A better-than-greedy approximation algorithm for the minimum set cover problem

Refael Hassin[1] and Asaf Levin[2]

February 3, 2005

Abstract

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 each $S \in \mathcal{F}$ has a positive cost c_S. The problem is to compute a sub-collection SOL such that $\bigcup_{S \in SOL} S_j = E$ and its cost $\sum_{S \in SOL} c_S$ is minimized. When $|S| \leq k \ \forall S \in \mathcal{F}$ we obtain the weighted k-set cover problem. It is well known that the 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 exact for the greedy algorithm for all constant values of k. In this paper we give the first improvement on this approximation ratio for all constant values of k. This result shows that the greedy algorithm is not the best possible for approximating the weighted set cover problem. Our method is a modification of the greedy algorithm that allows the algorithm to regret.

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 ($|S| \leq k \ \forall S \in \mathcal{F}$), 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

[1]Department of Statistics and Operations Research, Tel-Aviv University, Tel-Aviv 69978, Israel. hassin@post.tau.ac.il

[2]Department of Statistics, The Hebrew University, Jerusalem 91905, Israel. levinas@mscc.huji.ac.il

bound is tight even for the unweighted version of the problem (see, [10, 13]). For unbounded values of k, Slavík [17] 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 [16] 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 set cover problem (unless $NP \subseteq DTIME(n^{polylog\ n})$). The unweighted k-set cover problem is known to be NP-complete [11] and MAX SNP-hard for all $k \geq 3$ [3, 12, 14]. The hardness results obtained for the unweighted k-set cover problem do not exclude the possibility of obtaining a cH_k-approximation algorithm for some constant $c < 1$ for constant values of k, however such positive results are unknown. Another algorithm for the weighted set cover problem by Hochbaum [9] 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 [15] for a survey on these results.

In spite of these bad news that hold also for the unweighted set cover, 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.

This important progress in the approximability of the unweighted k-set cover does not help to approximate the weighted k-set cover problem within a factor better than H_k. The question whether there are similar results for the weighted problems was left unanswered. A first answer 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. Their method is based on extending the algorithm of [7] for this case, but this approach does not

extend further to the general case. In this paper we give the first positive answer for a general cost function by developing an approximation algorithm for the weighted k-set cover problem whose performance guarantee is bounded by $H_k - \frac{k-1}{8k^9}$. This bound is not tight, but it is sufficient to show that the greedy algorithm does not give the best possible performance guarantee for approximating the weighted set cover problem. Our method is based on a modification of the greedy algorithm that allows the algorithm to regret throughout its execution.

Paper overview: In Section 2 we review the greedy algorithm for the weighted minimum k-set cover problem, and its analysis. In Section 3 we present the greedy algorithm with withdrawals. We analyze its performance in Section 4. We conclude the paper in Section 5 by a short discussion on possible extensions and improvements of our results.

2 The greedy algorithm

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

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

Let w_S be the number of uncovered items in S, and the current *ratio of* S is $r_S = \frac{c_S}{w_S}$. Let S^* be a set such that r_{S^*} is minimized. The algorithm adds S^* to the collection of subsets of the solution, it defines the items of S^* as covered, and it 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^*).

Chvátal [2], extending previous results of Johnson [10] and Lovàsz [13] for the unweighted set cover, showed that this is an H_k-approximation algorithm for the weighted 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 c_S for S. When the i-th item of S was covered by the greedy algorithm, the algorithm could choose S as a feasible solution with current ratio

of $\frac{c_S}{|S|-i+1}$ (this is an upper bound on the current ratio because S has at least $|S|-i+1$ uncovered elements at this point). Therefore, the price assigned to the this item is at most $\frac{c_S}{|S|-i+1}$. It follows that the total price assigned to the items of S is at most $c_S \sum_{i=1}^{|S|} \frac{1}{|S|-i+1} = c_S \sum_{i'=1}^{|S|} \frac{1}{i'} \leq c_S H_k$, and therefore, the approximation ratio of the greedy algorithm is at most H_k.

3 The greedy algorithm with withdrawals

In this section we present our modification of the greedy algorithm where in each iteration we are allowed to withdraw a subset S from the solution. However, if we decide to do so, we must pick new subsets that cover all the items that were covered by S (i.e., an item that is covered in some iteration remains covered in any future iteration). Such a *withdrawal step* allows the algorithm to overcome some choices that are essential in the known bad examples for the greedy algorithm. Incorporating the withdrawal steps we will prove an improved performance guarantee. The first step of the algorithm adds to $\mathcal{F}$ all the subsets of every given set $S \in \mathcal{F}$. These subsets are given the same cost as S. If $S' \subseteq S_1, S_2$ where $S_1, S_2 \in \mathcal{F}$, then $c_{S'} = \min\{c_{S_1}, c_{S_2}\}$. It is clear that this addition does not change the value of a minimum cover and optimal solutions remain optimal. Moreover, given a solution to the new instance, it is easy to compute a feasible solution to the original instance with the same cost (by mapping each new subset to the original set that contains it). However, it will make the analysis of the algorithm easier, as reflected by Lemma 2. Formally, the algorithm is defined as follows:

Algorithm GAWW:

Let $\alpha_k = 1 - \frac{1}{k^3}$.

1. **Initialization:** $SOL := \emptyset$ [the solution collection of subsets], $U := E$ [the set of uncovered items].

   For every subset $S \in \mathcal{F}$ and every $S' \subseteq S$, add S' to $\mathcal{F}$ with cost $c_{S'} = c_S$. If $S' \subseteq S_1, S_2$ where $S_1, S_2 \in \mathcal{F}$, then $c_{S'} = \min\{c_{S_1}, c_{S_2}\}$. Let $\mathcal{F} := \{S_1, \ldots, S_n\}$ be the resulting extended collection, and denote by c_j the cost of S_j.

2. **Iteration:** While $U \neq \emptyset$ do:

   (a) For every j, let $w_j := |S_j \cap U|$. If $w_j \neq 0$, let $r_j := \frac{c_j}{w_j}$.

   (b) For every $S_j \in SOL$ and every sub-collection $C \subseteq \mathcal{F}$ of at most k subsets such that $S_j \subseteq \bigcup_{S \in C} S$, let $w(C) := |\bigcup_{S \in C} S \cap U|$ be the number of still uncovered items that belong to the subsets in C. If $w(C) \neq 0$, let $r(S_j, C) := \frac{\sum_{i:S_i \in C} c_i - c_j}{w(C)}$.

   (c) Let j^* be an index such that r_{j^*} is minimized (in case of a tie we prefer a minimal subset), and let $\tilde{j}, \tilde{C}$ be such that $r(S_{\tilde{j}}, \tilde{C})$ is minimized (again, in case of a tie we prefer minimal subsets).

   (d) **Greedy step:** If $r_{j^*} \leq \frac{1}{\alpha_k} r(S_{\tilde{j}}, \tilde{C})$, then add S_{j^*} to the solution and define the price of the newly covered items as r_{j^*}. Formally, do the following:

      i. For every $e \in S_{j^*} \cap U$, let $price(e) := r_{j^*}$.

      ii. $U := U \setminus S_{j^*}$.

      iii. $SOL := SOL \cup \{S_{j^*}\}$.

   (e) **Withdrawal step:** Otherwise (i.e., $r(S_{\tilde{j}}, \tilde{C}) < \alpha_k r_{j^*}$[3]), replace $S_{\tilde{j}}$ by the subsets in $\tilde{C}$ and define the price of the newly covered items as $r(S_{\tilde{j}}, \tilde{C})$. Formally, do the following:

      i. For every $e \in \bigcup_{S \in \tilde{C}} S \cap U$, let $price(e) := r(S_{\tilde{j}}, \tilde{C})$.

      ii. $U := U \setminus \bigcup_{S \in \tilde{C}} S$.

      iii. $SOL := (SOL \setminus \{S_{\tilde{j}}\}) \cup \tilde{C}$.

3. Return SOL.

In each iteration the size of U decreases until $U = \emptyset$, and therefore the number of iterations throughout the algorithm is at most n. Since each iteration takes at most $|\mathcal{F}| + |\mathcal{F}|^k$ and because k is a constant, each iteration takes polynomial time. Therefore, the greedy algorithm with withdrawals is a polynomial time algorithm that returns a feasible solution. Therefore, we establish the following lemma:

[3] We note that in this case each $C \in \tilde{C}$ satisfies $C \cap S_{\tilde{j}} \neq \emptyset$, however we do not use this property and therefore do not prove it.

Lemma 1 *The greedy algorithm with withdrawals is a polynomial time algorithm for every constant value of k.*

In the next section we analyze its performance.

4 The analysis of algorithm GAWW

In this section we prove the main theorem of this paper, that for all constant values of k, GAWW guarantees a better approximation ratio than the greedy algorithm.

Lemma 2 *We can assume w.l.o.g. that the sets in OPT are disjoint.*

Proof: The claim follows from the initialization step of GAWW in which it adds to $\mathcal{F}$ for each subset $S \in \mathcal{F}$ with cost c_S all the subsets of S, with the same cost c_S. ■

Lemma 3 *When analyzing the worst case performance of GAWW, we can assume w.l.o.g. that each set of OPT has exactly k elements.*

Proof: Assume that OPT contains sets with less than k elements. We construct a new instance by extending E with a set of *dummy elements*. We extend each subset of OPT with less than k elements using some dummy elements to create a new subset with the same cost. We also add to $\mathcal{F}$ the set $\{e\}$ for any dummy element e. All of these new singleton subsets have zero cost. GAWW will first choose a full cover of the dummy elements, and then continue like it acts on the original instance. However, the new instance has an optimal solution with disjoint k-sets. ■

Lemma 4 *In each step of GAWW the sets in SOL are disjoint.*

Proof: We prove the claim by induction. The claim clearly holds when SOL is empty. It is sufficient to prove that in each step GAWW adds subsets that are disjoint to the other subsets in SOL.

• Assume that the current step is a greedy step that adds a subset X. $\mathcal{F}$ contains all the subsets of X with costs of at most c_X. Let $X' = X \setminus \bigcup_{S \in SOL, S \neq X} S$. If $X' \neq X$, then because

X' has at most the ratio of X, and GAWW prefers to use a minimal subset, we obtain a contradiction. Therefore, X is disjoint to all the prior subsets in SOL.

• Assume that the current step is a withdrawal step that withdraws a subset X and inserts $C = \{X_1, X_2, \ldots, X_l\}$. Let SOL be the family of subsets in the solution before the withdrawal step took place. Recall that for all i, $\mathcal{F}$ contains all the subsets of X_i with a cost at most c_{X_i}. Let $X_i' = \left(X_i \setminus \left(\bigcup_{S \in SOL, S \neq X} S\right)\right) \setminus \bigcup_{j \neq i} X_j$. If there exists i such that $X_i' \neq X_i$, then let C' be as C but with X_i' replacing X_i. Because the withdrawal step that withdraws X and inserts C' has at most the same ratio as the withdrawal step that withdraws X and inserts C, and GAWW prefers to use minimal subsets, we obtain a contradiction. Therefore, every X_i, $i = 1, \ldots, l$ is disjoint to all the other subsets in SOL and to X_j $j \neq i$. ■

Lemma 5 *When analyzing the performance of the greedy algorithm with withdrawals, we can assume w.l.o.g. that every X that entered the solution in some step of the algorithm (X may be in the final solution or it might have been withdrawn by a withdrawal step) and every $O \in OPT$, satisfy $|X \cap O| \leq 1$.*

Proof: Consider an instance of the problem, and assume that the claimed property does not hold. Then, we construct a new instance that satisfies the claimed property and with the same approximation ratio as the original instance. The new instance is composed of k copies of the original instance. We also add for each $O \in OPT$, k subsets each of them has exactly one element of O from each copy, and the k subsets cover the elements of the k copies of O. Then, the new instance has an optimal solution that contains all these added subsets (k subsets for each original subset in OPT). GAWW applies a series of k copies of steps for each step in the original instance. Therefore, it never adds one of the new subsets but it adds only subsets that belong to a single copy of the original instance. The cost of the obtained solution is k times the cost of the resulting solution in the original instance. Therefore, the ratio in this new instance is the same as the old instance. However, the claimed property holds in the new instance. ■

Lemma 6 *The cost of SOL is equal to $\sum_{e \in E} price(e)$, the sum of prices assigned to the elements.*

Proof: The claim clearly holds during initialization phase. Throughout an iteration we partition the increase in the cost of SOL among the new covered elements. The claim follows because the price of each element is set exactly once. ■

Lemma 7 *Assume that $X \in SOL$ in some step of GAWW, and that $x_1 \in X$ is the last element of X that was covered by GAWW. Then, $price(x_1) \geq price(x_i)$ for all $x_i \in X$.*

Proof: We show a stronger claim: let $x_l, x_{l-1}, \ldots, x_1$ be the elements of X according to the order that GAWW covers the elements, then $price(x_1) \geq price(x_2) \geq \cdots \geq price(x_l)$. Suppose that X entered SOL in a greedy step. Since GAWW prefers to use a minimal subset, we conclude that all the elements of X were covered for the first time by X, and all of them have the same price.

Suppose that X entered SOL in a withdrawal step that withdraws Y and inserts $C = \{X = X_1, \ldots, X_l\}$. If $X \subseteq Y$, then the claim can be assumed to hold by an inductive argument. Therefore, w.l.o.g. we assume that $X \setminus Y \neq \emptyset$. Note that the elements of $X \setminus Y$ were given a common price. Then, using an inductive argument it is sufficient to show that $price(x_1) \geq price(x)$ for every $x \in X \cap Y$.

We now argue that $r(Y, C)$, the ratio associated with the withdrawal step that withdraws Y and inserts C, is larger than the ratio at the step in which the algorithm inserted Y.

- Suppose that Y entered in a greedy step. Using Lemma 4,

$$\begin{aligned} r(Y,C) &= \frac{\sum_{i=1}^{l} c_{X_i} - c_Y}{\sum_{i=1}^{l} |X_i \setminus Y|} = \frac{\sum_{i=1}^{l} \left(c_{X_i} \frac{|X_i \setminus Y| + |X_i \cap Y|}{|X_i|} - c_Y \frac{|X_i \cap Y|}{|Y|} \right)}{\sum_{i=1}^{l} |X_i \setminus Y|} \\ &\geq \frac{\sum_{i=1}^{l} \frac{c_{X_i}}{|X_i|} |X_i \setminus Y|}{\sum_{i=1}^{l} |X_i \setminus Y|} \geq \min \left\{ \frac{c_{X_i}}{|X_i|} : i = 1, \ldots, l \right\} \geq \frac{c_Y}{|Y|}. \end{aligned}$$

The first and third inequalities follow from $\frac{c_{X_i}}{|X_i|} \geq \frac{c_Y}{|Y|}$ which holds since Y was chosen instead of X_i and both Y and X_i were legal at the step in which GAWW inserts Y. The second inequality follows because the average is at least the minimum value.

- Suppose that Y entered SOL in a withdrawal step that withdraws a subset Z and inserts $C' = \{Y = Y_1, \ldots, Y_{l'}\}$. We first prove that $r(Y, C) \geq r(Z, C')$. The proof will assume the opposite, that $r(Y, C) < r(Z, C')$, and lead to a contradiction. Note that both ratios $r(Y, C)$ and $r(Z, C')$ refer to the time of the withdrawal that replaced Z by C'.

Consider the step in which C' replaced Z, and consider the "withdrawal step" that withdraws Z and inserts $C \cup C' \setminus \{Y\}$ (this step may be illegal because it may insert more than k subsets). By Lemma 4, the subsets in $C' \setminus \{Y\}$ and C are pairwise disjoint. We use this last fact to show that the number of uncovered elements in $C \cup C' \setminus \{Y\}$ is $\sum_{S \in C \cup C'} |S| - |Y| - |Z|$. To see this note that each element of $Y \cap Z$ is counted twice in $\sum_{S \in C \cup C'} |S|$ (once in C and once in C') and therefore zero times in total, each element of $Z \setminus Y$ is counted once in C' in $\sum_{S \in C \cup C'} |S|$ and therefore zero times in total, and the other elements are counted exactly once. Therefore,

$$\begin{aligned}
r(Z, C \cup C' \setminus \{Y\}) &= \frac{\sum_{S \in C \cup C'} c_S - c_Y - c_Z}{\sum_{S \in C \cup C'} |S| - |Y| - |Z|} \\
&= \frac{\frac{\sum_{S \in C} c_S - c_Y}{\sum_{S \in C} |S| - |Y|} \left(\sum_{S \in C} |S| - |Y|\right) + \frac{\sum_{S \in C'} c_S - c_Z}{\sum_{S \in C'} |S| - |Z|} \left(\sum_{S \in C'} |S| - |Z|\right)}{\sum_{S \in C \cup C'} |S| - |Y| - |Z|} \\
&= \frac{r(Y, C) \left(\sum_{S \in C} |S| - |Y|\right) + r(Z, C') \left(\sum_{S \in C'} |S| - |Z|\right)}{\left(\sum_{S \in C} |S| - |Y|\right) + \left(\sum_{S \in C'} |S| - |Z|\right)} \\
&< r(Z, C'),
\end{aligned}$$

The inequality follows by the assumption $r(Y, C) < r(Z, C')$. We now argue that there is a legal withdrawal step (that is, one that inserts at most k subsets) that withdraws Z with a ratio of at most $r(Z, C \cup C' \setminus \{Y\})$. By Lemma 4, because $|Z| \leq k$, there are at most k subsets in $C \cup C' \setminus \{Y\}$ that intersect Z. Let $A \in C \cup C' \setminus \{Y\}$ satisfy $A \cap Z = \emptyset$.[4] Then,

$$\begin{aligned}
r(Z, C \cup C' \setminus \{Y\}) &= \frac{\sum_{S \in C \cup C'} c_S - c_Y - c_Z}{\sum_{S \in C \cup C'} |S| - |Y| - |Z|} \\
&= \frac{\frac{c_A}{|A|}|A| + \frac{\sum_{S \in C \cup C' \setminus \{A\}} c_S - c_Y - c_Z}{\sum_{S \in C \cup C' \setminus \{A\}} |S| - |Y| - |Z|} \left(\sum_{S \in C \cup C' \setminus \{A\}} |S| - |Y| - |Z|\right)}{\sum_{S \in C \cup C'} |S| - |Y| - |Z|} \\
&= \frac{r_A |A| + r(Z, C \cup C' \setminus \{Y, A\}) \left(\sum_{S \in C \cup C' \setminus \{A\}} |S| - |Y| - |Z|\right)}{|A| + \left(\sum_{S \in C \cup C' \setminus \{A\}} |S| - |Y| - |Z|\right)} \\
&\geq \min\{r_A, r(Z, C \cup C' \setminus \{Y, A\})\}
\end{aligned}$$

Since GAWW could choose to add A greedily, $r_A > r(Z, C') > r(Z, C \cup C' \setminus \{Y\})$, and we conclude that $r(Z, C' \cup C \setminus \{Y, A\}) < r(Z, C')$. Repeating this argument we reach a legal

[4] It is possible to show that in this case $A \in C$. However, we do not use this property and therefore we do not prove it.

withdrawal step (that inserts at most k subsets) with ratio that is better than $r(Z, C')$ and this is a contradiction.

We have shown that $r(Y, C) \geq r(Z, C')$ where each of these the ratios is defined for the step that implemented the respective withdrawal. To complete the proof, we note that after Y is added to SOL and the withdrawal step that withdraws Y and inserts C becomes feasible, $r(Y, C)$ may only increase because the number of uncovered elements of Y is non-increasing. Thus, for every $x \in X \cap Y$, $price(x_1) = r'(Y, C) > r(Y, C) = price(x)$, where r' refers to the step that inserted X and r refers to the step that inserted Y. ■

We picture OPT as a matrix, where each column contains a subset $S_j \in OPT$. By Lemma 2 each element appears exactly once in this matrix, and by Lemma 3 each column has exactly k elements. We sort the elements of S_j according to the order they were covered by GAWW, so that the i-th element of S_j that is covered by the greedy algorithm with withdrawals is in the i-th row of the matrix. We break ties arbitrarily. By Lemmas 2 and 3, this representation is well-defined.

For a set $X \in SOL$, denote by $COL(X)$ the sub-matrix defined by the set of columns where the elements of X appear.

Given $X \in SOL$, the elements of X might have been covered during different steps of GAWW (if X entered SOL in a withdrawal step and X has previously uncovered elements). Given $X \in SOL$ such that X is not contained in the last row of the matrix, we will show that the total price paid by the algorithm for all the elements in $COL(X)$ is at most $\sum_{i:S_i \in COL(X)} c_i(H_k - \frac{1}{8k^8})$. Our proof distinguishes between a set X whose elements were covered in a single step and a set X whose elements were covered in at least two steps.

Lemma 8 *If the elements of X were covered in at least two steps, then the total price paid by GAWW for the elements in $COL(X)$ is at most $(H_k - \frac{1}{k^5+k}) \sum_{S_i \in COL(X)} c_i$.*

Proof: Assume that the element $x_1 \in X$ was covered last among the elements of X (other elements of X might have been covered during the same step). Assume that the elements of X are in columns $S_1, S_2, \ldots, S_l$ where $X \cap S_i = \{x_i\}$, and x_i is in row $k - n_i$ of its column. (Recall that by Lemma 5, $|X \cap S_i| \leq 1$.)

Since x_1 was covered last among the elements of X, by Lemma 7 $price(x_1) \geq price(x_i)\ \forall i$. Since GAWW prefers minimal subsets, the step in which X was inserted to SOL, is a withdrawal step. When applying the withdrawal step that inserts X into SOL, GAWW could apply a greedy step that inserts S_1 with a ratio of $\frac{c_1}{n_1+1}$. Therefore, $price(x_1) \leq \alpha_k \cdot \frac{c_1}{n_1+1} = \frac{k^3-1}{k^3} \cdot \frac{c_1}{n_1+1}$. Thus, $price(x_1) + \frac{price(x_1)}{k^3-1} \leq \frac{c_1}{n_1+1}$, and

$$\begin{aligned} price(x_1) &\leq \frac{c_1}{n_1+1} - \frac{price(x_1)}{k^3-1} \\ &\leq \frac{c_1}{n_1+1} - \frac{1}{k^3-1}\frac{1}{l}\sum_{i=1}^{l} price(x_i), \end{aligned}$$

where the second inequality holds because $price(x_1) \geq price(x_i)\ \forall i$. Therefore,

$$\begin{aligned} \sum_{i=1}^{l} price(x_i) &\leq \sum_{i=1}^{l} \frac{c_i}{n_i+1} - \frac{1}{k^3-1}\frac{1}{l}\sum_{i=1}^{l} price(x_i) \\ &\leq \sum_{i=1}^{l} \frac{c_i}{n_i+1} - \frac{1}{k^4}\sum_{i=1}^{l} price(x_i). \end{aligned}$$

It follows that

$$\sum_{i=1}^{l} price(x_i) \leq \left(1 - \frac{1}{k^4+1}\right)\sum_{i=1}^{l} \frac{c_i}{n_i+1}.$$

For elements $a \in S_i \setminus X$ in the $k - n_a$ row, the price of a is bounded by $price(a) \leq \frac{c_i}{n_a+1}$. Therefore,

$$\begin{aligned} \sum_{e \in COL(X)} price(e) &= \sum_{e \in COL(X)\setminus X} price(e) + \sum_{i=1}^{l} price(x_i) \\ &\leq \sum_{i=1}^{l}\left(H_k - \frac{1}{n_i+1}\right)c_i + \sum_{i=1}^{l}\left(1 - \frac{1}{k^4+1}\right)\frac{c_i}{n_i+1} \\ &\leq \sum_{i=1}^{l}\left(H_k - \frac{1}{k^5+k}\right)c_i, \end{aligned}$$

where the last inequality holds because $n_i + 1 \leq k$. ■

Lemma 9 *Assume that X is not contained in the last row. If the elements of X were covered in a single step, then the total price paid by the algorithm for all the elements in $COL(X)$ is at most $(H_k - \frac{1}{8k^8})\sum_{S_i \in COL(X)} c_i$.*

Proof: Assume that the elements of X are in columns $S_1, S_2, \ldots, S_l$ where $X \cap S_i = \{x_i\}$, and x_i is in row $k - n_i$ of its column. By assumption, $price(x_i) = \frac{c_X}{l}$ for all i.

• Assume that $price(x_1) = \frac{c_X}{l} \leq c_1 \left(\frac{1}{n_1+1} - \frac{1}{4k^5}\right)$. Partition $\{2, 3, \ldots, l\}$ into A and B such that $j \in A$ iff $c_j \left(\frac{1}{n_j+1} - \frac{1}{8k^7}\right) \geq \frac{c_X}{l}$. Since $\frac{1}{n_j+1} - \frac{1}{8k^7} \geq \frac{1}{k} - \frac{1}{8k^7} \geq \frac{1}{2k}$, it follows that $\frac{c_j}{2k} \leq \frac{c_X}{l}$ for $j \in B$. Therefore,

$$\sum_{j \in B} \frac{c_j}{8k^8} \leq \frac{1}{4k^7} \sum_{j \in B} \frac{c_X}{l} \leq \frac{c_X}{4lk^6} \leq \frac{c_1}{4k^6} \left(\frac{1}{n_1+1} - \frac{1}{4k^5}\right) \leq \frac{c_1}{4k^6},$$

where the second inequality follows since $|B| \leq |X| \leq k$, the third inequality by the assumption on x_1, and the last inequality follows because $n_1 + 1 \geq 1$. We conclude that

$$\begin{aligned}
&\sum_{e \in COL(X)} price(e) = \\
&\sum_{e \in COL(X) \setminus X} price(e) + \sum_{i=1}^{l} price(x_i) \leq \\
&\sum_{i=1}^{l} \left(H_k - \frac{1}{n_i+1}\right) c_i + \sum_{i \in A} \left(\frac{1}{n_i+1} - \frac{1}{8k^7}\right) c_i + \sum_{i \in B} \frac{c_i}{n_i+1} + \left(\frac{1}{n_1+1} - \frac{1}{4k^5}\right) c_1 \leq \\
&\sum_{i=1}^{l} \left(H_k - \frac{1}{n_i+1}\right) c_i + \sum_{i \in A} \left(\frac{1}{n_i+1} - \frac{1}{8k^7}\right) c_i + \sum_{i \in B} \left(\frac{1}{n_i+1} - \frac{1}{8k^8}\right) c_i + \left(\frac{1}{n_1+1} - \frac{1}{4k^5} + \frac{1}{4k^6}\right) c_1 \leq \\
&\left(H_k - \frac{1}{8k^8}\right) \sum_{i=1}^{l} c_i.
\end{aligned}$$

• Assume that for every i $price(x_i) = \frac{c_X}{l} \geq c_i \left(\frac{1}{n_i+1} - \frac{1}{4k^5}\right)$. Recall that the elements of X were covered in a single step, and not all of them are in the last row. Therefore, there exists an element $y \in COL(X)$ that was covered after X. Let y be the first element to be covered after X, and w.l.o.g. assume that $y \in S_1$.

When GAWW inserted X, it could also choose S_1 with a ratio of $\frac{c_1}{n_1+1}$. Therefore, $\frac{c_X}{l} \leq \frac{c_1}{n_1+1}$. Hence, by assumption, $c_i \left(\frac{1}{n_i+1} - \frac{1}{4k^5}\right) \leq \frac{c_1}{n_1+1}$, $i = 1 \ldots, l$, so that $c_i \left(1 - \frac{1}{4k^4}\right) \leq c_i \left(1 - \frac{n_i+1}{4k^5}\right) \leq c_1 \frac{n_i+1}{n_1+1}$. Therefore, for $i = 1, \ldots, l$

$$\begin{aligned}
c_i &\leq c_1 \frac{n_i+1}{n_1+1} \left(1 + \frac{1}{4k^4 - 1}\right) \\
&\leq c_1 \left(\frac{n_i+1}{n_1+1} + \frac{k}{4k^4 - 1}\right) \\
&\leq c_1 \left(\frac{n_i+1}{n_1+1} + \frac{1}{3k^3}\right).
\end{aligned}$$

When GAWW covered y it could use a withdrawal step that withdraws X and inserts $S_1, S_2, \ldots, S_l$. Therefore,

$$\begin{aligned}
price(y) &\leq \frac{1}{\alpha_k}\frac{\sum_{i=1}^{l} c_i - c_X}{\sum_{i=1}^{l} n_i} \\
&\leq \frac{1}{\alpha_k}\left(\frac{c_1 \sum_{i=1}^{l} \frac{n_i+1}{n_1+1} - c_X}{\sum_{i=1}^{l} n_i} + \frac{c_1}{3k^2}\right) \\
&= \frac{1}{\alpha_k}\left(\frac{c_1}{n_1+1}\left(1 + \frac{l}{\sum_{i=1}^{l} n_i}\right) - \frac{c_X}{\sum_{i=1}^{l} n_i} + \frac{c_1}{3k^2}\right) \\
&\leq \frac{c_1}{\alpha_k}\left(\frac{1}{n_1+1} + \frac{l}{4k^5 \sum_{i=1}^{l} n_i} + \frac{1}{3k^2}\right) \\
&\leq \frac{c_1}{\alpha_k}\left(\frac{1}{n_1+1} + \frac{1}{2k^2}\right) \\
&= \frac{c_1}{n_1+1} + \frac{c_1}{(k^3-1)(n_1+1)} + \frac{c_1 k^3}{2k^2(k^3-1)} \\
&\leq c_1\left(\frac{1}{n_1+1} + \frac{9}{13k^2}\right) \\
&\leq c_1\left(\frac{1}{n_1} - \frac{1}{4k^2}\right),
\end{aligned}$$

where the first inequality follows because we could use the above withdrawal step to cover y, the second inequality follows because $c_i \leq c_1\left(\frac{n_i+1}{n_1+1} + \frac{1}{3k^3}\right)$ for all i and $l \leq k$ (note that $\sum_{i=1}^{l} n_i$ can be as small as 1), the equation follows by simple algebra, the third inequality follows because by assumption $\frac{c_1}{n_1+1} - \frac{c_1}{4k^5} \leq \frac{c_X}{l}$ and therefore $\frac{c_1}{n_1+1}\frac{l}{\sum_{i=1}^{l} n_i} - \frac{c_X}{\sum_{i=1}^{l} n_i} \leq \frac{lc_1}{4k^5 \sum_{i=1}^{l} n_i}$, the fourth inequality follows because $\frac{1}{4k^4} + \frac{1}{3k^2} \leq \frac{1}{2k^2}$ for $k \geq 3$, the second equation follows by substituting $\alpha_k = 1 - \frac{1}{k^3}$, the fifth inequality follows because $n_1 + 1 \geq 2$ and for all values of $k \geq 3$ $\frac{1}{2(k^3-1)} + \frac{k}{2(k^3-1)} \leq \frac{9}{13k^2}$, and the last inequality follows because $n_1 \leq k - 1$. We conclude that

$$\begin{aligned}
&\sum_{e \in COL(X)} price(e) = \\
&\sum_{e \in COL(X) \setminus (X \cup \{y\})} price(e) + \sum_{i=1}^{l} price(x_i) + price(y) \leq \\
&\left(H_k - \frac{1}{n_1+1} - \frac{1}{n_1}\right) c_1 + \sum_{i=2}^{l}\left(H_k - \frac{1}{n_i+1}\right) c_i + \sum_{i=1}^{l} price(x_i) + \left(\frac{1}{n_1} - \frac{1}{4k^2}\right) c_1 \leq \\
&\sum_{i=1}^{l}\left(H_k - \frac{1}{n_i+1}\right) c_i + \sum_{i=1}^{l} price(x_i) - \frac{1}{4k^2}\frac{c_1}{n_1+1} \leq
\end{aligned}$$

$$\sum_{i=1}^{l}\left(H_k-\frac{1}{n_i+1}\right)c_i+\sum_{i=1}^{l}price(x_i)-\frac{1}{4k^2}\frac{1}{l}\sum_{i=1}^{l}price(x_i)\leq$$
$$\sum_{i=1}^{l}\left(H_k-\frac{1}{n_i+1}\right)c_i+\left(1-\frac{1}{4k^3}\right)\sum_{i=1}^{l}price(x_i)\leq$$
$$\sum_{i=1}^{l}\left(H_k-\frac{1}{4k^4}\right)c_i,$$

where the first inequality follows because $price(y)\leq c_1\left(\frac{1}{n_1}-\frac{1}{4k^2}\right)$, the second inequality follows because $n_1+1\geq 1$, the third inequality follows because $price(x_i)=price(x_1)\leq\frac{c_1}{n_1+1}$ for all i, the fourth inequality follows because $l\leq k$, and the last inequality follows because $price(x_i)\leq\frac{c_i}{n_i+1}$ and therefore $\left(1-\frac{1}{4k^3}\right)price(x_i)\leq\left(1-\frac{1}{4k^3}\right)\frac{c_i}{n_i+1}\leq c_i\left(\frac{1}{n_i+1}-\frac{1}{4k^4}\right)$. ■

The following lemma follows from the fact that when we cover an element we pay a price that is at most the price paid for this element by the greedy algorithm.

Lemma 10 *For every subset $S\in OPT$, the total price of the elements that belong to S is at most H_kc_S.*

Theorem 11 *The approximation ratio of the greedy algorithm with withdrawals is at most $H_k-\frac{k-1}{8k^9}$. The time complexity of the algorithm is polynomial for every constant value of k.*

Proof: The claim on the time complexity of GAWW follows by Lemma 1. It remains to show the approximation ratio of GAWW. We consider the solution SOL returned by GAWW. Denote by Y the elements of the last row, and consider the following column multi-set $MS=\{S\in COL(X):X\in SOL,\ X\not\subseteq Y\}$. For a copy of a column S that appears in MS as part of $COL(X)$, we associate a cost of $c_S(H_k-\frac{1}{8k^8})$. By Lemmas 8 and 9, the total price of the elements in MS, where each element is counted the number of times its column appears in MS, is at most the total associated cost of the columns in MS. By Lemma 5, the number of copies of each column in this collection is at least $k-1$ and at most k. For each column S that appears in MS $k-1$ times, we add an additional copy of it associated with a cost of H_kc_S. By Lemma 10, the total price of the elements in MS, where each element is counted the number of times its column appears in MS, is at most the total associated cost of the columns in MS. Since MS has exactly k copies of each column, by Lemma 6, it suffices to show that the average associated cost of each column S

is at most $(H_k - \frac{k-1}{8k^9})c_S$. The average cost of the copies in the resulting multi-set is at most $c_S(\frac{1}{k}H_k + \frac{k-1}{k}(H_k - \frac{1}{8k^8})) = c_S(H_k - \frac{k-1}{8k^9})$, and therefore $H_k - \frac{k-1}{8k^9}$ is an upper bound on the approximation ratio of GAWW. ■

Remark 12 The following is a bad instance for the greedy algorithm with withdrawals: Assume that $OPT = \begin{pmatrix} X & X & X \\ Y & Y & Y \\ a & b & c \end{pmatrix}$, where the costs of the sets in OPT (the columns of the matrix) are $c_1 = 936$, $c_2 = 1161$, and $c_3 = 1543$. Therefore $OPT = 3640$. Assume that $c_X = 936$, $c_Y = 1404$, $c_a = 774$, $c_b = 1161$, $c_c = 1543$. GAWW could pick the sets X, Y, a, b, c in this order, with a total cost of 5818. Therefore, the approximation ratio of GAWW is at least $\frac{5818}{3640} \sim 1.5984$. We note that the cost values in this example were chosen to gives the worst possible ratio for any instance with the same OPT and sequence of operations chosen by the algorithm.

5 Discussion

In this paper we addressed the weighted k-set cover problem, and introduced the first improvement over the greedy algorithm for any constant k. Although we are able to prove only a small improvement over the greedy algorithm, the worst example we found for $k = 3$ is with ratio of $\frac{5818}{3640}$, and the gap between these numbers is relatively large. Tightening this gap is left for future research.

In this paper we were mainly interested in proving an improvement over the greedy algorithm, and we did not try to optimize the constant and the exponent of the term $\frac{k-1}{8k^9}$.

We defined the greedy algorithm with withdrawals requiring that in each withdrawal step a single set in the current solution is withdrawn (and its elements must be covered by the new added subsets). We can generalize this algorithm and consider in each step the possibility of withdrawing at most l subsets from the current solution. This version of the algorithm is clearly polynomial for any constant l. It is natural to conjecture that the approximation ratio of the resulting algorithm is a decreasing function of l. In this paper we proved that $l = 1$ strictly improves the bound of $l = 0$ which corresponds to the greedy algorithm of Chvátal [2].

We conclude this discussion by noting that the withdrawal operation was crucial to obtain better approximation ratio. We mean that allowing the greedy algorithm to insert p subsets at each step (where p is some constant), does not improve the approximation ratio of the algorithm. To see this note that we can take p copies of an original instance, and the new algorithm will have the same performance guarantee on the new instance as the greedy algorithm has on the original instance.

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] D. S. Hochbaum, "Approximation algorithms for the weighted set covering and node cover problems," *SIAM Journal on Computing*, **11**, 555-556, 1982.

[10] D. S. Johnson, "Approximation algorithms for combinatorial problems," *Journal of Computer and System Sciences*, **9**, 256-278, 1974.

[11] 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.

[12] S. Khanna, R. Motwani, M. Sudan and U. V. Vazirani, "On syntactic versus computational views of approximability," *SIAM J. Comput.*, **28**, 164-191, 1998.

[13] L. Lovàsz, "On the ratio of optimal integral and fractional covers," *Discrete Mathematics*, **13**, 383-390, 1975.

[14] C. H. Papadimitriou and M. Yannakakis, "Optimization, approximation and complexity classes," *J. Comput. Syst. Sci.*, **43**, 425-440, 1991.

[15] V. T. Paschos, "A survey of approximately optimal solutions to some covering and packing problems," *ACM Computing Surveys*, **29**, 171-209, 1997.

[16] 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.

[17] P. Slavík, "A tight analysis of the greedy algorithm for set cover," *Journal of Algorithms*, **25**, 237-254, 1997.