Communications to SIMAI Congress,
ISSN 1827-9015, Vol. 1 (2006)

DOI: 10.1685/CSC06009

# Rectangular Dualization of biconnected plane graphs in linear time and related applications

**Massimo Ancona[1] Sara Drago[1] Gianluca Quercini[1]**

**[1] Dipartimento di Informatica e Scienze dell'Informazione (DISI), University of Genova, Via Dodecaneso 35, 1646, Genova, Italy {ancona, drago, quercini }@disi.unige.it**

## Introduction

Rectangular dualization is the computation of the rectangular dual of a planar graph. It was originally introduced to find rectangular topologies for floorplanning of integrated circuits: by a floorplan, a rectangular chip area is partitioned into rectilinear polygons corresponding to the relative location of functional entities of the circuit. Subsequently it found application in many other fields, in particular becoming effective in visualization problems. In this paper we briefly describe two possible applications: visualization of network topologies and design of 3D Virtual Worlds.
It is known that the engineering and the optimization of large communication networks is a very challenging problem, as often real networks are very huge, including hundreds and even thousands of nodes and links. In order to help human operators in maintaining and updating the description and documentation of the network structure, the network is described in form of a hierarchy of sub-networks. The rectangular dualization is a very useful technique in graph drawing, especially when applied to the hierarchical drawing of a structured graph.
The design of 3D Virtual Worlds is a completely different field. Virtual Worlds visualized in 3D are environments where people "meet". Such environments provide a consistent and immersive user interface that facilitates awareness of other participants. In other words, Virtual Worlds support to a certain extent the way humans operate and interact in the real world. In order to guarantee a good user experience, it is important to overcome navigation problems, that are strongly related to the human approach with the service interface, because if not well solved they may break the user immersive experience. The generation of the Virtual World is determined by the graph representing the map of the institution. Rectangular dualization of such a map helps to ease and quicken the navigation.
The use of rectangular dualization is strongly limited by the fact that not all planar graphs admit a rectangular dual. However it is possible to apply a minimal set of transformations to the original graph to obtain a graph that admits a rectangular dual representation.
In the following sections we first describe a linear-time algorithm which computes the rectangular dual of a planar graph, transforming graphs not satisfying the rectangular dual conditions; then we give an overview of the two applications mentioned above.

## 1. OCoRD: Optimal Constructor of a Rectangular Dual.

A rectangular dual of a planar graph G=(V,E) is a rectangle R partitioned into a set Γ=R1,....Rn of non overlapping rectangles such that:

- no four rectangles meet at the same point;
- there is a one-to-one correspondence $f$: V → Γ 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.

Graphs not admitting a rectangular dual contain separating triangles, where a separating triangle is a triangular region of the graph, that is not a face. The idea behind OcoRD, our tool aiming at constructing the rectangular dual of any planar graph in linear time, is to remove (we will use the term “break” from now on) all the separating triangles, if any, from the input graph, to obtain a graph admitting the rectangular dual. This can be accomplished by adding a crossover vertex on one edge of each separating triangle, as Lai and Leinwand proposed in [5]. However, their approach does not take care of adding a minimal set of crossover vertices, as they conjectured that was a NP-complete problem. In fact it is possible to break two or more separating triangles by adding only one crossover vertex on an edge shared by all of them, instead of adding a vertex for each separating triangle. In OCoRD this task is performed in five steps:

- four external vertices are added according to [4];
- the geometrical dual of the resulting graph is computed and separating triangles are detected;
- All the separating triangles are collapsed into macro-vertices;
- A weight is assigned to each edge e, corresponding to the number of separating triangles sharing the dual edge of e;
- A maximum weight matching is computed on this structured graph, obtaining the edges on which to add the crossover vertices.

At this point we have obtained a graph free from separating triangles and its rectangular dual can be computed in linear time with the algorithm described in [4]. In literature many other linear-time algorithms to build the rectangular dual of a graph not containing separating triangles can be found [2, 5].
We are developing a linear time version of OCORD. Until now all steps sketched above are implemented in linear time. The only step to be completed is the matching phase that, however, has a small impact on the execution time, given the small number of separating triangles generated in almost all practical cases.

## 2.0 Visualization of network topologies

The plainest way to describe a communication network is to model the relationship among sites and links by means of a weighted undirected graph, but unless some more assumptions are taken, this approach can raise several problems. Problems are encountered when networks have a hierarchical topology, i.e. nodes are classified in levels, usually two or three, according to their geographical position, number of users, and so on. Moreover, the optimization of large communication networks (for example in order to minimize the number of message hops, i.e. the number of graph nodes traversed by a message) requires the use of special methods which need a most suitable graph representation. Most of the adopted techniques are based on *clustering*, i.e., the network layout is configured into subnetworks. Usually, a *virtual graph* is often constructed, where each node represents a sub-graph of the original graph and edges represent relationship among these sets of nodes. The problem consists in finding an efficient method to organize the nodes of the network so that the number of hops performed by the signal is minimal. Many issues are encountered during network design phase, when repeated analysis and visualization of the network has to be performed: practical networks include hundreds or often thousands of nodes and links, so that even a simple description and documentation of the network 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*. In the following sections we stress the importance of the rectangular dualization in graph drawing.

### 2.1 The role of the rectangular dual graph in network drawing

Using a rectangular dual for graph drawing offers several advantages:

- Its computational complexity is optimal: O(n) time and O($n^2$) area, like other most efficient methods;
- It provides a symmetrical drawing with respect to ***x*** and ***y*** coordinates,
- It can be naturally extended to the hierarchical drawing of a structured graph G, by introducing the concept of *structured rectangular dual,* i.e., by imposing that the rectangular dual of each macro vertex H of G be encapsulated into the rectangle representing H in the rectangular dual of G in such a way that all the adjacency relations between each vertex of H and each vertex of G be reflected into the structured dual. A rectangular dual reflecting this property is called a *structured rectangular dual*.
- The construction of a 2-visibility drawing from a rectangular dual is immediate (see Figure 1),
- A 2-bend rectilinear drawing can be obtained from a rectangular dual in linear time.
- Also a 1-band rectilinear drawing can be obtained from a rectangular dual in linear time.

The 2-visibility drawing method has been introduced by Kant [3]. In a 2-visibility drawing the vertices of a given graph are represented by rectangles (rectangular boxes) and the adjacency relations are expressed by horizontal and vertical line drawn between boxes. The authors claim a high quality of the drawing. This kind of drawing can be trivially derived from the rectangular dual of a graph as follows.
Figure 1 shows the 2-visibility drawing obtained from the rectangular dual of the Astro graph. The figure on the left reports the rectangular dual with the required drawing embedded in it. The figure on the right displays the final drawing.

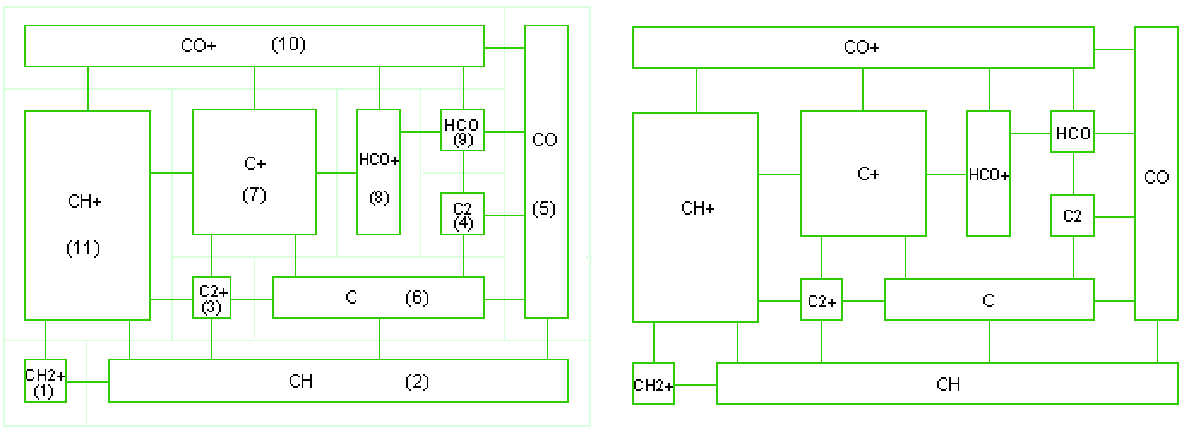

**Figure 1 - 2-Visibility drawing of the Astro Graph**

The approach is simple: two adjacent rectangles R1 and R2 of the dual graph share a portion of an edge that we call window. If inside R1 and R2 we draw two smaller rectangles (in green in Figure 1) enough large to be mutually visible through the window, they can be connected by a straight segment.

### 2.2 Two bends (and One bend) bus mode drawing

The bus drawing style (Figure 2) collect bundles of parallel edges into a single path until one edge of the bundle changes direction and emerges from the group. This drawing mode is useful for undirected graphs, since the arrowheads would introduce ambiguities. To create a bus style drawing, we simply have to ignore the offsets of the edges on the finer grid, and transform each bend into a curved corner. Despite its evident utility (see schematics drawing tools like Orcad etc.) this mode has not been considered by researchers in graph drawing. In this paper we use only bus-mode layout because it is particularly useful for connecting vertices of high degree, a frequent case in clustered graphs.

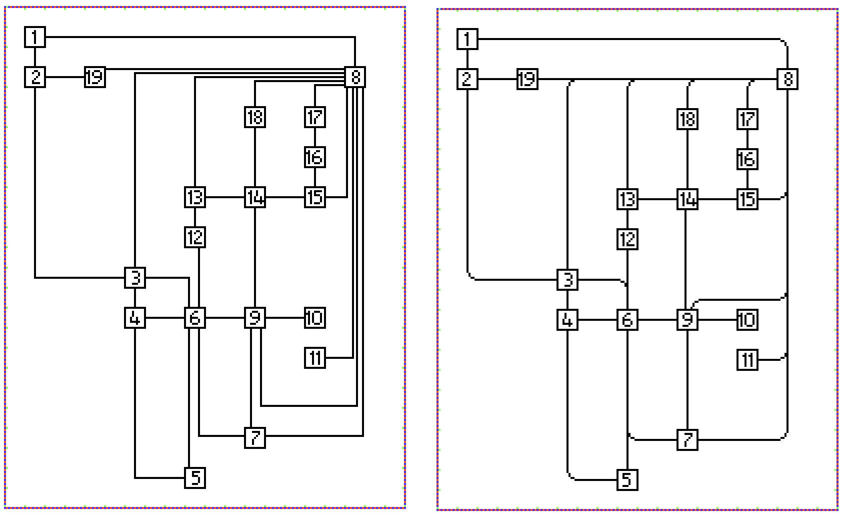

**Figure 2 - An example of  bus mode drawing**

Bus-mode drawing is a powerful way for *structuring* and *clustering* edges of a graph. Buses represent a structured set of interconnections: while a sub-graph can be represented by a single macro-vertex, a set of edges contributing to the same logical function can be condensed into a single bus. This is a special form of hyper-edge. The use of labels on the edges of a bus-mode drawing facilitates graph reading, especially in structured drawing with a drawing dispatched on several forms, but they can be used for strongly modelling interconnections.
Our algorithm for constructing the dual graph assigns integer coordinates to each rectangle associated with a vertex. In this way, by scaling the rectangle sizes we can allocate space both for squares representing vertices and zones dedicated to interconnections.
It is a trivial task to derive a bus-mode rectilinear drawing of a graph G from its rectangular dual. Such a drawing could be based on paths all composed by exactly two bends with a pleasant and clear visual effect. We call this kind of drawing the *naïf 2-bends* bus-mode drawing. However, with a minimum effort we can produce a 1-bend (maximum) drawing in linear time.

## 2.3 Hierarchical Drawing of Structured Graphs

A *structured graph (or clustered graph)* is a form of abstraction applied to a large graph in order to make it modular and more manageable. The abstraction consists in collapsing a subgraph to a single vertex (called a *macrovertex*), or to a single link (called a *macrolink*) thus obtaining a simpler and hierarchically described graph. The structuring operation is usually iterated recursively until a large network is decomposed into relatively small and manageable components and sub-components defined at several levels of nesting and adopting a methodology that is usually applied to every large project (software and hardware design) involving hundreds or thousands of components: i.e. “modularity”. Figure 3 shows an example of a structured graph (magari è il caso di spiegare meglio la figura?!).
Formally, a SG is a pair $\mathbf{H=(G,T)}$ where $\mathbf{G}$ is a connected simple (multi- or hyper-) graph and $\mathbf{T}$ is a tree describing the structure of $\mathbf{H}$. The leaves of $\mathbf{T}$ are exactly the vertices of $\mathbf{G}$ and each node $\mathbf{t \in T}$ represent a cluster $\mathbf{G_t=(V_t,E_t)}$ of $\mathbf{G}$ such that $\mathbf{V_t \subseteq V}$ is the subset of vertices that are leave of  the subtree of $\mathbf{T}$ rooted at $\mathbf{t}$. $\mathbf{G_t}$ is the subgraph generated by $\mathbf{V_t}$, while $\mathbf{H_t=(G_t,T_t)}$ is the SG associated with t. We note that the planarity of the underlying base graph does not imply the planarity of a structured graph if arbitrary subgraphs are collapsed. In order to preserve the planarity condition, only connected subgraphs are collapsed.

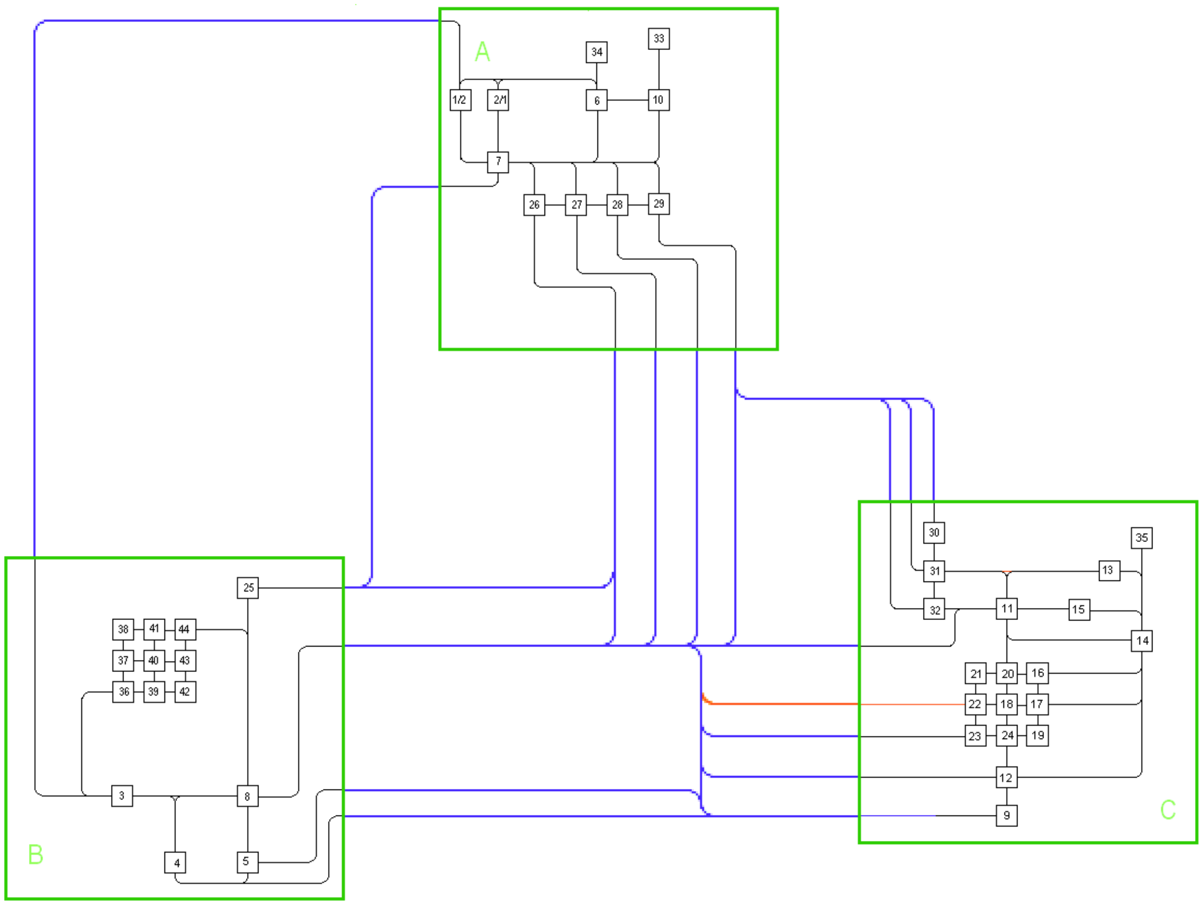

**Figure 3 - Example of a structured graph**

The hierarchical drawing of a structured graph is based on the parallel of structured rectangular dual graph (SRDG). The SRDG of a structured graph: a tree of rectangles **(R,T)** such that:
The (structured) rectangular dual **R(H)** of a structured (clustered) graph **H=(G,T)** is the rectangular dual graph **R(G)** of **G** such that:

- each graph of the hierarchy $\mathbf{G_t}$, $\mathbf{t \in T}$, admits a rectangular dual $\mathbf{R(G_t)}$ and
- the outermost rectangle $\mathbf{R(G_t)}$ of $\mathbf{G_t}$ is the rectangle representing **t** in the rectangular dual $\mathbf{R(G_s)}$, of the father node **s** of **t** in **T**.

## 3. Representation of 3D Electronic Institutions

The aim of this section is to show that rectangular dualization is a very versatile tool which can be applied to completely different fields. This is due to the fact that the underlying problem is always the same: the visualization of a structure which can be modelled with a graph.
3D Electronic Institutions is a methodology for designing highly secure and reliable immersive 3D solutions. This methodology consists of three steps:

- *Specification* of a 3D Electronic Institution;
- *Annotation* of the specification with components of a 3D Virtual World;
- *Generation* of the corresponding 3D environment.

At the first step the institution regulations are defined. An institution is composed of autonomous agents, software programs which, interacting one another, acts for a user or other programs. An Electronic Institution needs some rules, decided on the specification stage, to coordinate the interaction between the agents, according to three types of conventions: the Dialogical Framework, the Performative Structure and the Norms. The first convention refers to the roles agents can play

and what the incompatibilities and relationships among the roles are; the second one determines in which types of dialogues agents can engage. These dialogues are called scenes, which are interconnected in the form of a graph (called Performative Structure) in order to represent sequence of activities, concurrency of activities or dependencies among them. The transit of agents is regulated by special scenes called transitions. Finally the latter convention imposes restrictions on the agents actions within scenes.
In the 3D Virtual World generation scenes and transitions are transformed into 3D rooms and connections are the doors.
Rectangular dualization is used for creating a 2-dimensional map of the institution given the Performative Structure. The automatic generation of a 3D Virtual Word consists of the following four steps:

- The redundant information contained in the Performative Structure is filtered out. If some nodes of the graph are connected with more than one arc, only one randomly selected arc is left and all the others are deleted. Next, the Performative Structure is transformed into a format compatible with OcORD input;
- OcORD creates the rectangular dual of the transformed graph. In the rectangular dual, scenes and transitions are transformed into rooms and connections are visualized as doors;
- A 3D Virtual World is generated from the 2D map created at the previous step;
- The generated 3D Virtual World is visualized and connected to the infrastructure, which controls the correct behaviour of the participants.

## 4. Conclusions

The main reasons behind our interest on rectangular dualization technique have been presented in the previous sections. We briefly described OCoRD, our tool that computes the rectangular dual of a planar biconnected graph and two possible useful applications of it. However it is possible to cite many other application of the rectangular dual [1]. As a future work we plan to improve OcORD, in the sense specified in section 1, and to better integrate it in the automatic generation of 3D Electronic Institutions.

## References

1. Arita T., Motohashi T., Tsuchida K., Yaku T., “An Octal Degree Graph Representation for the Rectangular Dissection”. Proceedings of applied mathematics symposium, 2004, 131-136.
2. Bhasker J., Sahni S., “A linear algorithm to check for the existence of a rectangular dual of a planar triangulated graph”, Networks 7 (1987), pp. 307-317.
3. Fößmeier U., Kant G., Kaufmann M., “2-Visibility Drawing of Planar Graphs”, Lecture Notes In Computer Science; Vol. 1190.
4. Kant G., He X., “Two algorithms for finding rectangular duals of planar graphs.”, Technical report, June 03 1994
5. Lai Y., Leinwand M., “Algorithms for Floorplan Design Via Rectangular Dualization”, IEEE Transactions on Computer-Aided Design, V. 7, N. 12, De. 1988, pp. 1278-1289.