RUTCOR
RESEARCH
REPORT

# A Better Heuristic for Orthogonal Graph Drawings

Therese Biedl [a], Goos Kant [b]

RRR 12-95, June 1995. Updated September 1996.

RUTCOR • Rutgers Center for Operations Research • Rutgers University • P.O. Box 5062 • New Brunswick New Jersey • 08903-5062
Telephone: 908-445-3804
Telefax: 908-445-5472
Email: rrr@rutcor.rutgers.edu
http://rutcor.rutgers.edu/rrr

[a]RUTCOR, Rutgers University, P.O.Box 5062, New Brunswick, NJ 08903, *therese@rutcor.rutgers.edu*

[b]Department of Computer Science, Utrecht University, Padualaan 14, 3584 CH Utrecht, the Netherlands, *goos@cs.ruu.nl*

Rutcor Research Report
RRR 12-95, June 1995. Updated September 1996.

# A Better Heuristic for Orthogonal Graph Drawings

Therese Biedl Goos Kant

**Abstract.** An orthogonal drawing of a graph is an embedding in the plane such that all edges are drawn as sequences of horizontal and vertical segments. Linear time and space algorithms for drawing biconnected planar graphs orthogonally with at most $2n + 4$ bends on a grid of size $n \times n$ are known in the literature.

In this paper we generalize this result to connected and non-planar graphs. Moreover, we show that in almost all cases each edge is bent at most twice. The algorithm handles both planar and non-planar graphs at the same time.

**Acknowledgements**: Research of the second author was supported by the ESPRIT Basic Research Actions program of the EC under contract No. 7141 (project ALCOM II).

# 1 Introduction

The research area of *graph drawing* has become an extensively studied field which presents an exciting connection between computational geometry and graph theory. The wide spectrum of applications includes VLSI-layout, software engineering, and project management (see [6] for an up-to-date overview with more than 300 references). The aesthetic quality of a drawing cannot be precisely defined, and depending on the application different criteria have been used. Some important characteristics of a "readable" representation are the number of bends, the number of crossings, the sizes of the angles, and the required area.

In this paper we study the problem of *orthogonal drawings*, i.e. the drawing of a graph $G = (V, E)$ where each edge is represented by a sequence of alternating horizontal and vertical segments. Such a representation is possible only when every vertex has at most four incident edges. On the other hand, every such graph has an orthogonal drawing. In [15, 11] it was shown that deciding whether $G$ can be embedded in a grid of prescribed area is NP-complete. Schäffter showed that any graph can be embedded in a $2n \times 2n$-grid with at most two bends per edge [21]. For graphs with maximum degree 3, a very recent paper showed that they can be embedded in an $\lceil \frac{n}{2} \rceil \times \lceil \frac{n}{2} \rceil$-grid with $\lceil \frac{n}{2} \rceil + 2$ bends [3].

If $G$ is planar it can be embedded in an $n \times n$ grid with $2n + 4$ bends if it is biconnected, and $2.4n + 2$ bends otherwise [22, 24]. The number of bends along each edge is at most 4. For triconnected planar graphs better bounds are known [14]. Independently of this work, Liu, Morgana & Simeone [18] presented a linear-time algorithm for drawing biconnected planar graphs with at most 2 bends per edge (except the octahedron) and at most $2n + 4$ bends in total on an grid of size at most $(n + 1) \times (n + 1)$.

If a combinatorial embedding is given, an orthogonal representation of it with minimal number of bends can be computed in $\mathcal{O}(n^2 \log n)$ time [23]. However, the number of bends per edge can be large, which makes the drawing unattractive. If the embedding is not given, the problem becomes NP-hard [13]. In particular, it is even NP-hard to approximate the minimum number of bends in a planar orthogonal drawing with an $\mathcal{O}(n^{1-\epsilon})$ error for any $\epsilon > 0$ [13].

The latter motivates the research for a quick and very general heuristic to construct orthogonal representations of planar and non-planar graphs. In this paper we present a new algorithm that runs in $\mathcal{O}(n)$ time and produces orthogonal drawings of connected planar and non-planar graphs with the following properties: (i) the total number of bends is at most $2n + 2$; (ii) the number of bends along each edge is at most 2 (unless the graph is the octahedron and we want a planar drawing) (iii) the area of the embedding is $n \times n$. For planar biconnected graphs, linear time algorithms were already known, yielding an area of at most $n \times n$ and at most $2n + 4$ bends [22]. Hence our contribution is the fact that every edge is bent at most twice, and the generalization to non-biconnected and non-planar graphs. The result is obtained by constructing orthogonal drawings of the blocks using the so-called $st$-ordering and merging all these drawings into one orthogonal drawing of the entire graph.

The paper is organized as follows: Section 2 gives some definitions and introduces the *st*-ordering. In Section 3 the algorithm for biconnected graphs is explained. It is extended to non-biconnected graphs in Section 4. In Section 5 we explain how to arrive at a linear time implementation. In Section 6 we give remarks regarding issues that are vital for implementing the algorithm. Section 7 contains remarks and open problems.

# 2 Definitions

Let $G = (V, E)$ be a graph, $n = |V|, m = |E|$. For most of the paper, we consider only *4-graphs*, i.e. graphs of maximum degree 4. Here $m = \frac{1}{2}\sum_{v \in V} deg(v) \leq 2n$. Call such a graph *4-regular* if all vertices have exactly 4 neighbors. An *orthogonal drawing* of $G$ is an embedding of $G$ in the plane such that all edges are drawn as sequences of horizontal and vertical lines. A point where the drawing of an edge changes its direction is called a *bend* of this edge. If the drawing of an edge has at most $k$ bends we call this edge $k$-bent. We assume that all vertices and bends are drawn on integer points. If the drawing can be enclosed by a box of *width* $n_1$ and *height* $n_2$ we call it an embedding with *gridsize* $n_1 \times n_2$ and *area* $n_1 \cdot n_2$.

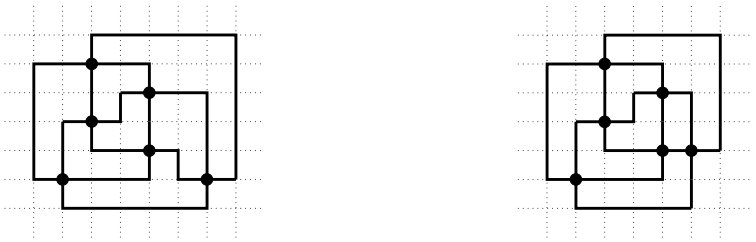

Figure 1: Two drawings of the octahedron, one in a $7 \times 6$-grid and the other in a $6 \times 6$-grid. The first one has been produced with BICONN of Section 3, the second with the improvement of it.

Unless otherwise specified we consider only simple and connected graphs. A vertex $v$ is called a *cutvertex* if removing $v$ and its incident edges from $G$ splits $G$ into two graphs. Call $G$ *biconnected* if it has no cutvertex. The *blocks* of $G$ are its maximal biconnected subgraphs.

A graph is called *planar* if it can be drawn without crossing. A *(combinatorial) embedding* of a planar graph is a representation in which at every vertex all edges are sorted in clockwise order with respect to a planar drawing. The *octahedron* is the unique simple planar 4-regular graph with 6 vertices, see also Figure 1.

An *st-ordering* is an ordering $\{v_1, v_2, \ldots, v_n\}$ of the vertices such that every $v_j$ $(2 \leq j \leq n-1)$ has at least one *predecessor* and at least one *successor*, i.e. neighbors $v_i, v_k$ with $i < j < k$. If $G$ is biconnected, then for any choice of vertices $s, t \in V$ there exists an st-ordering such that s is the first and t is the last vertex [17]. The edges from $v_i$ to its predecessors (successors) are called *incoming* (*outgoing*) edges of $v_i$. Their number is denoted by $indeg(v_i)$ respectively $outdeg(v_i)$.

# 3 Drawing biconnected graphs

The idea for drawing biconnected graphs is the same for planar and non-planar graphs. Independently, Liu et al. [18] came up with a similar technique for planar biconnected graphs. However, we show how to handle the non-planar case as well while maintaining the same bounds.

## 3.1 Embedding in an $(m - n + 1) \times (n + 1)$-grid

Given a biconnected 4-graph $G = (V, E)$. Choose some vertices $s$ and $t$, namely choose $t$ to have minimum degree and, in the planar case, $s$ to be on one face with $t$ which then becomes the outer-face. Obtain an $st$-ordering for $G$. If $G$ is planar, we compute a special $st$-ordering such that the edges $(v_1, v_2)$ and, possibly, $(v_{n-1}, v_n)$ are on the outer-face (see also Appendix A).

We hold the invariant that at every stage every edge where exactly one endpoint is drawn is associated to a column. The drawing of the edge ends in that column which is empty above that point. We draw the first vertex as shown in Figure 2.

Figure 2: Embedding of the first vertex for various degrees.

To embed $v_k$, $k \geq 2$, we distinguish cases by the indegree of $v_k$. We add a new row on top and as many columns to the right and left of $v_k$ as necessary to maintain the invariant. The last vertex might have four predecessors, in which case we add two rows to accommodate it. The procedure is demonstrated for a vertex of degree 4 in Figure 3. For a vertex of smaller degree, we use less columns.

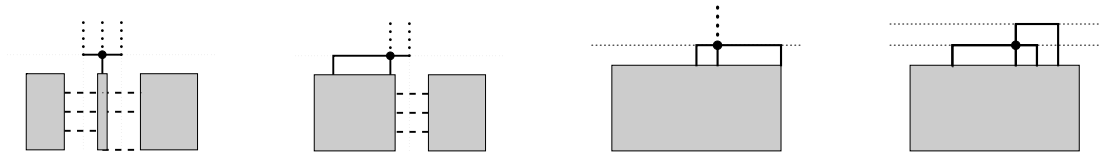

Figure 3: Embedding $v$ ($indeg(v) = 1, 2, 3, 4$).

Each outgoing edge of $v$ is assigned to one of the columns. This assignment is made according to the planar embedding for planar graphs. For non-planar graphs, any assignment is feasible.

We refer to this algorithm as BICONN. We estimate the obtained grid-size and number of bends in the following. Define the *characteristic function* $\chi(...)$ to be 1 whenever what is inside the parenthesis is true, and 0 otherwise.

**Lemma 3.1** *The width is $m - n + 1$, the height is at most $n + \chi(\deg(v_n) = 4)$.*

**Proof:** When embedding $v_i \neq v_1, v_n$ we have $1 \leq outdeg(v_i) \leq 3$, since we have an $st$-ordering. As one checks easily case by case, we increase the number of columns by $outdeg(v_i) - 1$. $v_1$ uses $outdeg(v_1)$ new columns. $v_n$ uses no new column, and $outdeg(v_n) = 0$. So the number of columns is $outdeg(v_1) + outdeg(v_n) + \sum_{v \neq v_1, v_n}(outdeg(v) - 1) = \sum_{v \in V}(outdeg(v) - 1) + 2 = m - n + 2$.

Every vertex $\neq v_n$, increases the number of rows by 1. $v_n$ needs two rows if it has degree 4, and one row otherwise. So the number of rows is $n + \chi(deg(v_1) = 4) + \chi(deg(v_n) = 4)$. The proof now is finished since the width (height) is one less than the number of used columns (rows). □

**Lemma 3.2** *There are at most $2m - 2n + 3 + \chi(\deg(v_1) = 4) + \chi(\deg(v_n) = 4)$ bends.*

**Proof:** Every $v_i \neq v_1, v_n$ has $1 \leq indeg(v_i) \leq 3$ and $1 \leq outdeg(v_i) \leq 3$, since we have an $st$-ordering. Case by case one checks that there are $indeg(v_i) - 1$ and $outdeg(v_i) - 1$, hence $deg(v_i) - 2$ new bends. Embedding $v_1$ gives at most 4 bends. Embedding $v_n$ gives $deg(v_n)$ bends if $deg(v_n) = 4$ and $deg(v_n) - 1$ bends otherwise. Hence the number of bends is at most $\sum_{v \in V}(deg(v) - 2) + 3 + \chi(deg(v_n) = 4) = 2m - 2n + 3 + \chi(deg(v_n) = 4)$. □

**Lemma 3.3** *All but at most two edges are 2-bent. The two exceptions are the edge at the bottom connection of $v_1$, which exists only if $\deg(v_1) = 4$; and the edge at the top connection of $v_n$, which exists only if $\deg(v_n) = 4$.*

**Proof:** Assume we have an edge $e = (v_i, v_j)$ with $i < j$. We consider the number of bends at $e$ when embedding either endpoint. If $outdeg(v_i) \leq 3$, then $e$ gets at most one bend at $v_i$. If $indeg(v_j) \leq 3$, then $e$ gets at most one bend at $v_j$. So the only way $e$ could have more than two bends is that either $outdeg(v_i) = 4$ or $indeg(v_j) = 4$.

This can happen only if $i = 1$ and $deg(v_1) = 4$; or if $j = n$ and $deg(v_n) = 4$. Moreover, it can happen only to the edge at the bottom connection of $v_1$, since only this edge gets two bends at $v_1$; or to the edge at the top connection $v_n$, since only this edge gets two bends at $v_n$. □

**Lemma 3.4** *All edges are 2-bent for non-planar drawings.*

**Proof:** Since we have an $st$-ordering, there must be edges $(v_1, v_2)$ and $(v_{n-1}, v_n)$. Assume $deg(v_1) = 4$. Since we have a non-planar drawing, we have the freedom to assign the edge $(v_1, v_2)$ to the connection at the bottom of $v_1$. Thus, this edge has only two bends.

Assume $deg(v_n) = 4$. If the left-most or right-most column of the incoming edges belongs to the edge $(v_{n-1}, v_n)$, then we can make this edge use the top connection of $v_n$, and thus the edge gets only two bends. Otherwise, we slightly change the drawing of $v_n$ and introduce a crossing, so that the edge gets only two bends. □

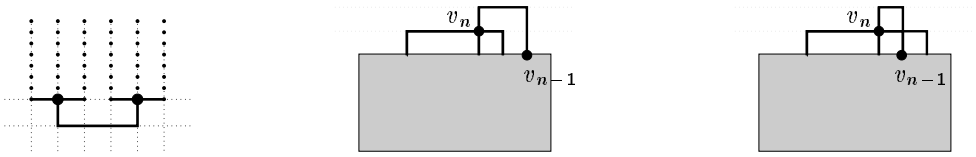


Figure 4: We use the bottom connection at $v_1$ for the edge $(v_1, v_2)$. This also saves one row. We use the edge $(v_{n-1}, v_n)$ for the top-connection at $v_n$, and introduce a crossing if necessary. Then all edges have at most 2 bends.

**Lemma 3.5** *All edges can be made 2-bent for planar graphs unless the graph is the octahedron.*

**Proof:** We apply exactly the same technique as in the previous proof. However, since we have no choice in the assignment of columns, we have to make sure that the edge $(v_1, v_2)$ is on the outer-face of the planar embedding. Also, since we may not introduce a crossing, we have to make sure that the edge $(v_{n-1}, v_n)$ is on the outer-face. Such an $st$-ordering exists (after a possible change of the planar embedding) for all graphs except the octahedron (see Appendix A). □

**Lemma 3.6** *BICONN gives a planar orthogonal drawing that exactly reflects the planar embedding which we had after computation of the st-ordering.*

**Proof:** Obviously there is no crossing after embedding $v_1$. We will show that we produce no crossing when adding $v_{i+1}$, $i > 1$. Let $G(i)$ be the subgraph induced by $v_1, \ldots, v_i$, embedded as induced by $G$. Let $E_i$ be the set of edges from $G(i)$ to $G - G(i)$. For an edge in $E(i)$ the endpoint in $G(i)$ must be on the outer-face of $G(i)$, otherwise we had a crossing. At least two edges of $E(i)$ must be on the outer-face, by biconnectivity. In fact, exactly two edges of $E(i)$ are on the outer-face, otherwise we did not have an $st$-ordering. Let $E(i)$ be sorted in clockwise order, starting and ending with the two edges on the outer-face.

Let $L(i)$ be the list of columns that contain an unfinished edge, at the time before embedding $v_{i+1}$. Let them be sorted from left to right. The crucial observation is that $E(i)$ and $L(i)$ are in correspondence, i.e. the $j$th element in $E(i)$ ends in the $j$th element of $L(i)$. We show this by induction on $i$.

The claim is true for $v_1$, since we assigned columns according to the planar embedding. Assume it was true after step $i$. By planarity the incoming edges of $v_{i+1}$ are consecutive in $E(i)$. For if there were another edge between them, then it would not be on the outer-face of $G(i+1)$, a contradiction. Since $E(i)$ and $L(i)$ are in correspondence, the columns of the incoming edges of $v_i$ are in consecutive order in $L(i)$, and we do not cross any other edge when embedding $v_{i+1}$.

For any $st$-ordering the incoming (outgoing) edges of a vertex $v$ are consecutive in the rotation around $v$ [24]. Since we assigned columns according to the planar embedding, after embedding $v_{i+1}$ we have the correspondence between $E(i+1)$ and $L(i+1)$. With that we also have shown that the drawing exactly reflects the planar embedding. □

### 3.2 Embedding in an $n \times n$-grid

It is possible to save a little bit in grid-size and bends for simple graphs. It might seem tedious to spend extra effort in order to reduce the width by just one unit. However, this is extremely useful when embedding non-biconnected graphs. Here each block is embedded separately, so in all we achieve a reduction in width equal to the number of blocks.

The crucial idea is that it is possible to find a vertex $v^*$ which can be embedded in the row of one of its neighbors. Finding $v^*$ needs a lengthy case analysis, which we defer to Appendix B. For now, it suffices to know that $v^*$ is chosen after we had an $st$-ordering and a planar embedding, and its choice does not change either.

**Lemma 3.7** *Assume $G$ is a simple. There exists a vertex $v^* \neq v_1, v_n$, and an improvement of BICONN, such that the width is $m-n+\chi(deg(v^*)=2)$, the height is $n-1+\chi(\deg(v_n)=4)$, and the number of bends is $2m-2n+1+\chi(deg(v^*)=2)+\chi(\deg(v_n)=4)$.*

**Proof:** See Appendix B. □

With this improvement, we get the following lemma for simple graphs.

**Lemma 3.8** *For simple graphs, the width and height is $n-1+\chi(G$ 4-regular$)$, and the number of bends is $m+2\cdot\chi(G$ 4-regular$)$.*

**Proof:** Apply BICONN and the improvement. We distinguish cases by the number of edges. Assume $m \leq 2n-2$. Then, by the choice of $v_n$ as a vertex of minimal degree, we have $deg(v_n) \leq 3$. Therefore, we get at most an $(m-n+1)\times(n-1)$-grid and $2m-2n+2$ bends. Applying $m \leq 2n-2$, we arrive at the results.

Assume $m \geq 2n-1$. Then, by the choice of $v_n$ as a vertex of minimal degree, we have $deg(v) \geq 3$ for all $v \neq v_n$. Since $v^* \neq v_n$, we have $deg(v^*) \geq 3$, and we get an $(m-n)\times(n-1+\chi(deg(v_n)=4))$-grid and $2m-2n+1+\chi(deg(v_n)=4)$ bends. If $m=2n-1$, then $deg(v_n)\leq 3$, and the bounds follow. If $m=2n$, then $G$ is 4-regular, and the bounds follow as well. □

We reformulate the worst case of Lemma 3.8, and combine it with the results of the previous section, to give the main theorem for biconnected graphs.

**Theorem 1** *Let $G$ be a biconnected simple 4-graph with $n$ vertices. Then $G$ can be embedded in an $n\times n$-grid with at most $2n+2$ bends. If $G$ is planar then so is the drawing. Every edge is 2-bent, unless $G$ is the octahedron and we want a planar drawing.*

## 4 Connected graphs

In this section we describe how to embed simple graphs which are not biconnected with bounds similar to those in Theorem 1. To do so we split the graph into its blocks and embed them separately.

We need to draw parts of the graph in a special way. To describe it, we need two definitions. We say that a graph is drawn with $v$ as *final vertex* if $v$ and its incident edges are drawn in the last row. This in particular implies that $deg(v) \leq 3$. We say that a vertex $w$ is drawn *with right angle* if $deg(w) = 2$, and the two incident edges form a right angle. Using BICONN, we can draw every biconnected graph such that one vertex is drawn as final vertex, and any number of vertices is drawn with right angle.

**Lemma 4.1** *Let $G$ be a simple 4-graph. Let $v$ with $\deg(v) \leq 3$ be given (and on the outer-face in the planar case). Let $w_1, \ldots, w_s$ be vertices in $V - \{v\}$ with $\deg(w_j) = 2$, $j = 1, \ldots, s$.*

*$G$ can be drawn with $v$ as final vertex in an $(n-1) \times (n-1)$-grid with $m$ bends. Every edge is 2-bent. Every $w_j$, $j = 1, \ldots, s$ is drawn with right angle.*

**Proof:** Choose an $st$-ordering with $v$ as last vertex, and produce a drawing $\Gamma$ of $G$ using BICONN and the improvement. This draws $v$ as final vertex. Every edge is 2-bent since $G$ is not 4-regular and therefore not the octahedron.

Now for each $w_j$, $j = 1, \ldots, s$, check whether it is drawn with right angle. If it is not, check whether any incident edge of $w_j$ has a bend. If so, move $w_j$ to this bend. Otherwise, by the construction of BICONN and the improvement, we know that $w_j$ is drawn using the top and bottom connection. Cut the drawing at the column of $w_j$, and add a new column, as shown in Figure 5. Place $w_j$ at one of the resulting right angles. Since both incident edges of $w_j$ had no bend before, all edges in the resulting drawing have at most 2 bends.


Figure 5: If $w_j$ is not drawn with a right angle, we can achieve this by cutting the drawing, and adding a column and a bend.

So in order to achieve a right angle at $w_j$, we at the worst add a column and one bend. We have $m \leq 2n - 1 - s$, since $deg(v) \leq 3$, and $deg(w_j) = 2$ and $w_j \neq v$ for $j = 1, \ldots, s$. Roughly speaking, this reduction in the number of edges makes up for the added columns and bends. To precisely analyze the obtained size, we distinguish cases depending on $v^*$, the vertex crucial for the improvement step.

If $deg(v^*) = 3$, then $\Gamma$ was an $(m-n) \times (n-1)$-grid and had $2m - 2n + 1$ bends. So the final width is at most $m - n + s \leq n - 1$, and the final number of bends is at most $2m - 2n + 1 + s \leq m$. If $deg(v^*) = 2$, then $v^*$ was drawn with right angle (see Appendix B). So if $v^* \in \{w_1, \ldots, w_s\}$, then we had to add at most $s - 1$ columns and bends. So the final width is at most $m - n + 1 + s - 1 \leq n - 1$, and the final number of bends is at most $2m - 2n + 2 + s - 1 \leq m$. If $deg(v^*) = 2$, but $v^* \notin \{w_1, \ldots, w_s\}$, then in fact $m \leq 2n - 2 - s$. The final width is at most $m - n + 1 + s \leq n - 1$, and the final number of bends is at most $2m - 2n + 2 + s \leq m$. □

The following two lemmas show how by merging such drawings we can get a drawing of the full graph of similar size. We distinguish by whether the graph has an edge whose removal disconnects the graph (such an edge is called a *bridge*). Also, we first treat graphs that have at least one vertex with degree less than four.

**Lemma 4.2** *Let $G$ be a simple 4-graph without bridges. Let $v$ with* $\deg(v) \leq 3$ *be given (and on the outer-face in the planar case). $G$ can be drawn with $v$ as final vertex in an $(n-1) \times (n-1)$-grid with $m$ bends. Every edge is 2-bent.*

**Proof:** We proceed by induction on the number of cutvertices. In the base case $G$ is biconnected and we apply BICONN and the improvement, choosing $v$ as last vertex for the $st$-ordering. This gives the desired bounds by Lemma 3.8, since $G$ is not 4-regular.

For the induction step note first that $v$ cannot be a cutvertex (otherwise by $deg(v) \leq 3$ it would be contained in a bridge). Let $G_0$ be the unique block containing $v$. Let $v_1, \ldots, v_s$ be the cutvertices of $G$ in $G_0$. Let $G_i$ be the subgraph of $G$ consisting of $v_i$ and the connected components of $G - v_i$ not containing $G_0$. Denote by $n_i$ respectively $m_i$ the number of vertices and edges of $G_i$ $(i = 0, \ldots, s)$. There is no edge in common between any of the $G_i$'s, so $\sum_{i=0}^{s} m_i = m$. Also, the intersection of the vertices of $G_0$ and $G_i$ $(1 \leq i \leq s)$ is $v_i$, and the intersection of the vertices of $G_i$ and $G_j$ is empty if $i \neq j$. Hence $\sum_{i=0}^{s} n_i = n + s$.

$v_i$ is not a cutvertex for $G_i$, so $G_i$ has fewer cutvertices than $G$. If $G$ was planar and we take as embedding of $G_i$ the one that is induced by $G$ then (since $v$ was on the outer-face of $G$) also $v_i$ is on the outer-face of $G_i$. So by induction $G_i$ can be embedded in an $(n_i - 1) \times (n_i - 1)$-grid with $m_i$ bends and $v_i$ as final vertex.

Embed $G_0$ as described in Lemma 4.1, and draw $v_1, \ldots, v_s$ with right angle. For each $G_i$, consider the placement of $v_i$ in the drawing of $G_0$, and assume it was drawn using the right and the top connection. Add $n_i - 1$ rows below the row of $v_i$ and add $n_i - 1$ columns left and right to the column of $v_i$. Merge $G_i$ into these added rows and columns. See also Figure 6.

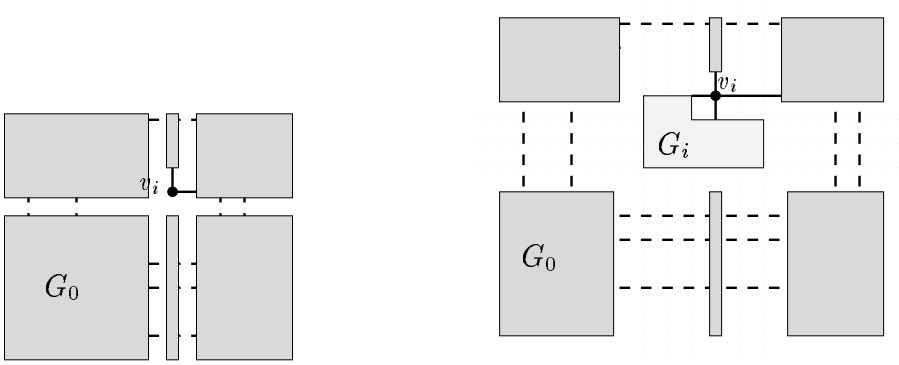


Figure 6: We add rows and columns next to the drawing of $v_i$ in $G_0$, and place $G_i$ in them.

The width of the drawing of $G_0$ was $n_0 - 1$. For each $G_i$, we added $n_i - 1$ columns. The drawing of $G_i$ fits into these columns, since the width of it was $n_i - 1$, and since we can reuse the column of $v_i$ for $G_i$. Therefore, the total width is $\sum_{i=0}^{s}(n_i - 1) = n - 1$. The estimation

for the height is the same. Each of the $G_i$ was drawn with $m_i$ bends. No bends were added when merging the graphs, so the total number of bends is $\sum_{i=0}^{s} m_i = m$. In each drawing of a subgraph, all bends had at most two bends, so the same holds for the final drawing. □

**Lemma 4.3** *Let $G$ be a simple 4-graph. Let $v$ with $\deg(v) \leq 3$ be given (and on the outer-face in the planar case). $G$ can be drawn with $v$ as final vertex in an $(n-1) \times (n-1)$-grid with $m$ bends. Every edge is 2-bent.*

**Proof:** We proceed by induction on the number of bridges. We are done if there are none, by the above lemma. So assume $G$ has a bridge $(v_1, v_2)$. Removing it splits $G$ into two graphs $G_1$ and $G_2$ (we assume that $v_i \in G_i$). Denote by $n_i$ and $m_i$ the number of their vertices and edges. Then $n_1 + n_2 = n$ and $m_1 + m_2 = m - 1$.

Assume $G_1$ contains $v$ (the vertex to be drawn as final vertex). Both $G_1$ and $G_2$ have at least one bridge less. By induction, embed $G_1$ with $v$ as final vertex, and $G_2$ with $v_2$ as final vertex. In the resulting drawing of $G_1$, $v_1$ has a connection in one direction free, assume it is to the left. Add $n_2$ columns to left of $v_1$ and $n_2 - 1$ rows above and below it. Rotate the drawing of $G_2$, place it in the space to the left of $v_1$ and connect $v_1$ and $v_2$. The resulting grid has width $n_1 - 1 + n_2 = n - 1$, height $n - 2$, and $m_1 + m_2 = m - 1$ bends. □

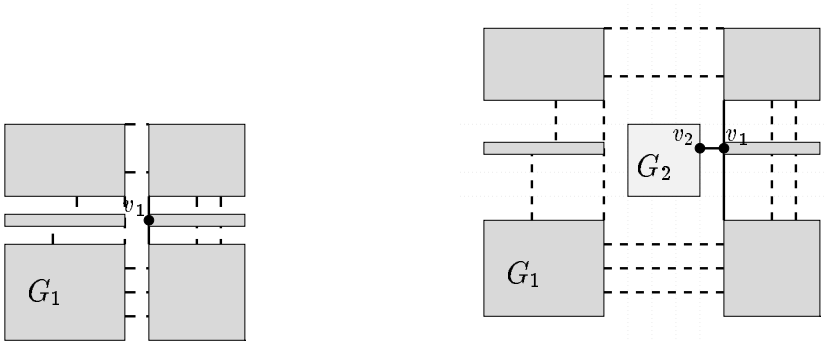


Figure 7: How to merge a subgraph connected by a bridge.

One can see that the grid-size and the number of bends is in fact less in the presence of bridges. By induction, one can show easily that the sum of width and height is at most $2n - 2 - br$ and the number of bends is at most $m - br$ for a graph with $br$ bridges.

So, if $G$ is not 4-regular, we choose any vertex of degree $\leq 3$ to be $v$, and embed $G$ as described in the proofs of the previous lemmas. If $G$ is 4-regular, then it cannot have a bridge (otherwise removing the bridge would give a subgraph with exactly one vertex of odd degree, a contradiction). We split $G$ at a cutvertex $v$ to obtain $G_1$ and $G_2$. If $G$ was planar, this also involves changing the embedding such that $v$ is on the outer-face. Since we have no bridge we know $deg_{G_i}(v) = 2$. Embed $G_1$ and $G_2$ with $v$ as final vertex. Rotate and move the drawing of $G_2$ such that the two drawings of $v$ coincide. The width and height are then at most $n_1 - 1 + n_2 - 1 = n - 1$ and the number of bends is $m_1 + m_2 = m$. See also Figure 8.

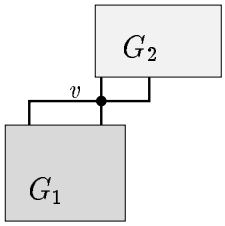


Figure 8: To draw a 4-regular graph, we split it at a cutvertex, and embed each of the subgraphs. Then, by rotating one of the drawings, we merge the two.

So to repeat the whole scheme: If the graph is 4-regular, we choose a cutvertex, change the embedding to bring it on the outer-face (if the graph is planar), and split the graph at this cutvertex. Otherwise, we choose one vertex $v$ with $deg(v) \leq 3$, and change the embedding to bring it to the outer-face (if the graph is planar).

Then, we iteratively remove all bridges, and define for each subgraph the vertex that needs to be drawn as final vertex. For the resulting subgraphs without bridges, we find all cutvertices that are in the same block as the final vertex. We split the graph at these cutvertices, and define the vertex that needs to be drawn as final vertex. This iterates until there are no cutvertices left in any subgraph, so all subgraphs are biconnected. We embed the blocks using BICONN and the improvement. For those blocks that have more than one cutvertex, we also apply the transformation described in Lemma 4.1. Merging the blocks happens in exactly the reverse order as we split them.

We wrap up these observations in one theorem.

**Theorem 2** *Let $G$ be a connected simple 4-graph with $n$ vertices, and with at least one cutvertex. Then $G$ can be embedded in an $(n-1) \times (n-1)$-grid with at most $m$ bends. If $G$ is planar then so is the drawing. Every edge is 2-bent.*

# 5 Linear time complexity

In this section we describe how to implement the algorithm so that it works in linear time. The basic preparations (splitting the graph into blocks, testing planarity, computing a planar embedding if necessary, bringing a particular vertex to the outer-face) can all be done in $\mathcal{O}(m)$ time (see e.g. [5]). We now analyze first the time complexity for embedding a biconnected graph, and then the complexity of merging the graphs.

## 5.1 Time complexity of BICONN

In order to run BICONN, we first need to determine an $st$-ordering, which can be done, for any choice of $s$ and $t$, in linear time [10]. As we show in Appendix A, determining the special order for planar graphs can also be done in linear time. So we only need to worry about the time needed for embedding one vertex.

Since we add new rows always on top of the previous drawing, the $y$-coordinate of a vertex is never changed later. So we can assign an absolute value for the $y$-coordinate of vertex $v$ when embedding it, by taking the value of the highest row and adding one.

We treat the $x$-coordinates differently. Note that we add columns anywhere in the drawing, therefore we can not keep absolute coordinates with the columns. So instead, we maintain a list *Columns*. Every used gridpoint contains a pointer to one element of *Columns* signifying that later all points with a pointer to the same element of *Columns* receive the same $x$-coordinate. Whenever we want to add a column we add a new element in *Columns*, the new columns are added directly left and right of an the columns of some known vertex $v$. So adding a column takes only $\mathcal{O}(1)$.

Let $v$ be the vertex we are dealing with. Let $e_1, \ldots, e_s$ be the incoming edges of $v$. Let $\{c_1, \ldots, c_s\}$ be the columns associated with $e_1, \ldots, e_s$. We have to sort these columns, so that $c_{i_1}$ is to the left of $c_{i_2}$, etc. The column of $v$ is then $c_{i_k}$, where $k = \lceil \frac{s}{2} \rceil$.

Note that sorting is not necessary for planar graphs: if we take $e_1, \ldots, e_s$ to be the incoming edges in counter-clockwise order as given by the planar embedding, then the $c_i$ are automatically in sorted order. So for planar graphs, handling $v$ takes only $\mathcal{O}(1)$ time.

For the non-planar case, we need to be able to compare two columns efficiently. This problem is called the *order maintenance problem*: determining which of two elements comes first in a list under a sequence of Insert and Delete operations. By storing the list as a balanced binary tree, this query can be answered in $\mathcal{O}(\log n)$ time by searching for the least common ancestor. Dietz & Sleator [8] presented a linear space data structure for this problem, answering the order queries in $\mathcal{O}(1)$ time. Since $v$ has only a constant indegree, we need only a constant number of comparisons for sorting. So handling $v$ can likewise be done in $\mathcal{O}(1)$ time for non-planar drawings.

The final $x$-coordinates are computed by traversing *Columns* and assigning ascending values. Every vertex then checks the value of the element it points to and stores that value as its $x$-coordinate. This takes linear time, so the total time for BICONN is linear.

There is another, much simpler, way to achieve linear time complexity for non-planar graphs. It is possible to always add the new columns for a vertex at the extreme right or left end at the drawing. Thus, we can assign fixed $x$-coordinate to all columns, and comparing two columns can be done trivially. However, as shown in experimental studies, this approach is unacceptable since the number of crossings increases dramatically (see [7]).

## 5.2 Complexity of merging graphs

For non-biconnected graphs, we apply a "black-box" scheme to merge the blocks efficiently. We store the coordinate relative to one reference-vertex, and update them once at the end.

Let $G_i$ be a subgraph that either has a cutvertex $v_i$ in common with $G_0$, or that is incident to a bridge $(v_i, w_i)$ with $w_i$ in $G_0$. Assume we have an embedding of $G_i$. Let $B_i$ be the block of $G_i$ containing $v_i$. Denote by $\mathit{width}(B_i), \mathit{height}(B_i)$ the width and height of this drawing and by $\mathit{left}(B_i)$ the number of columns to the left of the columns of $v_i$. With $B_i$, we also store a value $\mathit{rotate}(B_i)$, which is the degree by which the drawing of $G_i$ is rotated when merging.

Let $\Gamma$ be the drawing of $G_0$ that we obtained from Lemma 4.1, that is, all vertices of degree 2 are drawn with a right angle. If $G_i$ shared the cutvertex $v_i$ with $G_0$, then assume that is was drawn using the top and the right connection (the other cases are similar). Add $height(B_i)$ rows below the row of of $v_i$ (to do so, we need to store the rows of $\Gamma$ as a list as well, similarly as described for the columns in the previous section). Also, add $left(B_i)$ columns to the left and $width(B_i) - left(B_i)$ to the right of $v_i$. We set $rotate(B_i) = 0°$. If $G_i$ was incident to a bridge $(v_i, w_i)$, with $w_i \in G_0$, then assume that the left connection at $w_i$ was free in $\Gamma$ (the other cases are similar). We set $rotate(B_i) = 90°$. We add $height(B_i) + 1$ columns to the left of $v_i$. We add $left(B_i)$ rows below the row of $w_i$, and $width(B_i) - left(B_i)$ rows above $w_i$.

After we have done this for all subgraphs $G_i$, we assign new coordinates to the rows and columns of $G_0$. Then, by visiting all vertices $w \in G_0$ once, we calculate $dist_x(w)$ and $dist_y(w)$, the distance of $w$ to the vertex $v$ that was prescribed to be drawn as final vertex of $G$.

To compute the final coordinates we traverse the graph in top-down fashion. Assume that $v$ is already in its final position $(x_v, y_v)$. For every vertex $w \neq v$ in $G_0$ depending on $rotate(G_0)$ we set $x_w = x_v \pm dist_x(w)$, $y_w = y_v \pm dist_y(w)$ or $x_w = x_v \pm dist_y(w)$,$y_w = y_v \pm dist_x(w)$ to get the final coordinate of $w$. Before we proceed in the subgraphs we update $rotate(B_i) = rotate(B_i) + rotate(G_0)$ for the block $B_i$ of $G_i$ that contains $v_i$. Doing so we can compute all final coordinates in linear time.

# 6 Issues for implementing the algorithm

In the following section, we describe experiences that we gained while implementing the algorithm as part of the Graph Layout Toolkit at Tom Sawyer Software.[1]

## 6.1 Saving more rows and bends

With the improvement step, we showed how rows, columns, and bends can be saved by embedding more than one vertex in one row. We showed that this is possible at least once for every simple graph. In fact, if crossings are allowed, this is possible more often, and leads to a reduction of the worst-case bounds on the grid-size (see [20]).

For an implementation, it is vital to include a mechanism that checks whether it is feasible to embed two or more vertices in one row. For this, we need to check whether the columns of the incoming edges of two vertices are "separate", i.e. whether the columns of one vertex are all to the left of the columns of the other vertex. This leads to significant savings for many graphs.

[1] For more information, see [19], or contact *info@tomsawyer.com.*

## 6.2 Avoiding splitting into blocks

For the algorithm BICONN, we used an $st$-ordering, so there was only one source (a vertex without incoming edges), and only one sink (a vertex without outgoing edges). It is crucial that we have only one source, since $v_1$ needs special treatment. But the fact that we have only one sink is not crucial for the algorithm, only for the estimations on the obtained bounds.

It is therefore feasible to run the algorithm with any *pre-oder*, i.e. an order with only one source. For our implementation, we took the following approach: We split the graphs into blocks. Then, for each block we computed an $st$-ordering, such that $s$ and $t$ were two cutvertices (if such existed). We merged these orders into one big ordering. This is an ordering with only one source and with as few sinks as possible.

We applied BICONN with this ordering instead. Thus, there was no need to apply the black-box scheme for merging blocks, and the algorithm was greatly simplified. Since we checked at each step whether the improvement could be applied, we got the same worst-case bounds for this approach.

## 6.3 Graphs with higher maximum degree

Of course it is not possible to embed graphs of maximum degree exceeding 4 in an orthogonal fashion. Since the input graphs however did have higher degree, we split vertices of high degree into a chain of vertices. We sketch this approach here.

Assume we want to embed a vertex $v$ with $deg(v) > 4$. Let $\{e_1, \ldots, e_r\}$ ($r = indeg(v)$) be the incoming edges of $v$ in the order of their assigned columns and let $k = \lceil \frac{r+1}{2} \rceil$. We draw $v$ as a straight vertical line which is placed above the column of $e_k$. By adding $k-1$ rows we can connect all incoming edges of $v$ with this vertical line. With some analysis, we can show that we have at most an $(m - n + 1) \times (m - \frac{n}{2} + \frac{n_2}{2})$-grid and at most $2m - 2n + 4$ bends, where $n_2$ is the number of vertices of degree 2.

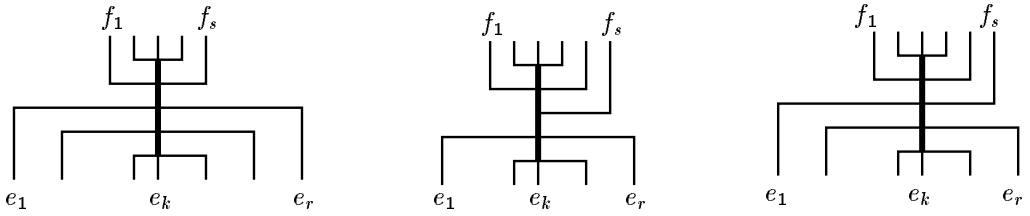


Figure 9: Conversion of a vertex $v$ for different parities of $indeg(v)$ and $outdeg(v)$.

## 6.4 Removal of free space

In our theoretical approach, we always added new rows and new columns for each vertex. This simplifies the description, but it produces a lot of empty space in the drawing, which could have been reused. There are two possible ways out of this: we can either check during the layout phase whether rows or columns can be re-used; or we can apply a post-process step to compact the drawing.

We found the second solution easier and more appealing. When re-using rows and columns, it is often not clear which of many free rows and columns should be taken. Therefore, we used compaction steps using a visibility graph, as described in Lengauer [16]. This greatly reduced the area of the drawing, often by 50% and more.

# 7 Conclusion

In this paper we have considered the problem of orthogonal drawings of graphs. A general linear time algorithm has been presented to construct an orthogonal representation of a connected graph. This algorithm works for planar and non-planar graphs. Our results are a big improvement for non-planar graphs and for non-biconnected planar graphs.

The reader might have noticed that algorithm BICONN also works for graphs with multiple edges. Therefore, we get almost the same results for biconnected multi-graphs (graphs without loops). To draw connected graphs that are not simple, we can again embed every block, and merge them together. This gives worse bounds though, since it is not possible to apply the improvement for each block. On the other hand, this cannot be avoided. It is impossible to drawn any non-biconnected multi-graph with $2n$ bends, since there are non-biconnected multi-graphs that need $\frac{7}{3}n$ bends in any drawing [1].

Also, BICONN also works for planar graphs where the planar drawing is fixed (so-called *plane* graphs). Hereto, we have to choose $v_1$ and $v_n$ such that they both lie on the outer-face. It is therefore not possible to choose $v_n$ to have minimum degree, but the worst-case bound of Theorem 1 holds nevertheless. It is also possible to extend the merging of blocks such that the embedding remains unchanged. This requires more details, the reader is referred to [4].

| | | Connected | Biconnected |
|---|---|---|---|
| Non-planar Graphs | Upper Bounds | $(n-1) \times (n-1)$ (T1)<br>$2n$ (T1) | half-per. $\frac{7}{4}n - 2$ [20]<br>$2n + 2$ (T2) |
| | Lower Bounds | area $\mathcal{O}(n^2)$ [26]<br>$\frac{11}{6}n$ [1] | area $\mathcal{O}(n^2)$ [26]<br>$\frac{10}{6}n$ [1] |
| Planar Graphs | Upper Bounds | $(n-1) \times (n-1)$ (T1)<br>$2n$ (T1) | $n \times n$ [22]<br>$2n + 2$ (T2) |
| | Lower Bounds | $\frac{n-1}{2} \times \frac{n-1}{2}$ [1]<br>$2n - 6$ [1] | $\frac{n-1}{2} \times \frac{n-1}{2}$ [1]<br>$2n - 6$ [1] |
| Plane Graphs | Upper Bounds | $\frac{6}{5}n \times \frac{6}{5}n$ [4]<br>$\frac{12}{5}n + 2$ [4] | $n \times n$ [22]<br>$2n + 2$ (T2) |
| | Lower Bounds | $(\frac{6}{5}n - 2) \times (\frac{6}{5}n - 2)$ [1]<br>$\frac{12}{5}n - 4$ [1] | $(n-1) \times (n-1)$ [1]<br>$2n - 2$ [25] |

Table 1: Overview of the results. The first number denotes the grid-size, unless otherwise specified. The second number denotes the number of bends. Results marked with (T1) or (T2) were proven in this paper, Theorem 1 or 2.

Our results are summarized in the Table 1, which gives the (to our knowledge) best lower and upper bounds for various classes of graphs.

In our algorithm the octahedron is drawn such that one edge has three bends. Even & Granot [9] proved that this in fact cannot be avoided. Hence the octahedron is the only 4-graph which cannot be drawn with at most two bends per edge in a planar drawing.

We end this paper by mentioning some open problems and directions for further research:

- The algorithm is optimal up to a small constant both for the number of bends and for the grid-size for planar biconnected graphs where the embedding may not be changed. However, if the embedding may be changed, there is a gap for the grid-size. If the graph is not planar, then there is also a gap (though small) for the number of bends.
- Recently, Papakostas & Tollis [20] announced an algorithm for drawing non-planar connected 4-graphs with at most $2n + 2$ bends on a grid of size $0.76n^2$, hence an improvement of our results. The natural question is to improve these bounds.
- Di Battista et al. [7] presented an extensive experimental study comparing four graph drawing algorithms, including the algorithm presented here. In that implementation, they placed outgoing edges in each step at the outside of the drawing, leading to a high number of crossings. [7] mention the running time and maximum number of bends per edge as very strong points of our algorithm.
- For planar graphs, an algorithm is known for triconnected 4-graphs that takes advantage of triconnectivity and improves the width and height to $\frac{2}{3}n + 1$ and the number of bends to $\frac{4}{3}n + 4$ [14, 2]. Can this algorithm be extended to non-planar graphs? Or is there another approach that makes use of triconnectivity?
- What are good treatments for graphs of higher degree? We showed a bound of $2m - 2n+4$ bends for graphs of any degree. Can these bounds be improved? In [12], Fößmeier & Kaufmann consider the problem of drawing high degree graphs with a small number of bends. They present efficient algorithms achieving the bend minimum under some additional constraints. It is interesting to investigate the required area in these cases, and to come up with lower bounds for the required area and the number of bends.

# References

[1] T. Biedl, New lower bounds for orthogonal graph drawing, *Proc. Graph Drawing '95*, Lecture Notes in Computer Science 1027, Springer-Verlag, 1995, pp. 28-39.

[2] T. Biedl, Optimal orthogonal drawings of triconnected plane graphs, *Proc. Scandinavian Workshop on Algorithm Theory*, Lecture Notes in Computer Science 1097, Springer-Verlag, 1996, pp. 333-344.

[3] T. Biedl, Improved orthogonal drawings of 3-graphs, *Proc. 8th Canadian Conference on Computational Geometry*, Fiala, Kranakis and Sack (Ed.), International Informatics Series, No. 5, Carleton University Press, 1996, pp. 295–299.

[4] T. Biedl, Optimal orthogonal drawings of connected plane graphs, *Proc. 8th Canadian Conference on Computational Geometry*, Fiala, Kranakis and Sack (Ed.), International Informatics Series, No. 5, Carleton University Press, 1996, pp. 306–311.

[5] N. Chiba, T. Nishizeki, S. Abe, and T. Ozawa, A linear algorithm for embedding planar graphs using PQ-trees, *J. of Comp. and System Sciences* 30 (1985), pp. 54–76.

[6] G. Di Battista, P. Eades, R. Tamassia, and I.G. Tollis, Algorithms for automatic graph drawing: an annotated bibliography, *Comp. Geom.: Theory and Applications* 4 (1994), pp. 235–282.

[7] G. Di Battista, A. Garg, G. Liotta, R. Tamassia, E. Tassinari, F. Vargiu, An experimental comparison of four graph drawing algorithms, *Proc. 11th Annual ACM Symp. on Computational Geometry*, 1995, pp. 306–315.

[8] P.F. Dietz, and D.D. Sleator, Two algorithms for maintaining order in a list, in: *Proc. 19th Annual ACM Symp. Theory of Computing*, 1987, pp. 365–372.

[9] S. Even, and G. Granot, *Rectilinear Planar Drawings with Few Bends in Each Edge*, Manuscript, Faculty of Comp. Science, the Technion, Haifa (Israel), 1993.

[10] S. Even, and R.E. Tarjan, Computing an *st*-numbering, *Theoretical Comp. Science* 2 (1976), pp. 436–441.

[11] M. Formann, and F. Wagner, The VLSI layout problem in various embedding models, *Graph-Theoretic Concepts in Comp. Science (16th Workshop WG'90)*, Springer-Verlag, 1992, pp. 130–139.

[12] U. Fößmeier, and M. Kaufmann, Drawing high degree graphs with low bend numbers, *Proc. Graph Drawing '95*, Lecture Notes in Comp. Science 1027, Springer-Verlag, 1996, pp. 254–266.

[13] A. Garg, and R. Tamassia, On the Computational Complexity of Upward and Rectilinear Planarity Testing, *Proc. Graph Drawing '94*, Lecture Notes in Comp. Science 894, Springer-Verlag, 1995, pp. 286–297.

[14] G. Kant, Drawing planar graphs using the canonical ordering, *Algorithmica* 16 (1996), pp. 4–32.

[15] M.R. Kramer, and J. van Leeuwen, The complexity of wire routing and finding minimum area layouts for arbitrary VLSI circuits, *Advances in Computer Research, Vol. 2: VLSI Theory, F.P. Preparata (Ed.)*, JAI Press, Reading, MA, 1992, pp. 129–146.

[16] Th. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout*, Teubner/Wiley & Sons, Stuttgart/Chichester, 1990.

[17] A. Lempel, S. Even, and I. Cederbaum, An algorithm for planarity testing of graphs, *Theory of Graphs, Int. Symp. Rome* (1966), pp. 215–232.

[18] Y. Liu, A. Morgana, and B. Simeone, General theoretical results on rectilinear embedability of graphs, *ACTA Mathematicae Applicatae Sinica* 7 (1991), pp. 187–191.

[19] B. Madden, P. Madden, S. Powers, and M. Himsolt, Portable Graph Layout And Editing, *Proc. Graph Drawing '95*, Lecture Notes in Computer Science 1024, Springer-Verlag, 1995, pp. 385-395.

[20] A. Papakostas, and I.G. Tollis, *A Pairing Technique for Area-Efficient Orthogonal Drawings*, to appear in *Proc. Graph Drawing '96*, Lecture Notes in Computer Science, Springer-Verlag, 1996.

[21] M. Schäffter, Drawing graphs on rectangular grids, *Discr. Appl. Math.* 63 (1995), pp. 75–89.

[22] J. Storer, On minimal node-cost planar embeddings, *Networks* 14 (1984), pp. 181–212.

[23] R. Tamassia, On embedding a graph in the grid with the minimum number of bends, *SIAM J. Comput.* 16 (1987), pp. 421-444.

[24] R. Tamassia, and I.G. Tollis, Efficient embedding of planar graphs in linear time, *Proc. IEEE Int. Symp. on Circuits and Systems*, Philadelphia, pp. 495–498, 1987.

[25] R. Tamassia, I.G. Tollis, and J.S. Vitter, Lower bounds for planar orthogonal drawings of graphs, *Inf. Proc. Letters* 39 (1991), pp. 35–40.

[26] L.G. Valiant, On non-linear lower bounds in computational complexity, *Proc. 7th Symp. on Theory of Computing*, 1975, pp. 45–53.

# A Computing the *st*-ordering for planar graphs

In this section, we describe how to compute a planar embedding and an *st*-ordering such that both the edge $(v_1, v_2)$ and the edge $(v_{n-1}, v_n)$ are on the outer-face. We call an *st*-ordering a *weak pl-st-ordering*, if $(v_1, v_2)$ is on the outer-face, and a *(strong) pl-st-ordering* if both $(v_1, v_2)$ and $(v_{n-1}, v_n)$ are on the outer-face. We also want to be able to specify the vertices $v_1$ and $v_n$.

As we show now, such an ordering exists for almost all biconnected planar graphs. We do not assume simplicity here, since we need this more general statement. For a face $F$ in a planar embedding, let the *degree* $deg(F)$ be the number of edges incident to the face.

**Lemma A.1** *Let $G$ be a biconnected planar graph. Let $s, t \in V$. If there is a planar embedding of $G$ such that $s, t$ belong to the outer-face $F$; and if $\deg(F) \geq 3$, then $G$ has a weak pl-st-ordering with $s$ as first and $t$ as last vertex.*

**Proof:** In the following for any given planar embedding with $s$ on the outer-face, denote by $v^*$ the neighbor of $s$ to the right on the outer-face. We start with a planar embedding of $G$ such that $F$ is the outer-face, and such that $t$ is not $v^*$ (if necessary, we reflect the embedding to achieve this. It is always possible since $deg(F) \geq 3$).

The next step consists to change the embedding such that $s$ and $v^*$ do not form a *cutting pair*, i.e. removing $s$ and $v^*$ leaves a connected graph. Assume $\{s, v^*\}$ indeed was a cutting pair. Hence $\{s, v^*\}$ share a face $F'$ not incident to $(s, v^*)$. We swap $(s, v^*)$ to $F'$. If there exists more than one edge $(s, v^*)$, then move all other copies of the edge $(s, v^*)$ to $F'$.

Now a new neighbor of $s$ appears on the outer-face and we repeat the argument with it. Can this process cycle, i.e. can the old $v^*$ ever become neighbor of $s$ on the outer-face again? After swapping all copies of $(s, v^*)$, the length of the path between $s$ and $v^*$ on the outer-face increases. It never decreases, therefore $v^*$ can never be neighbor of $s$ on the outer-face again.


Figure 10: Swap the edge away from the outer-face.

So we must stop after at most $deg(s)$ swaps, and then $s$ and its neighbor $v^*$ on the outer-face are not a cutting pair. Let $G_c$ be the graph resulting from contraction of the edge $(s, v^*)$, i.e. delete $s$, $v^*$, and all copies of the edge $(s, v^*)$; add a new vertex $v_c$, and connect it with all edges where one endpoint was $s$ or $v^*$ before.

Since $\{s, v^*\}$ is not a cutting pair, contracting $(s, v^*)$ does not destroy the biconnectivity, so $G_c$ is biconnected. Compute an $st$-ordering for $G_c$ with $v_c$ as first and $t$ as last vertex. Assume it was $\{w_1, \ldots, w_{n-1}\}$, then one checks easily that $\{s, v^*, w_2, \ldots, w_{n-1} = t\}$ is now the desired weak pl-$st$-ordering. □

**Lemma A.2** *Let $G$ be a biconnected planar graph, $s, t \in V$. If there is a planar embedding of $G$ such that $s, t$ belong to the outer-face $F$, and $\deg(F) \geq 4$, then $G$ has a strong pl-$st$-ordering with $s$ as first and $t$ as last vertex.*

**Proof:** We compute the graph $G_c$, the graph with one edge contracted, as described in the above proof. The degree of the outer-face never decreases during these operations. So the degree of the outer-face of $G$ before the contraction is at least 4, and the degree of the outer-face of $G_c$ is at least 3. By the above lemma we can find a weak pl-$st$-ordering of $G_c$ with $t$ as first and the contracted vertex $v_c$ as last vertex. Call it $\{u_1, \ldots, u_{n-1}\}$.

Since the reversal of an $st$-ordering is an $st$-ordering, $\{u_{n-1}, \ldots, u_2, u_1 = t\}$ is an $st$-ordering of $G_c$. Since $u_{n-1} = v_c$, we know that $\{s, v^*, u_{n-2}, \ldots, u_2, t\}$ is an $st$-ordering of $G$. One checks easily that it is in fact a strong pl-$st$-ordering. □

**Lemma A.3** *If $G$ is a simple planar 4-graph, and $G$ is not the octahedron, then BICONN draws $G$ with at most 2 bends per edge.*

**Proof:** Assume first that $G$ is not 4-regular. Then $v_n$ is a vertex with $deg(v_n) \leq 3$. Let $F$ be an incident face which has $deg(F) \geq 3$ by simplicity. Change the embedding such that $F$ is the outer-face. We can find a weak pl-$st$-ordering with Lemma A.1. Applying BICONN with this $st$-ordering, we avoid the edge with more than two bends at $v_1$. By $deg(v_n) \leq 3$ no edge is bent twice with the embedding of $v_n$.

Now assume $G$ is 4-regular, so $deg(v_n) = 4$. Choose any planar embedding of $G$, and then choose as the outer-face a face $F$ with maximum degree. If $F$ has $deg(F) \geq 4$, we choose $s$ and $t$ on this face, take a strong pl-$st$-ordering , and apply BICONN-PLANAR with this ordering. This avoids the edges with more than two bends at both $v_1$ and $v_n$.

So assume $G$ is 4-regular, but all faces in $G$ have degree 3. So $G$ is triangulated and has $3n-6$ edges. This gives $m(G) = 3n-6$, but also $m(G) = \frac{1}{2}\sum_{v \in V} deg(v) = 2n$, and $n = 6$. There is only one planar simple 4-regular graph with 6 vertices, namely, the octahedron. □

We claim that a pl-$st$-ordering can be computed in $\mathcal{O}(m)$ time. Hereto we first have to determine the cutting pairs $\{v_1, w\}$, with $w$ a neighbor of $v_1$. We mark all faces incident to $v_1$. Every neighbor $w$ of $v_1$ belonging to more than two marked faces forms a cutting pair with $v_1$. For every such $w$ mark a face that contains $w$ but not the edge $(v_1, w)$. This operation can be done beforehand in linear time.

# B An improvement for simple graphs

In this section, we study how to reduce the grid-size and the number of bends by a little bit for a simple graphs $G$. Assume an $st$-ordering $\{w_1, \ldots, w_n\}$ of $G$ is given. The first step consists in removing all vertices with indegree 1 and outdegree 1, and replacing them by an edge. We say that "a vertex of degree 2 has been absorbed by an edge". Call the resulting graph $G_r$. We layout $G_r$ first, and then expand the absorbed vertices back into the picture.

Let the number of vertices of $G_r$ be $r$. The $st$-ordering on $G$ induces an ordering $v_1, \ldots, v_r$ on $G_r$. One checks immediately that this also is an $st$-ordering, and that $w_1 = v_1$ and $w_n = v_r$. Define $l$ to be the smallest index such that $b = indeg(v_l) \geq 2$. This exists for $r \geq 3$, since by biconnectivity $indeg(v_r) \geq 2$. If $r \leq 2$, then $G$ was a path, and the claimed bounds hold anyway. Let $v_{i_1}, \ldots, v_{i_b}$ be the predecessors of $v_l$, and assume $i_1 \leq \ldots \leq i_b$. If we define a new ordering $\{v_1, \ldots, v_{i_b}, v_l, v_{i_b+1}, \ldots, v_{l-1}, v_{l+1}, \ldots, v_b\}$, then this is also an $st$-ordering. So we may as well assume $i_b = l-1$. We must have $outdeg(v_j) \geq 2$ for all $j < l-1$: this holds for $v_1$ by biconnectivity, and for all other vertices since otherwise they had been absorbed.

We need a little auxiliary lemma.

**Lemma B.1** *Let $v_l$ and $v_{l-1}$ be defined as before, and let $v_k$ be the unique predecessor of $v_{l-1}$. Either $v_{l-1}$ has no successor other than $v_l$, or we can achieve that the column of the edge $(v_k, v_{l-1})$ is to the left or to the right of all other columns of incoming edges of $v_l$.*

**Proof:** Assume first that $G$ was planar, then so is $G_r$. If the column of $(v_k, v_l)$ is not rightmost or leftmost, then the column of $(v_{l-1}, v_l)$ is also not rightmost or leftmost. But then $indeg(v_l) \geq 3$, and $v_{l-1}$ is not on the outer-face after embedding $v_l$. Therefore, $v_{l-1}$ can have no successor other than $v_l$, since this would imply a crossing.

Now assume $G$ was not planar. Since $outdeg(v_k) \geq 2$, we open a new column when embedding $v_k$. We can make an exception to the usual placement and add this column at the extreme left or right end of the drawing. Using this column for the edge $(v_k, v_{l-1})$, we are done. □

We study different cases regarding $v_{l-1}$ and $v_l$.

**Case 1**: The edge $(v_{l-1}, v_l)$ had absorbed a vertex of degree 2.

We embed $v_{l-1}$ exactly as in BICONN. We embed $v_l$ and make sure that at least one copy of the edge $(v_{l-1}, v_l)$ gets one bend. This is straightforward if $(v_k, v_l)$ has the leftmost or

rightmost column, since then also $(v_{l-1}, v_l)$ gets the leftmost or rightmost column. Otherwise, we must have a multiple edge between $v_{l-1}$ and $v_l$ by $outdeg(v_l) \geq 2$. One of the multiple edges always gets a bend, so again we are done. There are no savings when embedding $G_r$ in this case. The savings result when we expand the absorbed vertices.

**Case 2**: The edge $(v_{l-1}, v_l)$ is a simple edge that also existed in $G$.

Since $b \geq 2$, and since we have a simple edge, we have $i_b > i_1 \geq 1$, so $v_{l-1} \neq v_1$. This in particular implies $indeg(v_{l-1}) \geq 1$.

**Case 2a**: $indeg(v_l) = 2$.

We embed $v_l$ and $v_{l-1}$ together as shown in Figure 11. To see the improvement, we also show how the embedding with BICONN would have been. We see that we save one row, one column and two bends.


Figure 11: The improvement for $indeg(v_l) = 2$, contrasted with the old embedding.

**Case 2b**: $indeg(v_l) = 3$.

At the time of the embedding of $v_{l-1}$, there are three significant columns: $c_{l-1}$, the one assigned to the incoming edge of $v_{l-1}$, and $c_l^1$ and $c_l^2$, the two columns of the other two incoming edges of $v_l$. With the above lemma, we ensure that $c_{l-1}$ is leftmost or rightmost among these three columns. Quite similar as to the first case, we embed $v_l$ and $v_{l-1}$ together in one row. We save one row, one column and two bends.

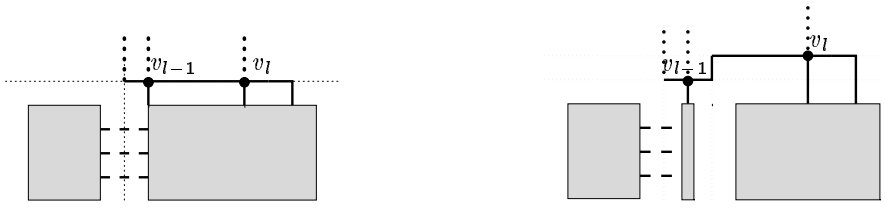


Figure 12: The improvement for $indeg(v_l) = 3$, contrasted with the old embedding.

**Case 2c**: $indeg(v_l) = 4$.

Only $v_r$ can have indegree 4, so $l = r$. Therefore, $v_{l-1} = v_{r-1}$ can have only one successor, $v_r$. By $outdeg(v_{l-1}) \geq 2$, there must be a multiple edge from $v_{l-1}$ to $v_l$. But since the original graph was simple, this implies that $(v_{l-1}, v_l)$ absorbed a vertex, therefore this case has been treated already in Case 1.

This ends the description of the cases. Now for expanding the vertices of degree 2. For each edge $e$ in $G_r$, let $w_{i_1}, \ldots, w_{i_k}$ be the vertices that were absorbed into it, and assume $i_1 < \ldots < i_k$ (the numbers are taken from the $st$-ordering of $G$). Considering the drawing

of $e$ in $G_r$. Notice that $e$ has been drawn with some vertical segment (the only exception to this is the edge between $v_{l-1}$ and $v_l$ in Case 2, but this edge then did not have any absorbed vertices by definition of the case). Let $S$ be the vertical segment.

If one or both of the endpoints of $S$ is a bend, then we place $w_{i_1}$ and/or $w_{i_k}$ at these bends. All other vertices are placed by adding a row between the upper and lower endpoint of $S$. In short: we add at most one row for each absorbed vertex, and neither columns nor bends.

We are now ready to prove all claims that we made about this improvement.

**Lemma B.2** *Assume $G$ is a simple. There exists a vertex $v^* \neq v_1, v_n$, and an improvement of BICONN, such that the width is $m-n+\chi(deg(v^*)=2)$, the height is $n-1+\chi(\deg(v_n)=4)$, and the number of bends is $2m-2n+1+\chi(deg(v^*)=2)+\chi(\deg(v_n)=4)$. If $\deg(v^*)=2$, then $v^*$ is drawn with right angle.*

**Proof:** Let $d$ be the number of vertices that have been absorbed. So $G_r$ has $r = n - d$ vertices and $m_r = m - d$ edges.

Let us first consider Case 1. $G_r$ was drawn with BICONN, so we get a width of $m_r - r + 1 = m - n + 1$ and a height of $r + \chi(deg(v_n) = 4)$. The number of bends is at the most $2m_r - 2r + 3 + \chi(deg(v_n) = 4)$. Now, since $(v_{l-1}, v_l)$ contained an absorbed vertex, and since we made sure that this edge got a bend, we have to add at most $d - 1$ rows and we remove one bend when expanding the absorbed vertices. So the width stays unchanged, the height is $n-d+\chi(deg(v_n)=4)+d-1$, and the number of bends is $2(m-d)-2(n-d)+3+\chi(deg(v_n)=4)-1$. Define $v^*$ to be the absorbed vertex that was placed on the bend. Then $v^* \neq v_1, v_n$ and $deg(v^*) = 2$. Also, $v^*$ is drawn with right angle.

Now consider Case 2, and define $v^* = v_{l-1}$. It is not possible that $v_{l-1} = v_1$, since $v_{l-1} = v_{i_b} \neq v_{i_1}$ (otherwise we had a multiple edge and therefore an absorbed vertex), and therefore $l-1 = i_b > i_1 \geq 1$. Therefore, $indeg(v_{l-1}) \geq 1$, and, by definition of $l$, $indeg(v_l) = 1$. Also, $outdeg(v_l) \geq 2$ (otherwise it would have been absorbed), so $deg(v_l) \geq 3$.

As can be seen from the construction, we save one row, one column, and two bends when embedding $v_{l-1}$ and $v_l$ together. For saving the row, it is important that all vertices were placed on different rows before. This is ensured for all vertices with the construction of BICONN. The only exception is the second vertex, which was placed in the same row as $v_1$, as in Figure 4. However, we never used these row-savings for the worst-case bounds. So if $v_{l-1} = v_2$, then we place $v_2$ and $v_1$ on different rows, and instead place $v_2$ and $v_l$ in the same row.

Therefore, the drawing of $G_r$ has width $m_r - r = m - n$, height $r - 1 + \chi(deg(v_n) = 4)$ and at most $2m_r - 2r + 1 + \chi(deg(v_n) = 4)$ bends. We add at the most $d$ rows to this, and are done. □

Obviously, all changes needed for the improvement can be done in linear time. We only need to compute $v_{l-1}, v_l$ and $v_k$ beforehand, change the column added at $v_k$ in the non-planar case, and embed $v_{l-1}$ and $v_l$ together.

# C An example in pictures

In this section, we give one example graph, and show how the algorithm forms an orthogonal drawing of it.

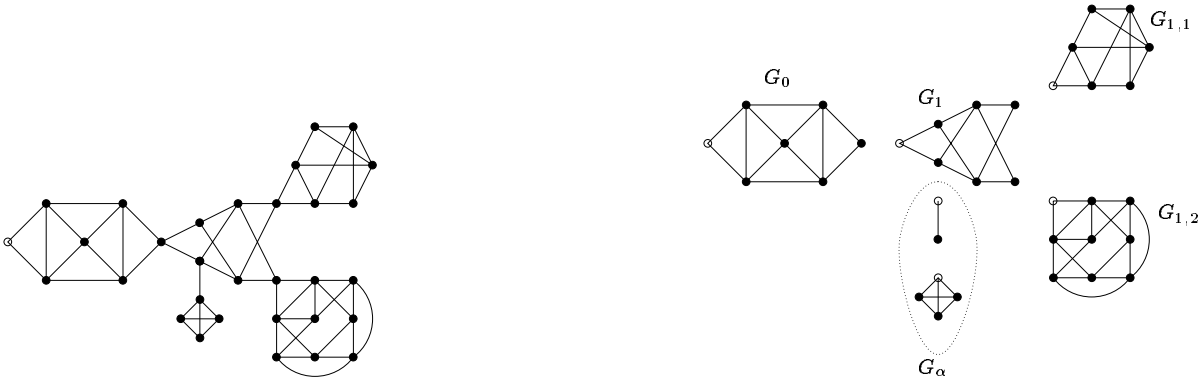


Figure 13: (a) The input-graph. It is not 4-regular, and not biconnected. We choose one vertex to be drawn as final vertex (shown white). (b) We split the graph into its blocks. Along the way, we determine for each block which vertex should be drawn as final vertex (shown white for each block).

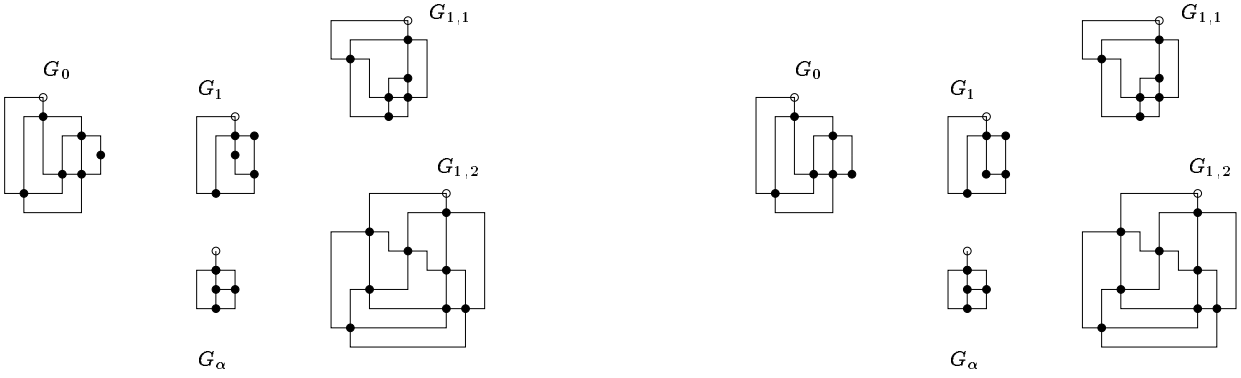


Figure 14: (a) We embed each component with BICONN and the improvement, using the white vertex as the last vertex. Each block turns out to be planar. (b) For $G_0$ and $G_1$, we move the cutvertices such that they are drawn with right angle. In this case here, no columns have to be added.

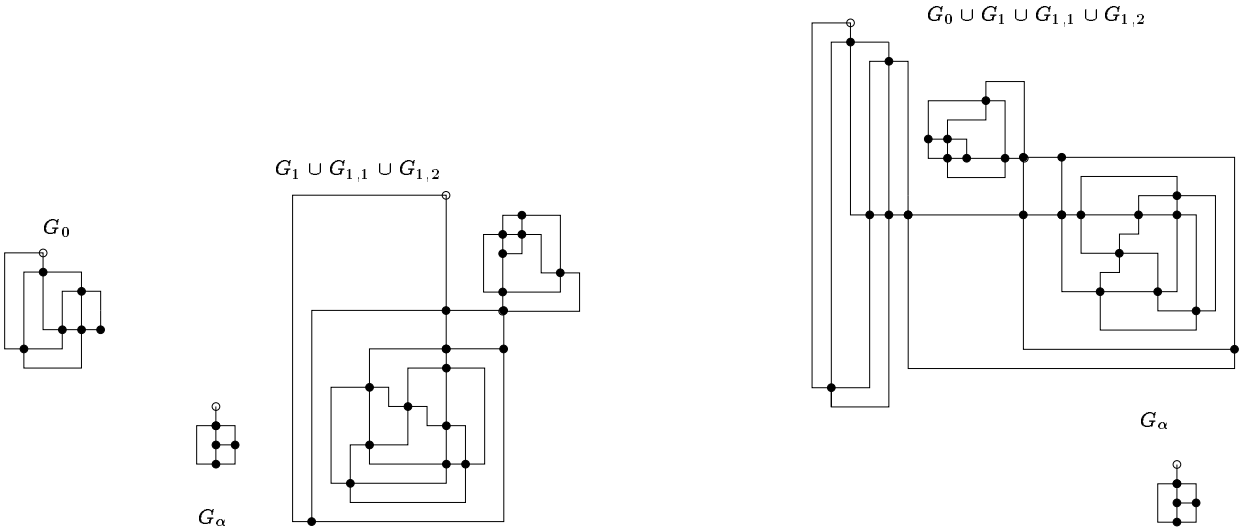


Figure 15: (a) We merge $G_{1,1}$ and $G_{1,2}$ into $G_1$. We have to rotate $G_{1,2}$ by 180 degree. (b) We merge the obtained subgraph into $G_1$. We have to rotate the subgraph.

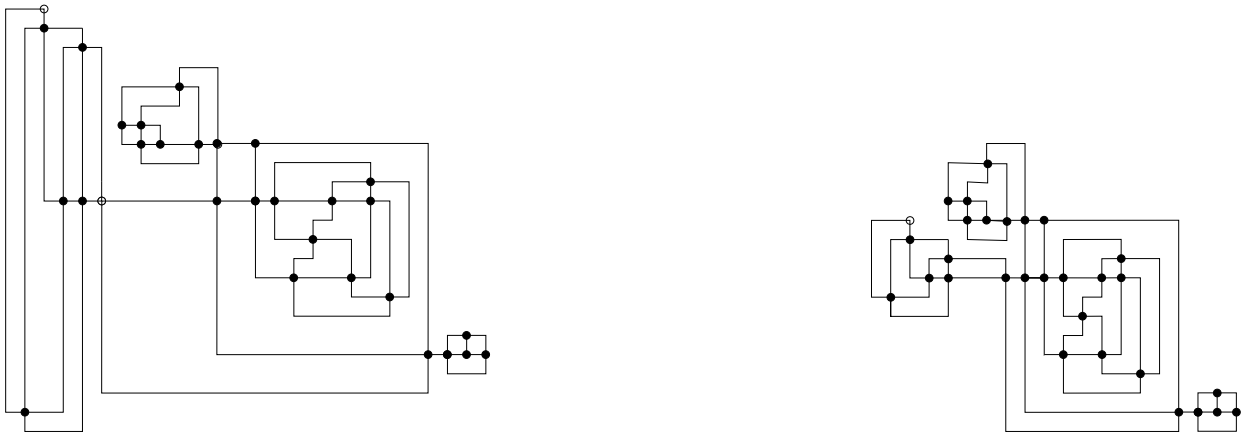

Figure 16: (a) Finally, we merge $G_\alpha$, the subgraph connected by a bridge. (b) After a suitable compaction step, the picture is significantly reduced in size.