The Within-Strip Discrete Unit Disk Cover Problem

Robert Fraser * Alejandro López-Ortiz †

Abstract

We investigate the Within-Strip Discrete Unit Disk Cover problem (WSDUDC), where one wishes to find a minimal set of unit disks from an input set $\mathcal{D}$ so that a set of points $\mathcal{P}$ is covered. Furthermore, all points and disk centres are located in a strip of height h, defined by a pair of parallel lines. We give a general approximation algorithm which finds a $3\lceil 1/\sqrt{1-h^2}\rceil$-factor approximation to the optimal solution. We also provide a 4-approximate solution given a strip where $h \leq 2\sqrt{2}/3$, and a 3-approximation in a strip if $h \leq 4/5$, improving over the 6-approximation for such strips using the general scheme. Finally, we show that WSDUDC is NP-complete for a strip with any height $h > 0$.

1 Introduction

We are interested in updating an area of seafloor terrain following a survey by a ship. A survey is typically performed by towing an echosounder behind a ship, which results in a swath of data points along the ship's path. If the ship performs this task while in transit, the surveyed area forms a strip along the seafloor. We wish to update the seafloor data set by treating the new data set $\mathcal{P}$ as the standard, while maintaining representative points from the old data set $\mathcal{Q}$ for completeness. Our goal is to find a minimum cardinality set $\mathcal{Q}^\star \subseteq \mathcal{Q}$ so that each point in $\mathcal{P}$ is within a predefined unit distance of a point in $\mathcal{Q}^\star$. This is an instance of the *Within-Strip Discrete Unit Disk Cover* (WSDUDC) problem.

In the WSDUDC problem, the input consists of a set of m unit disks $\mathcal{D}$ with centre points $\mathcal{Q}$, and a set of n points $\mathcal{P}$, all of which lie in the Euclidean plane. We define the *strip* s of height h as the region of the plane between two parallel lines ℓ_1 and ℓ_2, where $\mathcal{Q} \cap s = \mathcal{Q}$ and $\mathcal{P} \cap s = \mathcal{P}$. We assume that we are provided with the lines ℓ_1 and ℓ_2; alternatively, a minimum width strip may be computed. We wish to determine the minimum cardinality set of disks $\mathcal{D}^\star \subseteq \mathcal{D}$ such that $\mathcal{P} \cap \mathcal{D}^\star = \mathcal{P}$. This is a seemingly simpler context than the general Discrete Unit Disk Cover (DUDC) problem, which has no strip confining the positions of the points and disks. The DUDC problem is NP-complete [11], and has received attention due to applications in wireless networking and related optimization problems [15].

This paper addresses an open question regarding the hardness of the general DUDC problem. An implication of a polynomial time algorithm for WSDUDC for strips of any fixed width would be a simple PTAS for DUDC, using the shifting techniques of Hochbaum and Maass [10]. The recent PTAS for DUDC [13], as discussed shortly, uses fundamentally different techniques.

The notion of decomposing a problem into strip-based subproblems is natural, since an exact algorithm or PTAS for the subproblem can potentially be used to derive a general PTAS using the "shifting strategy" [10]. For example, the PTAS for the geometric unit disk cover problem (like DUDC except the centres of the disks are unrestricted) operates by dividing the problem into strips [10]. The maximum independent set of a unit disk graph may be found in polynomial time if the setting is confined to a strip of fixed height [12]. Geometric set cover on unit squares (precisely WSDUDC, except the disks are replaced with axis-aligned unit squares) may be solved optimally in $n^{O(k)}$ time when confined to strips of height k [8]. Considering these results, the hardness of WSDUDC is somewhat surprising.

The WSDUDC problem was formally introduced by Das et al. [7], as a subroutine for their DUDC approximation algorithm. In that work, it was demonstrated that points in a strip of height $1/\sqrt{2}$ (≈ 0.707) may be covered in $O(mn + n \log n)$ time using a fixed partitioning technique to obtain a 6-approximate algorithm.

The Strip-Separated Discrete Unit Disk Cover (SSDUDC) problem was first addressed by Ambühl et al. [2, Lemma 1]. The input consists of a set of points $\mathcal{P}$ located in a strip in the plane, like WSDUDC, but the set of unit disk centres $\mathcal{Q}$ lies strictly outside of the strip rather than in the strip. In the appendix, we outline an $O(m^2 n + n \log n)$ time exact algorithm for SSDUDC based on [2], which we use as a subroutine in our work. The Line-Separated Discrete Unit Disk Cover (LSDUDC) problem has a single line separating $\mathcal{P}$ from $\mathcal{Q}$. A version of LSDUDC was first discussed by [6], where a 2-approximate solution was given; an exact algorithm for LSDUDC was presented in [5]. Another generalization of this problem is the Double-Sided Disk Cover (DSDC) problem, where disks centred in a strip are used to cover points outside of the strip. This also has an exact dynamic programming solution [14].

Many papers have addressed DUDC using a variety of techniques, e.g. [4, 6]; a summary of such re-

*University of Waterloo, Canada, r3fraser@uwaterloo.ca
†University of Waterloo, Canada, alopez-o@uwaterloo.ca

sults is presented in [7]. Brönnimann and Goodrich [3] established the first constant factor approximation algorithm based on epsilon nets. Mustafa and Ray [13] described a PTAS for a more general version of DUDC based on local search. Interest in research on approximation algorithms for DUDC and related problems has remained high because of the large running time associated with the PTAS ($O(m^{65}n)$ for a 3-approximation, $O(m^{O(1/\varepsilon)^2}n)$ in general for $0 < \varepsilon \leq 2$). The best tractable result for DUDC is that of [7], which describes a 18-approximate algorithm which runs in $O(mn + n \log n)$ time.

1.1 Our Results

We provide a general $3\lceil 1/\sqrt{1-h^2}\rceil$-approximate algorithm for solving the Within-Strip Discrete Unit Disk Cover (WSDUDC) problem on strips of height $h < 1$, which runs in $O(m^2n + n \log n)$ time. Given a strip of height at most $2\sqrt{2}/3$ (≈ 0.94), a 4-approximate solution is given which refines the general algorithm by checking for simple redundancy while still running in $O(m^2n + n \log n)$ time. For a strip of height at most $4/5$, an $O(m^6n)$ time 3-approximate solution is provided which uses dynamic programming to solve all subproblems optimally (using the general $3\lceil 1/\sqrt{1-h^2}\rceil$-approximate algorithm on strips of height $2\sqrt{2}/3$ or $4/5$ would produce a 6-approximation). To conclude, we show that WSDUDC is NP-complete.

2 Approximation Algorithms for WSDUDC

In this section, we present algorithms for approximating the optimal WSDUDC solution. We begin with a general technique, followed by refinements which achieve better approximation factors in narrower strips.

Theorem 1 *Given a strip of height $h < 1$, we may find a $3\lceil 1/\sqrt{1-h^2}\rceil$-approximation to the WSDUDC problem in $O(m^2n+n \log n)$ time. If $h \leq 2\sqrt{2}/3$, we can improve the approximation factor to 4 in $O(m^2n+n \log n)$ time. Given a strip of height $h \leq 4/5$, a 3-approximate solution may be found in $O(m^6n)$ time.*

We define the set of rectangles $\mathcal{R}^\circ$, where R_i° is the largest rectangle of height $2h$ which may be covered by $D_i \in \mathcal{D}$, where the strip s has height h and is assumed to be horizontal. Further, we use a set of rectangles $\mathcal{R}$ of height h, defined as $R_i = R_i^\circ \cap s, \forall R_i^\circ \in \mathcal{R}^\circ$.

Observation 1 *Suppose we are given a strip of height $h < 1$ and a unit disk D whose centre lies in the strip. R° is defined as the rectangle of height $2h$ and width $k = 2\sqrt{1-h^2}$ which is circumscribed by D. If a point q is covered by R°, then D also covers q. Furthermore, R° covers the entire height of the strip.*

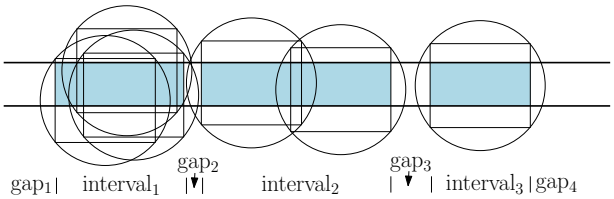

Figure 1: Intervals are continuous segments of the strip covered by the rectangles in $\mathcal{R}$, and gaps are the segments of the strip outside of the intervals.

We divide the set of points $\mathcal{P}$ into two sets $\mathcal{P} = \mathcal{P}_\mathcal{R} \cup \mathcal{P}_{\overline{\mathcal{R}}}$, where $\mathcal{P}_\mathcal{R}$ is the set of points covered by the set of rectangles $\mathcal{R}$, and $\mathcal{P}_{\overline{\mathcal{R}}} = \mathcal{P} \setminus \mathcal{P}_\mathcal{R}$, i.e. those points covered by $\mathcal{D}$ but not $\mathcal{R}$. The approximation algorithms proceed in two stages to compute the cover: first the points in $\mathcal{P}_{\overline{\mathcal{R}}}$ are covered, and then the remaining uncovered points in $\mathcal{P}_\mathcal{R}$ are covered. We refer to the points in $\mathcal{P}_{\overline{\mathcal{R}}}$ as occurring in the *gaps* of the strip, and the points in $\mathcal{P}_\mathcal{R}$ are in the *intervals* (see Figure 1). In our discussion, we assume that $h > 0$, so that $k = 2\sqrt{1-h^2} < 2$. [1]

2.1 Covering $\mathcal{P}_{\overline{\mathcal{R}}}$

The centres of all disks are separated from the points in $\mathcal{P}_{\overline{\mathcal{R}}}$ by vertical lines (those of the gap boundaries). For each gap of the strip, the points are covered optimally with the $O(m^2n+n \log n)$ time algorithm for SSDUDC. While points in each gap are covered optimally, we may lose optimality when we combine these solutions[2]. Recall that rectangles have width $k = 2\sqrt{1-h^2}$. There is a rectangle for each disk, and so no disk centre lies within a distance of $k/2$ of any gap. By interleaving rectangles with gaps of width ε, a disk may cover points in $2\lceil 1/k - 1/2\rceil$ gaps as $\varepsilon \to 0$. To see this, consider the right side of a disk D_i, where R_i defines an interval of width $k/2$ on this right side. Since D_i has unit radius, $\lceil (1 - k/2)/k\rceil$ additional intervals (and thus gaps, one to the left of each interval) may be at least partially covered by the right half of D_i. Thus, the union of the solutions for each gap has an approximation factor of $2\lceil 1/k - 1/2\rceil$ for covering $\mathcal{P}_{\overline{\mathcal{R}}}$.

2.2 Covering $\mathcal{P}_\mathcal{R}$

To cover the points remaining after the previous step, we iteratively add the right-most rectangle that covers the left-most remaining point to the solution, as detailed in Algorithm 1 (Greedy-Rectangles).

[1] If $h = 0$, all points and disk centres are collinear, and $\mathcal{P}_{\overline{\mathcal{R}}}$ is empty. This setting is solved optimally by the Greedy-Rectangles algorithm detailed in Section 2.2.

[2] Covering the points in the union of the gaps cannot be covered optimally in general, as the hardness proof for WSDUDC (Section 3) only has points in gaps.

Algorithm 1 GREEDY-RECTANGLES$(\mathcal{R}, \mathcal{P}_\mathcal{R})$

$\mathcal{R}' \leftarrow \emptyset$, sort $\mathcal{R}$ by x-coordinate, sort $\mathcal{P}_\mathcal{R}$ by left boundary
while $\mathcal{P}_\mathcal{R} \neq \emptyset$ **do**
  $p_\ell \leftarrow$ left-most point in $\mathcal{P}_\mathcal{R}$
  $R_r \leftarrow$ right-most rectangle in $\mathcal{R}$ covering p_ℓ
  $\mathcal{R}' = \mathcal{R}' \cup R_r$
  $\mathcal{P}_\mathcal{R} = \mathcal{P}_\mathcal{R} \setminus (R_r \cap \mathcal{P}_\mathcal{R})$
return $\mathcal{R}'$

Lemma 2 *A rectangle R'_i selected by* GREEDY-RECTANGLES *may overlap another rectangle R'_{i-1} (the previous rectangle chosen) by $k - \varepsilon$, for any $\varepsilon > 0$.*

Proof. The left rectangle R'_{i-1} covers a point p_ℓ which is to the left of the right rectangle R'_i, which in turn covers a point p_r to the right of R'_{i-1}. By pushing p_ℓ to the left edge of R'_{i-1} and p_r to the right edge of R'_i, the rectangles may be overlapped so that R'_{i-1} lies ε to the left of R'_i, and both are chosen. □

Lemma 3 *Let $\mathcal{R}' = \{R'_1, \ldots, R'_{|\mathcal{R}'|}\}$ be the set of rectangles found by* GREEDY-RECTANGLES*, indexed from left to right so that $\forall i, j, i < j \leftrightarrow left(R'_i, R'_j)$ where $left(R'_i, R'_j)$ indicates that R'_i is left of R'_j. Then $\forall i, j, j > i + 1 \rightarrow R'_i \cap R'_j = \emptyset$.*

Proof. Suppose that this is not the case and R'_i and R'_j intersect although there exist rectangles $R'_{i+1}, \ldots, R'_{j-1} \in \mathcal{R}'$, where $i + 1 \leq j - 1$. Let $\mathcal{P}'$ be the set of points covered by the rectangles $R'_{i+1}, \ldots, R'_{j-1}$. $\mathcal{P}'$ is covered entirely by $R'_i \cup R'_j$, and none of $R'_{i+1}, \ldots, R'_{j-1}$ are the right-most rectangle covering a point to the right of R'_i, so they would not be chosen by GREEDY-RECTANGLES. □

Lemma 4 GREEDY-RECTANGLES *computes a cover of $\mathcal{P}_\mathcal{R}$ with an approximation factor of $3\lceil 1/k - 1/2 \rceil$ times the optimal solution.*

Proof. Consider the maximum number of rectangles in the GREEDY-RECTANGLES solution that may be replaced by a single disk D_i in the strip. One of the rectangles available to the algorithm is $R_i \subset R_i^\circ$, where R_i° is circumscribed by D_i. By Lemma 2, there may be another rectangle ε to the left or right of R_i which will be selected by the algorithm, and so the approximation factor is at least 2. It may be possible to pack additional pairs of nearly overlapping rectangles as densely as permitted by Lemma 3 so that the points covered by these rectangles are also covered by D_i. Since all disks have unit radius and R_i° is circumscribed, each side of D_i can potentially cover all points covered by at most $2\lceil (1 - k/2)/k \rceil - 1$ additional rectangles (see Figure 2). This analysis is similar to Section 2.1, but now all rectangles are paired except for the right-most one (in a right-most pair, the region covered only by the right rectangle cannot be covered at all by D_i since we consider the *pairs* to have width k, i.e. $\varepsilon = 0$). Thus, the total approximation factor is $4\lceil 1/k - 1/2 \rceil$. □

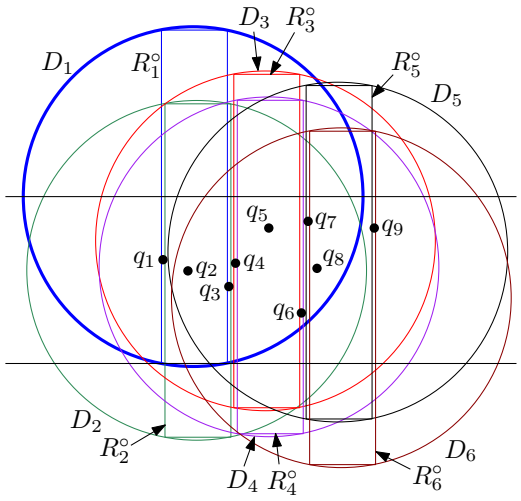

Figure 2: An illustration of Lemma 4. We wish to know the number of rectangles in which a disk, say D_1, may possibly cover all points. D_1 may be paired with another rectangle as described in Lemma 2, we illustrate this with D_2. We now wish to determine how many distinct rectangles may be covered to the right of this pair (the case for the left side is analogous). In this case, we have $h = 0.97$, and so $2\lceil (1-k/2)/k \rceil - 1 = 3$. We see this with $R_3^\circ, R_4^\circ, R_5^\circ$, as D_1 covers all points in these rectangles. It is possible that D_1 covers points contained in a rectangle paired with the right-most rectangle in this set (e.g., $q_8 \in D_1 \cap R_6^\circ$), but such points are covered by R_5° as well.

GREEDY-RECTANGLES requires both the set of rectangles $\mathcal{R}$ and the set of points $\mathcal{P}_\mathcal{R}$ to be sorted in left to right order. The sorted lists are each walked through once, so the running time is $O(m \log m + n \log n)$.

2-approximation when $k \geq 2/3$ ($h \leq 2\sqrt{2}/3$). The general algorithm for covering $\mathcal{P}_\mathcal{R}$ presented above has an approximation factor of 4 when $k \geq 2/3$. For each pair of consecutive rectangles R'_{i-1} and R'_i found by GREEDY-RECTANGLES, we determine whether there exists a disk D_j such that $(R'_{i-1} \cup R'_i) \cap \mathcal{P} \subseteq D_j \cap \mathcal{P}$. To do so, we run through $\mathcal{R}'$ in order, and check whether the current pair may be replaced by any disk in $\mathcal{D}$.

Consider a disk $D_i \in \mathcal{D}^\star$, which may or may not be a member of our refined solution set. D_i may intersect at most four rectangles in $\mathcal{R}'$. Every consecutive pair of rectangles in $\mathcal{R}'$ now requires at least two disks, so at least two disks are required to cover any four consecutive rectangles. Therefore, the overall approximation factor is two. This operation will scan m disks for every possible disk to remove from the solution, so the

operation takes $O(m^2n + n \log n)$ time.

Optimal solution when $k \geq 6/5$ ($h \leq 4/5$). In this case[3], the $\mathcal{P}_{\mathcal{R}}$ sub-problem may be solved optimally using dynamic programming. We define a set of disks D_s as *mutually spanning* if each disk in D_s covers a non-empty set of points which lies to the left of all other disks in D_s, as well as a non-empty set of points lying to the right of all other disks in D_s.

Lemma 5 *If $h \leq 4/5$, an optimal solution to $\mathcal{P}_{\mathcal{R}}$ requires mutually spanning sets of cardinality at most 3.*

Proof. Suppose that there exists a mutually spanning set of four disks in the optimal solution. Recall that each point in the set $\mathcal{P}_{\mathcal{R}}$ is covered by some rectangle circumscribed by a disk. Using Lemma 3, we show that a set of four rectangles exists which covers all of the points covered by the four disks. At least one rectangle is required to cover the left-most point, then there may be a pair of width at least k, and a final rectangle spans an additional k. The maximum width in a strip for a mutually spanning set of disks of any cardinality is $3-k/2$; note that $2k > 3-k/2$ when $k > 6/5$. Therefore, any solution which uses four mutually spanning disks to cover a set of points $\mathcal{P}'$ may use (at most) four rectangles to cover a set of points $\mathcal{P}''$, where $\mathcal{P}' \subseteq \mathcal{P}''$. □

By Lemma 5, a dynamic program which add disks to the solution in a left-to-right fashion need only consider up to triples of disks to terminate sub-problems to ensure that the sub-problems are independent and optimal. Such a dynamic program is described in Algorithm 2. In the algorithm, $\mathcal{D}^2$ and $\mathcal{D}^3$ are the sets of mutually spanning doubles and triples of disks respectively, and $\mathfrak{D}$ is the set of all sets of disks under consideration. Given two sets $\mathcal{D}_i, \mathcal{D}_j \in \mathfrak{D}$, if $\mathcal{D}_i$ covers points left of $\mathcal{D}_j$, and $\mathcal{D}_j$ does not cover points left of $\mathcal{D}_i$, we write $\mathcal{D}_i <_c \mathcal{D}_j$ to indicate this relationship. Otherwise, we consider them incomparable under this operator. Hence, we may establish a partially ordered set over all of the sets in $\mathfrak{D}$ w.r.t. the $<_c$ operator. Note that directed cycles are impossible in this set, since the transitive property holds for the $<_c$ operator. We impose a topological sorting $\mathfrak{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_{|\mathfrak{D}|}\}$ so that for any two sets $\mathcal{D}_i, \mathcal{D}_j$ in this ordering, we have that $i < j \rightarrow \mathcal{D}_j \not<_c \mathcal{D}_i$.

The correctness of Optimal-$\mathcal{P}_{\mathcal{R}}$ follows from the fact that all points left of a set $\mathcal{D}_i$ are covered in a valid solution to a subproblem terminating with $\mathcal{D}_i$, and all mutually spanning sets up to size three are considered. Optimal-$\mathcal{P}_{\mathcal{R}}$ runs in $O(m^6 n)$ time: there are $O(m^3)$ possible combinations of disks that we consider in two nested for loops, and inside the nested loop we check the disks against the point set $\mathcal{P}$.

[3]A similar dynamic programming algorithm applies to larger strips, but the running time increases rapidly with h.

Algorithm 2 Optimal-$\mathcal{P}_{\mathcal{R}}$ $(\mathcal{D}, \mathcal{P}_{\mathcal{R}})$ (Assumes $k \geq 6/5$)

```
𝔇 ← 𝒟 ∪ 𝒟² ∪ 𝒟³, m' ← |𝔇|
Topologically sort 𝔇 on the <_c operator
c[0] = 0, c[1 ... m'] = ∞
for i = 1 ... m' do
  for j = 0 ... i − 1 do
    size ← c[j] + |𝒟_i|
    if size < c[i] and no points lie between 𝒟_i and 𝒟_j then
      c[i] ← size
Backtrack on c to recover optimal cover 𝒟*
return 𝒟*
```

2.3 Combining solutions for $\mathcal{P}_{\overline{\mathcal{R}}}$ and $\mathcal{P}_{\mathcal{R}}$

Recall that the approximation factor for covering the entire set of $\mathcal{P}_{\overline{\mathcal{R}}}$ is $2\lceil 1/k - 1/2 \rceil$ and $4\lceil 1/k - 1/2 \rceil$ for covering $\mathcal{P}_{\mathcal{R}}$, where k is the width of the rectangles. We simply sum these factors to get an overall approximation factor of $6\lceil 1/k - 1/2 \rceil < 3\lceil 1/\sqrt{1-h^2} \rceil$ for strips of arbitrary height $h < 1$. The running time is $O(m^2 n + n \log n)$, effectively dominated by the SSDUDC algorithm used to cover $\mathcal{P}_{\overline{\mathcal{R}}}$.

4-approximation when $k \geq 2/3$ ($h \leq 2\sqrt{2}/3$). We have a 2-approximate algorithm for $\mathcal{P}_{\mathcal{R}}$ when $k \geq 2/3$, and we may solve each gap of $\mathcal{P}_{\overline{\mathcal{R}}}$ optimally. For the purposes of counting, we may assume that the disks forming the cover for each gap are equally distributed amongst the neighbouring intervals for both the approximate solution and the optimal one. We are not interested in the worst-case approximation factor in any given interval; rather we are interested in the approximation factor over the strip as a whole. For each gap, only disks found in adjacent intervals may form part of the solution. Disk centres are located at least a distance $1/3$ from the end of an interval, and so disk centres in non-adjacent intervals are more than unit distance away from the gap. Thus, for each interval of the strip, assume that n_ℓ (resp. n_r) disks are used for covering the gap to the left (resp. right), and n_s disks are used for covering the points in the interval. The minimum number of disks required is $\max\{n_\ell, n_s/2, n_r\}$, since both n_ℓ and n_r are optimal and n_s is a 2-approximation. We conclude that $n_\ell + n_s + n_r \leq 4 \cdot \max\{n_\ell, n_s/2, n_r\}$, and thus it is a 4-approximation algorithm. Again, the running time is $O(m^2 n + n \log n)$.

3-approximation when $k \geq 6/5$ ($h \leq 4/5$). We have optimal algorithms for computing the cover of each gap

of $\mathcal{P}_{\overline{\mathcal{R}}}$ and each interval of $\mathcal{P}_{\mathcal{R}}$. Further, the disks covering a gap only come from the two adjacent intervals, and the disks covering an interval only come from the interval itself. Since the disks in each interval can contribute to only three problems, each of which is solved optimally, the worst-case is that three times the optimal number of disks is used. The running time of the algorithm is dominated by OPTIMAL-$\mathcal{P}_{\mathcal{R}}$, so the overall running time is $O(m^6 n)$.

2.4 Application to DUDC

As a corollary of our results, we note that we may slightly improve upon the approximation factor of the previous best result for DUDC outside of the PTAS (an 18-approximation [7]), albeit with an increase in running time. Their algorithm first divides the plane into horizontal strips of height $1/\sqrt{2}$, and WSDUDC is run in each strip using a 6-approximate algorithm. Remaining uncovered points are covered using disks centred outside their strips with a 12-approximate algorithm. Substituting the 3- (resp. 4-) approximate WSDUDC algorithm outlined in this paper provides a 15- (resp. 16-) approximate algorithm for DUDC, which runs in $O(m^6 n)$ (resp. $O(m^2 n + n \log n)$) time.

Corollary 6 *There is a 15- (resp. 16-) approximate algorithm for DUDC, which runs in $O(m^6 n)$ (resp. $O(m^2 n + n \log n)$) time.*

3 Hardness of WSDUDC

We prove that WSDUDC is NP-complete by reducing from the minimum vertex cover problem (VERTEX-COVER) on planar graphs of maximum degree three, which is known to be NP-complete (and APX-hard) [9, 1]. Recall the setting for VERTEX-COVER: We are given a graph $G = (V, E)$, and we seek a minimum cardinality subset $V^\star \subseteq V$ such that for all $e_{(i,j)} = (v_i, v_j) \in E$, either $v_i \in V^\star$ or $v_j \in V^\star$. In other words, the vertex cover is a minimum cardinality hitting set of all of the edges in the graph.

Theorem 7 *WSDUDC is NP-complete.*

WSDUDC is in NP, since a certificate may be provided as a set of disks that covers all of the points in $\mathcal{P}$, which is trivial to verify.

In the reduction, we create an instance of WSDUDC from a planar graph so that a solution $\mathcal{D}^\star$ to the WSDUDC problem provides a solution $V^\star$ to the VERTEX-COVER problem on the graph. For our reduction, it is easier to consider the dual (disk piercing) setting of WSDUDC. The Within-Strip Discrete Unit Disk Piercing problem (WSDUDP) accepts a set of points $\mathcal{Q}$, a set of unit disks $\mathcal{D}_{\mathcal{P}}$ with centre points $\mathcal{P}$, and a strip of height h as inputs, and computes the minimum number of points $\mathcal{Q}^\star \subseteq \mathcal{Q}$ such that each disk in $\mathcal{D}_{\mathcal{P}}$ contains at least one point from $\mathcal{Q}^\star$. Let WS(G) be the WSDUDP instance created from a graph G. Note that a solution $\mathcal{Q}^\star$ for WSDUDP is exactly the set of centre points to $\mathcal{D}^\star$, the optimal solution to the WSDUDC problem in the primal setting.

Assume that we have a planar embedding of the graph and a horizontal strip so that the terms *left*, *right*, *above* and *below* are all well defined. Let ℓ^v_{vert} be a vertical line through vertex v. For the reduction, we make use of *dummy vertices*, which are simply extra vertices that we may place on an edge of the graph G. A *dummy edge* is an edge which is incident upon at least one dummy vertex. Informally, the steps of the reduction are:

1. Obtain a planar embedding of G where each vertex has a distinct x-coordinate.
2. For any vertex v with degree three where all incident edges are left or right of ℓ^v_{vert}, 'bend' the lowest edge with a dummy vertex so that the edge becomes incident to v from the opposite side of ℓ^v_{vert}, call this new graph $G' = (V', E')$.
3. For each vertex $v \in V'$, add a dummy vertex at each point where $\ell^v_{\text{vert}} \cap e \neq \emptyset, \forall e \in E'$.
4. Identify each vertex v of degree one or two where all edges are incident on the same side of ℓ^v_{vert}, say w.l.o.g. the edges are incident from the right. Place a vertical line ℓ_{vert} between v and the next vertex to the left, and add a dummy vertex at each point where $\ell_{\text{vert}} \cap e \neq \emptyset, \forall e \in E'$. This ensures that consecutive vertical arrays of vertices differ in cardinality by at most one.
5. For any pair of vertices $v_i, v_j \in V$, ensure that an even number of vertices occur any path from v_i to v_j in G', by adding additional dummy vertices.
6. Create the WSDUDP instance WS(G) from G' so that every edge in E' corresponds to a disk in $\mathcal{D}$ and every vertex in V' to a point in $\mathcal{Q}$. We then show that an optimal solution to WSDUDP provides an optimal cover for G', from which an optimal vertex cover for G may be found, as required.

Lemma 8 *Given an edge $e_{(i,j)}$ of the graph $G = (V, E)$, we can add a pair of adjacent dummy vertices $V_d = \{v_{i_1}, v_{i_2}\}$ along the edge $e_{(i,j)}$ to create the graph $G' = (V \cup V_d, E \cup \{e_{(i,i_1)}, e_{(i_1,i_2)}, e_{(i_2,j)}\} \setminus \{e_{(i,j)}\})$. The graph remains planar, and the cardinality of the optimal solution to* VERTEX-COVER *over G' is $|V^\star| + 1$, where $V^\star$ is the set of vertices in a minimum vertex cover of G.*

Proof. By placing the dummy vertices directly on the edge $e_{(i,j)}$ in any embedding, planarity is preserved. There are two possibilities for a minimum vertex cover

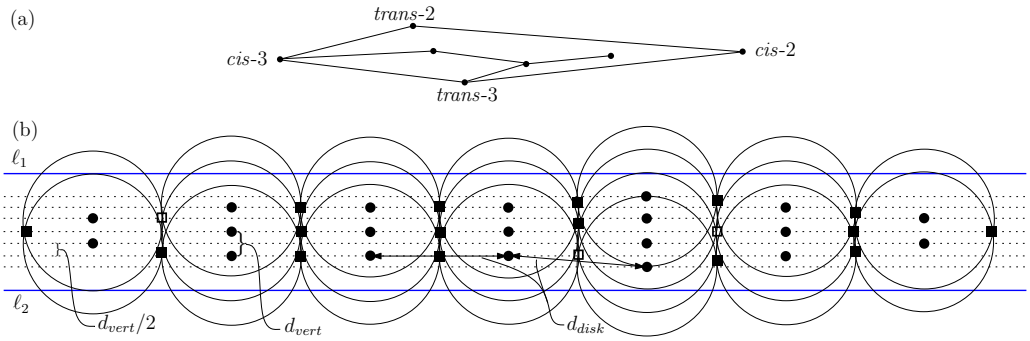

Figure 3: A sample WSDUDP construction WS(G) for the NP-hardness reduction. (a) Given a graph G, we compute a planar embedding (see Section 3.1 for vertex classes). (b) We construct a series of stacks of disks, where disks in adjacent stacks have slight overlap. The disk centres in each stack are aligned vertically and separated by a fixed distance d_{vert}. The number of disks in adjacent stacks may only vary by one. If two consecutive stacks have the same number of disks, the centres are aligned horizontally and separated by d_{disk}. If two consecutive stacks have differing numbers of disks, the centres are staggered vertically by $d_{\text{vert}}/2$, so that each disk centre is d_{disk} from two disk centres in the adjacent stack (thus, these stacks are distance $\sqrt{d_{\text{disk}}^2 - d_{\text{vert}}^2}$ apart). The points of $\mathcal{Q}$ are indicated by squares; those points stabbing three disks are empty. The centre points of the disks $\mathcal{P}$ are displayed as filled circles.

$V^\star$ over an edge $e_{(i,j)}$ in $G = (V, E)$; either one or both of its vertices are in $V^\star$. The addition of a pair of dummy vertices $\{v_{i_1}, v_{i_2}\}$ on $e_{(i,j)}$ creates three edges in place of $e_{(i,j)}$: $\{e_{(i,i_1)}, e_{(i_1,i_2)}, e_{(i_2,j)}\}$, with $e_{(i,i_1)}$ and $e_{(i_1,i_2)}$ sharing the dummy vertex v_{i_1}, and $e_{(i_1,i_2)}$ and $e_{(i_2,j)}$ sharing the other dummy vertex v_{i_2}.

In the first case, $e_{(i,j)}$ is covered by one vertex from $V^\star$, say v_i. $e_{(i,i_1)}$ is covered by $V^\star$, but $e_{(i_1,i_2)}$ and $e_{(i_2,j)}$ are not. The only optimal cover is to add the shared dummy vertex v_{i_2} to the solution. Therefore, the size of the optimal solution increases by one. In the second case, $e_{(i,j)}$ was covered by both v_i and v_j in $V^\star$, and so $e_{(i,i_1)}$ and $e_{(i_2,j)}$ are covered. Therefore, arbitrarily selecting one of the dummy nodes to cover $e_{(i_1,i_2)}$, say v_{i_1}, increases the size of the optimal solution by one. Although $e_{(i,i_1)}$ is covered by two vertices, the topology of the graph and cover local to v_i remains unchanged. □

Lemma 9 *Given any optimal solution $V^\star_{G'}$ to* Vertex-Cover *on G', we can find an optimal solution $V^\star_G$ to* Vertex-Cover *on G in polynomial time.*

Proof. Suppose there is only one pair of dummy nodes in G'. At least one of the dummy vertices from the pair $\{v_{i_1}, v_{i_2}\}$ must be in the optimal solution $V^\star_{G'}$, since they are the endpoints of the edge $e_{(i_1,i_2)}$. If only one is in the optimal solution, say v_{i_1}, we can discard it and return $V^\star_G = V^\star_{G'} \setminus \{v_{i_1}\}$ as an optimal solution to Vertex-Cover on G.

If both v_{i_1} and v_{i_2} are in $V^\star_{G'}$, we must discard both and add either v_i or v_j to the solution for $V^\star_G$. If v_i or v_j is also in $V^\star_{G'}$, then the adjacent dummy vertex is extraneous. Without loss of generality, say v_i, v_{i_1} and v_{i_2} are in $V^\star_{G'}$; this is a contradiction since $V^\star_{G'} \setminus \{v_{i_1}\}$ provides a vertex cover. Therefore neither v_i nor v_j is in $V^\star_{G'}$, and we may arbitrarily set $V^\star_{G'} = (V^\star_{G'} \setminus \{v_{i_1}\}) \cup \{v_i\}$. This provides another vertex cover for G' which is of equal cardinality, and thus is optimal. Now we are in the first case again.

If there are more pairs of dummy vertices, this argument may be applied iteratively to adjacent pairs in G' until each pair of dummy vertices has only one vertex in $V^\star_{G'}$. If there is exactly one dummy vertex from each pair in the optimal vertex cover, these may be removed to provide an optimal vertex cover $V^\star_G$ for G by Lemma 8. □

An example WSDUDP construction WS(G) is shown in Figure 3, to provide intuition for the gadgets used in the reduction. Each edge of the graph G' (actual or dummy) corresponds to a disk in WS(G), and each vertex (actual or dummy) corresponds to a point in $\mathcal{Q}$. A point in $\mathcal{Q}$ stabs two disks in WS(G) if the degree of the corresponding vertex in G' is two; the remaining points stab three disks and their corresponding vertices have degree three.

A *wire* w_i is a sequence of disks positioned so that consecutive centres are spaced d_{disk} units apart, not necessarily collinearly, where $2\sqrt{1-h_\ell^2} < d_{\text{disk}} < \sqrt{2+2\sqrt{1-(3h_\ell/4)^2}}$, so that there exists a small area of overlap between consecutive disks which contains a point in $\mathcal{Q}$.[4] Disk centres on adjacent wires are $d_{\text{vert}} = 3h_\ell/2$ units apart vertically, and we define a *stack* as a set of such vertically aligned disks. The centres of the disks in a stack are shifted within the strip by $d_{\text{vert}}/2$ relative to an adjacent stack when the number of disks in the two stacks differs, while the distance between consecutive centres in each wire remains d_{disk}.

[4] Note that $2\sqrt{1-h_\ell^2} < \sqrt{2+2\sqrt{1-(3h_\ell/4)^2}}$ for $h_\ell > 0$.

Lemma 10 *There is a non-empty area of intersection between three disks in consecutive stacks when the centres of the stacks are shifted by $d_{vert}/2$ relative to each other, and $d_{disk} < \sqrt{2+2\sqrt{1-(3h_\ell/4)^2}}$.*

Proof. Without loss of generality, assume that disks D_1, D_2 are in the left stack and D_3 is in the right stack, and their centrepoints are p_1, p_2, p_3 respectively. Let $\perp_{p_1,p_2}$ be the perpendicular bisector between p_1 and p_2; p_3 lies on $\perp_{p_1,p_2}$. Let ∂_i denote the boundary of D_i. Then there is a unique point to the right of p_1 and p_2 where $\perp_{p_1,p_2} \cap \partial_1 \cap \partial_2$, call this point p_r. Now if p_r and p_3 are less than unit distance apart, then D_3 covers p_r and there is a non-empty area of intersection between D_1, D_2, and D_3. Elementary arithmetic shows that this holds when $d_{\text{disk}} < \sqrt{2+2\sqrt{1-(3h_\ell/4)^2}}$. □

3.1 Gadgets

In the graph, we may encounter vertices of degree one, two, or three. With each vertex, wires may begin, end, split, merge, or continue unchanged. For vertices of degree one, the incident edge will correspond to a terminal disk on a wire. For vertices of degree two, if one edge leaves to the left and the other to the right in the embedding, this is a *trans*-2 vertex[5], and we handle it by continuing all wires. If both edges go in the same direction (left or right), we call this a *cis*-2 vertex, and we have a gadget to merge the pair of wires corresponding to the edges. Analogously, we have gadgets for both the *trans*-3 and *cis*-3 degree three vertices. Finally, we build a gadget to increase the number of vertices on an edge. With each gadget, we apply the analogous modification to G' by adding dummy vertices to the respective edges. This ensures that an optimal solution to WS(G) corresponds exactly to an optimal vertex cover for G'.

***cis*-2 Gadget.** In this case, a pair of wires will terminate, and since the two terminal disks correspond to a pair of edges sharing a vertex, we place a vertex in the area covered by both disks and no others. An extra column of dummy nodes should be used to extend all other wires if the vertex is on an interior face of the planar embedding of the graph, since two wires are terminated simultaneously, and we may only shift wires by $d_{\text{vert}}/2$ with each column.

***trans*-3 Gadget.** Suppose we have an upper wire ending in disk D_u and a lower wire ending in disk D_l, and they merge into a single wire beginning with disk D_c. Therefore, we can place D_c at a point so that the distance between the centres of both D_c to D_u and D_c to D_l is d_{disk}, as described in Lemma 10. By placing a vertex in $D_c \cap D_u \cap D_l$, a single point stabs three disks, which corresponds to a vertex which can cover three edges in the graph.

***cis*-3 Gadget.** For this gadget, we combine the *trans*-3 and *cis*-2 gadgets to build a *cis*-3 configuration. In the planar graph embedding, this corresponds to introducing a bend in the lowest edge incident to the *cis*-3 vertex with a dummy vertex, so that it becomes a *trans*-3 vertex.

CARD+ **Gadget.** If the total number of dummy vertices added to an edge of G is odd, we require a gadget which increases the number of disks between a pair of points on a wire by one. An extra disk whose centre is very close to the centre point of a disk on the wire (Figure 4) allows points to be placed so that the wire remains independent from adjacent wires, while increasing the number of disks on the wire by one.

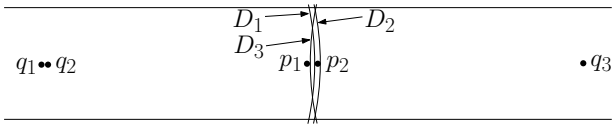

Figure 4: The CARD+ gadget. We add a disk D_2 into a wire by inserting the disk very close to one of the others on the wire. Given D_1 and D_3 on a wire with $p_1 \in D_1 \cap D_3$, we add a disk D_2 whose centre q_2 is beside q_1, and p_1 is moved so that $p_1 \in D_1 \cap D_2, p_1 \notin D_3$. Analogously, $p_2 \in D_2 \cap D_3, p_2 \notin D_1$. Now the wire contains one more disk that previously, and the new disk does not cover any points from adjacent wires.

Now an instance of WSDUDP WS(G) may be constructed from any planar graph G with no vertex of degree greater than three. A solution $\mathcal{Q}^\star$ to WS(G) is also a solution $V^\star_{G'}$ to the VERTEX-COVER problem on $G' = (V', E')$, where $v_i \in V'$ is mapped to $q_i \in \mathcal{Q}$ and $q_i \in D_j \leftrightarrow v_i \in e_j \in E'$. By Lemma 9, we can find a minimum vertex cover $V^\star_G$ for G from $V^\star_{G'}$ in polynomial time. Therefore, there is a hitting set of size $c + (|\mathcal{D}| - |V|)/2$ for WS(G) if and only if there exists a vertex cover of size c for G (exactly half of the extra points added in the construction of WS(G) from G are required for a hitting set for $\mathcal{D}$). The number of disks stacked vertically in any column of WS(G) is in $O(m)$, where m is the number of edges and n is the number of vertices in the graph G. The number of such stacks is in $O(n)$, so the total number of disks and points in the WSDUDP construction is $O(mn)$. This completes the proof of Theorem 7.

4 Conclusions

We have outlined several approximation algorithms for the WSDUDC problem, along with a proof

[5]Borrowing from the Latin terms in isomerism, *trans* means "on the other side", and *cis* means "on the same side". We label them relative to a vertical line through the vertex, so a *cis* vertex has all incident edges to one side of the vertical line.

of NP-completeness for the problem. The general $3\lceil 1/\sqrt{1-h^2}\rceil$-approximate algorithm and the 4-approximate algorithm for strips of height $\leq 2\sqrt{2}/3$ both run in $O(m^2n+n\log n)$ time. Finally, we presented a 3-approximate algorithm for strips of height $\leq 4/5$ which runs in $O(m^6n)$ time. These results were applied to the DUDC problem to provide a 15-approximate algorithm. The running time of the DUDC algorithm is dominated by WSDUDC, so further improvements to our results will directly apply to DUDC as well.

Acknowledgements

Previous work on the WSDUDC problem was the result of joint work with Gautam Das and Bradford Nickerson. Alejandro Salinger, Patrick Nicholson and Francisco Claude participated in helpful discussions on this set of results, and our presentation and several details have been significantly improved by comments and corrections from anonymous reviewers.

References

[1] P. Alimonti and V. Kann. Some APX-completeness results for cubic graphs. *Th. Comp. Sci.*, 237(1-2):123 – 134, 2000.

[2] C. Ambühl, T. Erlebach, M. Mihal'ák, and M. Nunkesser. Constant-factor approximation for minimum-weight (connected) dominating sets in unit disk graphs. In *APPROX*, pages 3–14, 2006.

[3] H. Brönnimann and M. Goodrich. Almost optimal set covers in finite VC-dimension. *Disc. and Comp. Geom.*, 14(1):463–479, 1995.

[4] P. Carmi, M. Katz, and N. Lev-Tov. Covering points by unit disks of fixed location. In *ISAAC*, pages 644–655, 2007.

[5] F. Claude, G. Das, R. Dorrigiv, S. Durocher, R. Fraser, A. López-Ortiz, B. Nickerson, and A. Salinger. An improved line-separable algorithm for discrete unit disk cover. *Disc. Math. Alg. & Appl.*, 2(1):77–87, 2010.

[6] G. Călinescu, I. I. Măndoiu, P.-J. Wan, and A. Z. Zelikovsky. Selecting forwarding neighbors in wireless ad hoc networks. *Mob. Net. & Appl.*, 9(2):101–111, 2004.

[7] G. Das, R. Fraser, A. López-Ortiz, and B. Nickerson. On the discrete unit disk cover problem. In *WALCOM: Alg. & Comp.*, pages 146–157. 2011.

[8] T. Erlebach and E. van Leeuwen. PTAS for weighted set cover on unit squares. In *APPROX*, pages 166–177. 2010.

[9] M. R. Garey and D. S. Johnson. The rectilinear steiner tree problem is NP-complete. *SIAM J. App. Math.*, 32(4):826–834, 1977.

[10] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems inimage processing and VLSI. *J. ACM*, 32:130–136, 1985.

[11] D. Johnson. The NP-completeness column: An ongoing guide. *J. of Alg.*, 3(2):182–195, 1982.

[12] T. Matsui. Approximation algorithms for maximum independent set problems and fractional coloring problems on unit disk graphs. In *JCDCG*, pages 194–200. 2000.

[13] N. Mustafa and S. Ray. Improved results on geometric hitting set problems. *Disc. & Comp. Geom.*, 44:883–895, 2010.

[14] X. Xu and Z. Wang. Wireless coverage via dynamic programming. In *WASA*, pages 108–118. 2011.

[15] D. Yang, S. Misra, X. Fang, G. Xue, and J. Zhang. Two-tiered constrained relay node placement in wireless sensor networks: Efficient approximations. In *(SECON)*, pages 1 –9, 2010.

Appendix

SSDUDC Algorithm

Ambühl et al. [2] outlined a dynamic program for the SSDUDC problem, and established that it was a polynomial time algorithm. In Algorithm 3, we provide pseudocode for SSDUDC based on their ideas, so that we may analyze the worst case running time. We make use of the function $\texttt{neq}(j, j')$, which evaluates to 1 if $j \neq j'$, and 0 otherwise. Assume that the lines ℓ_1 and ℓ_2 are horizontal, and that all points in $\mathcal{P}$ are contained in the strip defined by the lines. Furthermore, assume that all disks in $\mathcal{D}$ are centred outside of the strip. We denote the weight of disk D_i by w_{D_i}, and the weights of the lines are $w_{\ell_1} = w_{\ell_2} = 0$

Algorithm 3 SSDUDC($\mathcal{P}, \mathcal{D}$)

```
Sort P left-to-right
Divide D into D^U and D^L (disks centred above ℓ1 and below ℓ2 resp.)
D^U ← D^U ∪ {ℓ1}, D^L ← D^L ∪ {ℓ2}
for i := 1 to n do
    // Find optimal solution covering all points up to p_{i-1}
    w_opt ← ∞, j' ← 0, k' ← 0
    for j := 1 to |D^U| do
        for k := 1 to |D^L| do
            if i > 1 & w_opt > T[i-1, j, k] then
                w_opt ← T[i-1, j, k], j' ← j, k' ← k
    for j := 1 to |D^U| do
        for k := 1 to |D^L| do
            T[i, j, k] ← ∞ // Base case (i.e. if p_i ∉ D_j^U ∪ D_k^L)
            if p_i ∈ D_j^U ∪ D_k^L then
                if i = 1 // Initialize the DP table then
                    T[1, j, k] ← w_{D_j^U} + w_{D_k^L}
                else
                    T[i, j, k] ← min{T[i, j, k], T[i-1, j, k], T[i-1, j', k'] + neq(j, j') · w_{D_j^U} + neq(k, k') · w_{D_k^L}}
// Backtrack from optimal weight cover on p_n to find D*
return D*
```

This algorithm is dominated by the three for loops and the initial sorting operation; the latter contributes $O(n \log n)$ to the final running time. The outer loop may iterate $O(n)$ times, while the inner two loops may iterate $O(m)$ times each. All the work performed in the inner-most loop may be done in constant time, so the total running time of this implementation is $O(m^2 n + n \log n)$.

It is tempting to use a similar dynamic program to improve the running time of Algorithm 2 (Optimal-$\mathcal{P}_\mathcal{R}$). However, for the SSDUDC algorithm, we have the property that disks in the solution are each *active* (i.e. the lowest upper disk or highest lower disk [2]) on a contiguous set of points in the sorted set $\mathcal{P}$. This property does not hold for our formulation of the $\mathcal{P}_\mathcal{R}$ covering problem when $h \leq 4/5$ (refer to Section 2.2 for definitions); there may exist a sequence of points $\{p_i, p_{i+1}, p_{i+2}\} \in \mathcal{P}$ and mutually spanning sets $\mathcal{D}_j, \mathcal{D}_k$ in the optimal solution, where $p_i, p_{i+2} \in \mathcal{D}_j$ and $p_{i+1} \in \mathcal{D}_k$. Therefore, a dynamic program which iteratively determines the optimal cover for the points of $\mathcal{P}$ is not necessarily correct for the $\mathcal{P}_\mathcal{R}$ covering problem.