On Visibility Problems in the Plane – Solving Minimum Vertex Guard Problems by Successive Approximations [⋆]

Ana Paula Tomás[1], António Leslie Bajuelos[2] and Fábio Marques[3]

[1] DCC-FC & LIACC, University of Porto, Portugal
apt@ncc.up.pt
[2] Dept. of Mathematics & CEOC - Center for Research in Optimization and Control, University of Aveiro, Portugal
leslie@mat.ua.pt
[3] School of Technology and Management, University of Aveiro, Portugal
fabio@estga.ua.pt

Abstract. We address the problem of stationing guards in vertices of a simple polygon in such a way that the whole polygon is guarded and the number of guards is minimum. It is known that this is an NP-hard Art Gallery Problem with relevant practical applications. In this paper we present an approximation method that solves the problem by successive approximations, which we introduced in [21]. We report on some results of its experimental evaluation and describe two algorithms for characterizing visibility from a point, that we designed for its implementation.

1 Introduction

The classical *Art Gallery problem* for a polygon P is to find a minimum set of points G in P such that every point of P is visible from some point of G. We address MINIMUM VERTEX GUARD in which the set of *guards* G is a subset of the vertices of P and each guard has 2π range unlimited visibility. This is an NP-hard combinatorial problem both for arbitrary and orthogonal polygons [13, 17]. Orthogonal (i.e. rectilinear) polygons are interesting for they may be seen as abstractions of art galleries, for instance.

Many variants of art gallery problems have been proposed and studied over the years, with relevant practical applications [15, 19, 18]. An approach that has been neglected is that of finding algorithms that obtain approximate solutions [19], although, more recently, the use of some meta-heuristics (simulated annealing and genetic algorithms) was investigated by Canales [5]. Greedy algorithms for MINIMUM VERTEX GUARD were proposed by Ghosh [8] and Eidenbenz [7], the latter addressing also extensions to terrains. These approximation algorithms transform MINIMUM VERTEX GUARD instances into MINIMUM SET COVER instances, using decompositions of P. It may be required that each piece of the decomposition is either totally visible or totally invisible from each vertex, which guarantees that the approximation is exact [8]. This feature, that we call PIECE RESTRICTION, may render decompositions too grained.

[⋆] Partially funded by LIACC through *Programa de Financiamento Plurianual, Fundação para a Ciência e Tecnologia (FCT)* and *Programa POSI*, and by CEOC (Univ. of Aveiro) through *Programa POCTI, FCT, co-financed by EC fund FEDER.*

In [21], we proposed an anytime algorithm for solving MINIMUM VERTEX GUARD by successive approximations. It may start from a partition that does not satisfy PIECE RESTRICTION. Indeed, its main idea is to find a sequence of successively shorter intervals that enclose the minimum value $OPT(P)$.

To find a sequence of decreasing upper bounds, we successively refine an initial partition of the polygon. To approximate the optimum from below, we consider an increasing powerful subset of dominant pieces that are not visible by sections. A piece R is *visible by sections* iff no guard sees it completely but guards may collaborate to jointly guard it. When we decompose R, we obtain several smaller pieces that satisfy PIECE RESTRICTION. Dominance of visibility regions is exploited for reducing the problem dimension, as it is helpful to identify pieces or vertices that are irrelevant. It is also exploited for choosing the pieces that will be decomposed to get finer partitions.

Our algorithm applies both Computational Geometry and Constraint Programming techniques. In this paper we present some results of its experimental evaluation. The samples used consist of orthogonal polygons. For the initial decomposition, we took $\Pi_{\textit{r-cut}}$ (the partition of the polygon into rectangles by extension of all edges incident to its reflex vertices), which is constructed by vertical and horizontal sweeping. We describe the algorithms we developed to find how each rectangular piece of $\Pi_{\textit{r-cut}}$ is seen from each vertex and to compute the partition induced by the visibility regions in each piece. To make the paper self-contained, we first introduce some useful concepts and recall our approximation method.

2 Total Visibility, Visibility by Sections and Dominance

Given two points x and y in a simple polygon P, we say that x *sees* y (or y is *visible* to x) iff the closed segment $\overline{xy}$ does not intersect the exterior of P. The set $V(v)$ of all points of P visible to a vertex $v \in P$ is the *visibility region* of v. If $V(v)$ is the union of a polygon $Q \subseteq P$ and some line segments, we restrict $V(v)$ to Q, so that $V(v)$ will be a simple polygon.

A *partition* Π of polygon P is a division of P into sub-polygons (called *pieces*) that do not overlap except on their boundaries. A piece $R \in \Pi$ is *totally visible* from v iff v sees all points of R. It is *partially visible* iff v sees just a section $V_R^s(v)$ of R whose interior is nonempty. $V_R^s(v)$ is called a *visibility section*. A piece is *visible by sections* iff it is union of visibility sections. Γ_∞^Π refers to the set of pieces in Π that are not visible by sections and Γ_0^Π consists of those that are partially visible from at most one vertex. Clearly, $\Gamma_0^\Pi \subseteq \Gamma_\infty^\Pi$.

Our method performs lazy evaluation of visibility by sections, delaying the analysis of pieces not in Γ_0^Π until it becomes relevant. Some pieces or vertices may be found redundant in the meanwhile, that is *dominated* by other ones. In order to introduce this concept we need some further notation: G_v^t (and G_v^s) are the sets of all pieces that are totally (partially) visible from v and G_R^t (and G_R^s) are the sets of all vertices that are visible from every point of R (and from only part of the interior points of R). Thus, a vertex $v \in G_R^t$ iff $R \subseteq V(v)$ and $v \in G_R^s$ iff there are points p and q in the interior of R such that p is visible from v but q is not.

Definition 1. *Two vertices v_i and v_j are equivalent if $V(v_i) = V(v_j)$ and $\square$-equivalent if $G^t_{v_j} = G^t_{v_i}$. If $V(v_j) \subset V(v_i)$ then v_i strictly dominates v_j. If $G^t_{v_j} \subset G^t_{v_i}$ then v_i is $\square$-dominant over v_j. Two pieces $R_i, R_j \in \Pi$ are $\square$-equivalent if $G^t_{R_i} = G^t_{R_j}$. If $G^t_{R_i} \subset G^t_{R_j}$ then R_i $\square$-dominates R_j. If $R \in \Gamma^{\Pi}_{\infty}$ and $R' \in \Pi$, then R strictly dominates R' when $G^t_R \subset G^t_{R'}$ (implying that R' is necessarily guarded whenever R is guarded). If $R, R' \in \Gamma^{\Pi}_{\infty}$ and $G^t_R = G^t_{R'}$, then R and R' are equivalent.*

Example 1. In order to illustrate these notions, we present as small example in Fig. 1.

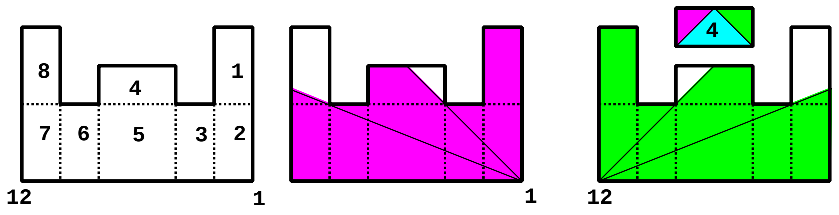

Fig. 1. The initial partition $\Pi_{\textit{r-cut}}$ and the visibility regions of vertices 1 and 12 (in CCW order). Pieces 1 and 8 are in Γ^{Π}_0, as well as 2 and 7. $G^t_{R_1} = \{v_1, v_2, v_3, v_4\}$ and $G^s_{R_1} = \{v_{12}\}$. $G^t_{R_2} = \{v_1, v_2, v_3, v_4, v_5, v_8, v_9, v_{12}\}$ and $G^s_{R_2} = \{v_7\}$. Piece 2 is redundant since piece 1 strictly dominates (and $\square$-dominates) it. $G^t_{R_4} = \{v_5, v_6, v_7, v_8\}$ and $G^s_{R_4} = \{v_1, v_{12}\}$. Piece 4 is visible by sections: the partition induced by its visibility sections ($V^s_{R_4}(v_1)$ and $V^s_{R_4}(v_{12})$) is shown also on the right, the triangle in the center being redundant.

3 Solving Minimum Vertex Guard by Successive Approximations

Our method solves a sequence of MINIMUM VERTEX GUARD problems, each one with the additional restriction that guards will not cooperatively guard pieces that they do not totally see on their own (i.e., visibility by sections is disregarded). Such a problem is called MINIMUM VERTEX GUARD WITH PIECE RESTRICTION. Given a set of pieces $\mathcal{F}$, $OPT(\mathcal{F})$ and $OPT_{\square}(\mathcal{F})$ refer to the optimal values of MINIMUM VERTEX GUARD and of MINIMUM VERTEX GUARD WITH PIECE RESTRICTION w.r.t. $\mathcal{F}$, for guards located in vertices of P. If each piece in Π is totally visible from at least one vertex of P, which is true for $\Pi_{\textit{r-cut}}$, then $OPT(\mathcal{F}) \leq OPT_{\square}(\mathcal{F})$, for all $\mathcal{F} \subseteq \Pi$. We now present the skeleton of our approximation algorithm.

```
ApproxMinVertexGuard($P$)
  $\Pi$ := Decompose($P$)   (each piece must be totally visible from some vertex)
  Compute $G^t_v$, $G^s_v$, for all vertices $v$
  Compute $G^t_R$ and $G^s_R$ for all $R \in \Pi$
        (delay decomposition of pieces by visibility sections)
  $\Gamma$ := $\Gamma^{\Pi}_0$
  while ($OPT_{\square}(\Gamma) < OPT_{\square}(\Pi)$) do
     $\Gamma, \Pi$ := Refine($\Pi$).
```

The algorithm obtains a sequence of non-increasing intervals that enclose $OPT(P)$ and such that

$$OPT_{\Box}(\varGamma_i) \le OPT_{\Box}(\varGamma_{i+1}) \le OPT(P) \le OPT_{\Box}(\varPi_{i+1}) \le OPT_{\Box}(\varPi_i),$$

where $\varGamma_i$ is the set of pieces of the current partition $\varPi_i$ that are known to be not visible by sections, up to iteration i.

If for some i, the solver finds that $OPT_{\Box}(\varGamma_i) = OPT_{\Box}(\varPi_i)$, the approximation stops, because $OPT(P)$ was achieved.

If $OPT_{\Box}(\varGamma_i) < OPT_{\Box}(\varPi_i)$, the partition $\varPi_i$ will be refined: a piece will be decomposed to obtain a finer partition $\varPi_{i+1}$. For that, we consider pieces that are $\Box$-dominant in $\varDelta_{i+1}$ with

$$\varDelta_{i+1} = (\varPi_i \setminus \varGamma_i) \setminus \{R \mid R \text{ is } \Box\text{-dominated by a piece in } \mathcal{D}(\varGamma_i)\},$$

where $\mathcal{D}(\varGamma_i)$ is a maximal set of strictly dominant pieces in $\varGamma_i$. We select a $\Box$-dominant piece $R \in \varDelta_{i+1}$, compute the partition $\mathcal{Z}_R$ determined by its visibility sections and check whether each piece of $\mathcal{Z}_R$ is visible from any vertex in G^s_R.

If this is the case, R is visible by sections and, since each piece in $\mathcal{Z}_R$ satisfies PIECE RESTRICTION, we define $\varPi_{i+1} := (\varPi_i \setminus \{R\}) \cup \mathcal{D}(\mathcal{Z}_R)$ and $\varGamma_{i+1} := \varGamma_i \cup \mathcal{D}(\mathcal{Z}_R)$. Otherwise, $\varPi_{i+1} := \varPi_i$ and $\varGamma_{i+1} := \varGamma_i \cup \{R\}$. When $\varDelta_{i+1} = \emptyset$, the dominant classes in $\varGamma_i$ and $\varPi_i$ coincide and $OPT_{\Box}(\mathcal{D}(\varGamma_i)) = OPT(\mathcal{D}(\varPi_i)) = OPT(P)$.

Finding $OPT_{\Box}(\mathcal{F})$. The reformulation of MINIMUM VERTEX GUARD WITH PIECE RESTRICTION as MINIMUM SET COVER allows us to model it as a Constraint Satisfaction Optimization Problem (CSOP) and to solve it using Constraint Programming techniques [14]. If $V_{\mathcal{F}}$ is the set of vertices that totally see pieces of $\mathcal{F}$, the decision variables are: $X_v \in \{0,1\}$, for $v \in V_{\mathcal{F}}$ (1 iff a guard is placed at vertex v) and $Y_R \in \{0,1\}$, for $R \in \mathcal{F}$ (1 iff piece R is guarded). The constraints are (1)–(3)

$$\sum_{R \in \mathcal{F}} Y_R = |\mathcal{F}| \tag{1}$$

$$\sum_{v \in G^t_R} X_v \ge Y_R, \text{ for all } R \in \mathcal{F} \tag{2}$$

$$\sum_{R \in G^t_v} Y_R \;\ge\; X_v\,|G^t_v|, \text{ for all } v \in V_{\mathcal{F}} \tag{3}$$

and state that all pieces must be visible, that R is visible only if there exists $v \in G^t_R$ with $X_v = 1$, and that a guard at v sees all pieces in G^t_v, respectively. The goal is to minimize $\sum_{v \in V_{\mathcal{F}}} X_v$, that is to find $OPT_{\Box}(\mathcal{F})$, as well as an optimal guard set. Additional constraints may be imposed here to reduce the search space. For orthogonal polygons, Kahn et al. [11] showed that $OPT(P) \le \lfloor \frac{n}{4} \rfloor$. The upper and lower bounds on $OPT(P)$ that we obtain in each approximation step, impose stronger constraints.

In order to keep the models small, instead of using $\mathcal{F}$ and $V_{\mathcal{F}}$ to formulate and solve this CSOP model, we consider relevant sets $\mathcal{D}_{\Box}(\mathcal{F})$ and $\mathcal{D}_{\Box}(V_{\mathcal{F}})$. To obtain them, $\Box$-dominance for vertices and for pieces in $\mathcal{F}$ is applied jointly until the dominant sets get invariant [21].

Example 2. For the polygon given in Example 1, R_7 and R_2 are strictly dominated by R_8 and R_1 (so that, they will be discarded) and R_5 is $\square$-dominated by R_4, for instance, meaning that it is irrelevant during the computation of $OPT_\square(\Pi_0)$. If we apply jointly $\square$-dominance for pieces and for vertices, we get

$$\begin{aligned}
G^t_{[R_8]} &= \{v_9, v_{10}, v_{11}, v_{12}\}\\
G^t_{[R_1]} &= \{v_1, v_2, v_3, v_4\}\\
G^t_{[R_6,R_3]} &= \{v_1, v_4, v_5, v_8, v_9, v_{12}\}\\
G^t_{[R_4]} &= \{v_5, v_6, v_7, v_8\}
\end{aligned}$$

where $[R_6, R_3]$ means that these pieces are $\square$-equivalent. Then, for vertices, we obtain

$$\begin{aligned}
G^t_{[v_1,v_4]} &= \{[R_1], [R_6, R_3]\}\\
G^t_{[v_5,v_8]} &= \{[R_4], [R_3, R_6]\}\\
G^t_{[v_9,v_{12}]} &= \{[R_8], [R_3, R_6]\}\\
G^t_{[v_2,v_3]} &= \{[R_1]\}, \quad \square\text{-dominated by } [v_1, v_4]\\
G^t_{[v_6,v_7]} &= \{[R_4]\}, \quad \square\text{-dominated by } [v_5, v_8]\\
G^t_{[v_{10},v_{11}]} &= \{[R_8]\}, \quad \square\text{-dominated by } [v_9, v_{12}].
\end{aligned}$$

If we remove the $\square$-dominated vertices and apply $\square$-dominance again, we obtain

for pieces	for vertices
$G^t_{[R_8]} = \{[v_9, v_{12}]\}$	$G^t_{[v_1,v_4]} = \{[R_1]\}$
$G^t_{[R_1]} = \{[v_1, v_4]\}$	$G^t_{[v_5,v_8]} = \{[R_4]\}$
$G^t_{[R_4]} = \{[v_5, v_8]\}$	$G^t_{[v_9,v_{12}]} = \{[R_8]\}$
$G^t_{[R_6,R_3]} = \{[v_1, v_4], [v_5, v_8], [v_9, v_{12}]\}$ $\square$-dominated by, e.g., $[R_4]$	

so that, the reduced CSOP model, will just involve six decision variables, namely $Y_{[R_8]}$, $Y_{[R_1]}$, $Y_{[R_4]}$, $X_{[v_1,v_4]}$, $X_{[v_5,v_8]}$ and $X_{[v_9,v_{12}]}$, yielding $OPT_\square(\Pi_0) = 3$.

When we apply a similar procedure to $\Gamma_0^{\Pi_0} = \{[R_8], [R_1], [R_5], [R_7], [R_2]\}$, we will find that the classes of $\square$-dominant pieces and vertices are:

$$\begin{aligned}
G^t_{[R_8]} &= \{[v_9, v_{12}]\}\\
G^t_{[R_1]} &= \{[v_1, v_4]\}\\
G^t_{[v_1,v_4]} &= \{[R_1]\}\\
G^t_{[v_9,v_{12}]} &= \{[R_8]\}
\end{aligned}$$

This means that the reduced CSOP for $OPT_\square(\Gamma_0^{\Pi_0})$ involves four decision variables only. We see that $OPT_\square(\Gamma_0^{\Pi_0}) = 2$.

Since $2 = OPT_\square(\Gamma_0^{\Pi_0}) \leq OPT(P) \leq 3 = OPT_\square(\Pi_0)$, the partition must be refined: either R_3, R_4 or R_6 will be then decomposed and the approximation proceeds.

4 Computing Visibility

There exist linear algorithms that compute the visibility region of a point in a polygon [10, 12]. We are interested in finding the relative location of each piece of $\Pi_{r\text{-}cut}$, within each visibility region and these methods do not answer directly this question. So, we designed a new method, adapting work by Aronov et al. [1], whose main idea is the propagation of visibility cones (as shown in Fig. 2). Pieces that are *adjacent* to v

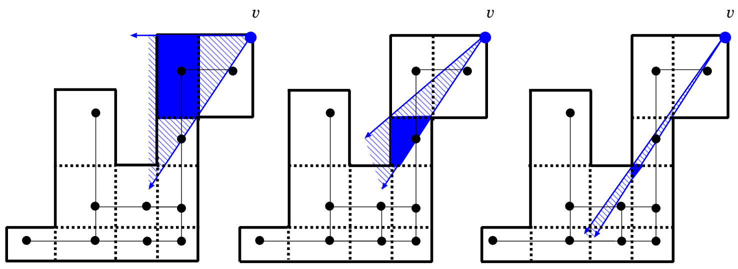

Fig. 2. Finding the visibility region of v by propagation of visibility cones.

(that is, that have v as vertex) propagate visibility to their neighbours and successively. The dual graph of the partition (that translates the adjacency relation) will be explored in breadth-first, starting from the pieces that are adjacent to v.

When v is a convex vertex there is a single piece, say $R_{0,0}$, to start the propagation. We mark $R_{0,0}$ as totally visible and put its adjacent pieces in the queue (in Fig. 2, $R_{0,0}$ has a single adjacent, which is enlighted on the left). $R_{0,0}$ determines the search directions: they are defined by the relative position of the vertex of $R_{0,0}$ that is diametrically opposite to v. In Fig. 2, these directions are (West,South).

Every reflex vertex has three adjacent pieces. The three quadrants they define may be explored at the same time. We just need to mark each adjacent piece as totally visible and insert their adjacent ones in the queue to start the propagation. Only pieces that are partially visible or totally visible may propagate visibility. Not all pieces will be visited, in general, but all that are visible from v will be visited. If $R_{i,j}$ is the head of the queue at a given step, the relative position of its centroid (wrt. v) determines the quadrant $R_{i,j}$ belongs to and the corresponding directions, say (d_1, d_2), of search. We determine the visibility section $V^s_{R_{i,j}}(v)$ by finding the intersection of $R_{i,j}$ with a visibility cone. This cone is defined by v and the *visible part* (i.e., *window*) of the edges shared by $R_{i,j}$ and its adjacent pieces in the directions $(-d_1, -d_2)$, which have been already visited. The propagation method ensures that $R_{i,j}$ was enqueued by some previously visited neighbour (either by $R_{i-1,j}$ or $R_{i,j-1}$).

After characterizing the visible section of $R_{i,j}$, we mark $R_{i,j}$ as already visited, enqueue each of its adjacent pieces, say $R_{i+1,j}$ and $R_{i,j+1}$, in the directions (d_1, d_2), provided they exist, are not already in the queue, and share with R_{ij} an edge that is *visible* (either totally or partially) from v. We say that an edge is visible if at least two of its interior points are visible from v. The method stops when the queue gets empty.

The propagation method may be slightly adapted to deal with other kind of partitions, namely the partition determined by extending the horizontal (or vertical) edges only, which has at most $r + 1$ pieces (where r is the number of reflex vertices), or triangulations. To simplify the analysis of the visibility section, it is important that the pieces are convex.

5 Finding the Partition Induced by Visibility Sections

We now describe the algorithm used for finding the partition $\mathcal{Z}_R$ induced by the set of visibility sections on R (ie., by $\{V_R^s(v) \mid v \in G_R^s\}$), and to determine the vertices that see each piece of $\mathcal{Z}_R$. The example given in Fig. 3 illustrates its main idea. The depicted

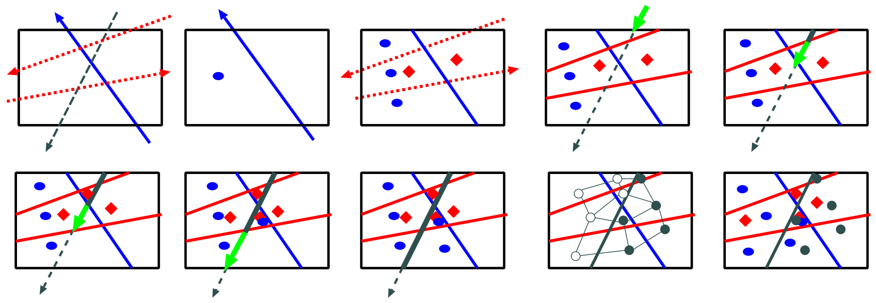

Fig. 3. The construction of $\mathcal{Z}_R$. In this case, R is not visible by sections. We show the insertion of the third visibility section (i.e., fourth segment) in the arrangement step by step and the faces it sequentially splits. The dual graph represented will be used to "propagate" visibility.

rays support the oblique edges of three visibility sections, one of them being defined by the two dotted rays. The visible part is to the left of each cutting ray (considering the CCW orientation for the visibility sections).

The pieces of $\mathcal{Z}_R$ are the faces of the arrangement of line segments that define the edges of the visibility sections and of R itself. Each visibility section is a simple convex polygon, always perfectly determined by one or two oriented oblique edges. Indeed, we saw that the section $V_R^s(v)$ is the intersection of R and a visibility cone, so that it has at most two oblique edges. Since R belongs to $\Pi_{\text{r-cut}}$, the horizontal and vertical edges of its visibility sections are necessarily contained in its boundary. Thus, these edges are redundant for the arrangement. When the vertices of $V_R^s(v)$ are labeled in counterclockwise order, $V_R^s(v)$ is to the left of each of its (oriented) edges, and, thus, it is to the left of its oblique edges. Lemma 1 states that $\mathcal{Z}_R$ satisfies PIECERESTRICTION, a property that has been proved and used in previous works, namely by Ghosh [8], Bosé et al. [4] and Guibas et al. [9].

Lemma 1. *Any two points in the interior of each face $\mathcal{C}$ of $\mathcal{Z}_R$ see exactly the same vertices of P. The interior of $\mathcal{C}$ is either totally visible or invisible to each vertex of P.*

This results is helpful for finding the vertices that see each piece of $\mathcal{Z}_R$, since it implies that we may decide whether a face $\mathcal{C}$ is totally visible from a vertex by checking visibility for a single point w in its interior. As the faces are convex sets [8], we could choose the *witness point* w as $\frac{1}{3}(w_1+w_2+w_3)$, for any three noncollinear (e.g., consecutive) vertices w_1, w_2 and w_3 of $\mathcal{C}$, based on a classical theorem by Carathéodory [16].

The algorithm we implemented for finding $\mathcal{Z}_R$ and the set of vertices of P that see each of its faces is essentially the algorithm for constructing line arrangements, described by Berg et al. [3]. We have adapted it to cope with data about visibility. The arrangement is constructed incrementally, starting from R. In each iteration, we consider a new section and insert its oblique edge(s), obtaining a new arrangement. As we saw, one or two oblique segments are inserted per iteration. When we insert a segment, at most a linear number of faces are splitted. The order in which they are visited is fixed by the segment's orientation, as we see in Fig. 3.

At each step, every face has an associated *guard set*, that consists of the vertices v in G_R^s from which it is (totally) visible, $V_R^s(v)$ being a visibility section already analysed. When a face is split, the two new faces inherit its guard set. Once we have finished the insertion of the oblique edges that define the current visibility section $V_R^s(v')$, we update the guard sets. The vertex v' will be added to the guard sets of the faces that are visible to v'. For that, we visit the dual graph of the arrangement in breadth-first, starting from the subface of the face that was splitted first and that is visible from v'.

Based on Lemma 1, when we reach a face $\mathcal{C}$, we decide whether it is visible from v' by computing its witness point and checking whether such point is visible from v'. It is easy to analytically check the position of v' wrt $V_R^s(v')$, using the algebraic expression(s) that defines the oblique edge(s). If $\mathcal{C}$ is visible, its adjacent faces are added to the queue (if they are not already there). If $\mathcal{C}$ is not visible, then none of its adjacent faces is added to the queue. In this way, we avoid visiting many faces that are not visible from v', without loosing the faces that are visible. The dual graph is a connected graph and the visibility region is connected also, so that, all visible faces are accessible from the first one.

6 Implementation and Experimental Results

We developed a prototype implementation of our method with a graphical interface, using Java and SICStus Prolog in an hybrid way. The Prolog system is essentially applied for finding $OPT_\square(\Pi)$ and $OPT_\square(\Gamma)$ for the sequence of partitions, by solving the associated CSOPs using `clpfd` (a SICStus module for dealing with constraints in finite domains [6]). SICStus is invoked from Java, using files to pass the relevant data, namely the program that defines the CSOP model (which SICStus then compiles and executes).

To carry out the experiments we used three kinds of polygons, all of them being *grid n-ogons* i.e., n-vertex orthogonal polygons that may be placed in a $n/2 \times n/2$ unit square grid and that do not have collinear edges [20]. One sample consists of randomly generated n-ogons created using RANDOM INFLATE-CUT [20]. The other two samples consist of MINAREA and FAT grid n-ogons: the classes for which the number of pieces of $\Pi_{\text{r-cut}}$ is minimum and maximum, the former restricted to have area $2r+1$. Fig. 4 contains the first members of these classes.

The maximum number of pieces of $\Pi_{\textit{r-cut}}$ is $(3r^2+6r+4)/4$ if r is even and $3(r+1)^2/4$ if r is odd, being $r=(n-4)/2$ the number of reflex vertices [2].

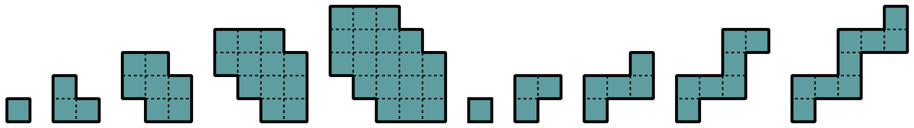

Fig. 4. The unique FAT and MINAREA grid n-ogons for $n = 4, 6, 8, 12, 14$ (up to symmetry).

We think FAT and MINAREA are representative of extremal behaviour, as the experimental results given in Fig. 5 show.

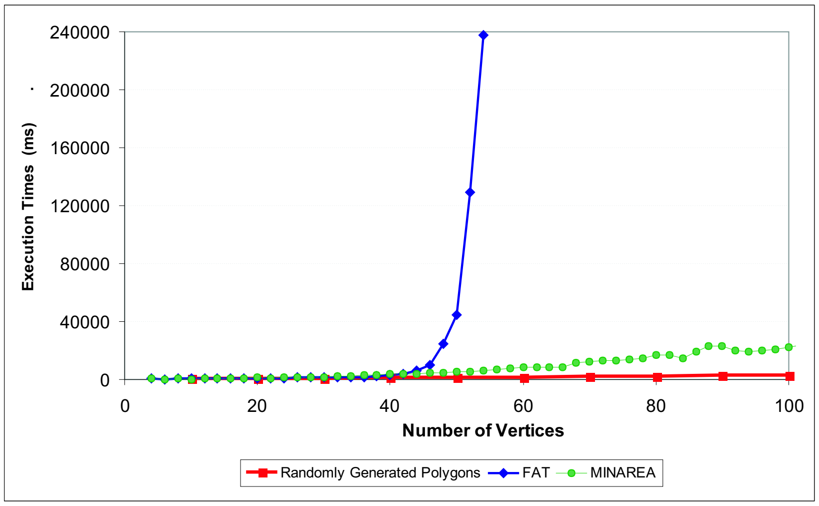

Fig. 5. Execution time for the three classes of samples, showing an exponential increase for FAT.

Although each FAT n-ogon requires only two guards for $n \geq 12$, the computational cost of the construction and update of Hasse diagrams for the dominance relations becomes too high in the current implementation. The reduced CSOP models are then solved quite quickly, but we think that the dominance checking module needs some improvement.

In contrast, for both MINAREA and randomly generated samples, the results show almost linear growth in the execution time: the correlation coefficient is 0.955 and 0.9996, respectively. The value $OPT(P)$ is close to $\frac{n}{7}$, although a bit greater. This value was suggested by the empirical results of Canales [5], but it is quite less than the theoretical bound $\lfloor \frac{n}{4} \rfloor$. These results are given in Fig. 6.

In Fig. 7 we present the average number of iterations that were needed to reach $OPT(P)$, again with almost linear growth, for all samples, with correlation coefficients 0.999, 0.943 and 0.999, respectively.

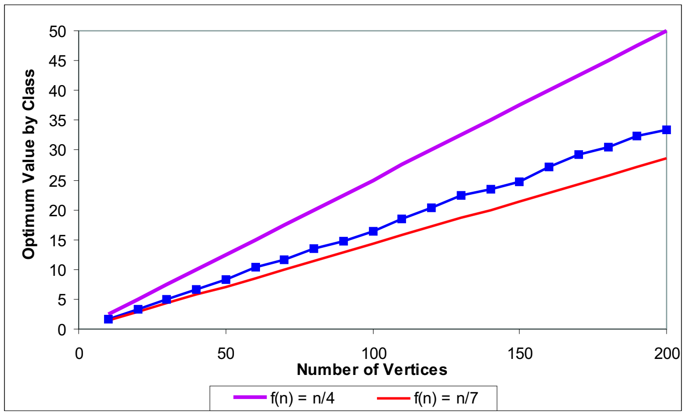

Fig. 6. Average number of guards.

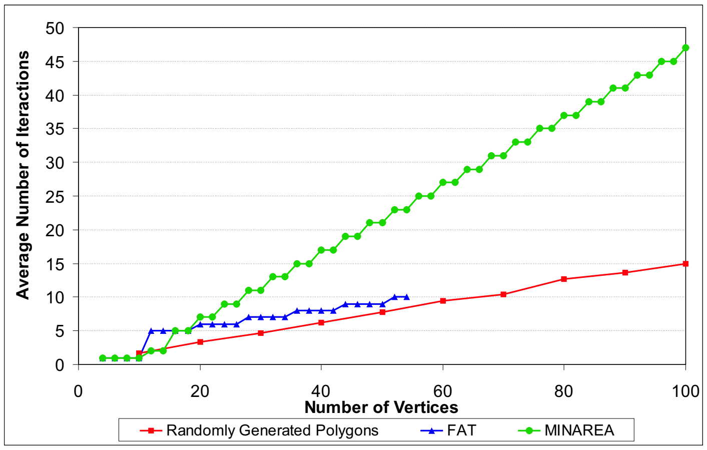

Fig. 7. Average number of iterations.

7 Conclusions and Future Work

We addressed the MINIMUM VERTEX GUARD problem, reviewing an anytime method we proposed in [21], that solves the problem by reduction to a sequence of MINIMUM SET COVER problems. This sequence is created by successive refinements of a partition of the given polygon. We reported on results of its experimental evaluation. For that purpose, we focused on the case of rectilinear polygons, and analysed in detail some of the algorithms developed for the implementation of the method. Neverthless, both our approximation method and the algorithms described in the paper for the analysis of visibility extend to generic simple polygons and partitions. The sole requirement is that each piece in the initial partition be convex and totally visible from at least one vertex. It is also possible to apply our approximation method when polygons have holes.

Each MINIMUM SET COVER problem is solved as a constraint satisfaction optimization problem. In order to keep its dimension small, we exploit visibility dominance, ordering vertices and pieces accordingly. Experiments have shown that this may become quite a burden when the number of pieces of the initial partition is large, so that we plan to improve this issue in future implementations.

No specific heuristics were designed to drive the refinement of the partition. We simply selected any of the dominant (more precisely $\square$-dominant) pieces that is likely visible by sections, starting from the ones that are (partially) visible from less vertices. We think it may be still worthwhile to devise heuristics for choosing a piece among these minimal ones, although the computational results point to a linear increase in the number of iterations (i.e., decomposition steps) with the number of vertices, for randomly generated polygons.

Our current work is also focusing on extensions of our method to solve problems where the visibility range is limited to π or $\pi/2$.

References

1. Aronov B., Guibas L., Teichmann M., Zang L.: Visibility queries and maintenance in simple polygons. Discrete and Computational Geometry, **27** (2002) 461–483.
2. Bajuelos A.L., Tomas A.P., Marques F.: Partitioning orthogonal polygons by extension of all edges incident to reflex vertices: lower and upper bounds on the number of pieces. In A. Laganà et al. (eds): Proc. of ICCSA 2004, LNCS 3045, Springer Verlag (2004) 127–136.
3. De Berg M., Van Kreveld M., Overmars M., and Schwarzkopf O.: Computational Geometry (Algorithms and Applications), Springer-Verlag (1997).
4. Bose P., Lubiw A., and Munro J. I.: Effi cient visibility queries in simple polygons. Proc. 4th Canad. Conf. Comput. Geom. (1992) 23–28.
5. Canales S.: Metodos Heuristicos en Problemas Geometricos: Visibilidad, Iluminacion y Vigilancia. PhD Thesis, Universidad Politecnica de Madrid, Spain (2004).
6. Carlsson M., Ottosson G., Carlson B.: An Open-Ended Finite Domain Constraint Solver. In Proceedings of PLILP'97, LNCS 1292. Springer-Verlag, (1997) 191–206.
7. Eidenbenz S.: Aproximation algorithms for terrain guarding. Information Processing Letters **82** (2002) 99–105.
8. Ghosh, S. K.: Approximation algorithms for art gallery problems. Proc. Canadian Information Processing Society Congress. (1987) 429–434.

9. Guibas L. J., Motwani R., Raghavan P.: The robot localization problem in two dimensions.Proc. 3rd ACM–SIAM Symp. Discrete Algorithms (1992) 259–268.
10. Joe B., Simpson R.B.: Corrections to Lee's Visibility Polygon Algorithm. BIT **27** (1987) 458–473.
11. Kahn, J., Klawe, M., Kleitman, D.: Traditional galleries require fewer watchmen. SIAM J. Algebraic and Discrete Methods **4** (1983) 194–206.
12. Lee, D. T.: Visibility of a simple polygon. Computer Vision, Graphics, and Image Processing **22** (1983) 207–221.
13. Lee, D. T., Lin, A. K.: Computational complexity of art gallery problems. IEEE Transaction on Information Theory **IT-32** (1986) 276–282.
14. Marriott K., Stuckey P.: Programming with Constraints – An Introduction. MIT Press (1998).
15. O'Rourke J.: Art Gallery Theorems and Algorithms. Oxford University Press (1987).
16. Schrijver A.: Theory of Linear and Integer Programming, Wiley-Interscience (1986).
17. Schuchardt D., Hecker H.: Two NP-hard problems for ortho-polygons. Math. Logiv Quart **41** (1995) 261–267.
18. Shermer T.: Recent results in art galleries. Proc. IEEE **80:9** (1992) 1384–1399.
19. Urrutia, J.: Art gallery and illumination problems. In J.-R. Sack and J. Urrutia, editors, Handbook on Computational Geometry. Elsevier (2000).
20. Tomas, A. P., Bajuelos, A. L.: Quadratic-Time Linear-Space Algorithms for Generating Orthogonal Polygons with a Given Number of Vertices. In A. Laganà et al. (Eds): Proc. of ICCSA 2004, LNCS 3045, Springer-Verlag (2004) 117–126.
21. Tomas A. P., Bajuelos A. L., Marques F.: Approximation algorithms to minimum vertex cover problems on polygons and terrains. In P.M.A Sloot et al. (Eds): Proc. of ICCS 2003, LNCS 2657, Springer-Verlag (2003) 869-878.