Representing a Planar Straight-Line Graph Using Few Obstacles*

Matthew P. Johnson† Deniz Sarıöz‡

Abstract

An *obstacle representation* of a planar straight-line graph (PSLG) G consists of the choice and placement of a set of opaque polygonal obstacles in such a way that the visibility graph on $V(G)$ induced by the obstacles equals G (i.e., u and v are visible to one another iff $(u, v) \in E(G)$). We investigate the problem of computing an obstacle representation of a PSLG, using a minimum number of obstacles. We call this minimum size the *obstacle number* of the drawing, and the problem of computing it *ORPG*.

First, we show that ORPG is NP-hard by reduction from planar vertex cover, resolving a question posed by Sarıöz (CCCG 2011 [7]). Second, we give a reduction from ORPG to maximum degree 3 planar vertex cover. Since this reduction preserves solution values, it follows that ORPG admits a polynomial-time approximation scheme (PTAS) and is fixed-parameter tractable (FPT).

1 Introduction

Let G be a planar straight-line graph (PSLG) with vertices in general position; that is, a straight-line drawing of a planar graph with no edge crossings and no three vertices on a line in which the vertices are identified with their positions. We refer to the open line segment between a pair of non-adjacent graph vertices as a *non-edge* of G. An *obstacle representation* of G is a pair $(V(G), \mathcal{O})$ where $\mathcal{O}$ is a set of polygons (not necessarily convex) called *obstacles*, such that:

1. G does not meet any obstacle, and
2. every non-edge of G meets at least one obstacle.

Equivalently, G is the visibility graph on $V(G)$ induced by the obstacles in $\mathcal{O}$. The size of an obstacle representation is the cardinality of $\mathcal{O}$. Denote by *ORPG* the problem of computing a minimum-size obstacle representation of the PSLG G (the optimum of which is called the *obstacle number* of G). Alpert, Koch, and Laison introduced the notions *obstacle representation* and *obstacle number* for abstract graphs [3] and noted that in any minim*al* (not necessarily mininu*um*) obstacle representation, each obstacle can be identified with the face it lies in. Hence, we will use the terms *face* and *obstacle* interchangeably. If the faces have weights then we can seek a minimum-weight obstacle representation.

Finding a minimum-size obstacle representation of a straight-line graph drawing was treated as a computational problem in the more general setting in which the drawing of G, and G itself, need not be planar [7]. This problem was reduced to hypergraph transversal (hitting set), with $O(n^4)$ faces available to pierce $O(n^2)$ non-edges ($O(n)$ faces and $\Theta(n^2)$ non-edges in the PSLG special case). A randomized $O(\log OPT)$-approximation algorithm based on bounding the Vapnik-Chervonenkis dimension of the corresponding hypergraph family was given in [7]. Left open was the question of whether better approximations or perhaps optimal algorithms were feasible.

In this paper we give partial answers to that question. We show that computing the obstacle number is NP-hard already in the special case of straight-line drawings (without edge crossings); nonetheless, we show that problem admits a polynomial-time approximation scheme (PTAS) and is fixed-parameter tractable (FPT). We show hardness by a reduction from planar vertex cover; the positive results are consequences of a solution value-preserving reduction to maximum degree 3 planar vertex cover.

2 Reduction from planar vertex cover

Theorem 1 *ORPG is NP-hard.*

Proof. We reduce from planar vertex cover. Recall that in the decision version of planar vertex cover, we are given an abstract planar graph G having (without loss of generality) no isolated vertex, and a number k. Let $n = |V(G)|$, $m = |E(G)|$, and $f = n - m + 2$ (the number of faces in any crossing-free planar drawing of G). We will transform G in polynomial time into a PSLG $\tilde{G}$ in such a way that G has a vertex cover of size k if and only if $\tilde{G}$ has an obstacle representation of size k' (for k' defined below).

First, we construct from the planar vertex cover instance G a planar vertex cover problem instance G^3 with maximum degree 3, adapting and extending the

*Research supported by grants from NSA (47149-0001) and PSC-CUNY (63427-0041). Research performed while the author was at the City University of New York Graduate Center.

†Lehman College and the Graduate Center, City University of New York, mpjohnson@gmail.com

‡Google, Inc., sarioz@google.com

construction of [6]. The graph G^3 admits a vertex cover of size k' if and only if G admits a vertex cover of size k. Second, we construct an ORPG instance $\tilde{G}$ in such a way that an obstacle representation of $\tilde{G}$ will correspond to a vertex cover of G^3 of the same size, and vice versa. Our complete reduction involves first performing the steps involved in the reduction of [6] because it is the particular class of degree-3 instances constructed by [6] that we then convert into ORPG instances, not the entire class of degree-3 vertex cover instances.

Constructing the maximum degree 3 planar vertex cover instance G^3. The planar graph G^3 is constructed as follows. We transform each vertex v_i of G into a cycle C^i of length $2b_i$, with $b_i \in \deg(v_i)+\{0,1,2\}$ (with the exact value decided below). We color the vertices of C^i alternating between blue and red. We then create a single leaf vertex z_i adjacent to some arbitrary red vertex of C^i. We transform each edge (v_i, v_j) of G into a path P_{ij} with *three* edges whose endpoints are *distinct* blue vertices of C^i and C^j. Finally, we create f copies of the 3-vertex path graph P_3, each constituting a component of G^3.

We claim that G has a vertex cover of size at most k if and only if G^3 has one of size at most $k' = k + f + m + \sum_i b_i$. (See appendix.)

Constructing the ORPG instance $\tilde{G}$. In the remainder of the proof, we show how to "implement" the graph G^3 as an equivalent ORPG problem instance. The basic building blocks of the construction are *empty triangles* and *diamonds*. An *empty triangle* is a face of a PSLG that is surrounded by three edges and has no vertex inside. A *diamond* consists of two empty triangles sharing an edge and having their *four* vertices in convex position. Observe that a diamond contains a non-edge between two of its vertices. Hence at least one empty triangle of every diamond must be chosen in an obstacle representation. The f copies of P_3 in G^3 will match the faces of $\tilde{G}$ besides empty triangles, all of which must be chosen. The remaining vertices of G^3 will match the empty triangles of $\tilde{G}$, such that the edges among them match the diamonds of $\tilde{G}$. Hence there is a natural bijection between vertex covers of G^3 and obstacle representations of $\tilde{G}$.

To begin the construction, we use the linear-time algorithm of de Fraysseix, Pach, and Pollack [5] to obtain a planar drawing of G on a $O(n) \times O(n)$ portion of the integer lattice and then perturb the coordinates to obtain general position. (We do not distinguish between G and this imbedding.) We first visualize $\tilde{G}$ as a bold drawing [8] of G, whose vertices are represented by small disks and edges by solid rectangles: we draw each vertex u_i of G as a disk D_i about u_i (with boundary $\tilde{C}^i$), and every edge u_iu_j as a solid rectangle R_{ij}. See Fig. 1(a). Each R_{ij} has two vertices t_{ij},v_{ij} on $\tilde{C}^i$ and two vertices t_{ji}, v_{ji} on $\tilde{C}^j$ such that the line u_iu_j is a midline of R_{ij}, and $t_{ij}u_iv_{ij}t_{ji}u_jv_{ji}$ is a counterclockwise ordering of the vertices of a convex hexagon.

We draw the disks small enough to ensure that they are well-separated from one another. We set the radius r of every disk to the smaller of $1/4$ and half of the minimum distance between a vertex u_i and an edge u_ju_k ($j \neq i \neq k$) of G. To fix a single width for all rectangles (i.e., $||t_{ij} - v_{ij}||$), we set a global angle measure α to the smaller of $45°$ and half of the smallest angle between two edges of $E(G)$ incident on the same vertex of $V(G)$.

$\tilde{G}$ is modeled on the bold drawing, by implementing each edge of G (path P_{ij} of G^3) with an edge gadget and each vertex of G (cycle C^i of G^3) with a vertex gadget. The edge gadget, consisting of four triangles forming three diamonds, is shown in Fig. 1(b). (Note that each pair $v_{ij}v_{ji}$ defines a non-edge.)

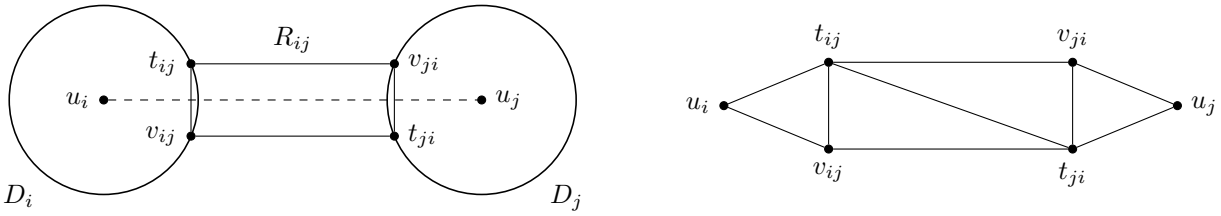
The vertex gadget is a modified wheel graph whose triangles correspond to the vertices of cycles C^i in G^3 (see Fig. 2). On every circle $\tilde{C}^i$, for every edge u_iu_j in G, we color blue the arc of measure α centered about the intersection of circle $\tilde{C}^i$ with u_iu_j (a non-edge in $\tilde{G}$). We place t_{ij} and v_{ij} at the endpoints of this arc so that $t_{ij}u_iv_{ij}$ is a counterclockwise triple. By the choice of α, all blue arcs are well-separated, and hence the rectangles are well-separated from one another and from other disks, by the choice of r. We color the remaining arcs red to obtain a red-blue striped pattern on each circle $\tilde{C}^i$, corresponding in color to the vertices of the corresponding C^i in G^3.

On every circle $\tilde{C}^i$, we will add the remaining edges between consecutive vertices of $\tilde{C}^i$ to complete the union of the triangles $t_{ij}u_iv_{ij}$, forming a wheel graph on hub u_i, such that every pair of triangles sharing a spoke form a diamond. If a red arc has measure at least $180° - \alpha$, however, we must add additional spokes. By the general position assumption, at most one red arc per wheel can have such great measure. If such a red arc has measure less than $270°$, we divide it evenly into *three* parts and color the middle part blue (see Fig. 2); otherwise, we divide it evenly into *five* parts and color the second and fourth parts blue (see Fig. 3), maintaining the striped pattern in both cases. We place dummy t_i and v_i vertices[1] at the newly created (*zero*, *two* or *four*) arc endpoints. Finally, we add the requisite edges to complete the wheel graph.

We place a vertex $\tilde{z}_i$ on an arbitrary red arc of $\tilde{C}^i$ and connect it in $\tilde{G}$ to the end vertices (say t_{ij} and v_{ik}) of that arc. Thus an empty triangle $t_{ij}z_iv_{ik}$ is formed in $\tilde{G}$ as part of a diamond with u_i, corresponding to z_i and its incident edge in G^3.

In the unbounded face of $\tilde{G}$ we place two isolated vertices inducing a non-edge inside the unbounded face,

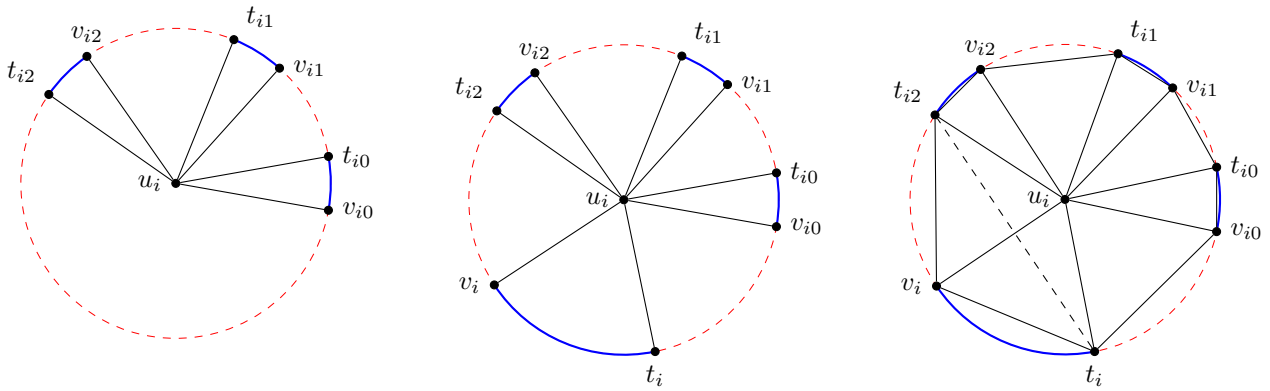
[1] Dummy vertices have no adjacencies with any vertices outside of D_i.



(a) Bold drawing of edge u_iu_j.

(b) Gadget in $\tilde{G}$ for edge u_iu_j of G.

Figure 1: Bold drawing and edge gadget for an edge of G.

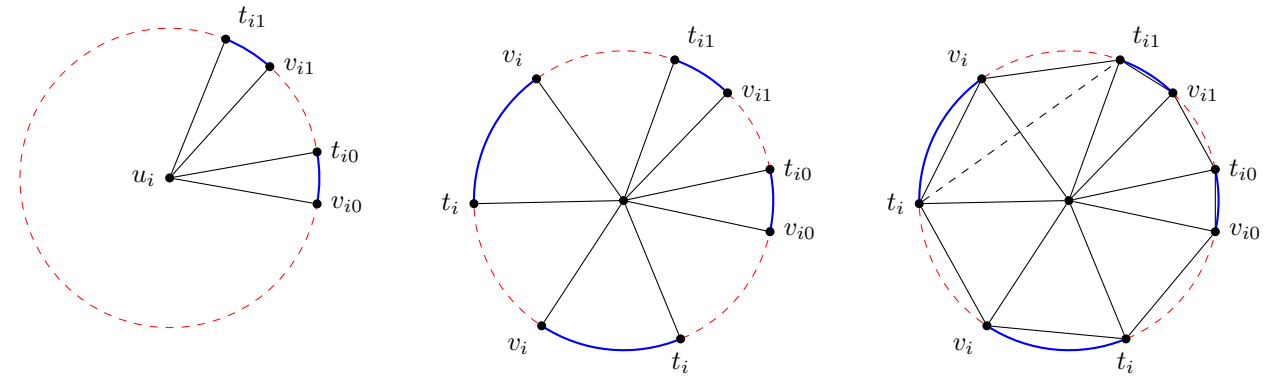


(a) Initial circle with blue (solid) arcs of measure α and red (dashed) arcs has a large red arc of measure in $[180° - \alpha, 270°)$.

(b) After subdividing the large red arc into *three*, coloring its middle part blue, and adding dummy v_i and t_i vertices.

(c) In the resulting wheel graph, each pair of triangles sharing an edge induce a diamond. One diamond's non-edge is shown dashed.

Figure 2: Constructing the wheel graph drawing in the case of a large red arc.



(a) An initial circle with blue (solid) and red (dashed) arcs has a very large red arc, of measure at least 270°.

(b) After subdividing the very large red arc into *five*, coloring its second and fourth parts blue, and adding dummies v_i and t_i.

(c) In the resulting wheel graph, each pair of triangles sharing an edge induce a diamond. One diamond's non-edge is shown dashed.

Figure 3: Constructing the wheel graph drawing in the case of a *very* large red arc.

thus requiring this face to be chosen in any solution. Every non-triangular face of $\tilde{G}$ must be selected as an obstacle, since every simple polygon with at least 4 vertices has an internal diagonal (i.e., a non-edge). The selection of these faces are forced moves and corresponds to the selection, in a vertex cover for G^3, of the central vertex of each P_3.

Since each pair of neighboring triangles in $\tilde{G}$ indeed form a diamond and every non-triangular face is indeed a forced move, the result follows. □

Remark 2 *To represent coordinates* exactly *as described would require a very permissive unit-cost RAM model of computation in which it is possible to represent real numbers and perform arithmetic and trigonometric functions in unit time. The reduction above can be modified in such a way that each vertex position of $\tilde{G}$ is represented using $O(\log n)$ bits.*

3 Reduction *to* vertex cover

Theorem 3 *Weighted ORPG is reducible to weighted max deg 3 planar vertex cover by an optimal solution value-preserving reduction.*

Proof. Given a PSLG G on n vertices in general position, we construct a graph $\hat{G}$ that admits a vertex cover of cost k if and only if G admits an obstacle representation of cost k.

Every bounded non-triangular face of G must be selected as an obstacle; moreover, the unbounded face must be chosen if and only if its convex hull boundary contains a non-edge. Since these are forced moves, we henceforth assume without loss of generality that every non-edge we must block meets at least two faces. Recall that an *empty triangle* is a bounded face on three vertices not containing any other vertices, and that a *diamond* consists of two empty triangles that share an edge and have their four vertices in convex position.

We claim that every non-edge must meet the two triangles forming some diamond, and hence meets those triangles' shared edge. (See appendix.) We now define $\hat{G}$, which is a subgraph of the dual of G: each edge of $\hat{G}$ corresponds to a diamond of G. The graph $\hat{G}$ is induced by these edges (with vertex weights set to the corresponding face weights). For each diamond, at least one its triangles must be chosen in any obstacle representation. Thus every obstacle representation of $\hat{G}$ corresponds to a vertex cover of G of the same cost, and vice versa. □

From Theorem 3 we immediately obtain the following.

Corollary 4 *Weighted ORPG admits a polynomial-time approximation scheme (PTAS) [4] and is fixed-parameter tractable (FPT) [9, 2].*

4 Discussion

In this short paper we studied the problem of finding a small set of obstacles that block *all* of a straight-line graph drawing's non-edges and *none* of its edges; we found the problem in this setting to be as easy as planar vertex cover (and hence, e.g., to admit a PTAS). For the more general setting of a straight-line drawing in the plane that might include edge crossings, the best positive result remains the $O(\log OPT)$ approximation of [7]. A natural generalization of this problem is to find a minimum obstacle representation of an abstract graph—i.e., an embedding in the plane of nodes *and* obstacles—but unfortunately to our knowledge this problem remains open.

We might also consider related optimization objectives. In a budgeted setting limiting us to the use of only k obstacles, we might permit obstacle representations of G that are only approximately accurate. Two natural relaxations of the original problem suggest themselves.

First, we could seek to maximize the number of non-edges that are blocked by the k obstacles, subject to the restriction that no edges of G are blocked. Analogous to the vertex cover reduction above, this problem reduces to (planar) max-vertex cover, and so admits a 3/4 approximation [1]. (Absent the restriction that the graph drawing be planar, this problem forms an instance of maximum coverage, which is approximable within $1 - 1/e$ by the greedy algorithm.)

Second, we could seek to minimize the number of G's edges that are blocked ("accidentally") by the k obstacles, subject to the restriction that *all* G's non-edges are blocked. Unfortunately, approximating this problem within *any* multiplicative factor is NP-hard, since for k equal to G's obstacle number, the optimal number of non-edges blocked by k obstacles is 0, and so any such approximation algorithm would solve the original problem optimally.

References

[1] A. A. Ageev and M. Sviridenko. Approximation algorithms for maximum coverage and max cut with given sizes of parts. In *IPCO*, pages 17–30, 1999.

[2] J. Alber, H. Fernau, and R. Niedermeier. Parameterized complexity: exponential speed-up for planar graph problems. *J. Algorithms*, 52(1):26–56, 2004.

[3] H. Alpert, C. Koch, and J. D. Laison. Obstacle numbers of graphs. *Discrete & Computational Geometry*, 44(1):223–244, 2010.

[4] B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. ACM*, 41:153–180, January 1994.

[5] H. de Fraysseix, J. Pach, and R. Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990.

[6] M. R. Garey and D. S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM Journal of Applied Mathematics*, 32:826–834, 1977.

[7] D. Sariöz. Approximating the obstacle number for a graph drawing efficiently. In *CCCG*, 2011.

[8] M. J. van Kreveld. Bold graph drawings. *Comput. Geom.*, 44(9):499–506, 2011.

[9] M. Xiao. A note on vertex cover in graphs with maximum degree 3. In *COCOON*, pages 150–159, 2010.

Appendix

Claim 5 *G has a vertex cover of size at most k if and only if G^3 has one of size at most $k' = k+f+m+\sum_i b_i$.*

Proof. ($\Rightarrow$): For each vertex v_i in a given vertex cover for G of size k, we select z_i and all the blue vertices of C^i, thus including an endpoint of each path P_{ij}; and for each v_i not in the cover, we select all the red vertices of C^i (a total so far of $k + \sum_i b_i$ vertices). Since for every path P_{ij} at least one of the cycles C^i and C^j will have all its blue vertices chosen, thus including at least one endpoint of P_{ij}, choosing one internal vertex from each P_{ij} (m more), and the central vertex of each P_3 (f more) suffices to complete a size k' vertex cover for G^3.

($\Leftarrow$): Given a vertex cover for G^3 of size k', we obtain a canonical vertex cover for G^3 of size $k'' \leq k'$ in the following way. Each copy of P_3 contributes at least one vertex to a cover, so have it contribute exactly its central vertex, for a total of f vertices. Each path P_{ij} contributes at least one of its internal vertices to cover its central edge. If both internal vertices of a path P_{ij} are in the given cover, take one internal vertex out and ensure that its blue neighbor is in, which makes for m internal vertices from these paths. Note that every cycle C^i contributes at least b_i vertices, lest some edge of the cycle be uncovered. This holds with equality only if C^i contributes (including "its" z_i) exactly its red vertices. Otherwise, ensure that C^i contributes exactly $1 + b_i$ vertices: "its" z_i and its blue vertices. Denote by k'' the size of this resulting canonical vertex cover. The cycles in G^3 contributing blue vertices therefore correspond to a vertex cover for G of size $k'' - f - m - \sum_i b_i \leq k' - f - m - \sum_i b_i = k$. □

Claim 6 *Every non-edge must meet the two triangles forming some diamond, and hence meets those triangles' shared edge.*

Proof. Assume for contradiction that some remaining non-edge s never crosses the diagonal edge of a diamond. Denote by u and v the endpoints of s, and orient the plane such that u is directly below v. Obtain a sequence of empty triangles $(f_0, f_1, \ldots, f_k)$ by tracing s from u (a vertex on f_0) to v (a vertex on f_k). Denote by v_i (for $1 \leq i \leq k$) the unique vertex in face f_i that is not a vertex of f_{i-1} (so that $v_k = v$). Without loss of generality, the reflex angle of f_0 and f_1 is to the right of s, which implies that v_1 is to the right of s. In order for f_2 to be the next face in this sequence, v_2 must be to the left of s. In general, in order for f_i to be the next face in this sequence, v_i must be on the other side of s from v_{i-1}. This pattern must continue indefinitely, lest two consecutive triangles form a diamond. The indefinite continuation of this pattern implies an infinite sequence of faces defined by s, and hence a contradiction. □