On Rectilinear Drawing of Graphs

Peter Eades[1,⋆], Seok-Hee Hong[2,⋆⋆], and Sheung-Hung Poon[3,⋆⋆⋆]

[1] School of Information Technologies, University of Sydney, Australia
{peter,shhong}@it.usyd.edu.au
[2] Department of Computer Science, National Tsing Hua University, Taiwan, R.O.C.
spoon@cs.nthu.edu.tw

Abstract. A *rectilinear drawing* is an orthogonal grid drawing without bends, possibly with edge crossings, without any overlapping between edges, between vertices, or between edges and vertices. Rectilinear drawings without edge crossings (*planar* rectilinear drawings) have been extensively investigated in graph drawing. Testing rectilinear planarity of a graph is NP-complete [10]. Restricted cases of the planar rectilinear drawing problem, sometimes called the "no-bend orthogonal drawing problem", have been well studied (see, for example, [13,14,15]).

In this paper, we study the problem of general *non-planar* rectilinear drawing; this problem has not received as much attention as the planar case. We consider a number of restricted classes of graphs and obtain a polynomial time algorithm, NP-hardness results, an FPT algorithm, and some bounds.

We define a structure called a "4-cycle block". We give a linear time algorithm to test whether a graph that consists of a single 4-cycle block has a rectilinear drawing, and draw it if such a drawing exists. We show that the problem is NP-hard for the graphs that consist of 4-cycle blocks connected by single edges, as well as the case where each vertex has degree 2 or 4. We present a linear time fixed-parameter tractable algorithm to test whether a degree-4 graph has a rectilinear drawing, where the parameter is the number of degree-3 and degree-4 vertices of the graph. We also present a lower bound on the area of rectilinear drawings, and a upper bound on the number of edges.

1 Introduction

A *rectilinear drawing* is an orthogonal grid drawing without bends, possibly with edge crossings, without any overlapping between edges and vertices. A graph is called a *rectilinear graph* if it admits a rectilinear drawing.

Rectilinear drawings without edge crossings (*planar* rectilinear drawings) have been extensively investigated in graph drawing. An undirected graph is *rectilinear planar* if it can be drawn in the plane such that every edge is a horizontal

⋆ Supported in part by grants from National Science Council (NSC), National Taiwan University and Academia Sinica in Taiwan, R.O.C.
⋆⋆ Supported in part by grant 98R0036-03 from National Taiwan University in Taiwan, R.O.C.
⋆⋆⋆ Supported in part by grant NSC 97-2221-E-007-054-MY3 in Taiwan, R.O.C.

D. Eppstein and E.R. Gansner (Eds.): GD 2009, LNCS 5849, pp. 232–243, 2010.

or vertical segment and no two edges cross. Garg and Tamassia [10] proved that testing rectilinear planarity of a graph is NP-complete. Restricted cases of the planar rectilinear drawing problem, sometimes called the "no-bend orthogonal drawing problem", have been well studied. Significant examples include linear-time algorithms to construct planar rectilinear drawings of *plane* graphs G of maximum degree three [13], subdivisions of planar triconnected cubic graphs [14], and series-parallel graphs of the maximum degree three [15].

Vijayan and Wigderson [16] considered the problem of rectilinear planar embedding with edge direction constraints. They gave a linear time testing algorithm and an $O(n^2)$ time embedding algorithm to construct such a drawing. Hoffman and Kriegel [11] improved the running time by presenting a linear time embedding algorithm. Bodlaender and Tel studied the connection between rectilinearity and angular resolution of planar graphs [5]. Recently, Eppstein [8] studied bendless orthogonal drawing problem in three dimensions, and showed that it is NP-complete to determine whether an arbitrary graph has such an embedding.

Many methods have been developed for constructing orthogonal drawings, aiming to minimize crossings as well as bends; see, for example, the original work of Batini et al. [2], or the "three phase method" of Biedl et al. [3]. However, non-planar rectilinear drawing has not been so well studied. Formann et al. [9] proved that given a graph G of maximum degree 4, it is NP-hard to decide whether G has a straight-line drawing with angular resolution $\frac{\pi}{2}$. In this paper, we investigate the problem of general non-planar rectilinear drawing.

Our work was also motivated by the recent development of *RAC (Right Angle Crossing)* drawing [6]. A RAC-drawing is a straight-line drawing of a graph, where all the crossings are at right angles. Research on RAC drawing arises from the controversial human experiments on the effects of crossing angles on performance of path tracing tasks. In 2006, Huang et al. [12] found that task response times decrease as the crossing angle increases, implying that drawings with large crossing angles are better for visualization. A rectilinear drawing can be regarded as an orthogonal-RAC drawing, that is, an orthogonal drawing with right angle crossings.

In this paper we present NP-completeness results and a linear time algorithm. The line between NP-completeness and a linear time algorithm is drawn with the concept of a "4-cycle block", defined in Section 2. We prove that one can test whether a graph that consists of a single 4-cycle block has a rectilinear drawing in linear time, and we can construct such a drawing in linear time. In contrast, we show that it is NP-complete to test whether a graph has a rectilinear drawing, even when it consists of a set of 4-cycle blocks connected by single edges. The NP-hardness remains even when the input graph consists of only degree-2 and degree-4 vertices.

Further, we present a linear time fixed-parameter tractable algorithm to test whether a degree-4 graph has a rectilinear drawing, where the parameter is the number of degree-3 and degree-4 vertices of the graph.

Note that the use of term "rectilinear drawing" is somewhat inconsistent in the literature. In 1941, Birkhoff [4] used the term in discussing density functions that can be approximated by sets of straight lines. In the Graph Theory literature the term has sometimes been used to mean the same as "straight-line drawing" (that is, without the requirement for orthogonality). In this paper we use the common meaning from the graph drawing literature, that is, as an orthogonal straight-line drawing.

This paper is organized as follows. In Section 2, we describe some basic properties of rectilinear drawings, including bounds on the area and density. Section 3 presents a linear time algorithm to test whether a graph with a "connected 4-cycle cover" has a rectilinear drawing. Section 4 presents hardness results on rectilinear drawings. In Section 5, we describe a fixed-parameter tractable algorithm to test whether a graph has a rectilinear drawing, where the parameter is the number of degree-3 and degree-4 vertices of the graph. Section 6 concludes with some open problems.

2 Some Basic Properties

In this Section we first describe some basic concepts and properties of rectilinear drawings, and then prove some simple bounds on their area and density.

2.1 Four-Cycle Covers and Blocks

Firstly we mention some simple consequences of the assumption that the drawing has no edge overlaps.

Lemma 1. *Suppose that G is a rectilinear graph. Then every vertex of G has degree at most 4 in G, no two 4-cycles of G share more than one edge, and no three 4-cycles share an edge.*

Motivated by Lemma 1, we say that a *4-cycle cover* C_4 of a graph $G = (V, E)$ is a set of 4-cycles that covers every edge, that is, every edge of G is in C_4. If G has a 4-cycle cover, then the *4-cycle incidence graph* is a graph G_4 with a vertex for each 4-cycle $c \in C_4$ and an edge (c, c') when the two 4-cycles c and c' share an edge. We say that C_4 is a *connected-4-cycle cover* if G_4 is connected.

Suppose that G is a graph of maximum degree 4 and that G' is a subgraph with a connected 4-cycle cover. If G' is maximal (that is, if edges or vertices are added then it no longer has a connected 4-cycle cover) then we say that G' is a *4-cycle block*. The concept of a 4-cycle block is critical in determining the line between polynomial time and NP-completeness.

A rectilinear drawing of a graph $G = (V, E)$ induces a partition $E = E_{hor} \cup E_{vert}$ of the edges into horizontal and vertical edges. Let $G_{hor} = (V, E_{hor})$ and $G_{vert} = (V, E_{vert})$. This partition has several useful properties.

Lemma 2. *Suppose that G is a rectilinear graph.*

(a) Both G_{hor} and G_{vert} are sets of disjoint paths.
(b) A path in G_{hor} meets a path in G_{vert} in at most one vertex.

(c) Every vertex of G is in exactly one path of G_{hor} and exactly one path of G_{vert}. (Note that we regard a single vertex as a trivial path).

Proof. Items (a) and (b) follow from the assumption that edges cannot overlap, and (c) is immediate. □

From Lemma 2 we can deduce some properties of cycles.

Lemma 3. *Suppose that G is a rectilinear graph. Then every cycle c in G contains at least two vertical edges and at least two horizontal edges. If c is a 4-cycle then the edges of c are alternately horizontal and vertical around c.*

Proof. Direct the cycle c clockwise. It is clear that every leftward horizontal edge has at least one corresponding rightward horizontal edge, and so c has at least two horizontal edges. Similarly, c has at least two vertical edges. If c has 4 edges then they must alternate. □

2.2 Density

From Lemma 1 we can deduce that if $G = (V, E)$ is a rectilinear graph with $n = |V|$ and $m = |E|$, then $m \leq 2n$. We can show a tighter bound as follows.

Lemma 4. *If $G = (V, E)$ is a rectilinear graph with $n = |V|$ and $m = |E|$, then $m \leq 2n - 2\sqrt{n}$. Further, for every n there is a rectilinear graph with n vertices and $2n - 2\lceil\sqrt{n}\rceil$ edges.*

Proof. Suppose that X_i is the set of vertices with x coordinate i; note that the induced subgraph on X_i is a set of paths and thus has at most $X_i - 1$ edges. Summing over all the sets X_i shows that the number of horizontal edges is at most $n-k$, where k is the number of such sets X_i, that is, the number of different x-coordinates of vertices. Similarly, if ℓ is the number of different y-coordinates of vertices, then the number of vertical edges is at most $n - \ell$. Thus

$$m \leq 2n - k - \ell. \tag{1}$$

Let x^* denote $\max_i |X_i|$. Now $\sum_i |X_i| = n$, and so $k \geq n/x^*$. Also, $\ell \geq x^*$, since no two vertices are at the same location; it follows that $k \geq n/\ell$. We can deduce from (1) that $m \leq 2n - (\ell + n/\ell)$. Minimizing the term $\ell + n/\ell$ over $1 \leq \ell \leq n$ gives the upper bound in the Lemma.

We can obtain the lower bound from grid graph of dimensions $\lceil\sqrt{n}\rceil \times \lceil\sqrt{n}\rceil$; this has at least n vertices and $2n - 2\lceil\sqrt{n}\rceil$ edges. □

2.3 Area

Area bounds for *planar* rectilinear drawings are have a long history (see, for example, [7]). It is straightforward to show that any rectilinear graph with n vertices has a rectilinear drawing on an $n \times n$ grid, and for planar graphs there is a corresponding lower bound. It could be tempting to suggest that allowing edge crossings allows smaller area rectilinear drawings. However, there is a graph on n vertices for which the minimum area grid drawing (planar or not) has area $\Omega(n^2)$; this is illustrated in Figure 1.

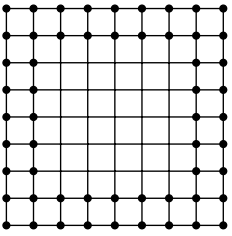

Fig. 1. The area of the rectilinear drawing is $\Omega(n^2)$

3 Graphs with a Connected-4-Cycle Cover

In this section we prove that rectilinearity can be tested in linear time is the graph has a connected-4-cycle cover, that is, it consists of a single 4-cycle block.

Theorem 1. *There is a linear time algorithm that tests whether a graph with a connected-4-cycle cover has a rectilinear drawing, and gives the drawing if it exists.*

The first step is check the necessary conditions of Lemma 1; if any of these conditions fails to hold, then we reject the input graph G. These conditions can be checked using a simple search in the set of vertices at distance at most 2 from each vertex; since the degree is bounded this takes linear time.

Next we partition the edge set E into E_{hor} and E_{vert}. This can be done with a depth-first traversal of the 4-cycle incidence graph G_4. At each step, we label the edges of a 4-cycle c as horizontal or vertical so that the labels alternate around c; we reject G if any inconsistency in labels is found. Since G_4 is connected, this labels every edge; also, the traversal takes linear time. Further note that the labeling is unique after the choice of labels on the first 4-cycle.

Next we check that the partition satisfies the conditions of Lemma 2. This is straightforward. Subsequently we assume that G_{hor} consists of paths $\{p_0, p_1, \ldots, p_{k-1}\}$ and G_{vert} consists of paths $\{q_0, q_1, \ldots, q_{\ell-1}\}$.

The next step is to assign a direction for each of the nontrivial paths in G_{hor}. Looking ahead, this direction induces a partial order on V; another partial order can be obtained from the direction of nontrivial paths in G_{vert}. By topologically sorting on each partial order, we can obtain $x-$ and y-coordinates for each vertex. However, we must be careful how the directions are assigned. Consider Figure 2. If the paths are directed as in Figure 2(a), then it is easy to see that the x-coordinates can be assigned by a topological sort. However, if a path happens to be in the wrong direction, as in Figure 2(b), then it is difficult to see how to assign x-coordinates. So we must choose a direction for each path.

We begin by assigning an arbitrary direction to each p_i; this gives an initial partial order on V. Let $\bar{p}_i$ denote the reverse of the path p_i. If there are 4 vertices a, b, c, d where $a, b \in p_i$, $c, d \in p_j$ such that $a, c \in q_s$ and $b, d \in q_t$ for some s, t, then p_i and p_j have a *compatibility* relationship, defined as follows. If $a < b$ and

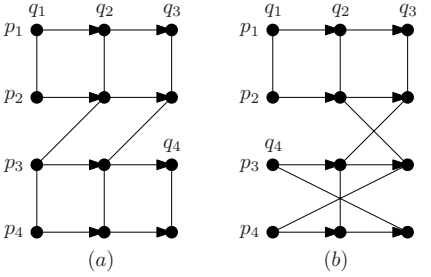

Fig. 2. The directed paths

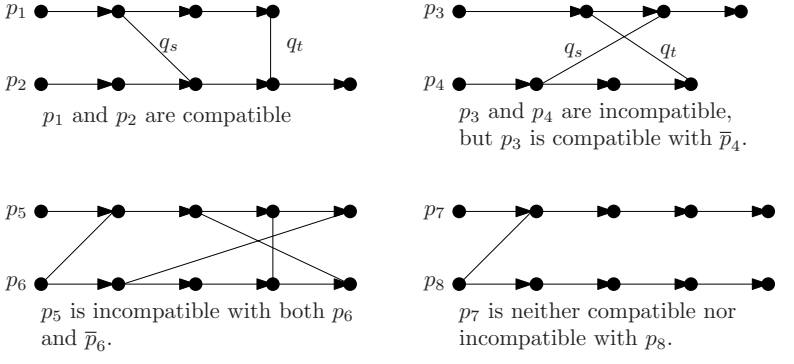

Fig. 3. Path direction examples

$c > d$ then we say that p_i is *incompatible* with p_j; if $a < b$ and $c < d$ then we say that p_i is *compatible* with p_j. In Figure 3, p_1 is compatible with p_2, and p_3 is incompatible with p_4; note, however, that p_3 is compatible with $\bar{p}_4$. The relationship of compatibility does not apply to the pair p_7, p_8, because these two paths share less than two paths in Q. Note that p_5 and p_6 are incompatible, further, p_5 and $\bar{p}_6$ are incompatible.

If there are two incompatible paths p_i and p_j, for which p_i and $\bar{p}_j$ are incompatible, then we reject G. Subsequently we assume that there are no such pairs of paths. In other words, we are assuming that for each pair p_i, p_j, if the compatibility relationship is defined then p_i is compatible with either p_j or $\bar{p}_j$.

Then we associate a boolean formula f with G as follows. For each path p_i we have a boolean variable x_i. If p_i is compatible with p_j then we insert the clause $(x_i = x_j)$ into f; if p_i is compatible with $\bar{p}_j$ then we insert the clause $(x_i \neq x_j)$ into f. Now $(x_i = x_j) = (x_i \vee \bar{x_j}) \wedge (\bar{x_i} \vee x_j)$, and $(x_i \neq x_j) = (x_i \vee x_j) \wedge (\bar{x_i} \vee \bar{x_j})$, and thus f is a 2SAT formula. Satisfiability for this formula can be tested in linear time. If f is not satisfiable, then we reject G; otherwise, a satisfying set of values for the x_i gives a direction for each path in G_{hor} so that every pair of paths is compatible.

Similarly, we can obtain a direction for each of the paths $q_1, q_2, \ldots, q_\ell$ of G_{vert} such that each pair q_i, q_j is compatible (if there is no such direction, then we reject G).

Lemma 5. *Suppose that G satisfies the conditions of Lemmas 1, 2, and 3. Further suppose that each pair of p_i, p_j of paths in G_{hor} is compatible, and each pair q_i, q_j of paths in G_{vert} is compatible. Then G has a rectilinear representation.*

Proof. We define a partial order on the paths of G_{vert} as follows. Suppose that p_i is a path in G_{hor}. Recall that every vertex in p_i is in exactly one path of G_{vert}. Suppose that $e = (u, v)$ is a (directed) edge of p_i, where $u \in q_j$ and $v \in q_{j'}$ for two paths q_j and $q_{j'}$ of G_{vert}; then we define $q_j < q_{j'}$. Since pairs of paths in G_{hor} are compatible, this relation is a partial order. Thus using a topological sort we can assign an x-coordinate to each path q_j in G_{vert}, and thus to each vertex in G (recall that every vertex of G is in exactly one path of G_{vert}). Similarly, one can assign a y-coordinate to each vertex in G. The resulting drawing is rectilinear. □

This completes the proof of Theorem 1.

4 Hardness Results

4.1 Graphs of 4-Cycle Blocks Connected by Edges

The previous section gives a linear time algorithm for testing rectilinearity when the graph consists of a single 4-cycle block. In this section, we show that a slight relaxation of this condition leads to NP-completeness.

Theorem 2. *The decision problem whether a graph consisting of a set of 4-cycle blocks connected by single edges has a rectilinear drawing is NP-complete.*

Proof. First we show that the problem is in NP. Note that a rectilinear graph has a rectilinear drawing on an $n \times n$ grid. Thus we can guess a location for each vertex on an $n \times n$ grid, and then check to see whether it is a rectilinear drawing.

We reduce from the 3SAT problem. The input instance for the problem is a set $\{x_1, x_2, \ldots, x_n\}$ of n variables, and a collection $\{c_1, c_2, \ldots, c_m\}$ of m clauses, where each clause consists of exactly three literals. The 3SAT problem is to determine whether there exists a truth assignment to the variables so that each clause has at least one true literal. In the following, we will describe our polynomial-time reduction.

First we construct a skeleton which contains ports connecting to the variable towers and the clause gadgets. The main component of the skeleton consists of a series of 4-cycles connecting together to form an L-shaped backbone. See Figure 4 for the illustration of its construction.

The upward spikes are ports connecting to the variable towers, and the 4-cycles hanging on the right hand side of the skeleton are ports connecting to the clause gadgets. The variable tower for variable x_i is constructed as shown in

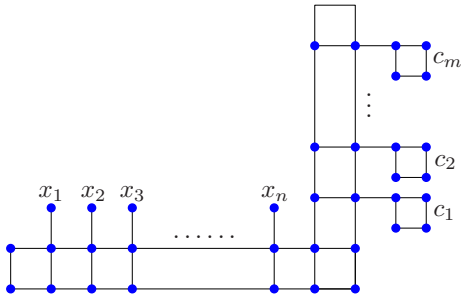

Fig. 4. Skeleton containing ports for connecting to variable towers and clause gadgets

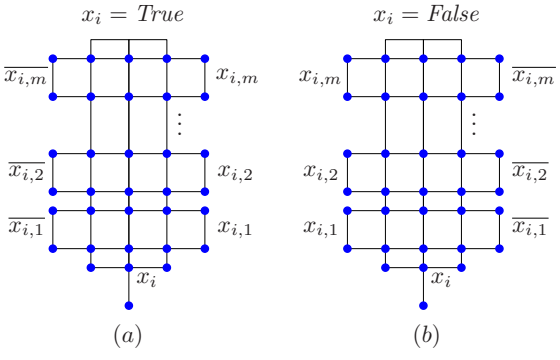

Fig. 5. Variable tower for x_i and the representation of its truth values

Figure 5(a), where we create a row of four consecutive 4-cycles R_j for each clause c_j and we set the two rightmost (resp. leftmost) vertices as connection ports for literal $x_{i,j}$ (resp. $\overline{x_{i,j}}$), where $x_{i,j}$ takes the same truth value as x_i. Moreover, we connect these m rows of consecutive 4-cycles by two adjacent columns of consecutive 4-cycles. See Figure 5(a).

This completes the construction of the variable tower for variable x_i.

We then connect this variable tower to the corresponding upward spike. The tower in Figure 5(a) represents the assignment of a *true* value to variable x_i, and its mirror image in Figure 5(b) represents the assignment of a *false* value to the variable x_i.

We then proceed to see how we construct the clause gadgets. Suppose that we want to construct the gadget for clause $c_j = x_i \vee \overline{x_k} \vee x_l$. First, let s_1^j, s_2^j be the two vertices on the 4-cycle σ_j at the connection port for c_j and adjacent to the connection edge between cycle σ_j and the skeleton. See Figure 6(a).

Note that the order of s_1^j and s_2^j are not essential since they are interchangeable. Now we connect vertex s_1^j to the upper ports of the two literals $x_{i,j}$ and $\overline{x_{k,j}}$, respectively, by a two-bend path. Moreover, at each of the two bending vertices of this connection path, we attach a 4-cycle. Then we perform the

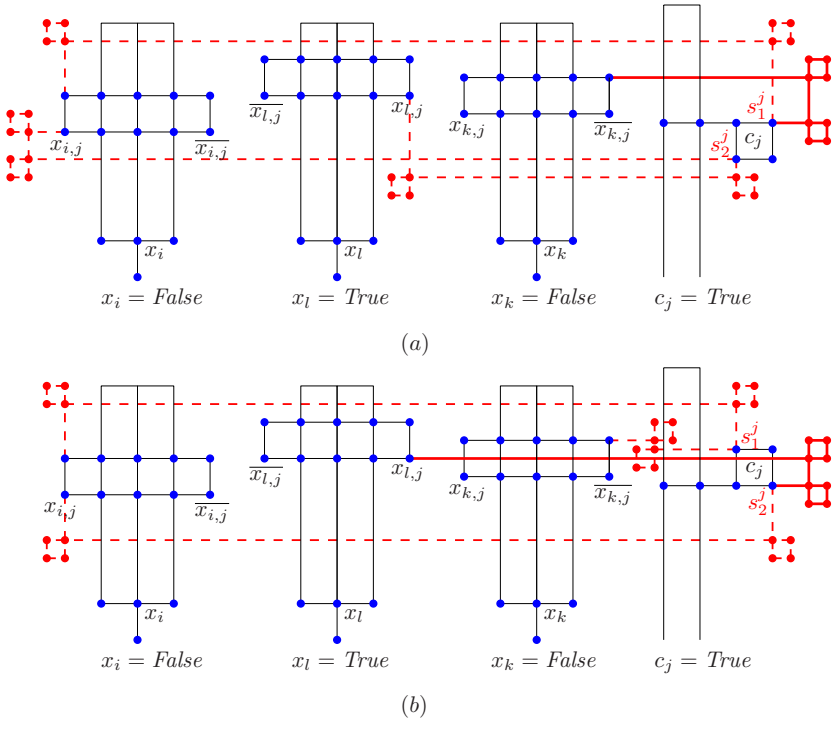

Fig. 6. Clause gadget for clause $c_j = x_i \vee \overline{x_k} \vee x_l$

similar construction by connecting vertex s_2^j to the lower ports of the two literals $x_{i,j}$ and $x_{l,j}$. This completes the construction of the gadget for clause c_j. We construct all clauses in the same fashion, and we finish the construction step. Clearly, our construction takes $O(mn)$ time. We let G be the graph we constructed. Below we proceed to show that the given 3SAT formula is satisfiable if and only if graph G has a rectilinear drawing.

We first consider the forward direction. Suppose the given 3SAT formula is satisfiable with some truth assignment. According to this truth assignment, we determine the embedding positions for the n variable towers according to the scenario in Figure 5. Then we will proceed to embed the clause gadgets one by one. Suppose that we are embedding the gadget for clause $c_j = x_i \vee \overline{x_k} \vee x_l$. As an example, assume that $\overline{x_k}$ is the true literal causing clause c_j to become true. As $\overline{x_k}$ is connected to s_1^j in our construction, we will rotate the 4-cycle σ_j at the connection port of clause c_j so that s_1^j is the rightmost vertex on cycle σ_j. See Figure 6(a). And we can connect the rightward port of literal $\overline{x_{k,j}}$ to the rightward port of s_1^j using the two-bend path between $\overline{x_k}$ and s_1^j. Other connection paths for this gadget can also be embedded accordingly. Figure 6(b) shows another example of embedding the gadget for clause c_j at the time that x_l is the true literal for c_j. Thus graph G has a rectilinear drawing.

We then consider the backward direction. Suppose G has a rectilinear drawing. Consider the embedded scenario of the gadget for clause $c_j = x_i \vee \overline{x_k} \vee x_l$. Without loss of generality, suppose that s_1^j is a rightmost vertex of 4-cycle σ_j. See Figure 6(a). As each connection path consists of two bends, the rightward port of s_1^j must connect to some rightward port of some variable tower. As such, if we set the truth values for the variables according to the embedding positions of their towers, all clauses will result in true values. Thus the given 3SAT formula is satisfiable. This finishes our NP-completeness proof. □

4.2 Graphs with Degree-2 and Degree-4 Vertices

With our result on graphs with connected-4-cycle cover above, at the first glance, it is tempting to claim that the rectilinear drawability of a graph with only degree-2 and degree-4 vertices may be polynomial-time solvable. However, by slight modification (see below) of the NP-hardness proof in [9], we obtain its NP-completeness proof for this problem. The vertices with degree one or three can be eliminated by connecting pairs of them by short paths. Finally, only degree-2 and degree-4 vertices remain. We conclude in the following theorem.

Theorem 3. *Let G be a graph with only degree-2 and degree-4 vertices. The decision problem whether G has a rectilinear drawing is NP-complete.*

5 Fixed-Parameter Algorithm

In this section, we show that the rectilinear drawing problem is fixed-parameter tractable, where the parameter is the number of degree-3 and degree-4 vertices in the graph.

Theorem 4. *Let G_k be a degree-4 graph with k vertices of degree 3 or 4 for some constant k. The decision problem whether G_k has a rectilinear drawing can be answered in linear time, or more precisely, in $O(24^k \cdot k^{2k} \cdot n)$ time.*

Proof. Let K be the set of these k degree-3 or degree-4 vertices. The vertices in K can be distributed among k vertical columns. Considering these columns as boxes, we can have at most k^k different ways to put these vertices into the boxes. On the other hand, the vertices in K also can be distributed among k horizontal rows, and there are also at most k^k different ways to put these vertices into the corresponding rows. Hence, there are essentially at most k^{2k} ways to embed the vertices in K on the plane.

Now suppose that K_ϵ is one such embedding of K in the plane. Each vertex v has degree at most 4 and there are only 4 directions in which edges incident to v can have; thus there are at most $4! = 24$ possible choices of directions for the edges incident to v. Therefore there are at most 24^k ways to select the directions of all edges incident to the vertices of K_ϵ.

We can check the embeddability of the whole graph for each selection of edge directions at each vertex of K. Suppose that we have selected the direction for all

edges incident to the vertices in K_ϵ. We proceed to embed all paths connecting between vertices in K_ϵ. For instance, suppose that we are going to embed the path $\eta_{u,v}$ between vertices u and v in K_ϵ, and we let e_u, e_v be the edges of path $\eta_{u,v}$ incident to u, v, respectively. We have known the directions for e_u and e_v, and we are going to decide the directions of other edges on path $\eta_{u,v}$. Note that routing of such paths can be done independently from each other, because planarity is not required. The feasibility of routing path $\eta_{u,v}$ from vertex u through e_u to vertex v through e_v depends on the length of $\eta_{u,v}$, and can be decided in time proportional to the length of $\eta_{u,v}$. Thus testing the feasibility of routing all paths in the graph G_k takes $O(n)$ time. In all, the decision problem whether G_k has a rectilinear drawing can be answered in $O(24^k \cdot k^{2k} \cdot n)$ time. □

6 Conclusion

We consider the problem of general non-planar rectilinear drawing for restricted class of graphs and obtain a polynomial time algorithm and NP-hardness results. The boundary between NP-completeness and polynomial time is determined by the structure of 4-cycle blocks in the graph. We also present tight bound on the drawing area and an FPT algorithm for the parameter of the number of vertices of degree ≥ 3. It is feasible that there is an FPT algorithm where the exponential complexity depends on the number of 4-cycle blocks in the graph; this is left as an open problem.

There are two directions that need to be explored before commercial visualization software can make use of this work. Firstly, the algorithm in Section 3 is restricted graphs with connected 4-cycle covers. This is a relatively restrictive condition, and it would be wise to investigate whether there are polynomial-time algorithms for some wider classes. Secondly, many graphs do not have a rectilinear drawing; it would be interesting to investigate methods that find large subgraphs with rectilinear drawings (see, for example, [3]).

Finally, we note that Eppstein [8] studied bendless orthogonal drawing problem in three dimensions, and showed that it is NP-complete to determine whether an arbitrary graph has such an embedding. It would be interesting to extend the concept of 4-cycle block to three dimensions and determine the boundary between polynomial time algorithms and NP-completeness in three dimensions.

References

1. Aspvall, B., Plass, M., Tarjan, R.: A linear-time algorithm for testing the truth of certain quantified boolean formulas. Information Processing Letters 8(3), 121–123 (1979)
2. Batini, C., Talamo, M., Tamassia, R.: Computer aided layout of entity relationship diagrams. Journal of Systems and Software 4(2-3), 163–173 (1984)
3. Biedl, T., Madden, B., Tollis, I.: The three-phase method: a unified approach to orthogonal graph drawing. Int. J. Comput. Geometry Appl. 10(6), 553–580 (2000)
4. Birkhoff, G.: Rectilinear drawing, in Three public lectures on scientific subjects. Three public lectures on scientific subjects 28(1), 51–76 (1941)

5. Bodlaender, H.L., Tel, G.: A note on rectilinearity and angular resolution. Journal of Graph Algorithms and Applications 8, 89–94 (2004)
6. Didimo, W., Eades, P., Liotta, G.: Drawing graphs with right angle crossings. In: Dehne, F., et al. (eds.) WADS 2009. LNCS, vol. 5664, pp. 206–217. Springer, Heidelberg (2009)
7. Dolev, D., Leighton, F.T., Trickey, H.: Planar embedding of planar graphs. Advances in Computing Research 2, 147–161 (1984)
8. Eppstein, D.: The topology of bendless three-dimensional orthogonal graph drawing. In: Tollis, I.G., Patrignani, M. (eds.) GD 2008. LNCS, vol. 5417, pp. 78–89. Springer, Heidelberg (2008)
9. Formann, M., Hagerup, T., Haralambides, J., Kaufmann, M., Leighton, F.T., Symvonis, A., Welzl, E., Woeginger, G.J.: Drawing graphs in the plane with high resolution. In: Proceedings of FOCS, pp. 86–95. IEEE, Los Alamitos (1990); SIAM Journal on Computing 22(5), 1035–1052 (1993)
10. Garg, A., Tamassia, R.: On the computational complexity of upward and rectilinear planarity testing. SIAM Journal on Computing 31(2), 601–625 (2002)
11. Hoffman, F., Kriegel, K.: Embedding rectilinear graphs in linear time. Information Processing Letters 29(2), 75–79 (1988)
12. Huang, W., Hong, S., Eades, P.: Effects of crossing angles. In: Proceedings of PacificVis 2008, pp. 41–46. IEEE, Los Alamitos (2008)
13. Rahman, M. S., Naznin, M., Nishizeki, T.: Orthogonal drawings of plane graphs without bends. In: Mutzel, P., Jünger, M., Leipert, S. (eds.) GD 2001. LNCS, vol. 2265, pp. 392–406. Springer, Heidelberg (2002)
14. Rahman, M.S., Egi, N., Nishizeki, T.: No-bend orthogonal drawings of series-parallel graphs. In: Liotta, G. (ed.) GD 2003. LNCS, vol. 2912, pp. 387–392. Springer, Heidelberg (2004)
15. Rahman, M.S., Egi, N., Nishizeki, T.: No-bend orthogonal drawings of series-parallel graphs. In: Healy, P., Nikolov, N.S. (eds.) GD 2005. LNCS, vol. 3843, pp. 409–420. Springer, Heidelberg (2006)
16. Vijayan, G., Wigderson, A.: Rectilinear graphs and their embeddings. SIAM Journal on Computing 14(2), 355–372 (1985)