Covering Problems with Hard Capacities*

Julia Chuzhoy Joseph (Seffi) Naor

Computer Science Department
Technion, Haifa 32000, Israel
E-mail: {cjulia,naor}@cs.technion.ac.il.

Abstract

We consider the classical vertex cover and set cover problems with *hard* capacity constraints. This means that a set (vertex) can only cover a limited number of its elements (adjacent edges) and the number of available copies of each set (vertex) is bounded. This is a natural generalization of the classical problems that also captures resource limitations in practical scenarios.

We obtain the following results. For the unweighted vertex cover problem with hard capacities we give a 3-approximation algorithm which is based on randomized rounding with alterations. We prove that the weighted version is at least as hard as the set cover problem, yielding an interesting separation between the approximability of weighted and unweighted versions of a "natural" graph problem. A logarithmic approximation factor for both the set cover and the weighted vertex cover problem with hard capacities follows from the work of Wolsey [27] on submodular set cover. We provide here a simple and intuitive proof for this bound.

1 Introduction

The *set cover* problem is defined as follows. Let $E = \{1, \ldots, n\}$ be a ground set of elements and let $\mathcal{S}$ be a collection of sets defined over E. Each $S \in \mathcal{S}$ has a non-negative *cost* $w(S)$ associated with it. A *cover* is a collection of sets such that their union is E. The goal is to find a cover of minimum cost. The set cover problem is a classic NP-hard problem that has been studied extensively in the literature, and the best approximation factor achievable for it is $\Theta(\log n)$ [9, 11, 19, 21].

We consider in this paper the set cover problem with *capacity* constraints, or the *capacitated set cover* problem. We assume that each set $S \in \mathcal{S}$ has a *capacity* $k(S)$ associated with it, meaning that it can cover at most $k(S)$ elements.

Generally, capacitated covering problems come in two flavors. In the case of *soft* capacities, an unbounded number of copies of each covering object is available. In the case of *hard* capacities, which is considered in this paper, each covering object (set S) has a bound

*An extended abstract of this paper appeared in the Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, pp. 481-489, 2002.

($m(S)$) on the number of available copies. Thus, a cover $\mathcal{C}$ is a multi-set of input sets that can cover all the elements, while $\mathcal{C}$ contains at most $m(S)$ copies of each $S \in \mathcal{S}$, and each copy covers at most $k(S)$ elements.

The capacitated (multi)-set cover problem is a natural generalization of a basic and well-studied problem that captures practical scenarios where resource limitations are present.

A special case of the capacitated set cover problem that we consider is the *capacitated vertex cover* problem, defined as follows. An undirected graph $G = (V, E)$ is given and each vertex $v \in V$ is associated with a *cost* $w(v)$, a *capacity* $k(v)$, and a *multiplicity* $m(v)$ (we assume that no parallel edges are present). The goal is to find a minimum cost multi-set U of vertices that cover all the edges, such that for each vertex $v \in V$, at most $m(v)$ copies appear in U, and each copy covers at most $k(v)$ edges adjacent to v. The capacitated vertex cover problem generalizes the well known vertex cover problem, probably one of the most studied problems (see [18] for an overview) in the area of approximation algorithms, for which the currently best known approximation factor is $2 - \frac{\log\log|V|}{2\log|V|}$ [3, 17].

The capacitated vertex cover problem was first introduced by Guha, Hassin, Khuller and Or [16]. They considered the version of the problem with *soft* capacities, a special case where the number of available copies of each vertex is unbounded. A straightforward rounding of a linear programming relaxation of the problem gives a 4-approximate solution. Guha *et al.* [16] show a 2-approximation primal-dual algorithm and they also give a 3-approximation for the case where each edge $e \in E$ has an (unsplittable) demand $d(e)$. (Gandhi *et al.* [13] provide further results on the capacitated vertex cover problem with soft capacities.) Guha *et al.* [16] motivate the study of the vertex cover problem with soft capacities by an application in glycobiology. The problem emerged in the redesign of known drugs involving glycoproteins and can be represented as an instance of the capacitated vertex cover problem.

Two other closely related capacitated covering problems are *capacitated facility location* and *capacitated k-median*. In both problems, the input consists of a set of facilities and a set of clients. For each facility and each client, there is a distance that defines the cost of assigning the client to the facility. Each facility f has a capacity k_f, and a number of available copies m_f. Each client i has a demand d_i. The goal is to open facilities and to assign all the clients to them. In the facility location problem, each facility f has a cost w_f. Any number of facilities can be opened and the cost of a solution is the total cost of the open facilities plus the assignment costs of the clients. In the k-median problem, we are given a bound k on the number of facilities. A solution must contain at most k facilities and the cost of the solution is the sum of the assignments costs of the clients to the facilities. The capacitated set cover problem is a special case of facility location with hard capacities, where all the distances are either 0 or ∞ (note that this distance function is not a metric). Bar-Ilan *et al.* [2] gave an $O(\log n + \log M)$-approximation for the facility location with hard capacities, where M is the value of the maximum input parameter.

Prior Work There is extensive research on the set cover problem and the reader is referred to the surveys in [15, 7, 1, 23, 18]. Feige [11] proved that it is impossible to obtain a better than $(1 - o(1)) \ln n$-approximation for set cover, unless NP has slightly super-polynomial

time algorithms. A greedy heuristic gives an $O(\log n)$-approximation [9, 21] for the set cover problem.

Wolsey [27] considered the *submodular set cover* problem. Let f be a real valued function defined over all subsets of a finite set of elements E. Function f is called *non-decreasing* if $f(S) \leq f(T)$ for all $S \subseteq T \subseteq E$, and *submodular* if $f(S) + f(T) \geq f(S \cap T) + f(S \cup T)$ for all $S, T \subseteq E$. The input to the submodular set cover problem is a family $\mathcal{S}$ of subsets of E together with a non-negative cost function. There is a non-negative non-decreasing submodular function f defined over all collections of the input sets. The goal is to find a minimum cost collection $\mathcal{P}$ of sets, such that $f(\mathcal{P}) = f(\mathcal{S})$. The special case where $f(S) = |S|$ for each set $S \in \mathcal{S}$ is the classical set cover problem.

Consider the capacitated set cover problem. We can assume without loss of generality that the multiplicities of all the sets are unit as is the case in the submodular set cover problem by viewing each one of the $m(S)$ copies of each set $S \in \mathcal{S}$ as a distinct set. For any family $\mathcal{A}$ of input sets, define $f(\mathcal{A})$ to be the maximum number of elements that $\mathcal{A}$ can cover (given the capacity constraints). It is not hard to see that f is a non-decreasing non-negative submodular function. Wolsey [27] showed using dual fitting that the approximation factor of a greedy heuristic for the submodular set cover problem is $1 + O(\log f_{\max})$, where $f_{\max} = \max_{S \in \mathcal{S}} f(\{S\})$.

Metric facility location, i.e., where the distance function defined on the clients is a metric, is a well studied field. Many heuristics, as well as approximation algorithms with bounded performance guarantees, were developed [6, 22, 24, 26]. For the metric facility location problem with hard capacities, Pál, Tardos and Wexler [25] recently gave a $(9 + \epsilon)$-approximation using local search. Unlike the facility location problem, it is not known whether the capacitated k-median problem has a constant approximation, even if the capacities are soft. Bartal *et al.* [4] show a constant factor bi-criteria approximation for the soft capacities version, where the number of open facilities k is exceeded by at most a constant. Chuzhoy and Rabani [8] provide a different constant factor bi-criteria approximation for the same problem. In their algorithm, the capacities are violated by at most a constant factor, while the number of facilities k is not exceeded.

1.1 Our Contribution

The first result we present is a randomized 3-approximation algorithm for the unweighted capacitated vertex cover problem in simple graphs (i.e., with no parallel edges). Our algorithm uses randomized rounding with *alterations*. The first rounding step in our algorithm applies randomized rounding where the probabilities are derived from a solution to a linear programming relaxation of the problem. However, the rounding may not yield a feasible cover and therefore we need to add more vertices to the cover. This is done in the alteration step. Our analysis uses a sophisticated charging scheme to bound the number of vertices that are added to the cover in this step. We also prove that the more general version where edges have unsplittable demands is not approximable in the presence of hard capacities. Contrast this with the 3-approximation algorithm of Guha *et al.* [16] for this case (with soft capacities).

We consider the weighted capacitated vertex cover problem and prove that it is set-cover hard. This means that the best approximation factor that can be achieved for this problem is $\Omega(\log n)$. Our hardness proof holds even for the case of $\{0,1\}$ weights and unit multiplicity. Interestingly, we are not aware of any other "natural" graph problem where there is a logarithmic separation between the approximability of weighted and unweighted versions. (However, there are several examples of problems where the unweighted version is polynomially solvable while the weighted version is NP-hard.)

We leave open the version of the capacitated vertex cover problem where the graph is unweighted, yet there are parallel edges. Neither our constant-factor approximation algorithms, nor our hardness results, are applicable to this version of the problem. Following our work, Gandhi *et al.* [12] obtained a 2-approximation for the unweighted vertex cover with hard capacities, using a modification of our algorithm.

We proceed to consider the capacitated set cover problem. As already noted, it follows from Wolsey's work [27] that a natural greedy heuristic achieves an approximation factor of $O(\log n)$ for this problem. We note that the integrality gap of the natural linear programming relaxation of the problem is unbounded, similar to the case of facility location with hard capacities [25]. Indeed, Wolsey used a different linear programming formulation (see Section 6 for a formulation of the linear program). We consider the same greedy heuristic as Wolsey and provide a direct combinatorial proof of the approximation factor of this heuristic. We believe that our proof is simple and intuitive. We note that the main obstacle in applying the "standard" (set cover) charging scheme in the presence of hard capacities is that it is not clear how to "charge" the sets in the optimal solution for the sets in the solution computed by the greedy algorithm. Since there are hard capacities, the assignment of elements to sets in the cover is dynamic, and, moreover, elements may be covered and uncovered several times during the iterations of the algorithm.

The rest of the paper is organized as follows. In Section 3, we show a 3-approximation algorithm for unweighted capacitated vertex cover. In section 4.1 we show that the weighted capacitated vertex cover problem is at least as hard as the set cover problem, even in the case where $m(v) = 1$ for all $v \in V$. In Section 4.2 we consider the version where edges have unsplittable demands and show that this version is not approximable in the presence of hard capacities. In Section 5 we provide a description of the greedy algorithm for the set cover problem with hard capacities, and give a simple proof that the algorithm achieves an $O(\log n)$-approximation, implying an $O(\log |V|)$-approximation for the weighted capacitated vertex cover problem. In Section 6 we discuss extensions of the algorithm to more general covering problems, such as submodular set cover and multi-set multi-cover.

2 Preliminaries

A set cover instance with hard capacities contains a ground set of elements $E = \{1, \ldots, n\}$ and a collection of sets $\mathcal{S}$ defined over E. Each set $S \in \mathcal{S}$ is associated with a non-negative cost $w(S)$, a capacity $k(S)$ that bounds the number of elements it can cover, and a bound $m(S)$ on the number of available copies of S. Let $\mathcal{P}$ be a multi-set of sets from $\mathcal{S}$. Then

$C \subseteq \mathcal{P} \times E$ is called a *partial cover* iff for each $(S, e) \in C$, $e \in S$. We say that element $e \in E$ is covered by S in C if $(S, e) \in C$. Without loss of generality we can assume that each element $e \in E$ is covered in C at most once. Cover C is *feasible* if $\mathcal{P}$ contains at most $m(S)$ copies of each $S \in \mathcal{S}$, and each copy covers at most $k(S)$ elements. The *value* of C, i.e., the number of elements it covers, is denoted by $|C|$. Given a multi-set $\mathcal{P}$, denote by $f(\mathcal{P})$ the maximal value of a feasible (partial) cover $C \subseteq \mathcal{P} \times E$. The cost of C is defined to be the sum of the costs of the sets belonging to C. The goal is to find a feasible cover of minimum cost.

Lemma 1 *Given an instance of set cover with hard capacities and a multi-set $\mathcal{P}$ of sets from $\mathcal{S}$, a cover C of value $f(\mathcal{P})$ can be computed in polynomial time. In particular, it can be established whether $\mathcal{P}$ defines a feasible solution to the set cover problem.*

Proof: For each $S \in \mathcal{S}$, let $m^{\mathcal{P}}(S)$ denote the number of copies of S that appear in $\mathcal{P}$. We build the following directed network. Let $G = (L, R, E')$ be the directed incidence graph of $\mathcal{P}$ and E, i.e., L contains a vertex for each copy of each set in $\mathcal{P}$: $L = \{v_i(S) | \ S \in \mathcal{S}, 1 \leq i \leq m^{\mathcal{P}}(S)\}$, $R = E$. For each $v_i(S) \in L$, $e \in R$, there is an edge $(v_i(S), e) \in E'$ of capacity 1 iff $e \in S$. Add a source vertex s and an edge $(s, v_i(S))$ of capacity $k(S)$ for each $S \in \mathcal{S}, 1 \leq i \leq m^{\mathcal{P}}(S)$. Add a sink vertex t an edge (e, t) of capacity 1 for each $e \in E$.

Consider the maximum flow in this network. The value of the flow is at least $f(\mathcal{P})$, since the optimal cover defines a feasible flow in the network. Also, the maximum flow in the network is integral, and thus it induces a feasible partial cover of the same value.

Clearly, $\mathcal{P}$ is a feasible solution to the set cover problem iff $f(\mathcal{P}) = |E|$ and for each $S \in \mathcal{S}$, $m^{\mathcal{P}}(S) \leq m(S)$. □

Since vertex cover with hard capacities is a special case of set cover with hard capacities, (where each vertex $v \in V$ can be viewed as a set whose elements are the edges adjacent to v), all the above definitions, as well as Lemma 1, can also be applied to the vertex cover problem.

3 Vertex Cover with Hard Capacities

In this section, we show a randomized 3-approximation algorithm for the unweighted capacitated vertex cover problem. Our starting point is the following linear programming relaxation of the problem. For $v \in V$, let $x(v)$ be a variable indicating the number of copies of v that belong to the cover. For $e = (u, v) \in E$, let $y(e, v)$ be a variable indicating whether vertex v covers edge e. Denote by (x, y) a solution to (UVC). For each $v \in V$, $N(v)$ denotes the set of edges adjacent to v.

Lemma 2 *Let (x, y) be a feasible solution to (UVC), where x is integral. Then, there exists a feasible solution (x, y') to (UVC) where y' is integral. Moreover, this solution can be computed in polynomial time (given x).*

Proof: Let U be the multi-set of vertices defined by x, i.e., for each vertex $v \in V$, there are exactly $x(v)$ copies of v in U. We use Lemma 1 to compute an (integral) cover y' of the

$$\min \quad \sum_{v \in V} x(v) \qquad \text{(UVC)}$$

s.t.

$$y(e,u) + y(e,v) = 1 \qquad \text{for all } e = (u,v) \in E \qquad (1)$$

$$y(e,v) \leq x(v) \qquad \text{for all } e \in E,\ v \in e \qquad (2)$$

$$\sum_{e \in N(v)} y(e,v) \leq k(v) \cdot x(v) \qquad \text{for all } v \in V \qquad (3)$$

$$x(v), y(e,v) \geq 0 \qquad \text{for all } v \in V,\ e \in E$$

$$x(v) \leq m(v) \qquad \text{for all } v \in V$$

edges by vertices in U. Note that y induces a fractional flow of value $|E|$ in the network constructed in the proof of Lemma 1. Thus, $f(U) = |E|$, and therefore, in y', all the edges are covered. □

3.1 A Simple 8-Approximation Algorithm

In this section we show a simple 8-approximation algorithm for the special case of unit multiplicities (i.e., for each vertex exactly one copy is available). The description and the analysis of the algorithm are presented in an informal way. The goal is to give an intuitive explanation of the ideas behind the algorithm. The next section contains a formal description and analysis of a modified (and more complicated) version of the algorithm, which achieves a 3-approximation for the general version (with arbitrary multiplicities).

Let (x, y) be a fractional optimal solution to (UVC). We will find a feasible solution (x', y') where x' is integral and the expected cost is at most 8 times the cost of the original solution (x, y). By Lemma 2, (x', y') can be coverted into an integral solution of the same cost. The algorithm consists of three steps.

Step 1: (Setting it up) We define $U = \{v \mid x_v \geq \frac{1}{2}\}$. Note that each edge $e \in E$ has at least one endpoint in U. Let $\overline{U} = V \setminus U$, and let E' denote all the edges with one endpoint in $\overline{U}$. The graph is as follows:

For each vertex $u \in U$, let $N'(u)$ denote the set of edges in E' incident to u. Define $\ell(u) = \sum_{e \in N'(u)} y(e,u)$ and $r(u) = \sum_{e=(u,v) \in N'(u)} y(e,v) = |N'(u)| - \ell(u)$. Note that the value of $\ell(u)$ denotes the total contribution of u to the coverage of edges in $N'(u)$ (or the "budget" of u), and $r(u)$ denotes the total contribution of vertices in $\overline{U}$ to the coverage of these edges (or the "deficit" of u). The idea is to choose a small subset of vertices $I \subseteq \overline{U}$, such that each $u \in U$ can receive a contribution of at least $r(u)$ from the vertices in I for covering the edges in $N'(u)$. The coverage of the edges with both endpoints in U remains the same as in the LP solution. The construction of set I is performed in two steps: randomized rounding and then alterations.

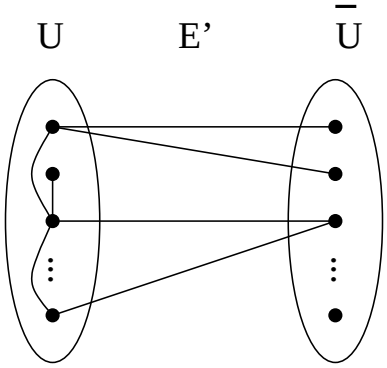

Figure 1: The sets U and $\overline{U}$

Step 2: (Randomized rounding) Each vertex $v \in \overline{U}$ is chosen randomly and independently into I with probability $2x_v$. Consider some edge $e = (u, v)$ with $u \in U$, $v \in I$. We set the contribution of v to the coverage of e to be $z(e, v) = \frac{y(e,v)}{x(v)}$. Note that since $y(e, v) \leq x(v)$ is required in (UVC), $z(e, v) \leq 1$, and it follows from Constraint (3) that the capacity of v is not exceeded. After this step, it is still possible that some vertices $u \in U$ receive a contribution smaller than $r(u)$. This is corrected in the next step.

Step 3: (Altering the rounding) We denote by P the vertices in U that are in "deficit", i.e.,

$$P = \left\{ u \in U \mid \sum_{e=(v,u) \in E', v \in I} z(e, v) < r(u) \right\}.$$

We proceed iteratively. In each iteration, a new vertex v which fractionally covers some edges adjacent to vertices in P is added to I. The set P is updated. We continue until P becomes an empty set. The cost of the newly added vertices is charged to the vertices in P.

An iteration is performed as follows. (See Fig. 2.) Let u be some vertex in P. Then there is at least one edge $e = (v, u) \in E'$, where $v \notin I$. Add v to I. For each edge $e' = (v, w)$, where $w \in P, w \neq u$, set the contribution of v to the coverage of e' to be $z(e', v) = \frac{y(e',v)}{x(v)}$. The contribution of v to the coverage of e is defined to be the minimum between 1 and the remaining capacity of v (which must be at least $y(e, v)$). The cost of v is charged to the vertices in P as follows. Each $w \in P$, $w \neq u$ such that $e' = (w, v) \in E$, pays $z(e', v)$, which is exactly the contribution of v to the coverage of edge e'. The remaining cost is charged to u.

Observe that at the end of this procedure, the cost charged to each vertex $u \in P$ is at most $r(u) + 1$: Let i be the last iteration when u belongs to P. In each iteration, u is charged with the amount equal to the contribution it receives in this iteration. Once the

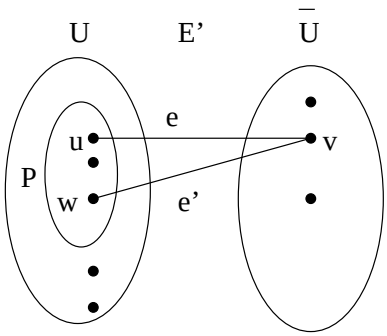

Figure 2: A Step (3) iteration

total contribution exceeds $r(u)$, u is removed from P. So at the beginning of iteration i, the total amount charged to u does not exceed $r(u)$, and u pays at most 1 in iteration i.

Analysis

The cost of the algorithm is divided into three parts.

1. The cost incurred in Step 1 is at most $2\sum_{u\in U} x(u)$.
2. The expected cost of randomized rounding is at most $2\sum_{u\in\overline{U}} x_u$.
3. The expected cost of Step 3 is bounded as follows. Consider some $u \in U$.
   - If $r(u) \leq 2$, then u pays at most $r(u)+1 = 3$ for Step 3.
   - Assume that $r(u) > 2$. In this case the probability that u belongs to P after Step 2 is at most $\frac{2}{r(u)}$ (this follows from a simple application of the Chebyshev inequality; since the next section contains a similar proof, we omit the proof here). Therefore, the expected cost incurred by u is at most $\frac{2}{r(u)} \cdot (r(u)+1) \leq 3$.

   In total, the expected cost of Step 3 is at most $3|U| \leq 6\sum_{u\in U} x(u)$.

Summing up over the three steps, the expected cost of the solution is bounded by

$$8 \cdot \sum_{u\in U} x(u) + 2\sum_{v\in\overline{U}} x(v) \leq 8 \cdot \sum_{v\in V} x(v).$$

3.2 A 3-approximation Algorithm

In this section we show a randomized 3-approximation algorithm for vertex cover with hard capacities and arbitrary multiplicities. The algorithm is based on the ideas presented in the previous section. Consider a fractional optimal solution (x, y) to (UVC). We show how to round this solution, obtaining a feasible solution (x', y'), where x' is integral. By Lemma 2, x' induces an integral capacitated vertex cover. As before, the rounding algorithm consists of three major steps.

Step 1: (Setting it up) We need the following definitions.

- Define $U = \{u | x(u) \geq \frac{1}{3}\}$ and $\overline{U} = V \setminus U$. Let U' be the multiset of vertices, where for each $u \in U$, there are $\lceil x(u) \rceil$ copies of u in U'.
- Define E' to be the set of edges with one endpoint in U and the other endpoint in $\overline{U}$.
- For each $u \in V$, $N'(u) = E' \cap N(u)$.
- For each $u \in U$, define $\ell(u) = \sum_{e \in N'(u)} y(e,u)$, and $r(u) = \sum_{e=(u,v) \in N'(u)} y(e,v) = |N'(u)| - \ell(u)$. Note that the value of $\ell(u)$ denotes the total contribution of u to the coverage of edges in $N'(u)$, and $r(u)$ denotes the total contribution of vertices in $\overline{U}$ to the coverage of these edges.
- For each $u \in U$, define $\epsilon(u) = \frac{\lceil x(u) \rceil}{x(u)} - 1$, and $h(u) = (1 - 2\epsilon(u))r(u)$. The meaning of these variables is explained below.

The constraints of (UVC) guarantee that each edge $e = (u,v) \in E$ has at least one endpoint in U: Since $y(e,u) + y(e,v) = 1$, $y(e,u) \leq x(u)$, and $y(e,v) \leq x(v)$, it follows that either $x(u) \geq \frac{1}{3}$ or $x(v) \geq \frac{1}{3}$ must hold.

Consider a vertex $u \in U$. Let (u,v) be some edge, such that $v \in \overline{U}$. Since $y(e,v) \leq x(v) < \frac{1}{3}$, it follows that $x(u) \geq y(e,u) > \frac{2}{3}$. It also follows that for each $u \in U$, $\ell(u) \geq 2r(u)$, since $y(e,u) \geq 2y(e,v)$ for each $(u,v) \in N'(u)$.

Our cover is going to contain the vertices of U' together with a subset $I \subseteq \overline{U}$, such that $U' \cup I$ can fractionally cover all the edges. (By Lemma 2, $U' \cup I$ is also an integral feasible vertex cover.)

First, we round up $x(u)$ to be equal to $\lceil x(u) \rceil$ for each vertex $u \in U$. As a result, u can increase its contribution to the coverage of the edges belonging to $N'(u)$ by a factor of $\lceil x(u) \rceil / x(u)$, i.e., now it can contribute $\frac{\lceil x(u) \rceil}{x(u)} l(u)$ to the coverage of $N'(u)$. By the definition of $\epsilon(u)$, the new contribution of u is at least $\ell(u)(1 + \epsilon(u)) \geq \ell(u) + 2r(u)\epsilon(u)$. If $\epsilon(u) \geq \frac{1}{2}$, this is enough to complete the coverage of $N'(u)$. Therefore, assume that $\epsilon(u) < \frac{1}{2}$. To complete the fractional cover, we need an additional coverage of value $(1 - 2\epsilon(u))r(u) = h(u)$ from vertices belonging to $\overline{U}$, since $\ell(u) + r(u)$ suffices to cover $N'(u)$. Our goal in the next two steps is to find $I \subseteq \overline{U}$ such that for each $u \in U$, the vertices from I can contribute at least $h(u)$ to the coverage of $N'(u)$.

Step 2: (Randomized rounding) Each vertex $v \in \overline{U}$ is independently chosen to be in I with probability equal to $3x(v)$. Note that for each $v \in \overline{U}$, we add at most one copy of v to I. For each vertex $v \in I$, for each $e \in N'(v)$, define a new cover of edge e by vertex v: $z(e,v) = \frac{y(e,v)}{x(v)}$.

Step 3: (Altering the rounding) In this step we start with a feasible fractional solution (x', y') and iteratively alter it until x' becomes integral, while maintaining feasibility of (x', y'). We denote by P the vertices in U that are in "deficit", i.e.,

$$P = \left\{ u \in U \mid \sum_{e=(v,u)\in E', v\in I} z(e,v) < h(u) \right\}.$$

Our initial feasible solution (x', y') for (UVC) is defined as follows: If $v \in U$, then $x'(v) = \lceil x(v) \rceil$. If $v \in I$, then $x'(v) = 1$. Otherwise $x'(v) = x(v)$. For $e = (u, v)$, $y'(e, v)$ and $y'(e, u)$ are defined as follows.

- If $u, v \in U$, then $y'(e, u) = y(e, u)$ and $y'(e, v) = y(e, v)$.
- If $u \in U \setminus P$: if $v \in I$, then $y'(e, v) = z(e, v)$, else $y'(e, v) = 0$. Set $y'(e, u) = 1 - y'(e, v)$. Note that since $u \notin P$, it has enough capacity to complete the cover of $N'(u)$.
- If $u \in P$: if $v \in I$, then $y'(e, v) = z(e, v)$ and $y'(e, u) = 1 - z(e, v)$. (Note that $y'(e, u) \leq y(e, u)$, since $z(e, v) \geq y(e, v)$). Else ($v \notin I$), set $y'(e, u) = y(e, u)$ and $y'(e, v) = y(e, v)$.

It is easy to see that (x', y') is a feasible solution for (UVC). We now show how to get rid of P by adding new vertices to I. We charge the cost of the new vertices added to I to the vertices of P.

Procedure Eliminate.

While $P \neq \emptyset$:

1. Let $u \in P$, $e = (u, v) \in E'$, such that $v \in \overline{U} \setminus I$ (there must be at least one such v). Let $P' = \{w \in P \mid w \neq u,\ e' = (w, v) \in E'\}$.
2. Add v to I (set $x'(v) = 1$).

   Update the cover: For each $w \in P'$ where $e' = (v, w) \in E'$, set $y'(e', v) = z(e', v) = \frac{y(e', v)}{x(v)}$. Set $y'(e', w) = 1 - y'(e', v)$. Note that the value of $y'(e', w)$ can only decrease. Set $y'(e, v)$ to be the minimum between 1 and the remaining capacity of v (which must be at least $y(e, v)$). Set $y'(e, u) = 1 - y'(e, v)$.

   Update the set P: For each $w \in P$ for which $\sum_{e=(w,a):a\in I} y'(e, a) \geq h(w)$, remove w from P. Update the cover of $N'(w)$ as follows. For each $e = (b, w) \in E'$ such that $b \notin I$, set $y'(e, b) = 0$ and $y'(e, w) = 1$. Note that w has enough capacity to cover all such edges.

It is easy to see that feasibility is maintained after each iteration. The number of iterations of Procedure Eliminate is bounded by $|\overline{U}|$, since $|I|$ is increased by one in each iteration. At the end, when P becomes empty, for each v with $x'(v) < 1$, we set $x'(v) = 0$. The final solution is $U \cup I$. The next theorem follows from the discussion.

Theorem 3 *The algorithm computes a feasible solution (x', y') to (UVC), where x' is integral.*

To obtain an integral capacitated vertex cover, we apply Lemma 2 to the solution (x', y').

3.3 Analysis

The analysis of the rounding is divided into two parts.

Charging scheme for Step (3) We show that we can charge the cost of adding vertices to I in PROCEDURE ELIMINATE to the vertices in P, such that each $u \in P$ pays at most $h(u) + 1$. Consider an iteration of PROCEDURE ELIMINATE. We charge the vertices of $P' \cup \{u\}$ for adding v to I. Each $w \in P'$, where $e' = (w, v) \in E'$, is going to pay $z(e', v)$ (which is exactly the contribution of v to the cover of e'). Vertex u is going to pay the remaining cost (if any remains), which is also at most the contribution of v to the cover of the edge (u, v). We now bound the total amount charged to $a \in P$. While a is still in P, in each iteration it pays exactly an amount equal to the coverage that edges in $N'(a)$ get from the newly added vertex v. Once the coverage of $N'(a)$ coming from vertices in I exceeds $h(a)$, a is removed from P. Therefore, in total a pays at most $h(a) + 1$.

Bounding the cost We now bound the total cost of the solution produced.

Claim 4 *Let $u \in U$ such that $r(u) \geq \frac{3}{4(1+\epsilon(u))^2}$. Then, the probability that $u \in P$ after Step (2) is at most $\frac{3}{4(1+\epsilon(u))^2 r(u)}$.*

Proof: Consider $e = (u, v) \in N'(u)$. We define the random variable

$$t_e = \begin{cases} z(e, v) & v \in I \\ 0 & \text{otherwise} \end{cases}.$$

Variables t_e are independent since there are no parallel edges in the graph. Note that:

- $u \in P$ iff $\sum_{e \in N'(u)} t_e < h(u)$.
- The expectation of $\sum_{e \in N'(u)} t(e)$ is:

$$\begin{aligned} \mu &= \mathbf{Exp}\left[\sum_{e \in N'(u)} t_e\right] \\ &= \sum_{e=(v,u) \in N'(u)} 3z(e, v) \cdot x(v) \\ &= 3r(u). \end{aligned}$$

- The variance of $\sum_{e \in N'(u)} t(e)$ is:

$$\begin{aligned}\sigma^2 &= \mathbf{Var}\left[\sum_{e \in N'(u)} t_e\right] \\ &= \sum_{e=(u,v) \in N'(u)} z^2(e,v) \cdot 3x(v) \cdot (1 - 3x(v)) \\ &\leq \mu.\end{aligned}$$

It follows from Chebyshev's Inequality, when applied to the random variable $\sum_{e \in N'(u)} t_e$, that

$$\begin{aligned}&\mathbf{Prob}\left[\sum_{e \in N'(u)} t_e < h(u)\right] \\ &\leq \mathbf{Prob}\left[\left|\sum_{e \in N'(u)} t_e - \mu\right| \geq \mu - r(u)(1 - 2\epsilon(u))\right] \\ &= \mathbf{Prob}\left[\left|\sum_{e \in N'(u)} t_e - \mu\right| \geq 2r(u)(1 + \epsilon(u))\right] \\ &\leq \frac{\sigma^2}{4r^2(u)(1+\epsilon(u))^2} \\ &\leq \frac{3}{4r(u)(1+\epsilon(u))^2}.\end{aligned}$$

□

We are now ready to compute the expected cost of the solution.

- For $v \in \overline{U}$, the expected cost we pay in Step (2) is $3x(v)$.
- For $u \in U$, where $N'(u) = \emptyset$ or $\epsilon(u) \geq \frac{1}{2}$, we pay at most $3x(u)$ in Step (1), and we do not pay in Step (3).
- For $u \in U$, where $N'(u) \neq \emptyset$ and $\epsilon(u) < \frac{1}{2}$: for the sake of convenience, denote $k = \lceil x(u) \rceil$, $\epsilon = \epsilon(u)$, $x = x(u)$. Consider two cases:
  - If $r(u) \geq \frac{3}{4(1+\epsilon)^2}$, then in Step (1) we pay k for the copies of u we use. In Step (3), we pay at most $h(u) + 1$ with probability at most $\frac{3}{4r(u)(1+\epsilon)^2}$. Thus, the expected cost is bounded by

$$\begin{aligned}k + (h(u)+1) \cdot \frac{3}{4r(u)(1+\epsilon)^2} &= k + ((1-2\epsilon)r(u) + 1)\frac{3}{4r(u)(1+\epsilon)^2} \\ &= k + \frac{3(1-2\epsilon)}{4(1+\epsilon)^2} + \frac{3}{4r(u)(1+\epsilon)^2} \\ &\leq k + 1 + \frac{3(1-2\epsilon)}{4(1+\epsilon)^2}.\end{aligned}$$

– If $r(u) < \frac{3}{4(1+\epsilon)^2}$, then in Step (1) we pay k for the copies of u, and at most $h(u)+1$ in Step (3). In total we pay:

$$\begin{aligned} k+1+h(u) &= k+1+(1-2\epsilon)r(u) \\ &\leq k+1+\frac{3(1-2\epsilon)}{4(1+\epsilon)^2}. \end{aligned}$$

In both cases it suffices to prove that

$$k+1+\frac{3(1-2\epsilon)}{4(1+\epsilon)^2} \leq 3x.$$

Since $x = \frac{k}{1+\epsilon}$, this is equivalent to showing that:

$$k+1+\frac{3(1-2\epsilon)}{4(1+\epsilon)^2} \leq \frac{3k}{1+\epsilon}.$$

Claim 5 *For any $k \geq 1$, $0 \leq \epsilon < \frac{1}{2}$, the following inequality holds:*

$$k+1+\frac{3(1-2\epsilon)}{4(1+\epsilon)^2} \leq \frac{3k}{1+\epsilon}$$

Proof:

The claim is equivalent to:

$$k\left(1-\frac{3}{1+\epsilon}\right)+1+\frac{3(1-2\epsilon)}{4(1+\epsilon)^2} \leq 0$$

Note that $1-\frac{3}{1+\epsilon} < 0$ for $\epsilon < \frac{1}{2}$. Therefore, the left hand side is maximized when $k=1$, and it is enough to prove that:

$$2-\frac{3}{1+\epsilon}+\frac{3(1-2\epsilon)}{4(1+\epsilon)^2} \leq 0.$$

This is equivalent to

$$8\epsilon^2-2\epsilon-1 \leq 0$$

which holds for all ϵ, $0 \leq \epsilon \leq \frac{1}{2}$. □

4 Hardness Results

4.1 Weighted Vertex Cover

We show that the capacitated vertex cover with arbitrary weights is at least as hard as the set cover problem. Given an instance of the set cover problem, let $G = (L, R, E')$ be its bipartite incidence graph, where $L = \mathcal{S}$, $R = E$, $(S, e) \in E'$ iff $e \in S$. For each vertex v in the graph, let $\delta(v)$ denote its degree. For each $v \in L$, define $w(v)$ to be the weight of the corresponding set, and $k(v) = \delta(v)$. For each $v \in R$, define $w(v) = 0$, and $k(v) = \delta(v) - 1$. For each vertex v in the graph, define the multiplicity to be $m(v) = 1$. Given a solution to the set cover instance, the solution to the capacitated vertex cover consists of all the vertices of R and the vertices from L corresponding to the sets in the set cover. The set vertices can cover all their adjacent edges. Since each element is covered in the set cover solution, for each $v \in R$, at least one of its adjacent edges is covered by a set vertex, so v has enough capacity to cover the remaining edges. The converse is also true. Given a feasible solution to the vertex cover problem, we can find a feasible solution to the set cover problem of the same cost. The solution to the set cover problem consists of the sets corresponding to the vertices of L that participate in the solution of the vertex cover instance.

4.2 Vertex Cover with Unsplittable Demands

We assume that each edge e has a demand $d(e)$ that must be supplied by one of its endpoints. For each $v \in V$, the sum of the demands of the adjacent edges that v supplies must not exceed the capacity $k(v)$. It is impossible to approximate this problem, since, given a problem instance, it is NP-hard to answer the question whether V (the set of all the vertices in the problem instance) is a feasible vertex cover, even if the demands are given in unary. The reduction is from the 3-partition problem, which is defined as follows. We are given a bound $B \in Z^+$ and a collection of $3m$ numbers, $a_1, \ldots, a_{3m}$, such that $\sum_{i=1}^{3m} a_i = mB$, and for each $i : 1 \leq i \leq 3m$, $B/4 < a_i < B_2$. The question is whether the numbers can be divided into m sets, such that the sum of numbers in each set is B. Note that if such a division exists, each set will contain exactly three numbers. This problem is NP-hard in the strong sense (i.e., it is NP-hard even if the numbers are given in unary) [14].

The reduction proceeds as follows. We have $3m$ vertices $v_1, \ldots, v_{3m}$ representing the $3m$ numbers. The capacity of vertex v_i, $1 \leq i \leq 3m$ is $(m-1)a_i$. We also have m vertices $u_1, \ldots, u_m$ representing the sets, and the capacity of each such vertex is B. Thus, the set of vertices is $V = \{v_i \mid 1 \leq i \leq 3m\} \cup \{u_j \mid 1 \leq j \leq m\}$. For each v_i, u_j: $1 \leq i \leq 3m$, $1 \leq j \leq m$, there is an edge between the two vertices with demand a_i. Suppose there is a valid partition of the $3m$ numbers into m sets $S_1, \ldots, S_m$. Then for each vertex u_j, $1 \leq j \leq m$ and for each vertex v_i, $1 \leq i \leq 3m$, such that $a_i \in S_j$, vertex u_j covers the edge connecting u_j and a_i. As the sum of the numbers in each S_j is exactly B, the capacity of u_j is enough to cover all these edges. Now for each vertex v_i, $1 \leq i \leq 3m$, one of the edges adjacent to this vertex is covered by one of the vertices $u_1, \ldots, u_m$, and therefore the capacity of v_i is sufficient to cover the remaining $(m-1)$ edges.

The converse direction is also true. Suppose the set V of vertices can cover all the edges. We show a valid partition of the input elements into m sets $S_1, \ldots, S_m$. Note that each vertex v_i, $1 \leq i \leq 3m$, has enough capacity to cover only $(m-1)$ of its adjacent edges. Therefore, at least one edge adjacent to v_i is covered by some u_j, $1 \leq j \leq m$. Set S_j contains all such elements a_i for which vertex u_j covers the edge that connects it with v_i. As the capacities of the vertices u_j, $1 \leq j \leq m$, are B, the elements in each set sum up also to at most B. Thus, a solution to the capacitated vertex cover problem defines a solution to the 3-partition problem.

5 Set Cover with Hard Capacities

In this section we consider the set cover problem with hard capacities. For the sake of simplicity, we assume that for each set $S \in \mathcal{S}$, only one copy is available, i.e., $m(S) = 1$. If this is not the case, we can view each available copy of each set as a distinct set. Note that the input size remains polynomial, as at most n copies of each set are needed. Thus, given two families $\mathcal{A}, \mathcal{B} \subset \mathcal{S}$ of sets, their union is now defined as usual: $\mathcal{A} \cup \mathcal{B} = \{S \mid S \in \mathcal{A} \text{ or } S \in \mathcal{B}\}$.

We need the following notation. Let $\mathcal{T} \subseteq \mathcal{S}$ be a collection of sets and let $f(\mathcal{T})$ denote the maximum number of elements that the sets belonging to $\mathcal{T}$ can cover without violating the capacity constraints. Note that $f(\mathcal{T})$ can be computed using Lemma 1. For $S \in \mathcal{S}$, define $f_{\mathcal{T}}(S) = f(\mathcal{T} \cup \{S\}) - f(\mathcal{T})$ (i.e., $f_{\mathcal{T}}(S)$ is the increase in the number of elements that can be covered when S is added to set family $\mathcal{T}$). Consider the following greedy algorithm for the set cover problem with hard capacities.

Algorithm Greedy Cover:

1. Initially, $\mathcal{P} = \emptyset$.
2. While $\mathcal{P}$ is not a feasible capacitated set cover:
   (a) Let $S = \arg\min_{S:\ f_{\mathcal{P}}(S) > 0} \frac{w(S)}{f_{\mathcal{P}}(S)}$.
   (b) Add S to $\mathcal{P}$.

Wolsey [27] showed using the *dual fitting* technique that Algorithm Greedy Cover achieves an approximation factor of $O(\log(\max_S |S|))$. We show a simpler and a more intuitive charging scheme that proves the same result.

Let $\mathcal{T} \subseteq \mathcal{S}$ be a collection of sets, and let $C \subseteq \mathcal{T} \times E$ be a feasible partial cover. Denote by $|C|$ the number of elements covered by C. We can assume without loss of generality that no element is covered by more than one set in $\mathcal{T}$. For each $\mathcal{T}' \subseteq \mathcal{T}$, we denote by $C_{\mathcal{T}'}$ the projection of C on $\mathcal{T}'$, and by $f_C(\mathcal{T}')$ the number of elements covered by sets belonging to $\mathcal{T}'$ in C. We need the following lemma.

Lemma 6 *Consider an instance of the set cover problem with hard capacities. Let $\mathcal{T}$ be a feasible cover and let $\mathcal{T}_1, \mathcal{T}_2$ be a partition of $\mathcal{T}$ into two disjoint subsets. Then, there is a feasible cover $C \subseteq \mathcal{T} \times E$, such that all the elements are covered in C and $f_C(\mathcal{T}_1) = f(\mathcal{T}_1)$.*

Proof: Let $C \subseteq \mathcal{T} \times E$ be a feasible cover, where each element $e \in E$ is covered by some $S \in \mathcal{T}$, and assume that $f_C(\mathcal{T}_1) < f(\mathcal{T}_1)$. Let $C' \subseteq \mathcal{T}_1 \times E$ be a feasible partial cover, where $f_{C'}(\mathcal{T}_1) = f(\mathcal{T}_1)$. Since C' is a partial cover, some elements may not be covered in C'. We gradually change the cover C, while maintaining its feasibility, until the lemma is satisfied. Perform the following procedure:

While $f_C(\mathcal{T}_1) < f(\mathcal{T}_1)$:

1. Let $S \in \mathcal{T}_1$ be a set, such that S covers fewer elements in C than it does in C'. There is at least one such set since $f_C(\mathcal{T}_1) < f(\mathcal{T}_1)$.
2. Let $e \in E$ be an element covered by S in C', but covered by some $T \neq S$ in C. (Note that T can belong to either $\mathcal{T}_1$ or $\mathcal{T}_2$).
3. Change C so that e is covered by S, i.e., remove (T, e) and add (S, e) to C.

It is clear that we can perform the procedure and maintain a feasible cover C, while $f_C(\mathcal{T}_1) < f(\mathcal{T}_1)$. Once a pair $(S, e) \in C'$ is added to C, it remains there till the end of the procedure. Thus, the number of iterations is bounded by $|C'|$ and is therefore finite. Upon termination of the procedure, we have a cover C that satisfies the conditions of the lemma. □

We now proceed with analyzing Algorithm Greedy Cover. Denote the solution computed by Algorithm Greedy Cover by $\mathcal{P} = \{S_1, S_2, \ldots, S_k\}$, and assume that the sets are added to the solution by the algorithm in this order. For each i, $0 \leq i \leq k$, let $\mathcal{P}_i = \{S_1, S_2, \ldots, S_i\}$ be the solution at the end of iteration i. Let OPT be an optimal solution. We "replay" the algorithm, while charging the costs of the sets added to $\mathcal{P}$ by Algorithm Greedy Cover to the sets in OPT.

Start with $\mathcal{P}_0 = \emptyset$. For each $S \in OPT$, let $a_0(S)$ be the number of elements covered by S in OPT (assuming every element is covered by exactly one set in OPT). For each iteration i of Algorithm Greedy Cover, new values $a_i(S)$ of sets in $OPT \setminus \mathcal{P}_i$ are defined. The following invariant holds throughout the analysis: we can cover all the elements by the sets in $OPT \cup \mathcal{P}_i$, even if the capacities of sets $S \in OPT \setminus \mathcal{P}_i$ are restricted to be $a_i(S)$.

The invariant is clearly true for $\mathcal{P}_0$ and a_0. Consider iteration i of Algorithm Greedy Cover. We add set S_i to the solution. Since the invariant holds for $\mathcal{P}_{i-1}$, a_{i-1}, the collection of sets $\mathcal{P}_i \cup OPT$ is a feasible cover, even if we restrict the capacities of sets $S \in OPT \setminus \mathcal{P}_i$ to be $a_{i-1}(S)$. By Lemma 6, there is a feasible cover $C \subseteq (\mathcal{P}_i \cup OPT) \times E$, where the sets in $\mathcal{P}_i$ cover exactly $f(\mathcal{P}_i)$ elements and each set $S \in OPT \setminus \mathcal{P}_i$ covers at most $a_{i-1}(S)$ elements. For each $S \in OPT \setminus \mathcal{P}_i$, define $a_i(S)$ to be the number of elements covered by S in C. Note that $a_i(S) \leq a_{i-1}(S)$.

If $S_i \in OPT$, we do not charge any sets for its cost, since OPT also pays for it. Otherwise, suppose $f_{\mathcal{P}_{i-1}}(S_i) = n_i$. The number of elements covered by sets in $OPT \setminus \mathcal{P}_i$ in C is $\sum_{S \in OPT \setminus \mathcal{P}_i} a_{i-1}(S) - n_i$. Therefore, $\sum_{S \in OPT \setminus \mathcal{P}_i} (a_{i-1}(S) - a_i(S)) = n_i$. We charge each $S \in OPT \setminus \mathcal{P}_i$ with $\frac{w(S_i)}{n_i} \cdot (a_{i-1}(S) - a_i(S))$. Note that the total cost charged to the sets in OPT in this iteration is exactly $w(S_i)$.

We now bound the cost charged to each $S \in OPT$. If $S \in \mathcal{P}$, let j denote the last iteration before S is added to $\mathcal{P}$ (i.e., S is added to $\mathcal{P}$ at iteration $j+1$). Otherwise, let j be the first iteration after which $a_j = 0$ (note that the equality holds for the last iteration). For each $i < j$, at the beginning of iteration i, $f_{\mathcal{P}_{i-1}}(S) \geq a_{i-1}(S)$. This follows from the way the value of $a_{i-1}(S)$ is determined. Since ALGORITHM GREEDY COVER chooses a set other than S in this iteration, $\frac{w(S_i)}{n_i} \leq \frac{w(S)}{a_{i-1}(S)}$. Therefore, the total value charged to S is:

$$\sum_{i=1}^{j} (a_{i-1}(S) - a_i(S)) \frac{w(S_i)}{n_i} \leq w(S) \sum_{i=1}^{j} \frac{(a_{i-1}(S) - a_i(S))}{a_{i-1}(S)}.$$

Observe that for each $i : 1 \leq i \leq j$, the term $\frac{a_{i-1}(S) - a_i(S)}{a_{i-1}(S)}$ can be written as

$$\sum_{\ell=a_i(S)+1}^{a_{i-1}(S)} \frac{1}{a_{i-1}(S)} \leq \sum_{\ell=a_i(S)+1}^{a_{i-1}(S)} \frac{1}{\ell}.$$

Thus, the value charged to S is bounded by $w(S)H(|S|)$ and the total cost of the solution is at most $OPT(1 + \ln(\max_S |S|))$.

6 Extensions

6.1 Submodular Set Cover

Recall the *submodular set cover* problem. Let f be a real valued function defined over all subsets of a finite set E of elements. Function f is called *non-decreasing* if $f(S) \leq f(T)$ for all $S \subseteq T \subseteq E$, and *submodular* if $f(S) + f(T) \geq f(S \cap T) + f(S \cup T)$ for all $S, T \subseteq E$. The input to the submodular set cover problem is a family $\mathcal{S}$ of subsets of E together with a a non-negative cost function. There is a non-decreasing non-negative submodular function f defined over all collections of the input sets. The goal is to find a minimum cost collection $\mathcal{P} \subset \mathcal{S}$ of sets, such that $f(\mathcal{P}) = f(\mathcal{S})$.

It is not hard to show that the natural linear programs for the set cover problem with hard capacities, as well as the more general submodular set cover problem, have an unbounded integrality gap. We note that ALGORITHM GREEDY COVER can be applied to the general submodular set cover problem as well. Indeed, Wolsey [27] showed using the dual fitting technique that ALGORITHM GREEDY COVER achieves a $1 + O(\log(f_{\max}))$-approximation for this problem, where $f_{\max} = \max_{S \in \mathcal{S}} f(\{S\})$. For the sake of completeness, we describe Wolsey's linear program (SSC) below. As before, for $\mathcal{T} \subseteq \mathcal{S}$ and $S \in \mathcal{S}$, $f_{\mathcal{T}}(S) = f(\mathcal{T} \cup \{S\}) - f(\mathcal{T})$. For each set $S \in \mathcal{S}$, there is an indicator variable $x(S)$ showing whether S is in the solution. The goal is to minimize the solution cost, i.e., $\sum_{S \in \mathcal{S}} w(S)x(S)$.

Consider now some collection of input sets $\mathcal{T} \subset \mathcal{S}$, and suppose $f(\mathcal{S}) - f(\mathcal{T}) > 0$. Let $\mathcal{P}$ denote any solution and $\mathcal{P}' = \mathcal{P} \cap (\mathcal{S} \setminus \mathcal{T})$. As $f(\mathcal{S}) - f(\mathcal{T}) > 0$, some sets in $\mathcal{S} \setminus \mathcal{T}$ must be in the solution, i.e., $\mathcal{P}'$ is non-empty. Moreover, $f(\mathcal{P}' \cup \mathcal{T}) - f(\mathcal{T}) \geq f(S) - f(\mathcal{T})$. Note

that due to the submodularity of f, $\sum_{S\in\mathcal{P}'}(f(\{S\}\cup\mathcal{T})-f(\mathcal{T}))\geq f(\mathcal{P}'\cup\mathcal{T})-f(\mathcal{T})$, and therefore $\sum_{S\in\mathcal{P}'}f_{\mathcal{T}}(S)\geq f(\mathcal{S})-f(\mathcal{T})$ must hold. This condition is expressed by the set of constraints (1).

$$\min \quad \sum_{S\in\mathcal{S}} w(S)x(S) \tag{SSC}$$

s.t. (1)

$$\sum_{S\in\mathcal{T}} f_{\mathcal{T}}(S)x(S) \;\geq\; f(\mathcal{S})-f(\mathcal{T}) \qquad \forall\mathcal{T}\subseteq\mathcal{S}$$

$$x(S) \;\geq\; 0 \qquad \forall S\in\mathcal{S}$$

We show that our analysis of Algorithm Greedy Cover can be extended to prove a similar approximation guarantee for the submodular set cover problem.

Denote the solution computed by Algorithm Greedy Cover by $\mathcal{P}=\{S_1,S_2,\ldots,S_k\}$, and assume that the sets are added to the solution by the algorithm in this order. For each i, $0\leq i\leq k$, let $\mathcal{P}_i=\{S_1,S_2,\ldots,S_i\}$ be the solution at the end of iteration i. Let OPT be an optimal solution. Choose an arbitrary ordering of sets in OPT, such that the sets in $\mathcal{P}\cap OPT$ appear at the beginning, in the same order in which they are added to the solution by Algorithm Greedy Cover. For each j, let $\mathcal{T}_j$ denote the first j sets in OPT.

Consider some iteration i of the algorithm. Let $S\in OPT$, and assume that S is the jth set in OPT. Define:

$$a_i(S)=f(\mathcal{T}_j\cup\mathcal{P}_i)-f(\mathcal{T}_{j-1}\cup\mathcal{P}_i)$$

If $S_i\in OPT$, then we do not have to charge its cost to sets in OPT. Otherwise, each set $S\in OPT\setminus\mathcal{P}_i$ is charged with $\frac{w(S_i)}{f(\mathcal{P}_i)-f(\mathcal{P}_{i-1})}(a_{i-1}(S)-a_i(S))$.

Observe that the amount charged in iteration i to sets in $OPT\setminus S_i$ is at least $w(S_i)$. This is true, since:

$$\begin{aligned}\sum_{S\in OPT\setminus\mathcal{P}_i}(a_{i-1}(S)-a_i(S)) &= \sum_j(f(\mathcal{T}_j\cup S_{i-1})-f(\mathcal{T}_{j-1}\cup S_{i-1}))\\ &\quad-\sum_j(f(\mathcal{T}_j\cup S_i)-f(\mathcal{T}_{j-1}\cup S_i))\\ &= f(OPT\cup\mathcal{P}_{i-1})-f(\mathcal{P}_{i-1})-f(OPT\cup\mathcal{P}_i)+f(\mathcal{P}_i)\\ &= f(OPT)-f(OPT)+f(\mathcal{P}_i)-f(\mathcal{P}_{i-1})\\ &= f(\mathcal{P}_i)-f(\mathcal{P}_{i-1})\end{aligned}$$

Observe also, that at the beginning of iteration $i+1$, for each $S\in OPT\setminus S_i$, $f(\{S\})\geq a_i(S)$: Since $f(\{S\})\geq f(\{S\}\cup\mathcal{P}_i)-f(\mathcal{P}_i)$ (due to the submodularity of f), it is enough to show that

$$f(\{S\} \cup \mathcal{P}_i) - f(\mathcal{P}_i) \geq f(\mathcal{T}_j \cup \mathcal{P}_i) - f(\mathcal{T}_{j-1} \cup \mathcal{P}_i).$$

Rearranging the sides, this is equivalent to

$$f(\{S\} \cup \mathcal{P}_i) + f(\mathcal{T}_{j-1} \cup \mathcal{P}_i) \geq f(\mathcal{T}_j \cup \mathcal{P}_i) + f(\mathcal{P}_i)$$

which holds by the submodularity of f.

Using the same reasoning as in the case of the proof of the set cover with hard capacities, the total amount charged to any $S \in OPT$ is at most $w(S)H(|S|)$, and the total cost of the solution is bounded by $OPT(1 + \ln(\max_S f(\{S\})))$.

6.2 Multi-set Multi-cover

An interesting special case of the submodular set cover problem is the multi-set multi-cover problem. In this problem, the input sets are actually multi-sets, i.e. an element $e \in E$ can appear in $S_j \in \mathcal{S}$ more than once, and the elements have splittable demands. An integer programming formulation of the multi-set multi-cover problem with unbounded set capacities is the following: $\min\{w^T x \mid Ax \geq d, 0 \leq x \leq b, x \in Z\}$. The constraints $x \leq b$ are called multiplicity constraints, and they generally make covering problems much harder, as the natural linear programming relaxation has an unbounded integrality gap. Dobson [10] gives a combinatorial greedy $H(\max_{1 \leq j \leq m} \sum_{1 \leq i \leq n} A_{ij})$-approximation algorithm, where $H(t)$ is the tth harmonic number. This is a logarithmic approximation factor for the case where A is a $\{0,1\}$ matrix (set multi-cover), but can be as bad as a polynomial approximation bound in the general case (multi-set multi-cover). Recently, Carr, Fleischer, Leung and Phillips [5] gave a p-approximation algorithm, where p denotes the maximum number of variables in any constraint. Their algorithm is based on a linear relaxation in the spirit of (SSC). Using similar ideas for strengthening the linear program, Kolliopoulos and Young [20] obtained an $O(\log n)$-approximation.

We can assume again that the multiplicities of the sets are unit, by viewing each copy of each set as a distinct set. We can then define, for each collection $\mathcal{T}$ of input sets, $f(\mathcal{T})$ to be the maximum number of elements that can be covered by $\mathcal{T}$ (with the capacity constraints). It is not hard to see that f is a non-negative non-decreasing submodular function, and thus the algorithm of Wolsey and our analysis hold for this case.

Notice that the number of sets now is not necessarily polynomial (as the initial set multiplicities $m(S)$ for $S \in \mathcal{S}$ are not necessarily polynomial in the input size. However, the function $f(\mathcal{T})$ can still be computed in polynomial time. Thus, Algorithm Greedy Cover can be implemented to run in polynomial time, achieving an approximation ratio of $O(\log(\max_{S \in \mathcal{S}} |S|))$.

Acknowledgement

We would like to thank the anonymous referee for pointing out the reduction in Section 4.2.

References

[1] E. Balas and M Padberg. Set partitioning: a survey. *SIAM Review*, 18:710–760, 1976.

[2] J. Bar-Ilan, G. Kortsarz, and D. Peleg. Generalized submodular cover problems and applications. In *Proceedings of the 4th Israel Symposium on Computing and Systems* 1996, pp. 110-118.

[3] R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27-45, 1985.

[4] Y. Bartal, M. Charikar and D. Raz. Approximating min-sum k-clustering in metric spaces. In *Proceedings of 33rd ACM Symposium on Theory of Computing*, pages 11-20, 2001.

[5] R.D. Carr, L.K. Fleischer, V.J. Leung and C.A. Phillips. Strengthening integrality gaps for capacitated network design and covering problems. In *Proceedings of the 11th ACM-SIAM Symposium on Discrete Algorithms*, pp. 106–115, 2000.

[6] M. Charikar, S. Guha, E. Tardos and D. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proceedings of the 31st Annual ACM Symposium on the Theory of Computing*, pp. 1-10, 1999.

[7] N. Christofides and S. Korman. A computational survey of methods for the set covering problem. *Management Science*, 21:591–599, 1975.

[8] J. Chuzhoy and Y. Rabani. Approximating k-median with non-uniform capacities. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms*, 2005.

[9] V. Chvátal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

[10] G. Dobson. Worst-case analysis of greedy heuristics for integer programming with non-negative data. *Mathematics of Operations Research*, 7(4): 515-531, 1972.

[11] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[12] Rajiv Gandhi, Eran Halperin, Samir Khuller, Guy Kortsarz, Aravind Srinivasan. An improved approximation algorithm for vertex cover with hard capacities. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, pp. 164–175, 2003.

[13] R. Gandhi, S. Khuller, S. Parthasarathy, A. Srinivasan. Dependent rounding in bipartite graphs. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 323–332, 2002.

[14] M. R. Garey and D. S. Johnson. Computers and intractability: a guide to the theory of NP-completeness. W.H. Freeman and Company, 1979.

[15] R.S. Garfinkel and G.L. Nemhauser. Optimal set covering: a survey. In *Perspectives on optimization: a collection of expository articles*, A.M. Geofrion, ed., 164–183, 1972.

[16] S. Guha, R. Hassin, S. Khuller, and E. Or. Capacitated vertex covering with applications. In *Proceedings of the 13th ACM-SIAM Symposium on Discrete Algorithms*, pp. 858–865, 2002.

[17] D.S. Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM Journal on computing*, 11:555-556, 1982.

[18] D.S. Hochbaum (editor). Approximation algorithms for NP-hard problems. PWS Publishing Company, 1996.

[19] D. S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.*, 9:256-278, 1974.

[20] S.G. Kolliopoulos and N.E. Young. Tight approximation results for general covering integer programs. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pp. 522–528, 2001.

[21] L. Lovász. On the ratio of optimal and fractional covers. *Discrete Mathematics*, 13:383-390, 1975.

[22] P. B. Mirchandani and R. L. Francis (editors). Discrete location theory. Wiley Interscience, 1990.

[23] M.W. Padberg. Covering, packing and knapsack problems. *Annals of Discrete Mathematics* 4:265–287, 1979.

[24] D. B. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for the facility location problem. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, pp. 265-274, 1997.

[25] M. Pál, É. Tardos, T. Wexler. Facility location with nonuniform hard capacities. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pp. 329–338, 2001.

[26] V. V. Vazirani. Approximation algorithms. Springer-Verlag, 2001.

[27] L. A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2:385-393, 1982.