T) \leq f(\{e\}) \leq f(S_{init})$. In particular, for any solution $S \subseteq \mathcal{G}$ with associated weight function w, we have

$$w^2(S) = \sum_{e \in S} w(e)^2 \leq |S| f(S_{init})^2 \leq n f(S_{init})^2 \ .$$

Additionally, from Lemma 6, each improvement we apply must increase $w^2(S)$ by at least α^2, and hence the number of improvements that Algorithm 2 can make is at most

$$\frac{w^2(S) - f(S_{init})^2}{\alpha^2} \leq \frac{n f(S_{init})^2 - f(S_{init})^2}{\alpha^2}$$
$$= (n-1)\left(\frac{f(S_{init})}{\alpha}\right)^2 = (n-1)\frac{n^2}{\delta^2} = O(n^3 \epsilon^{-2}) \ . \quad \blacktriangleleft$$

▶ **Corollary 8.** *For any $\epsilon > 0$, Algorithm 2 is a $\frac{k+3}{2} + \epsilon$ approximation algorithm, running in time $O(\epsilon^{-2} k^2 n^{k^2+4})$.*

5 Open Questions

We do not currently have an example for which the locality gap of Algorithm 2 can be as bad as stated, even for specific k-exchange systems such as k-set packing. In the particular case of weighted independent set in $(k+1)$-claw free graphs, Berman [2] gives a tight example that shows his algorithm can return a set S with $\frac{k+1}{2} w(S) = w(O)$. His example uses only unit weights, and so the non-oblivious potential function is identical to the oblivious one. However, the algorithm of Feldman et al. (given here as Algorithm 1) considers a larger class of improvements than those considered by Berman, and so this example no longer applies, even in the linear case. For the unweighted variant, Hurkens and Schrijver [15] give a lower bound of $k/2 + \epsilon$, where ϵ depends on the size of the improvements considered. Because the non-oblivious local search routine performs the same as oblivious local search on instances with unit weights (since $1 = 1^2$), this lower bound applies to Algorithm 1 in the linear case. From a hardness perspective, the best known bound is the $\Omega(k/\ln k)$ NP-hardness result of Hazan, Safra, and Schwartz [14], for the special case of unweighted k-set packing.

Another interesting question is whether similar techniques can be adapted to apply to more general problems such as matroid k-parity in arbitrary matroids (here, even an improvement over k for the general linear case would be interesting) or to non-monotone submodular functions. A major difficulty with the latter generalization is our proof's dependence on the weights' non-negativity, as this assumption no longer holds if our approach is applied directly to non-monotone submodular functions.

Acknowledgments The author thanks Allan Borodin for providing comments on a preliminary version of this paper, as well as anonymous referees for providing further suggestions.

References

1. E. M. Arkin and R. Hassin. On local search for weighted k-set packing. In *Proc. of 5th ESA*, pages 13–22, 1997.
2. P. Berman. A d/2 approximation for maximum weight independent set in d-claw free graphs. *Nordic J. of Computing*, 7:178–184, Sept. 2000.
3. R. A. Brualdi. Induced matroids. *Proc. of the American Math. Soc.*, 29:213–221, 1971.
4. G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *Proc. of 12th IPCO*, pages 182–196, 2007.
5. B. Chandra and M. Halldórsson. Greedy local improvement and weighted set packing approximation. In *Proc. of 10th ACM-SIAM SODA*, pages 169–176, 1999.
6. U. Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45:634–652, July 1998.
7. U. Feige, V. S. Mirrokni, and J. Vondrak. Maximizing non-monotone submodular functions. In *Proc. of 48th IEEE FOCS*, pages 461–471, 2007.
8. M. Feldman, J. S. Naor, and R. Schwartz. Nonmonotone submodular maximization via a structural continuous greedy algorithm. In *Proc. of 38th ICALP*, pages 342–353, 2011.
9. M. Feldman, J. S. Naor, R. Schwartz, and J. Ward. Improved approximations for k-exchange systems. In *Proc. of 19th ESA*, pages 784–798, 2011.
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. S. O. Gharan and J. Vondrák. Submodular maximization by simulated annealing. In *Proc. of 22nd ACM-SIAM SODA*, pages 1098–1116, 2011.
12. A. Gupta, A. Roth, G. Schoenebeck, and K. Talwar. Constrained non-monotone submodular maximization: Offline and secretary algorithms. In *Proc. of 7th WINE*, pages 246–257, 2010.
13. M. M. Halldórsson. Approximating discrete collections via local improvements. In *Proc. of 6th ACM-SIAM SODA*, pages 160–169, 1995.
14. E. Hazan, S. Safra, and O. Schwartz. On the complexity of approximating k-set packing. *Computational Complexity*, 15:20–39, May 2006.
15. C. A. J. Hurkens and A. Schrijver. On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems. *SIAM J. Discret. Math.*, 2(1):68–72, 1989.
16. S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. In *Proc. of 35th IEEE FOCS*, pages 819–830, 1994.
17. J. Lee, V. S. Mirrokni, V. Nagarajan, and M. Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *Proc. of 41st ACM STOC*, pages 323–332, 2009.
18. J. Lee, M. Sviridenko, and J. Vondrák. Matroid matching: the power of local search. In *Proc. of 42nd ACM STOC*, pages 369–378, 2010.
19. J. Lee, M. Sviridenko, and J. Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Math. of Oper. Res.*, 35(4):795 –806, Nov. 2010.
20. J. A. Soto. A simple PTAS for weighted matroid matching on strongly base orderable matroids. *Electronic Notes in Discrete Mathematics*, 37:75–80, Aug. 2011.