Maximising Lifetime for Fault-Tolerant Target Coverage in Sensor Networks[☆]

Thomas Erlebach[a,*], Tom Grant[a], Frank Kammer[b]

[a] *Department of Computer Science, University of Leicester, England*
[b] *Institut für Informatik, Universität Augsburg, Germany*

Abstract

We study the problem of maximising the lifetime of a sensor network for fault-tolerant target coverage in a setting with composite events. Here, a composite event is the simultaneous occurrence of a combination of atomic events, such as the detection of smoke and high temperature. We are given sensor nodes that have an initial battery level and can monitor certain event types, and a set of points at which composite events need to be detected. The points and sensor nodes are located in the Euclidean plane, and all nodes have the same sensing radius. The goal is to compute a longest activity schedule with the property that at any point in time, each event point is monitored by at least two active sensor nodes. We present a $(6 + \varepsilon)$-approximation algorithm for this problem by devising an approximation algorithm with the same ratio for the dual problem of minimising the weight of a fault-tolerant sensor cover. The algorithm generalises previous approximation algorithms for geometric set cover with weighted unit disks and is obtained by enumerating properties of the optimal solution that guide a dynamic programming approach.

Keywords: Approximation algorithm, unit disk graph, set multi-cover, dynamic programming

1. Introduction

Consider a sensor network whose task is to detect the occurrence of events at a given set of event points. This is also known as the *target coverage problem*. Since the nodes in a sensor network often have a limited battery supply that cannot be replenished, it is important to address the problem of maximising the lifetime of the network, i.e., the length of time during which the network can carry out its monitoring task successfully.

[☆] A preliminary version of this work was presented at the *23rd ACM Symposium on Parallelism in Algorithms and Architectures* (SPAA 2011).

[*] Corresponding author. Postal address: Department of Computer Science, University of Leicester, University Road, Leicester, LE1 7RH, UK. Phone: +44-116-2523411. Fax: +44-116-2523604. e-mail: te17@mcs.le.ac.uk

Email addresses: te17@mcs.le.ac.uk (Thomas Erlebach), tg53@mcs.le.ac.uk (Tom Grant), kammer@informatik.uni-augsburg.de (Frank Kammer)

The lifetime of the network can be prolonged by calculating an activity schedule in which only a subset of the sensor nodes is active at any point in time, and the remaining sensors are in a sleep mode that saves energy. The active nodes must be sufficient for performing the required monitoring task. Following [21, 18], we consider the setting where the events to be detected are *composite events*, i.e., events comprised of several simultaneous *atomic events* at the same location detected by different sensor types, and the sensor coverage is required to be *fault-tolerant*, i.e., the failure of any one sensor does not affect the sensing task.

An atomic event is a physical change in the environment such as temperature rising above a pre-defined threshold, and a composite event is a combination of several atomic events at the same location. As an example motivating the study of composite events, one can consider the scenario described by Vu et al. [21] where fire is detected by the composite event of temperature being above a given threshold and density of smoke being above a pre-defined value.

The settings described in [21, 18] also require that each event is covered by k sensors as a fault tolerance mechanism. One may wish to cover each event by k sensors such that if up to $k-1$ sensors fail, there will still be at least one active sensor that can detect and report the event. Obviously, the higher the value of k, the more robust the network will be against failing nodes. In this paper, we mainly consider the case of $k = 2$. This case is of interest in many application settings, because higher levels of fault tolerance are considered to consume too many resources. Moreover, to handle higher values of k is much more complicated.

We assume that the sensor nodes and event points are located in the Euclidean plane, and all sensor nodes have the same sensing radius. Each sensor node can monitor a certain set of event types, and the composite event to be detected at each event point is a combination of atomic events corresponding to different event types. We remark that the presence of multiple event types adds a non-geometric, combinatorial aspect to the problem.

A common approach to lifetime maximisation is to formulate the problem as a linear program and obtain an approximate solution by approximating the dual problem of computing a sensor cover of minimum weight (see Section 6 for details). We follow the same approach and hence mainly consider the dual problem of minimising the weight of a fault-tolerant sensor cover. We model the latter problem as a weighted multi-T-cover problem with unit disks, where T is the set of event types.

In the special case of atomic events of just one event type and no fault-tolerance requirements ($k = 1$), the minimum weight sensor cover problem is a standard geometric set cover problem where the aim is to cover a given set of points using unit disks of minimum total weight. This problem has been well studied. It is known to be $\mathcal{NP}$-hard even for the unweighted case [9], motivating the study of approximation algorithms. For the weighted case of geometric set cover with unit disks, the best known approximation ratio is $4+\varepsilon$ [12, 24]. Our setting poses the additional challenges of having to cover every point twice (turning the problem into a *multi-cover* problem) while avoiding the loss of a factor of two in the approximation ratio, and of dealing with different event types and composite events. Addressing these challenges requires us to refine the techniques that have been developed for the standard geometric set cover problem with unit disks.

1.1. Related Work

Sensor cover problems have been studied in several variants, including target coverage problems where a discrete set of points that need to be monitored is specified in the input, and region coverage problems where the area to be monitored is specified as a (typically convex) region in the plane. We refer to the survey by Thai et al. [20] for an overview. Some of the work on the lifetime maximisation version of sensor cover problems has focused on models where the sets of sensors that are active at different times must be disjoint [6], but it was pointed out in [3, 4, 7] that the use of non-disjoint sensor covers can yield significantly extended lifetime. Berman et al. [3, 4] show that the region coverage problem can be reduced to the target coverage problem and present an algorithm with logarithmic approximation ratio. They also show that a minimum cost sensor cover algorithm with approximation ratio ρ implies an approximation algorithm with ratio $\rho(1+\varepsilon)$ for the lifetime maximisation problem using the Garg-Könemann algorithm [15]. Dhawan et al. [11] study a target coverage problem where the sensor nodes can adjust their sensing range. They propose a greedy algorithm for the minimum cost sensor cover problem that yields a logarithmic approximation ratio. Zhao and Gurusamy [22] study the target coverage problem with the additional requirement that the sensors that are active at any time are connected. They obtain an algorithm with logarithmic approximation ratio and also present a performance evaluation based on simulation experiments. Sanders and Schieferdecker [19] show that the target coverage problem for sensors represented by unit disks with the objective of lifetime maximisation is $\mathcal{NP}$-hard. They also provide a $(1+\varepsilon)$-approximation algorithm using resource augmentation, i.e., their algorithm needs to increase the sensing range of every sensor node by a factor of $1+\delta$, for some fixed $\delta > 0$. Much of the previous work on target coverage problems has not considered fault-tolerance requirements, composite events or different sensor types. Vu et al. [21] and Marta et al. [18] consider fault-tolerant sensor cover problems with composite events. They present centralised and distributed heuristics and evaluate them in simulations. Contrary to their work, in this paper we aim at designing approximation algorithms with provable performance guarantees for fault-tolerant sensor cover problems with composite events.

A special case of the minimum cost sensor cover problem is the weighted geometric set cover problem with unit disks: Given a set of points and a set of weighted unit disks, compute a cheapest set of disks that covers all points. This problem has received considerable attention as it includes the weighted dominating set problem for unit disk graphs, which is relevant for routing backbone construction in wireless networks. This relationship also shows that the problem is $\mathcal{NP}$-hard, as the minimum dominating set problem for unit disk graphs is known to be $\mathcal{NP}$-hard [9]. The first constant factor approximation for weighted set cover with unit disks was a 72-approximation by Ambühl et al. [2]. The plane is partitioned into squares of constant size, and the algorithm computes a 2-approximation for each square separately and outputs the union of the solutions for all squares. As a subroutine, they show that the problem can be solved optimally in polynomial time by dynamic programming if the points to be covered are located in a strip and all disks have centres outside the strip. This inaugural constant approximation result was then improved to a $(6+\varepsilon)$-approximation by Huang et al. [17] by considering blocks made up of a bounded number of squares and applying the geometric shifting strategy [16]. They compute separate solutions for each horizontal and vertical strip of squares inside the block. They also introduce a *sandglass* technique

by which an algorithm 'guesses' properties of the optimal solution that allow it to decide which of the points in a square should be covered by disks with centre above or below the square, and which by disks with centre to the left or right of the square. The approach employed by Huang et al. [17] was amended by Dai and Yu [10], yielding a $(5 + \varepsilon)$-approximation. The improvement is obtained by calculating solutions over pairs of strips of squares simultaneously, combined with techniques from [17]. A further improvement to a $(4+\varepsilon)$-approximation was then attained by Erlebach and Mihalák [12] and, independently, by Zou et al. [24]. The main idea is to compute solutions for K adjacent strips of squares simultaneously using a 'split sweepline' technique that ensures that disks that cover points in different strips are met by the corresponding sweepline pieces at the same time.

All the results discussed in the previous paragraph apply to weighted geometric set cover with unit disks and thus also to the minimum-weight dominating set problem in unit disk graphs. For the latter problem, one is also interested in the connected variant, i.e., the minimum-weight connected dominating set problem. The approach followed in [2, 10, 12, 17, 24] is first to compute a cheap dominating set, and then to solve a node-weighted Steiner tree problem to connect that dominating set. The node-weighted Steiner tree problem admits a 2.5α-approximation algorithm in unit disk graphs [13, 23], where α is the approximation ratio of the best known approximation algorithm for edge-weighted Steiner trees, which is used as a subroutine. With the recent result by Byrka et al. [5], we have $\alpha < 1.39$ and thus an approximation algorithm with ratio less than 3.475 for node-weighted Steiner trees in unit disk graphs. Together with the $(4 + \varepsilon)$-approximation algorithm for minimum-weight dominating sets from [12, 24], this gives a 7.475-approximation algorithm for minimum-weight connected dominating sets in unit disk graphs.

The unweighted set multi-cover problem has been studied in geometric settings by Chekuri et al. [8]. They present an $O(\log \textsc{opt})$-approximation algorithm for set systems of bounded VC dimension, where $\textsc{opt}$ is the size of an optimal cover, and constant-factor approximation algorithms for covering points by half-spaces in three dimensions or for covering points with pseudo-disks in the Euclidean plane. Their results only apply to the unweighted case.

1.2. Our Results

We model the fault-tolerant target coverage problem with composite events as a generalised geometric multi-cover problem with unit disks and present a $(6+\varepsilon)$-approximation algorithm, both for the lifetime maximisation variant and for the minimum cost sensor cover variant of the problem. On a high level, we solve the minimum cost sensor cover problem by providing a 6-approximation algorithm for the case where all event points are located in a square of bounded size (which we refer to as *block*) and employing the geometric shifting strategy [16, 17]. To obtain the 6-approximation algorithm for a block, we 'guess' a number of properties of an optimal solution by enumeration, and then apply dynamic programming along horizontal and vertical strips of smaller squares. Because of the results of the 'guessing' step, we only need to handle the case where disks with centre outside a strip are used to cover points inside the strip, which makes a dynamic programming approach feasible. Our algorithm requires significant adaptations compared to previous work because the multi-cover aspect requires a more involved 'guessing' step and the algorithm also needs to handle different event types. Using our approximation

algorithm for minimum cost sensor cover as a subroutine in the Garg-Könemann algorithm [15], we obtain the same approximation ratio $6 + \varepsilon$ for the lifetime maximisation problem. Furthermore, provided that the communication radius of a sensor node is at least twice its sensing radius, we can use the known approximation algorithm for the node-weighted Steiner tree problem in unit disk graphs and obtain approximation ratio 9.475 for the problem variants where the sets of active sensors are required to form a connected communication network.

The remainder of the paper is structured as follows. In Section 2, we formally define the problems under consideration and briefly describe the aspects of our approach that are fairly standard, i.e., partitioning the plane into squares of bounded size and applying the geometric shifting strategy. In Section 3, we present a high-level description of our algorithm for approximating the minimum cost fault-tolerant sensor cover problem with composite events in a block. Sections 4 and 5 then present the details of two parts of the algorithm, namely the enumeration procedure that 'guesses' certain properties of an optimal solution and the dynamic programming approach that exploits these guesses and solves strip problems to optimality. In Section 6, we describe how the Garg-Könemann algorithm [15] can be applied, using a ρ-approximation algorithm for the minimum cost sensor cover problem as a subroutine, to give a $\rho(1 + \varepsilon)$-approximation algorithm for the lifetime maximisation variant of the problem. We show that this general approach that has already been employed in previous work applies to our setting as well. In Section 7, we show how our results can be adapted to the problem variants where sensor covers are required to form a connected communication graph. Finally, in Section 8, we conclude the paper and point to directions for future work.

2. Preliminaries

In this section, we introduce basic notation, formally define the problems considered in this paper, and describe the partitioning of the plane that is used by our algorithms.

Consider the two-dimensional Euclidean plane. The x-coordinate and y-coordinate of a point p is denoted by x_p and y_p, respectively. The Euclidean distance between two points p and q is denoted by $\delta(p, q)$. If d is a disk, we also use d to refer to the centre of d, so that we can write $\delta(d, p)$ for the Euclidean distance between the centre of d and a point p. Note that in this paper $\delta(d, p)$ does *not* denote the minimum distance between p and any point in d. We say that a point p is *in* a disk d of radius r if $\delta(d, q) \leq r$. We also say that p is *on* d if it lies on the boundary of the disk, i.e., $\delta(d, p) = r$. The power set of a set S, i.e., the set of all $2^{|S|}$ subsets of S, is denoted by $\mathcal{P}(S)$.

An algorithm for a maximisation problem is a ρ-approximation algorithm if it runs in polynomial time and always outputs a solution with objective value at least $\textsc{opt}/\rho$. Similarly, an algorithm for a minimisation problem is a ρ-approximation algorithm if it runs in polynomial time and always outputs a solution with objective value at most $\rho \cdot \textsc{opt}$. In both cases, $\textsc{opt}$ denotes the objective value of an optimal solution.

2.1. Problem Definitions

An instance of the *weighted (geometric) set cover problem with unit disks* is given by a set P of points in the two-dimensional Euclidean plane and a set D of weighted unit disks. All disks have the same radius r, and without loss of generality we assume

$r = 2$ throughout this paper (the choice of $r = 2$ is consistent with previous work, e.g., [12], where the dominating set problem for unit disks of radius 1 was transformed into an equivalent geometric set cover problem with disks of radius 2.). The weight of a disk $d \in D$ is non-negative and denoted by $w(d)$ or w_d. The total weight of a set $D' \subseteq D$ of disks is denoted by $w(D') = \sum_{d \in D'} w(d)$. The goal of the weighted set cover problem with unit disks is to select a set of disks of minimum total weight such that every point in P is in at least one of the selected disks.

In the context of the target coverage problem, the disks in D correspond to sensor nodes (with r representing the sensing radius) and the points in P correspond to targets (event points) that need to be monitored.

To model fault-tolerance requirements, we consider the multi-cover extension of the set cover problem: Every target $p \in P$ specifies a positive integer k_p as its coverage requirement, and a set $D' \subseteq D$ of disks is a *feasible multi-cover* if every $p \in P$ is in at least k_p distinct disks of D'. The goal of the *weighted multi-cover problem with unit disks* is to compute a feasible multi-cover D' of minimum total weight.

Furthermore, to model different event types, we assume that there is a (small) set T of different event types (e.g., smoke, temperature, etc.) and every sensor $d \in D$ has sensing components for a subset $T_d \subseteq T$ of event types. Moreover, each target $p \in P$ needs to be monitored with respect to a subset $T_p \subseteq T$ of event types, corresponding to the occurrence of a composite event comprised of atomic events for each type in T_p. A set $D' \subseteq D$ of disks is a *feasible multi-T-cover* if, for each p in P and each $t \in T_p$, p is in at least k_p distinct disks d' in D' with $t \in T_{d'}$, i.e., if

$$\forall p \in P : \forall t \in T_p : |\{d' \in D' : \delta(d', p) \leq r, t \in T_{d'}\}| \geq k_p \,.$$

To simplify matters, we reduce composite events to atomic events as follows: We replace every point p that needs to be monitored with respect to a set T_p of event types by $|T_p|$ copies of point p, each with the requirement to be monitored with respect to a distinct $t \in T_p$, and with the same coverage requirement k_p. For each copy p' of p, we denote by $t_{p'}$ its corresponding event type. It is easy to see that a set D' of disks is a feasible multi-T-cover of the original points with composite monitoring requirements if and only if it is a feasible multi-T-cover of the modified points with only atomic monitoring requirements. Therefore, without loss of generality, we can assume that every point $p \in P$ requires to be monitored only with respect to one event type t_p. Note that, after this transformation, P may contain points with identical coordinates. We can now say that a disk $d \in D$ *covers* a point $p \in P$ if p is in d and $t_p \in T_d$. A set $D' \subseteq D$ of disks *meets the coverage requirements* of a point $p \in P$ if p is covered by at least k_p distinct disks in D'.

We now formally define the weighted multi-cover problem under consideration.

Definition 1 (Weighted multi-T-cover problem (WMCUD-T)). For a set T of event types, the weighted multi-T-cover problem with unit disks is denoted by WMCUD-T. We are given a set D of disks of radius $r = 2$, each disk $d \in D$ associated with a non-negative weight w_d and a set T_d of event types, and a multi-set P of points, each point $p \in P$ associated with one event type $t_p \in T$ and a coverage requirement k_p. A subset $D' \subseteq D$ is a feasible multi-T-cover if:

$$\forall p \in P : |\{d' \in D' : \delta(d', p) \leq r, t_p \in T_{d'}\}| \geq k_p$$

The objective is to compute a feasible multi-T-cover of minimum total weight. The restriction of WMCUD-T to the case where $k_p \leq 2$ for all $p \in P$ is denoted by W2CUD-T.

For most of this paper, we will consider the restricted problem W2CUD-T. Furthermore, we note that the assumption that the cardinality of T is small (i.e., bounded by a fixed constant) is natural in the application setting. It is also necessary because if T is allowed to be arbitrarily large, even covering a single target requires us to solve a general set cover problem, and constant-factor approximation is ruled out unless $\mathcal{P} = \mathcal{NP}$ [14, 1]. The running-time of our algorithms is exponential in $|T|$.

Finally, let us define the lifetime maximisation problem.

Definition 2 (Maximum lifetime multi-T-cover problem (MLMCUD-T)). We are given points and disks as in an instance of WMCUD-T, but additionally each disk $d \in D$ specifies an initial battery level b_d, expressed in suitable units so that b_d is the total duration during which d can be active before its battery runs out. A schedule is a set of pairs (D_i, x_i), where $D_i \subseteq D$ is a feasible multi-T-cover and $x_i \geq 0$. A schedule is feasible if for each $d \in D$, the sum of the x_i values of all pairs (D_i, x_i) with $d \in D_i$ does not exceed b_d. The lifetime of a schedule is the sum of the x_i values of all its pairs (D_i, x_i). The goal is to compute a feasible schedule of maximum lifetime. We refer to this problem as the maximum lifetime multi-T-cover problem with unit disks (MLMCUD-T), and the restricted version where $k_p \leq 2$ for all $p \in P$ as ML2CUD-T.

2.2. Plane Partition

As in previous work (e.g., [17]), our algorithms employ a partition of the plane. Imagine an infinite grid that partitions the plane into squares of side length 1.4 (any number sufficiently close to, but strictly less than, $\sqrt{2}$ would do). Consider an arbitrary such square S. Note that any disk of radius 2 with centre in S contains the whole square. We can assume without loss of generality that no point or disk centre lies exactly on the boundary between two adjacent squares. The neighbouring infinite regions of a square S are referenced as in Figure 1, with UL standing for 'upper left,' CR for 'centre right,' LM for 'lower middle', etc. Furthermore, let UPPER be the union of the regions UL, UM, UL, let LOWER be the union of LL, LM, LR, let LEFT be the union of UL, CL, LL, and let RIGHT be the union of UR, CR, LR.

For an integer constant $K > 0$ (which determines the ε term in the final approximation ratio), consider a partition of the plane into *blocks* so that each block B consists of $K \times K$ squares S. For ease of presentation, we index the squares in a block from $S_{0,0}$ in the top left corner to $S_{K-1,K-1}$ in the bottom right corner, i.e., the indices of a square S_{ij} are local to the block containing it. The first index i refers to the row and the second index j to the column in which S_{ij} is located within block B. Let $P_{ij} \subseteq P$ be the set of points from P that lie in S_{ij}. If we have a ρ-approximation for W2CUD-T instances whose points lie in one block, we can obtain a $\rho(1 + O(1/K))$-approximation for general instances of W2CUD-T using the standard geometric shifting strategy [16]. A sketch of this approach is as follows. The algorithm calculates a ρ-approximate solution for each block and amalgamates these solutions for individual blocks into a solution for the complete plane. Each block is then shifted up and right by four squares, and a new solution is calculated for this new set of blocks. This is repeated $K/4$ times. The best of

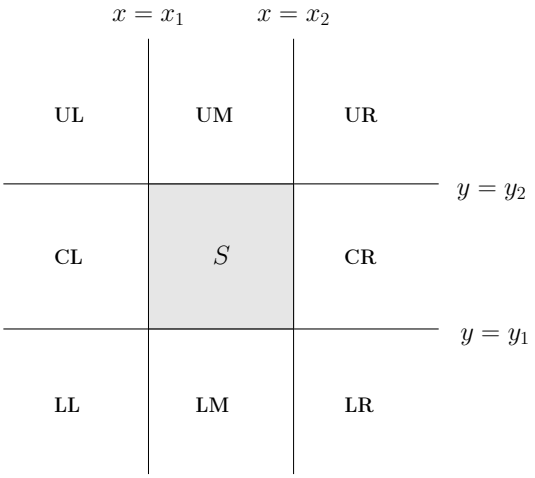


Figure 1: Square S and neighbouring regions

the $K/4$ solutions is then output as the overall solution. The analysis of this approach is based on the observation that a disk can cover points in different blocks for only two of the $K/4$ shifted cases, because each disk overlaps at most four horizontal or vertical strips of squares. For any given optimal solution OPT, there is a shifted position for which the total weight of disks in OPT that overlap block boundaries is at most $8w(\text{OPT})/K$, and thus the union of ρ-approximate solutions in the blocks is within a factor of $\rho(1+O(1/K))$ of the total weight of the optimal solution. The details of this are fairly standard and can be found for a similar setting, e.g., in [17]. Thus, the key to obtaining a good approximation algorithm for W2CUD-T is to achieve a good approximation ratio for instances where the points are located in one block.

3. A $(6+\varepsilon)$-Approximation Algorithm for W2CUD-T in a Block

Let an instance of W2CUD-T in a $K \times K$ block B be given by a set of points P_B (each point associated with an event type t_p and a coverage requirement $k_p \leq 2$) and a set D of disks. Our approach to solve this instance of W2CUD-T consists of two stages: In the first stage, using enumeration we 'guess' properties of a fixed optimal solution, denoted by OPT_B. (Here and in the following, any reference to 'the optimal solution' refers to that fixed optimal solution.) In the second stage, we approximate the best solution with these properties using dynamic programming.

As motivation for our approach to guessing properties of an optimal solution, observe that an optimal solution may contain an arbitrarily large number of disks with centre in the same square. As an example, consider the case two horizontally adjacent squares S_1 and S_2. Assume that there is a single event type and no fault-tolerance requirements. Place n targets on a vertical line ℓ_1 in square S_1. Place the centres of n disks on a vertical line ℓ_2 in square S_2 so that ℓ_2 has distance 2 from ℓ_1. Furthermore, let the centre of each disk be at the same y-coordinate as one of the n targets. Then each of the n disks covers exactly one of the n targets, and the only solution that covers all targets consists

of all n disks. This shows that an approach that enumerates all disks with centre in a given square that are in the optimal solution is not feasible. We overcome this obstacle by enumerating guesses only for certain properties of the optimal solution.

The enumeration stage produces a polynomial number of guesses of the properties of OPT_B. Each such guess π specifies a set of disks D_π that are contained in OPT_B, which may reduce the coverage requirements of some points in P_B accordingly. For example, the coverage requirement of a point p with $k_p = 2$ that is contained in one disk $d \in D_\pi$ with $t_p \in T_d$ is reduced to $k'_p = 1$. Furthermore, each point p whose remaining coverage requirement k'_p is not zero is classified according to the regions $\text{UL}, \text{UM}, \ldots$ (with respect to the square in which p is contained) where the centres of k'_p disks that cover p in the optimal solution lie.

Definition 3 (Classified coverage requirement). The symbol $\Updownarrow$ specifies for a point p that p must be covered by a disk with centre in $\text{UPPER} \cup \text{LOWER}$, the symbol $\updownarrow$ that the point is covered by a disk with centre in $\text{UM} \cup \text{LM}$, the symbol $\Leftrightarrow$ that the point is covered by a disk with centre in $\text{LEFT} \cup \text{RIGHT}$, and the symbol $\leftrightarrow$ that the point is covered by a disk with centre in $\text{CL} \cup \text{CR}$. If a point p has remaining coverage requirement $k'_p = 1$, the *classified remaining coverage requirement* π_p can be $\{\Leftrightarrow\}$ or $\{\Updownarrow\}$. If a point has remaining coverage requirement $k'_p = 2$, the *classified remaining coverage requirement* π_p can be $\{\Leftrightarrow, \Leftrightarrow\}$, $\{\Updownarrow, \Updownarrow\}$, $\{\Leftrightarrow, \updownarrow\}$, or $\{\leftrightarrow, \Updownarrow\}$.

Definition 4. A set D' of disks meets the classified coverage requirement $\pi_p = \{\Leftrightarrow, \Leftrightarrow\}$ of a point p if the point p is covered by two distinct disks $d_1, d_2 \in D'$ with centre in $\text{LEFT} \cup \text{RIGHT}$. A set D' of disks meets the classified coverage requirement $\pi_p = \{\Leftrightarrow, \updownarrow\}$ of a point p if the point p is covered by one disk $d_1 \in D'$ with centre in $\text{LEFT} \cup \text{RIGHT}$ and by one disk $d_2 \in D'$ with centre in $\text{UM} \cup \text{LM}$. For the other possible classified coverage requirements, the conditions are analogous.

Definition 5. A guess π is *consistent* with the optimal solution OPT_B if $D_\pi \subseteq \text{OPT}_B$ and $\text{OPT}_B \setminus D_\pi$ meets the classified coverage requirement π_p of every point $p \in P$.

The important property of the enumeration stage of our algorithm is stated in the following lemma, whose proof is deferred to Section 4.

Lemma 1. *There is a polynomial-time algorithm that enumerates in polynomial time a set of guesses such that at least one of the guesses is consistent with* OPT_B.

We remark that if a point p is covered in OPT_B by one or two disks whose centres lie in the square in which p lies, then our enumeration stage will ensure that the required disks covering p will be contained in D_π for the guess π that is consistent with the optimal solution. Therefore, it suffices to consider the case that the remaining coverage requirement of each point p needs to be satisfied by disks with centres outside the square in which p lies.

For each guess π, we want to find a solution that is consistent with π and has small weight.

Lemma 2. *There is a polynomial-time algorithm that takes as input a guess* π *and either outputs a solution that is consistent with* π*, or asserts that no solution consistent with* π *exists. If* π *is consistent with the optimal solution* OPT_B*, the solution output by the algorithm has weight at most* $6 \cdot w(\text{OPT}_B)$.

The algorithm of Lemma 2 is obtained by applying dynamic programming to each horizontal and each vertical strip of squares contained in the block B. For this, we first define two strip problems.

Definition 6 (Horizontal strip problem). Consider a horizontal strip H of squares, consisting of K squares S_{ij} for fixed i and $0 \leq j \leq K-1$, in a block. We are given a set P_H of points in the strip, and a set $D_{\bar{H}}$ of disks with centre above or below the strip (i.e., all the disks have centres in the union of the regions UPPER $\cup$ LOWER for all squares S_{ij} in the strip H). Each disk $d \in D_{\bar{H}}$ is associated with a weight $w(d)$ and a set T_d of event types. Each point $p \in P_H$ has an event type t_p and a classified coverage requirement π_p that can be $\{\Updownarrow\}$, $\{\updownarrow\}$, or $\{\Updownarrow, \Updownarrow\}$. A set $D' \subseteq D_{\bar{H}}$ is a feasible solution if it meets the classified coverage requirements of all points in P_H. The goal of the *horizontal strip problem* is to compute a feasible solution of minimum weight.

The *vertical strip problem* is defined analogously.

In Section 5, we prove the following result.

Lemma 3. *There is a polynomial-time algorithm that computes a minimum-weight solution to any (horizontal or vertical) strip problem (or asserts that no feasible solution exists).*

Using this lemma, we are now ready to prove Lemma 2.

PROOF (OF LEMMA 2). Let π be an arbitrary guess. It is easy to check whether π admits a feasible solution, because it suffices to check whether D (the set of all disks) meets the classified coverage requirements of all points in P_B. Therefore, we can assume in the following that there is a solution that is consistent with π.

The algorithm solves a vertical strip problem for each of the K vertical strips of K squares contained in block B, and a horizontal strip problem for each of the K horizontal strips of K squares contained in block B. The inputs to the $2K$ strip problems are constructed as follows. For each horizontal strip H, the set of disks in the input to the horizontal strip problem for H consists of all disks from $D \setminus D_\pi$ with centre outside H. Similarly, for each vertical strip V, the set of disks in the input to the vertical strip problem for V consists of all disks from $D \setminus D_\pi$ with centre outside V. The points that form the input of a strip problem are determined as follows. Consider a point $p \in P_B$ that lies in some square S_{ij} that belongs to a horizontal strip H_p and a vertical strip V_p. If $\pi_p = \{\Leftrightarrow\}$, then p is added with classified coverage requirement $\{\Leftrightarrow\}$ to the vertical strip problem for V_p. If $\pi_p = \{\Updownarrow\}$, then p is added with classified coverage requirement $\{\Updownarrow\}$ to the horizontal strip problem for H_p. If $\pi_p = \{\Updownarrow, \Updownarrow\}$, then p is added with classified coverage requirement $\{\Updownarrow, \Updownarrow\}$ to the horizontal strip problem for H_p. If $\pi_p = \{\Leftrightarrow, \Leftrightarrow\}$, then p is added with classified coverage requirement $\{\Leftrightarrow, \Leftrightarrow\}$ to the vertical strip problem for V_p. If $\pi_p = \{\Leftrightarrow, \updownarrow\}$, then p is added with classified coverage requirement $\{\Leftrightarrow\}$ to the vertical strip problem for V_p and with classified coverage requirement $\{\updownarrow\}$ to the horizontal strip problem for H_p. If $\pi_p = \{\leftrightarrow, \Updownarrow\}$, then p is added with classified coverage requirement $\{\leftrightarrow\}$ to the vertical strip problem for V_p and with classified coverage requirement $\{\Updownarrow\}$ to the horizontal strip problem for H_p.

The algorithm of Lemma 3 is applied to each of the $2K$ strip problems. If one of the strip problems does not admit a feasible solution, the algorithm outputs that the

given guess π does not admit a feasible solution. If all $2K$ strip problems admit feasible solutions, then the union of the $2K$ solutions, together with the set of disks D_π that has been determined to be in the solution by the guess, is then output as the solution. It is easy to see that the algorithm outputs a feasible solution if one exists for the given guess π, and otherwise asserts correctly that no feasible solution exists.

Now consider a guess π that is consistent with OPT_B. Let $\text{OPT}_B(H)$ be the subset of $\text{OPT}_B \setminus D_\pi$ consisting of disks that are in UPPER $\cup$ LOWER for a horizontal strip H and overlap H, and let $\text{OPT}_B(V)$ be the subset of $\text{OPT}_B \setminus D_\pi$ consisting of disks that are in LEFT $\cup$ RIGHT for a horizontal strip V and overlap V. We observe that $\text{OPT}_B(H)$ and $\text{OPT}_B(V)$ form feasible solutions to the strip problems for strips H and V, respectively. Hence, the solutions $A(H)$ and $A(V)$ computed by the algorithm from Lemma 3 for strips H and V have costs at most $w(\text{OPT}_B(H))$ and $w(\text{OPT}_B(V))$, respectively. The cost of the solution output by the algorithm is therefore at most

$$\begin{aligned} w(D_\pi) + \sum_H w(A_H) + \sum_V w(A_V) &\leq w(D_\pi) + \sum_H w(\text{OPT}_B(H)) + \sum_V w(\text{OPT}_B(V)) \\ &\leq 6 \cdot w(\text{OPT}_B)\,. \end{aligned}$$

The last inequality follows as each disk d in $\text{OPT}_B \setminus D_\pi$ is in UPPER $\cup$ LOWER for at most three horizontal strips and in LEFT $\cup$ RIGHT for at most three vertical strips. For vertical strips, this is because a disk of diameter 4 intersects at most four (consecutive) vertical strips of width 1.4, and in one of them the disk has centre inside the strip and therefore cannot be in LEFT $\cup$ RIGHT. For horizontal strips the reasoning is analogous. Finally, note that disks in D_π are counted only once. □

By combining Lemmas 1 and 2, we obtain the following lemma.

Lemma 4. *There is a* 6*-approximation algorithm for instances of* W2CUD-T *where all points lie in one* $K \times K$ *block.*

As discussed in Subsection 2.2, a 6-approximation algorithm for W2CUD-T in a block implies a $(6+\varepsilon)$-approximation algorithm for general instances of W2CUD-T. Thus, Lemma 4 implies the following result.

Theorem 1. *For every fixed* $\varepsilon > 0$*, there is a* $(6+\varepsilon)$*-approximation algorithm for the problem* W2CUD-T.

4. Guessing Properties of the Optimal Solution by Enumeration

In this section we prove Lemma 1. For the following, fix an arbitrary optimal solution to the given instance of W2CUD-T in a block B, and denote it by OPT_B. We present the enumeration technique using the notion of 'guessing.' When we write that the algorithm 'guesses' a property of OPT_B, this means that the algorithm enumerates all possibilities for that property in such a way that one of the possibilities is guaranteed to be the desired property of the optimal solution. The enumeration is done separately for each of the K^2 squares S_{ij} contained in B. Consider one such square S_{ij}. Let m denote the number of given disks that overlap S_{ij}. Let the left and right boundary of S_{ij} lie on the line $x = x_1$ and $x = x_2$, respectively, and the bottom and top boundary on the line $y = y_1$ and $y = y_2$, respectively. See again Figure 1 for an illustration.

4.1. Disks with Centre in S_{ij}

First, for every event type t_l in T, we guess whether OPT_B contains 0, 1 or at least 2 disks with centre in S_{ij} that monitor event type t_l. Furthermore, in the second and third case we also guess the one disk or two of those disks, respectively. Note that a disk with centre in S_{ij} must contain the whole square S_{ij}, as the disk has radius $r = 2$ and the square has side length less than $\sqrt{2}$. Furthermore, two disks with centre in S_{ij} that cover event type t_l are sufficient to meet the coverage requirements of all points $p \in P_{ij}$ with $t_p = t_l$, hence it is not necessary to guess more than two such disks. All the disks that are guessed in this way form the set D_π, and the coverage requirements of all points that are covered by disks in D_π are reduced accordingly. In the following, we use k_p to denote the remaining coverage requirement of a point p.

4.2. Overview of Square Partition

Next, we aim at guessing a partition of the square S_{ij} into areas such that for the points in the same area, we know whether they are covered twice by $\text{UPPER} \cup \text{LOWER}$, twice by $\text{LEFT} \cup \text{RIGHT}$, or once by $\text{UPPER} \cup \text{LOWER}$ (possibly restricted to $\text{UM} \cup \text{LM}$) and once by $\text{LEFT} \cup \text{RIGHT}$ (possibly restricted to $\text{CL} \cup \text{CR}$). In other words, we want to guess the classified coverage requirements of all points. We guess a separate such square partition for each event type $t_l \in T$. For each $t_l \in T$, the steps involved in guessing the partition are as follows. First, we determine areas called *2-watching t_l sandglasses* (the terminology is motivated by a similar sandglass concept in [17]) for the four regions UM, LM, CL and CR. For each of these areas, we can require that points are covered only by $\text{UPPER} \cup \text{LOWER}$ (classified coverage requirement $\{\Updownarrow\}$ if $k_p = 1$ or $\{\Updownarrow, \Updownarrow\}$ if $k_p = 2$) or only by $\text{LEFT} \cup \text{RIGHT}$ (classified coverage requirement $\{\Leftrightarrow\}$ if $k_p = 1$ or $\{\Leftrightarrow, \Leftrightarrow\}$ if $k_p = 2$). Second, we consider 1-watching envelopes, i.e., envelopes of the disks in OPT_B with centre in one of the regions UM, LM, CL and CR, and guess the four points where adjacent envelopes intersect. Based on the locations of these intersection points, we can segment the square into smaller areas and deduce for each of the smaller areas whether the points located in the area are to be watched from $\text{UPPER} \cup \text{LOWER}$ or from $\text{LEFT} \cup \text{RIGHT}$ (or from both). The details of the partition of the square into areas for one specific event type t_l are presented in the following subsections.

4.3. 2-Watching Sandglasses

We define the 2-watching t_l sandglass for the region LM. The sandglasses for the regions UM, CR, and CL are defined similarly (by rotation).

Let P'_{ij} be the set of points in P_{ij} that have event type t_l, i.e., the set of points p with $t_p = t_l$. Consider the set $P^2_{\text{LM}} \subseteq P'_{ij}$ of all points in P'_{ij} that are covered by two distinct disks from LM in OPT_B, but not covered by any disk from OPT_B that does not have centre in LM. For each $p \in P^2_{\text{LM}}$, consider the line l_p through p with slope 1 and let p' be the point where l_p intersects the line $y = y_1$. Let p_l be the point in P^2_{LM} for which p' is leftmost. Similarly, let l'_p be the line through p with slope -1 and let p'' be the intersection point of l'_p and $y = y_1$. Let p_r be the point in P^2_{LM} for which p'' is rightmost. The 2-watching t_l sandglass for LM is now defined as the area that is obtained as the intersection of the halfplane below l_{p_l}, the halfplane below l'_{p_r}, and the square S_{ij}. See

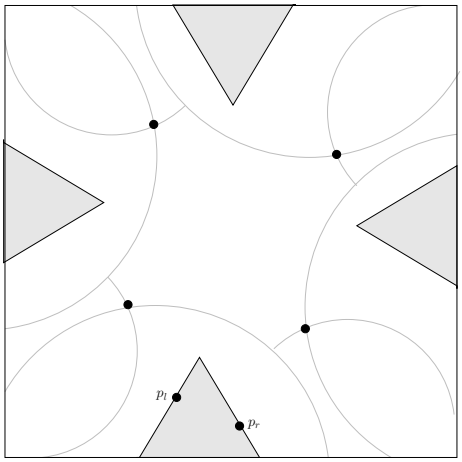


Figure 2: 2-watching sandglasses

Figure 2 for an illustration.[1] Note that this sandglass is uniquely determined by p_l and p_r and there are $O(|P'_{ij}|^2)$ possibilities for guessing p_l and p_r.

We show that the coverage requirements of any point of P'_{ij} located in the 2-watching t_l sandglass for LM are met by disks from UPPER $\cup$ LOWER. Define the *lower-shadow* of a point p to be the region that is the intersection of the halfplane below the line with slope 1 through p, the halfplane below the line with slope -1 through p, and the square S_{ij}. The left-shadow, up-shadow, and right-shadow of a point are defined analogously. We state the following lemma due to Huang et al. [17].

Lemma 5. [17] *If a point $p \in P'_{ij}$ is covered by a disk d from* LM*, then any point in the lower-shadow of p is also covered by the same disk from* LM*.*

The lemma directly implies the following corollary.

Corollary 1. *If a point $p \in P'_{ij}$ is covered by two disks d_1, d_2 from* LM*, then any point in the lower-shadow of p is also covered by the same two disks d_1, d_2 from* LM*.*

We also require the following lemma, which we prove by application of Lemma 5.

Lemma 6. *The coverage requirement for any point $p \in P'_{ij}$ that lies inside the* 2*-watching t_l sandglass for* LM *is met by disks in* OPT_B *from* UPPER $\cup$ LOWER*.*

PROOF. Consider the points p_l and p_r defining the 2-watching t_l sandglass for LM. For points in the lower-shadow of p_l or in the lower-shadow of p_r, the lemma follows from Corollary 1. Let p be a point that lies in the 2-watching t_l sandglass for LM, but not in the shadows of p_l or p_r. Assume that p is covered by a disk d with centre in CL or CR by OPT_B. Then, by Lemma 5 applied to the left-shadow or right-shadow of p, respectively, we find that p_l or p_r is also covered by d, a contradiction to the choice of p_l and p_r. $\square$

[1] Figure 2 and subsequent figures are not drawn to scale as they serve only illustrative purposes.

It follows that the coverage requirements of all points in the 2-watching t_l sandglasses for LM and for UM are satisfied by disks in OPT_B from UPPER $\cup$ LOWER, and the coverage requirements of all points in the 2-watching t_l sandglasses for CL and for CR are satisfied by disks in OPT_B from LEFT $\cup$ RIGHT. Hence, all the points from P'_{ij} that lie in 2-watching t_l sandglasses can be classified accordingly (classified coverage requirements $\{\Leftrightarrow\}$, $\{\Leftrightarrow, \Leftrightarrow\}$, $\{\Updownarrow\}$, or $\{\Updownarrow, \Updownarrow\}$). These points are ignored for the classifications of points described in the following subsections, i.e., their classification is not changed if they are contained in one of the areas under consideration there.

4.4. 1-Watching Envelopes

It remains to deal with points from P'_{ij} that do not lie in any of the 2-watching t_l sandglasses. For each of the four regions UM, LM, CL and CR, we define a 1-watching t_l envelope as follows (see Figure 3 for an illustration). Note that we have separate envelopes for each $t_l \in T$.

Definition 7 (1-watching t_l envelope). Consider a square S_{ij}, and let R be one of the regions UM, LM, CL and CR with respect to S_{ij}. The 1-*watching* t_l *envelope for region* R is the intersection of the square S and the boundary of the union of the disks in OPT_B that monitor event type t_l and have centre in region R. The respective boundary of the square (i.e., the top boundary for the 1-watching t_l envelope for UM, the right boundary for the 1-watching t_l envelope for CR, etc.) is used to fill in parts of the envelope where no disk from the respective region intersects the square.

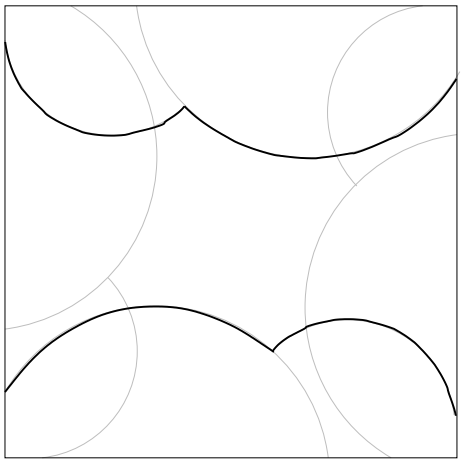


Figure 3: 1-watching envelopes for LM and UM

4.5. Intersection Points of 1-Envelopes

We call the 1-watching envelopes of CL and UM *adjacent*, and similarly those of UM and CR, etc. Allow us to define *intersection points* of adjacent 1-watching envelopes and describe how they are used to partition the remainder of square S_{ij} into areas such that we can specify for the points in each area whether they are covered by OPT_B using disks in UPPER $\cup$ LOWER or in LEFT $\cup$ RIGHT.

Lemma 7. *Each pair of adjacent* 1*-watching* t_l *envelopes intersects in exactly one point.*

PROOF. By the dimension of the square S_{ij} and the radius of the disks, it follows that the direction of any tangent to the 1-watching t_l envelope for UM or LM is in the open interval $(-\pi/4, \pi/4)$, and the direction of any tangent to the 1-watching t_l envelope for CL or CR is in the open interval $(\pi/4, 3\pi/4)$. Therefore, it is impossible that two adjacent 1-watching t_l envelopes have more than one intersection point. As each envelope is a curve connecting points on opposite sides of the square, it follows that two adjacent envelopes always intersect. □

Hence, there are four (not necessarily distinct) intersection points between adjacent 1-watching t_l envelopes. Note that each of these intersection points is uniquely specified by the two disks whose boundaries intersect at that point. Hence, it suffices to guess eight disks in order to determine the four intersection points.

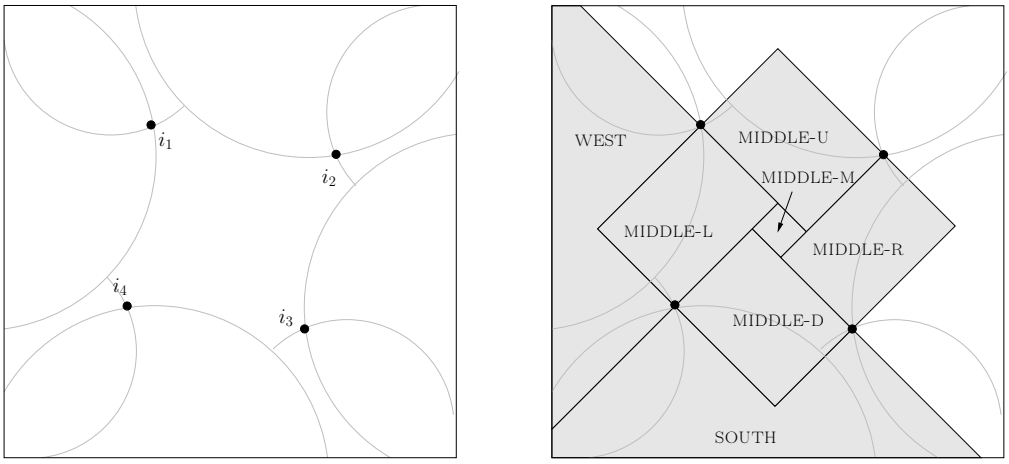
Let the intersection point of the 1-watching t_l envelopes for CL and UM be denoted by $i^{\text{UM}}_{\text{CL}}$. Similarly, let $i^{\text{UM}}_{\text{CR}}$ be the intersection point of the 1-watching t_l envelopes of UM and CR, $i^{\text{LM}}_{\text{CR}}$ the intersection point of the 1-watching t_l envelopes for CR and LM, and $i^{\text{LM}}_{\text{CL}}$ the intersection point of the 1-watching t_l envelopes for LM and CL.

4.6. Areas in a Square

Assume that $i^{\text{UM}}_{\text{CL}}$ is to the left of $i^{\text{UM}}_{\text{CR}}$ (i.e., has smaller x-coordinate), $i^{\text{UM}}_{\text{CR}}$ is above $i^{\text{LM}}_{\text{CR}}$ (i.e., has larger y-coordinate), $i^{\text{LM}}_{\text{CR}}$ is to the right of $i^{\text{LM}}_{\text{CL}}$, and $i^{\text{LM}}_{\text{CL}}$ is below $i^{\text{UM}}_{\text{CL}}$. We call this the *standard configuration*, and we will deal with other alternative configurations in subsequent sections. For the standard configuration, define i_1 to be $i^{\text{UM}}_{\text{CL}}$, i_2 to be $i^{\text{UM}}_{\text{CR}}$, i_3 to be $i^{\text{LM}}_{\text{CR}}$, and i_4 to be $i^{\text{LM}}_{\text{CL}}$, as shown in Figure 4a. (For the alternative configurations to be discussed later, the correspondence between $i_1, \ldots, i_4$ and the intersection points will be different.) For an arbitrary intersection point i_s ($s \in \{1, 2, 3, 4\}$), let l_s and l'_s be the two lines through i_s with slope 1 and -1, respectively. These lines allow us to define the following areas for which we can make further deductions regarding the disks watching points in these areas.

Define MIDDLE to be the area that is the intersection of the halfplanes below l_1 and l'_2 and the halfplanes above l_3 and l'_4. As illustrated in Figure 4b, let MIDDLE-L be the area that is the intersection of the halfplanes below l_1 and l'_1 and the halfplanes above l_4 and l'_4, and similarly let MIDDLE-R be the area that is the intersection of the halfplanes below l_2 and l'_2 and the halfplanes above l_3 and l'_3.

We claim that the coverage requirements of all points in the areas MIDDLE-L and MIDDLE-R are met in OPT_B by disks with centre in LEFT ∪ RIGHT. We state the arguments for points within the area MIDDLE-L, and identical arguments apply to MIDDLE-R. Observe that the area MIDDLE-L lies entirely below the 1-watching t_l envelope for UM. This is because MIDDLE-L is contained in the 90 degree cone below i_1 that lies between l_1 and l'_1, while the 1-watching t_l envelope for UM lies in the union of the halfplanes above l_1 and above l'_1. Similarly, MIDDLE-L lies entirely above the 1-watching t_l envelope for LM. Therefore, it is not possible that a point in MIDDLE-L is covered by a disk with centre in UM or LM in OPT_B, and so the coverage requirements of all points in MIDDLE-L are indeed met in OPT_B by disks from LEFT ∪ RIGHT. Thus, we can classify the points from P'_{ij} that lie in MIDDLE-L and MIDDLE-R accordingly (i.e., such a point p is assigned classified coverage requirement $\{\Leftrightarrow\}$ if $k_p = 1$ and $\{\Leftrightarrow, \Leftrightarrow\}$ if $k_p = 2$).



(a) Intersection points (b) Areas

Figure 4: Intersection points of 1-watching envelopes, and resulting middle and peripheral areas

The areas MIDDLE-U and MIDDLE-D are defined analogously to MIDDLE-L and MIDDLE-R, with MIDDLE-U enclosed by lines l_1, l_1', l_2, l_2' and MIDDLE-D enclosed by lines l_3, l_3', l_4, l_4', see Figure 4b. Furthermore, the region obtained by removing MIDDLE-L, MIDDLE-U, MIDDLE-R and MIDDLE-D from MIDDLE (which may be empty) is denoted by MIDDLE-M. By arguments analogous to those given above, the coverage requirements of points in areas MIDDLE-U and MIDDLE-D are met in OPT_B by disks from UPPER $\cup$ LOWER. Hence, the points in these areas can be classified accordingly (i.e., such a point p is assigned classified coverage requirement $\{\Updownarrow\}$ if $k_p = 1$ and $\{\Updownarrow, \Updownarrow\}$ if $k_p = 2$).

Finally, the area MIDDLE-M lies outside all four 1-watching t_l envelopes, and so the points in that area can only be covered in OPT_B by disks from regions UL, UR, LL or LR. These regions are in UPPER $\cup$ LOWER and in LEFT $\cup$ RIGHT, so we can (arbitrarily giving preference to the former) classify these points as points that have to be covered by disks in UPPER $\cup$ LOWER (i.e., such a point p is assigned classified coverage requirement $\{\Updownarrow\}$ if $k_p = 1$ and $\{\Updownarrow, \Updownarrow\}$ if $k_p = 2$).

We refer to the part of S_{ij} that is not in MIDDLE as the *peripheral part*. Consider the peripheral area SOUTH shown in Figure 4b. This is the area that is the union of the lower-shadow of i_3 and the lower-shadow of i_4. Recall that points in SOUTH that are also in a 2-watching t_l sandglass have already been classified and are not considered further. For any remaining point p located in SOUTH we know that p is covered by a disk from LM (as SOUTH lies below the 1-watching t_l envelope for LM). Furthermore, if p has to be covered by a second disk, we know that p is covered by at least one disk whose centre is not in LM because otherwise p would lie in the 2-watching t_l sandglass for LM and would have been classified already. That second disk covering p cannot have centre in UM, because SOUTH lies entirely below the 1-watching t_l envelope for UM. Hence, if $k_p = 1$, we specify that p must be covered by UPPER $\cup$ LOWER (classified coverage requirement $\{\Updownarrow\}$), and if $k_p = 2$, we specify that p must be covered once by LEFT $\cup$ RIGHT and once by LM $\cup$ UM (classified coverage requirement $\{\Leftrightarrow, \updownarrow\}$).

The areas WEST, NORTH and EAST are defined and handled analogously.

As each point in P'_{ij} is contained in one of the areas defined above, all these points are classified, i.e, we have determined for each point p a classified coverage requirement π_p.

4.7. Areas in a Square – Other Configurations

Recall that $i^{\text{UM}}_{\text{CL}}$ is the intersection point between the 1-watching t_l envelopes for UM and CL, and $i^{\text{LM}}_{\text{CL}}$, $i^{\text{LM}}_{\text{CR}}$ and $i^{\text{UM}}_{\text{CR}}$ are defined analogously. In the previous subsection, we assumed the standard configuration as shown in Figure 4a, i.e., $i^{\text{UM}}_{\text{CL}}$ is to the left of $i^{\text{UM}}_{\text{CR}}$, $i^{\text{UM}}_{\text{CR}}$ above $i^{\text{LM}}_{\text{CR}}$, $i^{\text{UM}}_{\text{CL}}$ above $i^{\text{LM}}_{\text{CL}}$, and $i^{\text{LM}}_{\text{CL}}$ to the left of $i^{\text{LM}}_{\text{CR}}$. In the following, we discuss how to handle all other possible configurations.

The definition of 2-watching t_l sandglasses does not depend on the configuration of intersection points, so 2-watching sandglasses are handled as before. The definition of the peripheral areas is adapted as follows. The area SOUTH is the union of the lower-shadow of the lower of the two points $i^{\text{UM}}_{\text{CL}}$, $i^{\text{LM}}_{\text{CL}}$ and the lower-shadow of the lower of the two points $i^{\text{UM}}_{\text{CR}}$, $i^{\text{LM}}_{\text{CR}}$. The area WEST is the union of the left-shadow of the point that is further left among the two points $i^{\text{UM}}_{\text{CL}}$, $i^{\text{UM}}_{\text{CR}}$ and the left-shadow of the point that is further left among the two points $i^{\text{LM}}_{\text{CL}}$, $i^{\text{LM}}_{\text{CR}}$. The definitions of EAST and NORTH are analogous. We observe that each of the four peripheral areas can be handled in the same way as before. For example, we still have that every point in SOUTH is covered by a disk with centre in LM, and if the point has to be covered by a second disk, there is a disk covering it with centre in LEFT $\cup$ RIGHT.

Let us introduce some terminology. We say that the 1-watching t_l envelopes for CL and CR are *opposite*, and so are the envelopes for UM and LM. Furthermore, we say that the 1-watching t_l envelopes for CL and CR *overlap* if $i^{\text{UM}}_{\text{CL}}$ is to the right of $i^{\text{UM}}_{\text{CR}}$ and $i^{\text{LM}}_{\text{CL}}$ is to the right of $i^{\text{LM}}_{\text{CR}}$. In other words, the relations between $i^{\text{UM}}_{\text{CL}}$ and $i^{\text{UM}}_{\text{CR}}$ and between $i^{\text{LM}}_{\text{CL}}$ and $i^{\text{LM}}_{\text{CR}}$ are both reversed compared to the standard configuration. Similarly, we say that the 1-watching t_l envelopes for CL and CR *cross* if only one of the two relations is reversed compared to the standard configuration. For the 1-watching t_l envelopes for UM and LM, the notions of overlapping and crossing are defined analogously by considering the relations between $i^{\text{UM}}_{\text{CL}}$ and $i^{\text{LM}}_{\text{CL}}$ and between $i^{\text{UM}}_{\text{CR}}$ and $i^{\text{LM}}_{\text{CR}}$. Additionally, let in the following the areas MIDDLE, MIDDLE-L, etc. be defined in terms of the respective choices for i_1, i_2, i_3, i_4 as in Subsection 4.6.

4.7.1. A Pair of Opposite Envelopes Overlap.

The first alternative configuration that we consider is the case where at least one pair of opposite envelopes overlap. Without loss of generality, assume that the 1-watching t_l envelopes for UM and LM overlap. (The other case is symmetric.) Then $i^{\text{LM}}_{\text{CL}}$ is above $i^{\text{UM}}_{\text{CL}}$, and $i^{\text{LM}}_{\text{CR}}$ is above $i^{\text{UM}}_{\text{CR}}$. In each of the following cases, the choice of i_1, i_2, i_3, i_4 will ensure that the areas MIDDLE-M, MIDDLE-L, and MIDDLE-R are enclosed within the 1-watching t_l envelopes for both UM and LM, i.e., the coverage requirement for points in these areas can be classified as $\{\Updownarrow\}$ or $\{\Updownarrow, \Updownarrow\}$.

Case 1: *The Other Pair of Envelopes are Standard.*

In this case, the envelope for CL and the envelope for CR are standard, i.e., they neither overlap nor cross. The situation is sketched in Figure 5a. Let $i_1 = i^{\text{LM}}_{\text{CL}}$, $i_2 = i^{\text{LM}}_{\text{CR}}$, $i_3 = i^{\text{UM}}_{\text{CR}}$, and $i_4 = i^{\text{UM}}_{\text{CL}}$. The areas MIDDLE-U and MIDDLE-D are always outside the 1-watching t_l

envelopes for CL and CR, and therefore—as in the standard configuration—the points in these areas can be classified by the coverage requirement $\{\Updownarrow\}$ or $\{\Updownarrow, \Updownarrow\}$.

Case 2: *The Other Pair of Envelopes also Overlap.*

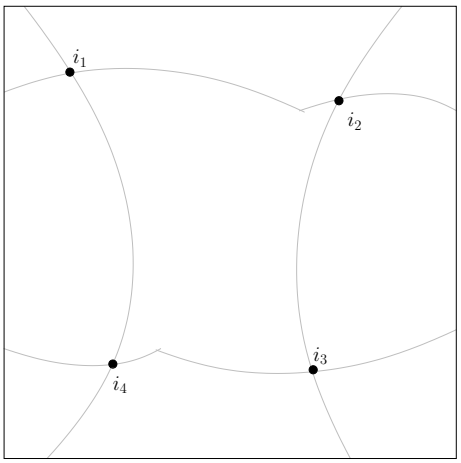
As illustrated in Figure 5b, we have that the 1-watching t_l envelopes for CL and CR overlap, and the 1-watching t_l envelopes for UM and LM overlap. Let $i_1 = i_{\text{CR}}^{\text{LM}}$, $i_2 = i_{\text{CL}}^{\text{LM}}$, $i_3 = i_{\text{CL}}^{\text{UM}}$, $i_4 = i_{\text{CR}}^{\text{UM}}$. The points within the areas MIDDLE-U and MIDDLE-D are covered twice by disks in LEFT $\cup$ RIGHT as those areas are enclosed within the 1-watching t_l envelopes for both CL and CR. These points can be classified by the coverage requirement $\{\Leftrightarrow\}$ or $\{\Leftrightarrow, \Leftrightarrow\}$.

Case 3: *The Other Pair of Envelopes Cross.*

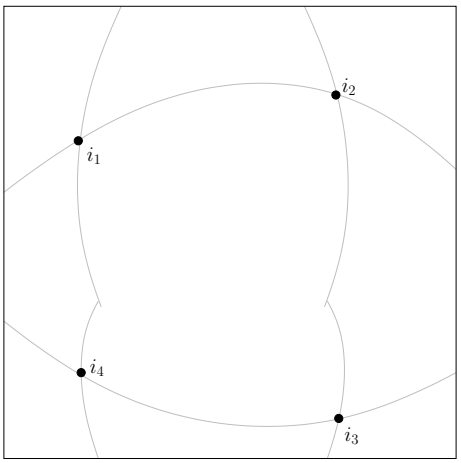
Without loss of generality, assume that $i_{\text{CL}}^{\text{UM}}$ is to the right of $i_{\text{CR}}^{\text{UM}}$ and $i_{\text{CL}}^{\text{LM}}$ is to the left of $i_{\text{CR}}^{\text{LM}}$, as illustrated in Figure 6a. Let $i_1 = i_{\text{CL}}^{\text{LM}}$, $i_2 = i_{\text{CR}}^{\text{LM}}$, $i_3 = i_{\text{CL}}^{\text{UM}}$, $i_4 = i_{\text{CR}}^{\text{UM}}$. The points in MIDDLE-U can be handeled as in Case 1 and the points in MIDDLE-D as in Case 2.

4.7.2. One Pair of Opposite Envelopes Cross and the Other Pair of Envelopes Are Standard.

Without loss of generality, assume that the 1-watching t_l envelopes for CL and CR cross, and that $i_{\text{CL}}^{\text{UM}}$ is to the left of $i_{\text{CR}}^{\text{UM}}$ and $i_{\text{CL}}^{\text{LM}}$ is to the right of $i_{\text{CR}}^{\text{LM}}$, as shown in Figure 6b. (The other cases are symmetric.) Let $i_1 = i_{\text{CL}}^{\text{UM}}$, $i_2 = i_{\text{CR}}^{\text{UM}}$, $i_3 = i_{\text{CL}}^{\text{LM}}$, and $i_4 = i_{\text{CR}}^{\text{LM}}$. As in the standard configuration, the areas MIDDLE-L, MIDDLE-R, and MIDDLE-M are outside the 1-watching t_l envelopes for UM and LM, i.e., we can classify the coverage requirements for the points in these areas as $\{\Leftrightarrow\}$ or $\{\Leftrightarrow, \Leftrightarrow\}$. For a similar reason, the coverage requirements for the points in MIDDLE-U can be classified as $\{\Updownarrow\}$ or $\{\Updownarrow, \Updownarrow\}$. The area MIDDLE-D is within the 1-watching envelopes for both CL and CR, so the coverage requirements for points in MIDDLE-D can be classified as $\{\Leftrightarrow\}$ or $\{\Leftrightarrow, \Leftrightarrow\}$.

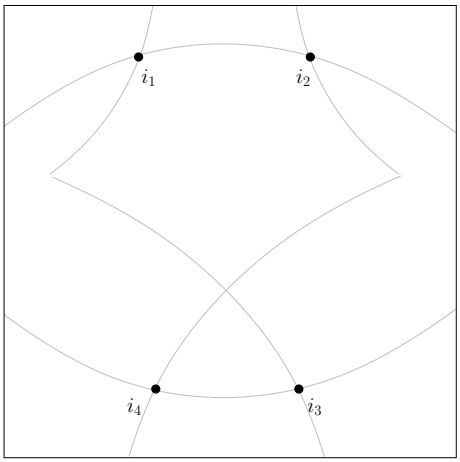


(a) The UM envelope and the LM envelope overlap

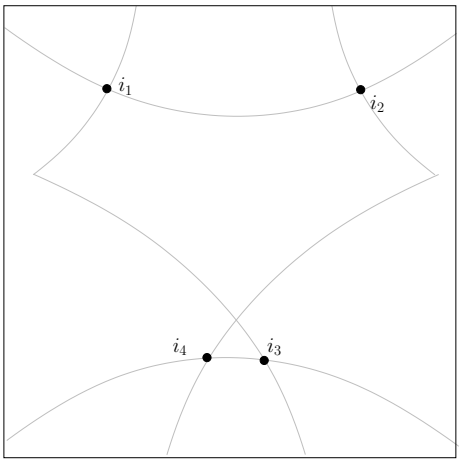


(b) UM envelope overlaps LM envelope, and CL envelope overlaps CR envelope

Figure 5: Configurations with overlapping envelopes



(a) UM envelope overlaps LM envelope, CR and CL envelopes cross



(b) CL envelope crosses CR envelope

Figure 6: Configurations with crossing envelopes

4.7.3. Both Pairs of Opposite Envelopes Cross.

The only configuration that has not yet been considered is the case where both pairs of opposite envelopes cross. One such configuration would have $i_{\text{CL}}^{\text{UM}}$ strictly to the the left of $i_{\text{CR}}^{\text{UM}}$, $i_{\text{CL}}^{\text{LM}}$ strictly to the right of $i_{\text{CR}}^{\text{LM}}$, $i_{\text{CL}}^{\text{UM}}$ strictly above $i_{\text{CL}}^{\text{LM}}$, and $i_{\text{CR}}^{\text{UM}}$ strictly below $i_{\text{CR}}^{\text{LM}}$. (We can assume strict relationships since if two points coincide, we are free to choose the relation and arrive at a previously considered configuration where it is not the case that both pairs of envelopes cross.) To show that this is impossible, consider the line l of slope -1 through $i_{\text{CL}}^{\text{UM}}$. Since $i_{\text{CL}}^{\text{UM}}$ is above $i_{\text{CL}}^{\text{LM}}$ and both points lie on the CL envelope, we get that $i_{\text{CL}}^{\text{LM}}$ is strictly below the line l. Since $i_{\text{CL}}^{\text{UM}}$ is on the line l and $i_{\text{CR}}^{\text{UM}}$ is to the right of $i_{\text{CL}}^{\text{UM}}$ and also lies on the UM envelope, we get that $i_{\text{CR}}^{\text{UM}}$ is above l. Since $i_{\text{CR}}^{\text{UM}}$ is below $i_{\text{CR}}^{\text{LM}}$ and both points are on the CR envelope, it follows that $i_{\text{CR}}^{\text{LM}}$ is above l. Since $i_{\text{CL}}^{\text{LM}}$ is to the right of $i_{\text{CR}}^{\text{LM}}$ and both points are on the LM envelope, we obtain that $i_{\text{CL}}^{\text{LM}}$ is strictly above the line l. This is a contradiction to the conclusion that $i_{\text{CL}}^{\text{LM}}$ is strictly below l that we derived above. Hence, this configuration is not possible. The other configurations where both envelopes cross can be excluded similarly.

4.8. Complexity of Enumeration

For each of the K^2 squares S_{ij} in a block, and for each event type t_l in T, we enumerate up to two disks with centre in S_{ij} (these form the set D_π of disks that are determined to be in the solution by the guessing stage), eight points defining the 2-watching sandglasses, and four intersection points (each identified by two disks) of the 1-watching envelopes. Hence, if there are m disks and n points in total, there are $O((m^2n^8m^8)^{|T|}) = (m+n)^{O(|T|)}$ choices per square, and thus $(m+n)^{O(K^2|T|)}$ choices for the whole block B. Since K and $|T|$ are constants, this is a polynomial number of choices. This concludes the proof of Lemma 1.

5. Dynamic Programming Algorithm

In this section we prove Lemma **??** by providing a dynamic programming algorithm for the strip problems. Vertical strip problems can be solved in the same way as horizontal strip problems (just roate the plane by 90 degrees), so we consider only horizontal strip problems in this section.

5.1. Input to the Dynamic Program

Consider a horizontal strip H of squares, consisting of K squares S_{ij} for fixed i and $0 \leq j \leq K-1$ in a block. We are given a set P_H of points in the strip, and a set $D_{\bar{H}}$ of disks with centre above or below the strip (i.e., all the disks have centres in the union of the regions UPPER $\cup$ LOWER for all squares S_{ij} in the strip H). Each disk $d \in D_{\bar{H}}$ is associated with a weight $w(d)$ and a set T_d of event types. Each point $p \in P$ has an event type t_p and a classified coverage requirement π_p (cf. Definition 3). We let n_H denote the number of points in P_H.

For ease of presentation, we extend the strip H by two empty squares on the left side and also on the right side. This ensures that every disk in $D_{\bar{H}}$ that covers some point in P_H has its centre in the region UM or in the region LM with respect to some square S of H.

As it is easy to detect infeasible instances, we only consider the case that there is a feasible solution, i.e., a set of disks $D' \subseteq D_{\bar{H}}$ that meets the classified coverage requirements of all points in P_H. The goal is now to compute a feasible solution of minimum weight for the given instance of the horizontal strip problem.

Let the points in $P_H = \{p_1, p_2, \ldots, p_{n_H}\}$ be ordered by non-decreasing x-coordinates, i.e., for every $1 < i \leq n_H$ we have $x_{p_{i-1}} \leq x_{p_i}$. If two points have the same x-coordinate then we order them arbitrarily.

5.2. Outer and Inner Envelopes

For a square S in H, let UM(S) denote the region UM with respect to S, and let LM(S) denote the region LM with respect to S. Let $T' \in \mathcal{P}(T) \setminus \{\emptyset\}$ be an arbitrary non-empty combination of event types in T. We define outer and inner envelopes formed by those disks in a solution that have centres in UM(S) (or LM(S)) and monitor the event types in T'. The purpose of these envelopes is to represent the disks lying in a particular region that cover some points from P_H, in the sense that any point in P_H that is covered once or twice by disks from that region is also covered at least the same number of times by disks that are part of the two envelopes of that region. Our dynamic programming algorithm then aims at computing envelopes corresponding to a solution of minimum cost.

Definition 8 (Outer T' envelope for UM(S)). Let S be a square in the horizontal strip H, let $T' \in \mathcal{P}(T) \setminus \{\emptyset\}$, and let D be a set of disks. The set of disks d in D that have centre in UM(S) and satisfy $T_d = T'$ is denoted by $D^{T'}_{\text{UM}(S)}$. The *outer T' envelope for* UM(S) (for a set of disks D) is the intersection of the boundary of the union of all disks in $D^{T'}_{\text{UM}(S)}$ with the strip H. (If at some x-coordinate there is no disk from $D^{T'}_{\text{UM}(S)}$ that overlaps H, we let the upper boundary of H form a part of the envelope.) A disk is said to be *on* the envelope if the boundary of the disk forms a part of the envelope that consists of more than a single point. The set of disks on the envelope is denoted by $\bar{D}^{T'}_{\text{UM}(S)}$.

We also require inner envelopes, defined as follows.

Definition 9 (Inner T' envelope for $\textsc{um}(S)$). The *inner T' envelope for* $\textsc{um}(S)$ (for a set of disks D) is defined to be the outer T' envelope for $\textsc{um}(S)$ for the set of disks $D \setminus \bar{D}^{T'}_{\textsc{um}(S)}$. The set of disks on the inner T' envelope for $\textsc{um}(S)$ is denoted by $\bar{D}^{T'}_{I(\textsc{um}(S))}$.

The outer and inner T' envelopes for $\textsc{um}(S)$ in some fixed optimal solution are denoted by $\textsc{opt}^{T'}_{\textsc{um}(S)}$ and $\textsc{opt}^{T'}_{I(\textsc{um}(S))}$, respectively.

The outer and inner T' envelopes for $\textsc{lm}(S)$, for a set of disks D, are defined and denoted analogously (by considering disks with centre in $\textsc{lm}(S)$ instead of $\textsc{um}(S)$, and replacing $\textsc{um}(S)$ by $\textsc{lm}(S)$ in the notation).

For a given set D of disks, these definitions give us four different envelopes for each square S and for each set T' of event types. We view the set of disks on each of these envelopes as ordered by non-decreasing x-coordinates of the centres of the disks. (We can assume without loss of generality that no two disks on an envelope have the same centre.)

Note that each disk from D is on at most one envelope. Furthermore, if we trace an envelope from left to right, the disks of the envelope appear on the envelope in the order of increasing x-coordinates of their centres, and each disk appears on an envelope at most once (since all disks have the same radius.)

5.3. Dynamic Programming Table

We have a table W_{p_i} for every point $p_i \in P_H$. Let S be the square in which p_i lies. Let S^- and S^+ be the adjacent squares to the left and right of S, respectively. Let S^{--} be the square to the left of S^-, and S^{++} be the square to the right of S^+. Let $\mathcal{S}(p_i) = \{S^{--}, S^-, S, S^+, S^{++}\}$. Note that all disks from $D_{\bar{H}}$ that overlap S must be in $\textsc{um}(U)$ or $\textsc{lm}(U)$ for some square U in $\mathcal{S}(p_i)$, because the disks have radius 2 and the squares have side length 1.4.

For every $T' \in \mathcal{P}(T) \setminus \{\emptyset\}$ we have the following indexes for the table W_{p_i}: For the outer T' envelope for each of the ten regions in $\{\textsc{um}(U), \textsc{lm}(U) \mid U \in \mathcal{S}(p_i)\}$, we have a set of up to three disks that are candidates for the disk d that is on the outer T' envelope for that region at position $x = x_{p_i}$, for the disk just before d on that envelope, and for the disk just after d on that envelope. For the inner T' envelope of each of the ten regions, we have one disk that is a candidate for being the disk on that envelope at position $x = x_{p_i}$. Hence, an entry of the table W_{p_i} is indexed by $40 \cdot (2^{|T|} - 1)$ disks (three disks for each of the ten outer envelopes, and one disk for each of the ten inner envelopes, for each choice of T'). For ease of presentation, we write the indexes for the table W_{p_i} as two sets of disks D_{U} and D_{L}, where D_{U} contains all the disks from inner and outer T' envelopes for any T' and regions $\textsc{um}(U)$ for $U \in \mathcal{S}(p_i)$, and D_{L} contains all the disks from inner and outer T' envelopes for any T' and regions $\textsc{lm}(U)$ for $U \in \mathcal{S}(p_i)$.

Consider the case that the indexes for the table W_{p_i} for each T' are chosen as the disks that actually form the envelopes under consideration in an optimal solution to the horizontal strip problem, i.e., the indexes contain the corresponding disks on the envelopes $\textsc{opt}^{T'}_{\textsc{um}(S)}$, $\textsc{opt}^{T'}_{I(\textsc{um}(S))}$, $\textsc{opt}^{T'}_{\textsc{um}(S^+)}$, etc. We observe that, if the classified coverage requirement of p_i is met by the optimal solution, then it is also met by the disks constituting the indexes for the table. To illustrate this, assume that p_i is covered in the optimal solution by two disks d_1^* and d_2^* from $\textsc{um}(S)$. If $T_{d_1^*} = T_{d_2^*}$, then for $T' = T_{d_1^*}$

we must have that p_i is covered once by the disk d_1 that forms the outer T' envelope for $\text{UM}(S)$ at $x = x_{p_i}$, and a second time by the disk before or after d_1 on the same envelope, or by the disk on the inner T' envelope for $\text{UM}(S)$ at $x = x_{p_i}$. In the other cases, the reasoning is similar.

Furthermore, we note that the disks $D_{\text{U}} \cup D_{\text{L}}$ specified as indexes for a table entry $W_{p_i}(D_{\text{U}}, D_{\text{L}})$ completely separate the solution to the right of the line $x = p_i$ from the solution to the left of the line $x = p_i$. In other words, if D' and D'' are different solutions for which the disks $D_{\text{U}} \cup D_{\text{L}}$ are the disks on the 20 envelopes relevant to p_i for all $T' \subseteq T$, then the left part of D' (disks of D' that appear before $D_{\text{U}} \cup D_{\text{L}}$ on their respective envelopes) can be combined with $D_{\text{U}} \cup D_{\text{L}}$ and the right part of D'' (disks of D'' that appear after $D_{\text{U}} \cup D_{\text{L}}$ on their respective envelopes) to form a new feasible solution. Furthermore, a disk d cannot simultaneously be in the left part of D' or D'' and also in the right part of D' or D'' (because d appears either before or after a disk in $D_{\text{U}} \cup D_{\text{L}}$ on its respective envelope).

The value of an entry of table W_{p_i} is set to infinity if the disks indexing the table entry do not meet the classified coverage requirement for p_i, and otherwise the minimum cost of a set of disks that includes all the disks indexing the table entry of W_{p_i} and that also meets the classified coverage requirements of all points preceding p_i in P_H. Once all the tables W_{p_i} have been computed, the set of disks corresponding to the minimum value of any entry of table W_p for the last point $p = p_{n_H}$ is output as the solution.

5.4. Computing the Table Entries

Let the table entries for the leftmost point $p_1 \in P_H$ be initialised as follows. For every choice of indexes D_{U} and D_{L}, the table entry $W_{p_1}(D_{\text{U}}, D_{\text{L}})$ is set to $w(D_{\text{U}}) + w(D_{\text{L}})$ if $D_{\text{U}} \cup D_{\text{L}}$ meets the coverage requirement of point p_1, and to ∞ otherwise. For subsequent points $p_i \in P_H$, the value of an entry $W_{p_i}(D_{\text{U}}, D_{\text{L}})$ such that $D_{\text{U}} \cup D_{\text{L}}$ meets the coverage requirement of p_i is calculated as the cheapest cost that can be obtained in the following way: Take the cost of a set of disks covering all points up to p_{i-1} from a table entry $W_{p_{i-1}}(D'_{\text{U}}, D'_{\text{L}})$ for some D'_{U} and D'_{L}, and add the cost of all disks that are in $D_{\text{U}} \cup D_{\text{L}}$ but not in $D'_{\text{U}} \cup D'_{\text{L}}$. This means that for every $p_i \in P_H \setminus \{p_1\}$ we calculate the table entry for each combination of disks on the envelopes as follows:

$$W_{p_i}(D_{\text{U}}, D_{\text{L}}) = \begin{cases} \infty, & \text{if } D_{\text{U}} \cup D_{\text{L}} \text{ does not meet the classified coverage requirement for } p_i \\ \min_{D'_{\text{U}}, D'_{\text{L}}} \{W_{p_{i-1}}(D'_{\text{U}}, D'_{\text{L}}) + w(D_{\text{U}} - D'_{\text{U}}) + w(D_{\text{L}} - D'_{\text{L}})\}, & \text{otherwise} \end{cases} \quad (1)$$

Consider the last point $p_{n_H} \in P_H$. The minimum value in the table $W_{p_{n_H}}$ is the cost of the minimum weight solution that covers point p_{n_H} and all other points that preceded it in the ordered set P_H. If we remember for each $W_{p_i}(D_{\text{U}}, D_{\text{L}})$ the choice of D'_{U} and D'_{L} that attained the minimum in (1), standard traceback techniques can then be used to attain a set of disks that forms a feasible solution and has cost equal to that table entry.

Lemma 8. *The dynamic program computes an optimal solution to the horizontal strip problem in polynomial time.*

PROOF. As we assume that the size of T is bounded by a constant, the number of disks required to index a table entry is bounded by a constant as well. The tables W_{p_i} are of polynomial size, and the computation of each table entry can be carried out in polynomial time. Thus, the overall running time is polynomial.

Assume that $W_{p_{n_H}}(D_{\text{U}}, D_{\text{L}}) = v$ is the minimum value in table $W_{p_{n_H}}$, and that this value is not ∞. Then the set of disks output by the algorithm has weight v, as the weight of every disk added to the solution is accounted for in (1). Furthermore, the solution is feasible, as the corresponding entry in each table W_{p_i} is not ∞, and therefore the classified coverage requirement of each point p_i is met.

It remains to show that, if the optimal solution to the strip problem has weight v^*, then the algorithm outputs a solution of weight at most v^*. For a point p_i in P_H, let $\mathcal{S}(p_i)$ be defined as in Subsection 5.3 and let $\text{OPT}_{\text{U}}(p_i)$ be the disks corresponding to indexes of table W_{p_i} (i.e., three disks from each outer envelope and one disk from each inner envelope) that are on the outer and inner T' envelopes for $\text{UM}(U)$ for $U \in \mathcal{S}(p_i)$ for the fixed optimal solution. Let $\text{OPT}_{\text{L}}(p_i)$ be defined analogously based on the disks on the outer and inner T' envelopes for $\text{LM}(U)$ for $U \in \mathcal{S}(p_i)$ for the optimal solution. Let $v^*(p_i)$ be the cost of all disks in the optimal solution that are not to the right of any of the disks in $\text{OPT}_{\text{U}}(p_i) \cup \text{OPT}_{\text{L}}(p_i)$ on their respective envelopes.

We claim that

$$W_{p_i}(\text{OPT}_{\text{U}}(p_i), \text{OPT}_{\text{L}}(p_i)) \leq v^*(p_i)$$

holds for all $p_i \in P_H$.

We prove the claim by induction. For p_1, we have

$$W_{p_1}(\text{OPT}_{\text{U}}(p_1), \text{OPT}_{\text{L}}(p_1)) = w(\text{OPT}_{\text{U}}(p_1) \cup \text{OPT}_{\text{L}}(p_1)) = v^*(p_1)\,,$$

proving the claim for $i = 1$. For $i > 1$, we have by (1) that

$$\begin{aligned} W_{p_i}(\text{OPT}_{\text{U}}(p_i), \text{OPT}_{\text{L}}(p_i)) \quad \leq \quad & W_{p_{i-1}}(\text{OPT}_{\text{U}}(p_{i-1}), \text{OPT}_{\text{L}}(p_{i-1})) \\ & + w(\text{OPT}_{\text{U}}(p_i) \setminus \text{OPT}_{\text{U}}(p_{i-1})) \\ & + w(\text{OPT}_{\text{L}}(p_i) \setminus \text{OPT}_{\text{L}}(p_{i-1})). \end{aligned}$$

By induction, we know that $W_{p_{i-1}}(\text{OPT}_{\text{U}}(p_{i-1}), \text{OPT}_{\text{L}}(p_{i-1})) \leq v^*(p_{i-1})$. Furthermore, the weight $v^*(p_i) - v^*(p_{i-1})$ includes all disks that are in $\text{OPT}_{\text{U}}(p_i) \cup \text{OPT}_{\text{L}}(p_i)$ but not in the part of the optimal solution corresponding to $v^*(p_{i-1})$. One can observe that this includes all disks contained in $\text{OPT}_{\text{U}}(p_i) \setminus \text{OPT}_{\text{U}}(p_{i-1})$ and in $\text{OPT}_{\text{L}}(p_i) \setminus \text{OPT}_{\text{L}}(p_{i-1})$. This is because the disks in $\text{OPT}_{\text{U}}(p_i) \setminus \text{OPT}_{\text{U}}(p_{i-1})$ and in $\text{OPT}_{\text{L}}(p_i) \setminus \text{OPT}_{\text{L}}(p_{i-1})$ appear on their respective envelopes to the right of the disks in $\text{OPT}_{\text{U}}(p_{i-1}) \cup \text{OPT}_{\text{L}}(p_{i-1})$ and thus are not contained in the part of the optimal solution corresponding to $v^*(p_{i-1})$. Hence, we have that

$$w(\text{OPT}_{\text{U}}(p_i) \setminus \text{OPT}_{\text{U}}(p_{i-1})) + w(\text{OPT}_{\text{L}}(p_i) \setminus \text{OPT}_{\text{L}}(p_{i-1})) \leq v^*(p_i) - v^*(p_{i-1}),$$

and the claim is established. It follows that $W_{p_{n_H}}(\text{OPT}_{\text{U}}(p_{n_H}), \text{OPT}_{\text{L}}(p_{n_H})) \leq v^*$, and the algorithm outputs a feasible solution of cost at most v^* (the cost must actually be equal to v^* as the algorithm outputs a feasible solution and v^* is the optimal cost). □

6. Lifetime Maximisation

In this section we describe a known general method [3, 4, 11] for approximating the maximum lifetime problem by using an approximation algorithm for the minimum weight sensor cover problem, and its application to our setting. A linear program Π of the form

$$\{\max c^{\mathrm{T}} x \mid Ax \leq b, x \geq 0\}, \tag{2}$$

where A, b and c are non-negative, is called a *packing problem*. The linear program may be given implicitly and the number of variables x_j may be exponential. For a given vector w, the problem of finding a column j of A such that $\sum_i A_{i,j} w_i / c_j$ is minimised is called the problem of *computing a column of minimum length* with respect to Π. It is known [3] that, if a packing problem Π' admits a ρ-approximation algorithm for the problem of computing a column of minimum length with respect to Π' for any given vector w, then the algorithm by Garg and Könemann [15] can be used to compute an $(1+\varepsilon)\rho$-approximate solution to Π'.

The natural linear programming formulation of the problem of maximising the lifetime of a sensor network is as follows:

$$\max \sum_{D' \in \mathcal{D}} x_{D'} \tag{3}$$

$$\text{s.t.} \sum_{D' \in \mathcal{D}: d \in D'} x_{D'} \leq b_d, \quad \text{for all } d \in D \tag{4}$$

$$x_{D'} \geq 0, \quad \text{for all } D' \in \mathcal{D} \tag{5}$$

Here, D is the set of sensor nodes and b_d is the initial battery level of node d, specified in such a way that the battery level is sufficient for d to be active for b_d units of time. The set $\mathcal{D}$ contains all possible sensor covers, i.e., all subsets of D that satisfy the required sensor coverage constraint. The variable $x_{D'}$ represents the length of the part of the schedule during which the nodes of the sensor cover $D' \in \mathcal{D}$ are active. The objective (3) is to maximise the total length of the schedule. The constraints (4) specify that a node d can take part in sensor covers for a total amount of time that is bounded by b_d. The linear program (3)–(5) does not have polynomial size, as the number of variables $x_{D'}$ can be exponential. However, it is a packing problem, and the algorithm by Garg and Könemann [15] can be applied. The problem of computing a column of minimum length is simply the problem of computing a set $D' \in \mathcal{D}$ of minimum cost, where the cost of a node $d \in D$ is given by some weight w_d.

Theorem 2. [3] *Let $\mathcal{D}$ be the set of valid sensor covers. If there is a ρ-approximation algorithm for the problem of computing a set $D' \in \mathcal{D}$ of minimum cost, for any given vertex weights w_d, then for every fixed $\varepsilon > 0$ there is a $\rho(1+\varepsilon)$-approximation algorithm for the lifetime maximisation problem.*

We remark that Theorem 2 applies to a wide variety of lifetime maximisation problems, because the specification of the set $\mathcal{D}$ of valid sensor covers can be arbitrary. For example, if a graph class admits a ρ-approximation algorithm for the weighted connected dominating set problem, then that graph class admits a $(\rho+\varepsilon)$-approximation algorithm for the lifetime maximisation problem where at any point of time the active nodes must form a connected dominating set.

Theorem 2 along with Theorem 1 in Section 5 implies the following corollary.

Corollary 2. *For every fixed $\varepsilon > 0$, there is a $(6+\varepsilon)$-approximation algorithm for the problem* ML2CUD-T.

7. Connected Sensor Cover

Up to now we have considered only the condition that the selected sensors meet the coverage requirement of each point in P. In many applications, such as the settings described in [18, 21], it is additionally required that the selected sensors form a connected network. For this, it is assumed that each sensor node is equipped with a wireless radio that allows it to transmit messages to any other node that is located within a certain communication radius r_c from it. (This corresponds to a communication graph where the sensor nodes are represented by disks of radius $r_c/2$ and two nodes are adjacent if their disks intersect.) It is natural to expect that r_c is larger than the sensing radius r. Under the assumption that $r_c \geq 2r$, we can extend our approximation algorithms for W2CUD-T and ML2CUD-T to the problem variants with connectivity requirement.

For W2CUD-T with connectivity requirement, we first compute a $(6+\varepsilon)$-approximate solution D' to the problem without the connectivity requirement, by using the algorithm from Theorem 1. Then, viewing the given disks as disks of radius $r_c/2$, we solve the minimum node-weighted Steiner tree problem for the disks in D' as terminals, using the algorithm with approximation ratio less than 3.475 for node-weighted Steiner trees in unit disk graphs [13, 23, 5]. Let S be the set of Steiner nodes output by the algorithm. The set $D' \cup S$ is then output as a solution to W2CUD-T with connectivity requirement. Let $\textsc{opt}_c$ be an optimal solution to W2CUD-T with connectivity requirement. Observe that $\textsc{opt}_c$ is a (superset of a) feasible solution to the Steiner tree problem considered above: $\textsc{opt}_c$ is connected and contains disks covering every point in P. Every disk in D' covers a point in P, and hence the centre of any disk in D' is within distance $r + r \leq r_c$ of the centre of some disk in $\textsc{opt}_c$. Consequently, $\textsc{opt}_c \cup D'$ is connected. This shows that the Steiner tree approximation algorithm produces a set S of cost less than 3.475 times the cost of $\textsc{opt}_c$. As the cost of D' is within a factor of $6 + \varepsilon$ of the optimal solution to W2CUD-T without connectivity requirement, and thus within the same factor of the cost of $\textsc{opt}_c$, the overall approximation ratio is bounded by 9.475 if ε is chosen sufficiently small.

Theorem 3. *For the variants of* W2CUD-T *and* ML2CUD-T *where the active disks need to be connected and $r_c \geq 2r$, there is a* 9.475*-approximation algorithm.*

8. Conclusion

We have presented a $(6+\varepsilon)$-approximation algorithm for the target coverage problem with composite events and fault-tolerance requirements, both for the lifetime maximisation variant and for the problem of covering all event points by sensors of minimum total cost. Our approach is based on guessing properties of the optimal solution (by enumeration) and then using these properties to guide a dynamic programming algorithm. This is a generalisation of the approach employed by Huang et al. [17] to obtain a $(6 + \varepsilon)$-approximation for weighted set cover with unit disks. For the latter problem, subsequent work has improved the approximation ratio to $5 + \varepsilon$ [10] and then to $4 + \varepsilon$ [12, 24]. The main idea of these improvements is to perform the dynamic programming

in several strips simultaneously. One possible direction for future work would be to see whether these improvements can also be adapted to the fault-tolerant target coverage problem with composite events.

Another question of interest is whether our approach can be adapted to arbitrary coverage requirements k_p, i.e., without the restriction $k_p \leq 2$ for all $p \in P$. Repeated application of our $(6+\varepsilon)$-approximation algorithm would incur an extra factor of $k/2$ in the approximation ratio, where $k = \max_p k_p$, which is not desirable.

For W2CUD-T and also for the special case of weighted geometric set cover with unit disks, it is an interesting open question whether a polynomial-time approximation scheme (PTAS) can be obtained. So far, no hardness-of-approximation results are known for these problems.

References

[1] N. Alon, D. Moshkovitz, and S. Safra. Algorithmic construction of sets for k-restrictions. *ACM Transactions on Algorithms*, pages 153–177, 2006.

[2] C. Ambühl, T. Erlebach, M. Mihalák, and M. Nunkesser. Constant-factor approximation for minimum-weight (connected) dominating sets in unit disk graphs. In *Proceedings of the 9th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2006)*, LNCS 4110, pages 3–14. Springer, 2006.

[3] P. Berman, G. Calinescu, C. Shah, and A. Zelikovsky. Power efficient monitoring schedules in sensor networks. In *IEEE Wireless Communication and Networking Conference (WCNC 2004)*, pages 2329–2334, 2004.

[4] P. Berman, G. Calinescu, C. Shah, and A. Zelikovsky. Efficient energy management in sensor networks. In Y. Xiao and Y. Pan, editors, *Ad Hoc and Sensor Networks, Wireless Networks and Mobile Computing*, volume 2. Nova Science Publishers, 2005.

[5] J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. An improved LP-based approximation for Steiner tree. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 583–592. ACM, 2010.

[6] M. Cardei and D.-Z. Du. Improving wireless sensor network lifetime through power aware organization. *Wireless Networks*, 11:333–340, May 2005.

[7] M. Cardei, M. T. Thai, Y. Li, and W. Wu. Energy-efficient target coverage in wireless sensor networks. In *Proceedings of INFOCOM 2005*, March 2005.

[8] C. Chekuri, K. L. Clarkson, and S. Har-Peled. On the set multi-cover problem in geometric settings. In *Proceedings of the 25th Annual Symposium on Computational Geometry (SCG'09)*, pages 341–350. ACM, 2009.

[9] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86:165–177, 1990.

[10] D. Dai and C. Yu. A $(5+\epsilon)$-approximation algorithm for minimum weighted dominating set in unit disk graph. *Theoretical Computer Science*, 410(8-10):756–765, 2009.

[11] A. Dhawan, C. T. Vu, A. Zelikovsky, Y. Li, and S. K. Prasad. Maximum lifetime of sensor networks with adjustable sensing range. In *SNPD-SAWN 2006*, 2006.

[12] T. Erlebach and M. Mihalák. A $(4+\epsilon)$-approximation for the minimum-weight dominating set problem in unit disk graphs. In *Proceedings of the 7th International Workshop on Approximation and Online Algorithms (WAOA 2009)*, LNCS 5893, pages 135–146. Springer, 2009.

[13] T. Erlebach and A. Shahnaz. Approximating node-weighted multicast trees in wireless ad-hoc networks. In *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly (IWCMC 2009)*, pages 639–643. ACM, 2009.

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

[15] N. Garg and J. Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM J. Comput.*, 37(2):630–652, 2007.

[16] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32(1):130–136, 1985.

[17] Y. Huang, X. Gao, Z. Zhang, and W. Wu. A better constant-factor approximation for weighted dominating set in unit disk graph. *Journal of Combinatorial Optimization*, 18(2):179–194, 2009.

[18] M. Marta, Y. Yang, and M. Cardei. Energy-efficient composite event detection in wireless sensor networks. *Wireless Algorithms, Systems, and Applications*, pages 94–103, 2009.

[19] P. Sanders and D. Schieferdecker. Lifetime maximization of monitoring sensor networks. In *Proceedings of the 6th International Workshop on Algorithms for Sensor Systems, Wireless Ad Hoc Networks, and Autonomous Mobile Entities (ALGOSENSORS 2010)*, LNCS 6541, pages 134–147. Springer, 2010.

[20] M. T. Thai, F. Wang, D. H. Du, and X. Jia. Coverage problems in wireless sensor networks: designs and analysis. *International Journal of Sensor Networks*, 3:191–200, May 2008.

[21] C. Vu, R. Beyah, and Y. Li. Composite event detection in wireless sensor networks. In *IEEE International Performance, Computing, and Communications Conference (IPCCC 2007)*, pages 264–271, 2007.

[22] Q. Zhao and M. Gurusamy. Lifetime maximization for connected target coverage in wireless sensor networks. *IEEE/ACM Transactions on Networking*, 16:1378–1391, December 2008.

[23] F. Zou, X. Li, S. Gao, and W. Wu. Node-weighted Steiner tree approximation in unit disk graphs. *Journal of Combinatorial Optimization*, 18(4):342–349, 2009.

[24] F. Zou, Y. Wang, X. Xu, X. Li, H. Du, P. Wan, and W. Wu. New approximations for minimum-weighted dominating sets and minimum-weighted connected dominating sets on unit disk graphs. *Theoretical Computer Science*, 412:198–208, January 2011.