Exact Algorithms for Set Multicover and Multiset Multicover Problems

Qiang-Sheng Hua[1], Dongxiao Yu[1], Francis C.M. Lau[1], and Yuexuan Wang[2]

[1] Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong, China
qshua@cs.hku.hk, dxyu@cs.hku.hk, fcmlau@cs.hku.hk
[2] Institute for Theoretical Computer Science, Tsinghua University, Beijing, 100084, China
wangyuexuan@tsinghua.edu.cn

Abstract. Given a universe N containing n elements and a collection of multisets or sets over N, the multiset multicover (MSMC) or the set multicover (SMC) problem is to cover all elements at least a number of times as specified in their coverage requirements with the minimum number of multisets or sets. In this paper, we give various exact algorithms for these two problems, with or without constraints on the number of times a multiset or set may be picked. First, we can exactly solve the MSMC without multiplicity constraints problem in $O(((b+1)(c+1))^n)$ time where b and c ($c \leq b$ and $b \geq 2$) respectively are the maximum coverage requirement and the maximum number of times that each element can appear in a multiset. To our knowledge, this is the first known exact algorithm for the MSMC without multiplicity constraints problem. Second, we can solve the SMC without multiplicity constraints problem in $O((b+2)^n)$ time. Compared with the two recent results in [Hua et al., Set Multi-covering via inclusion-exclusion, Theoretical Computer Science, 410(38-40):3882-3892 (2009)] and [Nederlof, J.: Inclusion Exclusion for hard problems. Master Thesis. Utrecht University, The Netherlands (2008)], we have given the fastest exact algorithm for the SMC without multiplicity constraints problem. Finally, we give the first known exact algorithm for the MSMC or the SMC with multiplicity constraints problem in $O((b+1)^n|F|)$ time and $O((b+1)^n|F|)$ space where $|F|$ denotes the total number of given multisets or sets over N.

1 Introduction

In this paper, we study exact algorithms for the set multicover(SMC) and the multiset multicover(MSMC) problems. In the set multicover problem, we are given a universe N of n elements and a family of sets $F = \{S_1, \cdots, S_{|F|}\}$ where each S_i is a subset of N, and we need to find a minimum cardinality sub-family $F' \subseteq F$ such that each element $i \in N$ is covered b_i integral number of times. In the multiset multicover problem, we are given a collection of multisets instead of a collection of sets. Here a multiset S_i contains a specified number of copies of each element $i \in N$. Note that in order to minimize the total number of picked sets or multisets, each set or multiset can be chosen a number of times. Here if we further require that each set or multiset can be chosen at most a specified number of times, the SMC or the MSMC problem becomes the SMC or the MSMC with multiplicity constraints problem. Much attention has been given to approximation algorithms for these problems in the past several decades [11,10,3,4,5]. Besides approximate algorithms, recently there has also been some effort in understanding how fast we can exactly solve these covering problems.

By using the inclusion-exclusion principle and a fast zeta transform technique, Björklund et al. [1] have shown that the set cover problem can be exactly solved in $O^*(2^n)$ time using $O^*(2^n)$ space. Here, using the $O^*(f(n))$ notation we omit a $(\log f(n))^{O(1)}$ factor. Based on this observation, they proposed a family of exact algorithms for the set partitioning problems which improve all the previous algorithms. Later on they showed that similar faster algorithms can also be obtained by using the so called fast subset convolution [2]. Very recently, Hua et al. and Nederlof have independently given their exact algorithms for the set multicover problem in [6] and [9], respectively. In [6], we show that the set multicover problem can be exactly solved in $O^*((2b)^n)$ time using $O^*((b+1)^n)$ space or in $O^*(2^{O(bn^2)})$ time with polynomial space. In [9], based on a novel counting formulation, the set multicover problem can be solved in $O((b+1)^n|F_{mc}|)$ time and polynomial space. Here $|F_{mc}|$ means the total number of given sets. Although this result greatly outperforms the polynomial space exact algorithm given in [6], as discussed in [8], the algorithm given in [6] can also exactly count the number of set multicovers that satisfy the coverage requirements. We are not aware of any known exact algorithms for solving the multiset multicover, the set multicover with multiplicity constraints and the multiset multicover with multiplicity constraints problems. Some key notations and their definitions are given in Table 1.

Table 1. Some key notations and definitions

Notations	Definitions
$N = \{1, \cdots, n\}$	The universe set.
F_{ms} (F_{mc}) (F_{sc})	A collection of (multi)sets in a multiset multicover (set multicover) (set cover) instance.
$B = (b_1, \cdots, b_n)$	The positive integral coverage requirement vector indicating that each element i must be covered at least b_i times and $\mathbf{b} = \max_{i \in N}(b_i)$.
$C = (F_{ms}(S, i))$	The vector indicating the number of times that each element $i \in N$ appears in each multiset $S \in F_{ms}$ and $\mathbf{c} = \max_{i \in N, S \in F_{ms}}(F_{ms}(S, i))$. We assume $c \leq b$ and $b \geq 2$.
$c_k(F_{sc})$	The number of k-tuples $< s_1, \cdots, s_k >$ over F_{sc} such that the union of the sets $\bigcup_{i=1}^{k} s_i$ without removing duplicate elements satisfies the coverage requirements.

Table 2. Summary of exact algorithms for various covering problems

Problem	Time	Space	Ref.
Set Cover(SC)	$O^*(2^n\|F_{sc}\|)^1$	Polynomial	[1]
Set Cover(SC)	$O^*(2^n)$	$O^*(2^n)$	[1]
Set Multicover(SMC)	$O^*((2b)^n)$	$O^*((b+1)^n)$	[6]
Set Multicover(SMC)	$O^*(2^{O(bn^2)})$	Polynomial	[6]
Set Multicover(SMC)	$O^*((b+1)^n\|F_{mc}\|)^1$	Polynomial	[9]
Set Multicover(SMC)	$O((b+2)^n)$	See note^2	Here
Multiset Multicover(MSMC)	$O^*((b+1)^n\|F_{ms}\|)^1$	Polynomial	Here
Multiset Multicover(MSMC)	$O((b+1)^n\|F_{ms}\|)$	See note^3	Here
SMC with Multiplicity Constraints	$O((b+1)^n\|F_{mc}\|)$	$O((b+1)^n\|F_{mc}\|)$	Here
MSMC with Multiplicity Constraints	$O((b+1)^n\|F_{ms}\|)$	$O((b+1)^n\|F_{ms}\|)$	Here

^{1} It's easy to know that $|F_{sc}| = |F_{mc}| = O(2^n), |F_{ms}| = O((c+1)^n)$.
^{2} $\max\{\binom{n}{m_1}(b+1)^{n-m_1}, \binom{n}{m_2}(b+1)^{n-m_2}\}$, where $m_1 = \lfloor\frac{n+1}{b+2}\rfloor$ and $m_2 = \lceil\frac{n+1}{b+2}\rceil$.
^{3} $\max\{\binom{n}{m_1}c^{m_1}(b+1)^n, \binom{n}{m_2}c^{m_2}(b+1)^n\}$, where $m_1 = \lfloor\frac{c(n+1)}{c+1}\rfloor$ and $m_2 = \lceil\frac{c(n+1)}{c+1}\rceil$.

Our Results. In this paper, we give: (1) the fastest exact algorithm for the set multicover problem; (2) the first known exact algorithm for the multiset multicover problem; and (3) the first known exact algorithm for the set or multiset multicover with multiplicity constraints problem.

Table 2 summarizes previous related results and those given in this paper.

Preliminaries. The Inclusion-Exclusion Principle. [folklore]: Let S be a finite set with subsets $A_1, A_2, ..., A_n \subseteq S$, and with the convention that$\cap_{i\in\emptyset} A_i = S$, then we know the number of elements in S which lie in none of the A_i is

$$|\bigcap_{i=1}^{n} \overline{A_i}| = \sum_{X\subseteq N} (-1)^{|X|} \cdot |\bigcap_{i\in X} A_i| \tag{1}$$

Counting Set Covers. By using the inclusion-exclusion principle (Equation 1), Björklund et al. [1] prove that the number $c_k(F_{sc})$ of set covers can be calculated through Equation 2. Here $a_{sc}(X)$ denotes the number of sets in F_{sc} which avoid (do not cover) any element in the set $X \subseteq N$.

$$c_k(F_{sc}) = \sum_{X\subseteq N} (-1)^{|X|} a_{sc}(X)^k \tag{2}$$

Solving the Set Cover Problem via Counting Set Covers. According to the definition of $c_k(F_{sc})$ (c.f. Table 1), we can see that, in order to find the minimum number of sets that satisfy the coverage requirement, we just need to find the minimum k value that satisfy $c_k(F_{sc}) > 0$ using binary search. This is a standard approach which was first used in [1]. Hua et al. [6] also employed a similar approach for exactly solving the set multicover problem, i.e., searching the minimum k that guarantees a positive $c_k(F_{mc})$ number of set multicovers. In this paper, similar to what is done in [9], we will not directly count the number of multicovers; instead, we will first transform the multicover problems into the set or multiset cover problem, and then we search for the minimum k value that satisfies a positive number of set covers.

2 Two Formulations for Counting the Transformed Set Covers

We first explain how to transform the set or multiset multicover problem into the corresponding set cover problem, as follows. For each element $i \in N$ with b_i coverage requirement, we replace this element with b_i replicated elements. This means that the universe N with n elements will be augmented to become a new universe N' with at most bn elements. Accordingly, the collection of (multi)sets F_{mc} or F_{ms} will be respectively expanded into a new collection of (multi)sets F'_{mc} or F'_{ms}. For example, if $b_i = 2$ and $b_j = 1$, then the element i will be replaced with element i_1 and i_2; similarly, the element j will be replaced with the element j_1 or we just say that it remains unchanged. Then the set $\{i, j\}$ will be replaced with two new sets $\{i_1, j_1\}$ and $\{i_2, j_1\}$. Accordingly, the multiset $\{i, i, j\}$ will be replaced with three new multisets $\{i_1, i_1, j_1\}$, $\{i_1, i_2, j_1\}$ and $\{i_2, i_2, j_1\}$. In this case, we can count the $c_k(F'_{mc})$ number of set covers for the set multicover problem and can count the $c_k(F'_{ms})$ number of multiset covers for the multiset multicover problem.

Now a straightforward formulation for $c_k(F'_{mc})$ or $c_k(F'_{ms})$ is to directly apply the $c_k(F_{sc})$ formula given in Equation 2. By using $a_{mc}(X)$ or $a_{ms}(X)$ to denote the number of sets or multisets in F'_{mc} or F'_{ms} that do not cover any element in X, we can give the similar formulations for counting the transformed (multi)set covers, by Equations 3 and 4.

$$c_k(F'_{mc}) = \sum_{X \subseteq N'} (-1)^{|X|} a_{mc}(X)^k \tag{3}$$

$$c_k(F'_{ms}) = \sum_{X \subseteq N'} (-1)^{|X|} a_{ms}(X)^k \tag{4}$$

However, we can easily see that the straightforward formulations for calculating $c_k(F'_{mc})$ and $c_k(F'_{ms})$ are extraordinarily inefficient in terms of time complexities. For example, Equation 3 immediately yields an $O^*(|F'_{mc}|2^{bn})$ time and polynomial space algorithm. So in this paper, we need to employ another kind of efficient formulations. This new formulation for the set multicover problem was first given by Nederlof in [9]. In this section, we extend it to the multiset multicover problem.

These new formulations are obtained by taking advantage of the symmetry information behind Equations 3 and 4. By analyzing all the subsets X used in these two equations, and since the augmented universe N' is composed by many replicated elements for each single element with non-unit coverage requirement, we can see that there are many symmetric subsets $X \subseteq N'$ in the sense that this family of subsets $\{X\}$ have the same $a_{mc}(X)$ or $a_{ms}(X)$ values. From this

Table 3. Some notations for counting the transformed set covers

Notations	Definitions
$Y(X) = (Y(1), \cdots, Y(i))$[1]	Y is a **nonnegative integer** function on the set $X \subseteq N$.
$Y(X) \preceq B(X)$	For each $i \in X$, we have $Y(i) \leq b_i$.
F_{mc}^{Y} (F_{ms}^{Y})	A new collection of (multi)sets constructed on F_{mc} (F_{ms}) where each element $i \in N$ is replaced with $Y(i)$ elements. If $Y(i) = 0$ then any (multi)set $S \in F_{mc}$ (F_{ms}) which covers element i will be deleted.

[1] We will use Y instead of $Y(X)$ in a clear context.

observation, we can conclude that, in order to lower the time complexity, it is not necessary to calculate the $a_{mc}(X)$ or $a_{ms}(X)$ value anew for each subset $X \subseteq N'$. Instead, we can just calculate the $a_{mc}(X)$ or $a_{ms}(X)$ value once for all symmetric subsets $X \subseteq N'$. Now before delving into more details, we need to introduce some necessary notations in Table 3.

With these notations, we know that F_{mc}^{B-Y} and F_{ms}^{B-Y} respectively denote the new collection of sets or multisets constructed on either F_{mc} or F_{ms} where each element $i \in N$ is replaced with $b_i - Y(i)$ elements. From this we can see that for each $Y(N) \preceq B(N)$, F_{mc}^{B-Y} or F_{ms}^{B-Y} can group a class of symmetric subsets $X \subseteq N'$ that have the same $a_{mc}(X)$ or $a_{ms}(X)$ values (c.f. Equations 3 and 4). For example, if we set $Y = (b_1, \cdots, b_n)$, then $F_{mc}^{Y} = F'_{mc}$ and $F_{ms}^{Y} = F'_{ms}$.

From the above analysis, we can give the new formulations for calculating $c_k(F'_{mc})$ and $c_k(F'_{ms})$ in Equations 5 and 6. As mentioned earlier, a similar formulation for the set multicover problem was first used in [9].

$$c_k(F'_{mc}) = \sum_{Y \preceq B} (-1)^{\sum_{1 \leq i \leq n} Y(i)} \Big(\prod_{1 \leq i \leq n} \binom{b_i}{Y(i)} \Big) (|F_{mc}^{B-Y}|)^k \tag{5}$$

$$c_k(F'_{ms}) = \sum_{Y \preceq B} (-1)^{\sum_{1 \leq i \leq n} Y(i)} \Big(\prod_{1 \leq i \leq n} \binom{b_i}{Y(i)} \Big) (|F_{ms}^{B-Y}|)^k \tag{6}$$

Then the remaining question is how to calculate $|F_{mc}^{Y}|$ $(|F_{ms}^{Y}|)$, i.e., the new number of sets (multisets) in F_{mc} (F_{ms}). But before this, we need to give a helping lemma.(For an example, please refer to the first paragraph of this section.)

Lemma 1. *If an element a is replaced with r number of replicated elements, then the multiset which only contains s number of elements a will be expanded into* $\binom{r+s-1}{r-1}$ *number of new multisets.*

According to lemma 1, we give the formula for calculating $|F_{ms}^{Y}|$ in Equation 7. Here S denotes a multiset belonging to F_{ms} and $t(S)$ is a set composed by different elements in the set S. Also c_j denotes the number of times that the element j appears in a multiset S. If for all $j \in t(S)$ we set $c_j = 1$, then we can obtain a similar formula for calculating $|F_{mc}^{Y}|$ in Equation 8.

$$|F_{ms}^{Y}| = \sum_{S \in F_{ms}} \prod_{j \in t(S)} \binom{c_j + b_j - Y(j) - 1}{b_j - Y(j) - 1} \tag{7}$$

$$|F_{mc}^Y| = \sum_{S' \in F_{mc}} \prod_{j \in S'} (b_j - Y(j)) \tag{8}$$

With these two new formulations, for each $Y \preceq B$, by directly computing $|F_{ms}^Y|$ or $|F_{mc}^Y|$, we have Theorem 1.

Theorem 1. *The multiset multicover or the set multicover problem can be solved in $O^*((b+1)^n|F_{ms}|)$ ($O^*((b+1)^n|F_{mc}|)$) time using polynomial space.*

3 Dynamic Programming Based Algorithms For Calculating All F_{ms}^Y and All F_{mc}^Y

We first give some necessary notations in Table 4. Then we compute all F_{ms}^Y and F_{mc}^Y values through the following Algorithm 1 and Algorithm 2, respectively. Note that the multiplicative factors used in the recursions (steps 16 and 17 in Algorithm 1) are derived from the helping lemma 1.

Table 4. Some notations for calculating F_{ms}^Y and F_{mc}^Y

Notations	Definitions
$v(X) = (v(1), \cdots, v(i))$	v is a **positive integer** function on the set $X \subseteq N$.
$v(X) \preceq (c, \cdots, c)^{\|X\|}$	For each $i \in X$, we have $v(i) \leq c$ and there are $\|X\|$ c.
$(v(X), m)$	We append the m value at the end of the $v(X)$ vector.
X^Y	Replace each element $i \in X$ with $Y(i)$ elements.
$X^{Y(i)-1}$	The same as X^Y except that the element i is replaced with $Y(i) - 1$ elements.
$c(X^1, (N \backslash X)^Y)$	The number of sets in $F_{mc}^{Y'}$ that include all the elements in X. Here the new collection of sets $F_{mc}^{Y'}$ is constructed on F_{mc} as follows: each element $i \in X$ remains unchanged and each element $i \in N \backslash X$ is replaced with $Y(i)$ elements.
$d(X^Y, (N \backslash X)^Y, v(X))$	The number of multisets in $F_{ms}^{Y'}$ that include all the elements in X and each $i \in X$ appears $v(i)$ times in the multiset. Here $F_{ms}^{Y'}$ is constructed on F_{ms} as follows: each element $i \in X$ is replaced with $Y(i)$ elements and each element $j \in N \backslash X$ is replaced with $Y(j)$ elements.

The time and space complexities for Algorithm 1 are given in Lemma 2.

Lemma 2. *For all $Y \preceq B$, the F_{ms}^Y values can be calculated in $O(((b+1)(c+1))^n)$ time and $\max\{\binom{n}{m_1}c^{m_1}(b+1)^n, \binom{n}{m_2}c^{m_2}(b+1)^n\}$ space where $m_1 = \lfloor \frac{c(n+1)}{c+1} \rfloor$ and $m_2 = \lceil \frac{c(n+1)}{c+1} \rceil$.*

Proof. We first analyze the time and space used from Step 1 to Step 8. The used space can be calculated from the formula $\sum_{m=0}^{n} \binom{n}{m} c^m = (c+1)^n$. Since all the $d(X^1, (N \backslash X)^0, v(X))$ values can be obtained by scanning F_{ms}, then since $|F_{ms}| = O((c+1)^n)$, this takes only $O((c+1)^n)$ time. Second, we analyze the time

Algorithm 1. Calculating F_{ms}^Y using Dynamic Programming

```
Input: - $F_{ms}$ and the coverage requirement vector $B$
Output: The $F_{ms}^Y$ values for all $Y \preceq B$
 1: For each $X \subseteq N$ do
 2:   If $X$ is not empty then
 3:     For each $v(X) \preceq (c, \cdots, c)^{|X|}$ do
 4:       Set $d(X^1, (N \backslash X)^0, v(X)) = F_{ms}(X)$ where $F_{ms}(X)$ is the indicator function
          which equals 1 if $X \in F_{ms}$ and 0 otherwise
 5:     End For
 6:   If $X$ is an empty set, we just set $d(\emptyset^1, N^0, \emptyset) = 0$.
 7:   Store the $d(X^1, (N \backslash X)^0, v(X))$ value into a look-up table.
 8: End For
 9: For $t$ from $n$ downto 0 do
10:   For each $X \subseteq N$ and $|X| = t$ do
11:     For each $Y(N) \preceq B(N)$ where $Y(N)$ is from $(0, \cdots, 0)^n$ to $(b_1, \cdots, b_n)$ (using
        lexicographic order) do
12:       For each $v(X) \preceq (c, \cdots, c)^{|X|}$ (using lexicographic order) do
13:         If for some $i \in X$ where $Y(i) = 0$ then $d(X^Y, (N \backslash X)^Y, v(X)) = 0$
14:         If for all $i \in X$ we have $Y(i) = 1$ or if $X = \emptyset$ then
15:           If for some $j \in N \backslash X$ where $Y(j) = 1$ then $d(X^Y, (N \backslash X)^Y, v(X)) = d(X^Y, (N \backslash X)^{Y(j)-1}, v(X)) + \sum_{m=1}^{c} d((X \cup \{j\})^Y, (N \backslash (X \cup \{j\}))^Y, (v(X), m))$
16:           If for all $j \in N \backslash X$ we have $Y(j) \geq 2$ then $d(X^Y, (N \backslash X)^Y, v(X)) = d(X^Y, (N \backslash X)^{Y(j)-1}, v(X)) + \sum_{m=1}^{c} \frac{m}{Y(j)-1} d((X \cup \{j\})^{Y(j)-1}, (N \backslash (X \cup \{j\}))^Y, (v(X), m))$
17:         If for some $i \in X$ where $Y(i) \geq 2$ then $d(X^Y, (N \backslash X)^Y, v(X)) = \frac{Y(i)+v(i)-1}{Y(i)-1} \cdot d(X^{Y(i)-1}, (N \backslash X)^Y, v(X))$
18:         Store the calculated $d(X^Y, (N \backslash X)^Y, v(X))$ value into a table
19:       End For
20:     End For
21:   End For
22: Remove all $d(Z^Y, (N \backslash Z)^Y, v(Z))$ values from the table where $|Z| = |X| + 1$
23: End For
24: Return all the $F_{ms}^Y$ values and for each $Y \preceq B$ we have $F_{ms}^Y = d(\emptyset^Y, N^Y, \emptyset)$.
```

and space used from Step 9 to Step 23. The total time used in these steps can be computed using the formula $O(\sum_{m=0}^{n} \binom{n}{m} c^m (b+1)^n) = O((c+1)^n (b+1)^n)$. According to Step 22, the total space used in these steps is $\max_{0 \leq i \leq n} \{ \binom{n}{i} c^i (b+1)^n \}$. From this, we can easily obtain the result. □

When all the F_{ms}^Y values have been stored into a table, according to Lemma 2 and Equation 6, we can see that the time and space complexities for calculating $c_k(F'_{ms})$ are dominated by Algorithm 1. Thus we have Theorem 2. By comparing with Theorem 1, we can see that our algorithm can reduce the time complexity by a polynomial factor. However, this is achieved by paying exponential space. Thus this result leaves room for further improvement.

Algorithm 2. Calculating F_{mc}^Y using Dynamic Programming

```
Input: - $F_{mc}$ and the coverage requirement vector $B$
Output: The $F_{mc}^Y$ values for all $Y \preceq B$
 1: For each $X \subseteq N$ do
 2:   If $X$ is not empty, we set $c(X^1, (N\backslash X)^0) = F_{mc}(X)$ where $F_{mc}(X)$ is the indicator function which equals 1 if $X \in F_{mc}$ and 0 otherwise; If $X$ is an empty set, we set $c(\emptyset^1, N^0) = 0$.
 3:   Store the $c(X^1, (N\backslash X)^0)$ value into a look-up table.
 4: End For
 5: For $t$ from $n$ downto 0 do
 6:   For each $X \subseteq N$ and $|X| = t$ do
 7:     For each $Y(N\backslash X) \preceq B(N\backslash X)$ where $Y(N\backslash X)$ is from $(0,\cdots,0)^{|N\backslash X|}$ to $(b_1,\cdots,b_{|N\backslash X|})$ (using lexicographic order) do
 8:       for some $i \in N\backslash X$ where $Y(i) \neq 0$, we calculate $c(X^1, (N\backslash X)^Y)$ using the recursion $c(X^1, (N\backslash X)^Y) = c(X^1, (N\backslash X)^{Y(i)-1}) + c((X \cup \{i\})^1, (N\backslash(X \cup \{i\}))^Y)$ and then store it into a table
 9:     End For
10:   End For
11: Remove all $c(Z^1, (N\backslash Z)^Y)$ values from the table where $|Z| = |X| + 1$
12: End For
13: Return all the $F_{mc}^Y$ values and for each $Y \preceq B$ we have $F_{mc}^Y = c(\emptyset^1, N^Y)$.
```

Theorem 2. *The multiset multicover problem can be exactly solved in* $O(((b+1)(c+1))^n)$ *time using* $\max\{\binom{n}{m_1}c^{m_1}(b+1)^n, \binom{n}{m_2}c^{m_2}(b+1)^n\}$ *space where* $m_1 = \lfloor \frac{c(n+1)}{c+1} \rfloor$ *and* $m_2 = \lceil \frac{c(n+1)}{c+1} \rceil$.

The time and space complexities for Algorithm 2 are given in Lemma 3.

Lemma 3. *For all* $Y \preceq B$*, the* F_{mc}^Y *values can be calculated in* $O((b+2)^n)$ *time using* $\max\{\binom{n}{m_1}(b+1)^{n-m_1}, \binom{n}{m_2}(b+1)^{n-m_2}\}$ *space where* $m_1 = \lfloor \frac{n+1}{b+2} \rfloor$ *and* $m_2 = \lceil \frac{n+1}{b+2} \rceil$.

Proof. First, both the time and space used from Step 1 to Step 4 equal $O(2^n)$. Then we analyze the time and space complexities of Step 5 to Step 12. The total time used for these steps can be calculated through the formula $\sum_{m=0}^{n} \binom{n}{m}(b+1)^{n-m} = (b+2)^n$. Due to Step 11, the total space used for these steps is $\max_{0\leq i\leq n}\{\binom{n}{i}(b+1)^{n-i})\}$. Summing up the time and space used for these two parts, we can obtain the result. □

Now similar to Theorem 2, we have Theorem 3.

Theorem 3. *The set multicover problem can be solved in* $O((b+2)^n)$ *time using* $\max\{\binom{n}{m_1}(b+1)^{n-m_1}, \binom{n}{m_2}(b+1)^{n-m_2}\}$ *space where* $m_1 = \lfloor \frac{n+1}{b+2} \rfloor$ *and* $m_2 = \lceil \frac{n+1}{b+2} \rceil$.

4 An Exact Algorithm for Set or Multiset Multicover with Multiplicity Constraints Problem

In this section, we turn our focus to the SMC or MSMC with multiplicity constraints problem and give an exact algorithm called *EMCM*. Here we use the multiplicity constraints vector $D = (d_S)$ to indicate the maximum number of times that each multiset (set) $S \in F_{ms}$ (F_{mc}) can be chosen and $\mathbf{d} = \max_{S \in F_{ms}\ (F_{mc})}(d_S)$.

Algorithm 3. *EMCM*: Exact Algorithm for Set or Multiset Multicover with Multiplicity Constraints Problem

```
Input: $F_{ms}$ or $F_{mc}$, the vector $(d_S)$ and the coverage requirement vector $B$
Output: The minimum number of (multi)sets that satisfy $B$ and respect $(d_S)$
1: For each (multi)set $S_i \in F_{ms}$ $(F_{mc})$ where $1 \le i \le |F_{ms}|$ $(|F_{mc}|)$ we define $O((b+1)^n)$ vertices with labels from $(S_i, 0, \cdots, 0)$ to $(S_i, b, \cdots, b)$; We call all the vertices constructed on $S_i$ as level $i$ vertices.
2: Set an initial vertex with label $(S_0, 0, \cdots, 0)$ and we call this vertex as level 0 vertex.
3: For $i = 0$ to $|F_{ms}| - 1$ $(|F_{mc}| - 1)$ do
4:   For $j = 0$ to $d_{S_{i+1}}$ do
5:     For each vertex $(S_i, y_1, \cdots, y_n)$ with non-empty incoming edges (except the level 0 vertex) we add the (j+1)th directed edge with edge weight $j$ to $(S_{i+1}, y_1 + j * z_1, \cdots, y_n + j * z_n)$. Here $z_i$ means (multi)set $S_{i+1}$ contains $z_i$ element $x_i$. Note that if $y_i + j * z_i \ge b$ then we just set $y_i + j * z_i = b$.
6:   End For
7: End For
8: Find a shortest path from the vertex $(S_0, 0, \cdots, 0)$ to $(S_{|F_{ms}|}, b, \cdots, b)$ or $((S_{|F_{mc}|}, b, \cdots, b))$. If there does not exist such a path then we know we can not find a valid multicover, otherwise, just return the sum of the weights and the corresponding copies of the (multi)sets on the directed shorted path.
```

Theorem 4. *The EMCM algorithm can solve the MSMC or SMC with multiplicity constraints problem in $O((b+1)^n|F_{ms}|)$ $(O((b+1)^n|F_{mc}|))$ time using $O((b+1)^n|F_{ms}|)$ $(O((b+1)^n|F_{mc}|))$ space.*

Proof. First, we know that there are $O((b+1)^n|F_{ms}|)$ $(O((b+1)^n|F_{mc}|))$ vertices and $O((b+1)^n|F_{ms}|(d+1))$ $(O((b+1)^n|F_{mc}|(d+1)))$ directed edges. Second, since the constructed graph is a sparse directed acyclic graph, by using the Dijkstra's algorithm together with topological sorting, we can easily obtain the result. □

Remark: Observe that, for the SMC or the MSMC problem, each set or multiset in F_{mc} or F_{ms} can be used at most b times, the proposed *EMCM* algorithm can also be used as a constructive algorithm for these two problems.

5 Future Work

The time and space complexities of Algorithm 1 which is to calculate all F_{ms}^{Y} values are still very high. It should be possible to devise a more efficient algorithm which uses $O((b + c + 1)^n)$ time. This could be done by computing much fewer $d(X^Y, (N\backslash X)^Y, v(X))$ interim values. In addition, it should be very interesting to design an exact multiset multicover algorithm with $O^*((b + 1)^n)$ time—that is, the time is independent of the number of times that each element appears in a multiset.

We need to emphasize that counting the number of transformed set covers for multiset multicover, i.e., the $c_k(F'_{ms})$ value, is different from directly counting the number of multiset multicovers, i.e., the $c_k(F_{ms})$ value. Although there is now an exact algorithm for calculating $c_k(F_{ms})$ [8], the algorithm requires exponential space. So it is worthwhile to try to devise polynomial space efficient algorithms for computing $c_k(F_{ms})$.

It would be worthwhile also to apply our results to some practical scenarios, such as the minimum length wireless link scheduling problem [7] and the minimum cost cell planning problem [12].

Acknowledgments

This research is supported in part by a Hong Kong RGC–GRF grant(7136/07E) and the National Basic Research Program of China Grant 2007CB807900, 2007CB807901.

References

1. Björklund, A., Husfeldt, T., Koivisto, M.: Set partitioning via Inclusion-Exclusion. SIAM Journal on Computing, Special Issue for FOCS 2006 (to appear)
2. Björklund, A., Husfeldt, T., Kaski, P., Koivisto, M.: Fourier meets Möbius: fast subset convolution. In: Proc. 39th STOC, San Diego, CA, USA (2007)
3. Chuzhoy, J., Naor, J.: Covering Problems with Hard Capacities. SIAM J. Comput. 36(2), 498–515 (2006)
4. Kolliopoulos, S.G., Young, N.E.: Approximation algorithms for covering/packing integer programs. J. Comput. Syst. Sci. 71(4), 495–505 (2005)
5. Kolliopoulos, S.G.: Approximation Algorithms for Covering Integer Programs with Multiplicity Constraints. Discrete Applied Mathematics (129), 461–473 (2003)
6. Hua, Q.-S., Wang, Y., Yu, D., Lau, F.C.M.: Set multi-covering via inclusion-exclusion. Theoretical Computer Science V410(38-40), 3882–3892 (2009)
7. Hua, Q.-S., Lau, F.C.M.: Exact and approximate link scheduling algorithms under the physical interference model. In: Proc. 5th SIGACT-SIGOPS International Workshop on Foundation of Mobile computing (DIALM-POMC), Toronto, Canada (2008)
8. Hua, Q.-S., Yu, D., Lau, F.C.M., Wang, Y.: Exact Algorithms for Counting k-Set Multicover and Counting k-Multiset Multicover Problems (submitted, 2009)
9. Nederlof, A.J.: Inclusion-Exclusion for hard problems. Master Thesis. Utrecht University, The Netherlands (2008)

10. Rajagopalan, S., Vazirani, V.V.: Primal-dual RNC approximation algorithms for set cover and covering integer programs. SIAM Journal on Computing 28(2), 525–540 (1998)
11. Vazirani, V.V.: Approximation Algorithms. Springer, Berlin (2003)
12. Amzallag, D., Engelberg, R., Naor, J., Raz, D.: Capacitated Cell Planning of 4G Cellular Networks (Technical Report CS-2008-04). Computer Science Department, Technion, Haifa 32000, Israel (2008)