Improved Approximation Algorithms for Relay Placement

ALON EFRAT
Department of Computer Science, University of Arizona

SÁNDOR P. FEKETE
Department of Computer Science, Braunschweig University of Technology

JOSEPH S. B. MITCHELL
Department of Applied Mathematics and Statistics, Stony Brook University

VALENTIN POLISHCHUK
Communications and Transport Systems, Linköping University
Helsinki Institute for Information Technology HIIT
Department of Computer Science, University of Helsinki

JUKKA SUOMELA
Helsinki Institute for Information Technology HIIT,
Department of Computer Science, Aalto University

Abstract. In the relay placement problem the input is a set of sensors and a number $r \geq 1$, the communication range of a relay. In the *one-tier* version of the problem the objective is to place a minimum number of relays so that between every pair of sensors there is a path *through sensors and/or relays* such that the consecutive vertices of the path are within distance r if both vertices are relays and within distance 1 otherwise. The *two-tier* version adds the restrictions that the path must go *through relays, and not through sensors*. We present a 3.11-approximation algorithm for the one-tier version and a PTAS for the two-tier version. We also show that the one-tier version admits no PTAS, assuming P $\neq$ NP.

1 Introduction

A sensor network consists of a large number of low-cost autonomous devices, called *sensors*. Communication between the sensors is performed by wireless radio with very limited range, e.g., via the Bluetooth protocol. To make the network connected, a number of additional devices, called *relays*, must be judiciously placed within the sensor field. Relays are typically more advanced and more expensive than sensors, and, in particular, have a larger communication range. For instance, in addition to a Bluetooth chip, each relay may be equipped with a WLAN transceiver, enabling communication between distant relays. The problem we study in this paper is that of placing a *minimum number* of relays to ensure the connectivity of a sensor network.

Two models of communication have been considered in the literature [4, 7–9, 19, 20, 25, 28]. In both models, a sensor and a relay can communicate if the distance between them is at most 1, and two relays can communicate if the distance between them is at most r, where $r \geq 1$ is a given number. The models differ in whether direct communication between sensors is allowed. In the *one-tier* model two sensors can communicate if the distance between them is at most 1. In the *two-tier* model the sensors do not communicate at all, no matter how close they are. In other words, in the two-tier model the sensors may only link to relays, but not to other sensors.

Formally, the input to the relay placement problem is a set of n sensors, identified with their locations in the plane, and a number $r \geq 1$, the communication range of a relay (by scaling, without loss of generality, the communication range of a sensor is 1). The objective in the *one-tier* relay placement is to place a minimum number of relays so that between every pair of sensors there exists a path, *through sensors and/or relays*, such that the consecutive vertices of the path are within distance r if both vertices are relays, and within distance 1 otherwise. The objective in the *two-tier* relay placement is to place a minimum number of relays so that between every pair of sensors there exists a path *through relays* such that the consecutive vertices of the path are within distance r if both vertices are relays, and within distance 1 if one of the vertices is a sensor and the other is a relay (going directly from a sensor to a sensor is forbidden).

1.1 Previous Work

One-tier relay placement in the special case of $r = 1$ [4, 9] is equivalent to finding a Steiner tree with minimum number of Steiner nodes and bounded edge length – the problem that was studied under the names STP-MSPBEL [18], SMT-MSPBEL [20, 28], MSPT [21], and STP-MSP [7–9, 19, 25]. Lin and Xue [18] proved that the problem is NP-hard and gave a 5-approximation algorithm. Chen et al. [7, 8] showed that the algorithm of Lin and Xue is actually a 4-approximation algorithm, and gave a 3-approximation algorithm; Cheng et al. [9] gave a 3-approximation algorithm with an improved running time, and a randomised 2.5-approximation algorithm. Chen et al. [7, 8] presented a polynomial-time approximation scheme (PTAS) for minimising the *total* number of vertices in the tree (i.e., with the objective function being the number of the original points plus the number of Steiner vertices) for a restricted version of the problem, in which in the minimum spanning tree of the set the length of the longest edge is at most constant times the length of the shortest edge.

For the general case of *arbitrary* $r \geq 1$, the current best approximation ratio for one-tier relay placement is due to Lloyd and Xue [20], who presented a simple 7-approximation algorithm, based on "Steinerising" the minimum spanning tree of the sensors. In this paper we give an algorithm with an improved approximation ratio of 3.11.

Two-tiered relay placement (under the assumptions that the sensors are uniformly distributed in a given region and that $r \geq 4$) was considered by Hao et al. [16] and Tang et al. [26] who suggested constant-factor approximation algorithms for several versions of the problem. Lloyd and Xue [20, Thm. 4.1] and Srinivas et al. [25, Thm. 1] developed a general framework whereby given an α-approximate solution to Disk Cover (finding minimum number of unit disks to cover a given set of points) and a β-approximate solution to STP-MSPBEL (see above), one may find an approximate solution for the two-tier relay placement. In more details, the algorithm in Lloyd and Xue [20] works for arbitrary $r \geq 1$ and has an approximation factor of $2\alpha + \beta$; the algorithm in Srinivas et al. [25] works for $r \geq 2$ and guarantees an $(\alpha + \beta)$-approximate solution. Combined with the best known approximation factors for the Disk Cover [17] and STP-MSPBEL [7–9], these lead to $5 + \varepsilon$ and $4 + \varepsilon$ approximations for the relay placement respectively. In this paper we present a PTAS for the two-tiered relay placement; the PTAS works directly for the relay placement, without combining solutions to other problems.

A different line of research [5, 22] concentrated on a "discrete" version of relay placement, in which the goal is to pick a minimum subset of relays from a *given* set of possible relay locations. In this paper we allow the relays to reside anywhere in the plane.

1.2 Contributions

We present new results on approximability of relay placement:

- In Section 3 we give a simple $O(n \log n)$-time 6.73-approximation algorithm for the one-tier version.
- In Section 4 we present a polynomial-time 3.11-approximation algorithm for the one-tier version.
- In Section 5 we show that there is no PTAS for one-tier relay placement (assuming that r is part of the input, and P $\neq$ NP).
- In Section 6 we give a PTAS for two-tier relay placement.

Note that the *number* of relays in a solution may be exponential in the size of the input (number of bits). Our algorithms produce a succinct representation of the solution. The representation is given by a set of points and a set of line segments; the relays are placed on each point and equally-spaced along each segment.

2 Blobs, Clouds, Stabs, Hubs, and Forests

In this section we introduce the notions, central to the description of our algorithms for one-tier relay placement. We also provide lower bounds.

2.1 Blobs and Clouds

We write $|xy|$ for the Euclidean distance between x and y. Let V be a given set of sensors (points in the plane). We form a unit disk graph $\mathcal{G} = (V, E)$ and a disk graph $\mathcal{F} = (V, F)$ where

$$E = \big\{\{u, v\} : |uv| \leq 1\big\},$$
$$F = \big\{\{u, v\} : |uv| \leq 2\big\};$$

see Figure 1a.

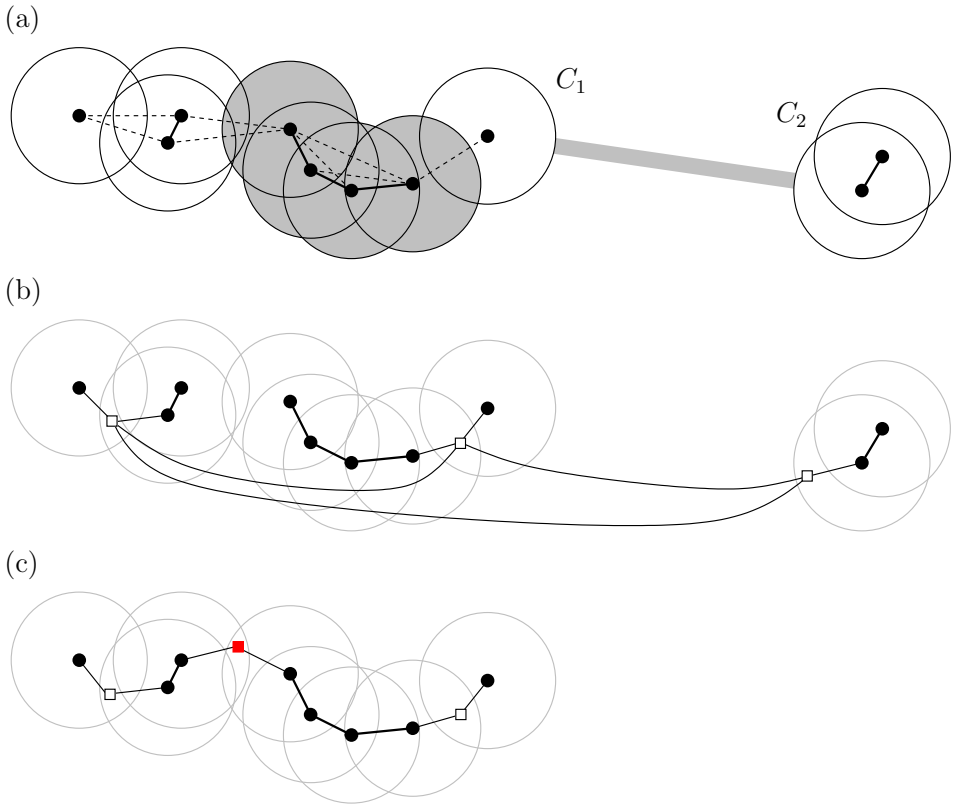

Figure 1: (a) Dots are sensors in V, solid lines are edges in E and F, and dashed lines are edges in F only. There are 5 blobs in $\mathcal{B}$ (one of them highlighted) and 2 clouds $C_1, C_2 \in \mathcal{C}$. The wide grey line is the only edge in MStFN($\mathcal{C}$), which happens to be equal to MSFN($\mathcal{C}$) here. (b) Stabs. (c) Hubs.

A *blob* is defined to be the union of the unit disks centered at the sensors that belong to the same connected component of $\mathcal{G}$. We use B to refer to a blob, and $\mathcal{B}$ for the set of all blobs.

Analogously, a *cloud* $C \in \mathcal{C}$ is the union of the unit disks centered at the sensors that belong to the connected component of the graph $\mathcal{F}$. The sensors in a blob can communicate with each other without relays, while the ones in a cloud might not, even though their disks may overlap. Each cloud $C \in \mathcal{C}$ consists of one or more blobs $B \in \mathcal{B}$; we use $\mathcal{B}_C$ to denote the blobs that form the cloud C.

2.2 Stabs and Hubs

A *stab* is a relay with an infinite communication range ($r = \infty$). A *hub* is a relay without the ability to communicate with the other relays (thus hubs can enable communication within one cloud, but are of no use in communicating between clouds). As we shall see, a solution to stab or hub placement can be used as the first step towards a solution for relay placement.

If we are placing stabs, it is necessary and sufficient to have a stab in each blob to ensure communication between all sensors (to avoid trivialities we assume there is more than one blob). Thus, stab placement is a special case of the set cover problem: the universe is the blobs, and the subsets are sets of blobs that have a point in common. We use Stab($\mathcal{B}'$) to denote the minimum set of stabs that stab each blob in $\mathcal{B}' \subseteq \mathcal{B}$. In the

example in Figure 1b small rectangles show an optimal solution to the stab placement problem; 3 stabs are enough.

If we are placing hubs, it is necessary (assuming more than one blob in the cloud), but not sufficient, to have a hub in each blob to ensure communication between sensors within one cloud. In fact, hub placement can be interpreted as a special case of the *connected* set cover problem [6, 24]. In the example in Figure 1c small rectangles show an optimal solution to the hub placement problem for the cloud $C = C_1$; in this particular case, 2 stabs within the cloud C were sufficient to "pierce" each blob in $\mathcal{B}_C$ (see Figure 1b), however, an additional hub (marked red in Figure 1c) is required to "stitch" the blobs together (i.e., to establish communication between the blobs). The next lemma shows that, in general, the number of additional hubs needed is less than the number of stabs:

Lemma 1. *Given a feasible solution S to stab placement on $\mathcal{B}_C$, we can obtain in polynomial time a feasible solution to hub placement on $\mathcal{B}_C$ with $2|S| - 1$ hubs.*

Proof. Let $\mathcal{H}$ be the graph, whose nodes are the sensors in the cloud C and the stabs in S, and whose edges connect two devices if either they are within distance 1 from each other or if both devices are stabs (i.e., there is an edge between *every* pair of the stabs). Switch off communication between the stabs, thus turning them into hubs. Suppose that this breaks $\mathcal{H}$ into k connected components. There must be a stab in each connected component. Thus, $|S| \geq k$.

If $k > 1$, by the definition of a cloud, there must exist a point where a unit disk covers at least two sensors from two different connected components of $\mathcal{H}$. Placing a hub at the point decreases the number of the connected components by at least 1. Thus, after putting at most $k - 1$ additional hubs, all connected components will merge into one. □

2.3 Steiner Forests and Spanning Forests with Neighbourhoods

Let $\mathcal{P}$ be a collection of planar subsets; call them *neighbourhoods*. (In Section 3 the neighbourhoods will be the clouds, in Section 4 they will be "clusters" of clouds.) For a plane graph G, let $\mathcal{G}_{\mathcal{P}} = (\mathcal{P}, E(G))$ be the graph whose vertices are the neighbourhoods and two neighbourhoods $P_1, P_2 \in \mathcal{P}$ are adjacent whenever G has a vertex in P_1, a vertex in P_2, and a path between the vertices.

The *Minimum Steiner Forest with Neighbourhoods* on $\mathcal{P}$, denoted MStFN($\mathcal{P}$), is a *minimum-length* plane graph G such that $\mathcal{G}_{\mathcal{P}} = (\mathcal{P}, E(G))$ is connected. The MStFN is a generalisation of the Steiner tree of a set of points. Note that MStFN is slightly different from Steiner tree with neighbourhoods (see, e.g., Yang et al. [27]) in that we are only counting the part of the graph *outside* $\mathcal{P}$ towards its length (since it is not necessary to connect neighbourhoods beyond their boundaries).

Consider a complete weighted graph whose vertices are the neighbourhoods in $\mathcal{P}$ and whose edge weights are the distances between them. A minimum spanning tree in the graph is called the *Minimum Spanning Forest with Neighbourhoods* on $\mathcal{P}$, denoted MSFN($\mathcal{P}$). A natural embedding of the edges of the forest is by the straight-line segments that connect the corresponding neighbourhoods; we will identify MSFN($\mathcal{P}$) with the embedding. (As with MStFN, we count the length of MSFN only *outside* $\mathcal{P}$.)

We denote by $|\text{MStFN}(\mathcal{P})|$ and $|\text{MSFN}(\mathcal{P})|$ the total length of the edges of the forests. It is known that

$$|\text{MSFN}(P)| \leq \frac{2}{\sqrt{3}}|\text{MStFN}(P)|$$

for a *point* set P, where $2/\sqrt{3}$ is the *Steiner ratio* [13]. The following lemma generalises this to neighbourhoods.

Lemma 2. *For any $\mathcal{P}$, $|\mathrm{MSFN}(\mathcal{P})| \leq (2/\sqrt{3})|\mathrm{MStFN}(\mathcal{P})|$.*

Proof. If $\mathcal{P}$ is erased, $\mathrm{MStFN}(\mathcal{P})$ falls off into a forest, each tree of which is a minimum Steiner tree on its leaves; its length is within the Steiner ratio of minimum spanning tree length. □

2.4 Lower Bounds on the Number of Relays

Let R^* be an optimal set of relays. Let $\mathcal{R}$ be the communication graph on the relays R^* alone, i.e., without sensors taken into account; two relays are connected by an edge in $\mathcal{R}$ if and only if they are within distance r from each other. Suppose that $\mathcal{R}$ is embedded in the plane with vertices at relays and line segments joining communicating relays. The embedding spans all clouds, for otherwise the sensors in a cloud would not be connected to the others. Thus, in $\mathcal{R}$ there exists a forest $\mathcal{R}'$, whose embedding also spans all clouds. Let $|\mathcal{R}'|$ denote the total length of the edges in $\mathcal{R}'$. By definition of $\mathrm{MStFN}(\mathcal{C})$, we have $|\mathcal{R}'| \geq |\mathrm{MStFN}(\mathcal{C})|$.

Let m, v, and k be the number of edges, vertices, and trees of $\mathcal{R}'$. Since each edge of $\mathcal{R}'$ has length at most r, we have $|\mathcal{R}'| \leq mr = (v-k)r$. Since $v \leq |R^*|$, since there must be a relay in every blob and every cloud, and since the clouds are disjoint, it follows that

$$|R^*| \geq |\mathrm{MStFN}(\mathcal{C})|/r, \tag{1}$$

$$|R^*| \geq |\mathrm{Stab}(\mathcal{B})|, \tag{2}$$

$$|R^*| \geq |\mathcal{C}|. \tag{3}$$

3 A 6.73-Approximation Algorithm for One-Tier Relay Placement

In this section we give a simple 6.73-approximation algorithm for relay placement. We first find an approximately optimal stab placement. Then we turn a stab placement into a hub placement within each cloud. Then a spanning tree on the clouds is found and "Steinerised".

Finding an optimal stab placement is a special case of the set cover problem. The maximum number of blobs pierced by a single stab is 5 (since this is the maximum number of unit disks that can have non-empty intersection while avoiding each other's centers). Thus, in this case the greedy heuristic for the set cover has an approximation ratio of $1 + 1/2 + 1/3 + 1/4 + 1/5 = 137/60$ [10, Theorem 35.4].

Based on this approximation, a feasible hub placement R_C within one cloud $C \in \mathcal{C}$ can be obtained by applying Lemma 1; for this set of hubs it holds that

$$|R_C| \;\leq\; \frac{137}{30}|\mathrm{Stab}(\mathcal{B}_C)| - 1.$$

We can now interpret hubs R_C as relays; if the hubs make the cloud C connected, surely it holds for relays.

Let $R' = \bigcup_C R_C$ denote all relays placed this way. Since the blobs $\mathcal{B}_C$ for different C do not intersect, $|\mathrm{Stab}(\mathcal{B})| = \sum_C |\mathrm{Stab}(\mathcal{B}_C)|$, so

$$|R'| \;\leq\; \sum_C |R_C| \;\leq\; \sum_C \left(\frac{137}{30}|\mathrm{Stab}(\mathcal{B}_C)| - 1\right) \;=\; \frac{137}{30}|\mathrm{Stab}(\mathcal{B})| - |\mathcal{C}|. \tag{4}$$

Next, we find $\mathrm{MSFN}(\mathcal{C})$ and place another set of relays, R'', along its edges. Specifically, for each edge e of the forest, we place 2 relays at the endpoints of e, and $\lfloor |e|/r \rfloor$ relays

every r units starting from one of the endpoints. This ensures that all clouds communicate with each other; thus $R = R' \cup R''$ is a feasible solution. Since the number of edges in MSFN($\mathcal{C}$) is $|\mathcal{C}| - 1$,

$$|R''| \;=\; 2(|\mathcal{C}| - 1) + \sum_e \left\lfloor \frac{|e|}{r} \right\rfloor \;<\; 2|\mathcal{C}| + \frac{|\mathrm{MSFN}(\mathcal{C})|}{r}. \tag{5}$$

We obtain

$$|R| \;=\; |R'| + |R''| \;\leq\; \left(\frac{137}{30} + 1 + \frac{2}{\sqrt{3}}\right)|R^*| \;<\; 6.73|R^*|$$

from (1)–(5) and Lemma 2.

3.1 Running Time

To implement the above algorithm in $O(n \log n)$ time, we construct the blobs (this can be done in $O(n \log n)$ time since the blobs are the union of disks centered on the sensors), assign each blob a unique colour, and initialise a Union-Find data structure for the colours. Next, we build the arrangement of the blobs, and sweep the arrangement 4 times, once for each $d = 5, 4, 3, 2$; upon encountering a d-coloured cell of the arrangement, we place the stab anywhere in the cell, merge the corresponding d colours, and continue. Finally, to place the hubs we do one additional sweep.

As for the last step – building MSFN($\mathcal{C}$) – it is easy to see that just as the "usual" minimum spanning tree of a set of points, MSFN($\mathcal{C}$) uses only the edges of the relative neighbourhood graph of the sensors (refer, e.g., to de Berg et al. [11, p. 217] for the definition of the graph). Indeed, let pq be an edge of MSFN($\mathcal{C}$); let p' and q' be the sensors that are within distance 1 of p and q, respectively. If there existed a sensor s' closer than $|p'q'|$ to both p' and q', the edge pq could have been swapped for a shorter edge (Figure 2).

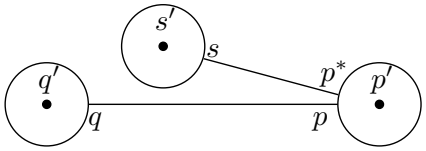

Figure 2: Edge pq could be swapped for a shorter edge.

It remains to show how to build and sweep the arrangement of blobs in $O(n \log n)$ time. Since the blobs are unions of unit disks, their total complexity is linear (see, e.g., de Berg et al. [11, Theorem 13.9]). Moreover, the arrangement of the blobs also has only linear complexity (see Lemma 3 below); this follows from the fact that every point can belong to only a constant number of blobs (at most 5). Thus, we can use sweep to build the arrangement in $O(n \log n)$ time, and also, of course, sweep the arrangement within the same time bound.

Lemma 3. *The arrangement of the blobs has linear complexity.*

Proof. The vertices of the arrangement are of two types – the vertices of the blobs themselves and the vertices of the intersection of two blobs (we assume that no three blobs intersect in a single point). The total number of the vertices of the first type is linear, so we focus on the vertices of the second type.

Let A be a tile in the infinite unit-square tiling of the plane. There is not more than a constant number K of blobs that intersect A (since there is not more than a constant

number of points that can be placed within distance 1 from A so that the distance between any two of the points is larger than 1). Let n_i be the number of disks from blob i that intersect A. Every vertex of the arrangement inside A is on the boundary of the union of some two blobs. Because the union of blobs has linear complexity, the number of vertices that are due to intersection of blobs i and j is $O(n_i + n_j)$. Since there is at most K blobs for which $n_i \neq 0$, we have

$$\sum_{i,j}(n_i + n_j) \leq \binom{K}{2} n(A),$$

where $n(A)$ is the total number of disks intersecting A. Clearly, each unit disk intersects only a constant number of the unit-square tiles, and only a linear number of tiles is intersected by the blobs. Thus, summing over all tiles, we obtain that the total complexity of the arrangement is $O(K^2 n) = O(n)$. □

4 A 3.11-Approximation Algorithm for One-Tier Relay Placement

In this section we first take care of clouds whose blobs can be stabbed with few relays, and then find an approximation to the hub placement by greedily placing the hubs themselves, without placing the stabs first, for the rest of the clouds. Together with a refined analysis, this gives a polynomial-time 3.11-approximation algorithm. We focus on nontrivial instances with more than one blob.

4.1 Overview

The basic steps of our algorithm are as follows:

(1) Compute optimal stabbings for the clouds that can be stabbed with few relays.
(2) Connect the blobs in each of these clouds, using Lemma 1.
(3) Greedily connect all blobs in each of the remaining clouds ("stitching").
(4) Greedily connect clouds into clusters, using 2 additional relays per cloud.
(5) Connect the clusters by a spanning forest.

Our algorithm constructs a set A_r of "red" relays (for connecting blobs in a cloud, i.e., relays added in steps 1–3), a set A_g of "green" relays (two per cloud, added in steps 4–5) and a set A_y of "yellow" relays (outside of sensor range, added in step 5). Refer to Figures 3 and 4. In the analysis, we compare an optimal solution R^* to our approximate one by subdividing the former into a set R_d^* of "dark" relays that are within reach of sensors, and into a set R_ℓ^* of "light" relays that are outside of sensor range. We compare $|R_d^*|$ with $|A_r| + |A_g|$, and $|R_\ell^*|$ with $|A_y|$, showing in both cases that the ratio is less than 3.11.

4.2 Clouds with Few Stabs

For any constant k, it is straightforward to check in polynomial time whether all blobs in a cloud $C \in \mathcal{C}$ can be stabbed with $i < k$ stabs. (For any subset of i cells of the arrangement of unit disks centered on the sensors in C, we can consider placing the relays in the cells and check whether this stabs all blobs.) Using Lemma 1, we can connect all blobs in such a cloud with at most $2i - 1$ red relays. We denote by $\mathcal{C}^i$ the set of clouds where the minimum number of stabs is i, and by $\mathcal{C}^{k+}$ the set of clouds that need at least k stabs.

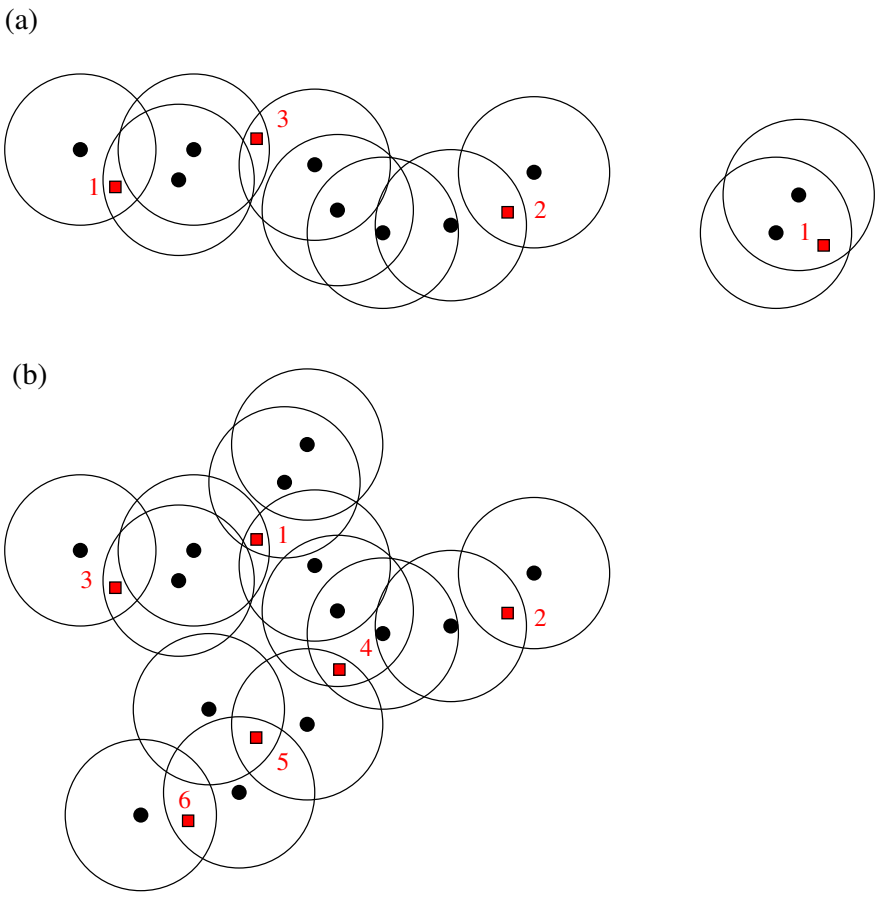

Figure 3: Red relays placed by our algorithm (the sensors are the solid circles); the numbers indicate the order in which the relays are placed within each cloud. (a) Stab the clouds that can be stabbed by placing few relays; the clouds are then stitched by placing the hubs as in Lemma 1. (b) Greedily stitch the other clouds.

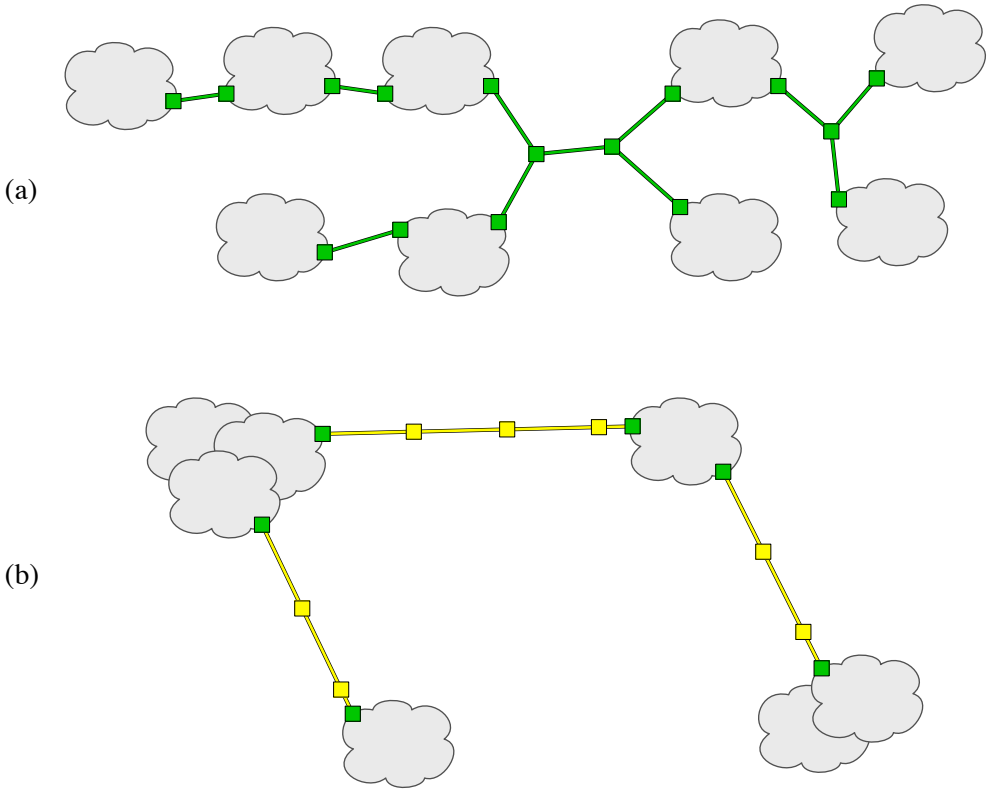

Figure 4: (a) Green relays connect clouds into clusters – on average, we use at most 2 green relays per cloud. (b) Green (inside clouds) and yellow (outside clouds) relays interconnect the cloud clusters by a spanning tree.

4.3 Stitching a Cloud from $\mathcal{C}^{k+}$

We focus on one cloud $C \in \mathcal{C}^{k+}$. For a point y in the plane, let

$$\mathcal{B}(y) = \{B \in \mathcal{B}_C : y \in B\}$$

be the set of blobs that contain the point; obviously $|\mathcal{B}(y)| \leq 5$ for any y. For any subset of blobs $\mathcal{T} \subseteq \mathcal{B}_C$, define $\mathcal{S}(\mathcal{T}, y) = \mathcal{B}(y) \setminus \mathcal{T}$ to be the set of blobs *not from* $\mathcal{T}$ containing y, and define $V(\mathcal{T})$ to be the set of sensors that form the blobs in $\mathcal{T}$.

Within C, we place a set of red relays $A_r^C = \{y_j : j = 1, 2, \dots\}$, as follows:

(1) Choose arbitrary $B_0 \in \mathcal{B}_C$.

(2) Initialise $j \leftarrow 1$, $\mathcal{T}_j \leftarrow \{B_0\}$.

(3) While $\mathcal{T}_j \neq \mathcal{B}_C$:

$$\begin{aligned} y_j &\leftarrow \arg\max_y \{|\mathcal{S}(\mathcal{T}_j, y)| : \mathcal{B}(y) \cap \mathcal{T}_j \neq \emptyset\}, \\ \mathcal{S}_j &\leftarrow \mathcal{S}(\mathcal{T}_j, y_j), \\ \mathcal{T}_{j+1} &\leftarrow \mathcal{T}_j \cup \mathcal{S}_j, \\ j &\leftarrow j + 1. \end{aligned}$$

That is, y_j is a point contained in a maximum number of blobs *not from* $\mathcal{T}_j$ that intersect a blob from $\mathcal{T}_j$. In other words, we stitch the clouds greedily; the difference from the usual greedy (used in the previous section) is that we insist that *some* blob stabbed by y_j is already in $\mathcal{T}_j$.

By induction on j, after each iteration, there exists a path through sensors and/or relays between any pair of sensors in $V(\mathcal{T}_j)$. By the definition of a cloud, there is a line segment of length at most 2 that connects $V(\mathcal{T}_j)$ to $V(\mathcal{B}_C \setminus \mathcal{T}_j)$; the midpoint of the segment is a location y with $\mathcal{S}(\mathcal{T}_j, y) \neq \emptyset$. Since each iteration increases the size of $\mathcal{T}_j$ by at least 1, the algorithm terminates in at most $|\mathcal{B}_C| - 1$ iterations, and $|A_r^C| \leq |\mathcal{B}_C| - 1$. The sets $\mathcal{S}_j$ form a partition of $\mathcal{B}_C \setminus \{B_0\}$.

We prove the following performance guarantee (the proof is similar to the analysis of greedy set cover.)

Lemma 4. *For each cloud C we have $|A_r^C| \leq 37|R_d^* \cap C|/12 - 1$.*

Proof. For each $B \in \mathcal{B}_C \setminus \{B_0\}$, define the weight $w(B) = 1/|\mathcal{S}_j|$, where $\mathcal{S}_j$ is the unique set for which $B \in \mathcal{S}_j$. We also set $w(B_0) = 1$. We have

$$\sum_{B \in \mathcal{B}_C} w(B) \;=\; |A_r^C| + 1. \tag{6}$$

Consider a relay $z \in R_d^* \cap C$, and find the smallest ℓ with $\mathcal{T}_\ell \cap \mathcal{B}(z) \neq \emptyset$, that is, $\ell = 1$ if $B_0 \in \mathcal{B}(z)$, and otherwise $y_{\ell-1}$ is the first relay that pierced a blob from $\mathcal{B}(z)$. Partition the set $\mathcal{B}(z)$ into $\mathcal{U}(z) = \mathcal{T}_\ell \cap \mathcal{B}(z)$ and $\mathcal{V}(z) = \mathcal{B}(z) \setminus \mathcal{U}(z)$. Note that $\mathcal{V}(z)$ may be empty, e.g., if $y_{\ell-1} = z$.

First, we show that

$$\sum_{B \in \mathcal{U}(z)} w(B) \;\leq\; 1.$$

We need to consider two cases. It may happen that $\ell = 1$, which means that $B_0 \in \mathcal{B}(z)$ and $\mathcal{U}(z) = \{B_0\}$. Then the total weight assigned to the blobs in $\mathcal{U}(z)$ is, by definition, 1. Otherwise $\ell > 1$ and $\mathcal{U}(z) \subseteq S_{\ell-1}$, implying $w(B) = 1/|S_{\ell-1}| \leq 1/|\mathcal{U}(z)|$ for each $B \in \mathcal{U}(z)$.

Second, we show that

$$\sum_{B\in\mathcal{V}(z)} w(B) \;\le\; \frac{1}{|\mathcal{V}(z)|} + \frac{1}{|\mathcal{V}(z)|-1} + \cdots + \frac{1}{1}.$$

Indeed, at iterations $j \ge \ell$, the algorithm is able to consider placing the relay y_j at the location z. Therefore $|\mathcal{S}_j| \ge |\mathcal{S}(\mathcal{T}_j, z)|$. Furthermore,

$$\mathcal{S}(\mathcal{T}_j, z) \setminus \mathcal{S}(\mathcal{T}_{j+1}, z) \;=\; \mathcal{B}(z) \cap \mathcal{S}_j \;=\; \mathcal{V}(z) \cap \mathcal{S}_j.$$

Whenever placing the relay y_j makes $|\mathcal{S}(\mathcal{T}_j, z)|$ decrease by a number a, exactly a blobs of $\mathcal{V}(z)$ get connected to $\mathcal{T}_j$. Each of them is assigned the weight $w(C) \le 1/|\mathcal{S}(\mathcal{T}_j, z)|$. Thus,

$$\sum_{B\in\mathcal{V}(z)} w(B) \;\le\; \frac{a_1}{a_1 + a_2 + \cdots + a_n} + \frac{a_2}{a_2 + a_3 + \cdots + a_n} + \cdots + \frac{a_n}{a_n},$$

where $a_1, a_2, \ldots, a_n$ are the number of blobs from $\mathcal{V}(z)$ that are pierced at different iterations, $\sum_i a_i = |\mathcal{V}(z)|$. The maximum value of the sum is attained when $a_1 = a_2 = \cdots = a_n = 1$ (i.e., every time $|\mathcal{V}(z)|$ is decreased by 1, and there are $|\mathcal{V}(z)|$ summands).

Finally, since $|\mathcal{B}(z)| \le 5$, and $\mathcal{U}(z) \ne \emptyset$, we have $|\mathcal{V}(z)| \le 4$. Thus,

$$W(z) \;=\; \sum_{B\in\mathcal{U}(z)} w(B) \;+\; \sum_{B\in\mathcal{V}(z)} w(B) \;\le\; 1 + \frac{1}{4} + \frac{1}{3} + \frac{1}{2} + \frac{1}{1} \;=\; \frac{37}{12}. \tag{7}$$

The sets $\mathcal{B}(z)$, $z \in R_d^* \cap C$, form a cover of $\mathcal{B}_C$. Therefore, from (6) and (7),

$$\frac{37}{12}|R_d^* \cap C| \;\ge\; \sum_{z\in R_d^*\cap C} W(z) \;\ge\; \sum_{B\in\mathcal{B}_C} w(B) \;=\; |A_r^C| + 1. \qquad \square$$

4.4 Green Relays and Cloud Clusters

At any stage of the algorithm, we say that a set of clouds is *interconnected* if, with the current placement of relays, the sensors in the clouds can communicate with each other. Now, when all clouds have been stitched (so that the sensors within any one cloud can communicate), we proceed to interconnecting the clouds. First we greedily form the collection of cloud *clusters* (interconnected clouds) as follows. We start by assigning each cloud to its own cluster. Whenever it is possible to interconnect two clusters by placing one relay within each of the two clusters, we do so. These two relays are coloured green. After it is no longer possible to interconnect 2 clusters by placing just 2 relays, we repeatedly place 4 green relays wherever we can use them to interconnect clouds from 3 different clusters. Finally, we repeat this for 6 green relays that interconnect 4 clusters.

On average we place 2 green relays every time the number of connected components in the communication graph on sensors plus relays decreases by one.

4.5 Interconnecting the Clusters

Now, when the sensors in each cloud and the clouds in each cluster are interconnected, we interconnect the clusters by a minimum Steiner forest with neighbourhoods. The forest is slightly different from the one used in the previous section. This time we are minimising the total number of relays that need to be placed along the edges of the forest in order to interconnect the clusters; we denote this forest by MSFN$'$. The forest can be found by

assigning appropriate weights to the edges of the graph on the clusters – the weight of an edge is the number of relays that are necessary to interconnect two clusters.

After MSFN$'$ is found, we place relays along edges of the forest just as we did in the simple algorithm from the previous section. This time though we assign colours to the relays. Specifically, for each edge e of the forest, we place 2 green relays at the endpoints of e, and $\lfloor |e|/r \rfloor$ yellow relays every r units starting from one of the endpoints. As with interconnecting clouds into the clusters, when interconnecting the clusters we use 2 green relays each time the number of connected components of the communication graph decreases by one. Thus, overall, we use at most $2|\mathcal{C}| - 2$ green relays.

4.6 Analysis: Red and Green Relays

Recall that for $i < k$, $\mathcal{C}^i$ is the class of clouds that require precisely i relays for stabbing, and $\mathcal{C}^{k+}$ is the class of clouds that need at least k relays for stabbing. An optimal solution R^* therefore contains at least

$$|R_d^*| \geq k|\mathcal{C}^{k+}| + \sum_{i=1}^{k-1} i|\mathcal{C}^i|$$

dark relays (relays inside clouds, i.e., relays within reach of sensors). Furthermore, $|R_d^* \cap C| \geq 1$ for all C.

Our algorithm places at most $2i - 1$ red relays per cloud in $\mathcal{C}^i$, and not more than $37|R_d^* \cap C|/12 - 1$ red relays per cloud in $\mathcal{C}^{k+}$. Adding a total of $2|\mathcal{C}| - 2$ green relays used for clouds interconnections, we get

$$\begin{aligned}
|A_r| + |A_g| &\leq \sum_{C \in \mathcal{C}^{k+}} \left(\frac{37}{12} |R_d^* \cap C| - 1 \right) + \sum_{i=1}^{k-1} (2i-1)|\mathcal{C}^i| + 2|\mathcal{C}| - 2 \\
&\leq \frac{37}{12} \left(|R_d^*| - \sum_{i=1}^{k-1} i|\mathcal{C}^i| \right) + |\mathcal{C}^{k+}| + \sum_{i=1}^{k-1} (2i+1)|\mathcal{C}^i| - 2 \\
&\leq \frac{37}{12} |R_d^*| + |\mathcal{C}^{k+}| < \left(3.084 + \frac{1}{k} \right) |R_d^*|.
\end{aligned}$$

4.7 Analysis: Yellow Relays

As in Section 2.4, let $\mathcal{R}$ be the communication graph on the optimal set R^* of relays (without sensors). In $\mathcal{R}$ there exists a forest $\mathcal{R}'$ that makes the clusters interconnected. Let $R' \subset R^*$ be the relays that are vertices of $\mathcal{R}'$. We partition R' into "black" relays $R_b^* = R' \cap R_d^*$ and "white" relays $R_w^* = R' \cap R_\ell^*$ – those inside and outside the clusters, respectively.

Two black relays cannot be adjacent in $\mathcal{R}'$: if they are in the same cluster, the edge between them is redundant; if they are in different clusters, the distance between them must be larger than r, as otherwise our algorithm would have placed two green relays to interconnect the clusters into one. By a similar reasoning, there cannot be a white relay adjacent to 3 or more black relays in $\mathcal{R}'$, and there cannot be a pair of adjacent white relays such that each of them is adjacent to 2 black relays. Refer to Figure 5. Finally, the maximum degree of a white relay is 5. Using these observations, we can prove the following lemma.

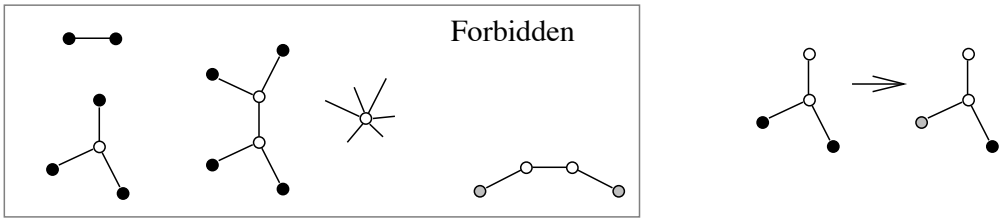

Figure 5: Forbidden configurations and grey relays.

Lemma 5. *There is a spanning forest with neighbourhoods on cloud clusters that requires at most*

$$\left(\frac{4}{\sqrt{3}} + \frac{4}{5}\right)|R_w^*| < 3.11|R_w^*|$$

yellow relays on its edges.

Proof. Let $\mathcal{D}$ be the set of cloud clusters. We partition $\mathcal{R}'$ into edge-disjoint trees induced by maximal connected subsets of white relays and their adjacent black relays. It is enough to show that for each such tree T that interconnects a subset of clusters $\mathcal{D}' \subseteq \mathcal{D}$, there is a spanning forest on $\mathcal{D}'$ such that the number of yellow relays on its edges is at most 3.11 times the number of white relays in T. As no pair of black relays is adjacent in $\mathcal{R}'$, these edge-disjoint trees interconnect all clusters in $\mathcal{D}$. The same holds for the spanning forests, and the lemma follows.

Trees with only one white relay (and thus exactly two black relays) are trivial: the spanning forest needs only one edge with one yellow relay (and one green in each end). Therefore assume that T contains at least two white relays.

We introduce yet another colour. For each white relay with two black neighbours, arbitrarily choose one of the black relays and change it into a "grey" relay (Figure 5). Let w be the number of white relays, let b be the number of remaining black relays, and let g be the number of grey relays in T.

First, we clearly have

$$b \leq w. \tag{8}$$

Second, there is no grey–white–white–grey path, and each white relay is adjacent to another white relay. Therefore the ratio $(b+g)/w$ is at most $9/5$. To see this, let w_2 be the number of white relays with a grey and a black neighbour, let w_1 be the number of white relays with a black neighbour but no grey neighbour, and let w_0 be the number of white relays without a black neighbour. By degree bound, $w_2 \leq 4w_1 + 5w_0 = 4w_1 + 5(w - w_2 - w_1)$; therefore $5w \geq 6w_2 + w_1$. We also know that $w \geq w_2 + w_1$. Therefore

$$\frac{9}{5}w \;\geq\; \frac{1}{5}(6w_2 + w_1) + \frac{4}{5}(w_2 + w_1) \;=\; (w_2 + w_1) + w_2 \;=\; b + g. \tag{9}$$

(The worst case is a star of $1 + 4$ white relays, 5 black relays and 4 grey relays.)

Now consider the subtree induced by the black and white relays. It has fewer than $b + w$ edges, and the edge length is at most r. By Lemma 2, there is a spanning forest on the black relays with total length less than $(2/\sqrt{3})(b + w)r$; thus we need fewer than $(2/\sqrt{3})(b + w)$ yellow relays on the edges.

Now each pair of black relays in T is connected. It is enough to connect each grey relay to the nearest black relay: the distance is at most 2, and one yellow relay is enough. In

summary, the total number of yellow relays is less than

$$\frac{2}{\sqrt{3}}(b+w)+g = \left(\frac{2}{\sqrt{3}}-1\right)(b+w)+b+g+w$$
$$\leq \left(\frac{2}{\sqrt{3}}-1\right)2w+\frac{9}{5}w+w = \left(\frac{4}{\sqrt{3}}+\frac{4}{5}\right)w < 3.11w.$$

The inequality follows from (8) and (9). □

Thus, $|A_y| < 3.11|R_w^*| \leq 3.11|R_\ell^*|$, and the overall approximation ratio of our algorithm is less than 3.11.

5 Inapproximability of One-Tier Relay Placement

We have improved the best known approximation ratio for one-tier relay placement from 7 to 3.11. A natural question to pose at this point is whether we could make the approximation ratio as close to 1 as we wish. In this section, we show that no PTAS exists, unless P = NP.

Theorem 1. *It is NP-hard to approximate one-tier relay placement within factor* $1+1/687$.

The reduction is from minimum vertex cover in graphs of bounded degree. Let $\mathcal{G} = (V, E)$ be an instance of vertex cover; let $\Delta \leq 5$ be the maximum degree of $\mathcal{G}$. We construct an instance $\mathcal{I}$ of the relay placement problem that has a feasible solution with $k + 2|E| + 1$ relays if and only if $\mathcal{G}$ has a vertex cover of size k.

Figure 6 illustrates the construction. Figure 6a shows the *vertex gadget*; we have one such gadget for each vertex $v \in V$. Figure 6b shows the *crossover gadget*; we have one such gadget for each edge $e \in E$. Small dots are sensors in the relay placement instance; each solid edge has length at most 1. White boxes are *good locations* for relays; there is one good location in each vertex gadget, and two good locations per crossover gadget. Dashed line shows a connection for relays in good locations in a crossover.

We set $r = 16(|V| + 1)$, and we choose $|E| + 1$ disks of diameter r such that each pair of these disks is separated by a distance larger than $|V|r$ but at most poly$(|V|)$. One of the disks is called $S(0)$ and the rest are $S(e)$ for $e \in E$. All vertex gadgets and one isolated sensor, called p_0, are placed within disk $S(0)$. The crossover gadget for edge e is placed within disk $S(e)$. There are noncrossing paths of sensors that connect the crossover gadget $e = \{u, v\} \in E$ to the vertex gadgets u and v; all such paths (*tentacles*) are separated by a distance at least 3. Good relay locations and p_0 cannot be closer than 1 unit to a disk boundary.

Figure 6c is a schematic illustration of the overall construction in the case of $\mathcal{G} = K_5$; the figure is highly condensed in x direction. There are 11 disks. Disk $S(0)$ contains one isolated sensor and 5 vertex gadgets. Each disk $S(e)$ contains one crossover gadget. Outside these disks we have only parts of tentacles.

There are $4|E| + 1$ blobs in $\mathcal{I}$. The isolated sensor p_0 forms one blob. For each edge there are 4 blobs: two tentacles from vertex gadgets to the crossover gadget, and two isolated sensors in the crossover gadget.

Theorem 1 now follows from the following two lemmata.

Lemma 6. *Let C be a vertex cover of $\mathcal{G}$. Then there is a feasible solution to relay placement problem $\mathcal{I}$ with $|C| + 2|E| + 1$ relays.*

Proof. For each $v \in C$, place one relay at the good location of the vertex gadget v. For each $e \in E$, place two relays at the good locations of the crossover gadget e. Place one relay at the isolated sensor p_0. □

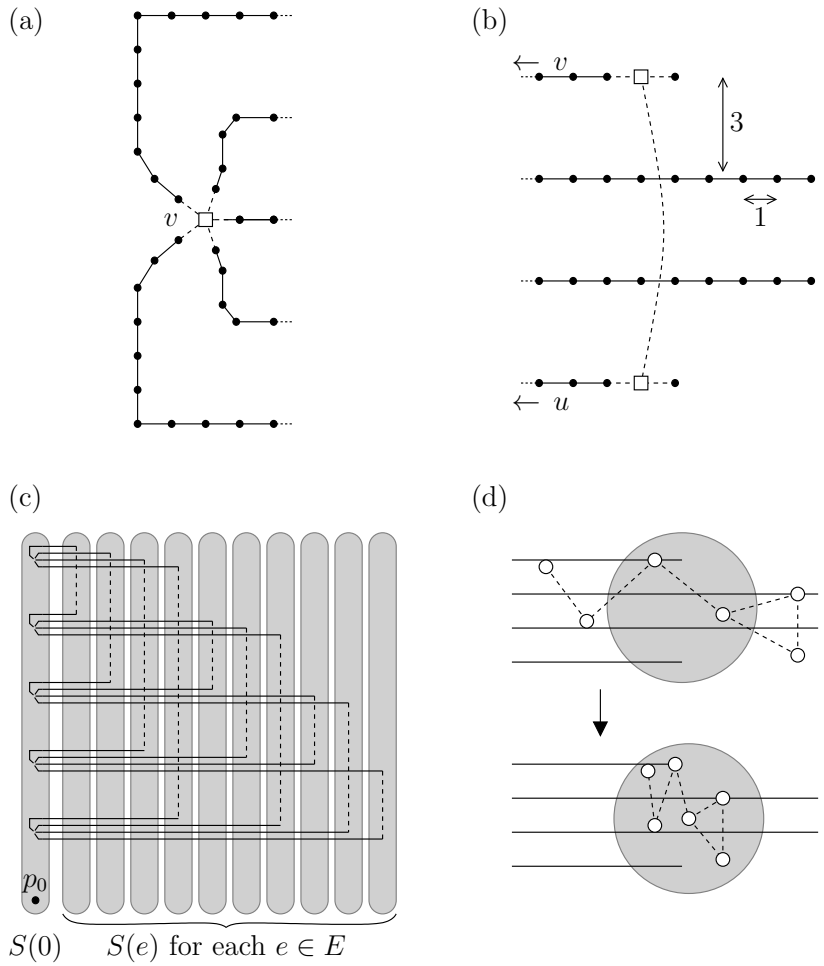

Figure 6: (a) Vertex gadget for $v \in V$. (b) Crossover gadget for $\{v, u\} \in E$. (c) Reduction for K_5. (d) Normalising a solution, step 1.

Lemma 7. *Assume that there exists a feasible solution to relay placement problem $\mathcal{I}$ with $k + 2|E| + 1$ relays. Then $\mathcal{G}$ has a vertex cover of size at most k.*

Proof. If $k \geq |V|$, then the claim is trivial: $C = V$ is a vertex cover of size at most k. We therefore focus on the case $k < |V|$.

Let R be a solution with $k+2|E|+1$ relays. We transform the solution into a canonical form R' of the same size and with the following additional constraints: there is a subset $C \subseteq V$ such that at least one relay is placed at the good relay location of each vertex gadget $v \in C$; two relays are placed at the good locations of each crossover gadget; one relay is placed at p_0; and there are no other relays. If R' is a feasible solution, then C is a vertex cover of $\mathcal{G}$ with $|C| \leq k$.

Now we show how to construct the canonical form R'. We observe that there are $2|E| + 1$ isolated sensors in $\mathcal{I}$: sensor p_0 and two sensors for each crossover gadget. In the feasible solution R, for each isolated sensor p, we can always identify one relay within distance 1 from p (if there are several relays, pick one arbitrarily). These relays are called *bound relays*. The remaining $k < |V|$ relays are called *free relays*.

Step 1. Consider the communication graph formed by the sensors in $\mathcal{I}$ and the relays R. Since each pair of disks $S(i)$, $i \in \{0\} \cup E$, is separated by a distance larger than $|V|r$, we know that there is no path that extends from one disk to another and consists of at

most k free relays (and possibly one bound relay in each end). Therefore we can shift each connected set of relays so that it is located within one disk (see Figure 6d). While doing so, we do not break any relay–relay links: all relays within the same disk can communicate with each other. We can also maintain each relay–blob link intact.

Step 2. Now we have a clique formed by a set of relays within each disk $S(i)$, there are no other relays, and the network is connected. We move the bound relay in $S(0)$ so that it is located exactly on p_0. For each $e \in E$, we move the bound relays in $S(e)$ so that they are located exactly on the good relay locations. Finally, any free relays in $S(0)$ can be moved to a good relay location of a suitable vertex gadget. These changes may introduce new relay–blob links but they do not break any existing relay–blob or relay–relay links.

Step 3. What remains is that some disks $S(e)$, $e \in E$, may contain free relays. Let x be one of these relays. If x can be removed without breaking connectivity, we can move x to the good relay location of any vertex gadget. Otherwise x is adjacent to exactly one blob of sensors, and removing it breaks the network into two connected components: component A, which contains p_0, and component B. Now we simply pick a vertex $v \in V$ such that the vertex gadget v contains sensors from component B, and we move x to the good relay location of this vertex gadget; this ensures connectivity between p_0 and B. □

Proof of Theorem 1. Let $\Delta, A, B, C \in \mathbb{N}$, with $\Delta \leq 5$ and $C > B$. Assume that there is a factor

$$\alpha = 1 + \frac{C - B}{B + \Delta A + 1}$$

approximation algorithm $\mathcal{A}$ for relay placement. We show how to use $\mathcal{A}$ to solve the following *gap-vertex-cover* problem for some $0 < \varepsilon < 1/2$: given a graph $\mathcal{G}$ with An nodes and maximum degree Δ, decide whether the minimum vertex cover of $\mathcal{G}$ is smaller than $(B + \varepsilon)n$ or larger than $(C - \varepsilon)n$.

If $n < 2$, the claim is trivial. Otherwise we can choose a positive constant ε such that

$$\alpha - 1 < \frac{C - B - 2\varepsilon}{B + \varepsilon + \Delta A + 1/n}$$

for any $n \geq 2$. Construct the relay placement instance $\mathcal{I}$ as described above.

If minimum vertex cover of $\mathcal{G}$ is smaller than $(B+\varepsilon)n$, then by Lemma 6, the algorithm $\mathcal{A}$ returns a solution with at most $b = \alpha((B + \varepsilon)n + 2|E| + 1)$ relays. If minimum vertex cover of $\mathcal{G}$ is larger than $(C - \varepsilon)n$, then by Lemma 7, the algorithm $\mathcal{A}$ returns a solution with at least $c = (C - \varepsilon)n + 2|E| + 1$ relays. As $2|E| \leq \Delta An$, we have

$$\begin{aligned} c - b &\geq (C - \varepsilon)n + 2|E| + 1 - \alpha\big((B + \varepsilon)n + 2|E| + 1\big) \\ &\geq \big(C - B - 2\varepsilon - (\alpha - 1)(B + \varepsilon + \Delta A + 1/n)\big)n > 0, \end{aligned}$$

which shows that we can solve the gap-vertex-cover problem in polynomial time.

For $\Delta = 4$, $A = 152$, $B = 78$, $C = 79$, and any $0 < \varepsilon < 1/2$, the gap-vertex-cover problem is NP-hard [3, Theorem 3]. □

Remark 1. We remind the reader that throughout this work we assume that radius r is part of the problem instance. Our proof of Theorem 1 heavily relies on this fact; in our reduction, $r = \Theta(|V|)$. It is an open question whether one-tier relay placement admits a PTAS for a small, e.g., constant, r.

6 A PTAS for Two-Tier Relay Placement

In the previous sections we studied one-tier relay placement, in which the sensors could communicate with each other, as well as with the relays. We gave a 3.11-approximation algorithm, and showed that the problem admits no PTAS (for general r). In this section we turn to the two-tier version, in which the sensors cannot communicate with each other, but only with relays.

The two-tier relay placement problem asks that we determine a set R of relays such that there exists a tree T whose internal nodes are the set R and whose leaves are the n input points (sensors) V, with every edge of T between two relays having length at most r and every edge of T between a relay and a leaf (sensor) having length at most 1.

We give a PTAS for this version of the problem, summarized in the following theorem.

Theorem 2. *The two-tier relay placement problem has a PTAS.*

We give an overview of the method here; details and proofs appear in the Appendix. Let m be a (sufficiently large) positive integer constant; we will give a $(1+O(1/m))$-approximate solution. We distinguish between two cases: the *sparse* case in which $\mathrm{diam}(V) \geq mnr$, and the *dense* case, in which $\mathrm{diam}(V) < mnr$.

In the sparse case, a solution can consist of long chains of relays, with a number of relays not bounded by a polynomial in n; thus, we output a succinct representation of such chains, specifying the endpoints (which come from a regular grid of candidate locations). The algorithm, then, is a straightforward reduction to the Euclidean minimum Steiner tree problem. See Appendix A.2.

In the dense case, we compute and output an explicit solution. In this case, the set of possible locations of relays that we need to consider is potentially large (but polynomial); we employ an "iterated circle arrangement" to construct the set, G, of candidate locations. Analysis of this set of candidates is done in Appendix A.4, where we prove the structure lemma, Lemma 8. Armed with a discrete candidate set, we then employ the m-guillotine method [23] to give a PTAS for computing a set of relays (a subset of G) that is within factor $1 + O(1/m)$ of being a minimum-cardinality set. The main idea is to optimize over the class of "m-guillotine solutions", which can be done using dynamic programming. An m-guillotine solution has a recursive property determined by "guillotine cuts" of the bounding box of the optimal solution (axis-parallel cuts of constant ($O(m)$) description complexity). We prove that an optimal solution that uses k^* relays can be augmented with a set of $O(k^*/m)$ additional relays so that it has the m-guillotine property.

7 Discussion

In Section 3 we presented a simple $O(n \log n)$-time 6.73-approximation algorithm for the one-tier relay placement. If one is willing to spend more time finding the approximation to the set cover, one may use the semi-local optimisation framework of Duh and Fürer [14], which provides an approximation ratio of $1 + 1/2 + 1/3 + 1/4 + 1/5 - 1/2$ for the set cover with at most 5 elements per set; hence we obtain a 5.73-approximation.

One can form a bipartite graph on the blobs and candidate stab locations as follows. Pick a point within each maximal-depth cell of the arrangement of the blobs (maximal w.r.t the blobs that contain the cell); call these points "red". Pick a point within each blob; call these points "blue". Connect each blue point to the red points contained in the blob, represented by the blue point. It is possible to pick the points so that the bipartite graph on the points is planar. Then the stab placement is equivalent to the Planar Red/Blue

Dominating Set Problem [12] – find fewest red vertices that dominate all blue ones. We believe that the techniques of Baker [2] can be used to give a PTAS for the problem. Combined with the simple algorithm in Section 3, this would result in a 4.16-approximation for the relay placement.

A more involved geometric argument may improve the analysis of yellow relays in Section 4, bringing the constant 3.11 in Lemma 5 down to 3, which would improve the approximation factor to 3.09. Combining this with the possible PTAS for the Planar Red/Blue Dominating Set would yield an approximation factor of $3 + \varepsilon$. We believe that a different, integrated method would be needed for getting below 3: various steps in our estimates are tight with respect to 3. In particular, as the example in Figure 7 shows, our algorithm may find a solution with (almost) 3 times more relays than the optimum.

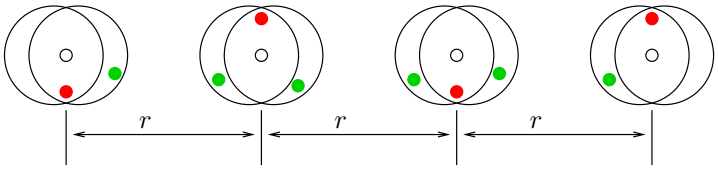

Figure 7: Unit disks centered on the sensors are shown. The optimum has 1 relay per blob (hollow circles). Our algorithm may place 1 red relay in every blob plus 2 green relays in (almost every) blob.

Acknowledgments

We thank Guoliang Xue for suggesting the problem to us and for fruitful discussions, and Marja Hassinen for comments and discussions. We thank the anonymous referees for their helpful suggestions.

A preliminary version of this work appeared in the *Proceedings of the 16th European Symposium on Algorithms* (ESA 2008) [15].

Parts of this research were conducted at the Dagstuhl research center. AE is supported by NSF CAREER Grant 0348000. Work by SF was conducted as part of EU project FRONTS (FP7, project 215270.) JM is partially supported by grants from the National Science Foundation (CCF-0431030, CCF-0528209, CCF-0729019, CCF-1018388, CCF-1540890), NASA Ames, Metron Aviation, and Sandia National Labs. JS is supported in part by the Academy of Finland grant 116547, Helsinki Graduate School in Computer Science and Engineering (Hecse), and the Foundation of Nokia Corporation.

References

[1] Sanjeev Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998. doi: 10.1145/290179.290180.

[2] Brenda S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994. doi: 10.1145/174644.174650.

[3] Piotr Berman and Marek Karpinski. On some tighter inapproximability results. In *Proc. 26th International Colloquium on Automata, Languages and Programming (ICALP 1999)*, volume 1644 of *LNCS*, pages 200–209, Berlin, 1999. Springer. doi: 10.1007/3-540-48523-6_17.

[4] Jonathan Bredin, Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Daniela Rus. Deploying sensor networks with guaranteed fault tolerance. *IEEE/ACM Transactions on Networking*, 18(1):216–228, 2010. doi: 10.1145/1816288.1816305.

[5] Paz Carmi, Matthew J. Katz, and Nissan Lev-Tov. Covering points by unit disks of fixed location. In *Proc. 18th Symposium on Algorithms and Computation (ISAAC 2007)*, volume 4835 of *LNCS*, pages 644–655, Berlin, 2007. Springer. doi: 10.1007/978-3-540-77120-3_56.

[6] J. Orestes Cerdeira and Leonor S. Pinto. Requiring connectivity in the set covering problem. *Journal of Combinatorial Optimization*, 9(1):35–47, 2005. doi: 10.1007/s10878-005-5482-5.

[7] Donghui Chen, Ding-Zhu Du, Xiao-Dong Hu, Guo-Hui Lin, Lusheng Wang, and Guoliang Xue. Approximations for Steiner trees with minimum number of Steiner points. *Journal of Global Optimization*, 18(1):17–33, 2000. doi: 10.1023/A:1008384012064.

[8] Donghui Chen, Ding-Zhu Du, Xiao-Dong Hu, Guo-Hui Lin, Lusheng Wang, and Guoliang Xue. Approximations for Steiner trees with minimum number of Steiner points. *Theoretical Computer Science*, 262(1–2):83–99, 2001. doi: 10.1016/S0304-3975(00)00182-1.

[9] Xiuzhen Cheng, Ding-Zhu Du, Lusheng Wang, and Baogang Xu. Relay sensor placement in wireless sensor networks. *Wireless Networks*, 14(3):347–355, 2008. doi: 10.1007/s11276-006-0724-8.

[10] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, Cambridge, MA, 2nd edition, 2001.

[11] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 2008.

[12] Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Springer, Berlin, 1999.

[13] Ding-Zhu Du and Frank K. Hwang. An approach for proving lower bounds: Solution of Gilbert–Pollak's conjecture on Steiner ratio. In *Proc. 31st Symposium on Foundations of Computer Science (FOCS 1990)*, pages 76–85, Piscataway, 1990. IEEE. doi: 10.1109/FSCS.1990.89526.

[14] Rong-Chii Duh and Martin Fürer. Approximation of k-set cover by semi-local optimization. In *Proc. 29th Symposium on Theory of Computing (STOC 1997)*, pages 256–264, New York, 1997. ACM Press. doi: 10.1145/258533.258599.

[15] Alon Efrat, Sándor P. Fekete, Poornananda R. Gaddehosur, Joseph S. B. Mitchell, Valentin Polishchuk, and Jukka Suomela. Improved approximation algorithms for relay placement. In *Proc. 16th European Symposium on Algorithms (ESA 2008)*, volume 5193 of *LNCS*, pages 356–367, Berlin, 2008. Springer. doi: 10.1007/978-3-540-87744-8_30.

[16] Bin Hao, Han Tang, and Guoliang Xue. Fault-tolerant relay node placement in wireless sensor networks: formulation and approximation. In *Proc. 2004 Workshop on High Performance Switching and Routing (HPSR 2004)*, pages 246–250, Piscataway, 2004. IEEE. doi: 10.1109/HPSR.2004.1303479.

[17] Dorit S. Hochbaum and Wolfgang Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32(1):130–136, 1985. ISSN 0004-5411. doi: 10.1145/2455.214106.

[18] Guo-Hui Lin and Guoliang Xue. Steiner tree problem with minimum number of Steiner points and bounded edge-length. *Information Processing Letters*, 69(2):53–57, 1999. doi: 10.1016/S0020-0190(98)00201-4.

[19] Hai Liu, Pengjun Wan, and Xiaohua Jia. On optimal placement of relay nodes for reliable connectivity in wireless sensor networks. *Journal of Combinatorial Optimization*, 11:249–260, 2006. doi: 10.1007/s10878-006-7140-y.

[20] Errol L. Lloyd and Guoliang Xue. Relay node placement in wireless sensor networks. *IEEE Transactions on Computers*, 56(1):134–138, 2007. doi: 10.1109/TC.2007.18.

[21] Ion I. Măndoiu and Alexander Z. Zelikovsky. A note on the MST heuristic for bounded edge-length Steiner trees with minimum number of Steiner points. *Information Processing Letters*, 75(4):165–167, 2000. doi: 10.1016/S0020-0190(00)00095-8.

[22] Satyajayant Misra, Seung Don Hong, Guoliang Xue, and Jian Tang. Constrained relay node placement in wireless sensor networks to meet connectivity and survivability requirements. In *Proc. 27th Conference on Computer Communications (INFOCOM 2008)*, pages 879–887, Piscataway, 2008. IEEE. doi: 10.1109/INFOCOM.2008.65.

[23] Joseph S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM Journal on Computing*, 28(4):1298–1309, 1999. doi: 10.1137/S0097539796309764.

[24] Tian-Ping Shuai and Xiao-Dong Hu. Connected set cover problem and its applications. In *Proc. 2nd Conference on Algorithmic Aspects in Information and Management (AAIM 2006)*, volume 4041 of *LNCS*, pages 243–254, Berlin, 2006. Springer. doi: 10.1007/11775096_23.

[25] Anand Srinivas, Gil Zussman, and Eytan Modiano. Mobile backbone networks – construction and maintenance. In *Proc. 7th Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2006)*, pages 166–177, New York, 2006. ACM Press. doi: 10.1145/1132905.1132924.

[26] Jian Tang, Bin Hao, and Arunabha Sen. Relay node placement in large scale wireless sensor networks. *Computer Communications*, 29(4):490–501, 2006. doi: 10.1016/j.comcom.2004.12.032.

[27] Yang Yang, Mingen Lin, Jinhui Xu, and Yulai Xie. Minimum spanning tree with neighborhoods. In *Proc. 3rd Conference on Algorithmic Aspects in Information and Management (AAIM 2007)*, volume 4508 of *LNCS*, pages 306–316, Berlin, 2007. Springer. doi: 10.1007/978-3-540-72870-2_29.

[28] Weiyi Zhang, Guoliang Xue, and Satyajayant Misra. Fault-tolerant relay node placement in wireless sensor networks: Problems and algorithms. In *Proc. 26th Conference on Computer Communications (INFOCOM 2007)*, pages 1649–1657, Piscataway, 2007. IEEE. doi: 10.1109/INFCOM.2007.193.

A Appendix

Here, we give details of the PTAS that was outlined in Section 6.

A.1 Preliminaries

Let m be a (sufficiently large) positive integer constant. We present an algorithm that computes an approximate solution to the two-tier relay placement problem, with approximation factor $1 + O(1/m)$ and running time that is polynomial in $n = |V|$. We focus on nontrivial instances in which at least 2 relays are required (i.e., there is no single point that is within distance 1 from all sensors).

We rephrase the statement of the two-tier relay placement problem: Determine a set R of relays such that there exists a tree T whose internal nodes are the set R and whose leaves are the n input points (sensors) V, with every edge of T between two relays having length at most r and every edge of T between a relay and a leaf (sensor) having length at most 1. We refer to edges of length at most r between two relays as *blue* edges and edges of length at most 1 between a relay and a sensor as *red* edges; we refer to relays that are incident on red edges as *red relays* and all other relays as *blue relays*. Our objective is to minimize $|R|$.

Let $D = \text{diam}(V) - 1$. We say that an instance of the two-tier relay placement problem is *sparse* (with respect to the constant m) if $D \geq mnr$, and *dense* if $D < mnr$. We treat sparse and dense instances with different techniques:

(i) A solution of a sparse instance typically consists of long chains of relays. The number of relays is not necessarily polynomial in n, and the algorithm must output a succinct representation that specifies the endpoints of such chains. However, the set of possible endpoints that we need to consider is relatively small, and we can use a simple regular grid to construct the set of candidate locations.

(ii) A solution of a dense instance can be given explicitly. However, the set of possible locations of relays that we need to consider is large, and we must use an iterated circle arrangement to construct the set of candidate locations.

We first show how to solve sparse instances in Section A.2; the algorithm is a straightforward reduction to the Euclidean minimum Steiner tree problem.

A.2 Sparse Instances

We first discretize the problem by using a regular grid. Let $s = D/(nm)$; note that $s \geq r \geq 1$. Let G be the regular square grid of spacing s that covers the bounding rectangle of V; the number of points in G is $O(n^2m^2)$.

We round the coordinates of each sensor $v \in V$ to the nearest grid point $G(v) \in G$; let $G(V) = \{G(v) : v \in V\}$ be the set of the rounded sensor coordinates. Then we use a polynomial-time approximation algorithm [1, 23] to find a Steiner tree S for $G(V)$ with Steiner points in G such that the total length $|S|$ of the edges is within factor $1 + 1/m$ of the smallest such tree. Without loss of generality, we can assume that S has $O(n)$ edges.

Then we construct a feasible solution R of the two-tier relay placement problem as follows:

(i) Replace each edge $\{a, b\}$ of S by a chain of relays with spacing r that connects a to b.

(ii) Connect each sensor $v \in V$ to the point $G(v)$ by a chain of relays.

Clearly the solution R is feasible and a compact representation of it can be constructed in polynomial time. Let us now analyze the approximation factor. To this end, consider an optimal solution R^* of the two-tier relay placement problem. Given R^*, we can construct a Steiner tree S^* for V such that there are at most $n-2$ Steiner points. Each Steiner point is a relay of R^* with degree larger than 2, and each $v \in V$ is a leaf node of S^*; moreover, $|S^*| \leq n + |R^*|r$.

If we round the coordinates of the vertices of S^* to the nearest grid points in G, we obtain a Steiner tree S_1 for $G(V)$ with $|S_1| \leq O(ns) + |R^*|r$. By construction, S is at most $1 + 1/m$ times larger than S_1; hence

$$|S| \;\leq\; O(ns) + (1 + 1/m)|R^*|r.$$

In step (i), we add at most $O(n) + |S|/r$ relays in total, and in step (ii), we add $O(ns/r + n)$ relays in total. Hence,

$$|R| \;\leq\; O(n + ns/r) + |S|/r \;\leq\; O(ns/r) + (1 + 1/m)|R^*|.$$

Now observe that $|R^*| \geq D/r$, as we must connect the pair of sensors that are at distance $\mathrm{diam}(V)$ from each other. Hence,

$$ns/r \;=\; D/(mr) \;\leq\; (1/m)|R^*|,$$

and we conclude that $|R|$ is within factor $1 + O(1/m)$ of $|R^*|$.

A.3 Dense Instances

In Section A.4 we prove the following lemma showing that we can focus on a polynomial-size set G of candidate relay positions.

Lemma 8. *For any fixed positive integer m and a given dense instance of the two-tier relay placement problem, we can construct in polynomial time a (polynomial-size) set of points G such that there exists a feasible, $(1 + O(1/m))$-approximate solution R_1 with $R_1 \subseteq G$.*

Lemma 8 implies that, in our quest for a PTAS for our two-tier relay placement problem, it suffices for us to consider sets of relays on the grid G.

Our PTAS method is based on the m-guillotine framework of Mitchell [23]. We review a few definitions. Let A be a connected set of line segments with endpoints in G. A *window* W is an axis-aligned rectangle whose defining coordinates are x- and y-coordinates of the grid points G. A *cut* is a horizontal or vertical line ℓ, through a point of G, that intersects $\mathrm{int}(W)$, the interior of window W. The intersection, $\ell \cap (A \cap \mathrm{int}(W))$, of a cut ℓ with $A \cap \mathrm{int}(W)$ consists of a discrete (possibly empty) set of crossing points where an edge from A crosses ℓ. Let the crossing points be denoted by $p_1, \ldots, p_\xi$, in order along ℓ. We define the *m-span*, $\sigma_m(\ell)$, of ℓ (with respect to W) to be the empty set if $\xi \leq 2(m-1)$, and to be the (possibly zero-length) line segment $p_m p_{\xi-m+1}$ otherwise.

A cut ℓ is *m-perfect with respect to W* if $\sigma_m(\ell) \subseteq A$. The edge set A is *m-guillotine with respect to window W* if either

(1) $A \cap \mathrm{int}(W) = \emptyset$, or

(2) there exists an m-perfect cut, ℓ, with respect to W, such that A is m-guillotine with respect to windows $W \cap H^+$ and $W \cap H^-$, where H^+, H^- are the closed halfplanes defined by ℓ.

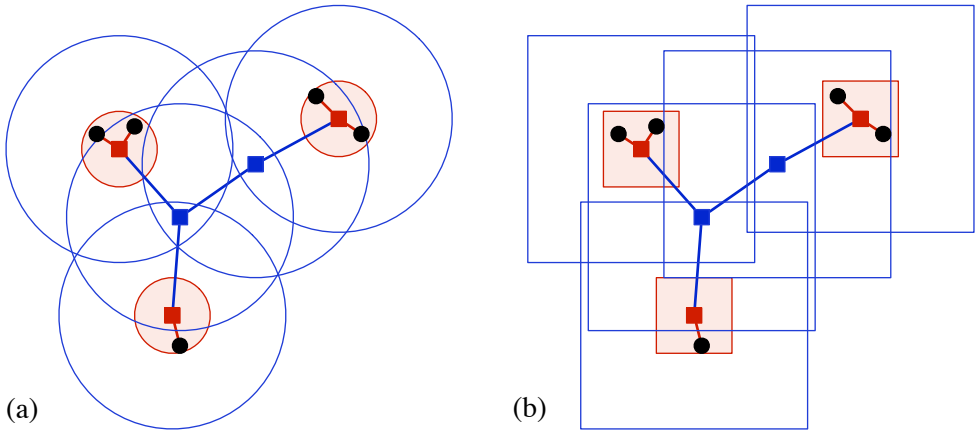

Figure 8: Optimal solution R^*_G and tree T^*. (a) Red and blue disks. (b) Red and blue squares.

We say that A is *m-guillotine* if A is m-guillotine with respect to the axis-aligned bounding box of A.

Let R^*_G denote an optimal solution to the two-tier relay placement problem restricted to points of the grid G. Associated with R^*_G is a tree T^* whose internal nodes are the set R^*_G and whose leaves are the n input points (sensors) V, with every edge of T^* between two relays having length at most r and every edge of T^* between a relay and a leaf (sensor) having length at most 1; see Figure 8. Some edges of T^* are blue (those of length at most r between two relays), and some edges are red (those of length at most 1 between a red relay and a sensor). Relays that are incident on red edges are *red* and all other relays are *blue*. We place *red disks* of radius 1 centered at each red relay and place *blue disks* of radius r centered at *every* relay (blue or red). Each red disk (resp., blue disk) has an associated bounding box, which we call a *red square* (resp., *blue square*). We observe that each blue relay has constant degree in T^* (in fact, at most 5, based on the degree bound of Euclidean minimum spanning trees in the plane), and no point in the plane lies in more than a constant number of blue squares (for the same basic reasons as the degree bound – otherwise, the set of relays could be reduced, while maintaining communication connectivity). Further, we observe that the union of the edges bounding all blue squares is connected (since the blue edges of T^* form a subtree of T^*, and any two relays joined by a blue edge must have their corresponding blue squares intersecting).

The m-guillotine method is typically applied to a set of *edges* of some network. We now define a related concept, that of a set of relays being "m-guillotine". Let $R \subseteq G$ be a set of relays that is feasible for V, meaning that there is an associated tree T with leaves at V, blue edges of lengths at most r and red edges of lengths at most 1. Let E_{blue} be the set of (axis-parallel) edges bounding the blue squares associated with R, and let E_{red} be the set of (axis-parallel) edges bounding the red squares associated with R. Consider a window W (on the grid induced by the points G) and a horizontal/vertical cut ℓ intersecting $\text{int}(W)$.

The intersection, $\ell \cap (E_{\text{blue}} \cap \text{int}(W))$, of a cut ℓ with $E_{\text{blue}} \cap \text{int}(W)$ consists of a discrete (possibly empty) set of crossing points where an edge from E_{blue} crosses ℓ. Without loss of generality, consider a vertical cut ℓ, and let $ab = \ell \cap W$ denote its intersection with the window W. We now define the *blue m-span*, $\sigma^{\text{blue}}_m(\ell)$, of ℓ (with respect to W), as follows. As we walk along ab from a towards b, we enter various blue squares; let p_m be the mth entry point, if it exists; if we enter fewer than m blue squares, then we define the blue m-span to be empty. (Note that the point a may lie within more than one blue square;

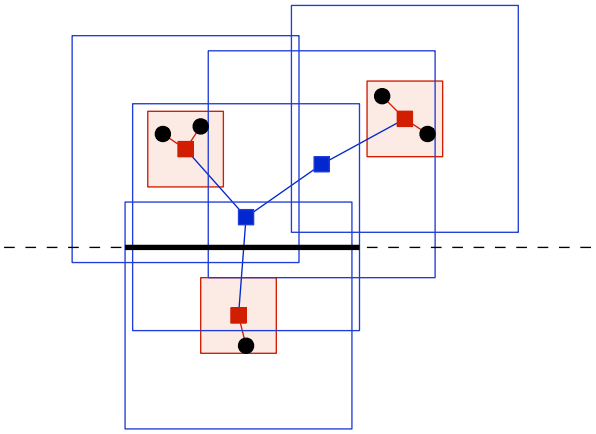

Figure 9: A horizontal cut ℓ (dashed line) and the blue m-span $\sigma_m^{\text{blue}}(\ell)$ for $m = 2$ (thick line).

however, we can always assume that no point lies within more than a constant number of blue squares.) Similarly, as we walk along ab from b towards a, we enter various blue squares; let q_m be the mth entry point, if it exists; if we enter fewer than m blue squares, then we define the blue m-span to be empty. Then, if p_m is closer to a than q_m, and the length of $p_m q_m$ is at least $2r$ (implying that $p_m q_m$ fully crosses at least one blue square, entering and exiting it), we define the blue m-span, $\sigma_m^{\text{blue}}(\ell)$, of ℓ (with respect to W), to be the segment $p_m q_m$, a subsegment of ℓ; otherwise, we define the blue m-span to be empty. See Figure 9 for an illustration. Note that, if nonempty, by definition the blue m-span has length at least $2r$, the side length of a blue square; further, if the blue m-span is empty, then ab intersects $O(m)$ blue squares. We similarly define the *red m-span*, $\sigma_m^{\text{red}}(\ell)$, of ℓ (with respect to W) based on the entry points along ℓ of red squares. Note that, if nonempty, by definition the red m-span has length at least 1; further, if the red m-span is empty, then ab intersects $O(m)$ red squares.

We say that a nonempty blue m-span $\sigma_m^{\text{blue}}(\ell)$ is *relay-dense* if R includes relays at all points of G_0^{blue} that are corners of $(r/2)$-by-$(r/2)$ grid cells intersected by $\sigma_m^{\text{blue}}(\ell)$; the corresponding relays are called a *relay-dense bridge*; see Figure 10. (G includes the points G_0^{blue} of the regular square grid of spacing $r/2$ that covers the bounding rectangle of V; see Section A.4.) Similarly, we say that a nonempty red m-span is *relay-dense* if R includes relays at all points of G_0^{red} that are corners of $(1/2)$-by-$(1/2)$ grid cells that contain points of V (which implies that it includes at least one (red) relay within each disk of radius 1, centered at V, that is intersected by $\sigma_m^{\text{red}}(\ell)$); the corresponding red relays are called a *(red) relay-dense bridge*. (G includes the points G_0^{red} at the corners of the cells of the regular square grid of spacing $1/2$ that contain points V; see Section A.4.)

A cut ℓ is *m-perfect with respect to R* if the nonempty m-spans (red and blue) along ℓ are relay-dense. The set R of relays is *m-guillotine with respect to window W* if either

(1) there are no blue squares of R interior to W, or

(2) there exists an m-perfect cut, ℓ, with respect to W, such that R is m-guillotine with respect to windows $W \cap H^+$ and $W \cap H^-$, where H^+, H^- are the closed halfplanes defined by ℓ.

We say that R is *m-guillotine* if R is m-guillotine with respect to the axis-aligned bounding box of R.

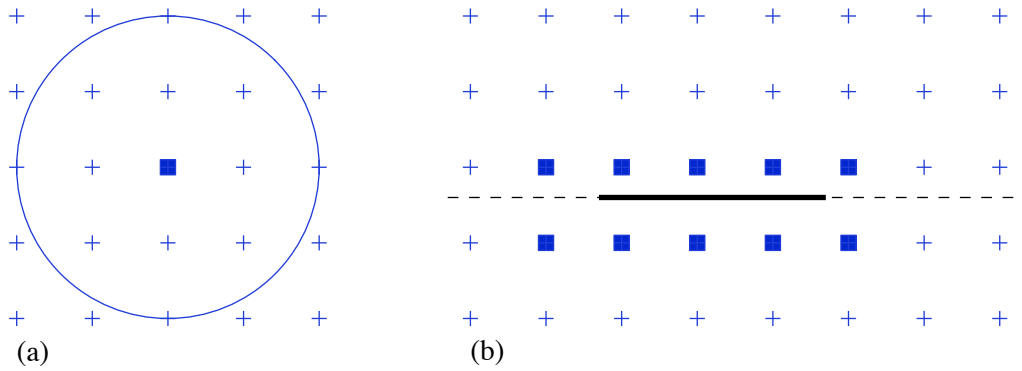

Figure 10: (a) Regular square grid G_0^{blue} of spacing $r/2$. (b) A relay-dense m-span $\sigma_m^{\text{blue}}(\ell)$.

Lemma 9. *Let m be a fixed positive integer, and let V be a given dense instance of the two-tier relay placement problem. Then, for any set R' of relays that is feasible for the instance, there exists an m-guillotine set R of relays, also feasible for the instance, with $|R| \leq (1 + O(1/m))|R'|$.*

Proof. We argue that we can add a small number, at most $O(1/m) \cdot |R'|$, of relays to R' to make it m-guillotine, if it is not already.

First, consider the set of blue and red squares associated with R'; these are the bounding boxes of the disks of radius r centered at all relays and disks of radius 1 centered at red relays. Let E_{blue} and E_{red} be the (axis-parallel) edges bounding the blue and red squares, respectively.

By the standard m-guillotine arguments [23], we know that E_{blue} can be augmented by "blue" bridges (m-spans) of length totalling $O(1/m) \cdot |E_{\text{blue}}|$, so that the resulting set of edges is m-guillotine (in the usual sense). A similar statement holds for the set E_{red} of red edges, which can be augmented by "red" bridges to make the set m-guillotine if it is not already m-guillotine. In fact, by a slight modification of the usual argument, we can claim that we can augment $E_{\text{blue}} \cup E_{\text{red}}$ by a set of red and blue bridges so that the resulting set is collectively m-guillotine, with a common recursive partitioning by cuts: For each cut ℓ in the recursive decomposition, we can choose ℓ (horizontal or vertical, in an appropriate position) so that the red m-span (with respect to E_{red}) and the blue m-span (with respect to E_{blue}) can each be charged off to the red and blue lengths of the input, charging the input with a factor of $O(1/m)$. In more detail, we define the cost of a vertical cut at position x to be $f_{\text{red}}(x) + (1/r)f_{\text{blue}}(x)$, where $f_{\text{red}}(x)$ is the length of the m-span at position x of the red edges E_{red}, and $f_{\text{blue}}(x)$ is the length of the m-span at position x of the blue edges E_{blue}. The union of m-spans over all x defines a "red region" and a "blue region" with respect to vertical cuts, and the integral, $\int_x [f_{\text{red}}(x) + (1/r)f_{\text{blue}}(x)]dx$ gives the weighted sum of the areas of these regions, weighting the length of the blue m-span by $1/r$ (this is to reflect the fact that the blue squares are r times as large as the red squares, and our goal is to count the cardinality of squares). Similarly, we can define red and blue regions with respect to *horizontal cuts*, and functions, $g_{\text{red}}(y)$ and $g_{\text{blue}}(y)$, that define the cost of such cuts.

The (weighted) areas of the red/blue regions with respect to horizontal cuts are given by the integrals $\int_y g_{\text{red}}(y)dy$ and $\int_y (1/r)g_{\text{blue}}(y)dy$; equivalently they are given by the integrals, $\int_y h_{\text{red}}(x)dx$ and $\int_x (1/r)h_{\text{blue}}(x)dx$, of the lengths $h_{\text{red}}(x)$ and $h_{\text{blue}}(x)$ of the vertical sections of the regions. These lengths ($h_{\text{red}}(x)$ and $h_{\text{blue}}(x)$) represent the "chargeable length" of vertical cuts at position x. Then, assuming that the weighted sum of

the areas of the red and blue regions with respect to horizontal cuts is greater than that with respect to vertical cuts, we get that there must be a vertical cut at some position x^* where $f_{\text{red}}(x^*) + (1/r)f_{\text{blue}}(x^*) \leq h_{\text{red}}(x^*) + (1/r)h_{\text{blue}}(x^*)$. A vertical cut through x^* can then have its red m-span charged to $O(1/m)$ of the length of edges bounding red squares (i.e., $O(1/m)$ times the number of red squares) and its blue m-span charged to $O(1/mr)$ times the length of edges bounding blue squares (i.e., $O(1/m)$ times the number of blue squares/relays).

Next, consider each blue bridge (m-span) that the above argument shows we can afford to add (charging $O(1/m)$ times the number of relays), to make E_{blue} m-guillotine (as a network). We can convert any such blue bridge of length λ into $O(\lceil \lambda/r \rceil)$ relays on or near the corresponding cut ℓ, in order to make the cut relay-dense (i.e., to include relays at all points of G_0^{blue} that are corners of $(r/2)$-by-$(r/2)$ grid cells intersected by the blue bridge). Similarly, each red bridge (m-span), of length λ, that is added in order to make E_{red} m-guillotine (as a network) can be converted into $O(\lceil \lambda \rceil)$ red relays that stab all disks of radius 1, centered at points of V, that are intersected by the red bridge. Similarly, we say that a nonempty red m-span is *relay-dense* if R includes at least one (red) relay from the set G within each disk of radius 1, centered at V, that is intersected by $\sigma_m^{\text{red}}(\ell)$. The purpose of relay-dense bridges of red relays is to ensure that sensors within distance 1 of the boundary of W are covered by the unit disks of the red relay-dense bridge, if they are not covered by the unit disks associated with the $O(m)$ unbridged red relays. This allows for "separation" between subproblems in the dynamic program.

Finally, we conclude that the total number of relays added is $O(1/m)$ times $|R|$. If we let the lengths of blue bridges be λ_i^{blue} and let the lengths of red bridges be λ_j^{red}, then we know, from the charging scheme for making E_{blue} and E_{red} m-guillotine, that $(1/r)\sum_i \lambda_i^{\text{blue}} + \sum_j \lambda_j^{\text{red}}$ is at most $O(1/mr)$ times the total perimeters of all blue squares plus $O(1/m)$ times the total perimeters of all red squares. This implies that $(1/r)\sum_i \lambda_i^{\text{blue}} + \sum_j \lambda_j^{\text{red}}$ is at most $O(1/m)$ times the number of relays of R. By construction, we know that if λ_i^{blue} is nonzero, then $\lambda_i^{\text{blue}} \geq r$, and that if λ_j^{red} is nonzero, then $\lambda_j^{\text{red}} \geq 1$. Thus, the total number of relays added, $O(\sum_i \lceil \lambda_i^{\text{blue}}/r \rceil + \sum_j \lceil \lambda_j^{\text{red}} \rceil)$ is at most $O(1/m)$ times the number of relays of R. □

Applying Lemma 9 and Lemma 8 yields the following

Corollary 1. *Let m be a fixed positive integer, and let V be a given dense instance of the two-tier relay placement problem. Then, there exists an m-guillotine set $R \subseteq G$ of relays with $|R| \leq (1 + O(1/m))|R^*|$.*

We now describe the algorithm that yields the claimed PTAS for computing R^* approximately. The algorithm is based on a dynamic program to compute a minimum-cardinality m-guillotine set, R, of relays that obeys certain connectivity and coverage constraints. A subproblem is specified by a window W (with coordinates from the grid G), together with a constant (depending on m) amount of boundary information:

(1) A set of $O(m)$ (unbridged) blue relays (from G) whose blue squares intersect the boundary of W, and a set of $O(m)$ red relays (from G) whose red squares intersect the boundary of W.

(2) The blue m-spans, which determine the set of relays forming the relay-dense bridge; in order to describe the relay-dense bridge, we have only to specify the endpoints of the blue m-span, as this gives a succinct encoding of the relays that form the relay-dense bridge. There are up to 4 blue m-spans, one per side of W.

(3) The red m-spans, which determine the set of red relays forming the relay-dense bridge; in order to describe the relay-dense bridge, we have only to specify the endpoints of the red m-span, as this gives a succinct encoding of the relays that form the relay-dense bridge.

(4) Connectivity requirements among the relays specified along the boundary. Specifically, there are $O(m)$ entities specified along the boundary of W: $O(m)$ unbridged relays per side of W, and up to two m-spans per side of W, which encode the relay-dense bridges. The connectivity requirements are specified as a collection of (disjoint) subsets of these $O(m)$ entities, with the understanding that each such subset of entities must be connected within W by edges of the communication graph that connects two relays if and only if they are at distance at most r from each other.

We require that the solution to a subproblem gives a set of red relays whose unit disks cover all sensors interior to W that are not covered by the relays specified along the boundary (either the $O(m)$ unbridged red relays or the relay-dense bridges of red relays), together with a set of blue relays within the subproblem that enable connectivity. The dynamic program optimizes over all choices of cuts ℓ (horizontal or vertical) of W, and all choices of boundary information along the cut ℓ that are compatible with the boundary information for W. Since, for fixed m, there are a polynomial number of different subproblems, and a polynomial number of choices of cuts and boundary information within the dynamic programming recursion, we have completed the proof of the main result, Theorem 2.

A.4 Proof of Lemma 8

Construction of G. Let G_0^{blue} be the regular square grid of spacing $r/2$ that covers the bounding rectangle of V; the points of G_0^{blue} are the corners of $(r/2)$-by-$(r/2)$ squares that form a regular grid. By construction, the grid G_0^{blue} has $O((D/r)^2 + 1) = O(n^2m^2)$ points.

Consider now the regular square grid of spacing $1/2$ that covers the bounding rectangle of V. Let G_0^{red} be the subset of these grid points that are within distance 1 of one of the n input points V. Then, G_0^{red} are the corners of $(1/2)$-by-$(1/2)$ squares of a regular grid, and there are only $O(n)$ such corner points.

We construct an *iterated circle arrangement* $C_0, C_1, \ldots, C_m$ recursively as follows. As the base case, let $C_0 = G_0^{\text{blue}} \cup V$. Then, for each $i = 1, 2, \ldots, m$, the arrangement C_i is constructed as follows: For each point $p \in C_{i-1}$, add p to C_i, as well as the *special points* located 1 unit below p and r units below p. Then draw circles of radii $1, r, r+1, 2r, 2r+1, 3r, 3r+1, \ldots, mr, mr+1$ centered at each point of C_{i-1}. Finally, add the vertices (crossing points) of this arrangement to C_i.

Finally, let $G = C_m \cup G_0^{\text{red}}$. The size of G is polynomial in n, for a constant m.

Construction of R_1. Consider an optimal solution R^* and the minimum-length communication tree T^* associated with it; let B^* be the blue subtree of T^*. Since B^* is an MST of R^*, the maximum degree of any relay in B^* is 5. Hence there is a set E' of $O(|R^*|/m)$ blue edges whose removal breaks B^* into subtrees, each having at most m relays.

We construct a new solution R_1 that conforms to G as follows; see Figure 11 for an illustration. First, for each edge $\{x, y\} \in E'$, we add a new relay z at a point of G_0^{blue} that is within distance r from x and y; then we replace the edge $\{x, y\}$ by two edges, $\{x, z\}$ and $\{z, y\}$. The new blue edge $\{x, z\}$ becomes part of the same subtree as x, and the edge $\{z, y\}$ becomes part of the same subtree as y.

So far we have added $O(|R^*|/m)$ new relays, and we have partitioned the communication tree into edge-disjoint subtrees. The leaf nodes of the subtrees are sensors or newly added

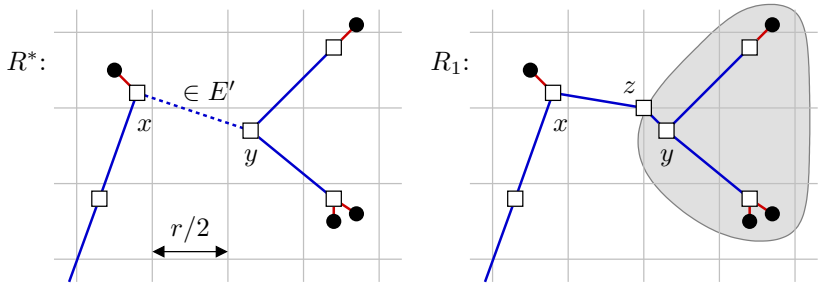

Figure 11: Construction of R_1 for dense instances. Each edge in the set E' is replaced by a path of length 2. The grey area shows one of the subtrees. The leaf nodes of the subtree are now pinned at points of C_0, and there are at most m internal nodes. Figure 12 shows how we can "pin" the internal nodes of the subtree to points of G.

relays; both of them are located at points of C_0. There are at most m internal nodes in each subtree. To prove Lemma 8, it is sufficient to show that we can move the internal relays so that they are located at points of G, and while doing so we do not break any sensor–relay or relay–relay connections.

We say that a relay is *pinned* if it is located at a point of G. We show how to pin all relays in m iterations. In the beginning, we have partitioned the communication tree into edge-disjoint subtrees where the leaf nodes are pinned at C_0 and there are at most m internal nodes in each subtree. In what follows, we will show that we can move the internal nodes and refine the partition so that after iteration i, we have partitioned the communication tree into edge-disjoint subtrees where the leaf nodes are pinned at C_i and there are at most $m - i$ internal nodes in each subtree. Hence after m iterations, we have pinned all nodes at $C_m = G$.

Observe that it is sufficient to show that if T is a subtree where leaf nodes are pinned at C_i, then we can move the internal nodes so that at least one node x becomes pinned at C_{i+1}. Then we can partition the subtree T into smaller, edge-disjoint subtrees that share the leaf node x. Each of the new subtrees has all leaf nodes pinned at C_{i+1} and they also have fewer internal nodes than T.

Consider a subtree T. Start translating (in any direction) the internal nodes of T, while keeping the leaf nodes in place. This translation will cause some edge e linking a leaf node p to an internal node u to reach its maximum length (r for blue edges or 1 for red edges). At this moment, the point u lies on a circle c of radius r or 1 centered at p; see Figure 12 for an illustration.

Now slide the point u along circle c until some other constraint becomes binding. We now do not consider T to be a rigid structure; rather, we allow the internal nodes to move arbitrarily, subject to the upper bound constraints on each edge. The edges serve as "cables", which readily shorten, but cannot extend beyond their maximum length.

If we can slide u along c to the special point located below p, then we have pinned u at a point of C_{i+1}. Otherwise some edges of T reach their upper bound of length while we slide u along c; i.e., one or more cables becomes taut. At this moment, some subset of the edges of T are taut (at their length upper bound); this subset of edges forms a tree, T'. There are two cases:

(i) T' is a path. Then T' is a path linking u to a leaf node p' of T. Further, T' is a *straight* path of edges, each at its upper length bound; in fact, all edges of T' will be of length r, the upper bound for relay–relay edges, except, possibly, the edge incident on p' (if p' is a sensor). Thus, in this position, u lies at the intersection of circle c

with a circle centered at a point of P of radius jr or $jr+1$, for some integer $j \leq m$. As we had $p' \in C_i$, we will have now $u \in C_{i+1}$.

(ii) T' is not a path. Then, T' has some node, w (possibly equal to u) that is connected to leaves of T by two (or more) straight paths, each of length jr or $jr+1$, for some integer $j \leq m$; see Figure 12b. This implies that w lies at a point of C_{i+1}.

In both cases we have moved the internal nodes of T so that one of them becomes pinned at C_{i+1}; moreover, none of the communication links are broken, leaf nodes are held in place, and no new relays are added.

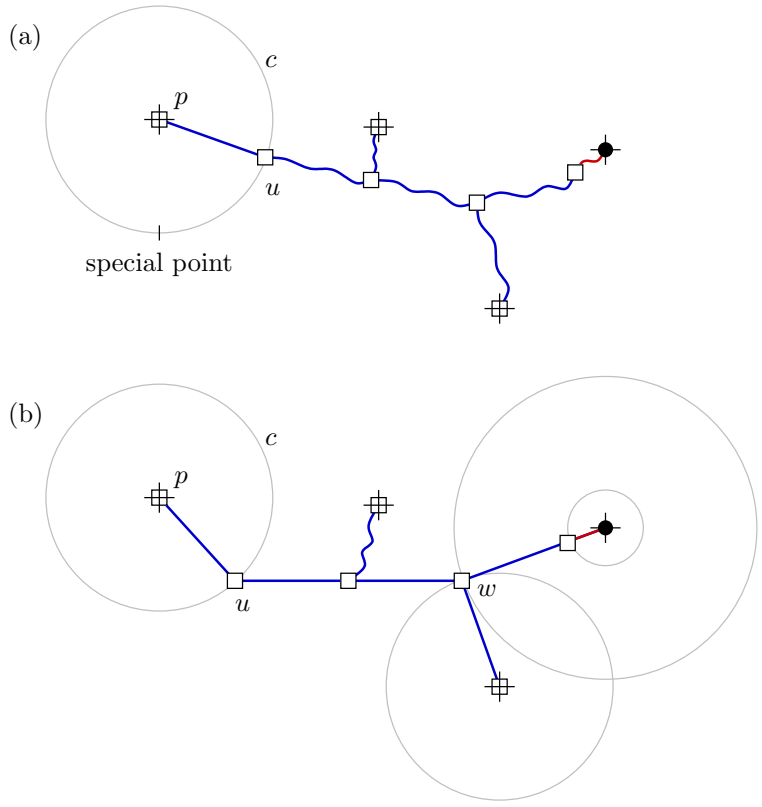

Figure 12: Pinning a subtree. Black dots are sensors and white boxes are relays. The leaf nodes are pinned at points of C_i. Straight lines denote "taut" edges, i.e., red edges of length 1 or blue edges of length r; other edges are shown with winding lines. (a) We have translated the internal nodes until the edge between a leaf node p and an internal node u becomes taut. Then we try to slide u along circle c towards the special point. If we succeed, we have pinned u at the special point, which is in C_{i+1}. (b) In this case we cannot slide u to the special point. However, from the subtree of taut edges, we can find a node w that is now located at a point of C_{i+1} – in this case at the intersection of a circle of radius $r+1$ centered at a sensor, and a circle of radius r centered at a pinned relay.