The k-lines Drawability Problem for Series-Parallel Digraphs*

Emilio Di Giacomo † *Giuseppe Liotta* †

digiacomo@diei.unipg.it liotta@diei.unipg.it

Stephen K. Wismath ‡

wismath@cs.uleth.ca

August 22, 2002

Abstract

1 Introduction

A straight-line drawing (also known as a *Fary embedding*) is a drawing of a graph in which vertices are points and edges are straight line segments which are pairwise crossing-free. In a straight-line *grid drawing* the additional constraint of having the vertices drawn as points with integer coordinates is imposed. The problem of determining the area required to draw classes of graphs in the 2D Euclidean space has been widely investigated in the literature [2] and an $\Omega(n^2)$ bound is known for planar graphs [1] (where n is the number of vertices). Suprisingly, only a few results are known about compact straight-line grid drawings of planar graphs in 3 dimensions. In particular, the basic question of whether every planar graph admits a straight line drawing in three dimensions with vertices at integer grid points and in $o(n^2)$ volume is still open.

In [6], a novel approach to three dimensional graph drawing is proposed: Given a specified three dimensional grid ϕ, which graphs admit a straight-line crossing-free drawing with vertices located at the grid points of ϕ? If ϕ is chosen so that it has volume V, then a volume bound of V can be determined for any class of graph drawable on ϕ. In particular, [6] focuses on two restricted integer

*Research supported in part by the Natural Sciences and Engineering Research Council of Canada, and by "Progetto Giovani Ricercatori" of the University of Perugia, Italy.

†Dipartimento di Ingegneria Elettronica e dell'Informazione, Universitá degli Studi di Perugia, Perugia, Italy

‡Department of Mathematics and Computer Science, University of Lethbridge, Lethbridge, Alberta, Canada

3D grids, called the *box* and the *prism*. A *box* is a grid consisting of four parallel lines of infinite length placed one grid unit apart from each other. A *prism* is a box with one fewer grid lines. In [6] it is shown that all outerplanar graphs can be drawn on a prism. This result gives the first algorithm to compute a crossing-free straight-line 3 dimensional drawing with linear volume for a non-trivial family of planar graphs.

A generalization of the box and the prism are three dimensional grids consisting of k parallel lines arranged around a cylinder. The study of this kind of grid is motivated, on one side by the problem mentioned above of determining a volume upper bound sub-quadratic for planar graphs, and on the other side by the attempt of ansewering to the following open problem: which is the minimum number of parallel lines arranged around a cylinder that are needed to support a straight-line drawing of any planar graphs?

In this paper we focus our attention on series-parallel graphs, which are a classical subject of investigation in graph drawing [2]. Dujmović, Morin, and Wood prove an $O(n \log^2 n)$ volume bound for series-parallel graphs [5]. Also, Wood [8] improves that result by proving that every series-parallel graph admits a straight-line 3D drawing in $O(n)$ with a constant factor of $1,78 \times 10^8$. The problem of reducing this factor is left open.

The result of this paper can be summarized as follows.

- A lower bound of 5 on the number of lines needed to support a straight-line drawing of each series-poarallel graph is given, by exhibiting a series-parallel graph that does not admit a box-drawing.

- A drawing algorithm is presented that computes a straight-line 3D drawing on 8 lines for a subclass of series-parallel graphs called regular series parallel graphs. The algorithm runs in $O(n)$ time and computes drawings of $16 \times n$ volume.

- The number of lines is reduced to 4 for series-parallel graphs having maximum vertex-degree at most three. The algorithm runs in $O(n)$ time and computes drawings of $4 \times n$ volume.

The remainder of the paper is organized as follows. In Section 2 some basic definitions are given. An example of a graph that can not be drawn on the box is presented in Section 3. The algorithm for computing a straight-line drawing of a regular series-parallel graph on 8 lines is presented in Section 4, while in Section 5 an algorithm for computing a box-drawing of 3-planar series-parallel graph is shown.

2 Preliminaries

In this section we recall some basic definitions and results that are used throughout the paper. A *series-parallel digraph* (also called an *SP-digraph* in the following) is a planar digraph recursively defined as follows [2]:

- An edge joining two vertices is an SP-digraph.
- Let G' and G'' be two SP-digraphs; the digraph obtained by identifying the sink of G' with the source of G'' is also an SP-digraph (*Series Composition*).
- Let G' and G'' be two SP-digraphs; the digraph obtained by identifying the source of G' with the source of G'' and the sink of G' with the sink of G'' is also an SP-digraph (*Parallel Composition*).

This defines the digraphs that are sometimes referred to as *two terminal series-parallel digraphs* (or *TTSP*s). A 3-planar SP-digraph is an SP-digraph having maximumm vertex degree at most three. An SP-digraph has one source and one sink that are called its *poles.* Two parallel component are said to be *nested* if they share a pole and they are not in series. Given a pole v, the *depth* of v is the number of nested parallel component having v has a pole. An SP-digraph *has depth* k if there exist an *st*-orientation in which each pole has depth at most k. A *regular series-parallel graph* is an SP-digraph of depth 1.

We introduce the definition of an *SPQ-tree* of an SP-digraph, which is a simplification of the definition of $SPQR$-trees of general biconnectible planar graphs [4].

A *split pair* of G is either a separation pair or a pair of adjacent vertices of G. A *split component* of G with respect to a split pair $\{s,t\}$ is either an edge (s,t) or a maximal subgraph C of G such that C is an *st*-graph and $\{s,t\}$ is not a split pair of C.

Let G be an SP-digraph with source pole s and sink pole t. An *SPQ-tree T of G* describes a recursive decomposition of G with respect to its split pairs. Intuitively, T represents a natural way to describe G in terms of its series and parallel compositions.

More formally, T is a rooted tree whose nodes are of three types: S, P, and Q. Each node μ of T has an associated SP-digraph (possibly with multiple edges), called the *skeleton* of μ and denoted by $skeleton(\mu)$. Tree T is recursively defined according to the following cases:

Series case: Graph G is not a biconnected graph. Let $c_2, \ldots, c_k$ $(k \geq 1)$ be the cutvertices of G. Since G is planarly biconnectible, each cutvertex c_i $(i = 2, \ldots, k)$ is contained in exactly two biconnected components G_i and G_{i+1} such that s is in G_1 and t is in G_k. Each G_i is a SP-digraph with poles c_i, c_{i+1} $(i = 1, \ldots, k)$, where $c_1 = s$, and $c_{k+1} = t$. Then, the root of T is an S-node μ. Digraph $skeleton(\mu)$ consists of the chain $e_1, \ldots, e_k$, where $e_i = (c_i, c_{i+1})$ $(i = 1, \ldots, k)$. Node μ has children $\nu_1, \ldots, \nu_k$, where ν_i is the root of the SPQ-tree T_i of G_i $(i = 1, \ldots, k)$. Graph G_i is called the *pertinent digraph* of ν_i, and edge e_i is called the *virtual edge* of ν_i in $skeleton(\mu)$. (See Figure 1(a)).

Parallel case: Graph G is a biconnected graph with at least two split components with respect to the split pair $\{s,t\}$. Denote these split components as $G_1, \ldots, G_k$ ($k \geq 2$). Each G_i ($i = 1, \ldots, k$) is a SP-digraph with poles s and t. Then, the root of T is a P-node μ. Digraph $skeleton(\mu)$ consists of a bundle of parallel edges e_i from s to t ($i = 1, \ldots, k$). Node μ has children $\nu_1, \ldots, \nu_k$, where ν_i is the root of the SPQ-tree T_i of G_i ($i = 1, \ldots, k$). Graph G_i is called the *pertinent digraph* of ν_i, and edge e_i is called the *virtual edge* of ν_i in $skeleton(\mu)$. (See Figure 1(b)).

Trivial case: G consists of a single edge $e = (s, t)$. Then, T consists of a single Q-node μ. Graph $skeleton(\mu)$ is G itself. We also say that μ is *associated* with e.

The SPQ-tree T so obtained has a Q-node associated with each edge of G.

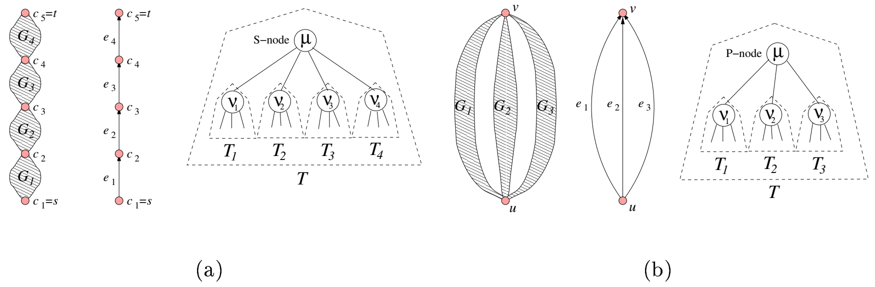


Figure 1: SPQ-tree definition: (a) Series case. (b) Parallel case.

A *restricted integer grid* is a grid ϕ, that is a proper subset of the integer plane or space [6]. In this paper we are interested in restricted integer grid consisting of k parallel lines passing trough the k-vertices of a polygon with k sides as regular as possible. Namely, the k lines should pass trough the k vertices of a regular k-gon, but they could not be, in general, grid lines. We choose the grid lines that better approximate the regular polygon. In particular in this paper we show two algorithm which compute drawings on grid with 4 and 8 lines, respectively. The grid with 4 lines, also called *box* in the following, is a $2 \times 2 \times \infty$ grid, *i.e.* it consists of four parallel lines of infinite length passing through the four vertices of a unit square[1]. The grid with 8 lines, also called 8-prism in the following, is a grid consisting of eight parallel lines of infinite length passing trough the vertices of the polygon shown in figure 2(a). An 8-prism is a $4 \times 4 \times \infty$ grid. We define also two other kind of restricted integer grid that will be used in the intermediate step of our algorithms. A *strip* is a $2 \times \infty$ planar grid, *i.e.* two parallel lines of infinite length. A *ingot* is a grid consisting of four parallel lines of infinite length passing trough the vertices of the polygon shown in figure 2(b). An ingot is a $3 \times 2 \times \infty$ grid. We assume that each grid is placed

[1] We adopt the convention of measuring the size of the grid by counting the number of grid lines used in each dimension rather than the distance between the maximal pair.

such that the lines are parallel to the x-axis, so that the position of each vertex in the drawing is uniquely defined when a line and an x-coordinate are assigned to it.

Figure 2: (a) definition of an 8-prism. (b) definition of an ingot

We say that a graph G is *box-drawable* if G admits a drawing such that every vertex is drawn at integer coordinates on one of the four lines of the box and the edges are drawn as straight-line segments such that no two edges intersect, except possibly at common end-points. The drawing of G on the box is a *box-drawing* of G. A graph is *box-forbidden* if it does not admit a box-drawing. Analogous definition can be given for the strip, the ingot and the 8-prism. Let G be a box-drawable graph; we say that G *covers* three lines if all box drawings of G have at least three mutually adjacent vertices drawn on different lines of the box. Given two lines l_a and l_b of a restricted integer grid, we denote by $\sigma_{a,b}$ the strip defined by l_a and l_b.

3 SP Box-forbidden graphs

In this section we construct a series-parallel graph that cannot be drawn on a box. We start by giving three crucial lemmas.

Lemma 1 *In a box-drawing, a vertex v drawn on a line l can be connected to at most two vertices drawn on the same line l.*

Lemma 2 *Let G_0 be the SP-graph of Figure 3(a) with poles s_0 and t_0. Then G_0 is box-drawable and in any box-drawing of G_0, s_0 and t_0 appear on different lines of the box.*

Proof: Graph G_0 is box-drawable as shown in Figure 3(b). Suppose that there exists a box-drawing of G_0 with s_0 and t_0 on the same line l. We adopt the notation of Figure 3(a). No other vertex c_i can be drawn on l since a 3-cycle cannot be drawn on a line. Therefore, since there are three lines different from l and five vertices different from s_0 and t_0, at least two vertices, say c_i and c_j, are on a same line l'. However, since both c_i and c_j are adjacent to s_0 and t_0, any box-drawing would necessarily have a crossing (see also Figure 3(c)) – a contradiction and therefore s_0 and t_0 must appear on different lines. $\square$

Lemma 3 *The SP-graph G_0 of Figure 3(a) covers three lines.*

Proof: By Lemma 2, vertices s_0 and t_0 are on different lines (call them l_s and l_t respectively). To prove the statement it suffices to prove that one vertex c_i different from s_0 and t_0 is on a third line. Suppose that all vertices c_i ($i = 1, \ldots, 5$) were either on line l_s or on line l_t. At least three of them would then be on the same line (suppose l_s), but by Lemma 1, this is not possible. $\square$

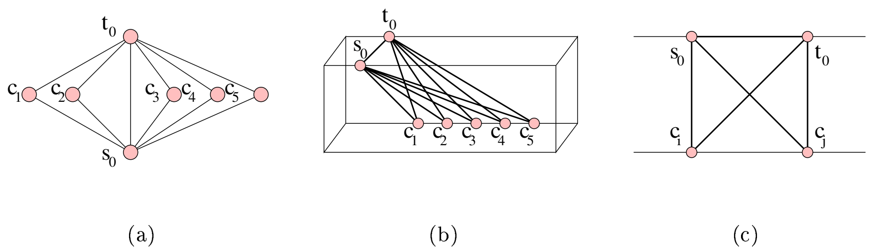


Figure 3: (a) Graph G_0. (b) A box-drawing of G_0. (c) Vertices s_0 and t_0 of G_0 cannot be drawn on the same line.

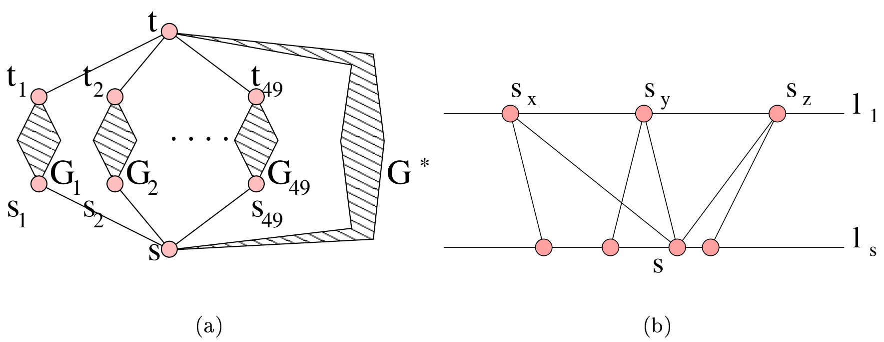


Figure 4: (a) Graph G of Theorem 1 (b) Vertex s cannot be drawn on a line covered by G_x, G_y and G_z.

We are now ready to prove the main result of the section.

Theorem 1 *There exist box-forbidden series-parallel graphs.*

Proof: Let G be the graph of Figure 4(a), consisting of 49 copies G_i ($i = 1, \ldots, 49$) of the graph G_0 depicted in Figure 3(a) and by another copy G^* of G_0 such that poles s and t of G^* are connected to poles s_i of G_i ($i = 1, \ldots, 49$) and to poles t_i of G_i ($i = 1, \ldots, 49$) respectively. We show that G is not box-drawable.

By Lemma 3, each G_i covers three lines of the box. Consider the maximum number N of graphs G_i that can be arranged on the four lines of the box so that there are not two graphs G_h and G_k ($h \neq k$, $1 \leq h, k \leq 49$) covering the same three lines and having s_h and s_k on the same line and t_h and t_k on the same line. Since the number of permutations of length p from n elements is $\frac{n!}{(n-p)!}$, then $N = \frac{4!}{(4-3)!} = 24$. The number of copies of G_i is $2 * N + 1 = 49$, so if a box-drawing Γ of G exists, then there would be at least three components G_x, G_y, G_z ($x \neq y \neq z$, $1 \leq x, y, z \leq 49$) such that:

- G_x, G_y, G_z cover the same three lines of the box (say l_1, l_2 and l_3);
- the sources s_x, s_y, and s_z are drawn on the same line of the box (say l_1); and
- the sinks t_x, t_y, and t_z are drawn on the same line of the box different from the one of the sources (say l_2).

Let l_s be the line of the source s of G. Line l_s can not be anyone of the lines l_1, l_2 and l_3. Namely, if $l_s = l_1$ then by Lemma 1 s cannot be connected to the three vertices s_x, s_y and s_z. If either $l_s = l_2$ or $l_s = l_3$, then edges (s, s_x), (s, s_y), and (s, s_z) lie on the strip $\sigma_{s,1}$. Since G_x, G_y, G_z cover l_1, l_2 and l_3, there are three edges on $\sigma_{s,1}$ incident on s_x, s_y and s_z. These three edges divide the strip in four regions, from anyone of which only, at most, two among s_x, s_y and s_z are visible. It follows that there is no room for drawing (s, s_x), (s, s_y), and (s, s_z) without creating a crossing (see Figure 4(b)). Therefore s must be drawn on a line different from l_1, l_2, and l_3. By a similar argument, t cannot be drawn on any of lines l_1, l_2, and l_3. However, s and t are the poles of G^* and by Lemma 2 they cannot be on the same line, thus establishing that no box-drawing of G exists. $\square$

4 Drawing Regular Series-Parallel Graphs on a 8-prism.

In this section we present an algorithm that can be used to draw regular series-parallel graphs on a 8-prism. We start by providing some more necessary definitions.

In the following G is a series-parallel regular graph. We assume that G is st-oriented and that edge (s, t) exists in G on the external face.

Let T be the SPQ-tree of G. Since G has always the edge (s, t) on the external face then the root of T is always a P-node. Also, since G is regular, for each S-node the leftmost and the rightmost children are both Q-nodes, *i.e.* the first and the last component of each series composition are single edges, as shown in the following lemma.

Lemma 4 *Let G be a regular SP-digraph whose decomposition tree is T. The leftmost and the rightmost children of each S-node μ of T are Q-nodes.*

Proof: Let ν_l and ν_r be the leftmost and the rightmost child of μ respectively. If they are not Q-nodes, then they are P-nodes. Consider now the father λ of μ: it is a P-node, and its poles are the source pole of ν_l and the sink pole of ν_r. It follows that this poles have depth at least two, and this is not possible in a regular SP-digraph. □

We say that an S-node μ of T is *simple* if all its children are Q-nodes, *i.e.* if its pertinent graph is a simple path; a P-node is *simple* if its children are either simple S-node or Q-node, *i.e.* if its pertinent graph is a parallel of simple paths. Also, we say that a graph G' is *simple* if its decomposition tree has all simple nodes.

A *level-numbering* of T is a numbering of each node μ of T defined as follows:

- if μ is a P-node then:
  - if $(\mu = root(T))$
    then $level(\mu) := 0$
    else $level(\mu) := level(father(\mu))$
- if μ is an S-node then:
  - if (μ is simple)
    then $level(\mu) := level(father(\mu))$
    else $level(\mu) := level(father(\mu)) + 1$
- if μ is a Q-node then:
  - $level(\mu) := level(father(\mu))$

Using the level-numbering defined above we assign a level to each vertex v of G. If v is a pole of only one P-node μ then $level(v) = level(\mu)$; if v is a pole of more than one P-node, it is a pole of two P-nodes μ' and μ'' which are consecutive children of an S-node; in this case $level(v) = level(\mu') = level(\mu'')$. If v is not a pole of some P-node then v is shared by two edges whose Q-nodes have the same level number and $level(v)$ is the level of these Q-nodes. We call *jumping edges* those edges whose endvertices are assigned to different level.

Let μ be a non-simple S-node whose decomposition tree is G_μ. We call *virtual pertinent graph* of μ the graph G^*_μ obtained from G_μ in the following way:

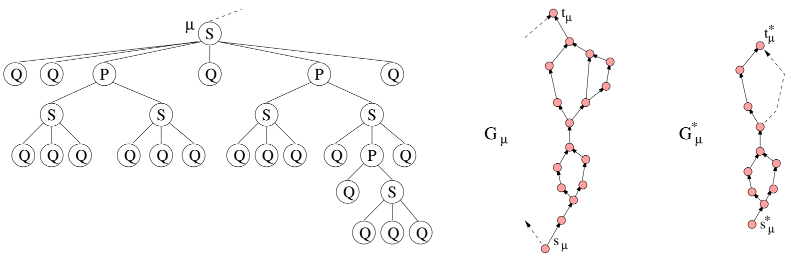
- remove the two edges associated with the rightmost and leftmost children of μ;
- for each child ν of μ which is a P-node:

– if any children λ of ν is non-simple, then replace its pertinent graph G_λ with a virtual edge between the poles of ν;

– if some multiple edges have been created, replace them with a single edge

The vertices of the virtual pertinent graph G^*_μ of μ are all the vertices of the pertinent graph G_μ of μ assigned to level $j = level(\mu)$. Namely the poles of μ are also pole of the P-node which is the father of μ, and thus they are assigned to level $j-1$. Also, for each child of μ which is a P-node we replace by a virtual edge, any component of the parallel which is not a simple path, *i.e.* any component whose vertices are assigned to level larger then j.

The virtual pertinent graph G^*_μ is an SP-digraph whose poles are the vertices adjacent to the poles of G_μ; we call *source core pole* the vertex adjacent to the source of G_μ and *sink core pole* the vertex adjacent to the sink of G_μ. We remark that the edges connecting the poles of G_μ with the poles of G^*_μ are jumping edges since they connect vertices of different levels. Figure 5 illustrates the definition above.



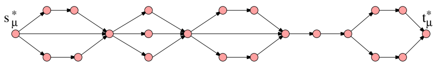
(a) (b)

Figure 5: (a) A subtree rooted at a non-simple S-node μ; (b) Its pertinent graph G_μ and its virtual pertinent core graph G^*_μ.

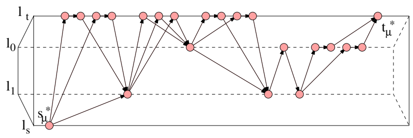
Let μ_i $(i = 1, \ldots, n_j)$ be the non-simple S-nodes having $level(\mu_i) = j$ ordered according to their left-to-right order in T. We define $L(j) = \{G^*_{\mu_1}, \ldots, G^*_{\mu_{n_j}}\}$ as the ordered set of the virtual pertinent graphs of all non-simple S-node of level j. Also, we define $L(0)$ as the set consisting of the only edge (s,t). In the special case in which the root of T is simple (i.e. the graph G is the parallel of simple paths) we define the only set $L(0) = \{G\}$. Each set $L(j)$ contains all the vertices of G whose level number is j, and all the edges connecting vertices whose level number is j. Thus the sets $L(j)$ $(j = 0, \ldots, n_{max}$, where n_{max} is the maximum number used in the level numbering) contain all the vertices of G, and all the edges of G except the jumping edges, which connect vertices having different level numbers.

We start the description of the algorithm by illustrating how a virtual pertinent graph G^*_μ of a non-simple S-node μ can be drawn on an ingot.

Let l_s, l_t, l_0 and l_1 be the four lines of an ingot arraged as shown in Figure 6.



(a)



(b)

Figure 6: (a) A virtual pertinent core graph G^*_μ. (b) An ingot-drawing of it.

Procedure *layoutVirtualRegular*

Input A virtual pertinent graph G^*_μ and a starting x-coordinate x_s;

Output A drawing of G^*_μ on the ingot defined by lines l_s, l_t, l_0, l_1;

Let ν_0, ν_1, ..., ν_h, ν_{h+1} be the children of the S-node μ; ν_0 and ν_{h+1} are Q-nodes which are associated to the jumping edges not belonging to G^*_μ.

for ($j = 1$ to h)

{

if ($j = 1$)

{

$l(source(\nu_1)) = l_s$; $x(source(\nu_1)) = x_s$; $i = 1$;

if (ν_1 is a P-node)

{

let $\pi_g = \{v_{g,1}, \ldots, v_{g,n_g}\}$ ($n_g \geq 0$, $g = 1, \ldots, l$) the l paths in the parallel;

for ($g = 1$ to l)

for ($y = 1$ to n_g)

{ $l(v_{g,y}) = l_t$; $x(v_{g,y}) = x_s + i$; $i = i + 1$; }

}

$l(sink(\nu_1)) = l_1$; $x(sink(\nu_1)) = x_s + i$; $i = i + 1$;

}

if ($1 < j < h$)

{

```
        if (ν_j is a P-node)
        {
            let π_g = {v_{g,1}, ..., v_{g,n_g}} (n_g ≥ 0, g = 1, ..., l) the l paths in
               the parallel;
            for (g = 1 to l)
                for (y = 1 to n_g)
                 { l(v_{g,y}) = l_t; x(v_{g,y}) = x_s + i; i = i + 1; }
        }
        x = j  mod 2;
        l(sink(ν_1)) = l_x; x(sink(ν_1)) = x_s + i; i = i + 1;
    }
    if (j = h)
    {
        if (ν_j is a P-node)
        {
            let π_g = {v_{g,1}, ..., v_{g,n_g}} (n_g ≥ 0, g = 1, ..., l) the l paths in
               the parallel;
            for (g = 1 to l)
                for (y = 1 to n_g)
                {
                  x = h  mod 2;
                  l(v_{g,y}) = l_x; x(v_{g,y}) = x_s + i; i = i + 1;
                }
        }
        l(sink(ν_h)) = l_t; x(sink(ν_1)) = x_s + i;
    }
}
```

Lemma 5 *Let G be a regular SP-digraph, with decomposition tree T, and let μ be a non-simple S-node of T with virtual pertinent graph G^*_μ. Then the ingot-drawing computed by Procedure* layoutVirtualRegular *is such that:*

- *the source core pole s^*_μ and the sink core pole t^*_μ of G^*_μ are on different lines;*
- *for each child ν of μ such that ν is a P-node, $source(\nu)$ and $sink(\nu)$ are on lines different from those of s^*_μ and t^*_μ;*
- *no two edges cross each other.*

Proof: The source core pole s^*_μ coincides with $source(\nu_1)$ which is assigned to line l_s; the sink core pole t^*_μ coincides with $sink(\nu_h)$ which is assigned to line l_t; thus property one of the statement hold. The source of a P-node ν_j coincides with the sink of the node ν_{j-1} preceding ν_j; except for s^*_μ and t^*_μ, all the poles are assigned either to line l_0 or to line l_1. Thus they are on lines different from those of s^*_μ and t^*_μ.

We now prove that the drawing obtained is crossing-free. Suppose that two edges $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ cross each other. A crossing is possible only if at least one of the endvertices of e_2 have x-coordinate between $x(u_1)$ and $x(v_1)$ or viceversa.

There are three types of crossings:

crossing on a line (i.e. overlapping); The only egdes having both endvertices on the same line are the edges of the paths which forms a parallel. The endvertices of these edges have consecutive x-coordinate, thus a crossing of this type is not possible.

crossing between edges laying on a strip defined by two lines; without loss of generality, suppose that e_1 has endvertices with non-consecutive x-coordinates. At least one of the endvertices of e_1 is a pole. We consider only the case in which u_1 is a source pole, the other cases are analogous. We denote by t_1 the sink pole of the parallel having u_1 as source pole. The vertex v_1 can be either the first vertex of any path π_g $(1 < g \leq 1)$, or it may coincide with t_1. If $v_1 \equiv t_1$ then the vertices having x-coordinates between $x(u_1)$ and $x(v_1)$ are the vertices of the paths in parallel between u_1 and v_1; no one of this vertex is either on the line of u_1 or on the line of v_1, thus edge e_2 cannot have an endvertex with x-coordinate between u_1 and v_1 and a crossing is not possible. If $v_1 \not\equiv t_1$ then the vertices having x-coordinates between $x(u_1)$ and $x(v_1)$ are a subset of the vertices of the paths in parallel between u_1 and t_1; these vertices are on the same line of v_1. The edges having one of this vertices as an end vertex, and laying on the same strip as e_1, have u_1 as the other endvertex. The edge e_2 must be one of these edges. Edges e_1 and e_2 cannot cross because they have a common endvertex.

crossing between edges laying on two different strips; in order to have a crossing of this type, edges e_1 and e_2 must lay on two different strips which cross each other along a straight line which is not one of the four lines l_s, l_t, l_0, and l_1. The only two strips which cross in such a way are the strip $\sigma_{s,0}$, and the strip $\sigma_{t,1}$. On the strip $\sigma_{s,0}$ we have no edges. Namely, the only vertex on line l_s is s^*_μ and it can be connected either to vertices on line l_t or to a vertex on line l_1. It follows that a crossing is not possible.

□

Procedure *layoutVirtualRegular* describes how to draw a single virtual pertinent graph of a set $L(j)$ $(j = 1, \ldots, n_{max})$. The drawing of the whole set $L(j)$ can be obtained by two different procedure described in the following.

Procedure *layoutVirtualSetRegularCongruent*

Input A set $L(j) = \{G_{\mu_1}, \ldots, G_{\mu_{n_j}}\}$ of the virtual pertinent graphs of the non-simple S-node of level j;

Output A drawing of $L(j)$ on on a ingot defined by lines l_s, l_t, l_0, l_1;

$x_l = 0$;

for $(i = 1$ to $n_j)$

layoutVirtualRegular(G_{μ_i},x_l);

set x_l equal to the maximum x-coordinate used in the drawing of G_{μ_i};

let $v_{x,1}, \ldots, v_{x,n_x}$ be the vertices on line l_x $(x = s, t, 1, 0)$;

for $(g = 1$ to $n_1)$
$x(v_{1,g}) = g$;

for $(g = 1$ to $n_0)$
$x(v_{0,g}) = n_1 + g$;

for $(g = 1$ to $n_s)$
$x(v_{s,g}) = n_1 + n_0 + g$;

for $(g = 1$ to $n_t)$
$x(v_{t,g}) = n_1 + n_0 + n_s + g$;

Procedure *layoutVirtualSetRegularNonCongruent*

Input A set $L(j) = \{G_{\mu_1}, \ldots, G_{\mu_{n_j}}\}$ of the virtual pertinent graphs of the non-simple S-node of level j;

Output A drawing of $L(j)$ on a ingot defined by lines l_s, l_t, l_0, l_1;

for $(i = 1$ to $n_j)$

layoutVirtualRegular(G_{μ_i},x_l);

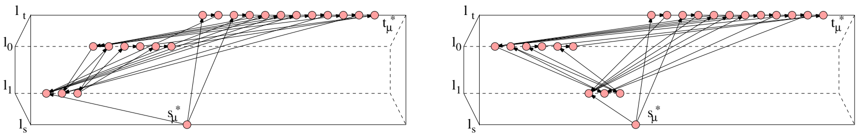
set x_l equal to the maximum x-coordinate used in the drawing of G_{μ_i};

let $v_{x,1}, \ldots, v_{x,n_x}$ be the vertices on line l_x $(x = s, t, 1, 0)$;

for $(g = 1$ to $n_0)$
$x(v_{0,g}) = g$;

for $(g = 1$ to $n_1)$
$x(v_{1,g}) = n_0 + g$;

for $(g = 1$ to $n_s)$
$x(v_{s,g}) = n_0 + n_1 + g$;



(a) (b)

Figure 7: (a) A congruent drawing of the graph of figure 6. (b) A non congruent drawing of the same graph.

for $(g = 1$ to $n_t)$
  $x(v_{t,g}) = n_0 + n_1 + n_s + g;$

A congruent and a non-congruent drawing of the graph of figure 6 is shown in figure 7.

The drawing produced by Procedure *layoutVirtualSetRegularCongruent* are such that all vertices on line l_1 precede all vertices on line l_0, which precede all vertices on line l_s, which precede all vertices on line l_t. The drawing produced by Procedure *layoutVirtualSetRegularNonCongruent* are such that all vertices on line l_0 precede all vertices on line l_1, which precede all vertices on line l_s, which precede all vertices on line l_t. We call *congruent* a drawing produced by Procedure *layoutVirtualSetRegularCongruent* and *non-congruent* a drawing produced by Procedure *layoutVirtualSetRegularNonCongruent*. Also, we refer to l_s as the *sources line* and to l_t as the *sinks line*.

Lemma 6 *Let G be a regular SP-digraph, with decomposition tree T, and let $L(j) = \{G_{\mu_1}, \dots, G_{\mu_{n_j}}\}$ be a set of virtual pertinent graphs of the non-simple S-node of level j. Then the ingot-drawing computed by Procedure* layoutVirtualRegularCongruent *and by Procedure* layoutVirtualRegularNonCongruent *are crossing free.*

Proof: Both the procedure has two main step. In the first one a drawing is computed by Procedure *layoutVirtualRegular* for each virtual pertinent graph in $L(j)$, and all these drawings are consecutively arranged on the four lines. In the second step vertices are shifted. The drawing produced by the first step is clearly crossing free. Namely, the drawing of each virtual pertinent graph is crossing free by Lemma 5, and two different drawings cannot cross since they are disjoint. We prove that the shifting of the vertices does not introduce a crossing. Consider a single strip defined by two lines. The shifting of the vertices on this two lines does not change the relative position of the vertices. Thus the shifting changes the slopes of the segments on this strip but it does not introduce a crossing. It follows that a crossing may occur only between edges laying on two different strips which cross each other along a line which is not one of l_s, l_t, l_0, and l_1. As pointed out in the proof of Lemma 5 there are only two such planes, but on one of them there are no edges. Thus a crossing cannot occur either in this case. □

We conclude the description of the algorithm by describing how to arrange the strip-drawing of each $L(j)$ $(j = 0, \ldots, n_{max})$ on a 8-prism in the Step 5 of our main algorithm. the eight lines L_0, L_1, L_2, L_3, L_4, L_5, L_6, and L_7 are arranged as shown in Figure 8

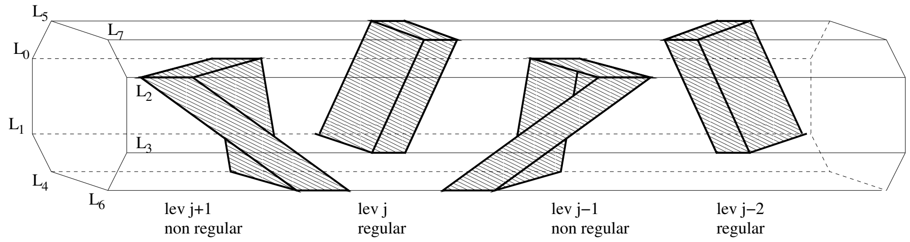


Figure 8: The 8 lines arrangement.

Procedure *arrangeDrawingRegular*

Input The sets $L(j) = \{G_{\mu_1}, \ldots, G_{\mu_{n_j}}\}$ $(j = 0, \ldots, n_{max})$ of the virtual pertinent graphs of the non-simple S-node of level j;

Output An arrangement of the drawings on a 8-prism defined by lines L_0, L_1, L_2, L_3, L_4, L_5, L_6, and L_7;

```
for (j = n_max to 0)
{
    a = j mod 4; b = (j + 2) mod 4; c = a + 4; d = b + 4;
    l_s=L_a; l_t = L_c; l_1 = L_b; l_0 = L_d;
    if (j is even)
    {
        layoutVirtualRegularNonCongruent(L(j));
    }
    else
    {
        layoutVirtualRegularCongruent(L(j));
    }
}
```

The procedure above arranges the drawings of the sets $L(j)$ ($j = 0, \ldots, n_{max}$) so that drawings of consecutive sets are alternatively on ingot P defined by lines L_0, L_2, L_4, and L_6, and on ingot P' defined by lines L_1, L_3, L_5, and L_7. Consecutive drawings on the prism P (respectively P') have sources line and sinks line alternatively on line L_0 (respectively L_1) and L_2 (respectively L_3). Finally the drawings on prism P are non-congruent, while drawings on prism P' are congruent.

In order to prove that the arrangement computed by Procedure *arrange-DrawingRegular* is such that the jumping edges can be added without creating a crossings, we need the following technical lemma.

Lemma 7 *Let l_a, l_b, l_c and l_d be four lines such that the strips $\sigma_{a,b}$ and $\sigma_{c,d}$ are parallel and the strips $\sigma_{a,c}$ and $\sigma_{b,d}$ cross each other. Let $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be two edges whose vertices u_1, u_2, v_1 and v_2 are drawn on lines l_a, l_b, l_c and l_d respectively. If $x(u_1) < x(u_2)$ and $x(v_1) < x(v_2)$ then e_1 and e_2 do not cross each other.*

Proof: Without loss of generality we can assume that the strips $\sigma_{a,b}$ and $\sigma_{c,d}$ are orthogonal to the Y-axis, *i.e.* lines l_a and l_b have the same y-coordinate y_1 and lines l_c and l_d have the same y-coordinate y_2. If e_1 and e_2 crossed each other, then there would exist a plane containing both e_1 and e_2, *i.e.* the four points representing u_1, u_2, v_1, and v_2 would be co-planar. The co-planarity condition of four vertices is expressed by the following equation:

$$\begin{vmatrix} 1 & 1 & 1 & 1 \\ x(u_1) & x(u_2) & x(v_1) & x(v_2) \\ y(u_1) & y(u_2) & y(v_1) & y(v_2) \\ z(u_1) & z(u_2) & z(v_1) & z(v_2) \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 & 1 \\ x(u_1) & x(u_2) & x(v_1) & x(v_2) \\ y_1 & y_1 & y_2 & y_2 \\ z_a & z_b & z_c & z_d \end{vmatrix} = 0 \quad (1)$$

where z_a, z_b, z_c and z_d are the z-coordanates of line l_a, l_b, l_c and l_d respectively. The above equation solves to:

$$(x(u_2) - x(u_1))(z_d - z_c) = (x(v_2) - x(v_1))(z_b - z_a)$$

Since lines l_a and l_b have the same y-coordinate they cannot have the same z-coordinate; thus $(z_b - z_a) \neq 0$. Analogously we have $(z_d - z_c) \neq 0$. Since $\sigma_{a,c}$ and $\sigma_{b,d}$ cross each other then $(z_b - z_a)$ and $(z_d - z_c)$ have opposite sign. By hypothesis $(x(u_2) - x(u_1)) > 0$ and $(x(v_2) - x(v_1)) > 0$; it follows that equation 1 does not admit a solution, *i.e.* e_1 and e_2 do not cross each other. □

The following lemma proves that the drawing obtained by Procedure *arrangeDrawingRegular* is a valid 8-prism-drawing.

Lemma 8 *Let G be a regular SP-digraph, with decomposition tree T, and let $\Gamma_1, \ldots, \Gamma_{n_{max}}$ the drawings of the sets of virtual pertinent graphs $L(1), \ldots L(n_{max})$ computed and arranged by Procedure* arrangeDrawingRegular. *The drawing obtained when also the jumping edges are added is a valid 8-prism-drawing.*

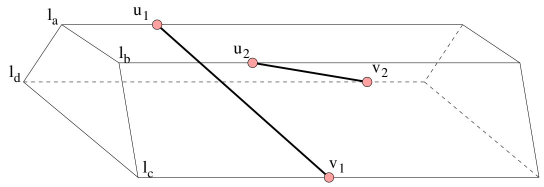


Figure 9: Illustration of Lemma 7.

Proof:

The drawing of a whole set $L(j)$ ($0 \leq j \leq n_{max}$) computed either by *layoutVirtualSetRegularCongruent* or by *layoutVirtualSetRegularNonCongruent* is crossing free by Lemma 6.

Now we show that the jumping edges can be added without creating crossings. The jumping edges connect poles (core poles or not) of level j to core poles of level $j+1$. First of all, we show that jumping edges on the same strip defined by two lines cannot cross each other. Namely, in each drawing of a set $L(j)$ the virtual pertinent graphs are ordered according to the left-to right order of the corresponding S-nodes in T; consider two jumping edges connecting, for example, two different source poles s_1 and s_2 of level j to two different source core poles c_1 and c_2 of level $j+1$. Suppose without loss of generality that s_1 precedes s_2, *i.e.* the S-node μ_1 whose pertinent graph contains s_1 is to the left of the S-node μ_2 whose pertinent graph contains s_2. The S-node ν_1 whose pertinent graph contains c_1 is in the subtree of T rooted at μ_1 and the S-node ν_2 whose pertinent graph contains c_2 is in the subtree of T rooted at μ_2. It follows that c_1 preceds c_2 and that the two jumping edges cannot cross each other.

To show that a crossing is not possible also between jumping edges on different strip, we consider different cases.

jumping edges connecting core poles to core poles; recall that the core poles are on line L_0 and L_2 on prism P and on line L_1 and L_3 on prism P'. Consider the drawing Γ_j of the vertices of level j and assume without loss of generality that the sources line is on line L_0 and the sinks line is on line L_2. The Procedure *arrangeDrawingRegular* is such that sources line and sinks line of level $j-1$ are on line L_3 and L_1, respectively; also sources line and sinks line of level $j+1$ are on line L_1 and L_3, respectively. Thus jumping edges between level j and $j+1$ are on the strip $\sigma_{0,1}$ and on the strip $\sigma_{2,3}$. Since $\sigma_{0,1}$ and $\sigma_{2,3}$ are parallel the jumping edges on $\sigma_{0,1}$ cannot cross jumping edges on $\sigma_{2,3}$. The jumping edges between level $j-1$ and j are on the strip $\sigma_{0,3}$ and on the strip $\sigma_{2,1}$. Suppose that two edges $e_1 = (u_1, v_1)$ on strip $\sigma_{0,3}$, and $e_2 = (u_2, v_2)$ on strip $\sigma_{2,1}$ cross each other. Without loss of generality we may assume that u_1 is on line L_0, v_1 is on line L_3, u_2 is on line L_2 and v_2 is on line L_1. It follows that u_1 is a source core pole of level j, v_1 is a source core pole of level $j-1$, u_2 is a sink core pole of level j and v_2 is a sink core pole of level $j-1$. In both the congruent and non-congruent drawing of a set $L(j)$ all source poles have x-coordinates smaller than all sink poles. Thus $x(u_1) < x(v_1) < x(u_2) < x(v_2)$ and the edges cannot cross by Lemma 7.

jumping edges connecting poles to core poles; recall that the source poles are on line L_4 and L_6 on prism P and on line L_5 and L_7 on prism P'. Moreover, a source pole of level j can be connected to both source core poles and sink core poles of level $j+1$. Consider the drawing Γ_j of $L(j)$ and the drawing Γ_{j+1} of $L(j+1)$. We distinguish two case:

j **is even;** assume without loss of generality that the sources line is on line L_0 and the sinks line is on line L_2. The poles of Γ_j which are not core poles are on line L_4 and L_6. The core poles of level $j+1$ are on line L_1 and L_3. It follows that the jumping edges are on the strips $\sigma_{3,6}$, $\sigma_{6,1}$, $\sigma_{3,4}$ and $\sigma_{4,1}$. Edges on strip $\sigma_{3,6}$ and $\sigma_{4,1}$ do not cross each other since they are disjoint. Edges on strip $\sigma_{3,6}$ do not cross edges on strips $\sigma_{6,1}$ and $\sigma_{3,4}$ since they cross only at common line. Analogously edges on $\sigma_{4,1}$ do not cross edges on strips $\sigma_{6,1}$ and $\sigma_{3,4}$. Thus the only edges that may cross are those on strip $\sigma_{6,1}$ and $\sigma_{3,4}$. Consider two edges $e_1 = (u_1, v_1)$ on strip $\sigma_{6,1}$ and $e_2 = (u_2, v_2)$ on strip $\sigma_{3,4}$. Without loss of generality suppose that u_1 is on line L_6, v_1 is on line L_1, u_2 is on line L_4 and v_2 is on line L_3. In this case u_1 and u_2 are poles of level j, v_1 is a source core pole of level $j+1$ and v_2 is a sink core poles of level $j+1$. The drawing Γ_j is a non-congruent drawing, therefore vertices on line L_4 follows vertices on line L_6. Thus we have $x(u_1) < x(u_2) < x(v_1) < x(v_2)$ and a crossing is not possible by Lemma 7.

j **is odd;** assume without loss of generality that the sources line is on line L_1 and the sinks line is on line L_3. The poles of Γ_j which are not core poles are on line L_5 and L_7. The core poles of level $j+1$ are on line L_0 and L_2. It follows that the jumping edges are on the strips $\sigma_{2,7}$, $\sigma_{7,0}$, $\sigma_{2,5}$ and $\sigma_{5,0}$. Edges on strip $\sigma_{2,7}$ and $\sigma_{5,0}$ do not cross each other since they are disjoint. Edges on strip $\sigma_{2,7}$ do not cross edges on strips $\sigma_{7,0}$ and $\sigma_{2,5}$ since they cross only at common line. Analogously edges on $\sigma_{5,0}$ do not cross edges on strips $\sigma_{7,0}$ and $\sigma_{2,5}$. Thus the only edges that may cross are those on strip $\sigma_{7,0}$ and $\sigma_{2,5}$. Consider two edges $e_1 = (u_1, v_1)$ on strip $\sigma_{7,0}$ and $e_2 = (u_2, v_2)$ on strip $\sigma_{2,5}$. Without loss of generality suppose that u_1 is on line L_7, v_1 is on line L_0, u_2 is on line L_5 and v_2 is on line L_2. In this case u_1 and u_2 are poles of level j, v_1 is a sink core pole of level $j+1$ and v_2 is a source core poles of level $j+1$. The drawing Γ_j is a congruent drawing, therefore vertices on line L_7 follows vertices on line L_5. Thus we have $x(u_2) < x(u_1) < x(v_2) < x(v_1)$ and a crossing is not possible by Lemma 7.

$\square$

The complete algorithm for computing a box-drawing of a regular series-parallel graph is shown in the following.

Algorithm *8PrismEmbedder*

Input A regular series-parallel graph G;

Output A drawing of G on a 8-prism;

Compute the SPQ-tree T of G;

Compute a level numbering of T and let n_{max} be the maximum level number assigned;

Compute the sets $L(j)$ $(1 \leq j \leq n_{max})$;

layoutVirtualSetRegular;

arrangeDrawingRegular;

Add the jumping edges.

The results of this section can be summarized as follows.

Theorem 2 *Let G be a regular series-parallel graph with n vertices. The algorithm* prismEmbedderRegular *computes a valid 8-prism-drawing of G in $O(n)$-time and with $O(n)$ volume.*

Proof:

Th drawing computed by *prismEmbedderRegular* is a valid 8 lines drawing by Lemma 6 and Lemma 8. Since a decomposition tree T of G can be constructed in $O(n)$ time [3, 7] and both the level numbering and the drawing strategy can be easily accomplished by an $O(n)$-time visit of T, the statement about the time complexity holds. Also, the volume of the computed drawing is $16 \times n$ since the n vertices are drawn at consecutive x-coordinates and the eight lines are on a grid 4×4.

□

5 Drawing 3-planar Series-Parallel Graphs on a box.

In this section we describe an algorithm, very similar to the one presented in Section 4, for computing a box-drawing of a 3-planar SP-digraph.

We start by remarking the following property of 3-planar SP-digraphs.

Lemma 9 *Let G be a 3-planar SP-digraph. The poles of every parallel component of G have depth 1 except at most the poles of G itself which can have depth at most 2.*

Proof: Let p be a pole of a parallel component of G, and let T the decomposition tree of G. Let μ be the deepest P-node of T having p as a pole. The pertinent graph of μ consists of at least two component in parallel. Thus at least two edges are incident on p in G_μ. If μ is the root of T then p has depth 1. If μ is not the root of T, let ν be the S-node which is the father of μ. If μ is not the leftmost or the rightmost child of ν then p is not the pole of ν, and thus it is not the pole of any other parallel component, thus having depth 1. If μ is the leftmost or rightmost child of ν then p is a pole also of ν. The father λ of ν is a P-node, and its pertinent graph is the parallel of the G_ν and of at least another component. Thus there is at least another edge incident on p. If λ is the root of T then p has depth 2 since it is the pole of two different parallel component (G_μ and $G_\lambda \equiv G$) which are not in series. If λ is not the root of T, let ξ be the S-node which is father of λ. If λ is not the rightmost or the leftmost child of ξ then p should be a pole of another child λ' of ξ, but this is not possible because otherwise there would be at least four edges incident on p. If λ is the rightmost or the leftmost child of ξ, then p is a pole also of ξ. The father θ of ξ is a P-node, and its pertinent graph is the parallel of the G_ξ and of at least another component. But this is not possible because otherwise there would be at least four edges incident on p.

The above argument proves that given a pole p of G it cannot have depth greater than 2, and that p can have depth equal to two, only if it is a pole of G itself.

□

The Lemma above shows that the 3-planar SP-digraphs are "almost" regular. Namely the property of "regularity" may not hold only for the poles of G. Based on this observation, we describe the algorithm for computing a box-drawing of a 3-planar SP-digraph, by considering first the case of regular 3-planar SP-digraphs. The algorithm for these graphs can be easily derived from the one for general regular SP-digraphs. At the end of the section we describe how to modify it in the case of 3-planar SP-digraph having poles of depth 2.

If a regular series-parallel graph has vertex degree at most three the following properties hold for its SPQ-tree:

- each P-node which is not the root of T, has exactly two children, one of which is an S-node;
- each child of an S-node which is a P-node is preceded and followed by a Q-node;

From this properties it follows that the virtual pertinent graph G^*_μ of a non-simple S-node is an alternating sequence of simple paths and parallel of a pair of simple paths.

The algorithm for drawing regular 3-planar SP-digraph on a box can be straightforwardly derived from the one for general regular SP-digraph. In particular the virtual pertinent graphs in the set $L(j)$ ($j = 1, \ldots, n_{max}$) are drawn on a strip. Let l_a and l_b the two lines defining the strip. We denote by $l(v)$ the line assigned to vertex v, and by $x(v)$ the x-coordinate assigned to v.

Procedure *layoutVirtual3planar*

Input A virtual pertinent graph G^*_μ and a starting x-coordinate x_s;

Output A drawing of G^*_μ on the strip $\sigma_{a,b}$;

Let ν_0, ν_1, $\ldots$, ν_h, ν_{h+1} be the children of the S-node μ; ν_0 and ν_{h+1} are Q-nodes which are associated to the jumping edges not belonging to G^*_μ.

for ($j = 1$ to h)

{

  if ($j = 1$)

  { $l(source(\nu_1)) = l_a$; $x(source(\nu_1)) = x_s$; $i = 1$;}

  if(ν_j is a P-node)

  {

    let $\pi_1 = \{v_1, \ldots, v_{n_1}\}$ ($n_1 \geq 0$) and $\pi_2 = \{u_1, \ldots, u_{n_2}\}$ ($n_2 \geq 0$) be the two paths in the parallel;

    for ($g = 1$ to n_1)
      { $l(v_g) = l_a$; $x(v_g) = x_s + i$; $i = i + 1$;}

    for ($g = 1$ to n_2)
      { $l(u_g) = l_b$; $x(u_g) = x_s + i$; $i = i + 1$;}

  }

  if(ν_j is a Q-node and $j \neq h$)
    $l(sink(\nu_j)) = l_a$; $x(sink(\nu_j)) = x_s + i$; $i = i + 1$;

  else

  $l(sink(\nu_j)) = l_b$; $x(sink(\nu_j)) = x_s + i$; $i = i + 1$

}

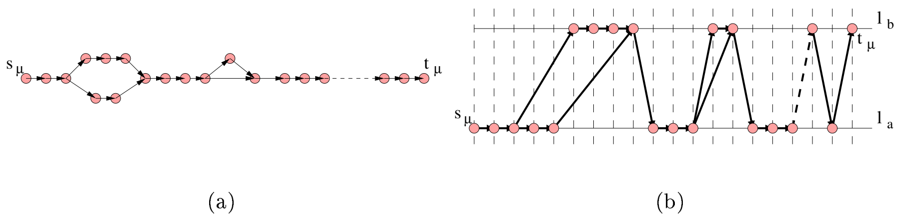


Figure 10: (a) A virtual pertinent core graph G^*_μ. (b) A strip drawing of it.

Lemma 10 *Let G be a 3-planar regular SP-digraph, with decomposition tree T, and let μ be a non-simple S-node of T with virtual pertinent graph G^*_μ. Then the strip drawing computed by Procedure* layoutVirtual3planar *is such that:*

- *the source core pole s^*_μ and the sink core pole t^*_μ of G^*_μ are on different lines;*
- *for each child ν of μ such that ν is a P-node, $source(\nu)$ is on the same line as s^*_μ and $sink(\nu)$ is on the same line as t^*_μ;*
- *no two edges cross each other.*

Proof: The source core pole s^*_μ coincides with $source(\nu_1)$ which is assigned to line l_a; the sink core pole t^*_μ coincides with $sink(\nu_h)$ which is assigned to line l_b; thus property one of the statement holds. Since each P-node is preceded by a Q-node, the source of a P-node ν_j coincides with the sink of the Q-node ν_{j-1} preceding ν_j; since $j-1 \neq h$ then the sink of ν_{j-1} is assigned to line l_a and therefore the source of ν_j is on the same line as s^*_μ. The sink of each P-node is assigned to line l_b, i.e. the same line as t^*_μ.

We now prove that the drawing obtained is crossing-free. Suppose that two edges $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ cross each other. A crossing is possible only if, at least one of the endvertices of e_2 has x-coordinates between $x(u_1)$ and $x(v_1)$ or viceversa. Thus, at least one among e_1 and e_2 has endvertices having not consecutive x-coordinates (without loss of generality asumme it is e_1). This is possible only if, at least one of the endvertices of e_1 is a pole. We consider only the case in which u_1 is a source pole, the other cases are analogous. We denote by t_1 the sink pole of the parallel having u_1 as source pole. The vertex v_1 can be either the first vertex of path π_2, or, in the case $n_2 = 0$, it coincides with t_1. In both cases u_1 and v_1 are on different lines. It follows that, in order to cross e_1, e_2 must have vertices on different lines. The only edge having an endvertex with x-coordinates between $x(u_1)$ and $x(v_1)$, and endvertices on different line, is the edge connecting the last vertex of the path π_1 to t_1. If $v_1 \equiv t_1$ then e_1 and e_2 cannot cross since they have a common endvertex; if $v_1 \not\equiv t_1$ then u_1 and u_2 are on the same line, v_1 and v_2 are on the same line and $x(u_1) < x(u_2) < x(v_1) < x(v_2)$. Also in this case a crossing is not possible. $\square$

Procedure *layoutVirtual3planar* describes how to draw a single virtual pertinent graph of a set $L(j)$ $(j = 1, \ldots, n_{max})$. The drawing of the whole set $L(j)$ can be obtained by the following Procedure.

Procedure *layoutVirtualSet3planar*

Input A set $L(j) = \{G_{\mu_1}, \ldots, G_{\mu_{n_j}}\}$ of the virtual pertinent graphs of the non-simple S-node of level j;

Output A drawing of $L(j)$ on the strip $\sigma_{a,b}$;

$x_l = 0$;

for $(i = 1$ to $n_j)$

{

layoutVirtual3Planar(G_{μ_i},x_l);

set x_l equal to the maximum x-coordinate used in the drawing of G_{μ_i};

}

We remark that the Procedure *layoutVirtualSet3planar* produce a drawing in which: the source (core) poles of all the virtual pertinent graph in the same set $L(j)$, are on the same line l_a, and the sink (core) poles are on the same line l_b. In the following we refer to l_a as *sources line* and to l_b as *sinks line.* further we denote by Γ_j the strip drawing of the set $L(j)$.

We conclude the description of the algorithm by describing how to arrange the strip-drawing of each $L(j)$ $(j = 0, \ldots, n_{max})$ on the box. The four lines l_0, l_1, l_2 and l_3 of the box are arranged as shown in Figure 11

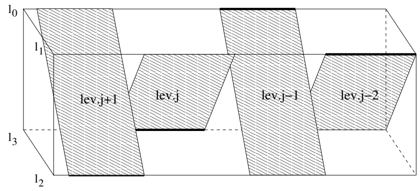


Figure 11: The arrangement of the strip drawings of different levels on the box. The bold lines indicate the sources line of each drawing.

Procedure *arrangeStripDrawing3planar*

Input A set of strip drawings $\Gamma_1, \ldots, \Gamma_{n_{max}}$ of the sets $L(1), \ldots L(n_{max})$;

Output An arrangement of the drawings on the box defined by lines l_0, l_1, l_2 and l_3;

$x_l = 0$;

for $(i = 1$ to $n_{max})$

{

$s = i \mod 4$; $t = (i + 2) \mod 4$;

arrange Γ_i on the box with the sources line on l_s and the sinks line on l_t starting form x-coordinates x_l;

set x_l equal to the maximum x-coordinate used in the drawing of G_{μ_i};

}

Lemma 11 *Let G be a 3-planar regular SP-digraph, with decomposition tree T, and let $\Gamma_1, \ldots, \Gamma_{n_{max}}$ the strip drawings of the sets of virtual pertinent graphs $L(1), \ldots L(n_{max})$ computed by Procedure* layoutVirtualSet3planar. *Let $\Gamma_1, \ldots, \Gamma_{n_{max}}$ be arranged on the box according to the Procedure* arrangeStrip-Drawing3planar. *The drawing obtained when also the jumping edges are added is a valid box-drawing.*

Proof:
The drawing of the virtual pertinent graph of each S-node is crossing-free by Lemma 10. The strip drawing of a whole set $L(j)$ $(0 \leq j \leq n_{max})$ computed by *layoutVirtualSet3planar* is crossing-free. Namely, each virtual pertinent graph of the set is drawn completely after the drawing of the previous one, thus avoiding any crossing. Now we show that the jumping edges can be added without creating crossings. Recall that the jumping edges connect source (respectively sink) poles of level j to source (respectively sink) core poles of level $j+1$. Consider the drawing Γ_j of the vertices of level j and assume without loss of generality that the sources line is on line l_0 and the sinks line is on line l_2. The Procedure *arrangeStripDrawing3planar* is such that sources line and sinks line of level $j-1$ are on line l_3 and l_1, respectively; also sources line and sinks line of level $j+1$ are on line l_1 and l_3, respectively. Thus jumping edges between level $j-1$ and j are on the strip $\sigma_{0,3}$ (*i.e.* the back side of the box) and on the strip $\sigma_{2,1}$ (*i.e.* the front side of the box), while the jumping edges between level j and $j+1$ are on the strip $\sigma_{0,1}$ (*i.e.* the top side of the box) and on the strip $\sigma_{2,3}$ (*i.e.* the bottom side of the box). This implies that jumping edges on different sides of the box do not cross each other since the four sides of the box do not intersect except at the four lines. Also, jumping edges drawn on a same side of the box do not cross each other. Consider two jumping edges connecting, for example, two different source poles s_1 and s_2 of level j to two different source core poles c_1 and c_2 of level $j+1$. Suppose without loss of generality that $x(s_1) < x(s_2)$. Let μ_1 and μ_2 be the P-nodes having s_1 and s_2 as source poles respectively. If μ_1 and μ_2 have different father then they belong to two different virtual prtinent graphs. Since the virtual pertinent graphs of the same level are drawn along the x-axis according to the left-to-right order of the corresponding S-nodes, then $x(s_1) < x(s_2)$ implies that μ_1 is on the left of μ_2. If μ_1 and μ_2 have the same father then they belong to the same virtual pertinent graph. The drawing of a virtual pertinent graph G^*_μ of an S-node μ, is such that the components of the series represented by μ, are drawn along x-axis according to the left-to-right order of the corresponding nodes. Thus, also in this case, $x(s_1) < x(s_2)$ implies that μ_1 is on the left of μ_2. The S-node ν_1 whose virtual pertinent graph has c_1 as source core pole, is in the subtree of T rooted at μ_1 and the S-node ν_2 whose pertinent graph has c_2 as source core pole, is in the subtree of T rooted at μ_2. It follows that ν_1 is on the left of ν_2 and therefore c_1 precedes c_2 and the two jumping edges cannot cross each other. □

By using procedures *layoutVirtual3Planar*, *layoutVirtualSet3Planar* and *arrangeStripDrawing3Planar* a box-drawing of a regular 3-planar SP-digraph can be computed. We now describe how to deal with the special case of 3-planar

SP-digraphs having pole of depth 2.

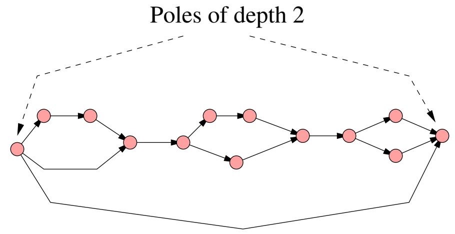


Figure 12: An example of a 3-planar SP-digraph with poles of depth 2.

As shown in Lemma 9 the only poles which may have depth two are the source s and the sink t of G itself. If this is the case, we have that the P-node ϱ which is the root of the decomposition tree T of G, has two children, one of which is a Q-node while the other (we call it ς) is an S-node whose leftmost and/or rightmost children are P-nodes (see Figure 12).

We slightly change the definition of level numbering of the decomposition tree T of G, as follows:

- if μ is a P-node then:
  - if $(\mu = root(T))$
    then $level(\mu) := 0$
    else $level(\mu) := level(father(\mu))$
- if μ is an S-node then:
  - if (μ is simple and $father(father(\mu)) \neq \varsigma$)
    then $level(\mu) := level(father(\mu))$
    else $level(\mu) := level(father(\mu)) + 1$
- if μ is a Q-node then:
  - $level(\mu) := level(father(\mu))$

The difference in this definition is that the S-node which are grandchildren of ς are assigned to level greater than that of their father. The level numbering of the vertices of G is computed as described in Section 4, except for s and t, which are poles of two different P-nodes having level number 0 and 1 respectively. The poles s and t are assigned to level 1.

For wath concerns the virtual pertinent graphs, they are computed as described in Section 4, with the two following exception:

- we consider also the virtual pertinent graphs of the simple S-node which are grandchildren of ς; these virtual pertinent graphs are simple paths with at least one vertex, and we call them *special virtual pertinent graph* in the following. If a special virtual pertinent graph has a single vertex, this vertex is called *singleton*.

- the virtual pertinent graph G^*_ς of ς is computed by replacing all the pertinent graphs of the children of ς which are P-nodes, by a virtual edge; G^*_ς is a simple path from s to t consisting of real or virtual edges, plus the edge (s,t).

The sets $L(j)$, $(j = 1, \ldots, l_{max})$ are defined as in the Section 4. Also in this case each set $L(j)$ contains all the vertices of level j and all the edges of the same level.

For j from l_{max} to 3, the strip-darwing of each set $L(j)$ and the arrangement of it on the box can be computed as described in the case of regular 3-planar graphs. A strip drawing for set $L(2)$ can be computed by the following Procedure.

Procedure *layoutLevel2*

Input The set $L(2) = \{G^*_{\mu_1}, \ldots, G^*_{\mu_{n_2}}\}$ of the virtual pertinent graphs of the S-nodes of level 2;

Output A drawing of $L(2)$ on the strip $\sigma_{a,b}$;

$x_l = 0$;

for $(i = 1$ to $n_2)$

{

  if ($G^*_{\mu_i}$ consists of a singleton v)

  {

    if (μ_i is the left child of his father)

    {

      $l(v) = l_b$; $x(v) = x_l$; $x_l = x_l + 1$;

    }

    else

    {

      $l(v) = l_a$; $x(v) = x_l$; $x_l = x_l + 1$;

    }

  }

  else

  {

    layoutVirtual3Planar($G^*_{\mu_i}$, x_l);

    set x_l equal to the maximum x-coordinate used in the drawing of $G^*_{\mu_i}$;

  }

}

The Procedure above draws each virtual pertinent graph which is not special as usual, *i.e.* by means of Procedure *layoutVirtual3Planar*. For what concern the special virtual pertinent graphs:

- if they have more than one vertex, then they are also drawn by means of Procedure *layoutVirtual3Planar*. The darwing obtained has all the vertices of the path on line l_a except the last one (the sink core pole) which is on line l_b;
- if they consists of a singleton then it is assigned either to line l_a or to line l_b. The assignment is such that, if two sibling nodes have two singletons as virtual pertinent graph, then these two singletons are on different lines.

  whose virtual pertinent graph has a single node, are assigned to different lines.

the following procedure describe how to draw the virtual pertinent graph of ς, *i.e.* the olny element of the set $L(1)$.

Procedure *layoutLevel1*

Input The virtual pertinent graph G^*_ς of ς;

Output A drawing of G^*_ς on the strip $\sigma_{a,b}$;

Let $s = v_1, v_2, \ldots, v_h = t$ the vertices of the path connecting s to t in G^*_ς.

$x_l = 0$;

for $(i = 1$ to $h - 1)$

{

$l(v_i) = l_a$; $x(v_i) = x_l$; $x_l = x_l + 1$;

}

$l(v_h) = l_b$; $x(v_h) = x_l$;

Lemma 12 *Let G be a 3-planar SP-digraph, with decomposition tree T, and let $\Gamma_3, \ldots, \Gamma_{n_{max}}$ be the strip drawings of the sets of virtual pertinent graphs $L(3), \ldots L(n_{max})$ computed by Procedure* layoutVirtualSet3planar*, and let Γ_2 and Γ_1 be the strip drawings of the sets of virtual pertinent graphs $L(2)$ and $L(1)$ computed by Procedures* layoutLevel2 *and* layoutLevel1 *respectively. Let $\Gamma_1, \ldots, \Gamma_{n_{max}}$ be arranged on the box according to the Procedure* arrangeStrip-Drawing3planar*. The drawing obtained when also the jumping edges are added is a valid box-drawing.*

Proof: The drawing consisting of $\Gamma_3, \ldots, \Gamma_{n_{max}}$ plus the jumping edges connecting them, is crossing-free by Lemma 11. We prove now that the drawing obtained by adding Γ_2 and Γ_1, and by adding the jumping egdes remaining, is a crossing-free drawing. There are not edges connecting vertices of the special virtual pertinent graphs of the set $L(2)$ to vertices of level 3. Thus the jumping edges between level 2 and 3, connect source (respectively sink) poles of non-special virtual pertinent graph of level 2 to source (respectively sink) core poles of level 3. These edges are in the same situation of any other jumping edges between level greater than 3 and therefore they do not cross.

The drawing of G^*_ς is trivially crossing-free. There are three different types of jumping edges between level 1 and level 2:

1. edges connecting source (respectively sink) pole of level 1 to source (respectively sink) core pole of non-special virtual pertinent graphs of level 2;

2. edges connecting source (respectively sink) pole of level 1 to source (respectively sink) core pole of special virtual pertinent graphs of level 2 with more than one vertex;

3. edges connecting source or sink poles of level 1 to singletos of level 2;

Recall that Γ_2 is arranged on the box so that its sources line is on line l_2 and its sinks line is on line l_0, while Γ_1 is arranged on the box so that its sources line is on line l_1 and its sinks line is on line l_3.

The jumping edges of type 1 and 2 are either on strip $\sigma_{1,2}$ or on strip $\sigma_{1,0}$. The only exception are the jumping edges (if present) connecting the vertex t to the sink core poles of level 2. The latter edges are on strip $\sigma_{3,0}$. Also the jumping edges of type 3 are either on strip $\sigma_{1,2}$ or on strip $\sigma_{1,0}$.

Since the strip $\sigma_{1,2}$, $\sigma_{1,0}$, and $\sigma_{3,0}$, do not cross each other except, possibly, at one of the four lines of the box, the jumping edges on different strips cannot cross each other.

Finally, we prove that there cannot be a crossing between jumping edges on the same strip. We consider strip $\sigma_{1,2}$, the argument for the other strips is similar. Consider two edges $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ on $\sigma_{1,2}$. Without loss of generality assume taht v_1 and v_2 are on line l_1, while u_1 and u_2 are on line l_2. If $v_1 \equiv v_2$ or $u_1 \equiv u_2$, then e_1 and e_2 do not cross since they have a common endvertex. Therefore we may assume that $v_1 \not\equiv v_2$ and $u_1 \not\equiv u_2$, and that v_1 precedes v_2 along l_1. There are four cases:

both e_1 and e_2 are jumping edges of type 1 or 2; the vertices v_1 and v_2 are source poles of two distinct P-node μ_1 and μ_2 respectively, which are children of ς. The drawing of G^*_ς is such that if v_1 precedes v_2, then μ_1 precedes μ_2 in the left to right order in T. It follows that the S-node ν_1 whose virtual pertinent graph has u_1 as source core pole, is in the subtree of T rooted at μ_1 and the S-node ν_2 whose virtual pertinent graph has u_2 as source core pole, is in the subtree of T rooted at μ_2. It follows that

ν_1 is on the left of ν_2 and therefore u_1 precedes u_2 and the two jumping edges cannot cross each other.

e_1 is a jumping edge of type 1 or 2 and e_2 is a jumping edge of type 3; the vertex v_1 is a source pole of a P-node μ_1 which is child of ς. The vertex v_2 is either a source pole or a sink pole of a P-node μ_2 which is child of ς. If $\mu_1 \not\equiv \mu_2$ then the case is analogous to the first one, and a similar argument holds. If $\mu_1 \equiv \mu_2$, then v_2 must be the sink pole of $\mu_1 \equiv \mu_2$ (otherwise v_1 and v_2 would coincide). It follows that u_1 is the source core pole of the virtual pertinent graph (special or not) of an S-node ν_1 which is child of $\mu_1 \equiv \mu_2$, and u_2 is a singleton associated with the other child ν_2 of $\mu_1 \equiv \mu_2$. Since u_2 is on line l_2 (*i.e.* the sources line for $L(2)$), then ν_2 is the right child of $\mu_1 \equiv \mu_2$. it follows that ν_1 is on the left of ν_2 and therefore u_1 precedes u_2 and the two jumping edges cannot cross each other.

e_1 is a jumping edge of type 3 and e_2 is a jumping edge of type 1 or 2; the vertex v_1 is is either a source pole or a sink pole of a P-node μ_1 which is child of ς. The vertex v_2 is a source pole of a P-node μ_2 which is child of ς. If $\mu_1 \not\equiv \mu_2$ then the case is analogous to the first one, and a similar argument holds. If $\mu_1 \equiv \mu_2$, then v_1 must be the sink pole of $\mu_1 \equiv \mu_2$ (otherwise v_1 and v_2 would coincide). This is not possible because in the drawing of G^*_ς a source pole precedes always the corresponding sink pole.

both e_1 and e_2 are jumping edges of type 3 the vertices v_1 and v_2 can be either source poles or sink poles of two (possibly coincident) P-node μ_1 and μ_2 respectively, which are children of ς. If $\mu_1 \not\equiv \mu_2$ then the case is analogous to the first one, and a similar argument holds. If $\mu_1 \equiv \mu_2$, then v_1 and v_2 are the two poles of the same parallel. Since both e_1 and e_2 are jumping edges of type 3, then both u_1 and u_2 must be singletons. Since on any line there is only one singleton connected to the same pair of poles and since u_1 and u_2 are distinct, this is not possible.

□

The complete algorithm for computing a box-drawing of a 3-planar series-parallel graph is shown in the following. An example of drawing is shown in Figure 13.

Algorithm *prismEmbedder3planar*

Input A 3-planar series-parallel graph G;

Output A drawing of G on a box;

Compute the SPQ-tree T of G;

Compute a level numbering of T and let n_{max} be the maximum level number assigned;

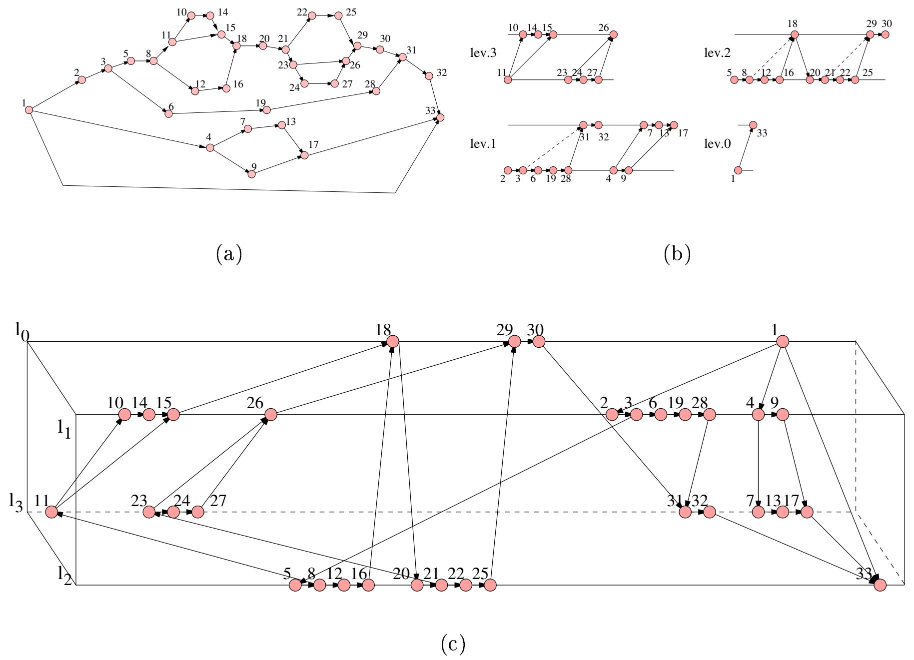


Figure 13: (a) A 3-planar Series-Parallel digraph G. (b) The strip drawings of the virtual pertinent graphs of the S-nodes of T. (c) The box-drawing of G.

Compute the sets $L(j)$ $(1 \leq j \leq n_{max})$;

if (G is regular)

{

  for $(j = l_{max}, \ldots, 0)$

    layoutVirtualSet3planar($L(j)$);

  arrangeStripDrawing3planar;

}

else

  for $(j = l_{max}, \ldots, 3)$

    layoutVirtualSet3planar($L(j)$);

  layoutLevel2

  layoutLevel1

  arrangeStripDrawing3planar;

Add the jumping edges.

The results of this section can be summarized as follows.

Theorem 3 *Let G be a 3-planar series-parallel graph with n vertices. The algorithm* prismEmbedder3planar *computes a valid box-drawing of G in $O(n)$-time and with $O(n)$ volume.*

Proof:
Th drawing computed by *prismEmbedder3planar* is a valid box-drawing by Lemma 10, Lemma 11 and Lemma 12. Since a decomposition tree T of G can be constructed in $O(n)$ time [3, 7] and both the level numbering and the drawing strategy can be easily accomplished by an $O(n)$-time visit of T, the statement about the time complexity holds. Also, the volume of the computed drawing is $4 \times n$ since the n vertices are drawn at consecutive x-coordinates and the four lines of the box are one grid unit apart. □

Acknowledgments

We thank Walter Didimo and David Wood for useful discussions on the subject of this paper.

References

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

[2] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing.* Prentice Hall, Upper Saddle River, NJ, 1999.

[3] G. Di Battista and R. Tamassia. On-line maintenance of triconnected components with SPQR-trees. *Algorithmica*, 15(4):302–318, 1996.

[4] G. Di Battista and R. Tamassia. On-line planarity testing. *SIAM Journal on Computing*, 25(5):956–997, October 1996.

[5] V. Dujmović, P.Morin, and D. Wood. Pathwidth and three-dimensional straight line grid drawings of graphs. In *Graph Drawing (Proc. GD '02)*, to appear.

[6] S. Felsner, G.Liotta, and S. Wismath. Straight line drawings on restricted integer grids in two and three dimensions. In P. Mutzel, M. Junger, and S. Leipert, editors, *Graph Drawing (Proc. GD '01)*, volume 2265 of *Lecture Notes Comput. Sci.* Springer-Verlag, 2001.

[7] J. E. Hopcroft and R. E. Tarjan. Dividing a graph into triconnected components. *SIAM J. Computing*, 2:135–158, 1973.

[8] D. R. Wood. Queue layouts and three-dimensional straight-line grid drawings. Technical Report TR-02-02, School of Computer Science, Carleton University, 2002.