On Nonlinear Multi-Covering Problems

Reuven Cohen* Mira Gonen† Asaf Levin‡ Shmuel Onn§

March 12, 2015

Abstract

In this paper we define the exact k-coverage problem, and study it for the special cases of intervals and circular-arcs. Given a set system consisting of a ground set of n points with integer demands $\{d_0, \ldots, d_{n-1}\}$ and integer rewards, subsets of points, and an integer k, select up to k subsets such that the sum of rewards of the covered points is maximized, where point i is covered if exactly d_i subsets containing it are selected. Here we study this problem and some related optimization problems.

We prove that the exact k-coverage problem with unbounded demands is NP-hard even for intervals on the real line and unit rewards. Our NP-hardness proof uses instances where some of the natural parameters of the problem are unbounded (each of these parameters is linear in the number of points). We show that this property is essential, as if we restrict (at least) one of these parameters to be a constant, then the problem is polynomial time solvable. Our polynomial time algorithms are given for various generalizations of the problem (in the setting where one of the parameters is a constant).

1 Introduction

In this paper we consider the exact k-coverage problem. Given a set system consisting of a ground set $P := \{0, \ldots, n-1\}$ of points with integer demands $\{d_0, \ldots, d_{n-1}\}$ and integer rewards $\{R_0, \ldots, R_{n-1}\}$, subsets $S_1, \ldots, S_m$ of P, and an integer k, consider the following optimization problem which we call *maximum exact k-coverage problem*:

$$\max_{\mathcal{S} \subseteq \{S_1, \ldots, S_m\}, |\mathcal{S}| \leq k} \left\{ \sum_{i=0}^{n-1} R_i \cdot \delta(i, \mathcal{S}) \right\},$$

where $\delta(i, \mathcal{S}) = 1$ if $c_{i,\mathcal{S}} = d_i$, and $c_{i,\mathcal{S}} = |\{S \in \mathcal{S} | i \in S\}|$, and $\delta(i, \mathcal{S}) = 0$ otherwise. Namely, our goal is to select up to k subsets such that the sum of rewards of the covered points is maximized, where point i is covered if exactly d_i subsets containing it are selected. The special case of our problem where $d_i = 1$ for all i, and $k = m$, is known as the unique coverage problem, and the variant of the case where $d_i = 1$ for all i, each subset has a cost, and we would like to select a subcollection of subsets of total cost at most k is called the budgeted unique coverage problem.

This problem is NP-hard in general settings, and thus we study cases that turns out to be polynomial-time solvable. We start our discussion by showing the surprising fact that the problem is NP-hard even when

*Department of Mathematics, Bar-Ilan University, Ramat-Gan 5290002 Israel. reuven@math.biu.ac.il.

†Department of Computer Science and Mathematics, Ariel University, Ariel 40700 Israel. mirag@ariel.ac.il.

‡Faculty of Industrial Engineering and Management, The Technion, Haifa 32000 Israel. levinas@ie.technion.ac.il.

§Faculty of Industrial Engineering and Management, The Technion, Haifa 32000 Israel. onn@ie.technion.ac.il.

the set system is given by a set of intervals on the real line. This is fairly surprising given the fact that most similar problems are polynomially time solvable when restricted to intervals on the real line (e.g. the set cover problem can be solved in linear time in this simple settings [8]). In this NP-hardness proof we construct instances of the exact k-coverage problem in which there are several underlying parameters that are linear in the number of points. Thus, we study the special cases resulting by imposing the constraints (one constraint for each special case) that one of the parameters is bounded by a constant. We show that in all these cases, the problem becomes polynomial-time solvable, and in fact our positive results are defined to generalizations of the problem.

The *maximum k-coverage problem* is the following related problem:

$$\max_{\mathcal{S}\subseteq\{S_1,\ldots,S_m\},|\mathcal{S}|\leq k}\left\{\sum_{i=0}^{n-1} R_i\cdot\delta^{>}(i,\mathcal{S})\right\},$$

where $\delta^{>}(i,\mathcal{S})=1$ if $c_{i,\mathcal{S}}\geq d_i$, and $\delta^{>}(i,\mathcal{S})=0$ otherwise. Note that an instance of the maximum k-coverage problem can be easily transformed into an equivalent instance of the exact k-coverage problem by replacing each point i (in every subset of the set system that contains i) by a set of $m-d_i+1$ points (i,d) for $d=0,1,\ldots,m-d_i$ such that the demand of (i,d) is d_i+d and its reward is R_i. The well-studied maximum coverage problem is the special case of the maximum k-coverage problem where $d_i=1$ for all i. In a similar way we can model the case where the reward of covering point i by d subsets is a function of d using an instance of the exact k-coverage problem.

Maximal cover problems arise naturally in many optimization problems. The problem of maximum coverage of points on the real line by intervals can be used, for example, to find the optimal locations along a road for positioning fuel stations, cellular communication towers or different facilities, given some metric along the road such as traffic density, the locations of farms or the expected revenue. Our natural generalization of the problem, maximum k-coverage, can be also used in the same applications for fault-tolerant designs and redundancy engineering.

The problems of maximum coverage of points on the real line by intervals and of points on the circle by circular-arcs are known to be polynomial [2], unlike the maximum coverage problem of general set systems which is well known to be NP-hard [7, 17, 6]. The unique coverage problem is hard to approximate in general settings and has $O(\log B)$ approximation algorithm if every subset in the set system contains at most B elements [4], and improved results in geometric settings (see e.g. [5, 9, 10, 14, 13, 12]). Here, we extend the unique coverage problem into the exact k-coverage problem and study it in the (very restrictive) geometric setting of intervals on the real line and circular-arcs.

The general problem of maximum coverage is well known to be NP-hard [7, 17, 6]. According to Feige [6] for all $\epsilon>0$, the maximum coverage problem cannot be approximated in polynomial time within a ratio of $1-1/e+\epsilon$, unless P = NP. In addition the greedy algorithm (iteratively selecting the sets that cover the largest number of yet uncovered elements) approximates the maximum coverage problem within a ratio of at least $1-1/e$ [3, 1].

Organization: In Section 2 we prove NP-hardness of the problem for intervals on the real line by reducing the exact cover by 3-sets problem. As our reduction assumes unit rewards it implies that the problem is NP-hard even for this case. In section 3 we show a polynomial-time algorithm for the case of unconstraint demands and constant number of points or sets, and some generalized non-linear optimization functions. Moreover, for the case of intervals we present a much faster algorithm. In section 4 we focus on the case of a constant upper bound on the demands, and in Section 5 we study the case of a constant bound on the maximum length of an interval (or circular-arc). We conclude with Section 6.

Preliminaries. Before proceeding we give some remarks about computational complexity notation. The *binary length* of an integer z, denoted $\langle z \rangle$, is the number $\Theta(\log |z|)$ of bits in its binary encoding. The binary length of a rational number r, vector v, matrix M, or finite set S of such objects, is the sum of lengths of its numerical constituents, and in particular accounts for relevant dimension or cardinality. The *unary length* of an integer z is the number $\Theta(|z|)$ of bits in its unary encoding, and the unary length of an object as above is again the sum of lengths of its numerical constituents, and accounts again for relevant dimension or cardinality. In formal algorithmic statements we indicate for each input object if it affects the running time through its unary or binary length. For example, saying that an algorithm runs in time polynomial in n, c, $\langle A, u \rangle$ with n integer, c, u vectors, and A matrix, means that the time is polynomial in the unary length of n, c and binary length $\langle A, u \rangle$ of A, u. If an objective function g is presented by an oracle then the running time includes also the number of oracle queries and time needed to manipulate oracle answers. The input may include, when relevant, the binary or unary length of a prior or posterior upper bound $\hat{g}$ on the absolute value $|g(x)|$ over the set of feasible points.

In our dynamic programming algorithms we compute the objective function of an optimal solution. It is well-known that it is possible to compute an optimal solution in the same time complexity using backtracking.

2 NP-hardness

We show that the decision version of the maximum exact k-coverage problem is NP-hard in case the ground set is a set of points on the real line and the sets are intervals, even in the case of unit rewards. The question if the maximum k-coverage problem for intervals or even for circular-arcs is NP-hard remains open.

Problem 1 Maximum exact k-coverage problem for line intervals - decision version: *given a set $P := \{0, \ldots, n-1\}$ of points on the real line with demands $\{d_0, \ldots d_{n-1}\}$, a set of subsets $M = \{S_1, \ldots, S_m\}$ of P, where each S_i is an interval on the real line, and integers ℓ, k, can at most k intervals be selected from M to cover at least ℓ points, where point i is covered if exactly d_i intervals containing it are selected.*

We present a reduction from the X3C (Exact Cover by 3-Sets) problem which is defined as follows: given a finite set $X = \{x_1, \ldots, x_n\}$ and a collection $E = \{e_1, \ldots, e_m\}$, $m \geq n/3$, of 3-element subsets of X, is there a subcollection $E' \subseteq E$ such that every element of X occurs in exactly one member of E', i.e., the members of E' are pairwise disjoint, and $\bigcup_{e \in E'} e = X$?

Theorem 1 *The Maximum exact k-coverage problem for intervals on the real line is NP-hard.*

Proof: We show a reduction from the X3C problem: given an instance $X = \{x_1, \ldots, x_n\}$, $E = \{e_1, \ldots, e_m\}$, $m \geq n/3$, of the X3C problem we define the following instance to the Maximum exact k-coverage problem. For each element x_i, $1 \leq i \leq n$ define a demand point i, with $d_i = i$. In addition, for each subset e_j, $1 \leq j \leq m$, define $n+1$ demand points in positions $n + (j-1) \cdot (n+1) + i$, for $1 \leq i \leq n+1$, where the ith point has demand $3 \cdot (i-1)$. Moreover, for each such subset $e_j = \{x_{j_1}, x_{j_2}, x_{j_3}\}$ we define 3 intervals $[j_1, n + j \cdot (n+1)], [j_2, n + j \cdot (n+1)], [j_3, n + j \cdot (n+1)]$. We prove that there is a subset of $k = n$ intervals that covers at least $n + m$ points if and only if there is a subcollection $E' \subseteq E$ such that every element of X occurs in exactly one member of E'. Assume first that there is a subcollection $E' \subseteq E$ such that every element of X occurs in exactly one member of E'. Then we pick n intervals as follows. For each subset $e_j = \{x_{j_1}, x_{j_2}, x_{j_3}\}$ in E' we take the intervals $[j_1, n + j \cdot (n+1)], [j_2, n + j \cdot (n+1)], [j_3, n + j \cdot (n+1)]$. Then, for each demand point i in $\{1, \ldots, n\}$ there is exactly one interval that starts at i, so it is covered exactly d_i times. (This is true since all elements in X occur in E' exactly once, and by the intervals definition

the interval that starts at point i also includes points $i+1,\ldots,n$). Moreover, for each subset e_j in E' the number of intervals that include the points $n+(j-1)\cdot(n+1)+i$ ($1 \le i \le n+1$) is a multiplicity of 3 so one of these points (one for each e_j) is covered. Therefore $n+m$ points are covered. For the opposite direction assume that there is a solution to the maximum exact k-coverage problem with a value of $n+m$ (namely, $n+m$ points are covered). Then every subset e_j includes exactly one covered point in $\{n+(j-1)\cdot(n+1)+1,\ldots,n+(j-1)\cdot(n+1)+n+1\}$. This is true since at most one such point can be covered for a given j, and if there exists j for which no such point is covered it is impossible to cover $n+m$ points. Now define the following solution to the X3C problem: pick all subsets e_j for which there are three intervals ending at point $n+j\cdot(n+1)$ in the solution to the maximum exact k-coverage problem. Thus for each e_j, either the three intervals ending at $n+j(n+1)$ are selected, or none of them are selected. Since there are at most n intervals in the solution to the maximum exact k-coverage problem, we picked at most $n/3$ subsets. In addition, for all $i=1,\ldots,n$ an interval starting at i is necessarily picked since point i is included in i intervals and point $i-1$ is included in $i-1$ intervals. Therefore for all elements x_i, $1 \le i \le n$ a subset containing x_i is picked. ■

Remark 2.1 *We observe that the reduction used in the proof of Theorem 1 uses linear values for the following three parameters: The number of distinct subsets in the set system, the maximum demand (i.e.,* $\max_i d_i$*), and the maximum cardinality of a subset in the set system (i.e.,* $\max_i |S_i|$*). In the rest of the paper, we show that this is essential in the sense that when we restrict one of the parameters to a constant then the problem (and some generalizations of it) is polynomial-time solvable.*

3 Polynomial-time algorithm for the case of constant number of points or sets and unconstraint demands

We next discuss the following case: assume that there is a constant number of points, or that there is a constant number of different sets, where set S_j has u_j copies. We assume that u_j is represented in binary form for all j. We have no constraints on d_i. We first consider a generalization of the maximum exact k-coverage problem to general set systems and a more general optimization functions, and give a polynomial algorithms for the cases of oracle-presented (quasi)-convex functions and step and staircase functions. Then we introduce an improved algorithm for the specific case of the problem for intervals.

Consider a set system consisting of a ground set which, for convenience, we now index as $P := [n] := \{1,\ldots,n\}$, and subsets $S_1,\ldots,S_m$ of P. We identify the system with its $n \times m$ incidence matrix A defined by $A_{i,j}=1$ if $i \in S_j$ and $A_{i,j}=0$ otherwise. We denote rows and columns of A by A_i and A^j respectively. So $S_j = \mathrm{supp}(A^j)$ is the support of column j. Let $l,u \in \mathbb{Z}^m$ with $\mathbf{0} \le l \le u$, and let $f : \mathbb{Z}^n \to \mathbb{Z}$ be a function. We consider the following rather general nonlinear optimization problem,

$$\max\{f(Ax) \,:\, l \le x \le u\,,\; x \in \mathbb{Z}^m\}\;. \tag{1}$$

This problem is interpreted as a covering problem as follows. Each vector x represents a *cover* of points by sets. The number x_j of copies of set S_j which can be used should obey the bounds $l_j \le x_j \le u_j$. The *weight of cover x on point i* is the number $A_i x = \sum\{x_j \,:\, i \in S_j\}$ of sets used by x containing point i. The objective is the function f evaluated at the vector $y := Ax$ of weights of all points under cover x.

Off hand the problem may seem simple, since the feasible points lie in a cube, but even with $l := \mathbf{0}$ and $u := \mathbf{1}$ the all-ones vector it captures hard combinatorial optimization problems. Moreover, with $n := m$ and $A := I_m$ the identity matrix (that is, $S_i = \{i\}$ for all i), the objective function $f(Ax) = f(x)$ can be arbitrary.

Example 3.1 Incomputability. If some variables were unbounded then the problem might become even incomputable. Let $n := m := 58$, $A := I_{58}$, $l := \mathbf{0} \in \mathbb{Z}^{58}$, and $u \in (\mathbb{Z} \cup \{\infty\})^{58}$ with $u_j := \infty$ for all j. Then the solution of Hilbert's 10-th problem (see [11]) implies that there exists no algorithm that, given integer polynomial f of degree 8 in 58 variables, decides if the optimal objective value of problem (1) is 0.

But even with finite bounds and simpler functions, the problem may be hard.

Example 3.2 Subset sum. Consider the NP-complete problem of deciding, given positive integers $a_0, a_1, \ldots, a_m$, if there is a subset $J \subseteq [m]$ with $\sum_{j \in J} a_j = a_0$. Let $n := 1 + \max_{j=1}^{m} \lfloor \log_2 a_j \rfloor$ and let A be the $n \times m$ matrix whose j-th column is the binary presentation of a_j, that is, $a_j = \sum_{i=1}^{n} 2^{i-1} A_{i,j}$. Define a linear function $f : Z^n \to \mathbb{Z}$ by $f(y) = \sum_{i=1}^{n} 2^{i-1} y_i$, and define $l, u \in \mathbb{Z}^m$ by $l := \mathbf{0}$ and $u := \mathbf{1}$. Then the optimal value of problem (1) is a_0 if and only if there is a feasible subset sum.

Nonetheless, we show here that under various useful and reasonable assumptions on the system A or function f, problem (1) can be efficiently solved or approximated.

3.1 General set systems

3.1.1 Oracle-presented (quasi)-convex functions

Here we consider problem (1) when the function f is presented by various oracles. We first consider presentation by a *comparison oracle* that, queried on $y, z \in \mathbb{Z}^n$, asserts whether or not $f(y) \leq f(z)$. As the next example demonstrates, solving the problem under this rather broad presentation, which reveals little information about the function, may require exponential time even for the trivial set system $A = I_m$.

Example 3.3 Exponential oracle time. For any $v \in \{0,1\}^n$ let $f_v : \mathbb{Z}^n \to \{0,1\}$ be the function defined by $f(y) = 1$ if $y = v$ and $f(y) = 0$ otherwise. Consider problem (1) with $n := m$, $A := I_m$ the $m \times m$ identity matrix, $l, u \in \mathbb{Z}^m$ with $l := \mathbf{0}$ and $u := \mathbf{1}$, and function f presented by comparison oracle. Suppose an algorithm trying to solve the problem makes less than 2^{m-1} oracle queries. Then some point $v \in \{0,1\}^m$ does not appear in any query "is $f(y) \leq f(z)$?" to the oracle. So if the oracle presents either f_v or the identically zero function, it replies "yes" to all queries, and hence the algorithm cannot tell if the optimal objective value is 0 or 1.

Nonetheless, we next show that if either the number n of points or the number m of sets is fixed, then we can solve the problem in polynomial time for a broad class of set systems and functions presented by comparison oracles. We use the theory of [16], see also [15, Chapter 2]. Recall that a function $f : \mathbb{R}^n \to \mathbb{R}$ is *convex* if for all $x, y \in \mathbb{R}^n$ and $0 \leq \lambda \leq 1$ we have $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$, and more broadly, f is *quasi-convex* if for all such x, y, λ, we have $f(\lambda x + (1-\lambda)y) \leq \max\{f(x), f(y)\}$.

Theorem 3.4 *For any fixed n, problem (1) with any set system A and any (quasi)-convex f presented by a comparison oracle can be solved in time polynomial in $\langle l, u \rangle$.*

Proof. Fix n. We describe two ways of solution. The first way is much more efficient and applies the theory of [16]. Let $S := \{x \in \mathbb{Z}^m : l \leq x \leq u\}$ be the set of feasible points. By [15, Theorem 2.16] it suffices to show that we can efficiently optimize linear functions over S and compute a set of all edge-directions of the polytope $\mathrm{conv}(S)$. An optimizer of any given linear function wx over S is simply computed by

$$x_i := \begin{cases} u_i, & \text{if } w_i \geq 0; \\ l_i, & \text{otherwise} \end{cases}, \; i = 1, \ldots, m \, .$$

A set of all edge-directions of $\mathrm{conv}(S)$ consists of the unit vectors $\mathbf{1}_1, \ldots, \mathbf{1}_m$. The resulting algorithm has time complexity $m^{O(n)}$ which is singly exponential in n.

For the second way, let $\bar{m} \leq 2^n$ be the number of distinct columns of A and let $\bar{A}$ be the $n \times \bar{m}$ compressed matrix consisting of one copy of each such column. For $k = 1, \ldots, \bar{m}$ let $J_k \subseteq [m]$ be the set of indices of columns of A equal to $\bar{A}^k$. Let $\bar{l}, \bar{u} \in \mathbb{Z}^{\bar{m}}$ be the compressed bounds with $\bar{l}_k := \sum_{j \in J_k} l_j$ and $\bar{u}_k := \sum_{j \in J_k} u_j$ for all k. The objective function of the following compressed problem remains (quasi)-convex,

$$\max \left\{ f(\bar{A}\bar{x}) \ : \ \bar{l} \leq \bar{x} \leq \bar{u} \, , \ \bar{x} \in \mathbb{Z}^{\bar{m}} \right\} \, , \tag{2}$$

and hence an optimal solution $\bar{x}$ is obtained at one of its $2^{\bar{m}}$ vertices. This solution can be expanded to a feasible solution $x \in \mathbb{Z}^m$ of the original problem as follows. Consider any $1 \leq k \leq \bar{m}$. We need to find x_j, $j \in J_k$, such that $l_j \leq x_j \leq u_j$ and $\sum_{j \in J_k} x_j = \bar{x}_k$. For simplicity assume first $l_j = 0$ for all $j \in J_k$. Start with $x_j := 0$ for all $j \in J_k$; while $\sum_{j \in J_k} x_j < \bar{x}_k$, pick any $x_j = 0$, $j \in J_k$, and set its value to

$$x_j \ := \ \min\{u_j, \bar{x}_k - \sum \{x_i \ : \ i \in J_k \setminus \{j\}\} \ .$$

In the general case, define $\bar{z}_k := \bar{x}_k - \sum_{j \in J_k} l_j$; use the above procedure to obtain z_j, $j \in J_k$ such that $0 \leq z_j \leq u_j - l_j$ and $\sum_{j \in J_k} z_j = \bar{z}_k$; and set $x_j := z_j + l_j$ for all $j \in J_k$. The point x obtained this way is an optimal solution to the original problem. However, here the time complexity is dominated by the number of vertices of the compressed problem which is $2^{\bar{m}} = O(2^{2^n})$ and hence doubly exponential in n. □

Theorem 3.5 *For any fixed m, problem (1) with any set system A and any (quasi)-convex f presented by a comparison oracle can be solved in time polynomial in $\langle l, u \rangle$.*

Proof. Fix m. Let $\bar{n} \leq 2^m$ be the number of distinct rows of A and let $\bar{A}$ be the $\bar{n} \times m$ compressed matrix consisting of one copy of each such row. For $k = 1, \ldots, \bar{n}$ let $I_k \subseteq [n]$ be the set of indices of rows of A which are equal to $\bar{A}_k$. Define a function $e : \mathbb{Z}^{\bar{n}} \to \mathbb{Z}^n \ : \ z \mapsto y$ by $y_i := z_k$ for all $k = 1, \ldots, \bar{n}$ and $i \in I_k$ and a function $g : \mathbb{Z}^{\bar{n}} \to \mathbb{Z}$ by $g(z) := f(e(z))$. A comparison oracle for g is easily realizable from that of f. Moreover, since e is linear, g maintains the (quasi)-convexity of f, that is,

$$\begin{aligned} g\left(\lambda z^1 + (1-\lambda) z^2\right) &= f\left(e\left(\lambda z^1 + (1-\lambda) z^2\right)\right) = f\left(\lambda e(z^1) + (1-\lambda) e(z^2)\right) \\ &\leq \lambda f\left(e(z^1)\right) + (1-\lambda) f\left(e(z^2)\right) = \lambda g(z^1) + (1-\lambda) g(z^2) \end{aligned}$$

holds in the convex case, and a similar calculation holds in the quasi-convex case.

Therefore we can now apply the algorithms of Theorem 3.4 to the problem

$$\max \left\{ g(\bar{A}x) \ : \ l \leq x \leq u \, , \ x \in \mathbb{Z}^m \right\} \ . \tag{3}$$

Any optimal solution of this problem is also optimal for the original problem (1). □

3.1.2 Step and staircase functions

A function $f : \mathbb{Z}^n \to \mathbb{Z}$ is *separable* if $f(y) = \sum_{i=1}^n f_i(y_i)$ for all $y \in \mathbb{Z}^n$, where $f_i : \mathbb{Z} \to \mathbb{Z}$ is univariate for each i. Substituting $y := Ax$, the objective function of problem (1) then becomes $f(Ax) =$

$\sum_{i=1}^{n} f_i(A_i x)$. A *separable-step function* is a separable f with each f_i a *step function* defined for some integers b_i, c_i, d_i as follows,

$$f_i(y_i) \;=\; \begin{cases} c_i, & \text{if } y_i \geq b_i; \\ d_i, & \text{otherwise.} \end{cases}$$

Thus, here the objective value $f(Ax)$ of cover x is the sum of values c_i of all points i whose weight under the cover x is at least the threshold b_i, plus the sum of values d_i of all points i whose weight under the cover x is less than the threshold b_i. Note that an additional constraint on the total number of sets which could be used, of the form $\sum_{j=1}^{m} x_j \leq w$, can be easily incorporated into the problem by adding a set $S_0 := P = \{1, \ldots, n\}$, $l_0 := 0$, $u_0 := 1$, $b_0 := w + 1$, $c_0 := -(1 + \sum_{i=1}^{n} \max\{|c_i|, |d_i|\})$, $d_0 := 0$.

If both m, n are variable, the problem remains hard with separable-step functions.

Example 3.6 Vertex cover. Consider the NP-complete problem of deciding, given $w \in \mathbb{Z}_+$ and graph $G = (V, E)$ with $V = \{v_1, \ldots, v_m\}$ and $E = \{e_1, \ldots, e_n\}$, if there is a subset $U \subseteq V$ such that $|U| \leq w$ and $U \cap e_i \neq \emptyset$ for all i. Let A be the $n \times m$ incidence matrix with $A_{i,j} = 1$ if $e_i \in S_j := \delta(v_j)$ and $A_{i,j} = 0$ otherwise. Define $l, u \in \mathbb{Z}^m$ by $l := \mathbf{0}$ and $u := \mathbf{1}$. For $i = 1, \ldots, n$ define identical step functions by

$$f_i(y_i) \;=\; \begin{cases} 1, & \text{if } y_i \geq 1; \\ 0, & \text{otherwise.} \end{cases}$$

The constraint $|U| \leq w$ becomes $\sum_{j=1}^{m} x_j \leq w$ and can be incorporated into the problem as explained above. The optimal objective function value of the augmented system is equal to n if and only if there is a vertex cover of cardinality at most w.

More generally, a *staircase function* is a univariate function of the following form,

$$h(y) \;=\; \begin{cases} c_p, & \text{if } y \in B_p := [b_p, \infty). \\ c_{p-1}, & \text{if } y \in B_{p-1} := [b_{p-1}, b_p); \\ \vdots & \vdots \\ c_1, & \text{if } y \in B_1 := [b_1, b_2); \\ c_0, & \text{if } y \in B_0 := (-\infty, b_1); \end{cases}$$

A *separable-staircase function* is a separable function $f = \sum_{i=1}^{n} f_i$ where each f_i is a staircase function. Duplicating some of the intervals in some of the f_i if necessary, we may and do assume that all f_i have the same number $p + 1$ of intervals. We denote by $c_{i,j}$, $b_{i,j}$, and $B_{i,j}$ for $i = 1, \ldots, n$ and $j = 0, 1, \ldots, p$, all constituents in the description of such f, and by $\langle f \rangle := \langle b, c \rangle$ the binary encoding length of f.

The following holds for separable-staircase (and separable-step) functions.

Theorem 3.7 *For every fixed n, problem (1) with any set system A and any separable-staircase function $f = \sum f_i$ can be solved in time polynomial in $\langle l, u, f \rangle$.*

Proof. Let $\bar{m} \leq 2^n$ be the number of distinct columns of A and let $\bar{A}$ be the $n \times \bar{m}$ compressed matrix consisting of one copy of each such column. For $k = 1, \ldots, \bar{m}$ let $J_k \subseteq [m]$ be the set of indices of columns of A equal to $\bar{A}^k$. Let $\bar{l}, \bar{u} \in \mathbb{Z}^{\bar{m}}$ be the compressed lower and upper bounds $\bar{l}_k := \sum_{j \in J_k} l_j$ and $\bar{u}_k := \sum_{j \in J_k} u_j$ for all k.

Now, for each of the $(p + 1)^n$ maps $t : [n] \to \{0, 1, \ldots, p\}$ do the following. Test in polynomial time feasibility of the following integer program in fixed dimension,

$$\left\{ \bar{x} \in \mathbb{Z}^{\bar{m}} \;:\; \bar{A}_i \bar{x} \in B_{i,t(i)} \text{ for } i = 1, \ldots, n \,,\; \bar{l} \leq \bar{x} \leq \bar{u} \right\} \;; \tag{4}$$

since we are working over the integers, each interval $B_{i,r}$ is in fact a closed interval $B_{i,r} = [b_{i,r}, b_{i,r+1} - 1]$ and so the constraints can be expressed by weak inequalities.

If there is a feasible point $\bar{x} \in \mathbb{Z}^{\bar{m}}$ then it can be expanded to a feasible point $x \in \mathbb{Z}^m$ of the original problem: for each $1 \leq k \leq \bar{m}$ find x_j for all $j \in J_k$ such that $l_j \leq x_j \leq u_j$ and $\sum_{j \in J_k} x_j = \bar{x}_k$, as described in detail in the second part of the proof of Theorem 3.4. The objective function value of this point x is $f(x) = \sum_{i=1}^{n} c_{i,t(i)}$.

The point x maximizing this value over all feasible points derived from all $t \in \{0, 1, \ldots, p\}^n$ is the optimal solution. The polynomial time complexity is clear. □

We next show that for fixed m such problems are also polynomial time solvable.

Theorem 3.8 *For every fixed m, problem (1) with any set system A and any separable-staircase function $f = \sum f_i$ can be solved in time polynomial in $\langle l, u, f \rangle$.*

Fix m. Let $\bar{n} \leq 2^m$ be the number of distinct rows of A and let $\bar{A}$ be the $\bar{n} \times m$ compressed matrix consisting of one copy of each such row. For $k = 1, \ldots, \bar{n}$ let $I_k \subseteq [n]$ be the set of indices of rows of A which are equal to $\bar{A}_k$. Define a function $e : \mathbb{Z}^{\bar{n}} \to \mathbb{Z}^n : z \mapsto y$ by $y_i := z_k$ for all $k = 1, \ldots, \bar{n}$ and $i \in I_k$ and a function $g : \mathbb{Z}^{\bar{n}} \to \mathbb{Z}$ by $g(z) := f(e(z))$. It can be verified that g is also separable-staircase, with representation which can be derived from that of f in polynomial time.

Therefore we can now apply the algorithm of Theorem 3.7 to the problem

$$\max \left\{ g(\bar{A}x) \ : \ l \leq x \leq u \, , \ x \in \mathbb{Z}^m \right\} \ . \tag{5}$$

Any optimal solution of this problem is also optimal for the original problem (1). □

3.2 Intervals on the real line

We next show that in the specific case of the maximum exact k-coverage problem for intervals on the real line when $n = c_p$ is constant, there is a constant number of intervals, c_I, and each interval I_j has u_j copies we can improve the algorithms of Theorems 3.7 and 3.8. (Notice that this problem for constant number of points and intervals is a special case of problem (1) in which the points are integers lying on the real line and each set in our system is an interval).

As before, w.l.o.g. assume that all intervals have close left endpoint, and open right endpoint, i.e. they are of the form $[a, b)$.

The following algorithm solves the problem in time polynomial in $< \sum_{j=1}^{m} u_j >$:

Algorithm 1 (Max exact k-coverage of intervals, $\mathcal{S}(k, c_I, u_1, \ldots, u_{c_I}, P)$)

1. *For every subset $P_\ell \subseteq P$ solve the following LP:*

   (a) *For every interval I_j let X_j be the number of times interval I_j is picked.*

$$\begin{array}{ll} \textit{Minimize} & \sum_j X_j \\ \textit{subject to} & \sum_{j : i \in I_j} X_j = d_i, \forall i \in P_\ell \\ & 0 \leq X_j \leq u_j, \forall j, \ 1 \leq j \leq c_I \end{array}$$

2. *Let $h(P_\ell)$ be the solution of the LP For P_ℓ.*

3. *Return* $\max_{P_\ell \subseteq P | h(P_\ell) \leq k} \sum_{i \in P_\ell} R_i$.

Theorem 2 *Algorithm 1 returns an optimal cover with polynomial time complexity.*

Proof: For a given set of points P_ℓ we define an $|P_\ell| \times c_I$ constraint matrix A^ℓ as follows: $A^\ell_{i,j} = 1$ if interval I_j covers point i, and $A^\ell_{i,j} = 0$ otherwise. W.l.o.g assume that the constraints are sorted according to the order of the points. Then each matrix A^ℓ has a consecutive set of 1's in every column (since obviously every interval covers consecutive set of points). Therefore each A^ℓ is totally unimodular, so each LP in the above algorithm gives an integral solution (since we can assume that we obtain a basic optimal solution). Since $|P| = c_p$, for each subset of points P_ℓ the size of the matrix A^ℓ is upper bounded by $c_p \times c_I$, and each variable X_j is upper bounded by u_j, the solution of all the LP's in the algorithm is polynomial. Since the algorithm goes over all subsets of points and selects the subset that can be optimally covered by picking at most k intervals, it is obviously optimal. ■

Notice that modifying the constraint $\sum_{j:i \in I_j} X_j = d_i$ to $\sum_{j:i \in I_j} X_j \geq d_i \ \forall i \in P_\ell$ results in a polynomial-time solution for the maximum k-coverage problem for intervals.

4 Polynomial-time algorithm for the case of constant maximum demand

We now discuss problems for intervals and circular-arcs in the case that $d_i < C$ for all i, where C is a constant. We show that in this case the problem can be solved in polynomial time. In the case of circular arcs assume that the set of input points $P = \{0, 1, \ldots, n-1\}$ is clockwise ordered. W.l.o.g. assume that all intervals and all circular-arcs have close left endpoint and open right endpoint, i.e., they are of the form $[a, b)$.

We consider the following generalization of the problem for a constant value of C. We are given a set system specified by a set of intervals on the real line or a set of circular-arcs. For each point i in P we are given a function $g(i, \ell)$ specifying the reward we obtain when we cover point i by exactly ℓ intervals (or circular-arcs). The value of ℓ is between 0 and C (and we use the convention that $g(i, \ell) = 0$ if $\ell > C$). In addition, we have a constant number of cost constraints, instead of unit costs in the original problem (where each set has cost 1 if it is selected and 0 if not). Given a constant p, let A be a matrix with p rows and m columns where each component of A is a non-negative integer. Let b be a vector of p components each of which is polynomially bounded (meaning at most a polynomial in $n, p, \max_i \log d_i$). Then, consider the following problem of maximizing $\sum_{i=0}^{n-1} g(i, x_i)$ subject to $\exists y \in \{0,1\}^m$ s.t $Ay \leq b$ and for all i we have $|\{j : i \in S_j, y_j = 1\}| = x_i$. In these notation, $y_j = 1$ if we select the subset S_j, and x_i is the (exact) number of selected subsets that cover i. The p budget constraints are $Ay \leq b$.

For solving the maximum k-coverage problem with several cost constraints for intervals in polynomial time we apply a dynamic programming algorithm The dynamic programming procedure computes the value of $F(i, S, t_1, \ldots, t_p)$ where i is the current point, S is a set of intervals which were already selected and cover point $i+1$ and do not start at point $i+1$, and $\sum_{e:I_e \in S} A_{q,j} \cdot y_j \leq b_q - t_q$ for all $1 \leq q \leq p$. t_q is the budget left from the qth constraint. $F(i, S, t_1, \ldots, t_p)$ is the optimal profit received by covering the given input points in the interval $[i, n]$, where the set of intervals S has already been selected, using additional budget $t_1, \ldots, t_p$. We would like to compute $\max_{b_1, \ldots, b_p \geq 0} F(-1, \emptyset, b_1, \ldots, b_p)$.

Observe that without loss of generality we can restrict ourselves to solutions in which the number of selected intervals ending at each point j is at most C as if there are more selected intervals (say $C + m_j$) then we can delete the shorter m_j intervals (among the ones ending at j) without decreasing the objective

function value. Thus, we consider all subsets of cardinality at most C of the set of intervals anding at j, and we denote by $S(j,\ell)$ the ℓ-th such subset and we denote by L_j the set of indices of ℓ corresponding to j. Similarly, in an optimal solution without loss of generality every point j is covered by at most $2C$ intervals (otherwise, we can delete from the solution any interval containing j beside the C intervals with smallest starting points, and C intervals with largest ending point without decreasing the objective function value). We let $S(j)$ be the set of all intervals in the set system ending at j and S^j the set of intervals in the set system starting at j.

Recall that $n-1$ is an upper bound on the rightmost right endpoint of the intervals. The algorithm is defined by a recursive formula for $F(i,S,t_1,\ldots,t_p)$.

1. If $i=n$, then if $0 \leq t_1 \leq b_1$,...,$0 \leq t_p \leq b_p$ and $S=\emptyset$ let $F(n,S,t_1,\ldots,t_p)=0$. Otherwise (for $i=n$) let $F(n,S,t_1,\ldots,t_p)=-\infty$. Moreover, for $i<n$ we have $F(i,S,t_1,\ldots,t_p)=\infty$ if there exists k such that $t_k<0$.

2. Otherwise, (i.e., $i<n$) we have $F(i,S,t_1,\ldots,t_p)=\max_{\ell \in L_i}\{g(i,|S \cup S(i,\ell))|)+F(i+1,S \setminus S(i+1),t_1-\sum_{e:I_e \in S(i,\ell)} A_{1,e} \cdot y_e,\ldots,t_p-\sum_{e:I_e \in S(i,\ell)} A_{p,e} \cdot y_e)\}$.

Remark 4.1 *The algorithm computes an optimal solution in* $O(n \cdot m^{3C} \cdot \left(\prod_{q=1}^{p} b_q\right) \cdot C)$ *time.*

Proof: The time complexity of the algorithm follows since i has at most $n+1$ values, S has at most m^{2C} values, t_q has at most b_q+1 values for all $1 \leq q \leq p$, and L_i has at most m^C indices. ■

For solving this generalized problem for circular-arcs we first guess a point i that is covered by at most C circular-arcs, and among the set of circular-arcs containing i we guess the subset selected by the optimal solution. The number of possibilities for this guessing step is nm^C, and thus we use exhaustive enumeration to implement this guessing step. For a specific value of the guess, we obtain an instance of the problem on the real line (as all other subsets do not contain i, and thus we can cut the circle in point i to obtain a set of points on the real line), and this instance on the real line uses an updated reward functions (for points that are covered by some of the intervals used to cover i) and updated budget constraints.

We now consider a variant of the above problem for which the reward function for each point i is polynomially bounded, and one of the components of b is not necessarily polynomially bounded. W.l.o.g assume that b_1 is not polynomially bounded. For solving this problem in polynomial-time we perform a binary search on the maximum value R for which the optimal value of the following problem is at most b_1: Minimize $A_1 \cdot y$, subject to $\sum_{i=0}^{n-1} g(i,x_i) \geq R$, $A_2 \cdot y \leq b_2,\ldots,A_p \cdot y \leq b_p$, for all i, $|\{j : i \in S_j, y_j = 1\}| = x_i$, and $y \in \{0,1\}^m$.

For solving this problem we apply the trivial modification to the above algorithm (this is reflected in a modified starting conditions that allow for covering constraints instead of packing constraints).

Next, we show that if there are more than one cost constraint in which the coefficients are not necessarily polynomially bounded then the problem becomes NP-hard even for intervals. For proving that we reduce the partition problem to this problem. Consider an input to a variant of the partition problem in which we would like to find a subset of cardinality $n/2$ of the input numbers $a_1,a_2,\ldots,a_n$ (all of them are positive integers) where the sum of the numbers in the subset is exactly A (where $\sum_{i=1}^{n} a_i = 2A$). Define a set of n demand points along the line, where each point has a reward of 1 and a unit demand. There are n intervals, where the i-th interval covers only point i. We have two cost constraints with coefficients $c_1,\ldots,c_n$ and $f_1,\ldots f_n$. The total c cost should be at most C, and the total f cost should be at most F, where the c and f coefficients are defined as follows. $c_i = a_i$ and the total budget of the c-cost is $C = A$. $f_i = A - a_i$, and the total budget for the f-cost is $F = ((n/2)-1)A$. The problem now is to find out if there exists

a solution of total reward at least $n/2$. Assume that there is a solution to the partition problem. Let $B \subseteq \{1, 2, \ldots, n\}$ a set of cardinality $n/2$ such that $\sum_{i \in B} a_i = A$. Let $I = \{I_i | i \in B\}$. Then $\sum_{i \in B} c_i = A$, $\sum_{i \in B} f_i = \sum_{i \in B} (A - a_i) = (n/2 - 1)A$. Therefore there exists a solution to the maximum k-coverage problem of total reward at least $n/2$. Now assume that there is a solution to the maximum k-coverage problem of total reward at least $n/2$. Let I be the set of the selected intervals. Then, by the bound on the total f-cost no more than $n/2$ intervals can be selected, so $|I| = n/2$. Let $B = \{i \in \{1, \ldots, n\} | I_i \in I\}$. Then $\sum_{i \in B} a_i = \sum_{i \in B} c_i \leq A$, and $(n/2)A - \sum_{i \in B} a_i = \sum_{i \in B} (A - a_i) \leq \sum_{i \in B} f_i \leq (\frac{n}{2} - 1)A$, and therefore $\sum_{i \in B} a_i \geq A$. Thus, $\sum_{i \in B} a_i = A$ and so there is a solution to the partition problem.

5 Polynomial-time algorithm for the case of constant maximum length interval

In this section we show how to modify our dynamic programming algorithm of the previous section to the case where the maximum demand is unbounded however the maximum length of an interval (or circular-arc) is bounded by a constant T. Here we assume that covering a point i by (exactly) ℓ intervals (or circular-arcs) results a non-negative reward of $g(i, \ell)$ (and this allows us to consider both the maximum exact k-coverage and the maximum k-coverage problems).

To do so we observe that every solution is defined by specifying for each point i and each length $j = 0, 1, 2, \ldots, T$ the number of intervals (or circular-arcs) of length j starting at i that are selected to the solution. Note that an interval that covers i starts no earlier than $i - T$, and thus when we process point i we will recall (in the definition of the states corresponding to point i) the number of intervals that we already selected and the set of intervals starting in point at least $i - T$ and at most $i - 1$. Therefore, there are at most $O(m^{T^2} k)$ states corresponding to point i.

The dynamic programming for the case of intervals is obtained using these observation by a slight modifications to the equations of the previous section, and we get a time complexity of $O(nm^{T^2+1}k)$, and for the case of circular-arcs we pick point 0 and we guess the set of circular-arcs covering point 0 in a fixed optimal solution (there are $O(m^{T^2})$ possibilities for this guess, and then we apply the algorithm for intervals in the resulting instance. Thus, we conclude the following.

Theorem 5.1 *There is a polynomial time algorithm for the maximum exact k-coverage and the maximum k-coverage problems for the cases where the set-system is defined by intervals or circular-arcs and the maximum cardinality of a subset in the set system is bounded by a constant.*

Note that if the maximum cardinality of a subset in the set system is bounded by 1, then the problems are trivially solved in linear time. However, for a set system where the maximum cardinality is bounded by 3 even if the demand of each element is 1, the problem is the Exact 3-Cover problem and thus NP-hard.

6 Conclusions

We defined here several variants of the maximum exact k-covering problem. We first defined the problem at hand, and showed that it is NP-hard for intervals. Then, we presented several polynomial-time algorithms for some cases of the problem and variants of it.

References

[1] A. A. Ageev and M. I. Sviridenko. Approximation algorithms for maximum coverage and max cut with given sizes of parts. In *Proc. IPCO*, pages 17 – 30, 1999.

[2] R. Cohen and M. Gonen. On interval and circular-arc covering problems.

[3] G. Cornuejols, G. L. Nemhauser, and L. A. Wolsey. Worst-case and probabilistic analysis of algorithms for a location problem. *Operations Research*, 28:847–858, 1980.

[4] E. D. Demaine, U. Feige, M. Hajiaghayi, and M. R. Salavatipour. Combination can be hard: Approximability of the unique coverage problem. *SIAM Journal on Computing*, 38(4):1464–1483, 2008.

[5] T. Erlebach and E. J. van Leeuwen. Approximating geometric coverage problems. In *Proc. of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA'08)*, pages 1267–1276, 2008.

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

[7] M. R. Garey and D. Johnson. *computers and intractability: A guide to the Theory of NP-Completeness*. Freeman, 1978.

[8] D. S. Hochbaum and A. Levin. Optimizing over consecutive 1's and circular 1's constraints. *SIAM J. on Optimization*, 17(2):311–330, 2006.

[9] T. Ito, S.-I. Nakano, Y. Okamoto, Y. Otachi, R. Uehara, T. Uno, and Y. Uno. A polynomial-time approximation scheme for the geometric unique coverage problem on unit squares. In *Algorithm Theory–SWAT 2012*, pages 24–35. 2012.

[10] T. Ito, S.-i. Nakano, Y. Okamoto, Y. Otachi, R. Uehara, T. Uno, and Y. Uno. A 4.31-approximation for the geometric unique coverage problem on unit disks. *Theoretical Computer Science*, 544:14–31, 2014.

[11] Y. Matiyasevich. *Hilbert's Tenth Problem*. MIT Press, 1993.

[12] N. Misra, H. Moser, V. Raman, S. Saurabh, and S. Sikdar. The parameterized complexity of unique coverage and its variants. *Algorithmica*, 65(3):517–544, 2013.

[13] N. Misra, V. Raman, S. Saurabh, and S. Sikdar. The budgeted unique coverage problem and color-coding. In *Computer Science-Theory and Applications*, pages 310–321. 2009.

[14] H. Moser, V. Raman, and S. Sikdar. The parameterized complexity of the unique coverage problem. In *Algorithms and Computation*, pages 621–631. 2007.

[15] S. Onn. *Nonlinear Discrete Optimization*. Zurich Lectures in Advanced Mathematics, European Mathematical Society, 2010.

[16] S. Onn and U. Rothblum. Convex combinatorial optimization. *Discrete and Computational Geometry*, 32:549–566, 2004.

[17] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.