White Space Regions

Shayan Ehsani, MohammadAmin Fazli, Mohammad Ghodsi,
MohammadAli Safari*, Morteza Saghafian, and Mohammad Tavakkoli

Dept. of Computer Engineering, Sharif University of Technology
{fazli,ehsani,mtavakoli,saghafian}@ce.sharif.edu,
{ghodsi,msafari}@sharif.edu

Abstract. We study a classical problem in communication and wireless networks called *Finding White Space Regions*. In this problem, we are given a set of antennas (points) some of which are noisy (black) and the rest are working fine (white). The goal is to find a set of convex hulls with maximum total area that cover all white points and exclude all black points. In other words, these convex hulls make it safe for white antennas to communicate with each other without any interference with black antennas. We study the problem on three different settings (based on overlapping between different convex hulls) and find hardness results and good approximation algorithms.

1 Introduction

The problem of finding white space regions arises in the context of *Cognitive Radio* systems. According to FCC Task Force report [1] in 2004 on the under-utilization of the wireless radio environment, primary users did not use their licensed spectrum from 15% to 85% of the time. Therefore, these silence ranges in time and frequency can be exploited by unlicensed users provided that these users do not make any interference with the primary licensed users.

Consider a primary network whose Base Stations(BSs) are fixed and communicate occasionally in a licensed band along with spatially random distributed spectrum sensing base stations of the cognitive radio networks distributed in a large region. Such a network is depicted in Fig. 1 in which primary users are shown by long antennas and the others by short antennas. The goal is to find the largest area in which no primary user transmission is detected. From now on, we call this area the *white space region* or WSR for short. Cognitive radio users can then safely use this WSR to communicate with each other without affecting primary users. Moreover, the more the area of WSR is the more non-licensed users can communicate with each other.

Abachi et al [3] first proposed a model based on the spatial information of the antennas. In their model, each antenna is a point with cartesian coordinates. They colored points corresponding to primary users as black and the rest as

* The research was in part supported by grant no. CS1389-4-09 from the institute for Research in Fundamental Sciences (IPM).

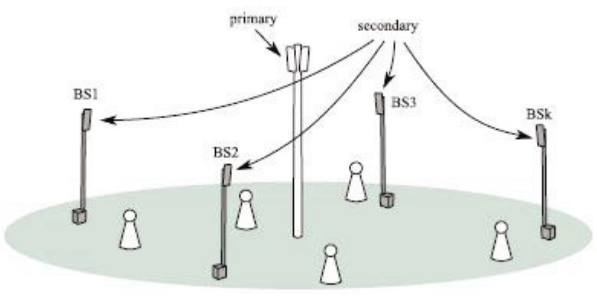

Fig. 1. Sensing Scenario [2]

white. In this model, WSRs are convex regions around white points whose circumcircle does not contain any black point. They proposed a pseudo polynomial algorithm for finding good WSRs.

In another related work, Fischer [4,5] focuses on finding just one WSR with the maximum area. In [4], he proposes a dynamic programming based $\mathcal{O}\left(n^4 log(n)\right)$-time algorithm for finding the maximum area convex polygon which does not contain any black points. In [5] he improved his result by learning techniques.

In this paper, we let more than one convex hull be chosen. Our aim is to find a set of WSRs which covers all white points and whose total area is maximum (We assume a single point or two points connected via a segment as a convex hull of area 0; so, it is always possible to cover all white points with convex hulls). Moreover, no convex hull should contain any black point inside.

1.1 Formal Problem Definition

As mentioned earlier we have a set of n points; some are black (n_b of them) and others are white (n_w of them), and we are looking for a set of convex hulls of the white points that:

- cover all the white points (a single point or a line segment between two points is assumed a convex hull).
- are free from black points.
- the area of their union is maximum.

One question is whether these convex hulls are allowed to have intersection. Depending on wether the convex hulls can overlap, we have three different problems(see Fig. 2):

Definition 1. *Three different problems of finding the white space regions are:*

- **Totally Disjoint Convex Covering (TDCC)***: In this problem WSRs cannot touch each other, i.e. they are not allowed to have common vertices or common area or one's edge lying on another's edge(Fig. 2 part (a))*

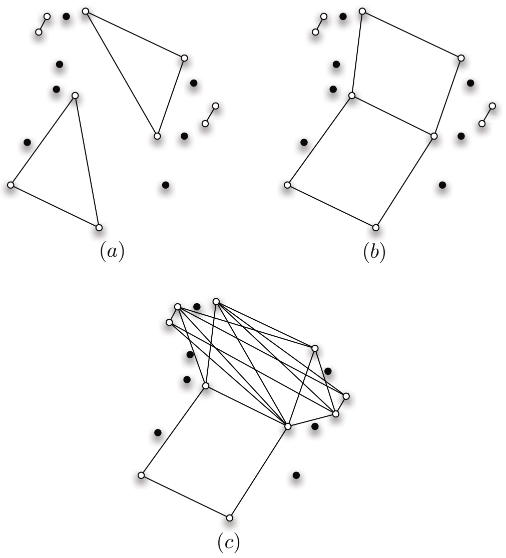

Fig. 2. Three different types of the problem

- **Nonoverlapping Convex Covering (NOCC)**: *In this problem WSRs are not allowed to have common areas but having common vertices is fine (Fig. 2 part (b)).*
- **Convex Covering with no Restriction (CCR)**: *There are no restrictions in regards to intersection (Fig. 2 part (c))*

Our main results are as follows. First, we obtain some hardness results.

Theorem 1. *Both NOCC and TDCC problems are NP-hard.*

Next we obtain two different, but similar approximation algorithms for both of the problems (OPT is the area of the optimal solution).

Theorem 2. *There is an approximation algorithm that computes a set of convex hulls with total area of at least* $(\frac{OPT}{2\log(2n/OPT)+2\log(n)})^{1/4}$ *for the* NOCC *problem.*

Theorem 3. *There is an approximation algorithm that computes a set of convex hulls with total area of at least* $\frac{3\sqrt{3}}{4.\pi}(\frac{OPT}{2\log(2n/OPT)+2\log(n)})^{1/4}$ *for the* TDCC *problem.*

Finally, we propose heuristic algorithms based on Fischer's algorithm and provide experimental and theoretical analysis for them.

For the CCR problem, it's straightforward to find an exact polynomial time algorithm.

Theorem 4. *CCR can be solved in polynomial time.*

Proof. It is enough to consider all possible triangles of white vertices that contain no black vertex and output their union. It is easy to see that this output is the maximum area convex covering. Since the number of possible triangles is polynomial ($\leq \binom{n}{3}$) the running time will be polynomial.

The structure of our paper is as follows. In the next section, we prove the hardness results. Then, in section 3, we propose good approximation algorithms for the problems. In section 4, we propose greedy algorithms based on Fischer's works [4,5]. We show by experimental evaluations that these greedy algorithms work well in practice, but may behave arbitrarily bad in the worst case.

2 Hardness Results

In this section we will prove the NP-Hardness of NOCC. The hardness of TDCC follows similarly. The idea is to reduce the independent set problem on triangle-free planar graphs. A proof for the hardness of the latter problem can be found at [6].

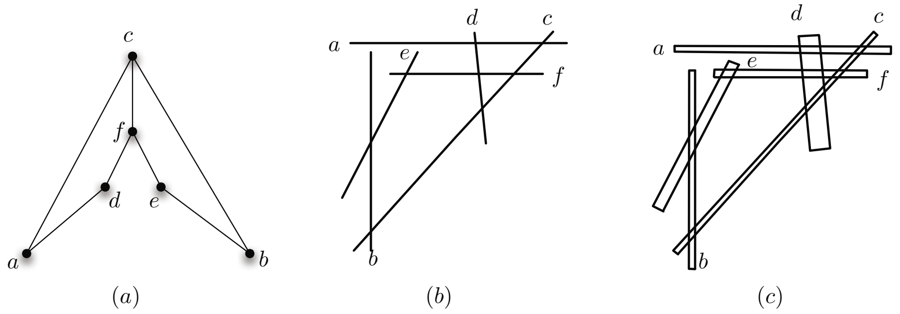

Fig. 3. Forming an intersection graph of rectangles with equal area from G

Definition 2. Intersection Graph *of a set of geometric objects S, is defined as a graph $G = (S, E)$ in which there is an edge between two nodes $s, s' \in S$ if $s \cap s' \neq \emptyset$.*

We use the fact that every planar graph is an intersection of line segments in 2D space, e.g. we can put line segments corresponding to vertices of every planar graph such that two vertices are connected if and only if their corresponding line segments have an intersection. Construction of this equivalent graph can be done in polynomial time[7].

Let G be a planar graph whose maximum independent set is I and $|I| = k$. We make an instance of NOCC whose optimal solution has area ϵk for some constant ϵ. That concludes the reduction as having a polynomial algorithm for NOCC would reveal the size of I.

To do the reduction, we first transform G into an intersection of line segments using the polynomial construction in [7]. It's easy to make the construction such that no two line segments are co-linear and intersections does not happen on the end-points.

Assume s_i is the corresponding line segment to vertex i. Next, we replace s_i with a rectangle r_i whose area is ϵ (i.e. one edge length is $|s_i|$ and the other edge length is $\frac{\epsilon}{|s_i|}$, where $|s_i|$ is the length of s_i). ϵ is chosen so small that every two lines s_i and s_j intersect if and only if their corresponding rectanbles r_i and $_j$ intersect.

Finally, for every two pairs $i \neq j$, we put a black point on the line between every vertex of r_i and every vertex of r_j. These black points must not be placed inside any of the rectangles. This latter step ensures that only r_i's will be selected when solving the NOCC problem. The value of ϵ is chosen so small that it does not interfere with the construction, i.e. it's smaller than the distance of every two points in line segment intersection graph. The process of constructing the intersection graph of the rectangles is shown in Fig. 3.

How big is the area of the optimal solution in this NOCC instance? It's easy to see that the optimal solution must be a set of intersecting-free r_i's which means the optimal answer is $|I|\epsilon$. This completes the proof of the Theorem 1.

3 Approximation Algorithms

In this section we propose an approximation algorithm for NOCC. One can assume that the optimal solution is a set of non-intersecting triangles because every polygon in the optimal solution can be triangulated. So the problem reduces to the problem of finding the maximum weighted independent set in the intersection graph of all the black point-free triangles with white vertices whose weights are equal to their area. That is, for every triangle of three white points that does not contain any black point, we put a vertex with weight equal to the area of the trinagle. Two vertices are adjacent if their corresponding triangles interset in area.

Agarwal et al. [8] consider a non-weighted version of this problem for the intersection graphs of n convex shapes in 2D space. They proposed an approximation algorithm that computes a solution whose output is at least $\sqrt[3]{\frac{\alpha}{2\log(2n/\alpha)}}$, where α is the cardinality of the optimum solution. We have manipulated their algorithm to solve our problem. Let's first outline the algorithm of Agarwal et al. and then state our manipulations.

3.1 Non-weighted Version

The algorithm in [8] gets n convex shapes in 2D space as input and computes the maximum independent set of their intersection graph. Their algorithm has a divide and conquer approach and is outlined below(The algorithm is stated for a set of triangles Δ instead of polygons).

Algorithm 1. ApproxNOCC($\varDelta$: a set of triangles)

```
Find a line $\ell$ which divides $\varDelta$ into two almost equal cardinality sets.
$C_\ell$ = all the triangles that intersect $\ell$.
$L_\ell$ = all the triangles whose position is completely to the left of $\ell$.
$R_\ell$ = all the triangles that are completely positioned to the right of $\ell$.
Compute $\zeta(C_\ell)$ by a separate approximation algorithm
return $max\{ApproxNOCC(L_\ell) + ApproxNOCC(R_\ell), \zeta(C_\ell)\}$
```

Given a vertical line ℓ every triangle is either completely to the left of ℓ, completely to the right of ℓ, or intersects ℓ. Based on this we have three sets of triangles L_ℓ, R_ℓ and C_ℓ. By the way, ℓ is chosen in such a way that $||R_\ell| - |L_\ell||$ is minimized. For C_ℓ they provide a separate algorithm that computes a solution whose size is $\zeta(C_\ell)$, for some function ζ. Notice that this problem is different from and simpler than the original problem as all triangles in C_ℓ intersect the vertical line ℓ. At the end their algorithm returns the maximum of $\zeta(C_\ell)$ and their approximation algorithm solution on the reduced set $L_\ell \cup R_\ell$.

Let $\mu(\varDelta)$ be the size of the solution returned by their algorithm. It's easy to prove that

$$\mu(\varDelta) = \max\{\mu(L_\ell) + \mu(R_\ell), \zeta(C_\ell)\} \tag{1}$$

Given the above recursive equation, Agarwal et al. prove the following lower bound for $\mu(\varDelta)$.

$$\mu(\varDelta) \geq \zeta(\frac{OPT}{2log(2t/OPT)}) \tag{2}$$

where $t = |\varDelta|$ and OPT is the size of the optimal solution. For C_ℓ they propose a dynamic programming based algorithm that computes a solution that has a cubic approximation factor:

$$\zeta(C_\ell) \geq \sqrt[3]{OPT}$$

where OPT is the size of the optimal solution on C_ℓ. This yields the above $\sqrt[3]{\frac{\alpha}{2log(2n/\alpha)}}$ approximation factor.

3.2 Our Modification

Our algorithm has the same structure except that our triangles are weighted and consequently, our approximation algorithm for the maximum independent set on C_ℓ is different as explained below.

First we need the following useful lemma for our construction.

Lemma 1. *Suppose that we are given a set I of weighted objects such that for each $x \in I$ we have $w(x) \geq 1$. Assume a partial order $\preceq$ over the members of I is given. Let $S = \sum_{x \in I} w(x)$. Then there exists a chain or anti-chain over the members of I whose total weight is at least $\sqrt{S}$.*

Proof. Let $\mathcal{A}$ be a maximum cardinality anti-chain and $\mathcal{C}$ be a partition of objects into minimum number of chains. By Dilworth's theorem [9], the cardinality of $\mathcal{A}$ equals the number of chains in $\mathcal{C}$.

If $\mathcal{A}$ has cardinality more than $\sqrt{S}$ then we are done as every object has weight more than one. Otherwise, $\mathcal{C}$ has at most $\sqrt{S}$ chains which means the heaviest chain must have total weight at least $\frac{S}{\sqrt{S}} = \sqrt{S}$.

Let ℓ be a vertical line. Like before, let C_ℓ be all the triangles that intersect ℓ. Suppose $OPT(C_\ell)$ is the total area of all the triangles in the maximum weighted independent set of C_ℓ. For each $s_i \in C_\ell$ define $r(s_i)$(resp. $l(s_i)$) to be the x-coordinate of the rightmost (resp. leftmost) point in s_i. Also define $c(s_i)$ to be the maximum y-coordinate of the intersection of s_i with line ℓ. The following lemma is obtained from [8] by minor modifications.

Lemma 2. *There exists a sequence $I = \prec s_{i_1}, s_{i_2}, ..., s_{i_m} \succ$ of the members of C_ℓ such that $\sum_{j=1}^{m} w(s_{i_j}) \geq \sqrt[4]{|\sum_{x \in C_\ell} w(x)|}$, where $w(x)$ is the area of the triangle x and I is a chain according to one of the following order conditions:*

- *$O_1(s_i, s_j)$: $r(s_i) < r(s_j)$ and $c(s_i) < c(s_j)$.*
- *$O_2(s_i, s_j)$: $r(s_i) < r(s_j)$ and $c(s_i) > c(s_j)$.*
- *$O_3(s_i, s_j)$: $r(s_i) > r(s_j)$.*

Proof. Let $S_l = |\sum_{x \in C_\ell} w(x)|$. First we apply lemma 1 on C_ℓ with partially order defined by O_1. So, there must be a chain or anti-chain D_1 with total weight at least $\sqrt{S_l}$. If it's a chain then we are done. Otherwise, apply lemma 1 to D_1 on partially order defined by O_2. According to this lemma, there exist a chain or anti-chain D_2 whose total weight is at least $\sqrt{\sqrt{S_l}} = \sqrt[4]{S_l}$. If it's a chain then it satisfies condition O_2. Otherwise, as it is an anti-chain by both O_1 and O_2, it must satisfy O_3.

The interesting fact is that the optimal answer that satisfies either of O_i's can be computed in polynomial time by a dynamic-programming algorithm.

For each $x \in \{1, 2, 3\}$, we shall compute the sequence with maximum total area that satisfies O_x. Let $s_1, \cdots, s_p$ be a topological ordering based on the order O_x on C_ℓ. Let i, j be two vertices such that $i < j$ and $s_i \cap s_j = \emptyset$. Define $\phi^x(i, j)$ to be the size of the maximum total area sequence from the set $\{s_i, s_{i+1}, \cdots, s_j\}$ which satisfies O_x. $\phi^x(i, j)$ equals

$$max_{\substack{i \leq k \leq j \\ s_k \cap s_i = \emptyset \\ s_k \cap s_j = \emptyset}} \ \phi^x(i, k-1) + \phi^x(k+1, j) + w(s_k)$$

which means one can use dynamic programming and compute ϕ^x in polynomial time. At the end, $\zeta(C_\ell))$ will be the maximum of three values,

$$\zeta(C_\ell) = max\{\phi^1(1,p), \phi^2(1,p), \phi^3(1,p)\} \geq \sqrt[4]{\sum_{x \in C_\ell} w(x)} \geq \sqrt[4]{OPT(C_\ell)}$$

Given the above lower bound for $\zeta(C_\ell)$ we plug it into Equation 2 and obtain a lower bound for the approximation factor of our algorithm.

$$\mu(\Delta) \geq \sqrt[4]{\frac{OPT}{2log(2t/OPT)}} \geq \sqrt[4]{\frac{OPT}{2log(2n/OPT) + 2log(n)}}$$

The last inequality follows from the fact that $t \leq \binom{n}{3}$.

The algorithm is outlined in Algorithm 1. The input of this algorithm is the set of all triangles of white vertices which contain no black points (Δ). This completes the proof of Theorem 2.

3.3 TDCC

The idea for approximating TDCC is similar to that of NOCC with some modifications. In NOCC we could safely assume that the optimal answer is a set of triangles and then found an approximate weighted independent set on the set of feasible triangles. This assumption, however, does not hold for the case of TDCC as a polygon cannot necessarily be partitioned into a set of totally disjoint triangles.

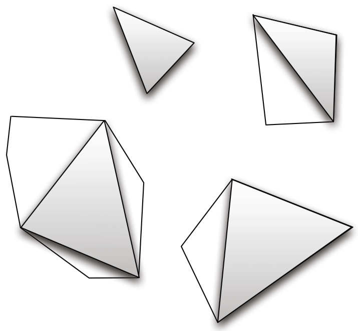

Fig. 4. Inscribed triangles with area of $\frac{3\sqrt{3}}{4\pi}$. OPT in TDCC problem.

To overcome this difficulty, we use the following useful lemma.

Lemma 3. *([10]) Let B be a compact convex body in the plane and B_k be the largest area k-edge polygon inscribed in B. Then $area(B_k) \geq area(B).\frac{k}{2\pi}sin\frac{2\pi}{k}$, where equality holds if and only if B is an ellipse.*

This means if we only focus on triangles for TDCC then the optimal solution would be less by no more than a factor of $\frac{3}{2\pi}sin\frac{2\pi}{3} = \frac{3\sqrt{3}}{4\pi}$.

Therefore, we can run the same algorithm as the one for NOCC (with the change that two triangles are considered connected even if they share an edge or a vertex) and obtain a solution whose total area is at least

$$\frac{3\sqrt{3}}{4\pi}\sqrt[4]{\frac{OPT}{2log(2n/OPT) + 2log(n)}}$$

This completes the proof of Theorem 3.

Algorithm 2. GreedyTDCC

```
$\mathcal{C} \leftarrow \emptyset$
$C$ = maximum convex hull of white points by the Fischer's algorithm[4]
while The area of $C$ is greater than 0 do
    $\mathcal{C} \leftarrow \mathcal{C} \cup C$
    Remove all the white points inside $C$
    Replace $C$'s vertices with black points
    foreach pair $(A, B)$ of points which coincide on the outside or boundary of $C$ do
        if segment $\overline{AB}$ has intersection with $C$ then
            Add a dummy black point on $\overline{AB}$ which resides inside $C$.
        end
    end
    Add a dummy black point inside $C$.
    $C$ = maximum convex hull of white points.
end
return $\mathcal{C}$
```

3.4 A Note on Measurement Units and the Root Function

It's clear that if we change our measurement unit(e.g. use meter instead of inch) then it affects our approximation factor which is a root function. To avoid this problem the area of triangles (and hence, the wight of vertices) are computed relative to the smallest triangle. Thus, all weights are at least one (which is needed for Lemma 1) and our approximation factor would not change by scaling.

4 Heuristic Approach

The algorithm of Fischer[4] for finding one single region with maximum area can be easily transformed into a greedy algorithm for our problem by repeatedly finding a region and deleting it.

In this section, we show that while this algorithm behaves well for random data, it can be arbitrarily bad in the worst case. First we explain the algorithm in detail and then provide experimental results. Finally we prove a lower bound for its behavior in the worst case. We explain the algorithm and experimental results for TDCC. The results for NOCC are essentially similar but they are more complex and need more details and is left to the journal version of our work.

4.1 Algorithm Description

The algorithm works as follows. It iteratively computes a maximum area convex hull of white regions using Fischer's algorithm. In order to prevent intersection, it puts enough black points around and inside the selected convex hulls to prevent future convex hulls to overlap existing ones.

The details of the algorithms is depicted in Algorithm 2.

Once a convex hull C is chosen, for any line segment AB of two white points A and B that intersect with C, the algorithm puts an arbitrary black point on the intersection of AB and the interior of C. That prevents line segment AB to be chosen as part of any future convex hull. It also replaces C's vertices with black points so as to prevent future convex hulls to have intersection with C's vertices.

4.2 Analysis

As we see shall see shortly this greedy algorithm could work arbitrarily bad, however, our experiments show its good behavior for random data.

First we build an example on which the solution of GreedyTDCC and the optimal differ by a factor of $\Omega(n)$. Our worst case example is depicted in Fig. 5.

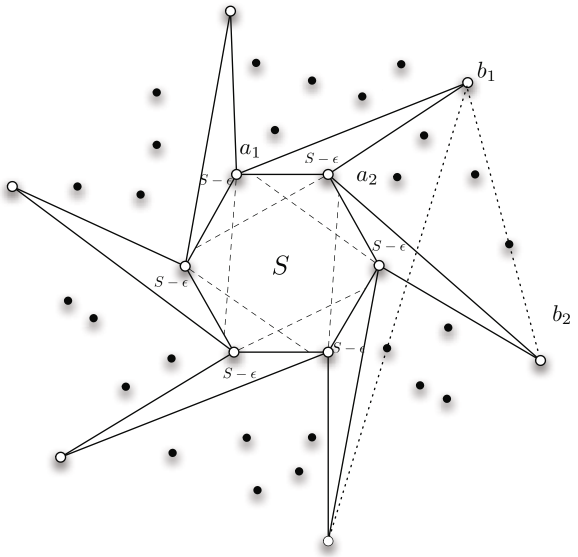

Fig. 5. Worst case example of GreedyTDCC

Let C be an n-regular polygon with area S and vertices $a_1, a_2, \cdots, a_n$. For each $1 \leq i < n$ we create a vertex b_i outside C such that the triangle $b_i a_i a_{i+1}$ has area $S - \epsilon$ (for an arbitrarily small ϵ). Assume $a_1 = a_{n+1}$ and $n \overset{2}{\equiv} 0$ for convenience.

Our aim is to let only C and triangles of the form $b_i a_i a_{i+1}$ be chosen in any solution. Therefore we only allow the following three types of line segments: $a_i a_{i+1}, b_i a_i$, and $b_i a_{i+1}$. For any other line segment we put a black point on it in a suitable place (i.e. is not contained in any of the triangle $b_i a_i a_{i+1}$) to prevent it from being chosen.

In this instance the GreedyTDCC algorithm will pick C as the first convex hull and then it can not find any other convex hull with area more than 0. So the area of the output of GreedyTDCC is S. However, the optimal answer will pick all the $\frac{n}{2}$ triangles of type $b_{2k}a_{2k}a_{2k+1}$, so $OPT = \frac{n}{2}.(S - \epsilon)$. This completes our claim that GreedyTDCC works arbitrarily bad in the worst case.

Theorem 5. *There is an instance L of n points such that*

$$\frac{OPT(L)}{GreedyTDCC(L)} \in \Omega\,(n)$$

where $GreedyTDCC(L)$ is the output of our greedy algorithm on the instance L and $OPT(L)$ is the optimal answer.

The algorithm, however, works well on random data. For this, we implemented the GreedyTDCC algorithm and tested it on seven different random distribution of the points. For each case, e.g. RAN1, we created 10 different random instances and computed the average and the worst (least) ratio of our algorithm to the optimal algorithm. The area of the optimal covering is obtained via a simple exhaustive search algorithm. The results are shown in table 1). The experimental result suggests that the output of GreedyTDCC algorithm is a good approximation of the optimal answer in practice.

Table 1. The ratio of GreedyTDCC's solution to optimal

Test Data	**#White Points**	**#Black Points**	**Worst case Ratio**	**Average Ratio**
RAN_1	10	3	0.64	0.7
RAN_2	12	12	0.49	0.66
RAN_3	16	10	0.48	0.59
RAN_4	14	20	0.41	0.72
RAN_5	15	16	0.6	0.7
RAN_6	20	40	0.56	0.68
RAN_7	20	20	0.48	0.63

5 Conclusion and Further Works

We considered three different versions of the problem of covering a set of white points without touching black points . We solved one and found hardness result as well as approximation algorithms for the other two.

The approximation factor of our algorithms are dependent to the result of Lemma 1. Any improvement in that lemma would improve the approximation factor. One may hope to find a bound like $\frac{S}{\sqrt{n}}$ (which is identical to Dilworth's bound for uniform weights), but we haven't been able to achieve it.

There are many variants of the problem that are worth studying. One particular problem that we are interested in is the weighted version of our problem. Black points have weights between zero and one and the aim is to find a set of convex hulls around white points so that the total weight of black points inside convex hulls is less than a threshold.

Acknowledgment
The authors are thankful to Abbas Mehrabian for his many useful comments throughout the preparation of this paper.

References

1. Federal Communications Commission, Notice of proposed rule making: Unlicensed operation in the TV broadcast bands. ET Docket No. 04-186 (FCC 04-113) (May 2004)
2. Cardoso, L.S., Debbah, M., Bianchi, P., Najim, J.: Cooperative spectrum sensing using random matrix theory. In: IEEE ISWPC, pp. 334–338 (May 2008)
3. Abachi, T., Hemmatyar, M.A., Fazli, M.A., Izadi, M.: Centralized Spectrum Sensing Using New Algorithmic Techniques. In: CRNet 2010, Bijing, China, August 25-27 (2010)
4. Fischer, P.: Finding Maximum Convex Polygons. In: Proceedings of the 9th International Symposium on Fundamentals of Computation Theory, pp. 234–243 (1993)
5. Fischer, P.: More or less efficient agnostic learning of convex polygons. In: Proceedings of the Eighth Annual Conference on Computational Learning Theory, pp. 337–344 (1995)
6. Veni Madhavan, C.E.: Approximation algorithm for maximum independent set in planar traingle-free graphs. In: Joseph, M., Shyamasundar, R.K. (eds.) FSTTCS 1984. LNCS, vol. 181, pp. 381–392. Springer, Heidelberg (1984)
7. de Castro, N., Cobos, F.J., Dana, J.C., Marquez, A., Noy, M.: Triangle-free planar graphs as segment intersection graphs. J. Graph Algorithms Appl. 6(1), 7–26 (2002) (electronic); Graph drawing and representations (Prague, 1999)
8. Agarwal, P.K., Mustafa, N.H.: Independent Set of Intersection Graphs of Convex Objects in 2D. In: Computational Geometry: Theory and Applications, pp. 83–95. Elsevier Science Publishers, Amsterdam (2006)
9. Dilworth, P.: A decomposition thorem for partially ordered sets. Annals of Mathematics 51, 161–166 (1950)
10. Sás, E.: On a certain extremum-property of the ellipse (in Hungarian, English summary). Mat. Fiz. Lapok 48, 533–542 (1941)