# Approximating the unweighted $k$-set cover problem: greedy meets local search

Asaf Levin[1]

Department of Statistics, The Hebrew University, Jerusalem, Israel.
levinas@mscc.huji.ac.il

**Abstract.** In the unweighted set-cover problem we are given a set of elements $E = \{e_1, e_2, \ldots, e_n\}$ and a collection $\mathcal{F}$ of subsets of $E$. The problem is to compute a sub-collection $SOL \subseteq \mathcal{F}$ such that $\bigcup_{S_j \in SOL} S_j = E$ and its size $|SOL|$ is minimized. When $|S| \leq k$ for all $S \in \mathcal{F}$ we obtain the unweighted $k$-set cover problem. It is well known that the greedy algorithm is an $H_k$-approximation algorithm for the unweighted $k$-set cover, where $H_k = \sum_{i=1}^{k} \frac{1}{i}$ is the $k$-th harmonic number, and that this bound on the approximation ratio of the greedy algorithm, is tight for all constant values of $k$. Since the set cover problem is a fundamental problem, there is an ongoing research effort to improve this approximation ratio using modifications of the greedy algorithm. The previous best improvement of the greedy algorithm is an $\left(H_k - \frac{1}{2}\right)$-approximation algorithm. In this paper we present a new $\left(H_k - \frac{196}{390}\right)$-approximation algorithm for $k \geq 4$ that improves the previous best approximation ratio for all values of $k \geq 4$. Our algorithm is based on combining local search during various stages of the greedy algorithm.

## 1 Introduction

In the WEIGHTED SET-COVER PROBLEM we are given a set of elements $E = \{e_1, e_2, \ldots, e_n\}$ and a collection $\mathcal{F}$ of subsets of $E$, where $\cup_{S \in \mathcal{F}} S = E$ and each $S \in \mathcal{F}$ has a positive cost $c_S$. The goal is to compute a sub-collection $SOL \subseteq \mathcal{F}$ such that $\bigcup_{S \in SOL} S = E$ and its cost $\sum_{S \in SOL} c_S$ is minimized. Such a sub-collection of subsets is called a *cover*. When we consider instances of the WEIGHTED SET-COVER such that each $S_j$ has at most $k$ elements ($|S| \leq k$ for all $S \in \mathcal{F}$), we obtain the WEIGHTED $k$-SET COVER PROBLEM. The UNWEIGHTED SET COVER PROBLEM and the UNWEIGHTED $k$-SET COVER PROBLEM are the special cases of the WEIGHTED SET COVER and of WEIGHTED $k$-SET COVER, respectively, where $c_S = 1 \ \forall S \in \mathcal{F}$.

It is well known (see [2]) that a greedy algorithm is an $H_k$-approximation algorithm for the weighted $k$-set cover, where $H_k = \sum_{i=1}^{k} \frac{1}{i}$ is the $k$-th harmonic number, and that this bound is tight even for the unweighted $k$-set cover problem (see, [12, 15]). For unbounded values of $k$, Slavík [19] showed that the approximation ratio of the greedy algorithm for the unweighted set cover problem is $\ln n - \ln \ln n + \Theta(1)$. Feige [5] proved that unless $NP \subseteq DTIME(n^{polylog\ n})$ the unweighted set cover problem cannot be approximated within a factor $(1-\epsilon)\ln n$, for any $\epsilon > 0$. Raz and Safra [18] proved that if $P \neq NP$ then for some constant $c$, the unweighted set cover problem cannot be approximated within a factor $c \log n$. This result shows that the greedy algorithm is an asymptotically best possible approximation algorithm for the weighted and unweighted set cover problem (unless $NP \subseteq DTIME(n^{polylog\ n})$). The unweighted $k$-set cover problem is known to be NP-complete [13] and MAX SNP-hard for all $k \geq 3$ [3, 14, 16]. Another algorithm for the weighted set cover problem by Hochbaum [10] has an approximation ratio that depends on the maximum number of subsets that contain any given item (the local-ratio algorithm of Bar-Yehuda and Even [1] has the same performance guarantee). See Paschos [17] for a survey on these results.

In spite of the above bad news Goldschmidt, Hochbaum and Yu [7] modified the greedy algorithm for the unweighted set cover and showed that the resulting algorithm has a performance guarantee of $H_k - \frac{1}{6}$. Halldórsson [8] presented an algorithm based on local search that has an approximation ratio of $H_k - \frac{1}{3}$ for the unweighted $k$-set cover, and a $(1.4 + \epsilon)$-approximation algorithm for the unweighted 3-set cover. Duh and Fürer [4] further improved this result and presented an $(H_k - \frac{1}{2})$-approximation algorithm for the unweighted $k$-set cover. We will base our algorithm on the algorithm of Duh and Fürer [4], and therefore we will review their algorithm and results in Section 3. All of these improvements [7, 8, 4] are based on running the greedy algorithm until each new subset covers at most $t$ new elements (where $t = 2$ in [7] and larger values of $t$ in [8, 4]) and then switch to another algorithm.

Regarding approximation algorithms for the weighted $k$-set cover problem within a factor better than $H_k$, a first improvement step was given by Fujito and Okumura [6] who presented $H_k - \frac{1}{12}$-approximation algorithm for the $k$-set cover problem where the cost of each subset is either 1 or 2. More recently, Hassin and Levin [9] provided an $\left(H_k - \frac{k-1}{8k^9}\right)$-approximation algorithm.

The MAXIMUM SET PACKING PROBLEM is the following related problem: We are given a set of elements $E = \{e_1, e_2, \ldots, e_n\}$ and a collection $\mathcal{F}$ of subsets of $E$, where $\cup_{S\in\mathcal{F}} S = E$, and the goal is to compute a maximum size set packing, i.e., a sub-collection $\mathcal{F}' \subseteq \mathcal{F}$ of disjoint subsets. Hurkens and Schrijver [11] proved that a local-search algorithm for the maximum set packing problem where each subset in $\mathcal{F}$ has at most $k$ elements, is a $\left(\frac{2}{k} - \epsilon\right)$-approximation algorithm. Therefore, this local-search algorithm has a better performance guarantee than the greedy selection rule that returns any maximal sub-collection. The greedy selection rule has an approximation ratio of $\frac{1}{k}$.

We first observe that w.l.o.g., there is an optimal solution to the set cover problem such that each element is covered by exactly one subset of the optimum: Let an optimal solution to the problem consist of a collection of sets $S_j^*$, $j \in J^*$, with $\cup_{j\in J^*} S_j^* = E$. We now construct another optimal solution formed of element-disjoint sets $S_j'$ where $S_j' \subseteq S_j^*$ for all $j \in J^*$. To do that, we assign each element $e \in E$ to the smallest index set $S_j^*$, $j \in J^*$ that contains $e$. We modify the instance by adding the sets $S_j'$ for all $j$ to the collection $\mathcal{F}$. If the algorithm or the optimal solution decides to pick such a set $S'_j$, we interpret this as picking the set $S^*_j$. Henceforth, any optimal solution will be considered to have this disjointness property, so each $e \in E^*$ belongs to exactly one set $S^*_j$. We consider an optimal solution $OPT$ that satisfies the disjointness property.

We define a *$j$-set* to be a set with $j$ elements. We fix an optimal solution $OPT$, and we say that a $k$-set is an *optimal $k$-set* if it is contained in $OPT$.

**Paper overview:** In Section 2 we review the greedy algorithm for the unweighted minimum $k$-set cover problem, and its analysis. In Section 3 we review the semi-local optimization algorithm of [4]. In Section 4 we present our improved algorithm. We analyze its performance in Section 5, i.e., we show that our algorithm is an $\left(H_k - \frac{196}{390}\right)$-approximation algorithm for the unweighted $k$-set cover problem where $k \geq 4$, improving the earlier $\left(H_k - \frac{1}{2}\right)$-approximation algorithm of [4]. We conclude in Section 6 by discussing open questions.

## 2 The greedy algorithm

In this section we review the greedy algorithm for the unweighted $k$-set cover problem and the proof of its performance guarantee.

The greedy algorithm starts with an empty collection of subsets in the solution and no item being covered. Then, it iterates the following procedure until all items are covered:

Let $w_S$ be the number of uncovered items in a set $S \in \mathcal{F}$, and the current *ratio of* $S$ is $r_S = \frac{1}{w_S}$. Let $S^*$ be a set such that $r_{S^*}$ is minimized. The algorithm adds $S^*$ to the collection of subsets of the solution, defines the items of $S^*$ as covered, and assigns a *price* of $r_{S^*}$ to all the items that are now covered but were uncovered prior to this iteration (i.e., the items that were first covered by $S^*$).

Johnson [12], Lovász [15] and Chvátal [2] showed that the greedy algorithm is an $H_k$-approximation algorithm for the unweighted $k$-set cover.

Chvátal's proof is the following: first note that the cost of the greedy solution equals the sum of prices assigned to the items. Second, consider a set $S$ that belongs to an optimal solution $OPT$. Then, $OPT$ pays 1 for $S$. When the $i$-th item of $S$ is covered by the greedy algorithm, the algorithm could choose $S$ as a feasible set with a current ratio of $\frac{1}{|S|-i+1}$. Therefore, the price assigned to the this item is at most $\frac{1}{|S|-i+1}$. It follows that the total price assigned to the items of $S$ is at most $\sum_{i=1}^{|S|} \frac{1}{|S|-i+1} = \sum_{i'=1}^{|S|} \frac{1}{i'} \leq H_k$, and therefore, the approximation ratio of the greedy algorithm is at most $H_k$.

## 3 The semi-local optimization algorithm

Duh and Fürer [4] suggested the following procedure to approximate the unweighted 3-set cover problem. In a pure local improvement step, we replace a number of sets with fewer sets to form a new cover with a reduced cost. To define a semi-local step, they observed that once the 3-sets are selected the remaining instance can be solved optimally in polynomial time. Thus a local change in the

3-sets allows any global changes in the 2-sets and 1-sets and such a change is called a *semi-local change*.

They allowed the algorithm to remove one 3-set and insert at most a pair of 3-sets if one of the following happens: either the total cost is reduced, or the total cost remains the same and the number of 1-sets in the resulting solution is reduced (thus the total cost is the primary objective whereas the number of 1-sets is a secondary objective). This results in the approximation algorithm for the unweighted 3-set cover of [4]. They showed that this is a $\frac{4}{3}$-approximation algorithm. More precisely, the following proposition was proved in [4].

**Proposition 1.** *Assume that an optimal solution for the unweighted 3-set cover instance has $b_1$, $b_2$, and $b_3$ 1-sets, 2-sets and 3-sets, respectively. Then the solution that the semi-local optimization algorithm returns, costs at most $b_1+b_2+\frac{4}{3}b_3$. Moreover, the number of 1-sets in the solution that the algorithm returns, is at most $b_1$.*

In order to extend their result to larger values of $k$, they suggested the following algorithm:

1. **Greedy Phase** For $j = k$ down to 6 do:
   greedily choose a maximal collection of $j$-sets.
2. **Restricted Phase** For $j = 5$ down to 4 do:
   choose a maximal collection of $j$-sets with the restriction that the choice of these $j$-sets does not increase the number of 1-sets.
3. **Semi-local Optimization Phase** Run the semi-local optimization algorithm on the remaining instance.

Note that the following question is answered within polynomial time during the Restricted phase: Does the addition of a $j$-set $S$ to the current solution increase the number of 1-sets in the resulting solution (returned by the algorithm)? Duh and Fürer proved that this algorithm is an $\left(H_k - \frac{1}{2}\right)$-approximation, and they also showed that this bound is tight for the semi-local optimization algorithm.

## 4 The algorithm

In this section we present our modification of the semi-local optimization algorithm where we use a local-search algorithm during the phase where each new set covers exactly four previously uncovered elements.

**Algorithm A:**

1. **Greedy Phase** For $j = k$ down to 6 do:
   greedily choose a maximal collection of disjoint $j$-sets (each covering exactly $j$ new elements).
2. **Restricted Phase** Choose a maximal collection of disjoint 5-sets with the restriction that the choice of these 5-sets does not increase the number of 1-sets.

3. **Local-Search Phase** Choose a collection of disjoint 4-sets such that the choice of these 4-sets does not increase the number of 1-sets and this collection has a local maximum size. The requirement of local maximum size means that removing a 4-set from this collection does not allow us to add at least a pair of 4-sets (without increasing the number of 1-sets).
4. **Semi-Local Optimization Phase** Run the semi-local optimization algorithm on the remaining instance.

In Phase 3 we are using local-search whose neighborhood is defined by removing one 4-set and inserting at least a pair of 4-sets as long as the number of 1-sets in the returned solution does not increase. The use of this local-search procedure is motivated by the approximation algorithm of [11] for the maximum set packing problem. This improved phase is the corner stone on which our improved approximation ratio is based.

Each iteration takes polynomial time because checking whether the number of 1-sets in the resulting solution increases, takes polynomial time. Therefore, Algorithm A is a polynomial time algorithm that returns a feasible solution. Therefore, we establish the following lemma:

**Lemma 1.** *For every value of $k$, Algorithm A returns a feasible solution in polynomial time.*

In the next section we analyze the performance guarantee of Algorithm A.

## 5 The analysis of Algorithm A

In this section we analyze the performance guarantee of Algorithm A. We consider an optimal solution $OPT$, and bound the performance guarantee of A. Recall that we assume that $OPT$ is a partition of the element set $E$. We now further characterize the structure of $OPT$.

**Lemma 2.** *W.l.o.g., each set of $OPT$ is a $k$-set.*

*Proof.* Assume that the claim does not hold on an instance $I$. We create a new instance $I'$ such that the optimal solution $OPT'$ for $I'$ costs $k$ times the cost of $OPT$, and the solution returned by A on $I'$ costs more than $k$ times the solution returned by algorithm A on $I$, and we will conclude that if there is a bad example for the algorithm there is a bad example for the algorithm that shows the same approximation ratio such that the property of the lemma holds.

To do so we first take $k$ disjoint copies of the instance $I$. Clearly, the optimal solution $\hat{OPT}$ for this new instance costs exactly $k$ times the cost of $OPT$, and it is a union of $k$ copies of $OPT$. Then, we add new elements to $\hat{OPT}$'s existing sets so that each set in this sub-collection is a $k$-set. Note that the number of the new elements is divisible by $k$. Last, we add new $k$-sets of these new elements, such that the algorithm picks these new $k$-sets (of the new elements) in its first steps, and then continue like it acts on $I$ on each of the $k$ copies of $I$. Therefore, $OPT'$ costs exactly $k$ times the cost of $OPT$ (we can use the sets of $\hat{OPT}$ that we increased), however the cost of the solution returned by A on $I'$ is strictly larger than $k$ times the cost of the solution returned by A on $I$. $\square$

Next, we allocate a price for each element in the following way:

– For an element that is covered by an $i$-set during Phases 1, 2 and 3 of Algorithm A, we allocate a price of $\frac{1}{i}$.
– We consider special 2-sets and 3-sets that are named *sibling 2-sets* defined as follows (see [4] for introduction of this term): a 2-set or a 3-set $S$ chosen by the semi-local optimization phase such that one of the elements in $S$ is the last uncovered element of an optimal $k$-set (this element is called the *primary element*) and the remaining elements of $S$ belong to a common optimal $k$-set (i.e., to the same set in the optimal solution). The elements of a sibling 2-set that are not primary are called *secondary elements.* A sibling 2-set is the result of the fact that the Semi-Local Optimization phase does not create a new singleton, and therefore, if an optimal $k$-set has $k-1$ covered elements at the end of Phase 3 out of which at least one is covered during Phases 2 or 3, then the last element belongs to at least a 2-set (and is not a singleton). We allocate a cost of $\alpha = \frac{4}{5}$ for the primary element of a sibling 2-set, and for each of its secondary elements we allocate a cost of $1 - \alpha = \frac{1}{5}$.
– For the other elements that we cover during Phase 4, we assign at most a unit price for each selected set such that the following holds (such an allocation of prices is feasible according to Proposition 1):
  - For each three elements that belong to a common $k$-set of $OPT$, are covered during Phase 4, and do not intersect with a sibling 2-set, we assign a total price of $\frac{4}{3}$.
  - For each pair of elements that belong to a common $k$-set of $OPT$, are covered during Phase 4, and do not intersect with a sibling 2-set, we assign a total price of one unit.

By the allocation of the prices and Proposition 1, we conclude the following lemma:

**Lemma 3.** *The cost of the solution returned by Algorithm A is at most the total price of all the elements.*

Next, we define a *bad set.* Given an optimal $k$-set $S$, if $k \geq 5$, then $S$ is a *bad set* if at the end of Phase 1 $S$ has exactly five uncovered elements from which exactly one element is covered during Phase 3 and one of the following holds: Either exactly one element of $S$ is covered during Phase 2 and none of the three remaining elements belongs to a sibling 2-set, or none of the elements of $S$ is covered during Phase 2 and exactly one element of $S$ belongs to a sibling 2-set. If $k = 4$, then $S$ is a bad set if exactly one of its elements is covered during Phase 3. An optimal $k$-set that is not bad is a *good set.*

The outline of the proof of our improved approximation ratio is as follows: in Lemma 4 we will prove that the total price of an optimal set is better than $H_k - \frac{1}{2}$ if the optimal set is good and it equals $H_k - \frac{1}{2}$ for bad sets. Afterwards, in Lemma 5 we will show that there is a constant proportion of the optimal sets that have to be good sets. Combining the two results together we will establish our improved approximation ratio.

**Lemma 4.** *Assume that $k \geq 4$. The total price assigned to an optimal bad $k$-set is at most $\rho_b = H_k - \frac{1}{2}$. The total price assigned to an optimal good $k$-set is at most $\rho_g = H_k - \frac{16}{30}$.*

*Proof.* Let $S$ be an optimal $k$-set. Denote by $price(S)$ the total price assigned to the elements of $S$. First assume that $S$ is a bad set. If $k \geq 5$, then the $j$-th covered element from $S$ during Phase 1 is assigned a price of at most $\frac{1}{k-j+1}$, the element that is covered during Phase 2 is assigned a price of $\frac{1}{5}$ (if it exists), the element that belongs to a sibling 2-set is assigned a price of $\frac{1}{5}$ (if it exists), the element that is covered during Phase 3 is assigned a price of $\frac{1}{4}$, and the remaining three elements are assigned a total price of at most $\frac{4}{3}$. Hence, $price(S) \leq \sum_{i=6}^{k} \frac{1}{i} + \frac{1}{5} + \frac{1}{4} + \frac{4}{3} = H_k - \frac{1}{2} = \rho_b$. If $k = 4$, then $S$ has a single element covered during Phase 3 that pays a price of $\frac{1}{4}$ and the three remaining elements are assigned a total price of at most $\frac{4}{3}$. So again $price(S) \leq H_k - \frac{1}{2} = \rho_b$.

It remains to prove the second part of the lemma regarding the total price of a good set. So assume that $S$ is a good set. We denote by $N_g$ the number of elements of $S$ that remains uncovered at the end of Phase 1. We denote by $N_r$ ($N_l$) the number of elements of $S$ that are covered during Phase 2 (Phase 3). Our proof is based on a detailed case analysis.

First assume that $k = 4$. Then, the Greedy phase and the Restricted phase do not select sets, and therefore $N_g = 4$ and $N_r = 0$.

- Assume that $N_l = 4$. Then, each element of $S$ is covered during Phase 3 and pays a price of $\frac{1}{4}$. Therefore, $price(S) = 1 < H_4 - \frac{16}{30} = \rho_g$.
- Assume that $N_l = 3$. Then, each element of $S$ that is covered during Phase 3 pays a price of $\frac{1}{4}$, and the remaining element pays a price of at most $\frac{4}{5}$ (this is because since no singletons are created, this remaining element either belongs to a sibling 2-set and then it pays at most $\frac{4}{5}$, or it belongs to a 3-set and in this case it pays $\frac{1}{3}$). Therefore, $price(S) \leq \frac{3}{4} + \frac{4}{5} = \frac{93}{60} = \frac{125}{60} - \frac{32}{60} = H_4 - \frac{16}{30} = \rho_g$.
- Assume that $N_l = 2$. Then, each element of $S$ that is covered during Phase 3 pays a price of $\frac{1}{4}$. The two remaining elements pay a total price of at most 1. Thus, $price(S) \leq \frac{3}{2} < \rho_g$.
- Assume that $N_l = 1$. Then, the element of $S$ that is covered during Phase 3, pays a price of $\frac{1}{4}$. Since $S$ is a good set, it contains at least one element that belongs to a sibling 2-set that pays $\frac{1}{5}$ (since $N_l = 1$, it is not the primary element). The other two elements of $S$ have total price of at most 1. Therefore, $price(S) \leq \frac{1}{4} + \frac{1}{5} + 1 = \frac{87}{60} < \frac{93}{60} = \rho_g$.
- Assume that $N_l = 0$. Since $S$ is not added to the solution during Phase 3, it must contain an element that belongs to a sibling 2-set, and pays a price of $\frac{1}{5}$. The other three elements pay a total price of at most $\frac{4}{3}$ (this is also the case if some of them belong to sibling 2-sets). Therefore, $price(S) \leq \frac{1}{5} + \frac{4}{3} = \frac{92}{60} < \rho_g$

It remains to consider the case where $k \geq 5$. First note that by the greedy selection rule during the greedy phase, we conclude that $N_g \leq 5$. Moreover, the $j$-th covered element from $S$ during the greedy phase (for $1 \leq j \leq k-5$) is assigned a price of at most $\frac{1}{k-j+1}$. Moreover, since the Restricted phase and the Local-Search phase do not create new singletons, we conclude that if $N_g \geq 2$, then the maximum price of an element of $S$ is at most $\frac{4}{5}$.

- Assume that $N_g \leq 2$. Then, the $k-4$-th, the $k-3$-rd, and the $k-2$-nd covered elements from $S$ are covered during Phase 1, and therefore assigned a price of at most $\frac{1}{6}$ for each. The last two elements of $S$ are assigned a total price of at most $\frac{4}{5}+\frac{1}{4}$ (this is the case where one of them is covered during Phase 3 and the last element is from sibling 2-set, and the other cases cause a smaller cost). Therefore, $\mathit{price}(S) \leq H_k - H_5 + \frac{3}{6} + \frac{4}{5} + \frac{1}{4} = H_k - H_5 + \frac{31}{20} = H_k - \frac{137}{60} + \frac{93}{60} = H_k - \frac{44}{60} < \rho_g$.
- Assume that $N_g = 3$. Then, the $k-4$-th and the $k-3$-rd covered elements from $S$ are covered during Phase 1, and therefore assigned a price of at most $\frac{1}{6}$ for each.
  - If $N_r + N_l = 0$, then the last three elements of $S$ are covered during Phase 4, and pay a total price of at most $\frac{4}{3}$. Therefore, $\mathit{price}(S) \leq H_k - H_5 + \frac{2}{6} + \frac{4}{3} = H_k - \frac{137}{60} + \frac{5}{3} = H_k - \frac{37}{60} < \rho_g$.
  - If $N_r + N_l = 1$, then the last two elements of $S$ are covered during Phase 4, and pay a total price of at most 1. The $k-2$-nd element of $S$ is covered during either Phase 2 or Phase 3, and so it pays a price of at most $\frac{1}{4}$. Therefore, the last three elements of $S$ pay a total price of at most $\frac{5}{4} < \frac{4}{3}$ and again $\mathit{price}(S) < \rho_g$.
  - If $N_r + N_l = 2$, then the last uncovered element pays at most $\frac{4}{5}$ (if it belongs to a sibling 2-set, and otherwise it pays less). The $k-2$-nd and the $k-1$-st covered elements from $S$ are covered during either Phase 2 or Phase 3, and therefore each of these is assigned a price of at most $\frac{1}{4}$. Again the last three elements of $S$ pay at most $\frac{4}{5} + \frac{2}{4} < \frac{4}{3}$, and therefore $\mathit{price}(S) < \rho_g$.
  - If $N_r + N_l = 3$, then each of the last three elements of $S$ pays a price of at most $\frac{1}{4}$, and in total they pay less than $\frac{4}{3}$. Therefore, $\mathit{price}(S) < \rho_g$.
- Assume that $N_g = 4$. Then, the $k-4$-th covered element from $S$ is covered during Phase 1, and therefore pays a price of at most $\frac{1}{6}$. Among the last four elements of $S$ there is at least one element that pays at most $\frac{1}{4}$. To see this fact note that if none of the elements of $S$ belong to a set that is chosen during Phase 2 or Phase 3, then $S$ has an element that belongs to a sibling 2-set (otherwise, we add $S$ to the solution during Phase 3 contradicting the maximality of the collection that we choose during Phase 3), and in each of these cases the element pays at most $\frac{1}{4}$. The other three elements pay at most $\max\{\frac{4}{3}, \frac{4}{5} + \frac{2}{4}\} = \frac{4}{3}$. Therefore, $\mathit{price}(S) \leq H_k - H_5 + \frac{1}{6} + \frac{1}{4} + \frac{4}{3} = H_k - \frac{137}{60} + \frac{105}{60} = H_k - \frac{32}{60} = \rho_g$.
- Assume that $N_g = 5$.
  - Assume that $N_r = N_l = 0$. By the maximality of the sets chosen during Phase 3, we conclude that $S$ has at least two elements that belong to

sibling 2-sets, and therefore each of these pays $\frac{1}{5}$. The other three elements of $S$ pay at most $\frac{4}{3}$. Therefore, $price(S) \leq H_k - H_5 + \frac{2}{5} + \frac{4}{3} = H_k - \frac{137}{60} + \frac{104}{60} = H_k - \frac{33}{60} < \rho_g$.

- Assume that $N_r = 1$ and $N_l = 0$. The element of $S$ that is covered during Phase 2, pays a price of $\frac{1}{5}$. By the maximality of the sets chosen during Phase 3, we conclude that $S$ has an element that belongs to a sibling 2-set and pays $\frac{1}{5}$. The remaining three elements pay at most $\frac{4}{3}$. Therefore, $price(S) \leq H_k - H_5 + \frac{1}{5} + \frac{1}{5} + \frac{4}{3} = H_k - \frac{33}{60} < \rho_g$.
- Assume that $N_r \geq 2$. The elements of $S$ that are covered during Phase 2 pay a price of $\frac{1}{5}$ each. The last three elements pay a total price of at most $\frac{4}{3}$. Therefore, $price(S) \leq H_k - H_5 + \frac{2}{5} + \frac{4}{3} = H_k - \frac{33}{60} < \rho_g$.
- Assume that $N_r \leq 1$ and $N_l = 1$. Since $S$ is a good set, we conclude that either $N_r = 1$ and $S$ has an element that belongs to a sibling 2-set, or $S$ has at least two elements that belong to sibling 2-sets. The element of $S$ that is covered during Phase 2 (if it exists) pays a price of $\frac{1}{5}$, the element of $S$ that is covered during Phase 3 pays a price of $\frac{1}{4}$, and each element of $S$ that belongs to a sibling 2-set pays $\frac{1}{5}$. The two last remaining elements have a total price of at most 1. Therefore, $price(S) \leq H_k - H_5 + \frac{1}{5} + \frac{1}{4} + \frac{1}{5} + 1 = H_k - \frac{137}{60} + \frac{99}{60} = H_k - \frac{38}{60} < \rho_g$.
- Assume that $N_r \leq 1$ and $N_l = 2$. By the maximality of the sets that we choose during Phase 2, we conclude that if $N_r = 0$ then $S$ has an element that belongs to a sibling 2-set and pays $\frac{1}{5}$. Therefore, $S$ has an element that pays $\frac{1}{5}$ (this is the one that is covered during Phase 2, or the one that belongs to a sibling 2-set). Each of the elements of $S$ that is covered during Phase 3, pays a price of $\frac{1}{4}$. The two remaining elements pay a total price of 1. Therefore, $price(S) \leq H_k - H_5 + \frac{1}{5} + \frac{2}{4} + 1 = H_k - \frac{137}{60} + \frac{102}{60} = H_k - \frac{35}{60} < \rho_g$.
- Assume that $N_r \leq 1$ and $N_l = 3$. By the maximality of the sets that we choose during Phase 2, we conclude that if $N_r = 0$, then $S$ has an element that belongs to a sibling 2-set and pays $\frac{1}{5}$. Therefore, $S$ has an element that pays $\frac{1}{5}$ (this is the one that is covered during Phase 2, or the one that belongs to a sibling 2-set). Each of the elements of $S$ that is covered during Phase 3, pays a price of $\frac{1}{4}$. The remaining element pays at most $\frac{4}{5}$. Therefore, $price(S) \leq H_k - H_5 + \frac{1}{5} + \frac{3}{4} + \frac{4}{5} = H_k - \frac{137}{60} + \frac{105}{60} = H_k - \frac{32}{60} = \rho_g$.
- Assume that $N_r \leq 1$ and $N_l = 4$. By the maximality of the sets that we choose during Phase 2, we conclude that if $N_r = 0$, then $S$ has an element that belongs to a sibling 2-set and pays $\frac{1}{5}$. Therefore, $S$ has an element that pays $\frac{1}{5}$ (this is the one that is covered during Phase 2, or the one that belongs to a sibling 2-set). Each of the elements of $S$ that is covered during Phase 3, pays a price of $\frac{1}{4}$. Therefore, $price(S) \leq H_k - H_5 + \frac{1}{5} + \frac{4}{4} = H_k - \frac{137}{60} + \frac{72}{60} = H_k - \frac{65}{60} < \rho_g$.

□

Denote by $n_b$ the number of bad sets in $OPT$ and by $n_g$ the number of good sets in $OPT$.

**Lemma 5.** $n_b \leq 12n_g$.

*Proof.* Consider a bad set $S$ in $OPT$. At the beginning of Phase 3, $S$ has four uncovered elements such that none of these belong to a sibling 2-set. Since $S$ is a bad set we cover exactly one of its elements during Phase 3. Consider a set $S'$ chosen in Phase 3. Then, there is a good set $S'' \in OPT$ such that $S'' \cap S' \neq \emptyset$. To see this note that otherwise during Phase 3 we could replace $S'$ by the bad sets it intersects (each such set has four elements that consist of a 4-set that we could add to the solution without increasing the number of singletons). Since we did not apply this step, we conclude that at least one of its intersecting sets from $OPT$ is a good set.

A good set $S \in OPT$ can intersect at most four sets that we choose during Phase 3. These four sets can intersects at most 12 other sets of $OPT$. These 12 sets might be bad sets. Therefore, the claim follows. $\square$

**Theorem 1.** *Algorithm A is a $\left(H_k - \frac{196}{390}\right)$-approximation algorithm for the unweighted $k$-set cover problem.*

*Proof.* By Lemma 1, the algorithm returns a feasible solution in polynomial time. It remains to establish its approximation ratio.

$$\begin{aligned}
A &\leq n_g \cdot \rho_g + n_b \cdot \rho_b \\
&= n_g \cdot \left(H_k - \frac{16}{30}\right) + n_b \cdot \left(H_k - \frac{1}{2}\right) \\
&\leq (n_g + n_b) \cdot \left[\frac{1}{13} \cdot \left(H_k - \frac{16}{30}\right) + \frac{12}{13} \cdot \left(H_k - \frac{1}{2}\right)\right] \\
&= OPT \cdot \left[\frac{1}{13} \cdot \left(H_k - \frac{16}{30}\right) + \frac{12}{13} \cdot \left(H_k - \frac{1}{2}\right)\right] \\
&= OPT \cdot \left(H_k - \frac{196}{390}\right),
\end{aligned}$$

where the first inequality follows by Lemma 3, the first equation follows by Lemma 4, the second inequality follows by Lemma 5, the second equation follows because the cost of $OPT$ is exactly $n_b + n_g$, and the last equation follows by simple algebra. $\square$

## 6 Concluding remarks

In this paper we addressed the fundamental problem of unweighted $k$-set cover problem, and introduced an improvement over the previously best known algorithm for all values of $k$ such that $k \geq 4$. Although we obtain a small improvement over the algorithm of Duh and Fürer [4], we think that our analysis is not tight and the approximation ratio of our algorithm can be improved. Improving the analysis of our Algorithm A is left for future research.

In this paper we showed that incorporating a local-search procedure in various stages of the greedy algorithm instead of only where each set has at most

three uncovered elements, provides a better approximation ratio. We conjecture that incorporating local-search procedures in each greedy phase decreases the approximation ratio further. Such an algorithm replaces the Greedy phase by the following phase:
**Improved phase:** For $j = k, k-1, k-2, \ldots, 6$ do: apply local-search to choose an approximated maximum size collection of $j$-sets (each covering exactly $j$ new elements).
It is easily noted that using the Improved phase instead of the Greedy phase in Algorithm A does not harm the approximation ratio of the resulting algorithm. We leave the analysis of this improved algorithm for future research.

## References

1. R. Bar-Yehuda and S. Even, "A linear time approximation algorithm for the weighted vertex cover problem," *Journal of Algorithms*, **2**, 198-203, 1981.
2. V. Chvátal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, **4**, 233-235, 1979.
3. P. Crescenzi and V. Kann, "A compendium of NP optimization problems", http://www.nada.kth.se/theory/problemlist.html, 1995.
4. R. Duh and M. Fürer, "Approximation of $k$-set cover by semi local optimization," *Proc. STOC 1997*, 256-264, 1997.
5. U. Feige, "A threshold of $\ln n$ for approximating set cover", *Journal of the ACM*, **45**, 634-652, 1998.
6. T. Fujito and T. Okumura, "A modified greedy algorithm for the set cover problem with weights 1 and 2," *Proc. ISAAC 2001*, 670-681, 2001.
7. O. Goldschmidt, D. S. Hochbaum and G. Yu, "A modified greedy heuristic for the set covering problem with improved worst case bound," *Information Processing Letters*, **48**, 305-310, 1993.
8. M. M. Halldórsson, "Approximating $k$ set cover and complementary graph coloring," *Proc. IPCO 1996*, 118-131, 1996.
9. R. Hassin and A. Levin, " A better-than-greedy approximation algorithm for the minimum set cover problem," *SIAM J. Computing*, **35**, 189-200, 2006.
10. D. S. Hochbaum, "Approximation algorithms for the weighted set covering and node cover problems," *SIAM Journal on Computing*, **11**, 555-556, 1982.
11. 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 Journal on Discrete Mathematics*, **2**, 68-72, 1989.
12. D. S. Johnson, "Approximation algorithms for combinatorial problems," *Journal of Computer and System Sciences*, **9**, 256-278, 1974.
13. R. M. Karp, "Reducibility among combinatorial problems," Complexity of computer computations (R.E. Miller and J.W. Thatcher, eds.), Plenum Press, New-York, 1972, 85-103.
14. S. Khanna, R. Motwani, M. Sudan and U. V. Vazirani, "On syntactic versus computational views of approximability," *SIAM Journal on Computing*, **28**, 164-191, 1998.
15. L. Lovász, "On the ratio of optimal integral and fractional covers," *Discrete Mathematics*, **13**, 383-390, 1975.
16. C. H. Papadimitriou and M. Yannakakis, "Optimization, approximation and complexity classes," *Journal of Computer System Sciences*, **43**, 425-440, 1991.

17. V. T. Paschos, "A survey of approximately optimal solutions to some covering and packing problems," *ACM Computing Surveys*, **29**, 171-209, 1997.
18. R. Raz and S. Safra, "A sub-constant error-probability low-degree test, and sub-constant error-probability PCP characterization of NP", *Proc. STOC 1997*, 475-484, 1997.
19. P. Slavík, "A tight analysis of the greedy algorithm for set cover," *Journal of Algorithms*, **25**, 237-254, 1997.