Massimo Ancona, Walter Cazzola, Sara Drago, and Gianluca Quercini. Visualizing and Managing Network Topologies via Rectangular Dualization. In Proceedings of the IEEE Symposium on Computers and Communications (ISCC'06)*, pages 1000–1005, Pula, Sardinia, Italy, June 2006.*

Visualizing and Managing Network Topologies via Rectangular Dualization

Massimo Ancona* Walter Cazzola† Sara Drago* Gianluca Quercini*

* Università degli Studi di Genova
Dipartimento di Informatica e Scienze dell'Informazione
via Dodecaneso 35, 16146 Genova, Italy
{ancona,drago,quercini}@disi.unige.it

† Università degli Studi di Milano
Dipartimento di Informatica e Comunicazione
via Comelico 39/41, 20135 Milano, Italy
cazzola@dico.unimi.it

Abstract

Rectangular dualization is an effective, hierarchically oriented visualization method for network topologies and can be used in many other problems having in common with networks the condition that objects and their interoccurring relations are represented by means of a planar graph. However, only 4-connected triangulated planar graphs admit a rectangular dual. In this paper we present a linear time algorithm to optimally construct a rectangular layout for a general class of graphs and we discuss a variety of application fields where this approach represents an helpful support for visualization tools.

1. Introduction

The reengineering of a large communication network is a complex problem, which consists of different aspects that can be strongly affected by the way of describing data. The plainest way to describe a communication network is to model the relationship among sites and links by means of a weighted undirected graph. An issue is encountered when network analysis and visualization have to be realized: practical networks include hundreds or often thousands of nodes and links, so that even a simple description and documentation of their structure is hard to maintain and update. In this case the network is usually described in form of a hierarchy of sub-networks that are represented by collapsing some sub-networks into single nodes or single links to be described in separate documents. Such a hierarchical approach to network (and graph) description can be formalized into a complex but flexible graph structure called *structured graph* [8]. This representation is useful not only for documenting and drawing a large network but also for making computations on it without reconstructing the whole graph plan every time some computation has to be performed on it.

Rectangular dualization was originally introduced to generate rectangular topologies for floorplanning of integrated circuits: by a floorplan, we partition a rectangular chip area into rectilinear polygons corresponding to the relative location of functional entities of the circuit. In spite of the specialized problems that motivated its origin, rectangular dualization contributes to the resolution of many other visualization problems, such as network configuration issues, when human interventions of design or topology adjustment are needed and a physical or logical layout representation becomes essential for the human operator. In fact, a very serious problem to cope with in graph drawing is how to represent edges in such a way that they do not appear too close together. The aim is to enhance the readability of the drawing, making easier to find out which nodes are connected by an edge. The very first solution to this problem is to avoid edge crossings, and this motivates the interest for *planar graphs*, that are precisely those graphs that can be drawn in the plane with no edge crossings. The choice for planar graphs is not only a representation facility but is primarily validated by real-world examples where the presence of crossing links may produce technical drawbacks. Further on, since a major optical effort is encountered in the proximity of vertices, where adjacent edges need to meet in a point, several studies have been spent in devising drawing algorithms capable of maximizing *angular resolution*, i.e.

the smallest angle between adjacent edges, in such a way that lines representing connections are kept as separate as possible. Orthogonal graph drawing solves the angular resolution issue by forcing all angles between adjacent edges to be $\frac{\pi}{2}$ and in drawing edges as polylines whose constituting segments are axis-parallel. A further important issue is the layout space compaction. In fact, minimizing the drawing area is not only an aesthetics concern: a representation where nodes and edges fill the space homogeneously is not only more pleasant and readable, but also more adaptive to the requirements of different visualization media. In the particular case of orthogonal graph drawing, techniques to optimally find a grid embedding are matter of interest. Attention has recently been paid to the dynamic drawing of hierarchically structured graphs [12] or the possibility of incrementing the layout visualization at subsequent steps of the graph visit; since many visualization supports such as screens, windows or paper sheets have a rectangular topology, rectangular dualization, with its multiscale capability of containing nested rectangles, has a predisposition for representing structured graphs and hierarchically organized networks.

In the rest of the paper, we will give an overview of the state of the art (section 3), we will show our algorithm to build a rectangular dualization in linear time (section 4) and draw a network (section 5). Finally we will show some other applications of the rectangular dualization (section 6), examine some related works (section 2) and draw our conclusions

2 Related Works

Thomassen proved that every planar graph is the intersection graph of a collection of three-dimensional boxes, with intersections occurring only in the boundaries of the boxes. Furthermore, he characterized the graphs that have such representations (called strict representations) in the plane. These are precisely the proper sub-graphs of 4-connected planar triangulations. Together with earlier work [21], his work yields an algorithm for testing a graph G to see if it admits a rectangular dual and, if so, constructing such a representation. His proof does not look to lead to an efficient algorithm, however: a straightforward implementation of his method requires at least $O(n^3)$ time and he does not bound the layout area. Bhasker and Sahni [5] developed linear-time algorithms to find a rectangular dual for 4-connected planar triangulation. In [14] Kant and He explain how to construct a rectangular dual from a *regular edge labeling* (REL, for short) and present two algorithms to compute such labeling, one based on an edge contraction technique and the other on a *canonical ordering*. A later work of Saidur presents a linear time algorithm which finds a rectangular grid drawing using a depth first search [17]. In [9] it is shown how the works above may be combined to produce an efficient algorithm for constructing rectangular duals with asymptotically bounded area. A large scale application of the aforecited linear methods is strongly limited by the fact that not all graphs admit a rectangular dual, especially those containing separating triangles (see section 3 for details). Lai and Leinwand [16] first presented the idea of forcing rectangular dual admissibility by introducing crossover vertices breaking all separating triangles. They conjectured that finding a minimal set of crossover vertices was a *NP*-complete problem and performed non optimal introduction of crossover vertices in linear time. Our method performs an optimal introduction of crossover vertices and extends the class of graphs treatable by the rectangular dualization linear methods discussed in the first part of this section, such as the one of [14] (see the schema of fig 3 for a visual comparison between our method and existing approaches).

3 Definitions

In this section we present the graph-theory terminology necessary for the comprehension of the rest of the paper. A *rectangular dual* of a planar graph $G = (V, E)$ is a rectangle R with a partition of R into a set $\Gamma = R_1, \ldots, R_n$ of non overlapping rectangles such that:

- no four rectangles meet at the same point;
- there is a one-to-one correspondence $f : V \longrightarrow \Gamma$ such that two vertices u and v are adjacent in G if and only if their corresponding rectangles $f(u)$ and $f(v)$ share a common boundary.

It is easy to see that if a graph admits a rectangular dual, it may not be unique (see fig 1).

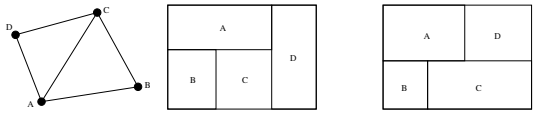


Figure 1. A planar triangulated graph and two possible rectangular layouts.

On the other side, some graphs do not admit rectangular dual. The most important point for the existence of a rectangular dual is the absence of *separating triangles*, (that is, 3-vertex cycles with at least one vertex in their interior) a condition that in planar triangulations is equivalent to *4-connectivity* whose meaning is that the removal of any set of 3 vertices leaves the remainder of G connected [14]. One of the most critical point of our algorithm is the separating

triangles removal. This involves a well-known concept in graph theory: the edge covering problem. An edge covering for a graph G is a set of edges C so that every vertex of G is adjacent to at least one edge in C. We solve this problem by reducing it to a matching problem. A *matching* in G is a subset M of edges such that for every vertex v, at most one edge e covers v, that is v is an endvertex of e. In our procedure the edges of the graph are weighted and we compute a maximum weighted matching, that is a matching where the sum of the matched edges is maximum with respect to the other possible matchings. Matching is a very challenging problem in general graphs. Fortunately we have to solve it only for the 3-regular bridgeless graphs. A graph is *k-regular* if every vertex has degree k, that is k incident edges. A *bridge* is an edge whose removal disconnects G. If the graph has not bridges, it is called *bridgeless*. Whenever we speak of a planar graph, we assume that some planar embedding has been fixed, which corresponds to the idea of depicting an existent physical network (in this perspective, it would be more accurate to speak of *plane graphs*, i.e. planar graphs with a fixed embedding in the plane).

In the following section we give an $O(n)$ time algorithm that transforms planar graphs that do not admit rectangular duals into graphs admitting one by adding the minimum number of new vertices.

4 All Separating Triangles in a Plane Graph can be Optimally Broken in Linear Time

In the practice of network representation not all the cases of interest satisfy 4-connectivity, but it is always possible to turn a planar input graph into a 4-connected planar triangular graph by breaking all the separating triangles. In [1], Ancona et al. showed that all separating triangles can be optimally broken in polynomial time and presented an asymptotical bound of $O(n^3)$, which some discussion can refine up to $O(mn \lg n)$. This method performs this transformation in three steps:

- ❶ the geometrical dual of a biconnected graph is computed and faces belonging to a separating triangle are detected and clustered in a single macro-vertex;
- ❷ a covering affecting macro-vertices is computed; the effect is that all separating triangles are optimally broken by inserting new vertices in some strategic places along some of their constituting edges;
- ❸ the resulting graph is triangulated with the algorithm described in [7].

We have so far obtained a graph satisfying triangularity and four-connectivity. In the modification perspective, the input graph is not even required to satisfy biconnection: there is a preliminary method to provide this degree of connectivity [19]. The complexity of the whole method is affected by the largest complexity among its intermediate steps, that is due to the deletion of separating triangles. We have to break all the separating triangles by adding a minimum number of crossover vertices. To this aim, we assign to each edge e of the geometrical dual graph a weight, which is the number of separating triangles shared by the dual edge of e. Then we compute a minimum weighted macro-covering, obtaining the edges on which crossover vertices are to be added. The minimum weighted covering can be computed by resolving a sequence of minimum *weighted* covering problems on each simple graph of the structured dual. In [18], Parekh showed how to reduce a minimum weighted edge cover of a specified subset of the vertices of G to a maximum weighted b-matching, a well solved problem [10] that is tackled by implementing the $O(mn \lg n)$ algorithm presented in [13]. Breaking separating triangles has some computational cost, but the proposed *divide and conquer* approach due to the macro-covering step makes the method effectively usable on very large graphs, as resulted from the implementation of the method [4]. However, the main reason of such a cost is that the aforecited matching algorithm holds for general graphs, a much wider class of graphs than the one we deal with by assuming planar networks. Instead, a matching in a 3-regular bridgeless graph can be found in linear time [6]. Since the collection of all planar 3-regular bridgeless graphs is exactly the collection of duals of planar triangulations where the outside face is a triangle, we may tighten the bound by solving the matching problem on the dual of a planar triangulation and we obtain this by anticipating the application of the algorithm [7], which triangulates without adding new separating triangles.

5 From the Rectangular Dual of a Graph to its Orthogonal Drawing

The graph resulting from the sequence of steps described in the previous section satisfies the condition for the algorithm of visibility representation based on canonical ordering [14]. Once a visibility representation has been computed, an orthogonal embedding can be constructed by applying Tamassia and Tollis algorithms [20]. Alternatively, we can produce an orthogonal drawing directly from a *rectangular dual* representation of the input graph. Figure 2 is an example of automatically created orthogonal drawing, shaded rectangles correspond to breaking points completing the graph up to 4-connectivity.

In both orthogonal grid embedding methods, augmenting vertices introduced by the separating triangles breaking method and augmenting edges introduced by the triangulation step need to be removed from the final orthogonal grid embedding. The orthogonal drawing definition we pre-

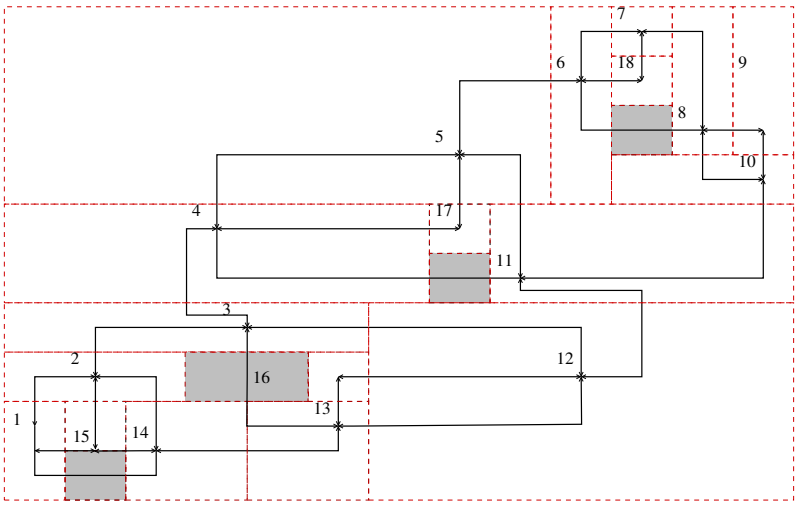


Figure 2. An orthogonal drawing derived from a rectangular dual representation (dashed lines, to be removed in the final drawing). Shaded rectangles are due to breaking points.

viously referred to has practical importance only for graphs with maximum degree 4; when considering graphs of higher degree, it is not any longer possible to draw vertices as points while maintaining an angular resolution of $\frac{\pi}{2}$ in every meeting point. In [15] are discussed a number of accredited solutions (such as to draw vertices as boxes with a sufficient number of grid lines for adjacent edges) to overcome this issue. In our approach, we weakened the orthogonal drawing definition by accepting a different angular resolution around vertices and found that, for medium vertex degree cases, this relaxation does not compromise readability (see Fig. 4(a)). An alternative approach consists in limiting the number of drawn lines by grouping several edges sharing the same direction in hyper-edges and by labeling merging points with the target name (Fig. 4(b)). Whatever orthogonal embedding technique is adopted, a final optimization step is needed to reduce the number of bends; this is obtained by using the bend-stretching transformations introduced in [20].

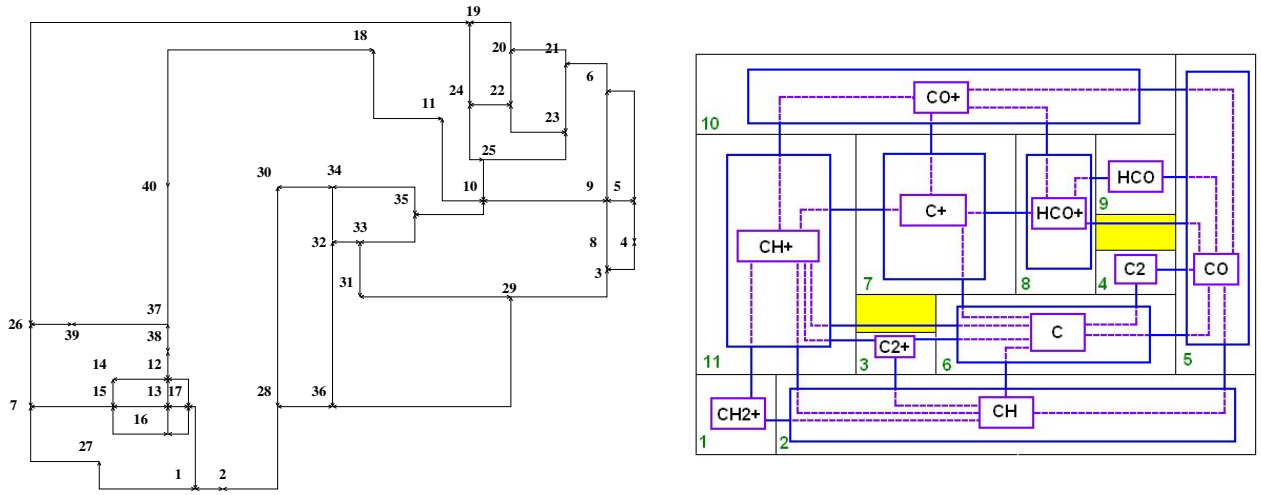
6 Put Rectangular Dualizations at Work

Using a rectangular dual for graph drawing offers several advantages:

- its complexity is linear ($O(n)$ time, where n is the number of nodes),
- the class of tractable graphs can be widened without penalizing the layout area of the best cases; in the worst case, the area of the layout is $O(n^2)$, like the other most efficient drawing methods,
- it provides symmetrical drawing with respect to x and y coordinates,
- it leads to an alternative approach to orthogonal graph drawing, that has revealed to be usable in many real world examples (see fig. 5(a))
- to construct a 2-visibility drawing from a rectangular dual is immediate (see figure 5(b)),
- a 1-band rectilinear drawing can be obtained from a rectangular dual in linear time by easily transforming the 2-visibility algorithm [14] (figure 5(b)).
- hierarchical dualization (and drawing) is naturally embeddable into a hierarchy of rectangular dual graphs,

A 2-visibility drawing [14] is a drawing in which each vertex is represented by a rectangle and the adjacency relations are expressed by horizontal and vertical lines drawn between the rectangles. Figure 5(b) shows a 2-visibility drawing of the astro-graph of [11] obtained from the rectangular dual (outlined in red) of the astro-graph.

In figure 5(b) you can see the dual rectangle (the yellow dashed one) of a broken separating triangle and how the corresponding rectangle is used for drawing. In fact each separating triangle is broken by a new vertex that disappears in the final drawing: the corresponding rectangle is traversed by a single horizontal (or vertical) line representing the broken edge. Figure 5(b) shows also how the 2-visibility drawing can be transformed into a 1-bend rectilinear drawing of the same graph. The algorithm squeezes the rectangles



(a) A real case medium-sized example.

(b) 1-bend drawing of astro-graph and 2-visibility representation

Figure 5.

representing vertices until a 1-bend drawing becomes impossible. Here again the property of the rectangular dual guarantees the existence of the solution.

Structured graphs (also called clustered graphs [2,3]) are graphs with a recursive clustering structures over the vertices (or edges). Drawing clustered Graphs on an orthogonal grid has been performed by the output drawing has order $O(n^2)$ and the algorithm is $O(n)$. We obtain the same optimal figures by using the rectangular dual of the graph for drawing the graph. The construction of the structured dual of a structured graph can be constructed in two ways:

❶ recursively applying the construction to each graph of the hierarchy

❷ by forcing the rectangles representing a cluster in the plain graph (they must be adjacent) to form a single rectangle.

Attention has recently been paid to the dynamic drawing of hierarchically structured graphs [12] or the possibility of incrementing the layout visualization at subsequent steps of the graph visit; since many visualization supports such as screens, windows or paper sheets have a rectangular topology, rectangular dualization, with its multi-scale capability of containing nested rectangles, has a predisposition for representing structured graphs and hierarchically organized networks. In our case, as seen in the next section, the hierarchical structure is part of the network architecture (graph) in the sense it is used for optimizing communications and building the network itself. In this case, being our networks almost planar, a rectangular dual representation is an Euclidian view of the graph useful for drawing the network, or for its physical implementation.

7 Conclusions

We described a method showing how it is possible to optimally complete in linear time a generic planar graph up to 4-connectivity by adding crossover vertices in order to eliminate separating triangles. This method can be used to preprocess the input graph in order to apply the visibility representation algorithm based on canonical ordering. Such an approach has revealed to be usable in many real world examples and to be adaptive to practical use relaxations of the orthogonal drawing concept in case of graphs having a degree which is greater than 4. Future work is aimed at shaping up the rectangular dual capability of capturing nested rectangles to draw hierarchically organized graphs.

References

[1] A. Accornero, M. Ancona, and S. Varini. All Separating Triangles in a Plane Graph Can Be Optimally "Broken" in Polynomial Time. *International Journal of Foundations of Computer Science*, 11(3):405–421, 2000.

[2] M. Ancona, W. Cazzola, E. Martinuzzi, P. Raffo, and V. I. Bogdan. Clustering Algorithms for the Optimization of Communication Graphs. In *Proceedings of the Fourth Conference Italo-Latino American of Industrial and Applied Mathematics*, pages 328–334, Havana, Cuba, 2001.

[3] M. Ancona, W. Cazzola, P. Raffo, and V. I. Bogdan. Virtual Path Layout Design Via Network Clustering. In *Proceedings of International Conference Communications 2000*, pages 352–360, Bucharest, Romania, 2000. IEEE Computer Society Press.

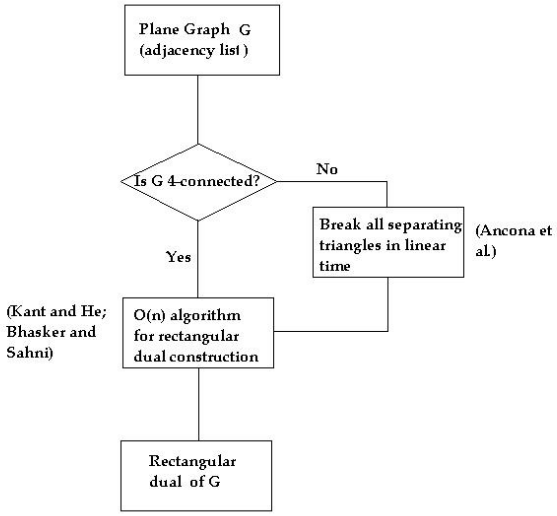
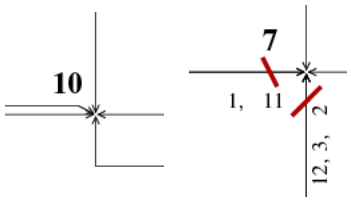


Figure 3. Schema of the method



(a) nonorthogonal (b) hyper-edges

Figure 4. To deal with higher in-degrees.

[4] M. Ancona, S. Drago, and P. A. Mazzarello. OCoRD: Optimal Constructor of a Rectangular Dual-Software Documentation. Internal report, DISI, Università degli Studi di Genova, Aug. 2004.

[5] J. Bhasker and S. Sahni. A Linear Algorithm to Find a Rectangular Dual of a Planar Triangulated Graph. *Algorithmica*, 3:247–278, 1988.

[6] T. C. Biedl, P. Bose, E. D. Erik D. Demaine, and A. Lubiw. Efficient Algorithms for Petersen's Matching Theorem. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 130–139, N.Y., Jan. 1999. ACM-SIAM.

[7] T. C. Biedl, G. Kant, and M. Kaufmann. On Triangulating Planar Graphs Under the Four-Connectivity Constraint. *Algoritmica*, 19(4):427–446, Dec. 1997.

[8] E. Bruzzone, L. De Floriani, J. S. S. Deogun, M. Ancona, and K. S. Bagga. Structured Graph Models: An Efficient Tool for VLSI Desgin. In *Great Lakes Computer Science Conference*, pages 307–312, 1989.

[9] A. L. Buchsbaum, E. R. Gansner, C. M. Procopiuc, and S. Venkatasubramanian. Rectangular layouts and Contact Graphs. AT&T Technical Report TD-59JQx5.

[10] J. Edmonds and E. L. Johnson. Matching: A Well-Solved Class of Integer Linear Programs. In M. Jünger, G. Reinelt, and G. Rinaldi, editors, *Combinatorial Optimization: Eureka, You Shrink!, Papers Dedicated to Jack Edmonds*, LNCS 2570, pages 27–30. Springer, July 2003.

[11] U. Fößmeier, G. Kant, and M. Kaufmann. 2-Visibility Drawings of Planar Graphs. In S. North, editor, *Proceedings Graph Drawing*, pages 155–168, Berkeley, California, USA, 1997. Springer.

[12] Y. Frishman and A. Tal. Dynamic Drawing of Clustered Graphs. In *Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'04)*, pages 191–198, 2004.

[13] Z. Galil. Efficient Algorithms for Finding Maximum Matching in Graphs. *ACM Computing Surveys*, 18(1):23–38, Mar. 1986.

[14] G. Kant and X. He. Two Algorithms for Finding Rectangular Duals of Planar Graphs. In J. van Leeuwen, editor, *Proceedings of the 19th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'93)*, LNCS 790, pages 396–410, Utrecht, The Netherlands, June 1993. Springer.

[15] M. Kaufmann and D. Wagner, editors. *Drawing Graphs: Methods and Models*, volume 2025 of *Lecture Notes in Computer Science*. Springer, 2001.

[16] Y.-T. Lai and S. M. Leinwand. Algorithms for Floorplan Design Via Rectangular Dualization. *IEEE Transaction on Computer-Aided Design*, 7:1278–1289, 1988.

[17] R. Md. Saidur. Efficient Algorithms for Drawing Planar Graphs. Phd thesis, Department of System Information Science, Graduate School of Information Sciences, Tohoku Univesity, Japan, Jan. 1999.

[18] O. Parekh. Edge Dominating and Hypomatchable Sets. In *Proceedings of 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'02)*, pages 287–291, San Francisco, CA, USA, Jan. 2002. ACM/SIAM.

[19] R. C. Read. A New Method for Drawing a Graph given the Cyclic Order of the Edges at Each Vertex. *Congr. Numer.*, 56:31–44, 1987.

[20] R. Tamassia and I. G. Tollis. Planar Grid Embedding in Linear Time. *IEEE Transactions on Circuits and Systems*, 36(9):1230–1234, Sept. 1989.

[21] C. Thomassen. Plane Representations of Graphs. *Progress in Graph Theory*, pages 43–69, 1984.