Covering Moving Points with Anchored Disks

C. Bautista-Santiago*, J.M. Díaz-Báñez†, R. Fabila-Monroy ‡,
D. Flores-Peñaloza§, D. Lara¶, J. Urrutia‖

July 21, 2011

Abstract

Consider a set of mobile clients represented by n points in the plane moving at constant speed along n different straight lines. We study the problem of covering all mobile clients using a set of k disks centered at k fixed centers. Each disk exists only at one instant and while it does, covers any client within its coverage radius. The task is to select an activation time and a radius for each disk such that every mobile client is covered by at least one disk. In particular, we study the optimization problem of minimizing the maximum coverage radius. First we prove that, although the static version of the problem is polynomial, the kinetic version is NP-hard. Moreover, we show that the problem is not approximable by a constant factor (unless P=NP). We then present a generic framework to solve it for fixed values of k, which in turn allows us to solve more general optimization problems. Our algorithms are efficient for constant values of k.

Keywords: Combinatorial optimization; Covering algorithms; Mobile objects; Client-server communication; Computational geometry.

*Instituto de Matemáticas, Universidad Nacional Autónoma de México, crevel@matem.unam.mx

†Departamento Matemática Aplicada II, Universidad de Sevilla, dbanez@us.es, partially supported by grant MEC MTM2009-08625.

‡Departamento de Matemáticas, Centro de Investigación y Estudios Avanzados deI Instituto Politécnico Nacional, Mexico City, Mexico , ruyfabila@math.cinvestav.edu.mx

§Facultad de Ciencias, Universidad Nacional Autónoma de México, dflorespenaloza@gmail.com

¶Instituto de Matemáticas, Universidad Nacional Autónoma de México, dlara@matem.unam.mx

‖Instituto de Matemáticas, Universidad Nacional Autónoma de México, urrutia@matem.unam.mx, partially supported by grant MEC MTM2009-08625 and by SEP-CONACYT of Mexico, Proyecto 80268.

1 Introduction

Let S be a set of satellites, each one moving along a different straight line over a region R in the plane, and let Q be a set of satellite ground stations located within R. In this paper we study the situation in which a signal alert, that emanates from the ground stations, must be broadcasted to all satellites. We assume that each ground station has a cost function which is linear on its range of transmission. Therefore, it is desirable to minimize the maximum power of transmission over all the stations.

We model this problem as follows: Represent the set of satellites as a set $S = \{p_1, \ldots, p_n\}$ of n points in the plane. Each point moves along a different straight line at constant speed –the speeds of the points are not necessarily the same–. The set of grounds stations is represented by a set $Q = \{q_1, \ldots, q_k\}$ of k points in the plane. Our objective is to find transmission times $t_1, \ldots, t_k$ and broadcast radii $r_1, \ldots, r_k$ for all the ground stations, such that the following happens:

1. Ground station q_j broadcasts the alarm signal at time $t_j, 1 \leq j \leq k$.
2. Each satellite p_i is reached by at least one ground station q_j at the moment this one broadcasts. That is, at time t_j, p_i must lie within distance r_j from q_j.
3. The largest r_j, $1 \leq j \leq k$, is as small as possible.

This problem is closely related with the k-center problem, however as we will show later, is not a particular case. The k-center problem is defined as: given a set P of points (*clients*) in a metric space $\mathbb{M}$, a k-center of P is a set of k points (*servers*) in $\mathbb{M}$ such that the maximum distance between a client and its closest server is minimized. If $\mathbb{M}$ is the Euclidean space, then the problem is known as the *Euclidean k-center problem*. If in addition, the set of servers is required to be a subset of P, then the problem is known as *discrete Euclidean k-center*. In what follows we will refer to the elements of P alternately as points or clients, and to the elements of the k-center of P alternately as servers or base stations.

The design and deployment of networks where the set of clients move over time, such as mobile wireless communication networks, gave origin to many variations of the k-center problem in mobile environments. We review some results in Section 2. In this context, different scenarios arise depending on whether or not the clients and servers move. One such posible scenario is the *static client/static server* problem. In such a problem, the transmission radii of each base station is given, and the set of clients is represented by a set of points in the plane. The locations of the servers must be a subset of the set of points representing clients. The goal is to select a minimum subset of locations to place base stations, such that each client is covered by at least one base. This is a well-known NP-hard problem, named the *discrete disk cover problem* with numerous applications, in particular in wireless network design [8].

In the opposite scenario, that is, when the set of clients and the set of potential locations move continuously over time (*mobile-client/mobile-server*), various results have been obtained. For instance, in [17], a randomized algorithm that approximates the discrete Euclidean k-center in the case of mobile-client/mobile-server is presented. Two hybrid scenarios are the *static-client/mobile-server* and the *mobile-client/static-server*. The first scenario is related to travelling salesman problems. The second scenario, allows the definition of several allocation problems –where each mobile client must be assigned to a base station–. Recently, Tayi et al. [34] considered the problem of assigning clients to existing base stations in order to minimize data access costs under base station load constraints. Contrary to the other scenarios, the *mobile-client/static-server* one has scarcely been

considered. The problem we study in this paper can be seen as a variation of the discrete Euclidean k-center problem for moving clients and fixed servers. Let us introduce the model formally as a covering problem under the mobile-client/static-server framework.

Let q be a point in the Euclidean two-dimensional space $\mathbb{E}^2$, and let $B_q(t, r)$ be a disk centered at q representing a server positioned at point q. The server transmits an instant message only once at time t, and with *transmission range* equal to r. Let p be a point moving along a straight line in $\mathbb{E}^2$ and with constant velocity. We say that p *receives* the message that $B_q(t, r)$ transmits if at time t (that is when the server sends the message), the point p is within the transmission range of the server; i.e. if at time t it happens that $d(q, p) \leq r$, where $d(\cdot)$ denotes the Euclidean distance. We call t the *activation time* of $B_q(t, r)$. Our problem is the following:

Given: a set $S = \{p_1, \ldots, p_n\}$ of points representing clients moving over straight lines at constant velocity within an interval of time, and set of servers $Q = \{q_1, \ldots, q_k\}$ that all transmit the same message, and that are positioned at given fixed points,

determine: for each server q_j an activation time t_j and a transmission range r_j, such that by the time the last server has been activated, all clients have received the message and the maximum transmission range is minimized.

We call this problem *minimax anchored covering set problem (minimaxACS-problem)*. In what follows we will refer to the elements of S alternately as satellites, clients or moving points, and to the elements of Q alternately as ground stations or servers. Also, from now on, we denote by $B_j(t, r)$ the disk centered at q_j with activation time t and radius r. In the next paragraphs we include some observations.

We call $\mathcal{C} = \{B_1(t_1, r_1), \ldots, B_k(t_k, r_k)\}$ an *anchored covering set (ACS)* if for every p_i there is a q_j such that p_i is contained in the disk $B_j(t_j, r_j)$ at time t_j. Let $C(B_j(t_j, r_j))$ be the set of clients covered by $B_j(t_j, r_j)$. We say that two anchored covering sets are equivalent if the mobile clients are covered by the same disks in both sets. Then the minimaxACS-problem is to find an anchored covering set (ACS) for a set of moving clients such that its maximum radius is minimized.

For simplicity's sake, we assume that the points representing clients are in *general moving position*, that is, there are no three points in S that are, at the same time, at the same distance from any q_j in Q. Degenerate positions can be removed by using standard techniques.

It is important to clarify that, contrary to classic k-center type problems, in ours the set of k servers is given. Besides, the fact that each server transmits the message only once and that this is ephemeral, makes our problem completely different to any k-center variant. Also notice that, if the clients were static, the solution to our problem would be given by computing the Voronoi diagram of the k servers. If, on the contrary, the clients were moving but the communication between clients and servers has to be maintained at all times, then the solution would be again be given by computing the Voronoi diagram of the servers and changing the server associated with a client whenever this changes from one Voronoi cell to another. Such problems have been considered in the server location area [28, 1].

Finally, we would like to point out that the well-known KDS structures [21] cannot be adapted to our problem. To the best of our knowledge, KDS are used in problems in which one is interested in maintaining some geometric structure over time, which is not the case in this work.

We do understand that the geometrical model we are proposing may not be valid in some real cases. However, achieving solutions to simpler models may shed light in the development of methods for more complex models. In fact, some generalizations are given in Section 6. Let us also note that

although the centers are given in advance and the minimaxACS-problem can be considered "easier" than the k-center problem, we prove it is also NP-complete. Moreover, we show that there is no polynomial-time constant approximation algorithm for the minimaxACS-problem.

The remainder of the paper is structured as follows. Section 2 summarizes previous work on related problems. Section 3 is devoted to proving the NP-hardness of the minimaxACS-problem. Section 4 depicts a general framework that is useful for solving several related optimization problems. Section 5 provides a more specific algorithm for the minimaxACS-problem which runs in polynomial-time for a constant value of k. Finally, Section 6 concludes with some generalizations and some directions for future research.

2 Related Work

There exists a vast amount of covering problems, some of which are static and others are mobile. We outline some known results. Several variants of the disk covering problem have been widely studied for the cases where sets, the potential locations for base stations, and the clients, are all static points in the Euclidean plane [29]. In general, the task is to select a number of locations q_j for the base stations, and assign a transmission range r_j to each q_j such that for a given set $\{p_1, \ldots, p_n\}$ of n clients, each client p_i is covered. We say that client p_i is covered if and only if p_i is within the range of some transmission point q_j, that is, $d(q_j, p_i) \leq r_j$. The resulting cost per base station is some known function f, for example $f(r) = r^\alpha$. If the goal is to minimize the maximum cost over all placements of k servers that cover the set of clients and the base stations can be located anywhere in the Euclidean plane, then the problem is known as the *Euclidean k-center problem* [30], which is NP-hard. It is also NP-hard to approximate it within an error factor of 1.82 [15]. Many heuristics give a 2-approximation, the first of these is in [19]. When k is part of the input and the transmission ranges are fixed, although the problem is still NP-hard, it is possible to find polynomial time approximation schemes (PTAS) [20, 24].

On the other hand, if a set of possible locations is given, the problem is known as the *discrete Euclidean k-center problem*, which is also NP-complete [16] and it is known to admit a PTAS.

The case when the set of potential base station locations is a subset of the set of clients is studied in [7]. When points are in $\mathbb{R}^2$, an algorithm that computes a $(1+\epsilon)$-approximation in time $n^{O(1/\epsilon^4)}$ is given for $\alpha = 1$, and in time $n^{O(\alpha^4/\epsilon^6)}$ for any α. These results are generalizable to higher dimensions. However, if k is fixed, it is well known that an optimal solution can be found in polynomial time by enumerating the $O(n^k)$ possible solutions. An algorithm for the Euclidean k-center in the plane that runs in $O(n^{O(\sqrt{k})})$ time can be found in [27]. For the case when $k = 2$ a deterministic near-linear $O(n \log^2 n \log^2 \log n)$ time algorithm is given in [10]. The discrete problem is closely related to the *discrete unit disk cover problem* where a set $\mathcal{D}$ of unit disks of fixed location is given, and the goal is to find a minimum-cardinality subset $\mathcal{D}' \subset \mathcal{D}$ that cover the set of clients. This problem is a geometric set cover problem that allows for a constant factor approximation [9]. Recently, a PTAS was also proposed in [31].

In the mobile case, when the set of clients and the set of potential locations move continuously, various results have been obtained. A kinetic data structure (KDS) for maintaining an ϵ-approximation of the Euclidean 1-center of a set of (linearly) moving points in the plane is used in [2]. Such a data structure uses $O(1/\epsilon^{5/2})$ events (combinatorial changes) and spends a total of $O((n/\sqrt{\epsilon}) \log n)$ time at those events. In [17], a randomized algorithm to approximate the discrete Euclidean k-center of a set of moving points in the plane is given for the case when the radii are fixed. This algorithm chooses and maintains as centers a subset of points such that the number of centers selected is a

constant-factor approximation of the minimum possible number. This is done by means of a KDS, which, as the points move, updates the centers as necessary. In [23], given a set of moving points in $\mathbb{R}^d$, a static k-center is computed such that at any time this static k-center is competitive with the optimal k-center at that time. That is, if in the optimal solution the number of centers is k and the points move with degree of motion μ, then this scheme guarantees a $2^{\mu+1}$-approximation of the radius with $k^{\mu+1}$ centers chosen from the set of input points before the points start to move. An efficient and compact KDS for maintaining the diameter, width and smallest area or perimeter bounding rectangle of a set of moving points in the plane can be found in [2]. This paper includes a $(1+\epsilon)$-approximation of the mobile Euclidean 1-center. See [14] for approximation algorithms to the mobile Euclidean 2-center. Finally, the authors of [6] focus on exact and approximate versions of the Euclidean 1-center problem for a set of moving points in the plane where the center is constrained to have bounded velocity. They demonstrate that the velocity of the exact Euclidean 1-center can be arbitrarily high. We also refer to [13] for a study on the behavior of the mobile 1-center problem.

Another variant of the last problem is that of finding flocks in a given set of moving data. In [5] a flock is a set of m points such that for every discrete time step in a given interval of time, there is a disk of a given radius that contains all m points. The movement of the points is described by means of trajectories given as polygonal lines that can intersect themselves. The authors present various algorithms for reporting flocks.

Unlike some of the papers mentioned above, we refrain from using KDS since the covering disks may all exist at different times. The approach of maintaining a solution through time does not seem adequate. Instead, we choose to make the problem somewhat more static. We do this by translating the problem into a merely combinatorial one. This is described in detail in Section 4.

3 MinimaxACS is NP-hard

In this section we prove that the minimax anchored covering set problem is NP-hard, and moreover, that there is no polynomial-time constant approximation algorithm for it unless P=NP. Notice that, since the centers are fixed in our problem, the hardness of the discrete Euclidean k-center problem does not imply hardness of the minimaxACS-problem. Moreover, the static version of the problem can be easily solved in polynomial time by using the Voronoi diagram of the k centers, that is, the optimal allocation is given by the Voronoi regions. It is interesting to observe that the hybrid version of the *mobile-client/static-server* minimax problem is NP-hard, and the proof is not a direct reduction from the static version. We give a proof that uses a static problem for the reduction and permits to introduce, in some sense, movement into a static scenario. The study of hybrid scenarios may provide surprising results on the hardness of the problems. See [12] for an example in which restricted motion is also considered.

For showing the NP-hardness, the reduction is from the p-PairSupplier problem [25], a restricted version of the p-center optimization problem that can be stated as follows:

Instance: Given a complete weighted bipartite graph $G = (V \cup U, E)$, where V and U are disjoint sets of vertices such that $|U| = n$ and $V = V_1 \cup \ldots \cup V_p$ is the union of p pairwise disjoint sets of vertices, each one with cardinality two. Let $w_{i,j} \geq 1$ be the weight of the undirected edge $\{v_i, u_j\}$, where $v_i \in V$ and $u_j \in U$.

p-PairSupplier problem: Select a set C of p nodes, one from each V_i such that the maximum distance (weight) of any node in U to its nearest neighbor in C is minimized.

The decision problem associated with the p-PairSupplier problem is NP-complete [25, 26]. Furthermore, there is no polynomial-time α-approximation algorithm for the p-PairSupplier problem for any constant α, unless $P = NP$ [25].

Theorem 1. *The minimaxACS-problem is NP-hard. Furthermore, there is no polynomial-time α-approximation algorithm for the minimaxACS-problem for any constant α, unless P=NP.*

Proof. Consider the decision problem for minimaxACS: *Is there an Anchored Covering Set of cost (maximum radius) less than or equal to w?* Clearly this problem is in NP. We now proceed to prove its completeness.

Given an instance $G = (V \cup U, E)$ of the p-PairSupplier problem, we construct an instance of the ACS problem as follows. The key idea is to put the centers both in the vertices of V and U, and moving clients representing the edges of G in such a way a covering set in an instance implies a covering set in the other one. The details of the construction are as follows.

Let w_{max} be the largest weight of the edges of G. We first place the centers and the mobile clients. The radius and activation time for each disk centered at them will be specified later. Refer to Figure 1:

- For each vertex $v_i \in V$, $1 \leq i \leq 2p$, place a center q_i such that:

$$q_i := \begin{cases} (i \cdot (-3w_{\max}), 0) & \text{if i is odd,} \\ (i \cdot (-3w_{\max}), 2w_{\max}) & \text{if i is even.} \end{cases}$$

- For each vertex $u_j \in U$, $1 \leq j \leq n$, place $2p - 1$ centers $q_{2p+j} = (j \cdot (-3w_{\max}), y)$ (we will choose an appropriate value for y below). Note that we put $2p + (2p - 1)n$ centers in total.

- We now place $2pn$ mobile clients as follows. Let $w_{i,j}$ be the weight associated with the edge $\{v_i, u_j\}$ in G. For every $0 \leq i \leq 2p$ and every $1 \leq j \leq n$, we place a mobile client $c_{i,j}$ at distance $w_{i,j}$ from q_i at time $t = 0$ and moving on a straight line from its current position (at $t = 0$) towards q_{2p+j}.

  To keep the mobile clients in good working order, some details are needed. Firs, take note that we can choose y big enough so that the distance from every $c_{i,j}$ to every point q_k (other than q_i and q_{2p+j}) is always greater than $w_{\max}$. This can be guaranteed since as y tends to infinity, the trajectories of the mobile clients tend to vertical lines.

  On the other hand, speeds for the mobile clients $c_{i,j}$ can be chosen so that they arrive sequentially to every q_{2p+j}, that is, at every instant there is at most one mobile client at distance less or equal to $w_{\max}$ from q_{2p+j}.

- Finally, for each pair of centers (q_i, q_{i+1}) (i odd and $1 \leq i \leq 2p - 1$), we add an *extra mobile client* c_i^*. The trajectory of c_i^* is the directed line passing through q_i and q_{i+1}. See Figure 1. We choose the initial position and velocity of each point c_i^* in such a way that, by the time c_i^* reaches q_i, all initial mobile clients $c_{i,j}$ have arrived at their respective point q_{2p+j}. The purpose of considering these extra mobile clients (as we will see later) is to use all centers to cover all moving clients. Note that we put $2pn + p$ mobile clients overall.

We are now ready to show the reduction. First, we will prove that for a solution of the p-PairSupplier problem with cost at most w we can obtain an anchored covering set (ACS) with maximum radius w. Secondly, we will show that an anchored covering set of radius at most w generates a solution for the p-PairSupplier problem with cost at most w. Suppose that there is a solution to the instance

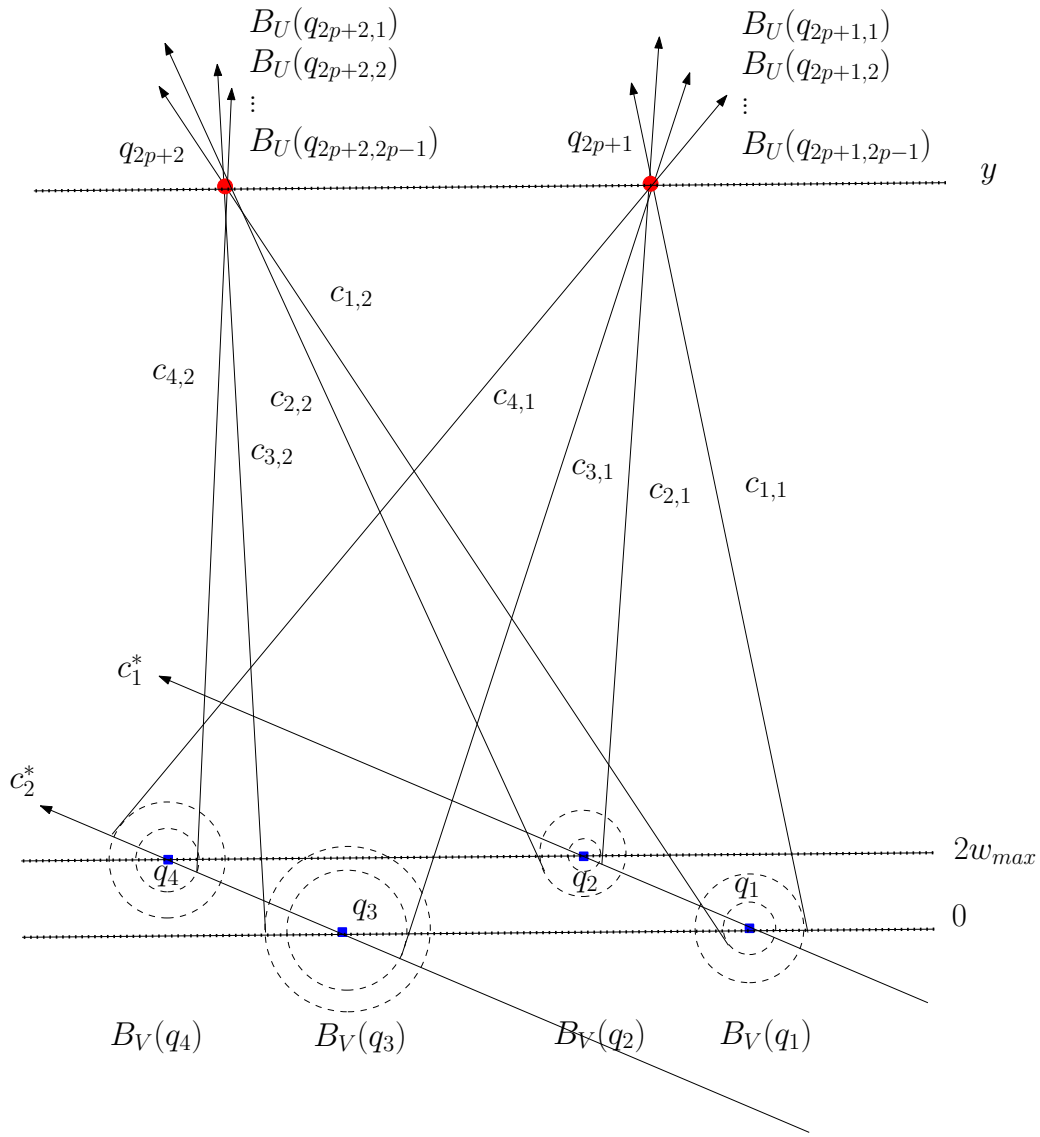

Figure 1: Centers, anchored disks and mobile clients for the NP-hardness reduction.

of the p-PairSupplier problem with a cost less than or equal to w, that is, we have a set C of selected suppliers (Figure 1 bottom) of V such that the maximum distance to its associated clients (Figure 1 top) is w. Keep in mind that a supplier $v_i \in C$ *serves* a client $u_j \in U$ if there is not a supplier v_k, $k \neq i$, such that $w_{k,j} < w_{i,j}$ in G. For every $v_i \in C$, let u_{v_i} be the client served by the v_i farthest away from it, and let w_{v_i} be the weight of the edge $\{v_i, u_{v_i}\}$ in G. Thus, let $w = max\{w_{v_i}, 1 \leq i \leq p\}$ and $B_V(q_i)$, $1 \leq i \leq 2p$, be the disk centered at q_i with coverage radius w_{v_i}, and whose activation time is the first instant in which a mobile client is inside its coverage range.

By construction, we know that at least n mobile clients of the type $c_{i,j}$ are covered by disks $B_V(q_i)$. Therefore, there are at most $(2p-1)n$ mobile clients that are not covered by such disks, all these of type $c_{i,j}$. However, this is exactly the number of centers located at vertices of U, and each one of these mobile clients, by construction, is within the coverage range of at least one disk of type $B_U(q_{2p+j})$. Therefore, all of the remaining mobile clients $c_{i,j}$ are covered by such disks with radius 0 and activation time whenever they arrive at q_{2p+j}. Finally, the extra mobile clients of type c_i^* are covered later by the non-activated disk $B_V(q_i)$ (with radius 0). Thus, there exists an Anchored Covering Set (i.e. a radius and an activation time for every center) that covers all mobile clients with a maximum radius w.

On the other hand, a solution of radius at most w for the anchored covering set problem activates at most one supplier disk $B_V(q_i)$ of each pair, otherwise some extra mobile client c_i^* is not covered. Therefore, such a solution generates a selection of cost w for the p-PairSupplier problem, and the reduction follows.

Note that there are $2pn$ mobile clients c_{ij} and p extra mobile clients c_i^*. Since each of them can be calculated independently by solving polynomials, the instance of the minmaxACS can be constructed and the whole reduction can be done in polynomial time. We also note that this reduction preserves approximability and therefore there is no constant factor approximation algorithm for the minimaxACS unless P=NP. This finishes the proof. ■

4 Framework

Having established the complexity of the minimaxACS-problem, we now provide a generic framework that will be useful for solving a set of optimization problems. This framework will allow us to generate all non-equivalent covering sets in a Gray code-like order, in that two consecutive covering sets differ in exactly one disk. We first give a detailed description of the framework and the approach to generate it, and then we analyze the complexity of the method. In the following section we will be able to optimize our framework further in order to solve the minimaxACS-problem more efficiently.

4.1 Description

Roughly speaking, our framework makes use of arrangements of curves which are defined as follows. Let $d_{j,i}(t)$ be the function defined by the squared Euclidean distance between the j-th static center and the i-th mobile client, that is, between q_j and $\sigma_i(t)$, at time t. We use the squared distance to simplify our analysis and denote by $\mathcal{A}_j$ the arrangement of curves defined by the functions $d_{j,i}(t)$, $1 \leq i \leq n$ (Figure 2). Note that when we refer to a point (t, r) in $\mathcal{A}_j$ we actually mean the point (t, r^2), and also that (t, r) can be associated with $B_j(t, r)$, the disk centered at q_j at time t with radius r. Analogously, each disk centered at the point q_j can be identified with a point in $\mathcal{A}_j$. Note

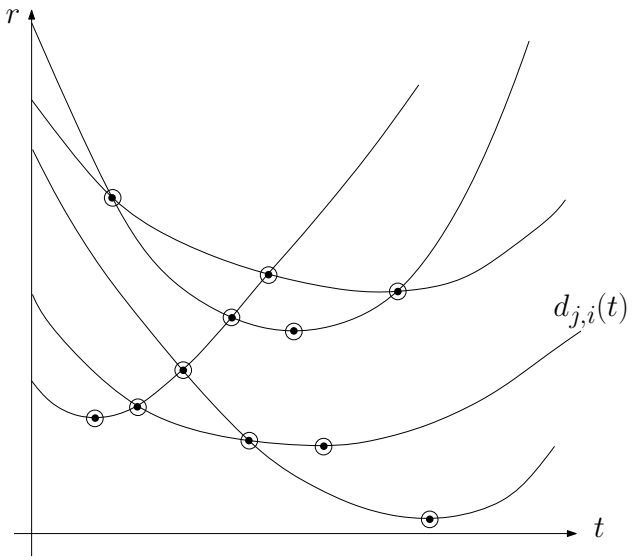

Figure 2: Arrangement of curves $\mathcal{A}_j$ and its critical points.

that $C(B_j(t,r))$, the set of points covered by $B_j(t,r)$, consists of the set of clients represented by the curves below (t,r).

Let f be a face in $\mathcal{A}_j$. An edge e of f is on the *lower boundary* of f if and only if there are no other edges of f below e. From now on, a face f in $\mathcal{A}_j$ will be composed of its interior together with its lower boundary. As a consequence, any two disks represented by two points in f cover exactly the same clients.

Certain points of these arrangements allow us to discretize the search space of the covering sets. These points are located either at the minimum of the functions $d_{j,i}(t)$ or at the intersection of two such functions. We will refer to them as *critical points* (marked points in Figure 2). The following lemma shows that every anchored disk can be realized as one of the critical points. Therefore from now on, we will assume that all disks are given by these points.

Lemma 1. *For every $B_j(t,r)$, there exists $B_j(t',r')$ such that $r' \leq r$, (t',r') is a critical point in $\mathcal{A}_j$ and $C(B_j(t,r)) = C(B_j(t',r'))$.*

Proof. Without loss of generality, suppose that $C(B_j(t,r))$ is not empty. Let $\sigma_i(t)$ in $C(B_j(t,r))$ be the point farthest away from q_j at time t. Let r'' be the distance between q_j and $\sigma_i(t)$. Clearly, both disks, $B_j(t,r)$ and $B_j(t,r'')$, cover exactly the same clients. Let (t^*,r^*) be the minimum of $d_{j,i}(t)$. From (t,r''), move continuously along $d_{j,i}(t)$ towards (t^*,r^*) until a critical point (t',r') is reached. Observe that this point may be (t^*,r^*). Therefore $C(B_j(t,r)) = C(B_j(t',r'))$ and $r' \leq r$. ■

4.2 Approach

We will now proceed to describe a method that allows the generation of all non-equivalent possible solutions to a generic optimization problem. In order to find a covering set, k disks must be chosen (one per arrangement). To preserve the simplicity of the presentation, we first describe how the k-th disk is selected, assuming that the first $k-1$ have already been chosen. We will then present a procedure to choose these first $k-1$ disks.

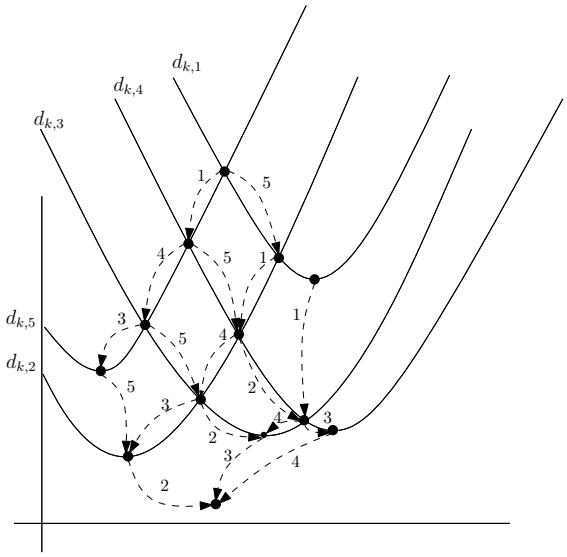

Figure 3: The pointer data structure on the arrangement $\mathcal{A}_k$. Edges are represented by dashed lines.

The strategy for selecting the k-th disk is based on the observation that once the first $k-1$ disks are chosen, the k-th disk must cover all clients not yet covered. Therefore its corresponding point in $\mathcal{A}_k$ must be above the curves representing the clients that have not yet been covered. In other words, this point must lie in the upper envelope [3] of the subarrangement consisting of such curves. Since any point in the upper envelope will do, we choose the point with minimum height.

At this point, it would be natural to proceed by dynamically updating the upper envelope of $\mathcal{A}_k$, as the first $k-1$ disks are selected. We refrain from this approach since, as we will show later, the upper envelope may undergo a change in linear size. Instead, we present a simpler static data structure that enables us to find the k-th disk in time $O(n)$.

Consider the points in $\mathcal{A}_k$ which are of minimum height in their corresponding faces; note that they must be critical points. We construct a directed graph having these points as vertex set. Let $min(f)$ be the minimum point of face f. We add a directed edge with label i from $min(f)$ to $min(f')$ for two neighboring[1] faces f and f' if any disk represented by a point in f covers the i-th client, and if this cannot be done using a point in f', that is, if for every $(t,r) \in f$ and $(t',r') \in f'$, $C(B_k(t,r)) = C(B_k(t',r')) \setminus \{\sigma_i\}$. Finally, we add an artificial point representing the possibility of not placing any disk at the k-th center (that is, considering a disk with radius 0) and we join it with the other vertices by the same rule as above. Refer to Figure 3.

In order to find the k-th disk, we start from the minimum point of the upper envelope of $\mathcal{A}_k$ and follow a directed edge if its label corresponds to a client that has already been covered. To keep track of covered clients we maintain an array A which stores in $A[i]$ we store the number of times the i-th client is inside a disk placed at any of the first $k-1$ centers. Eventually we will arrive at a critical point whose edge we can no longer follow. This point corresponds to the desired k-th disk.

We now describe the Gray code-like order in which we generate the first $k-1$ disks. See [35, 22] for more information on Gray codes in combinatorial algorithms.

Consider two neighboring faces in $\mathcal{A}_j$ for $1 \leq j \leq k-1$. Note that two disks represented by points in neighboring faces in $\mathcal{A}_j$ differ by only one in the points they cover. For every $\mathcal{A}_j$, let $\mathcal{D}_j$ be its

[1]We say that two faces in an arrangement are neighbors if they share an edge.

dual graph. $\mathcal{D}_j$ is the graph having as vertex set the faces of $\mathcal{A}_j$, two of them adjacent if they are neighbors in $\mathcal{A}_j$.

The faces of a single $\mathcal{A}_j$ can be visited using DFS (*Depth First Search*) [11] on the dual graph $\mathcal{D}_j$. Each time a face f is visited in $\mathcal{A}_j$ we take its minimum point (t, r) and consider the disk $B_j(t, r)$. Note that some vertices may be visited more than once. In each step we update the array A.

To generate the set of all possible $k-1$ disks, we first run DFS on $\mathcal{D}_1$, stopping at each vertex to recursively visit the faces of $\mathcal{D}_2, \ldots, \mathcal{D}_{k-1}$. That is, we will then run DFS on $\mathcal{D}_2$ stopping at each vertex to do a DFS on $\mathcal{D}_3$ and so on. Two consecutive sets visited in this order differ in only one disk. This disk, as noted before, covers one point more or one point less than the previous disk. For this reason, we mentioned earlier that this visiting is done in a Gray code like manner.

4.3 Complexity

We now analyze the time and space complexity of the construction of the framework that solves a class of optimization problems in a mobile-client/static-server scenario.

Since distance functions $d_{j,i}(t)$ intersect pairwise at most twice, each arrangement of curves, and thus its dual graph, can be constructed in $O(n^2)$ time and space [3]. For the same reason, there are $O(n^2)$ critical points in each arrangement. Therefore, the number of ways of choosing a disk in each of the k arrangements is $O(n^2)$; hence, there are $O(n^{2k})$ non-equivalent covering sets. However, for our purposes, only $O(n^{2(k-1)})$ non-equivalent covering sets suffice, since we assume that the k-th disk is defined by the first $k-1$ disks.

The traversal of each dual graph $\mathcal{D}_j$ takes $O(n^2)$ time ($\mathcal{D}_j$ is planar) [11]. Thus, it takes $O(n^{2(k-1)})$ time to traverse all the first $k-1$ arrangements.

Let us remember that in order to obtain the k-th disk, we use the directed graph constructed above on the arrangement $\mathcal{A}_k$, in which we follow a directed edge labeled with i iff $A[i] \geq 1$. In this path, at most n edges are followed since no two edges in the path have the same label. Thus, the k-th disk is selected in $O(n)$ time.

Given the Gray code property of the traversal of the first $k-1$ disks, two consecutive sets of $k-1$ disks differ only in one disk. Therefore, the array A can be updated in constant time.

In summary, we have:

Theorem 2. *Let $\mathcal{C}$ be an ordered collection of non-equivalent covering sets, and let $\mathcal{R}$ be a range assignment cost function on $\mathcal{C}$. If two consecutive covering sets C' and C'' of $\mathcal{C}$ differ in at most one disk, then $\mathcal{R}$ can be optimized in $O(n^{2k-1})$ time and $O(n^{2(k-1)})$ space.*

As a consequence of this result we have a generic approach for solving a class of combinatorial optimization problems. For instance, a fundamental problem in Ad-Hoc Wireless Networks is the so-called *minimum energy range assignment problem* [33], where the transmission range of a station depends on the energy invested by the station. In particular, the *power* required by a station s_i to correctly transmit data to another station s_j is modeled as $d_2(s_i, s_j)^\alpha$, where $d_2(.)$ is the Euclidean distance and $\alpha \geq 1$ is the *distance-power gradient*. Our approach may be used to discretize the problem in the mobile-client/static-station version and solve it in $O(n^{2k-1})$ time. Note also that this framework would yield a solution to the minimaxACS in $O(n^{2k-1})$ time. We are able to further optimize this framework for the minimaxACS problem in the next section.

As we mentioned earlier, instead of using the directed graph we introduced, one could consider making dynamic updates to the upper envelope in $\mathcal{A}_k$. Such a dynamic data structure for main-

taining the upper envelope of a set of straight lines does exists [32]. This data structure maintains the upper envelope of n straight lines under insertions and deletions with a cost of $O(\log^2(n))$ per operation. However, in the case of quadratic functions, such a data structure might unfortunately need linear time for updating. This is because, after an insertion or a deletion, there could be a change of linear size in the upper envelope. For instance, take any quadratic polynomial $p(x)$ with a leading coefficient greater than one. Choose any m real numbers $x_1 < x_2 < \cdots < x_m$. For every $1 \leq i \leq m$, let $q_i(x) = p(x) - (x - x_i)^2$. Note that $q_i(x)$ is a polynomial of degree 2 and that $p(x) - q_i(x) = (x - x_i)^2$. Therefore, the graphs of $p(x)$ and $q_i(x)$ only intersect at $(x, p(x))$. Choose $\epsilon > 0$ so that the upper envelope of $p(x), q_1(x) + \epsilon, \ldots, q_m(x) + \epsilon$, as seen from left to right, consists of $p(x), q_1(x), p(x), \ldots, p(x), q_m(x), p(x)$. Note that $p(x)$ intersects the upper envelope of $q_1(x), \ldots, q_m(x)$ a linear number of times. Thus, the change after an insertion or deletion in the upper envelope of polynomials of degree 2 can be linear.

Due to previous comment, it seems to be adequate to use our simpler data structure, since it is easy to implement and the cost for finding the k-th disk is also linear.

5 Algorithm for the minimaxACS-problem

In this section we show an alternative algorithm for the minimaxACS-problem that is efficient for small constant values of k. The strategy is commonly used to solve optimization problems with the minimax criterion. The idea is to use the associated decision problem to apply binary searching.

5.1 The Fixed Radius Decision Problem

Consider the the *fixed-radiusACS problem*: *Decide if an anchored covering set exists for a given fixed radius r.*

As we did previously, we discretize the problem as follows. Let $d(t)$ be the constant distance function with value r. $d(t) = r$ is a horizontal line in the arrangement $\mathcal{A}_j$. For every anchored covering set with fixed radius $\mathcal{C} = \{B_1(t_1, r), \ldots, B_k(t_k, r)\}$ there exists another anchored covering set $\mathcal{C}' = \{B_1(t'_1, r), \ldots, B_k(t'_k, r)\}$ such that (t'_j, r) is an intersection point of the line $d(t) = r$ and the function $d_{j,i}(t)$ in $\mathcal{A}_j$, and $C(B_j(t_j, r)) \subseteq C(B_j(t'_j, r))$. Therefore we will only look at disks given by these intersection points.

In order to compute the possible covering sets, we proceed as before: first we generate the first $k-1$ disks in a Gray code like manner, and then we determine if there exists a k-th disk that covers the remaining clients. To generate the first disks, we visit the intersections of the line $d(t) = r$ with the distance functions in $\mathcal{A}_1$ in time order, stopping at each point to recursively visit the intersection points for $\mathcal{A}_2, \ldots, \mathcal{A}_{k-1}$. Note that two sets of disks visited in this order differ by at most one in the mobile clients they cover.

Let us assume that the first $k - 1$ disks have been chosen. We will now proceed to find the last disk. Since the radius is fixed, there may be clients that can never be covered by a disk centered at q_k, that is, clients that are always at distance greater than r from q_k. Let us take note that if these clients are not covered by the first $k-1$ disks, no covering set using these k disks exists. Therefore, it is important to keep track of these clients so that we know in constant time whether the clients not reachable from q_k are covered.

Let us consider the case where all clients unreachable from q_k have been covered. In $\mathcal{A}_k$, each client not yet covered by the first $k - 1$ disks defines an interval in the horizontal line $d(t) = r$ whose

distance to q_k is less than or equal to r. Thus, a solution exists if and only if the intersection of all these intervals is non-empty. Every interval has a starting point and an ending point. A solution then exists if and only if the last starting point lies before the first ending point in time. This can be easily done and maintained in time $O(\log n)$ as points are added or deleted.

Every $d_{j,i}(t)$ intersects line $d(t) = r$ in at most two points in $\mathcal{A}_j$. Therefore, there are at most $2n$ such intersections in every $\mathcal{A}_j$ and all intersections can be found in time $O(kn)$. Sorting the intersection points in all the arrangements can be done in $O(kn \log kn)$ time. Therefore we conclude:

Theorem 3. *The fixed-radiusACS problem can be solved in $O(n^{k-1} \log n)$ time.*

5.2 The minimaxACS-problem

We use the above approach as a sequential algorithm to solve the minimaxACS-problem. Note that for every $B_j(t_j, r_j)$ in an anchored covering set, we have that $C(B_j(t_j, r_j)) \subseteq C(B_j(t_j, r))$ for any $r > r_j$. Therefore, we may assume that all the disks in a solution for the minimaxACS-problem have the same radius r.

As we stated before, the solution must be given by a set of critical points, and since there are $O(kn^2)$ critical points in total, this is also the number of candidates for radius in a solution. By sorting all these radii in $O(kn^2 \log(kn))$ time and running a binary search on them using the fixed radius decision algorithm, we arrive at the main result of this section.

Theorem 4. *The minimaxACS-problem can be solved in $O(n^{k-1} \log n \log(kn))$ time.*

6 Conclusions and Future Work

In this paper we studied the complexity of a kinetic constrained covering problem where the centers are fixed in advance and the points to cover are moving on straight lines at constant velocity. A specific problem, minimizing the maximum radius, has been studied in detail and a proof of NP-hardness was provided. We hope the key idea of our proof can be used to prove the NP-hardness of similar problems whose corresponding static versions are polynomial. A general framework was developed to enumerate all combinatorially distinct candidate values for a collection of optimization problems. Finally, an efficient algorithm for constant values of k was also presented for the minmaxACS-problem.

Although the geometric model studied in this paper is the simplest one, the same strategy can be used for more general settings. Let us give some generalizations. Firstly, the problems were presented in the plane, it is straightforward to use the same techniques to solve them in arbitrary dimensions. Note that in the more general setting of $\mathbb{R}^d$, the distance functions to a given anchored point are also quadratic on t and the curve arrangements can be constructed in the same way. Once the curve arrangements A_j are constructed, the discussion follows exactly as in the plane.

Moreover, no extra difficulty is involved if the anchored points are now allowed to move along straight lines at constant speed, since the distance functions from the mobile clients to any fixed (now moving) center are again quadratic.

On the other hand, the complexity of the problem does change if instead of linear motion, an algebraic motion of degree m is allowed. In this case the position of every moving point set $\sigma_i(t)$ is given by $\sigma_i(t) = (\sigma_i^1(t), \sigma_i^2(t))$, where $\sigma_i^j(t)$ is a polynomial of degree at most m on t. For this case, a similar approach can be taken by constructing the curve arrangements of the squared distance

functions to each anchored point. The squared distance functions will now be polynomials of degree at most $2m$. Two squared distance function curves in a given arrangement will intersect at most $2m$ times, and the curve arrangement of every anchored point has complexity $O(mn^2)$. The same framework can be used by replacing every quadratic term n^2 by mn^2 in its complexity. It remains to be seen how much this complexity can be reduced.

Other possible variations rely on the objective function. The optimization problems dealt with the radii of the covering disks. Another class of problems to be studied is time optimization. For example, a restriction on the solution (such as fixed radius) might be given and we would be interested in the first such solution. This problem can also be solved using the general framework, but a better solution probably exists.

Finally, note that for a fixed k, the exhaustive enumeration of all candidate values of a range optimization problem provides polynomial-time algorithms. If, instead, k is allowed to vary, the algorithms become exponential on k. We conjecture that for a varying k, other optimization problems, such as minimizing the sum of radii, are NP-hard. Another interesting open problem for future research is to design faster exact exponential algorithms for practical situations in the sense of [36].

References

[1] A. Agnetis, E. Grande, P.B. Mirchandani and A. Pacifici. Covering a line segment with variable radius discs. Computers and Operations Research, 2009, 36 (5), 1423–1436.

[2] P. K. Agarwal, S. Har-Peled, Maintaining Approximate Extent Measures of Moving Points. In: Proceedings 12th ACM-SIAM symposium on Discrete algorithms, 2001, 148–157.

[3] P. K. Agarwal and M. Sharir. Davenport-Schinzel Sequences and Their Geometric Applications, Cambridge University Press, 1995.

[4] D. Barbará and T. Imielinski. Sleepers and workaholics: Caching strategies for mobile environments. In Proceedings of the ACM SIGMOD Conference on Management of Data, 1994, 1–12.

[5] M. Benkert, J. Gudmundsson, F. Hubner, T. Wolle. Reporting Flock Patterns. Lecture Notes in Computer Sciences, 2006, 4168, 660–671.

[6] S. Bereg, B. Bhattacharya, D. Kirkpatrick, and M. Segal. Competitive Algorithms for Maintining a Mobile Center. Mobile Networks and Applications, 2006, 11(2), 177–186.

[7] V. Bilò, I. Caragiannis, C. Kaklamanis, P. Kanellopoulos. Geometric Clustering to Minimize the Sum of Cluster Sizes. Lecture Notes in Computer Science, 2005, 3669, 460–471.

[8] G. Calinescu, I. Mandoiu, P.J. Wan, and A. Zelikovsky. Selecting Forwarding Neighbors in Wireless Ad Hoc Networks. MONET, 2004, 9(2), 101–111.

[9] P. Carmi, M. Katz, and N. Lev-Tov. Covering points by unit disks of fixed location. In Proc. ISAAC 2007, LNCS, Springer, 2007, 644–655.

[10] T. M. Chan, More Planar Two-Center Algorithms. Computational Geometry, 1999, 13(3), 189–198.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. Introduction to Algorithms, Second Edition. MIT Press, McGraw-Hill, 2001.

[12] J. M. Díaz-Báñez, R. Fabila-Monroy, D. Flores-Peñaloza, M. A. Heredia and J. Urrutia. Min-energy broadcast in mobile ad hoc networks with restricted motion. Journal of Combinatorial Optimization, 2011, published online, DOI 10.1007/s10878-011-9397-z.

[13] S. Durocher and D. Kirkpatrick. The Steiner Centre of a Set of Points: Stability, Eccentricity and Applications to Mobile Facility Location. International Journal of Computational Geometry and Applications, 2006, 16(4), 345 – 371.

[14] S. Durocher, D. Kirkpatrick. Bounded-Velocity Approximation of Mobile Euclidean 2-Centres. International Journal of Computational Geometry and Applications, 2008, 18(3), 161–183.

[15] T. Feder, D. Greene. Optimal Algorithms for Approximate Clustering, In: Proc. 20th ACM Symp. on the Theory of Computing, 1988, 434–444.

[16] R. J. Fowler, M. S. Paterson, S. L. Tanimoto. Optimal Packing and Covering in the Plane are NP-complete. Information Processing Letters, 1981, 12(3), 133–137.

[17] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang and A. Zhu. Discrete Mobile Centers. Discrete Comput. Geom., 2003, 30, 45–63.

[18] M. Gerla and J. Tsai. Multicluster, mobile, multimedia radio network. ACM–Baltzer Journal of Wireless Networks, 1995, 1(3):255–265.

[19] T. Gonzalez. Clustering to Minimize the Maximum Intercluster Distance. Theoretical Computer Science, 1985, 38, 293–306.

[20] T. Gonzalez. Covering a Set of Points in Multidimensional Space. Information Processing Letters, 1991, 40, 181–188.

[21] L.J. Guibas. Kinetic Data Structures: A State of the Art Report. Robotics: The Algorithmic Perspective. A K Peters, Ltd 1998, 191–209.

[22] M. Habib , L. Nourinel , G. Steiner. Gray codes for the ideals of interval orders. Journal of Algorithms, 1997, 25, 52–66.

[23] S. Har-Peled. Clustering Motion. Discrete Comput. Geom., 2004, 31(4), 545–565.

[24] D.S. Hochbaum, W. Maas. Approximation Schemes for Covering and Packing Problems in Image Processing and VLSI. Journal of the ACM, 1985, 32, 130–136.

[25] D. S. Hochbaum, A. Pathria. Generalized p-Center Problems: Complexity results and approximation algorithms. European Journal of Operational Research, 1997, 100, 594–607.

[26] O. Hudec. On alternative p-Center problems. Zeitschrift für Operations Research–Methods and Models of Operations Research, 1991, 36, 439–445.

[27] R. Z. Hwang, R. C. T Lee, R.C. Chang. The slab dividing approach to solve the Euclidean p-center problem. Algorithmica, 1993, 9, 1–22.

[28] S. Jadhav, A. Mukhopadhyay and B.K. Bhattacharya. An optimal algorithm for the intersection radius of a set of convex polygons. Journal of Algorithms, 20, 1996, 244–267.

[29] N. Lev-Tov, D. Peleg. Polynomial Type Approximation Schemes for Base Station Coverage with Minimum Total Radii. Computer Networks, 2005, 47, 489–501.

[30] N. Megiddo, K. J. Supowit. On the Complexity of Some Common Geometric Location Problems. SIAM J. on Computing, 1984, 30(1), 182–196.

[31] N. H. Mustafa and S. Ray. Improved results on geometric hitting set problems. Discrete and Computational Geometry, 2010, 44(4). 883–895.

[32] M.H. Overmars and J. van Leeuwen. Maintenance of Configurations in the Plane. J. Computer Syst. Sci., 1981, 23, 166–204.

[33] D. Peleg. Time-Efficient Broadcasting in Radio Networks: A Review. In Proc. of the 4th Int. Conf. on Distributed Computing and Internet Technology (ICDCIT), 2007.

[34] G. Tayi, D. Rosenkrantz and S. Ravi. Local base station assignment with time intervals in mobile computing environments. European Journal of Operational Research 57 (1,2) (2004), 267–285.

[35] H. S. Wilf, Combinatorial algorithms: an update, SIAM, 1989, ISBN 0-89871-231-9. Chapters 1–3.

[36] G.J. Woeginger. Exact algorithms for NP-hard problems: a survey. In: Combinatorial Optimization—Eureka! You shrink!, Lecture Notes in Computer Science, 2003, 2570, 185–207.