On the Minimum Relay Node Cover Problem

Fanchao Meng (fcmeng@udel.edu), Errol Lloyd (elloyd@udel.edu)

Abstract

This research studied the *Minimum Relay Node Cover (MRNC)* problem in several perspectives. It starts with a straightforward method delivering precise solutions yet running beyond polynomial time. Its significance lies in helping to understand why and where this problem is hard, as well as playing an important role in designing corresponding approximation algorithms. As the NP-hardness of this problem is trivially implied, from its generalization of the *Minimum Geographic Disk Cover (MGDC)* problem [1], the state-of-the-art method, the *Shifting Strategy* [7], remarkably proved to be a *Polynomial Time Approximation Scheme* (*PTAS)* [2], merits bearing the brunt of the analysis where how it perform on the *MRCN* problem is gripping most our attention. Specifically, three cases of the problem, from special to general, are discussed. The outcomes presented a problematic questioning if the validity of applying the *Shifting Strategy* to the *MRNC* problem is well-grounded, especially when the expectancy of ideally delivering another *PTAS* is highly emphasized. The questionableness underlying sources from the undetermined approximability (or in-approximability), therefrom a dedicated discussion was developed, in which two distinct ways are proposed trying to disprove its approximableness with the ratio bounded by a constant.

Introduction

The *MRNC* problem is a typical coverage problem in the Wireless Sensor Network area. It is meaningful because it is trying figure out the minimum number of relay nodes to cover a set of sensors in a Euclidean plane so that to lower the cost of the network. If considering the connectivity between relay nodes, the problem is called the *Relay Node Placement (RNP)* problem [1]. However, in the *RNP* problem, the sensor nodes coverage part is a traditional *Minimum Geographic Disk Cover (MGDC)* problem. The major difference between the *MRNC* problem and the *MGDC* problem is that the former one has non-unique communication ranges for the sensors whereas the latter one has a unique range. By now, no specific studies focus on the *MRNC* problem exist. The relaxation of the communication range assumption makes the problem closer to the real world. Obviously, it is harder than the *MGDC* problem, but how hard it is and if it also has a PTAS as the *MGDC* does are unknown, which makes this problem interesting.

Related Works

Two-Tiered Relay Node Placement Problem

The Multi-Tiered network model was proposed by Aura Gam et al. [28], though it was used for wireless cellular network instead of wireless sensor network. Jay Nemeroff et al. [29] brought up the idea introducing two-tiered architecture to wireless sensor network, in which the upper sub-layer can be cued by the lower sub-layer and provides a gateway link to higher nodes. Jianping Pan et al. [30] formulated the two-tiered wireless sensor network model, and studied majorly the topology control problem in the upper layer

constituted by application nodes (a less powerful kind of relay nodes) and base-stations (a more powerful kind of relay nodes). In [31], Bin Hao et al. studied a minimum relay node placement problem with two fault-tolerant constraints, first each sensor node can communicate with at least two relay nodes, and second the network of relay nodes is 2-connected, under the assumption the communication range of sensor nodes, r, and the communication range of relay nodes, R, satisfy, first r>0 and all r are the same, and second 2r≤R. They presented a polynomial time approximation with the performance bounded by O(DlogN), in which D is the (2,1)-Diameter of the network formed by a sufficient set of possible positions for relay nodes, and N is the number of sensor nodes in the network. In [32], Hai Liu et al. studied 1-connected and 2-connected two-tiered relay node placement problem under an assumption r=R, and for the former one they presented a (6 + ε)-approximation algorithm for any ε > 0 with polynomial running time when ε is fixed and two another approximation algorithm with (24+ε) and (6/T+12+ε) respectively, where T is the ratio of the number of relay nodes placed in case one to the number of sensor nodes. Jian Tang et al. [33] formulated the relay node placement problem into two optimization problems, Connected Relay Node Single Cover (CRNCS) problem and 2-Connected Relay Node Double Cover (2CRNDC) problem respectively, in which the latter one is a fault-tolerant version of the former one. These two problems were studied under the assumption 4r≤R, and the authors presented 4.5-approximation algorithms for both of them. In [34], Errol L. Lloyd et al. relaxed the communication range constraint to r≤R, and presented a polynomial time (5+ε)-approximation algorithm and another polynomial time randomized (4.5+ε)-approximation algorithm. S. M. Saif Shams et al. [35] studied the problem under the same assumption, r≤R, and presented two polynomial time approximation algorithms, 4-approximation for the case with base-station and 5-approximation for the case without base-station, whose worst case time complexity is bounded by O(N^2). In [36], Gang Wang et al. focused on the reliability of the relay node placement problem (k-connected), and presented an approximation algorithm with the performance ratio no worse than $\left(1+\left\lceil\frac{\sqrt{2}D}{2R}\right\rceil\right)(\ln n-\ln\ln n+O(1))$ for the two-tiered case. In [37], Guangting Chen et al. figured out a polynomial time (5+ε)-approximation algorithm for the 1-coverage 1-connected case with base-stations.

Minimum Disk Cover Problem

Masuyama Shigeru et al.[38] and Fowler, Robert J. et al.[39] proved this problem to be NP-hard. Hochbaum et al. presented a polynomial time approximation scheme (PTAS) in [40] with approximation ratio $\left(1+\frac{1}{l}\right)^2$, and the running time is $O\left(l^2\left\lceil l\sqrt{2}\right\rceil^2 n^{2\left\lceil l\sqrt{2}\right\rceil^2+1}\right)$. In [41], Teofilo F. Gonzalez et al. presented another approximation algorithm for the Hypersquare Cover problem with the approximation ratio 2^{d-1}, and the running time $O(dn+n\log s)$, where d means d-dimension, and s is the number of hypersquares in an optimal solution. The algorithms proposed in [42] according to the authors can be easily adapted to the Disk Cover problem. In [43], Brönnimann et al. presented a constant ratio, yet not determined, approximation algorithm takes no more than $O(n^3\log n)$. In [44], Franceschetti et al. developed an approximation algorithm with a better running time O(Kn) with approximation ratio $\alpha\left(1+\frac{1}{l}\right)^2$, where K is a constant and $\alpha\in\{3,4,5,6\}$. In [45], Călinescu Gruia, et al. proved that there is a 102-approximation algorithm for the Minimum Disk Cover Problem. In [46], Narayanappa et al. improved the approximation ratio to 72. In [47], Paz Carmi et al. got an even better result with 38 as the ratio. In [48], Francisco Claude et al. introduced their line-separable algorithm to solving this problem combining the results in [49], then a 22-approximation algorithm was born and its running time is $O(m^2n^4)$, where m is the number of unit disks and n is the number of points in the plane. In [50], Gautam K. Das et al. made another further step to achieve a 18-approximation algorithm with running time $O(n\log n+m\log m+mn)$. In [51], Chen Liao et al. proposed a PTAS for this problem with a factor of (1+ ε) in $O\left(mn^{O\left(\frac{1}{\epsilon^2}log^2\frac{1}{\epsilon}\right)}\right)$.

Problem Formulations

Definition 1 **(Communication Range Assumptions)**:

In this project, it assumes that,

1. All relay nodes have the exact same communication range, $R > 0$.
2. Every sensor node, s_i, has an individual communication range r_i, $R \geq r_i > 0$, and for any two different sensor nodes, s_i, s_j, such that $i \neq j$, it is possible that $r_i \neq r_j$.

Definition 2 **(Coverage Requirement for a Given Set of Sensor Nodes)**:

The coverage requirement for a given set S of sensor nodes is satisfied if and only if each sensor node, $s_i \in S$, is ***covered*** by at least one relay node. A given sensor node, $s_i \in S$, is said to be ***covered*** if and only if $\exists a_j$ is a relay node such that the ***Euclidean distance*** between s_i and a_j holds $\|s_i a_j\| \leq r_i$, where r_i is the communication range of s_i.

Definition 3 **(Connectivity Requirement for a Given Set of Relay Nodes)**:

A given set of relay nodes, each one as a vertex, form an undirected graph. The ***edge*** between two relay nodes, a_i and a_j, is defined if and only if $\|a_i a_j\| \leq R$. And a_i, a_j are said to be ***adjacent*** if there is an edge between them.[1] The connectivity requirement for a given set of relay nodes is satisfied if and only if the undirected graph formed by the set of relay nodes is ***connected***. By ***connected***, it means every relay node in this graph is reachable from all other relay nodes. [2] And it means the same as 1-connected defined in [6] does.

Definition 4 **(Strict Reduction)**[1] [18]:

Given two NPO problems F and G, a strict reduction from F to G w.r.t. a measure $\mathcal{E}$ of approximation quality is a pair $f = (t_1, t_2)$ such that

1. t_1, t_2 are polynomial time computable functions.
2. $t_1 : \mathcal{I}_F \to \mathcal{I}_G \; and \; \forall x \in \mathcal{I}_F \; and \; \forall y \in S_G\big(t_1(x)\big), t_2(x, y) \in S_F(x)$.
3. $\forall x \in \mathcal{I}_F \; and \; \forall y \in S_G\big(t_1(x)\big), \mathcal{E}_F\big(x, t_2(x, y)\big) \leq \mathcal{E}_G(t_1(x), y)$.

If there is a strict reduction from F to G we write $F \leq_s^p$G, and if at the same time $G \leq_s^p$F holds as well, then we write $F \equiv_s^p$G.

Note: $\mathcal{I}$ is the set of instances of a given problem. S is the set of feasible solutions given an instance of a problem.

Definition 5 **(Overlap)**

In the Euclidean plane $\mathbb{R}^2$, given a set $\mathcal{C} = \{\langle c_1, r_1\rangle, \ldots, \langle c_n, r_n\rangle\}$ of circles, where $c_i \in \mathbb{R}^2$ is the center of the ith circle and $r_i \in \mathbb{R} \wedge r_i > 0$ is the radius of this circle, an ***overlap*** is defined as a set $\mathcal{O}$ of points such that, $\mathcal{O}^{\{c_1,\ldots,c_k\}} = \big\{\mathcal{p} \in \mathbb{R}^2 | f_{c_1}(\mathcal{p}) = true \wedge \ldots \wedge f_{c_k}(\mathcal{p}) = true\big\}$, where $c_j \in \mathcal{C}$, $k \leq n$, and f_{c_j} represents the corresponding predicate function that returns true if the input point, $\mathcal{p}$, is inside or on the border of the circle. f_{c_j} can be easily implemented by using the Euclidean distance, e.g. $f_{c_j}(\mathcal{p}) = \begin{cases} true, if \|c_j \mathcal{p}\| \leq r_j \\ false, ow \end{cases}$.

[1] This definition will be explained below.

The superscript $\{c_1, \dots, c_k\}$ of $\mathcal{O}$ represents the set of circles that form the *overlap*, and we say these *k* circles are ***overlapping***. ***k*** is the cardinality of the circle set, and it is called the ***degree*** of this *overlap*.

Definition 6 **(Maximum Overlap)**

In the Euclidean plan $\mathbb{R}^2$, given an *overlap* $\mathcal{O}^{\{c_1,\dots,c_k\}}$, if there does not exist any $c_i \in \mathcal{C}\backslash\{c_1, \dots, c_k\}$ such that $\exists\, p\left(p \in \mathcal{O}^{\{c_1,\dots,c_k\}} \wedge f_{c_i}(p) = true\right)$, then the *overlap* $\mathcal{O}^{\{c_1,\dots,c_k\}}$ is called a ***maximum overlap***.

Problem 1 (**Two-Tiered Relay Node Placement with Non-Uniform Communication Ranges of Sensor Nodes (2tRNP-NUCRSN) Problem**):

Given:

1. The communication range assumptions.
2. A set of sensor nodes, $S = \{s_1, s_2, \dots, s_N\}$ in the Euclidean plane, $\mathbb{R}^2$.

Seek:

A placement of a set of relay nodes, $D = \{d_1, d_2, \dots, d_M\}$, such that,

1. The coverage requirement for S is satisfied.
2. The connectivity requirement for D is satisfied.
3. The cardinality of D, denoted by $|D|$, is minimum.

Problem 2 (**Minimum Relay Node Cover (MRNC) problem**):

Given:

1. The communication range assumptions.
2. A set of sensor nodes, $S = \{s_1, s_2, \dots, s_N\}$ in the Euclidean plane, $\mathbb{R}^2$.

Seek:

A placement of a set of relay nodes, $C = \{c_1, c_2, \dots, c_M\}$, such that,

1. The coverage requirement for C is satisfied.
2. The cardinality of C, denoted by $|C|$, is minimum.

Problem 3 **(Minimum Geographic Disk Cover (MGDC) problem)**:

Given:

1. A set of points, $X = \{x_1, x_2, \dots, x_N\}$, in the Euclidean plane $\mathbb{R}^2$.
2. A constant value r as the radius of disk.

Seek:

A set of disks $B = \{b_1, b_2, \dots, b_K\}$ such that

1. for each point $x_i \in X$, there exists a point $b_j \in B$ such that $\|x_i b_j\| \leq r$.
2. The cardinality of B, denoted by $|B|$, is minimum.

Problem 4 **(Minimum Vertex Disjoint Cycle Cover (MVDCC) Problem)**:

Given:

A graph $G = \langle V, E \rangle$.

Seek:

A family F of vertex disjoint cycles covering V, and $|F|$ is minimum.

Problem 5 **(Minimum Clique Partition (MCP) Problem)**:

Given:

A graph $G = \langle V, E \rangle$.

Seek:

A clique partition for G, i.e. a partition of V into disjoint subsets $V_1, V_2, \ldots, V_k$ such that, for $1 \leq i \leq k$, the subgraph induced by V_i is a complete graph, and the cardinality of the clique partition is minimum.

A Straightforward Method For MRNC

Method

One rather straightforward method to tackle the *MRNC* problem is to find all overlaps formed by the sensors. A feasible relay node coverage will be easily found by placing a relay node at an arbitrary point on each overlap. For each order in which the overlaps being gone through, there will be a corresponding coverage, and for each two coverages found, they may not the same. By enumerating all feasible coverages, the one (or ones) of the minimum cardinality is the optimal solution.

Algorithm 1 A Straightforward Method for *MRNC*

INPUT: A set of n sensor nodes $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ in the Euclidean plane. For each $x_i \in \mathcal{X}$, there is a communication range $0 < r_i \leq R$ associated, where R represents the communication range of relay nodes.

OUTPUT: A set $\mathcal{D} = \{d_1, \ldots, d_k\}$ of relay nodes.

Step 1: In the Euclidean plane, for each sensor node $x_i \in \mathfrak{X}$, it can be represented as a circle by given the center of the node and the communication range r_i.

Step 2: Find all *maximum overlaps*. We call the collection of them the set $\mathfrak{M} = \{\mathcal{O}_1, \ldots, \mathcal{O}_m\}$, where each $\mathcal{O}_i$ is a *maximum overlap*.

Step 3: Generate all possible permutations of $\mathfrak{M}$. We call the collection of the permutations the set $\mathfrak{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_l\}$, where $\mathcal{P}_i$ is a sequence of *maximum overlaps* in a certain order, $|\mathcal{P}_i| = |\mathfrak{M}|$ and $l = |\mathfrak{M}|!$.

Step 4: Find the corresponding coverage for each permutation.

Let $\mathfrak{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_l\}$ be the set of relay node coverages, i.e. for each $\mathcal{P}_i \in \mathfrak{P}$, there will be a $\mathcal{D}_i$ associated. Initially, for each $\mathcal{D}_i \in \mathfrak{D}$, $\mathcal{D}_i = \emptyset$.

1. **for each** $\mathcal{P}_i \in \mathfrak{P}$, **do**
2. **for each** $\mathcal{O}_j^{\mathcal{A}_j} \in \mathcal{P}_i$, **do**
3. Arbitrarily pick a point $p \in \mathcal{O}_j^{\mathcal{A}_j}$.
4. $\mathcal{D}_i = \mathcal{D}_i \cup \{p\}$.
5. **for each** $\mathcal{O}_q^{\mathcal{A}_q} \in \mathcal{P}_i \backslash \left\{\mathcal{O}_j^{\mathcal{A}_j}\right\}$, **do**
6. **for each** $c_h \in \mathcal{A}_q$, **do**
7. **if** $f_{c_h}(p) = true$, **then**
8. $\mathcal{A}_q = \mathcal{A}_q \backslash \{c_h\}$.

9. **if** $\mathcal{O}_q^{\mathcal{A}_q} = \emptyset$, **then**
10. $\mathcal{P}_i = \mathcal{P}_i \backslash \left\{\mathcal{O}_q^{\mathcal{A}_q}\right\}$.
11. **endif**
12. **endif**
13. **endfor**
14. **endfor**
15. $\mathcal{P}_i = \mathcal{P}_i \backslash \left\{\mathcal{O}_j^{\mathcal{A}_j}\right\}$.
16. **endfor**
17. **endfor**

Step 5: Output $\mathcal{D}_i \in \mathfrak{D}$ of the minimum cardinality.

Note: To implement line 3, we can take advantage of the $Overlapping(.,.)$ storing an intersection as the picked point, or if there is a circle contained by all other circles in an overlap, then we take the center of this circle as the picked point.

Complexity

It is obvious to see that the time complexity is beyond polynomial, and in fact determining if a given a set of circles are overlapping together cannot be done in polynomial time.

Find Maximum Overlaps

FACT 1: Given a set of sensor nodes and their distribution in the Euclidean plane, the set of *maximum overlaps* is unique.

FACT 2: Given a set $\mathcal{C} = \{\langle c_1, r_1\rangle, \dots, \langle c_n, r_n\rangle\}$ of circles in the Euclidean plane $\mathbb{R}^2$, if an *overlap* $\mathcal{O}^{\{c_1,\dots,c_k\}}$ and a circle $c_i \in \mathcal{C}\backslash\{c_1, \dots, c_k\}$[2] are overlapping, then $\exists p\left[p \in \mathcal{O}^{\{c_1,\dots,c_k\}} \wedge \left(f_{c_1}(p) = true \wedge \dots \wedge f_{c_k}(p) = true \wedge f_{c_i}(p) = true\right)\right]$.

FACT 3: Given an *overlap* $\mathcal{O}^{\mathcal{A}}$ and a circle c_l, if $\mathcal{O}^{\mathcal{A}\cup\{c_l\}} \neq \emptyset$, then either the circle c_l contains $\mathcal{O}^{\mathcal{A}}$ completely or it contributes an "edge"[3] to the new *overlap* $\mathcal{O}^{\mathcal{A}\cup\{c_l\}}$.

Theorem 1: Given a set of circles in the Euclidean plane, a non-empty *overlap* and a circle not in the circle set of the *overlap* and the radius of the circle is less or equal to the radius of anyone in the circle set, then the circle and the *overlap* are *overlapping* iff either the circle is contained by everyone in the circle set or at least one of the intersections (one or both) of the circle and someone in the circle set is inside or on the border of all other circles in the circle set.

Let $\mathcal{C} = \{\langle c_1, r_1\rangle, \dots, \langle c_n, r_n\rangle\}$ be the given set of circles in the Euclidean plane $\mathbb{R}^2$.

Let $\mathcal{O}^{\{c_1,\dots,c_k\}} \neq \emptyset$ be the given *overlap*.

Let $\langle c_l, r_l\rangle \in \mathcal{C}\backslash\{c_1, \dots, c_k\}$ be the given circle. For any $c_i \in \{c_1, \dots, c_k\}$, its corresponding radius $r_i \geq r_l$.

We are trying to prove:

---

[2] The notation of the set of circles is abused slightly here. For convenience, we represent circles in the set as c_i instead of $\langle c_i, r_i\rangle$, but it is still consistent with how it has been used above in terms of its meaning.

[3] Here "edge" means an arc instead of a straight line segment, and in some special cases, it could be only a point.

$$\exists \mathcal{O}^{\{c_1,\dots,c_{k+1}\}}\left(\mathcal{O}^{\{c_1,\dots,c_{k+1}\}} = \mathcal{O}^{\{c_1,\dots,c_k\}\cup\{c_l\}}\right)$$
$$\Leftrightarrow \left[\forall p\left(f_{c_l}(p) = true\right) \Rightarrow \forall c_b\left(c_b \in \{c_1,\dots,c_k\} \wedge f_{c_b}(p) = true\right)\right]$$
$$\vee \left[\exists c_h\left(c_h \in \{c_1,\dots,c_k\} \wedge Intersect(c_l, c_h) \neq \emptyset\right)\right.$$
$$\left.\wedge \exists q_j \left(q_j \in Intersect(c_l, c_h) \wedge \forall c_d\left(c_d \in \{c_1,\dots,c_k\}\backslash\{c_h\} \wedge f_{c_d}(q_j) = true\right)\right)\right]$$

where $Intersect(.,.)$ is a function takes two circles as input, and outputs a set of their intersection(s) if exists, otherwise it outputs an empty set.

Figure [1][4] gives an example.

Algorithm 2 A Method to Find Maximum Overlaps

INPUT: A set of n sensor nodes $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$ in the Euclidean plane.

OUTPUT: The set $\mathfrak{M} = \{\mathcal{O}_1, \dots, \mathcal{O}_m\}$ of maximum overlaps.

Step 1: Sort the given set of sensor nodes by their radii in descending order.

Step 2: Find all maximum overlaps

Let $\mathcal{C} = \{\langle c_i, r_i\rangle\}$ be the set of circles representing sensor nodes in the Euclidean plane $\mathbb{R}^2$.

1. **for each** $c_i \in \mathcal{C}$, **do**
2. $\quad \mathfrak{M} = \mathfrak{M} \cup \left\{\mathcal{O}^{\{c_i\}}\right\}$
3. **endfor**
4. **for each** $\mathcal{O}_j^{\mathcal{A}_j} \in \mathfrak{M}$, **do**
5. $\quad$ **for each** $c_k \in \mathcal{C}\backslash\mathcal{A}_j$, **do**
6. $\quad\quad$ **if** $Overlapping\left(c_k, \mathcal{O}_j^{\mathcal{A}_j}\right) = true$, **then**
7. $\quad\quad\quad \mathcal{A}_j = \mathcal{A}_j \cup \{c_k\}$.
8. $\quad\quad$ **endif**
9. $\quad$ **endfor**
10. $\quad$ **if** $\mathcal{O}_j^{\mathcal{A}_j} \in \mathfrak{M}\backslash\left\{\mathcal{O}_j^{\mathcal{A}_j}\right\}$, **then**
11. $\quad\quad \mathfrak{M} = \mathfrak{M}\backslash\left\{\mathcal{O}_j^{\mathcal{A}_j}\right\}$.
12. $\quad$ **endif**
13. **endfor**

Theorem 2: In the Euclidean plane, given an *overlap* $\mathcal{O}^{\mathcal{A}} \neq \emptyset \wedge |\mathcal{A}| \geq 4$ and a circle c_l,

$$\mathcal{O}^{\mathcal{A}\cup\{c_l\}} \neq \emptyset \Leftrightarrow \forall \mathcal{B}_i\left(\mathcal{B}_i \subset \mathcal{A} \cup \{c_l\}\backslash\emptyset \wedge \mathcal{O}^{\mathcal{B}_i} \neq \emptyset\right).$$

It means that for any given overlap, any subset of the overlapping circles should also be overlapping.

Corollary 1: In the Euclidean plane, given an *overlap* $\mathcal{O}^{\mathcal{A}} \neq \emptyset$, then

$$\forall \mathcal{B}_i\left[(\mathcal{B}_i \subset \mathcal{A} \wedge \mathcal{B}_i \neq \emptyset) \Longrightarrow \left(\mathcal{O}^{\mathcal{A}} \subseteq \mathcal{O}^{\mathcal{B}_i}\right)\right].$$

It means that for any given overlap, it should be a subset of any overlaps formed by a subset of the overlapping circles.

[4] All figures in this report can be found in the Appendix.

Lemma 1: In the Euclidean plan, given an *overlap* $\mathcal{O}^{\mathcal{A}} \neq \emptyset \ \wedge \ |\mathcal{A}| \geq 4$ and a circle c_l,

$$\mathcal{O}^{\mathcal{A}\cup\{c_l\}} \neq \emptyset \Leftrightarrow \forall c_i \forall c_j \Big(c_i \in \mathcal{A} \cup \{c_l\} \wedge c_j \in \mathcal{A} \cup \{c_l\} \wedge i \neq j \wedge \mathcal{O}^{\mathcal{A}\cup\{c_l\}\backslash\{c_i\}} \neq \emptyset \wedge \mathcal{O}^{\mathcal{A}\cup\{c_l\}\backslash\{c_j\}} \neq \emptyset \wedge \mathcal{O}^{\mathcal{A}\cup\{c_l\}\backslash\{c_i\}} \cap \mathcal{O}^{\mathcal{A}\cup\{c_l\}\backslash\{c_j\}} \neq \emptyset \Big).$$

It means that for any two subsets of a given overlap, they should overlap together.

Algorithm 3 Determine Overlapping Between Circle and Overlap ($Overlapping\big(c_l, \mathcal{O}^{\mathcal{A}}\big)$)

INPUT: An *overlap* $\mathcal{O}^{\mathcal{A}} \neq \emptyset \ \wedge \ |\mathcal{A}| \geq 4$ and a circle c_l

OUTPUT: True if $\mathcal{O}^{\mathcal{A}\cup\{c_l\}} \neq \emptyset$, False otherwise.

Step 1: Sort $\mathcal{A} \cup \{c_l\}$ by radius in descending order, we call the sorted set $\mathcal{B}$.

Step 2:

1. Take the circle c_s of the smallest radius in $\mathcal{B}$
2. **if** $c_s = c_l$, **then**
3. **return** $Overlapping\_s(c_s, \mathcal{B}\backslash\{c_s\})$
4. **else**
5. Take the circle c_s' of the smallest radius in $\mathcal{B}\backslash\{c_s\}$
6. **if** $Overlapping\_s(c_s', \mathcal{B}\backslash\{c_s, c_s'\}) = True$, **then**
7. **return** $Overlapping\_s(c_s, \mathcal{B}\backslash\{c_s\})$
8. **else**
9. **return** $False$
10. **endif**
11. **endif**

Explanation

Actually there is another way to think about this problem. Generally speaking, the only task is that given a set of circles we need to determine if they could overlap together. Based on the *Theorem 2*, to accomplish this task we have to know if every two circles, every three circles, until every $n-1$ circles could overlap together. Therefore, the dynamic programming could be introduced to construct the algorithm. For example, if we have five circles, A, B, C, D and E, we need to determine if they overlap together. For every two circles and every three circles, we have to compute them manually. These two steps can be done in polynomial time. Particularly, for the every three circles case, the reason we have to do that is there is a chance that three circles do overlap in pairs but not overlap together. For the rest computation, e.g. to determine if A, B, C and D are overlapping together, what we need to do is to check if ABC, ABD, ACD and BCD are all overlaps. If so, then ABCD is an overlap according to the *Theorem 2*. However, w.r.t. the time complexity, since we have to go through all combinations of circles, then the running time in the worst case can be exponential.

Algorithm 4 Determine Overlapping Between Small Circle and Overlap ($Overlapping\_s\big(c_s, \mathcal{O}^{\mathcal{A}}\big)$)

INPUT: A given circle c_s, a given overlap $\mathcal{O}^{\mathcal{A}}$, and $c_s \notin \mathcal{A}$.

OUTPUT: $True$ if $\mathcal{O}^{\mathcal{A}\cup c_s} \neq \emptyset$, $False$ otherwise.

Step 1: Let $\mathfrak{I}$ be the set of all intersections between c_s and $c_i \in \mathcal{A}$. Let h be the flag to indicate if a specific intersection is contained by all circles in $\mathcal{A}$. Let g be the flag to indicate if the given circle is contained by all circles in $\mathcal{A}$.

Step 2: Initially $\Im = \emptyset$, $h = True$, $g = True$.

1. **if** $\mathcal{A} = \emptyset$, **then**
2. **return** $True$
3. **endif**
4. **for each** $c_i \in \mathcal{A}$, **do**
5. **if** $intersect(c_s, c_i) \neq \emptyset$, **then**
6. $\Im = \Im \cup intersect(c_s, c_i)$
7. **else**
8. **if** $(\|c_s c_i\| \leq r_s + r_i) \wedge (h = True)$, **then**
9. $h = True$
10. **else**
11. $h = False$
12. **endif**
13. **endif**
14. **endfor**
15. **if** $\Im = \emptyset \wedge h = True$, then
16. **return** $True$
17. **endif**
18. **if** $\Im = \emptyset \wedge h = False$, then
19. **return** $False$
20. **endif**
21. **for each** $\mathcal{I}_j \in \Im$, **do**
22. $g = True$
23. **for each** $c_i \in \mathcal{A}$, **do**
24. **if** $f_{c_i}(\mathcal{I}_j) = False$, **then**
25. $g = False$
26. **endif**
27. **Endfor**
28. **if** $g = True$, **then**
29. **return** $True$
30. **endif**
31. **endfor**
32. **return** $False$

Explanation

This algorithm is a direct application of the *Theorem 1*. It determines if a given circle, c_s, that is of a radius smaller than or equal to any circle's radius in the given overlap, $\mathcal{O}^{\mathcal{A}}$, could overlap with $\mathcal{O}^{\mathcal{A}}$. According to the *Theorem 1*, there will only two cases if the new overlap could be established. One is that c_s is fully contained by all circles in $\mathcal{O}^{\mathcal{A}}$, the other one is that there exists at least one intersection between c_s and another circle in $\mathcal{O}^{\mathcal{A}}$ such that this intersection is contained by all circles in $\mathcal{O}^{\mathcal{A}}$.

Therefore, line 4 through line 20 is going through the set of circles of the overlap, $\mathcal{O}^{\mathcal{A}}$, to collect all intersections meanwhile to check if c_s is contained by all circles in the overlap. There is a chance that c_s has no intersections with anyone but it is fully contained by a part of circles, not all of them, in the overlap. This case will be considered as c_s is not overlapping with $\mathcal{O}^{\mathcal{A}}$.

Line 21 through the end goes through the set of intersections to see if there exists such an intersection contained by all circles in the overlap. As long as we could find one, then it is safe to say that c_s is overlapping with $\mathcal{O}^{\mathcal{A}}$.

Complexity

Line 4 through line 20 will cost $O(|\mathcal{A}|)$. $intersect(\cdot,\cdot)$ and $\|\cdot\cdot\|$ will cost constant amount of time. Line 21 through the end will cost $O(|\Im| \cdot |\mathcal{A}|)$. $f_{c_i}(\cdot)$ will cost constant amount of time. $|\Im|$ at most is equal to $2 \cdot |\mathcal{A}|$, which means if the cardinality of $\mathcal{A}$ is n, then the worst case running time of this algorithm is $O(2 \cdot n^2)$.

MRNC vs MGDC

These two problems can be *strictly reduced* to each other. The *strict reduction* is able to preserve the approximability w.r.t. the measure of approximation.[18] From the definition of the *strict reduction*, it is easy to see that t_1 and t_2 are two reduction functions. t_1 transforms instances of F to instances of G, and t_2 transforms feasible solutions to an instance of G to feasible solutions to the corresponding instance of F transformed by t_1. This definition is telling that for a given problem as long as the reduction will never degrade the approximation quality and can be done within polynomial time, then the reduction is strict.

Proposition 1 $MGDC \leq_s^p MRNC$

Proof

t_1 transforms instances of $MGDC$ to instances of $MRNC$. We map points given in a $MGDC$ instance to sensor nodes in the $MRNC$ instance. Here both the points and the sensor nodes are coordinates in the Euclidean plane. For each sensor node in the $MRNC$ instance mapped from $MGDC$, its communication range is equal to r given in the $MGDC$ instance. Here r originally means the radius of disks in the $MGDC$ problem. The radius of relay nodes in the $MRNC$ instance can also be mapped as being equal to r, i.e. $R = r$.

t_2 transforms feasible solutions to the $MRNC$ instance to feasible solutions to the $MGDC$ instance. Therefore, the set of relay nodes as a feasible solution to the $MRNC$ instance can be mapped to the set of disks to the $MGDC$ instance, i.e. the coordinates of disks will be mapped from the coordinates of relay nodes, and for each disk, its radius is r.

Since neither t_1 nor t_2 would be introducing new relay nodes or disks to the solutions, the cardinalities of the two feasible solutions to the $MRNC$ instance and the $MGDC$ instance respectively will be the same. Therefore, the $MGDC$ problem is *strictly reducible* to the $MRNC$ problem.

The Shifting Strategy

General Idea

Given a set of points enclosed in an area $\mathcal{I}$ in a specified space[5], e.g. the Euclidean plane. The goal is to find a minimal number of disks of diameter D to cover all the points. Given a constant $l \geq 1$ called the ***shifting parameter***, we partition $\mathcal{I}$ into vertical strips of width D, and for each strip, it is left closed and right open. Then we make each l consecutive strips into a group. As a result, we have a set of strips of width $l \cdot D$ called a ***shift partition***. Apparently, there are l ways to derive such a *shift partition*, and we can make them in order such that for each one it can be derived from the previous one by shifting it to the right over distance D. Therefore, we will have l distinct shift partitions. Then for each vertical strip, we do the same thing to the other dimension, i.e. $\mathcal{I}$ finally will be divided into a grid. For each square in the grid, we search for a corresponding local optimal solution. Combining all local optimal solutions, we will get a final approximate solution for one *shift partition*. Eventually we pick the best solution among all *shift partitions*, and that will be the output.

[5] Here we explain the shifting strategy in the context of the Euclidean space, though actually it can be applied to arbitrary dimensions spaces.

Why It Works for MGDC

In [3], Hochbaum, Dorit S. et al. gave the ***Shifting Lemma*** and its proof. The *Shifting Lemma* is stated as follows.

The Shifting Lemma

Let A be a local algorithm that delivers a solution in any strip of width $l \cdot D$ (or less). Given a *shift partition* $S_i \in \{S_1, \dots, S_l\}$, $A(S_i)$ means to apply the algorithm A to each strip in S_i and output a set of disks covering all points.

Let S_A be the shift algorithm defined for A delivering the set of disks of the minimum cardinality among $\{A(S_1), \dots, A(S_l)\}$.

Let OPT denote the globally optimal solution, and $|OPT|$ denote the its cardinality.

Let r_B be the supremum of $\frac{Z^B}{|OPT|}$ over all problem instances, i.e. the approximation ratio of the algorithm B.

$$r_{S_A} \leq r_A \cdot \left(1 + \frac{1}{l}\right)$$

Proof

Let Z^B denote the value of the solution delivered by algorithm B, e.g. $Z^{A(S_i)}$ means the cardinality of the set of disks delivered by applying algorithm A to S_i.

Let $A(J)$ be applying the algorithm A to a strip J.

By definition of r_A, since r_A is the least upper bound of the ratio over all instances, then

$$r_A \geq \frac{Z^{A(J)}}{|OPT_J|}$$

It is easy to see that

$$Z^{A(S_i)} = \sum_{J\ in\ S_i} Z^{A(J)} \leq r_A \cdot \sum_{J\ in\ S_i} |OPT_J| \quad (1)$$

where J is the index of strips in S_i, and OPT_J is the local optimal solution for the J^{th} strip.

Let $OPT^{(1)}, \dots, OPT^{(l)}$ be the set of disks in OPT covering points in two adjacent $l \cdot D$ strips in the $1,2, \dots, l$ *shift partitions*. Then the following inequality holds.

$$\sum_{J\ in\ S_i} |OPT_J| \leq |OPT| + |OPT^{(i)}| \quad (2)$$

Here is the reason. If we divide OPT into strips in the exact way as S_i does, then there are two different types of disks. One is only covering points within a certain strip, whereas the other one is covering points across two adjacent strips. The set of second type of disks is $OPT^{(i)}$. It is obvious to see that if $|OPT^{(i)}| = 0$, then $\sum_{J\ in\ S_i} |OPT_J| = |OPT|$, otherwise $\exists OPT_J$ which is not a local optimal solution for J. Hence, the reason that the union of all local optimal solutions is not globally optimal is that there exists disks in two adjacent local optimal solutions can be replaced by only one disk covering same points. Therefore, the local

optimal solutions may have redundant disks in this type. But in terms of one OPT_J, its size should be still the smallest among all feasible solutions to the corresponding strip J, so the inequality holds.

Since the smallest possible strip is of width D, and each strip is left closed and right open, then there will be no disk can cover points across two adjacent strips in more than one *shift partition*. Therefore, the sets $OPT^{(1)}, \dots, OPT^{(l)}$ are disjoint, and obviously $\sum_{i=1}^{l}\left|OPT^{(i)}\right| \le |OPT|$. Then it follows that

$$\sum_{i=1}^{l}\left(|OPT| + \left|OPT^{(i)}\right|\right) \le (1+l)|OPT| \quad (3)$$

From (2) and (3), it is easy to get that

$$\min_{i=1,\dots,l} \sum_{J\ in\ S_i} \left|OPT_J\right| \le \frac{1}{l}\sum_{i=1}^{l}\left(\sum_{J\ in\ S_i}\left|OPT_J\right|\right) \le \frac{1}{l}\sum_{i=1}^{l}\left(|OPT| + \left|OPT^{(i)}\right|\right) \le \frac{(1+l)}{l}|OPT| \quad (4)$$

From (1) and (4), we have

$$Z^{S_A} = \min_{i=1,\dots,l}\left(Z^{A(S_i)}\right) \le \min_{i=1,\dots,l}\left(r_A \cdot \sum_{J\ in\ S_i}\left|OPT_J\right|\right) = r_A \cdot \min_{i=1,\dots,l}\sum_{J\ in\ S_i}\left|OPT_J\right| \le r_A \cdot \frac{(1+l)}{l}|OPT| \quad (5)$$

■

Explanation

The shifting lemma implies that as long as there exists a polynomial time algorithm A that can deliver an optimal solution to a given strip, then a *PTAS* exists for the *MGDC* problem. Now the question is if such an algorithm A exists.

In [3], Hochbaum, Dorit S. et al. gave a brutal search algorithm to accomplish this goal. Actually the most important thing here is to prove such an algorithm is of polynomial running time. Here is the general idea. For a given strip J, we apply the *shifting strategy* again, i.e. the entire plan will be divided into a grid. For each square inside, it should be of $l \cdot D \times l \cdot D$ in size. Then how many disks should we use to cover such a square?

Hochbaum, Dorit S. et al. claimed that with $2 \cdot l^2$ disks of diameter D, we can cover a $l \cdot D \times l \cdot D$ square compactly, but it may not be true. Let us consider a $l \cdot D \times l \cdot D$ square as a $l \times l$ grid, in which each cell is a $D \times D$ square. For each cell, it can be covered by at least two disks of diameter D. As shown in Figure [2], the green square is the largest square that the two disks can cover. The point A is the center of the left disk, the point B is the upper left vertex of the square, the point C is an intersection of the two disks, and the point E is the intersection of BC and the vertical diameter of the left disk. It holds that $AE \perp BC$, $AB = AC = \frac{D}{2}$. Let the length of side of the square be x. Then $AE = \frac{x}{2}$, $CE = \frac{x}{4}$. It is easy to get that $x = \frac{2}{\sqrt{5}}D < D$, which means two disks of diameter D cannot cover any square of side length greater than $\frac{2}{\sqrt{5}}D$. However, from this example, we can also know that four disks of diameter D can cover a square of side length $\sqrt{2}D$. As shown in Figure [3], the red rectangle is a square and is $\frac{1}{4}$ of the green one. Therefore, to cover a $l \cdot D \times l \cdot D$ square, we can use at most $4 \cdot l^2$ disks of diameter D.

To search for the local optimal solution, based on what is stated in the *Proposition 1*, we can use the straightforward method presented above. We consider each point in the plane as the center of a circle of

diameter D, then what we are seeking will be converted to all *maximum overlaps*. For each *maximum overlap*, we arbitrarily find a point, and the set of such points will be the output of this problem.

The running time of the Shifting Strategy is polynomial. The total number of disks need to be considered is bounded by $l^2 \cdot (4 \cdot l^2)$, since in the plane we have l^2 amount of $l \cdot D \times l \cdot D$ squares, and for each square we need at most $4 \cdot l^2$ disks to cover it. As explained above, for each square, we are using the straightforward method to find a local optimal solution, i.e. seeking *maximum overlaps*, and according to the FACT 3, it is obvious to see that each *maximum overlap* must be contained completely by an *overlap* in degree 2. Therefore, there are at most $\binom{n}{2} \approx O(n^2)$ possible places for each disk to go in a square, where n is the number of points in the plane. Then it will cost $O\left(l^2 \cdot n^{2 \cdot 4 \cdot l^2}\right)$ to find a coverage for one *shift partition*. We can make the shift l times on each dimension, hence there are l^2 cases we have to consider. Thereby, the running time of the Shifting Strategy is polynomial.

The key to achieve this result is that the amount of disks need to be considered can be bounded by a constant, $l^2 \cdot (4 \cdot l^2)$. From *the Shifting Lemma*, it is easy to see that the solution will be getting closer to the optimal one while the *shifting parameter* l keeps increasing; however, the squares will be getting larger as well, which means at some point all points in the plane will be contained in a certain square, then the problem becomes NP-Hard again. Decreasing l will lead to more squares so that sub-problems will be easier to solve but the final solution is going further from the optimal one. The core reason *the Shifting Strategy* works for the *MGDC* problem is that it finds a way to divide the entire plane into a constant number, l^2, of cells, and for each cell it can be fully covered by a constant number, $4 \cdot l^2$, of disks so that the local algorithm A can deliver a local optimal solution for this cell within polynomial running time.

Apply the Shifting Strategy to MRNC

CASE 1: Only a constant number of sensor nodes are of the different communication range from others

CASE1-1: The special communication range is equal or smaller than the regular one.

Algorithm [5]

INPUT: A set of sensor nodes $\mathcal{S} = \mathcal{S}_M \cup \mathcal{S}_N$ in the Euclidean plane $\mathbb{R}^2$;

$(\forall s_i \in \mathcal{S}_M, r_i = r_M) \wedge \left(\forall s_j \in \mathcal{S}_N, r_j = r_N\right) \wedge (r_M \geq r_N)$, where r_M and r_N are constants;

$|\mathcal{S}_M| = M, |\mathcal{S}_N| = N$;

N is a constant;

The *Shifting Parameter* $l \in \mathbb{N}$.

OUTPUT: A set of relay nodes fully cover the sensor nodes.

Step 1:

Divide the plane into squares in $l \cdot 2r_M \times l \cdot 2r_M$.

Step 2:

For each square, apply the straightforward method to find a local optimal solution.

Step 3:

Enumerate all shifts, and for each one, go over Step 1 and 2 to get the final solution with the minimum cardinality.

Complexity

To the set $\mathcal{S}_M$, it is a typical application of the *Shifting Strategy*, hence for each square there are $O(M^2)$ overlaps and $4 \cdot l^2$ relay nodes at most need to be considered. However, since the sensor nodes in $\mathcal{S}_N$ are of the smaller communication range than r_M, it is not necessary for the squares to have the upper bound, $4 \cdot l^2$, of the number of relay nodes, e.g. Figure [4]. Therefore, in the worst case, the upper bound will be $(4 \cdot l^2 + N)$. Although this bound has been increased in a certain amount, $(4 \cdot l^2 + N)$ is still constant because N is constant. The maximum number of overlaps need to be considered for $\mathcal{S}_M \cup \mathcal{S}_N$ actually will be, more precisely, $O((M+N)^2)$, though it is still equal to $O(M^2)$ in this case. That means the complexity will not rise significantly.

Approximation Ratio

Let us reexamine the proof of the *Shifting Lemma*. OPT here means an optimal solution, i.e. a set of relay nodes of the minimum cardinality fully covering all the sensor nodes. According to the straightforward method, correspondingly there exists a set of *maximum overlaps*. Each relay node will locate in an individual overlap. OPT_J is the subset of OPT in the strip J, and also there is a corresponding subset of the optimal *maximum overlaps*. A relay node in $OPT^{(i)}$ means that this node can cover sensor nodes in two adjacent strips. In other words, correspondingly there will be at least two sensor circles in two adjacent strips overlapping, e.g. Figure [5]. Those overlaps are *maximum overlaps*, and the corresponding relay nodes locate in them are elements in OPT. For each sensor circle, it will not overlap with others across two adjacent strips. Therefore, the proof is still working, and the approximation ratio will not change.

Algorithm [6]

INPUT: A set of sensor nodes $\mathcal{S} = \mathcal{S}_M \cup \mathcal{S}_N$ in the Euclidean plane $\mathbb{R}^2$;

$(\forall s_i \in \mathcal{S}_M, r_i = r_M) \wedge (\forall s_j \in \mathcal{S}_N, r_j = r_N) \wedge (r_M \geq r_N)$, where r_M and r_N are constants;

$|\mathcal{S}_M| = M, |\mathcal{S}_N| = N$;

N is a constant;

The *Shifting Parameter* $l \in \mathbb{N}$.

OUTPUT: A set of relay nodes fully cover the sensor nodes.

Step 1: Divide the plane into squares in $l \cdot 2r_N \times l \cdot 2r_N$.

Step 2 and 3 are the same as the *Algorithm [5]*.

Complexity

For each $l \cdot 2r_N \times l \cdot 2r_N$ square, we still need to consider $4 \cdot l^2$ relay nodes, because $r_M \geq r_N$. W.r.t. the number of overlaps need to be considered, it is also $O((M+N)^2) = O(M^2)$ at most. Therefore, the running time basically will be the same as what it is when applying the *Shifting Strategy* to the typical *MGDC* problem in terms of complexity degree.

Approximation Ratio

The errors occur typically when the local search cannot find all *maximum overlaps*, which means some sensors can actually overlap together across several strips but the local search is not able to seek these overlaps out in such a wide range, instead the sensors will only be considered within their own strips. To evaluate the approximation performance we need to know how many such errors at most could occur when our algorithm doing the best facing the worst case.

Since $r_M \geq r_N$, it would not be necessarily true that no sensor circles will not overlap with others across two or more adjacent strips, e.g. Figure [6]. As a result of this fact, $OPT^{(1)}, \ldots, OPT^{(l)}$ may not be disjoint sets, thus $\sum_{i=1}^{l}\left|OPT^{(i)}\right| \leq |OPT|$ may not hold as well. Therefore, the *Shifting Lemma* may not hold in this case.

In the proof of the *Shifting Lemma*, it requires that no disk in OPT can cover points in two adjacent strips in more than one shifts. Actually it means that errors will only occur in $OPT^{(i)}$, i.e. for each relay node in $OPT^{(i)}$, there might be at most one superfluous relay nodes associated existing in the final solution. However, in this case, more than one superfluous relay nodes might occur. In other words, $OPT^{(i)}$ directly reflects the errors for the ith shift, and $\bigcup_{i=1}^{l} OPT^{(i)}$ is the set of errors over all shifts. Then the question is what the upper bound of $\sum_{i=1}^{l}\left|OPT^{(i)}\right|$ is. Before the following analysis going on, it is more appropriate to express the errors by using another notation instead of $OPT^{(i)}$. Thus, let the set of errors in the ith shift be $ERR^{(i)}$. Then (4) can be modified to

$$\min_{i=1,\ldots,l} \sum_{J\ in\ S_i} \left|OPT_J\right| \leq \frac{1}{l}\sum_{i=1}^{l}\left(\sum_{J\ in\ S_i}\left|OPT_J\right|\right) \leq \frac{1}{l}\sum_{i=1}^{l}\left(|OPT| + \left|ERR^{(i)}\right|\right) \quad (4*).$$

What (4*) expresses is that the value of the best solution is bounded the average value of the summation of the value of the optimal solution and the amount of errors. The errors part obviously catches the most focus. There are two questions need to be answered.

1. Given a relay node in the optimal solution, how many shifts will there be where the errors caused by it may occur when doing the local search?
2. How many errors will there be in each such shift?

Let the given relay node be $d_i \in OPT$, then the corresponding *maximum overlap* be $\mathcal{O}_i^{\mathcal{A}_i}$. Let the set of shifts where the errors occur be $\mathfrak{F}_i$, and the number of errors in jth shift be $\mathbb{E}_i^j$. Then, the total number of errors over all shifts when doing the local search will be

$$\sum_{k=1}^{l}\left|ERR^{(k)}\right| = \sum_{i=1}^{|OPT|}\left(\sum_{\mathfrak{f}\in\mathfrak{F}_i}\mathbb{E}_i^{\mathfrak{f}}\right) \quad (4**).$$

Therefore, from (4*) and (4**), it holds that

$$\min_{i=1,\ldots,l} \sum_{J\ in\ S_i} \left|OPT_J\right| \leq \frac{1}{l}\sum_{i=1}^{l}\left(\sum_{J\ in\ S_i}\left|OPT_J\right|\right) \leq |OPT| + \frac{1}{l}\cdot\sum_{i=1}^{|OPT|}\left(\sum_{\mathfrak{f}\in\mathfrak{F}_i}\mathbb{E}_i^{\mathfrak{f}}\right) \quad (4***).$$

For the first question, let s_l and s_r be the leftmost and the rightmost sensors given a *maximum overlap* corresponding to a relay node in an optimal solution. If $\|s_l s_r\| > l \cdot D$, where D is the width of the unit strip, then the errors would occur in all shifts; otherwise, if $\|s_l s_r\| \leq (l - x) \cdot D$, where $1 \geq x \geq 0$, then the number of shifts where the errors occur will be bounded by $l - x - 1$. In Figure [7], it shows two examples for these two cases respectively.

For the second question, if $\|s_l s_r\| \leq l \cdot D$, then there will be at most one error for a given overlap occurs in one shift; otherwise, if $\|s_l s_r\| > l \cdot D$, there will be $\min\left(\mathbb{E}_i^j, |\mathcal{A}_i|\right)$ errors at most. Figure [8] shows an example.

In a simpler case where for a maximum overlap in a globally optimal solution, $O^{\mathcal{A}_i}$, $|\mathcal{A}_i| = 2$. If we consider the overlap as a kind of connection between two sensor nodes, then the overlap corresponding to a relay node in $ERR^{(i)}$ means that this connection might be cut off when searching for local optimal solutions. It is obvious that if two sensors, s_M and s_N, have $\|s_M s_N\| \leq D$, then there is only one chance at most that the overlap of them would be cut off over all shifts. That is why in the proof of the *Shifting Lemma* it holds that $\sum_{i=1}^{l}|ERR^{(i)}| \leq |OPT|$. However, along with the increasing of $\|s_M s_N\|$, the chance is getting greater.

Let $ERR^{(i)} = ERR_{IO}^{(i)} \cup ERR_{IR}^{(i)}$, where $ERR_{IO}^{(i)}$ represents the set of relay nodes that cover only irregular sensor nodes in two adjacent strips in the ith shift, and $ERR_{IR}^{(i)}$ is for relay nodes that cover regular sensor nodes including the regular-irregular and the regular-regular cases. Then, we have $|ERR^{(i)}| = |ERR_{IO}^{(i)}| + |ERR_{IR}^{(i)}|$, and apparently $\sum_{i=1}^{l}|ERR^{(i)}| = \sum_{i=1}^{l}|ERR_{IO}^{(i)}| + \sum_{i=1}^{l}|ERR_{IR}^{(i)}|$. As known, $\sum_{i=1}^{l}|ERR_{IO}^{(i)}| \leq |OPT|$. In the worst case, every overlap formed by at least one large sensor node will be torn apart in all shifts, then $|ERR_{IR}^{(i)}| \leq |OPT|$. Hence, $\sum_{i=1}^{l}|ERR_{IR}^{(i)}| \leq l \cdot |OPT|$. It turns out that

$$\sum_{i=1}^{l}|ERR^{(i)}| \leq |OPT| + l \cdot |OPT| \quad (6)$$

$$\sum_{i=1}^{l}(|OPT| + |ERR^{(i)}|) \leq (1+l)|OPT| + l \cdot |OPT| \quad (7)$$

$$\min_{i=1,\dots,l} \sum_{J\ in\ S_i} |OPT_J| \leq \frac{1}{l}\sum_{i=1}^{l}\left(\sum_{J\ in\ S_i}|OPT_J|\right) \leq \frac{1}{l}\sum_{i=1}^{l}(|OPT| + |ERR^{(i)}|) \leq \frac{(1+2\cdot l)}{l}|OPT| \quad (8)$$

This result is not exactly a PTAS, but still bounded by a constant ratio. The reason we have $|ERR_{IR}^{(i)}| \leq |OPT|$ is that in this simpler case, $|\mathcal{A}_i| = 2$. For each maximum overlap, in one shift, there could only occur one error of it; however, if we generalize this case to $|\mathcal{A}_i| = O(M+N)$, then the amount of errors of an overlap in one shift may increase. Figure [8] shows an example case. Without loss of generality, we assume that a *maximum overlap* in the globally optimal solution, $O^{\mathcal{A}_i}$, $|\mathcal{A}_i| = O(M+N)$, and it consists of X_R regular sensors and X_I irregular sensors, i.e. $|\mathcal{A}_i| = X_R + X_I$. If we did not pick a good *Shifting Parameter*, l, then it is possible that for each regular sensor it will be peeled off from the overlap in every shift when searching local optimal solutions. The set of irregular sensors will have one error overall. Therefore, the total amount of errors over all shifts could be $l \cdot X_R + 1$ at most. The upper bound of $|\mathcal{A}_i|$ is $O(M+N)$. Hence, the upper bound of the total amount of errors for one overlap over all shifts will be $O(l \cdot (M+N) + 1)$ when nearly all members of this overlap are regular sensors. This result could make the ratio be $O(n)$.

However, that is not the best our algorithm could do. The approximation ratio measures how good an algorithm could do when facing the worst case instance. Since the *Shifting Parameter* is an input not a constant value, then there are chances to modify this value to make the algorithm perform as good as possible. The major reason we got a bad result above is that the width of strips is not wide enough to cover some of the sensors in the overlap so that they will be peeled off when doing local search. As discussed above, for a pair of sensors in an overlap, if $\|s_1 s_2\| \leq (l-x)\cdot D \wedge \|s_1 s_2\| > (l-x-1)\cdot D$, the max

number of shifts that errors could occur will be $\left\lceil \left| \frac{\|s_1 s_2\|}{D} - 1 \right| \right\rceil$, where $D = 2 \cdot r_N$ in this case. In other words, the number of unit strips between s_1 and s_2 determines how many errors would occur for them.

Here is a way to compute the overall errors for a given maximum overlap, $\mathcal{O}_j^{\mathcal{A}_j}$.

Step 1: Find the leftmost and the rightmost sensors of this overlap. Let them be s_{lm}^j and s_{rm}^j.

Step 2: Count the number of $l \cdot D$ strips to cover the overlap in each shift, e.g. Figure [9]. Let the set of numbers be $\mathcal{E}_j = \{e_j^1, e_j^2, \dots\}$, where e_j^i is for the ith shift.

Step 3: The total number of errors for one overlap over all shifts will be $\mathbb{E}_j = \sum_{e_j^i \in \mathcal{E}_j} (e_j^i - 1)$.

It can be easily seen that

$$e_j^i \leq \left\lceil \frac{\|s_{lm}^j s_{rm}^j\|}{l \cdot D} \right\rceil + 1 \quad (9).$$

Hence, based on the two questions mentioned above, the upper bound of $\mathbb{E}_j$ will be

$$\mathbb{E}_j \leq |\mathcal{E}_j| \cdot \max_j \left(\left\lceil \frac{\|s_{lm}^j s_{rm}^j\|}{l \cdot D} \right\rceil \right) \leq l \cdot \max_j \left(\left\lceil \frac{\|s_{lm}^j s_{rm}^j\|}{l \cdot D} \right\rceil \right) \quad (10).$$

Then, for all corresponding maximum overlaps in OPT, it holds that

$$\sum_{i=1}^{l} \left(|OPT| + |ERR^{(i)}| \right) \leq l \cdot |OPT| + \sum_{j=1}^{|OPT|} \mathbb{E}_j \leq \left[l + l \cdot \max_j \left(\left\lceil \frac{\|s_{lm}^j s_{rm}^j\|}{l \cdot D} \right\rceil \right) \right] \cdot |OPT| \quad (11).$$

From (4***), it follows that

$$\begin{aligned} \min_{i=1,\dots,l} \sum_{J \text{ in } S_i} |OPT_J| &\leq \frac{1}{l} \sum_{i=1}^{l} \left(\sum_{J \text{ in } S_i} |OPT_J| \right) \leq |OPT| + \frac{1}{l} \cdot \sum_{j=1}^{|OPT|} (\mathbb{E}_j) \\ &\leq |OPT| + \left[\max_j \left(\left\lceil \frac{\|s_{lm}^j s_{rm}^j\|}{l \cdot D} \right\rceil \right) \right] \cdot |OPT| = \left[1 + \max_j \left(\left\lceil \frac{\|s_{lm}^j s_{rm}^j\|}{l \cdot D} \right\rceil \right) \right] \cdot |OPT| \quad (12). \end{aligned}$$

In (12), $|OPT| + \frac{1}{l} \cdot \sum_{j=1}^{|OPT|} (\mathbb{E}_j)$ actually means the average value of the final solution. This value consists of two parts, one is an optimal solution, the other part is for the errors. Its upper bound, $\left(1 + \left\lceil \frac{\|s_{lm}^j s_{rm}^j\|}{l \cdot D} \right\rceil \right) \cdot |OPT|$, also supports this point. Now that the second part is for the errors, then it should be the average value of the errors could occur in one shift. Naturally, it should be presented in the form $\frac{\sum_k \mathbb{E}^k}{l}$, where $\mathbb{E}^k$ is the number of errors for the kth shift. If thinking about the upper bound got in (12) more carefully, it is easy to find that $\max_j \left(\left\lceil \frac{\|s_{lm}^j s_{rm}^j\|}{D} \right\rceil \right)$ means for a given overlap the maximum number of errors could occur over all shifts, which in fact is a tighter bound than what we found in (10). Therefore, the average value of

the errors will be $\frac{\max_j\left(\left\lceil\frac{\left\|s_{lm}^j s_{rm}^j\right\|}{D}\right\rceil\right)}{l}$. From (4***), $\frac{1}{l}\cdot\sum_{i=1}^{|OPT|}\left(\sum_{\not{f}\in\mathfrak{F}_i}\mathbb{E}_i^{\not{f}}\right)$ means the average number of errors for one shift. In the worst case, $|\mathfrak{F}_i|=l$, which means the errors may occur in all shifts. Hence,

$$\frac{1}{l}\cdot\sum_{i=1}^{|OPT|}\left(\sum_{\not{f}\in\mathfrak{F}_i}\mathbb{E}_i^{\not{f}}\right)\le\frac{1}{l}\cdot\sum_{i=1}^{|OPT|}\left(l\cdot\max_{\not{f}\in\mathfrak{F}_i}\mathbb{E}_i^{\not{f}}\right)=\sum_{i=1}^{|OPT|}\left(\max_{\not{f}\in\mathfrak{F}_i}\mathbb{E}_i^{\not{f}}\right),$$

and from (12), it is obvious to see that

$$\min_{i=1,\dots,l}\sum_{J\ in\ S_i}\left|OPT_J\right|\le|OPT|+\sum_{i=1}^{|OPT|}\left(\max_{\not{f}\in\mathfrak{F}_i}\mathbb{E}_i^{\not{f}}\right)\le\left[1+\frac{\max_j\left(\left\lceil\frac{\left\|s_{lm}^j s_{rm}^j\right\|}{D}\right\rceil\right)}{l}\right]\cdot|OPT|\quad(13).$$

From (13), if $r_N=r_M$, then it is obvious to see that $\left\lceil\frac{\left\|s_{lm}^j s_{rm}^j\right\|}{D}\right\rceil=1$, since $\left\|s_{lm}^j s_{rm}^j\right\|\le 2r_M$ and $D=2r_N$. Hence,

$$\min_{i=1,\dots,l}\sum_{J\ in\ S_i}\left|OPT_J\right|\le\left[1+\frac{\max_j\left(\left\lceil\frac{\left\|s_{lm}^j s_{rm}^j\right\|}{D}\right\rceil\right)}{l}\right]\cdot|OPT|\le\left(1+\frac{1}{l}\right)\cdot|OPT|\quad(14),$$

which is consistent with (4). For our *CASE 1-1*, the result would worsen if $\left\|s_{lm}^j s_{rm}^j\right\|$ increases; moreover, since $\left\|s_{lm}^j s_{rm}^j\right\|\le 2r_M$, it means that if $\frac{r_M}{r_N}$ getting larger, then the result could change from a PTAS to a constant ratio approximation, when $\frac{\max_j\left(\left\lceil\frac{\left\|s_{lm}^j s_{rm}^j\right\|}{D}\right\rceil\right)}{l}=k$, where k is a constant, and to even worse ones, e.g. a polynomial ratio when k becomes a polynomial.

CASE 1-2: The special communication range is greater than the regular one.

Algorithm [7]

INPUT: A set of sensor nodes $\mathcal{S}=\mathcal{S}_M\cup\mathcal{S}_N$ in the Euclidean plane $\mathbb{R}^2$;

$(\forall s_i\in\mathcal{S}_M, r_i=r_M)\wedge\left(\forall s_j\in\mathcal{S}_N, r_j=r_N\right)\wedge(r_M<r_N)$, where r_M and r_N are constants;

$|\mathcal{S}_M|=M, |\mathcal{S}_N|=N$;

N is a constant;

The *Shifting Parameter* $l\in\mathbb{N}$.

OUTPUT: A set of relay nodes fully cover the sensor nodes.

Step 1:

Divide the plane into squares in $l\cdot 2r_M\times l\cdot 2r_M$.

Step 2:

For each square, apply the straightforward method to find a local optimal solution.

Step 3:

Enumerate all shifts, and for each one, go over Step 1 and 2 to get the final solution with the minimum cardinality.

Complexity

It will be still similar as the complexity of applying the *Shifting Strategy* to the MGDC problem.

Approximation Ratio

Actually this case applying the *Algorithm [7]* is more similar as applying the *Algorithm [6]* to the *CASE 1-1*, except this time we only have a constant number of large sensors. Then the performance will be

$$\min_{i=1,\dots,l} \sum_{J \text{ in } S_i} |OPT_J| \le \left(1+\frac{1}{l}\right)\cdot |OPT| + O(N) \quad (15).$$

Algorithm [8]

INPUT: A set of sensor nodes $\mathcal{S} = \mathcal{S}_M \cup \mathcal{S}_N$ in the Euclidean plane $\mathbb{R}^2$;

$(\forall s_i \in \mathcal{S}_M, r_i = r_M) \wedge (\forall s_j \in \mathcal{S}_N, r_j = r_N) \wedge (r_M < r_N)$, where r_M and r_N are constants;

$|\mathcal{S}_M| = M, |\mathcal{S}_N| = N$;

N is a constant;

The *Shifting Parameter* $l_N, l_M \in \mathbb{N}$.

OUTPUT: A set of relay nodes fully cover the sensor nodes.

Step 1:

Divide the plane into squares in $l_N \cdot 2r_N \times l_N \cdot 2r_N$.

Step 2: Apply the Shifting Strategy twice.

1. **for each** $l_N \cdot 2r_N \times l_N \cdot 2r_N$ square, **do**
2. Divide the square into smaller squares $l_M \cdot 2r_M \times l_M \cdot 2r_M$.
3. **for each** shift in one large square, **do**
4. **for each** $l_M \cdot 2r_M \times l_M \cdot 2r_M$ square, **do**
5. Apply the straightforward method to find a local optimal solution.
6. **endfor**
7. **endfor**
8. Find the best solution for the large square.
9. **endfor**

Step 3:

Enumerate all shifts for partitioning the plane, and for each shift, go over Step 1 and 2 to get the final solution with the minimum cardinality.

Complexity

For each large square, the *Shifting Strategy* is applied, and all sensors in it are in $\mathcal{S}_M$. That means the running time for this step will be the same as applying the *Shifting Strategy* to the MGDC problem. We divide the entire plane into ${l_N}^2$ amount of large squares. Therefore, a coefficient ${l_N}^2$ will be added to the final running time, and it is still a polynomial.

Approximation Ratio

Let the final solution for the regular sensors be

$$\min \sum_i \left|SOL_M^i\right| \leq \min \sum_i \left(1+\frac{1}{l_M}\right) \cdot \left|OPT_M^i\right| \quad (16),^{6}$$

where SOL_M^i represents a local optimal solution for the set of regular sensors in a specific large strip, and OPT_M^i represents a corresponding optimal solution. W.r.t. $\mathcal{S}_M$, OPT_M^i actually is a local optimal solution. Hence, it holds that

$$\min \sum_i \left|OPT_M^i\right| \leq \left(1+\frac{1}{l_N}\right) \cdot |OPT_M| \quad (17),$$

where OPT_M represents an optimal solution for $\mathcal{S}_M$. Then from (16) and (17),

$$\min \sum_i \left|SOL_M^i\right| \leq \left(1+\frac{1}{l_M}\right) \cdot \left(1+\frac{1}{l_N}\right) \cdot |OPT_M| \quad (18).$$

Since the number of special sensors is a constant, then at most they will cause a constant number of errors. Then the final performance will be

$$\min \sum_j \left|SOL^j\right| \leq \left(1+\frac{1}{l_M}\right) \cdot \left(1+\frac{1}{l_N}\right) \cdot |OPT_M| + O(N) \quad (19).$$

CASE 1 Conclusion

In this case, we only considered two different kinds of sensors, i.e. the large range and the small range ones respectively. Two sub cases are, 1. n large sensors and a constant number of small sensors, 2. n small sensors and a constant number of large sensors. For each case, there are two different ways to apply the *Shifting Strategy*, i.e. partitioning the plane by the large range or the small range. Generally speaking, using the dominant range to apply the *Shifting Strategy* would be relatively better since the rest constant number of special sensors may not bring on significant amount errors. Thereby, without concerning the special sensors when apply the Shifting Strategy the approximation performance will stay in the PTAS. Even though in the worst case, the special sensor indeed would cause more errors, the amount will be bounded by a constant number. Hence, the final approximation performance will be still fairly close to the PTAS.

On the other hand, doing the opposite way, i.e. using the minor range to partitioning, may lead to worse results. Particularly, from (13), it is obvious to see that how well the algorithm work may depend on the distribution of the sensors. More specifically, at least it may depend on how wide a *maximum overlap* could expand, and, from (10), how many "wide" maximum overlaps exist. What is even worse is that there is no upper bound issued w.r.t the approximation performance.

[6] We only consider one dimension for the analysis.

Another important observation yet not shown explicitly much is that there is a conflict between the complexity and the approximation performance. The *Algorithm [6]* and the *Algorithm [8]* actually is a typical pair to demonstrate this observation. Although in the *Algorithm [8]* the running time is polynomial, and the approximation performance is also close to the PTAS, the underlying reason is that we applied the *Shifting Strategy* twice, otherwise, if we did not apply it to each large square, then for each such square, it could hardly be covered by a constant amount of relay nodes at most. Therefore, the running time of the algorithm if applying the Shifting Strategy only once in the worst case could go beyond polynomial significantly. More specifically, $4 \cdot {l_N}^2 + M$ could be the upper bound fairly covering all the sensors in one square, but it is not a constant, which means the running time could rise to $O(M^{c \cdot M})$, where c is a constant, according to the proof of the *Shifting Lemma*. As a result, by doing so to the *Algorithm [8]*, it will become the exact same as the *Algorithm [6]*. Both of them, the *Algorithm [6]* and the *Algorithm [8]*, are dealing with the same case, and they all divide the plane in $l_N \cdot 2r_N \times l_N \cdot 2r_N$. The only thing varying is the size relationship between r_M and r_N, i.e from $r_M \geq r_N$ to $r_M < r_N$. Along with its varying, the running time is changing from polynomial to beyond polynomial, while the approximation performance is changing from no upper bound (or saying sensor distribution dependent) to close to PTAS.

Let us call this conflict between the complexity and the approximation performance the complexity-and-approximability dilemma. Actually it is not a novel thing in the theoretical algorithm area but still rather tough to tackle. Then seeking a boundary or an area in this dilemma where a trade-off solution could be found such that a polynomial running time and a fair approximation performance could be established becomes interesting.

CASE 2: m sensor nodes are of the special communication range from others where m is not constant.

From the *CASE 1*, it is not hard to see that applying the Shifting Strategy directly once may not work for this case. In this case neither the number of the regular sensors nor the number of the irregular sensors is constant. If we divide the plane by using the small range then we have to make $O(n)$ amount of large sensors (assuming there are n large sensors and m small sensors) fit into the strips. The hard part is the *Shifting Lemma* will not hold anymore, and the approximation performance can depend on the distribution of the sensors as seen in the *Algorithm [6]*. On the other hand, if we divide the plane by using the large range then for each strip or square doing the local search may not be done within polynomial time as discussed in the conclusion for the *CASE 1*.

The *Algorithm [8]* is a special one. It applies the *Shifting Strategy* twice. Actually it would work better than the previous ways we discussed, but we need to make some changes. First of all, the upper bound of the approximation performance has an item, $O(N)$, which cannot be acceptable in this case. The reason this item exists is that if N is a constant, we do not have to care about it too much because these special sensors would lead to merely a constant number of errors at most, which will not hurt the approximation performance much. In principle, two applications of the *Shifting Strategy* can be explained into two different ways. The first way is that, Step 1, divide the plane by using the large range, and then seek the local optimal solution for the large sensors; Step 2, divide each large square by using the small range, and then seek the local optimal solution for the small sensors in this square, and finally combine all such local solutions together; Step 3, merge the solution for the large sensors and the solution for the small sensors together to get the final solution. The merge step can be done within polynomial time. We will discuss it in the *Algorithm [9]*. The second way it that, Step 1, divide the plane by using the large range; Step 2, divide each large square by using the small range, and then seek the local optimal solution for both the large and the small sensors in this square; Step 3, combine all local solutions together to get the final solution. Apparently, the first way is more interesting because the second way will encounter the same problem met in the *Algorithm [6]*, i.e. the approximation performance may depend on the distribution of the sensors.

Nonetheless, inspired by the *Algorithm [8]*, we could think about this twice-application of the *Shifting Strategy* in another way.

One core problem here is that the way we applied the *Shifting Strategy* is too rigid. This way is trying to "force" a set of unfitting sensors to fit into the grid. A more "natural" way is to deal with the two different types of sensors individually, then try to merge the sub-solutions together. In other words, we could change it to a Divide-and-Conquer style method.

Algorithm [9]

INPUT: A set of sensor nodes $\mathcal{S} = \mathcal{S}_M \cup \mathcal{S}_N$ in the Euclidean plane $\mathbb{R}^2$;

$(\forall s_i \in \mathcal{S}_M, r_i = r_M) \wedge (\forall s_j \in \mathcal{S}_N, r_j = r_N) \wedge (r_M < r_N)$, where r_M and r_N are constants;

$|\mathcal{S}_M| = M, |\mathcal{S}_N| = N$;

The *Shifting Parameter* $l_N, l_M \in \mathbb{N}$.

OUTPUT: A set of relay nodes fully cover the sensor nodes.

Step 1:

Apply the *Shifting Strategy* to $\mathcal{S}_M$, and for each relay node in the final solution, it can be presented as an overlap. We call the resultant set of overlaps $\mathfrak{O}_M$.

Step 2:

Do the same thing to $\mathcal{S}_N$, and get $\mathfrak{O}_N$.

Step 3:

Merge $\mathfrak{O}_M$ and $\mathfrak{O}_N$ by using the *Algorithm 4*.

1. **for each** $\mathcal{O}_i^{\mathcal{A}_i} \in \mathfrak{O}_M$, **do**
2. **for each** $c_j \in \mathcal{A}_i$, **do**
3. **for each** $\mathcal{O}_l^{\mathcal{B}_l} \in \mathfrak{O}_N$, **do**
4. **if** $Overlapping\_s(c_j, \mathcal{O}_l^{\mathcal{B}_l}) = True$, **then**
5. $\mathcal{B}_l = \mathcal{B}_l \cup \{c_j\}$
6. $\mathcal{A}_i = \mathcal{A}_i \backslash \{c_j\}$
7. Stop going through the rest overlaps in $\mathfrak{O}_N$
8. **endif**
9. **endfor**
10. **endfor**
11. **if** $\mathcal{A}_i = \emptyset$, **then**
12. $\mathfrak{O}_M = \mathfrak{O}_M \backslash \{\mathcal{O}_i^{\mathcal{A}_i}\}$
13. **endif**
14. **endfor**

Step 4: Output a set of relay nodes found from $\mathfrak{O}_M \cup \mathfrak{O}_N$.

Complexity

Both the cardinalities of $\mathfrak{O}_M$ and $\mathfrak{O}_N$ are at most $4 \cdot l_M^4$ and $4 \cdot l_N^4$ respectively, where l_M and l_N are the *Shifting Parameters*, according to the proof of the *Shifting Lemma*. $Overlapping\_s(\cdot,\cdot)$ will cost $O(2 \cdot N^2)$. Since in the worst case the step 3 will go through all sensors in $\mathcal{S}_M$, then the running time of this step will be $O(2 \cdot N^2 \cdot M)$. The rest two steps in this algorithm will follow the running time got from the Shifting Strategy.

Approximation Ratio

Let OPT_M and OPT_N be the optimal solutions in terms of overlaps for $\mathcal{S}_M$ and $\mathcal{S}_N$ respectively. Let OPT be the optimal solution in terms of overlaps for all sensors. Let SOL_M and SOL_N be the approximate solutions got from the step 1 and step 2 respectively. Then $|SOL_M| \leq \left(1 + \frac{1}{l_M}\right) \cdot |OPT_M|$, and $|SOL_N| \leq \left(1 + \frac{1}{l_N}\right) \cdot |OPT_N|$. Let SOL_F be the final solution got from the step 3. We have $|SOL_F| \leq |SOL_M| + |SOL_N| \leq \left(1 + \frac{1}{l_M}\right) \cdot |OPT_M| + \left(1 + \frac{1}{l_N}\right) \cdot |OPT_N| \leq \left[\left(1 + \frac{1}{l_M}\right) + \left(1 + \frac{1}{l_N}\right)\right] \cdot Max(|OPT_M|, |OPT_N|)$. Since both OPT_M and OPT_N are partial solutions, then $|OPT_M| \leq |OPT|$ and $|OPT_N| \leq |OPT|$. Therefore, $|SOL_F| \leq \left[\left(1 + \frac{1}{l_M}\right) + \left(1 + \frac{1}{l_N}\right)\right] \cdot |OPT|$.

However, actually this result is not tight. OPT_M means the overlaps from OPT only involving $\mathcal{S}_M$, and similarly OPT_N means the overlaps from OPT only involving $\mathcal{S}_N$. According to the *Corollary 1*, $|OPT| \leq |OPT_M| + |OPT_N|$. It implies that some overlaps in OPT_M and some other overlaps in OPT_N can be merged. According to the *FACT 1*, given a set of circles, the set of maximum overlaps is unique. Hence, if we merge OPT_M and OPT_N, then the result will be OPT. Otherwise, OPT_M or OPT_N might not be the optimal solutions. According to the *Shifting Lemma*, $SOL_M = OPT_M \cup R_M$, and $SOL_N = OPT_N \cup R_N$, where R is the set of errors. In the step3, we merge SOL_M and SOL_N together, which means $SOL_F \subseteq OPT \cup R_M \cup R_N$. Therefore, $|SOL_F| \leq \left(1 + \frac{1}{l_M} + \frac{1}{l_N}\right) \cdot |OPT|$.

CASE 2 Conclusion

The most significant achievement in this case is that basically we are pretty sure applying the Shifting Strategy directly and only once would hardly work. The *Algorithm [9]* is good start to think about the MRNC problem. The Divide-and-Conquer heuristic is working in this case. Given a set of sensors, we divide it into two distinct groups and try to find solutions for both of them, and finally merge the results together. However, it may not work well for the *CASE 3*, the general case, since the approximation performance could keep worsening along with the increasing of the amount of different communication ranges. We will see how it behave in the *CASE 3*.

CASE 3: no certain number of sensor nodes for either range

This case is a generalization of the *CASE 2*. The *Algorithm [9]* can be tried to tackle this case only with minor modifications.

Algorithm [10]

INPUT: A set of sensor nodes $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \ldots \cup \mathcal{S}_n$ in the Euclidean plane $\mathbb{R}^2$;

A set of communication ranges $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ corresponding to different sets of sensors;

It is not necessarily true that $\forall r_i, r_j (r_i = r_j)$;

$|\mathcal{S}_i| = N_i$;

The *Shifting Parameter* $l_1, l_2, \ldots, l_n \in \mathbb{N}$.

OUTPUT: A set of relay nodes fully cover the sensor nodes.

Step 1:

1. **for each** $\mathcal{S}_i$, **do**
2. Apply the Shifting Strategy on $\mathcal{S}_i$.
3. Get the corresponding set of overlaps, $\mathfrak{O}_i$.
4. **endfor**

Step 2: Sort the set of $\mathfrak{O}_i$ by r_i in the descending order. Let the result be $\mathbb{O}$.

Step 3:

1. **for each** $\mathfrak{O}_i \in \mathbb{O}$, **do**
2. Merge $\mathfrak{O}_i$ to all $\mathfrak{O}_j$, $j \leq i$.
3. **if** $\mathfrak{O}_i = \emptyset$ after merging, **then**
4. $\mathbb{O} = \mathbb{O} \backslash \{\mathfrak{O}_i\}$
5. **endif**
6. **endfor**

Step 4: Output a set of relay nodes found from $\bigcup_{\mathfrak{O}_i \in \mathbb{O}} \mathfrak{O}_i$.

Complexity

In the Step 1, for each application of the *Shifting Strategy*, the time complexity will be the same as MGDC. The Step 2 will cost $O(n \log n)$ for sorting. In the Step 3, the merging will happen at most $\frac{n \cdot (n-1)}{2}$ times, so its running time will be $O\left(n^2 \cdot \left(\max_i N_i\right)^3\right)$, which is still polynomial.

Approximation Ratio

According to the *Algorithm [9]*, the approximation performance should be

$$|SOL_F| \leq \left(1 + \frac{1}{l_1} + \frac{1}{l_2} + \cdots + \frac{1}{l_n}\right) \cdot |OPT|.$$

Since each l_i is a constant value, and it could be equal or greater than 1, then

$$|SOL_F| \leq O(n) \cdot |OPT|.$$

The performance will depend on the properties of the series $1 + \frac{1}{l_1} + \frac{1}{l_2} + \cdots + \frac{1}{l_n}$. This result is not a PTAS, and obviously when n increasing the performance will be getting worse in polynomial times.

CASE 3 Conclusion

Even though there might be some other better algorithm applying the Shifting Strategy, the *Algorithm [10]* sill tells that at least it is quite hard to achieve a PTAS by merely applying this strategy. The major places where the errors occur are still the partitions, same as what have seen a lot before. If thinking about it carefully, we may find a fact that for any specific instance, it may not be necessary to make such many partitions on some $\mathcal{S}_i$s, or even to deal with every communication range individually. If there is a way that the distribution of the sensors could be considered more carefully and reasonably, then a better result may exist.

Another way to think about how to apply the *Shifting Strategy* is to apply it on each dimension individually. Since so far all of our practices are directly using it on the Euclidean plane, no matter dividing the plane into strips or squares, we never tried to apply it on each dimension first. The reason this idea is brought up is that essentially what we are doing is to divide a big problem into some smaller problems, then by using local search we deliver an approximate solution. However, the granularity of the divide, or saying the number of local problems, is conflict with the correctness, and also that is why the complexity and the approximation performance is a dilemma as we concluded in the *CASE 1*. Therefore, if in a Euclidean plane, the size of each square can depend on the distribution of the sensors, then the approximation performance might be improved. Applying the strategy on each dimension individually could give a heuristic to make close to this purpose.

Inapproximability Analysis

Typically there are two ways to prove a given optimization problem does not have a PTAS. One is call the *Gap Technique*, the other one is to use L-reduction from a target problem to the given problem, if the target problem does not have a PTAS then the given problem does not have a PTAS.

Gap Technique [17]

Let X be an NP-hard decision problem, let Y be a minimization problem, and let τ be a polynomial time computable transformation from the instances of X into the set of instances of Y that satisfies the following two conditions for fixed integers a $< b$:

1. Every YES-instance of X is mapped into an instance of Y with optimal objective value at most a.
2. Every NO-instance of X is mapped into an instance of Y with optimal objective value at least b.

Then problem Y does not have a polynomial time ρ-approximation algorithm with worst case ratio $\rho < b/a$ unless P=NP. Especially, problem Y does not have a PTAS unless P=NP.

Theorem 3 [17]

Assume that X and Y are optimization problems, and that there is an L-reduction from X to Y. If there is a PTAS for Y, then there is a PTAS for X. Equivalently, if X does not have a PTAS, then Y does not have a PTAS.

L-reduction

Let X and Y be NPO problems. An L-reduction from problem X to problem Y is a pair of funtions (R, S) that satisfy the following properties. R is a function from the set of instances of X to the set of instances of Y. S is a function from the set of feasible solutions of the instances of Y to the set of feasible solutions of the instances of X. Both funtions are computable in polynomial time.

1. For any instance I of X with optimal cost $OPT(I)$, $R(I)$ is an instance of Y with optimal cost $OPT\big(R(I)\big)$ such that for some fixed positive constant α
$$OPT\big(R(I)\big) \leq \alpha \cdot OPT(I).$$
2. For any feasible solution s of $R(I)$, $S(s)$ is a feasible solution of I such that from some fixed positive constant β
$$\big|c\big(S(s)\big) - OPT(I)\big| \leq \beta \cdot \big|c(s) - OPT\big(R(I)\big)\big|.$$
Here $c\big(S(s)\big)$ and $c(s)$ represent the costs of the feasible solutions $S(s)$ and s respectively.

Idea

We will start with the second method to disprove the approximability of the *MRNC* problem. L-reduction actually is trying to establish a close connection between two optimization problems, where "close" means they will have the similar behaviors. Then the problem is to pick an appropriate *X* problem. The *MVDCC* problem might be a possible candidate, but it is really hard to put down a valid proof. The *MVDCC* is known as not in the APX class [17], hence, if the *MVDCC* problem could be L-reduce to the *MRNC* problem, then the latter one is not in the APX class as well.

Another try is to use the *MCP* problem. This problem is not approximable within $|V|^{\frac{1}{7-\epsilon}}$ for any $\epsilon > 0$ [27]. The *MRNC* problem actually can be considered in another perspective. Now that we have seen that seeking the set of relay nodes covering the sensors is equivalent to seek the set of maximum overlaps, this problem can be expressed as a graph partition problem. The sensors are the vertices. For each pair of sensors, if their Euclidean distance is less or equal to the sum of their communication ranges, then there will be an edge established in the graph. What the problem seeks is a partition such that in each partition it is a complete graph. The measure is the number of partitions. This problem is exactly the *MCP* problem. However, this way to express the *MRNC* problem is not so precise, because given a clique in the graph, the corresponding sensors in it may not necessarily overlap together. A typical counter example is that three circles are tangent in pairs but not overlapping together. There will be a wondering that whether or not it means the MRNC problem is harder than the MCP problem, since even for a given clique in the graph, determining if the sensors within the clique could overlap together cannot be done in polynomial time in the worst case. The clique only tells that each pair of sensors are overlapping together. From the *Algorithm 3*, we have seen that much more is needed. Even though this way still may not tell the precise upper bound or if the upper bound exists w.r.t. the approximation performance, it may tell that it is hard to have even a constant ratio, and this result is consistent with what the *Algorithm [10]* delivered.

Conclusion

In this study, three major things are discussed. The first one is that a straightforward method is given to address the *MRNC* problem. This step is useful to make us understand why the *MRNC* problem is hard, and also it can be used in our approximation algorithms. The second one is that we analyzed how the Shifting Strategy would perform in the *MRNC* problem. Three cases are discussed, and we find that it is hard to have a PTAS by doing this way. The last thing is the inapproximability analysis. Two different ways to disprove that the MRNC problem could have a PTAS are proposed. Further work is still expected.

Furture Work

Four major tasks will be in the future. The first one is to use algebraic ways to try the dynamic programming algorithm mentioned in the *Algorithm 3*. It may not help to improve the approximation performance but still be interesting to think about the algorithm in another perspective. The second one is to keep studying the complexity-and-approximation dilemma. It will be a very interesting entry point to explore how the structure of the NP class is, and if there is a way to adjust the complexity and the approximation performance more flexibly and smoothly. The third one is to keep trying different ways to apply the *Shifting Strategy* to the MRNC problem, since somehow this method has given the *MGDC* problem a PTAS result, and that is significant. The last one is to work on the inapproximability of the *MRNC* problem, since even though we have not given up to find a PTAS or a fairly close result on this problem, many clues have implied that there is a great chance it does not have any PTAS or even constant ratios.

References

[1] Lloyd, Errol L., and Guoliang Xue. "Relay node placement in wireless sensor networks." Computers, IEEE Transactions on 56.1 (2007): 134-138.

[2] Cormen, Thomas H., et al. Introduction to algorithms. Vol. 2. Cambridge: MIT press, 2001.

[3] Hochbaum, Dorit S., and Wolfgang Maass. "Approximation schemes for covering and packing problems in image processing and VLSI." Journal of the ACM (JACM) 32.1 (1985): 130-136.

[4] Schrijver, Alexander. Combinatorial optimization: polyhedra and efficiency. Vol. 24. Springer, 2003.

[5] Masuyama, Shigeru, Toshihide Ibaraki, and Toshiharu Hasegawa. "The Computational Complexity of the< I> m</I>-Center Problems on the Plane." IEICE TRANSACTIONS (1976-1990) 64.2 (1981): 57-64.

[6] Fowler, Robert J., Michael S. Paterson, and Steven L. Tanimoto. "Optimal packing and covering in the plane are NP-complete." Information processing letters 12.3 (1981): 133-137.

[7] Hochbaum, Dorit S., and Wolfgang Maass. "Approximation schemes for covering and packing problems in image processing and VLSI." Journal of the ACM (JACM) 32.1 (1985): 130-136.

[8] Gonzalez, Teofilo F. "Covering a set of points in multidimensional space." Information processing letters 40.4 (1991): 181-188.

[9] Brönnimann, Hervé, and Michael T. Goodrich. "Almost optimal set covers in finite VC-dimension." Discrete & Computational Geometry 14.1 (1995): 463-479.

[10] Franceschetti, Massimo, Matthew Cook, and Jehoshua Bruck. "A geometric theorem for approximate disk covering algorithms." (2001).

[11] Călinescu, Gruia, et al. "Selecting forwarding neighbors in wireless ad hoc networks." Mobile Networks and Applications 9.2 (2004): 101-111.

[12] Narayanappa, Sada, and Petr Vojtechovský. "An Improved Approximation Factor For The Unit Disk Covering Problem." CCCG. 2006.

[13] Carmi, Paz, Matthew J. Katz, and Nissan Lev-Tov. "Covering points by unit disks of fixed location." Algorithms and Computation. Springer Berlin Heidelberg, 2007. 644-655.

[14] Claude, Francisco, et al. "Practical discrete unit disk cover using an exact line-separable algorithm." Algorithms and Computation. Springer Berlin Heidelberg, 2009. 45-54.

[15] Das, Gautam K., et al. "On the discrete unit disk cover problem." WALCOM: Algorithms and Computation. Springer Berlin Heidelberg, 2011. 146-157.

[16] Liao, Chen, and Shiyan Hu. "Polynomial time approximation schemes for minimum disk cover problems." Journal of combinatorial optimization 20.4 (2010): 399-412.

[17] Schuurman, Petra, and Gerhard J. Woeginger. "Approximation schemes-a tutorial." Lectures on scheduling. 2000.

[18] Kann, Viggo. On the approximability of NP-complete optimization problems. Diss. Royal Institute of Technology, 1992.

[19] Arora, Sanjeev, et al. "Proof verification and the hardness of approximation problems." Journal of the ACM (JACM) 45.3 (1998): 501-555.

[20] Mayr, Ernst W., Hans Jürgen Prömel, and Angelika Steger, eds. Lectures on proof verification and approximation algorithms. Vol. 1367. Springer Science & Business Media, 1998.

[21] Ausiello, Giorgio, Pierluigi Crescenzi, and Marco Protasi. "Approximate solution of NP optimization problems." Theoretical Computer Science 150.1 (1995): 1-55.

[22] Papadimitriou, Christos, and Mihalis Yannakakis. "Optimization, approximation, and complexity classes." Proceedings of the twentieth annual ACM symposium on Theory of computing. ACM, 1988.

[23] Papadimitriou, Christos H. Computational complexity. John Wiley and Sons Ltd., 2003.

[24] Dinur, Irit. "The PCP theorem by gap amplification." Journal of the ACM (JACM) 54.3 (2007): 12.

[25] Bartal, Yair, Moses Charikar, and Danny Raz. "Approximating min-sum k-clustering in metric spaces." Proceedings of the thirty-third annual ACM symposium on Theory of computing. ACM, 2001.

[26] Sahni, Sartaj, and Teofilo Gonzalez. "P-complete approximation problems." Journal of the ACM (JACM) 23.3 (1976): 555-565.

[27] Crescenzi, Pierluigi, and Viggo Kann. "A compendium of NP optimization problems." URL: http://www. nada. kth. se/~ viggo/problemlist/compendium. html (1997).

[28] Ganz, Aura, Zygmunt J. Haas, and C. M. Krishna. "Multi-tier wireless networks for PCS." Vehicular Technology Conference, 1996. Mobile Technology for the Human Race., IEEE 46th. Vol. 1. IEEE, 1996.

[29] Nemeroff, Jay, et al. "Application of sensor network communications." Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE. Vol. 1. IEEE, 2001.

[30] J. Pan, Y.T. Hou, L. Cai, Y. Shi, S.X. Shen; Topology control for wireless sensor networks; ACM MobiCom'03, pp. 286–299.

[31] Hao Bin, Jian Tang, and Guoliang Xue. "Fault-tolerant relay node placement in wireless sensor networks: formulation and approximation." High Performance Switching and Routing, 2004. HPSR. 2004 Workshop on. IEEE, 2004.

[32] Liu, Hai, Peng-Jun Wan, and Xiaohua Jia. "Fault-tolerant relay node placement in wireless sensor networks." Computing and combinatorics. Springer Berlin Heidelberg, 2005. 230-239.

[33] Tang, Jian, Bin Hao, and Arunabha Sen. "Relay node placement in large scale wireless sensor networks." Computer communications 29.4 (2006): 490-501.

[34] Lloyd, Errol L., and Guoliang Xue. "Relay node placement in wireless sensor networks." Computers, IEEE Transactions on 56.1 (2007): 134-138.

[35] Shams, SM Saif, et al. "A fast approximation algorithm for relay node placement in double-tiered wireless sensor network." Military Communications Conference, 2008. MILCOM 2008. IEEE. IEEE, 2008.

[36] Wang, Gang, et al. "Reliable relay node placement in wireless sensor network." Communications and Networking in China, 2008. ChinaCom 2008. Third International Conference on. IEEE, 2008.

[37] Chen, Guangting, and Suhui Cui. "Relay node placement in two-tiered wireless sensor networks with base stations." Journal of Combinatorial Optimization 26.3 (2013): 499-508.

[38] Masuyama, Shigeru, Toshihide Ibaraki, and Toshiharu Hasegawa. "The Computational Complexity of the< I> m</I>-Center Problems on the Plane." IEICE TRANSACTIONS (1976-1990) 64.2 (1981): 57-64.

[39] Fowler, Robert J., Michael S. Paterson, and Steven L. Tanimoto. "Optimal packing and covering in the plane are NP-complete." Information processing letters 12.3 (1981): 133-137.

[40] Hochbaum, Dorit S., and Wolfgang Maass. "Approximation schemes for covering and packing problems in image processing and VLSI." Journal of the ACM (JACM) 32.1 (1985): 130-136.

[41] Gonzalez, Teofilo F. "Covering a set of points in multidimensional space." Information processing letters 40.4 (1991): 181-188.

[42] Brönnimann, Hervé, and Michael T. Goodrich. "Almost optimal set covers in finite VC-dimension." Discrete & Computational Geometry 14.1 (1995): 463-479.

[43] Franceschetti, Massimo, Matthew Cook, and Jehoshua Bruck. "A geometric theorem for approximate disk covering algorithms." (2001).

[44] Călinescu, Gruia, et al. "Selecting forwarding neighbors in wireless ad hoc networks." Mobile Networks and Applications 9.2 (2004): 101-111.

[45] Narayanappa, Sada, and Petr Vojtechovský. "An Improved Approximation Factor For The Unit Disk Covering Problem." CCCG. 2006.

[46] Carmi, Paz, Matthew J. Katz, and Nissan Lev-Tov. "Covering points by unit disks of fixed location." Algorithms and Computation. Springer Berlin Heidelberg, 2007. 644-655.

[47] Claude, Francisco, et al. "Practical discrete unit disk cover using an exact line-separable algorithm." Algorithms and Computation. Springer Berlin Heidelberg, 2009. 45-54.

[48] Das, Gautam K., et al. "On the discrete unit disk cover problem." WALCOM: Algorithms and Computation. Springer Berlin Heidelberg, 2011. 146-157.

[49] Liao, Chen, and Shiyan Hu. "Polynomial time approximation schemes for minimum disk cover problems." Journal of combinatorial optimization 20.4 (2010): 399-412.

Appendix

(See the other file.)