# Planar Orientations

September 29, 2005

# 1 Introduction

In this chapter we will focus on algorithms and techniques used for drawing planar graphs. The algorithms we will use are based on numbering the vertices and orienting the edges from the lower numbered vertices to high numbered vertices. In this way, we can construct two kinds of geometric representation of a planar graph, the visibility representation and the tessellation representation. We will first examine the method of numbering the vertices of a digraph, and focus especially on the *st*-numbering algorithm. We will then explore the properties of planar acyclic graphs. Understanding these properties is essential for understanding how the above geometric representations of graphs work.

We say that two horizontal segments of a given set are *visible* if they can be joined by a vertical that does not intersect any other horizontal segment. A *visibility representation* of a graph draws vertices as nonoverlapping horizontal segments and edges as vertical segments drawn between visible vertex-segments (see figure 1).

A *tessellation representation* of an embedded planar graph draws each vertex, edge and face as a rectangular tile with horizontal and vertical sides such that

- There is an intersection between the interiors of any two tiles
- The boundaries of two tiles intersect if and only if the corresponding objects in the graph are incident
- The union of all tiles is a rectangle

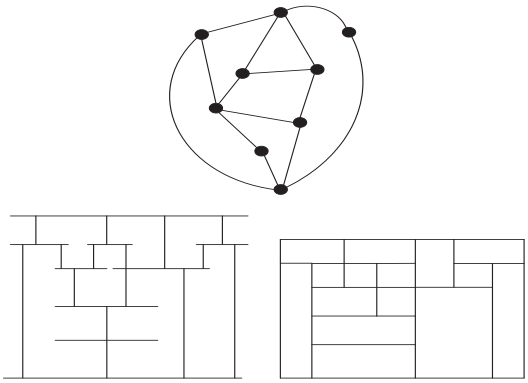

Figure 1: A planar graph, its visibility representation (left) and its tessellation representation (right).

# 2 Numbering Directed Acyclic Graphs

Let $G = (V, E)$ be a digraph with no cycles. A *topological numbering* is a function $x : V \to \mathbb{R}$ such that

$$x(v) > x(u) \quad \forall (u, v) \in E \tag{2.1}$$

Respectively, we can talk about topological sorting of a directed acyclic graph when vertices are assigned distinct integers from the set $\{1, \ldots, n\}$. Therefore, a *topological sorting* is a function $y : V \to \{1, \ldots, n\}$ such that

$$y(v) > y(u) \quad \forall (u, v) \in E \tag{2.2}$$

It is easy to see that equation 2.1 defines a linear ordering of all its vertices such that if $G$ contains an edge $(u, v)$ then $u$ appears before $v$ in the ordering. Note that topological sorting assigns distinct integers to every vertex of $G$. Thus, a topological numbering can easily be derived from a topological sorting.

Additionally, the numberings defined above can be redefined as *weighted* numberings if we assign positive weights $w(e)$ for every edge $e \in E$ and add to the right part of the inequalities 2.1, 2.2 the quantity $w(e),\ e = (u, v)$. Thus a weighted topological numbering of a directed acyclic graph $G = (V, E)$ with a positive weight function $w$ is a function $r : V \to \{1, \ldots, n\}$ such that

$$r(v) \geq r(u) + w(e) \quad \forall e = (u, v) \in E \tag{2.3}$$

Note that the normal numberings can be derived form the weighted numberings if we set all the weights of the graph equal to zero. Additionally, we say that a weighted topological is optimal if the quantity $\max_{u,v} |r(v) - r(u)|$ is minimized.

One very clever algorithm for the computation of the topological sorting of a directed acyclic graph $G = (V, E)$ is based on bucket sorting. The algorithm works as follows. Our graph has for sure a source with in-degree equal to zero. We build up a list with $n = \max_{i \in V}\{deg^-(i)\} + 1$ entries. Each entry $0 \leq i \leq n$ is a linked list of nodes $k$ with $deg^-(k) = i$. Each node in this list has pointers to its reachable nodes (i.e. nodes $t$ that can be reached following the outgoing edges $(i, t)$). The algorithm subtracts one by one nodes $k$ such that $deg^-(k) = 0$, giving them an increasing topological sorting index and simultaneously decreasing the in-degree of their out-neighborhood by one. The algorithm goes on till all nodes get an index. It easy to prove that the algorithm indeed produces a legal topological sorting as every directed arc $(u, v)$ is processed such as its origin $u$ is always subtracted before its destination $v$. Thus $v$ will never get a lower number and the topological sorting will be legal.

One very important observation is that a topological sorting of a directed graph $G$ is not unique unless $G$ has a directed path that visits every vertex. It is easy to show that all the following statements are equivalent:

- $G$ is acyclic
- $G$ admits a topological numbering
- G admits a topological sorting

Topological sortings and topological numberings are defined on directed graphs. What happens when the graph is undirected? Then, we must compute an *st*-numbering [1] of the undirected graph and transform it to a directed one.

# 3 Properties of Planar Acyclic Digraphs

We define an *st*-graph, as a planar acyclic digraph with one source vertex $s$ and one single sink vertex $t$. If we apply a topological numbering on an *st*-graph $G$, we can see that the way the vertices are numbered, give a sense of direction, from a vertex with a low number to a vertex with a higher number, to the edges. The following properties hold:

- Given a topological numbering of an *st*-graph $G$, each directed path of $G$ visits vertices with increasing numbers.

- For every vertex $v$ of an $st$-graph $G$, there exists at least one directed path $P$ from $s$ to $t$ that contains $v$

The first property holds because of the way the numbers correspond with the directions of the edges. It is easy to see why the second property is true: if it wasn't, there would be either no path from $s$ to $v$, or from $v$ to $t$, thus $s$ or $t$ wouldn't be the source or sink vertices respectively.

A planar $st$-graph is an $st$-graph that is planar and embedded with vertices $s$ and $t$ on the boundary of the external face. It is customary to visualize a planar $st$-graph as drawn upward in the plane (with $s$ at the bottom and $t$ at the top), as shown in figure 2. All planar $st$-graphs, being acyclic, admit a topological ordering (numbering).

Let now $G$ be a planar $st$-graph and $F$ be its set of faces. $F$ contains two representatives of the external face: the "left external face" $s^*$, which is incident with the edges on the left boundary of $G$ and the "right external face" $t^*$, which is incident with the edges on the right boundary of $G$. Additionally, for each $e = (u, v)$ we define $orig(e) = u$ and $dest(e) = v$. Also, we define $left(e)$ (respectively $right(e)$) to be the face to the left (respectively right) of $e$. Following, we give the definition of the dual graph $G^*$ of a planar

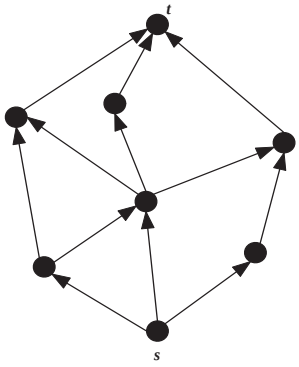


Figure 2: A planar $st$-graph.

$st$-graph $G$. The dual graph $G^*$ is a graph for which the following hold:

- The vertex set of $G^*$ is the set $F$ of faces of $G$ including the faces $s^*$ and $t^*$.
- For every edge $e \neq (s, t)$ of $G$, $G^*$ has one edge $e^* = (f, g)$ where $f = left(e)$ and $g = right(e)$.

The dual graph of the graph depicted in Figure 2 can be seen in Figure 3. If we rotate $G^*$ 90 degrees, we can see that $G^*$ is also a planar $st$-graph.

Given a vertex $v$ of a planar $st$-graph, the face separating the incoming from the outgoing edges in the clockwise direction is called $left(v)$, and the other separating face is called $right(v)$(see figure 4).

Following, we give some theoretical results.

**Lemma 3.1.** *Each face $f$ of a planar st-graph $G$ consists of two directed paths with common origin, called orig($f$), and common destination, called dest($f$).*

*Proof.* Let's suppose that the lemma is not true for a face $f$ of the graph. Then there should be two vertices $w$ and $u$ on the boundary edges of $f$, which create a path directed

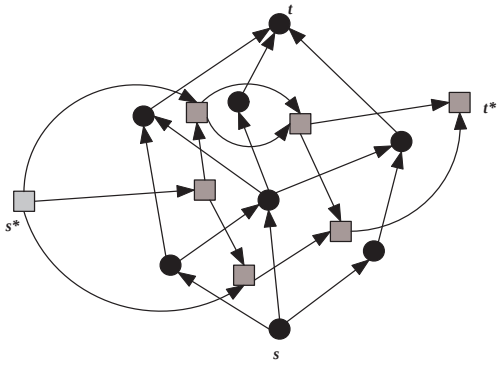


Figure 3: Constructing the dual graph from the primal one.

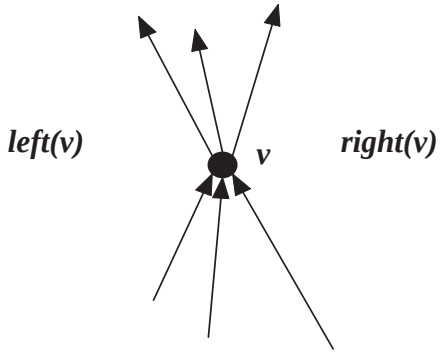


Figure 4: The left ($left(v)$) and right ($right(v)$) faces of a vertex $v$.

from $dest(f)$ to $orig(f)$. Then, there should also be a directed path $P_1$ from $s$ to $w$ and a path $P_2$ from $u$ to $t$. But these two paths must intersect, and since $G$ is planar, there should be a vertex $x$ at their intersection point. But then, we can clearly see that $G$ has a cycle, between vertices $w$, $u$ and $x$ which contradicts the fact that $G$ is a planar $st$-graph (see Figure 5). □

**Lemma 3.2.** *The incoming edges for each vertex $v$ of a planar $st$-graph $G$ appear consecutively around $v$, and so do the outgoing edges.*

*Proof.* The lemma is true for vertices $s$ and $t$. Let's suppose that the lemma is not true for a vertex $v$ of $G$. This means that there are edges, as they appear in Figure 6. Then there should be a directed path $P_1$ from $s$ to $b$, and a directed path $P_2$ from $c$ to $t$. But these two paths intersect, and since $G$ is planar, there should be a vertex $x$ at their intersection point. But then a cycle between vertices $v$, $c$, $x$ and $b$ is created, contradicting to the fact that $G$ is a planar $st$-graph. □

**Lemma 3.3.** *For every two faces $f$ and $g$ of a planar $st$-graph, exactly one of the following holds:*

- *$G$ has a directed path from $dest(f)$ to $orig(g)$*
- *$G$ has a directed path from $dest(g)$ to $orig(f)$*

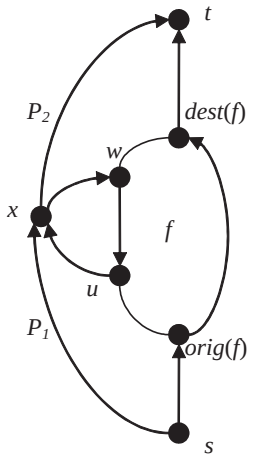


Figure 5: Proof of Lemma 3.1.

- *$G^*$ has a directed path from $f$ to $g$*
- *$G^*$ has a directed path from $g$ to $f$*

*Proof.* Before we prove lemma 3.3, we must give two more definitions that we will use in this proof. We call the leftmost path from a vertex $u$ the path that always takes the leftmost outgoing edge. Similarly, the rightmost path of a vertex $u$ is the path that always takes the rightmost edge. Now let's assume that $G$ is topologically sorted and that the number of $dest(f)$ is less than the number of $orig(g)$. The rightmost and leftmost paths from $dest(f)$ to $t$ are called $P_1$ and $P_2$ respectively. Similarly, the leftmost and rightmost paths from $orig(g)$ to $t$ are called $P_3$ and $P_4$ respectively. The lemma is obviously true if there is a directed path from $dest(f)$ to $orig(g)$. Otherwise $P_2$ and $P_3$ or $P_1$ and $P_4$ intersect. Let's assume that $P_2$ intersects $P_3$ at a vertex $x$. Then from lemma 3.2, every edge incident to every vertex on $P_2$, from the right, is incoming, as it also happens with every edge incident to $P_3$ from the left. If we construct $G^*$, we will see that there is a directed path in $G^*$ from $f$ to $g$. (see Figure 7) □

In the above lemma we deal with a special case of a more general property of planar *st*-graphs. We can make the abstraction and call an element of the set $V \cup E \cup F$ of a planar *st*-graph $G$ an object. The definitions of $orig()$, $dest()$, $left()$, $right()$ can then be extended as follows. For a vertex $v$ we define $orig(v) = dest(v) = v$ and for a face $f$ we define $left(f) = right(f) = f$. Lemma 3.3 can then be generalized as follows:

**Lemma 3.4.** *For any two objects $o_1$ and $o_2$ of a planar st-graph $G$, exactly one of the following holds:*

- *$G$ has a directed path from $dest(o_1)$ to $orig(o_2)$*
- *$G$ has a directed path from $dest(o_2)$ to $orig(o_1)$*
- *$G*$ has a directed path from $right(o_1)$ to $left(o_2)$*
- *$G*$ has a directed path from $right(o_2)$ to $left(o_1)$*

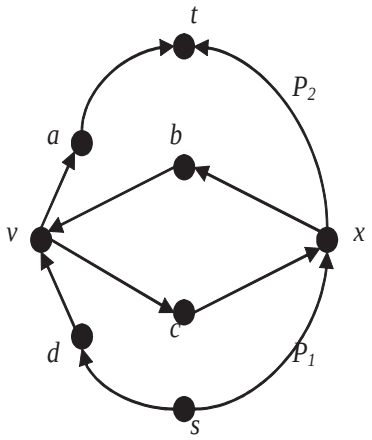


Figure 6: Proof of Lemma 3.2.

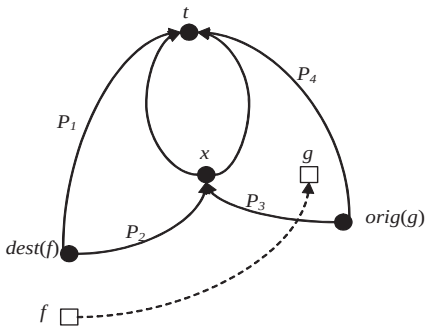


Figure 7: Proof of Lemma 3.3.

# 4 Tessellation Representations

## 4.1 Plane Tessellation Representation

A tessellation representation on the plane for a planar *st*-graph G is a partition of the plane into disjoint tiles, each associated with vertex, edge or face of G, such that the topological incidencies correspond to geometric adjacencies between tiles.

A tile is a rectangle with sides parallel to the coordinate axes. A tile can be unbounded or can degenerate to a segment or a point. Two tiles are horizontally or vertically adjacent if they share a portion of a vertical or horizontal side. The coordinates of a tile $\theta$ will be denoted by $x_L(\theta)$, $x_R(\theta)$, $y_B(\theta)$, $y_T(\theta)$.

Let $G$ be a planar *st*-graph. As usual, we denote the sets of vertices, edges and faces of $G$ by $V$, $E$ and $F$, respectively. A tessellation representation $\Theta$ for $G$ maps each object (vertex, edge, or face) $o$ of $G$ into a tile $\Theta(o)$, such that:

- The interiors of tiles $\Theta(o_1)$ and $\Theta(o_2)$ are disjoint whenever $o_1 \neq o_2$
- The union of all tiles $\Theta(o)$, $o \in V \cup E \cup F$ is a rectangle

- Tiles $\Theta(o_1)$ and $\Theta(o_2)$ are horizontally adjacent if and only if $o_1 = left(o_2)$ or $o_1 = right(o_2)$ $o_2 = left(o_1)$ or $o_2 = right(o_1)$
- Tiles $\Theta(o_1)$ and $\Theta(o_2)$ are vertically adjacent if and only if $o_1 = orig(o_2)$ or $o_1 = dest(o_2)$ or $o_2 = orig(o_1)$ or $o_2 = dest(o_1)$

Let now $G$ be a planar $st$-graph. We want to construct a tessellation representation $\Theta$ of $G$. To do so, we firstly compute the dual planar $st$-graph $G^*$. Then, we compute a topological numbering $Y$ of $G$ and a topological numbering $X$ of $G^*$. Finally, for each object $o \in V \cup E \cup F$, we let the coordinates of tile $\Theta(o)$ be

$$\begin{aligned} x_L(o) &= X(left(o)) \\ x_R(o) &= X(right(o)) \\ y_B(o) &= Y(orig(o)) \\ y_T(o) &= Y(dest(o)) \end{aligned}$$

**Theorem 4.1.** *Let $G$ be a planar st-graph with $n$ vertices. The algorithm described above constructs a tessellation representation of $G$ in $O(n)$ time.*

*Proof.* The tiles of any two distinct objects are separated either by a vertical or by horizontal line, according to Lemma 3.4. Each step of the algorithm takes linear time, so the above algorithm constructs the plane tessellation representation in $O(n)$ time. □

An example of a run of the plane tessellation representation algorithm is shown below in Figure 8.

Notice that the above algorithm constructs a tessellation representation on the plane, in which the tiles associated with the faces and the vertices (except $s$ and $t$) are degenerate. In particular, the external face is associated with a tile at infinity. It is possible to modify the construction so that only one tile is degenerate, namely the one associated with the external face.

We can modify the plane tessellation representation algorithm to support user-defined constraints on the size of the edge-tiles. Namely, let $h(e)$ and $w(e)$ be non-negative numbers associated with each edge $e$ of $G$. By replacing the first two steps of the plane tessellation representation algorithm with the following ones, we obtain a tessellation representation of $G$, such that the tile of each edge $e$ has height at least $h(e)$ and width at least $w(e)$:

- Assign weight $h(e)$ to each edge $e$ of $G$ and compute an optimal weighted topological numbering $Y$ of $G$.
- Assign weight $w(e^*)$ to each edge $e^*$ of $G^*$ and compute an optimal weighted topological numbering $X$ of $G^*$.

The plane tessellation representation algorithm can also be further modified to support user-defined constraints on the size of the vertex-and face tiles. Namely, we construct from $G$ a new planar st-graph $G'$ as follows:

- Let $G' = G$.
- For each vertex $v$ of $G'$, we expand $v$ into vertices $v'$ and $v''$, joined by an edge $e_u$ from $v'$ to $v''$, such that $v'$ contains the incoming edges of $v$ and $v''$ contains the outgoing edges of $v$.

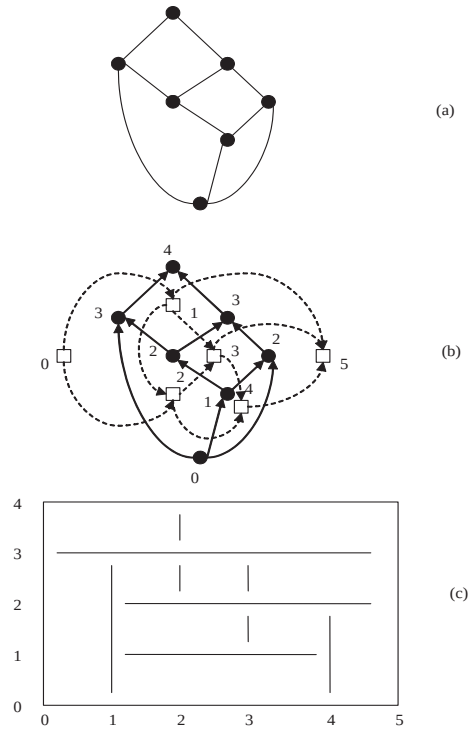


Figure 8: Example of a run of the plane tessellation representation algorithm: (a) a planar graph $G$; (b) planar $st$-graphs $G$ and $G^*$ labelled by topological numberings $Y$ and $X$, respectively; (c) tessellation representation $\Theta$ of $G$ constructed by the plane tessellation representation algorithm

- For each face $f$ of $G'$, we add an edge $e_f$ into face $f$ from $orig(f)$ to $dest(f)$.

Every object of $G$ is associated with an edge of $G'$. We then simply apply Plane tessellation representation algorithm to $G'$ and represent each object of G with the tile of the associated edge of $G'$. We conclude:

**Theorem 4.2.** *Given a planar st-graph $G$ with $n$ vertices and nonnegative numbers $h(o)$ and $w(o)$ for each object $o$ of $G$, a minimum-area tessellation representation $\Theta$ for $G$, such that each tile $\Theta(o)$ has height at least $h(o)$ and width at least $w(o)$ can be constructed in time $O(n)$. In particular, if $h(o) = w(o) = 1$ for each object $o$ of $G$, then $\Theta$ has integer coordinates and area $O(n^2)$.*

*Proof.* We construct a plane tessellation representation such that each tile $\Theta(o)$ has height and weight, by expanding each vertex into two new vertices. The only essential difference between the initial graph $G$ and graph $G'$ obtained by the changes we apply, is that the second one has more objects. So, according to theorem 4.1, we can construct a tessellation representation on the plane in $O(n)$ time for a graph $G$. If $h(o) = w(o) = 1$, then the objects $o$ of G will have integer dimensions. So $\Theta$ will have integer coordinates. Also $\Theta$ will have area $O(n^2)$, because vertices and faces will be represented as rectangles and not as lines. □

## 4.2 Sphere Tessellation Representation

A sphere $S$ is the locus of points at the same distance from a point, called the center of the sphere. The intersection of $S$ with the horizontal plane that passes through the center of $S$ defines a circle, called the equator. Similarly, the intersection of $S$ with planes parallel to the plane of the equator defines the parallels. The line that passes through the center of $S$ and is orthogonal to the plane of the equator, called the axis of the sphere, intersects the sphere into two points, the North Pole and the South Pole. Every plane that is orthogonal to the plane of the equator and passes through the two Poles defines a circle called a meridian. Every point $p$ of $S$ will be denoted by a pair $(x, y)$ where $x$ is the latitude measured with respect to the South Pole, and $y$ is the longitude measured with respect to a reference meridian. The notion of horizontal and vertical is extended to the sphere by considering horizontal the parallels and vertical the meridians.

A spherical $st$-graph is an embedded planar acyclic digraph with exactly one source $s$ and exactly one sink $t$. It is convenient to visualize a spherical $st$-graph as drawn on a sphere with $s$ at the "South Pole" and $t$ at the "North Pole". A tile on the sphere is the portion of the sphere delimited by two parallels and two meridians. On the sphere, we allow tiles containing one or both Poles.

Now we consider a spherical $st$-graph $G$. Let $p$ be a path in $G$ from $s$ to $t$, which does not contain its extreme vertices $s$ and $t$. We construct from $G$ a new planar $st$-graph $G_p$ by "cutting" $G$ along path $p$ and duplicating the vertices and edges of $p$. Note that the graph $G_p$ is a planar $st$-graph. The two copies of $p$, denoted $p'$ and $p''$ are the left and the right boundary of $G_p$, respectively. Also, we denote by $o'$ and $o''$ the two copies of a vertex or edge $o$ of $p$ in $p'$ and $p''$, respectively. For any other object $o$ of $G$ which is not in $p$, both $o'$ and $o''$ denote the unique object of $G_p$ associated with $o$.

Now we are going to describe an algorithm that constructs a sphere tessellation representation. Input is a spherical $st$-graph $G$. The algorithm outputs a tessellation representation $\Theta$ for $G_p$ on the plane such that for each object $o$ of $p$ the tiles $\Theta(o')$ and $\Theta(o'')$ have the same $y$-coordinates.

Initially, the algorithm constructs the planar $st$-graph $G_p$. The it constructs its dual $G_p^*$. Afterwards it computes a topological numbering $X$ of $G_p$ and a topological numbering $Y$ of $G_p^*$. Then it simply sets

$$x_0 = \frac{2\pi}{Y(t)}$$

and

$$y_0 = \frac{2\pi}{X(t)}$$

Following, for every object $o \in V \cup E \cup F$ it sets

$$\begin{aligned} x_L(o) &= Y(orig(o))x_0 \\ x_R(o) &= Y(dest(o))x_0 \\ y_B(o) &= X(left(o))y_0 \\ y_T(o) &= X(right(o))y_0 \end{aligned}$$

The above algorithm constructs a tessellation representation, in which the tiles associated with the faces and the vertices are degenerate. It is possible to modify the construction, so that no tile is degenerate.

**Theorem 4.3.** *Let $G$ be a spherical st-graph with $n$ vertices. The sphere tessellation representation algorithm constructs a tessellation representation of $G$ on the sphere in $O(n)$ time.*

*Proof.* The sphere tessellation representation, essentially, constructs a plane tessellation representation on the plane for $G_p$, with the property that for each object $o$ of $p$ tiles $\Theta(o')$ and $\Theta(o'')$ have the same $y$-coordinates. So, according to Theorem 4.1, the sphere tessellation representation algorithm constructs a tessellation representation on the sphere in $O(n)$ time. □

# 5 Visibility Representations

Let $G$ be a planar $st$-graph. A *visibility representation* $\Gamma$ of $G$ draws each vertex $v$ as a horizontal segment, called *vertex-segment* $\Gamma(v)$, and each edge $(u, v)$ as a vertical segment, called *edge-segment* $\Gamma(u, v)$, such that

- The vertex-segments do not overlap
- The edge-segments do not overlap
- Edge-segment $\Gamma(u, v)$ has its bottom endpoint on $\Gamma(u)$, its top end-point on $\Gamma(v)$, and does not intersect any other vertex-segment.

A visibility representation of a planar $st$-graph $G$ can easily be constructed from a tessellation representation of $G$ with degenerate vertex tiles and non-degenerate face-tiles, which can be produced by the algorithm of theorem 4.2. Indeed, the tessellation representation provides a floorplan for drawing each vertex-segment as the degenerate vertex-tile itself, and each edge-segment as any vertical segment spanning its tile (see Figure 9).

In order to construct a visibility representation of a planar $st$-graph $G$, we apply the following algorithm. Initially, we construct the dual graph $G^*$, we assign unit weights to both graphs $G$ and $G^*$ and then we compute their optimal weighted topological numberings $Y$, $X$ respectively. Then

1. For each $v \in V$, draw the vertex-segment $\Gamma(v)$ by setting

$$\begin{aligned} y(\Gamma(v)) &= Y(v) \\ x_L(\Gamma(v)) &= X(left(v)) \\ x_R(\Gamma(v)) &= X(right(v)) - 1 \end{aligned}$$

2. For each $e \in E$, draw the edge-segment $\Gamma(e)$ by setting

$$\begin{aligned} x(\Gamma(e)) &= X(left(e)) \\ y_B(\Gamma(e)) &= Y(orig(e)) \\ x_T(\Gamma(e)) &= Y(dest(e)) \end{aligned}$$

The proof of correctness of the above algorithm is based on the following observations. By Lemma 3.4 and the construction of the algorithm, any two vertex-segments are separated by a horizontal or vertical strip of at least unit width. Also, any two edge-segments on opposite sides of a face are separated by a vertical strip of at least one unit width, and no two faces intersect in the representation constructed by the algorithm, except for their common edges.

The described algorithm obviously takes time $O(n)$. Hence

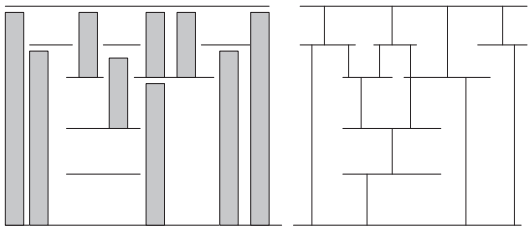

Figure 9: Construction of a visibility representation (right) from the tessellation representation (left).

**Theorem 5.1.** *Let $G$ be a planar st-graph with $n$ vertices. The described algorithm constructs in $O(n)$ time a visibility representation of $G$ with integer coordinates and $O(n^2)$ area.*

# 6 Constrained Visibility Representation

In this section we examine an algorithm that constructs a special case of visibility representations, called *constrained visibility representations*. These geometric representations of planar *st*-graphs have many interesting applications.

Let $G$ be a planar *st*-graph with $n$ vertices. Two paths $\pi_1$ and $\pi_2$ of $G$ are said to be *nonintersecting* if they are edge disjoint and do not cross at common vertices, i.e., they are also vertex disjoint paths.

Given a collection $\Pi$ of pairwise nonintersecting paths of $G$, we consider the problem of construction a visibility representation $\Gamma$ of $G$, such that, for every path $\pi$ of $\Pi$, the edges of $\pi$ are vertically aligned. More formally, for any two edges $e'$ and $e''$ of $\pi$, the edge-segments $\Gamma(e')$ and $\Gamma(e'')$ have the same $x$-coordinate. An example of a constrained visibility representation is given in Figure 10

In order to simplify the description of the algorithm, without loss of generality we assume that the set $\Pi$ of nonintersecting paths covers the edges of $G$. Otherwise, each edge originally not in $\Pi$ is inserted into $\Pi$ as a single-edge path.

To construct a constrained visibility representation we apply the following algorithm. Initially we construct a graph $G_\Pi$ (which is a planar *st*-graph) with vertex set $F \cup \Pi$ end edge set

$$\{(f,\pi)|f = left(e) \wedge e \in \pi\} \cup \{(\pi,g)|g = right(e) \wedge e \in \pi\}$$

Then we assign unit weights to the edges of $G$ and compute an optimal weighted topological numbering $Y$ of $G$, such that $Y(s) = 0$. Following, we assign half-unit weights to the edges of $G_\pi$ and compute an optimal weighted topological numbering $X$ of $G_\pi$, such that $X(s^*) = -1/2$. Finally, we execute the following *for* loops:

1. $\forall \pi \in \Pi \wedge \forall e \in \pi$ **do**:

$$\begin{aligned} x(\Gamma(e)) &= X(\pi) \\ y_B(\Gamma(e)) &= Y(orig(e)) \\ y_T(\Gamma(e)) &= Y(dest(e)) \end{aligned}$$

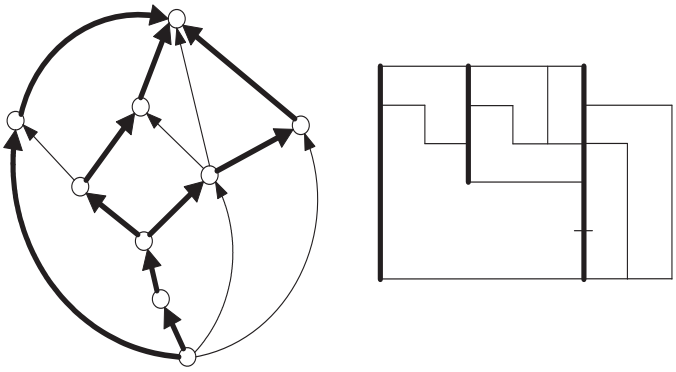

Figure 10: Construction of a constrained visibility representation. The paths belonging to $\Pi$ are shown with thick lines.

2. $\forall v \in V$ **do**

$$y(\Gamma(v)) = Y(v)$$
$$x_L(\Gamma(v)) = \min_{v \in \pi} X(\pi)$$
$$x_R(\Gamma(v)) = \max_{v \in \pi} X(\pi)$$

The computations performed by the described algorithm are equivalent to the following construction. First it modifies $G$ by duplicating each path in $\Pi$, thus forming a new face for each path. Second, it constructs a visibility representation for the modified graph such that the edge segments of the left side of the boundary of each face are vertically aligned, and two copies of an original vertex are horizontally aligned. Finally, it removes the right copy of every duplicated edge and joins the copies of the duplicated vertices. Finally, it easy to prove that $G_\Pi$ is a planar $st$-graph.

The described algorithm computes a correct visibility representation, as each edge $e$ of a path $\pi$ is assigned the same $x$-coordinate $x(\Gamma(e)) = x(\pi)$. The area needed is $O(n^2)$ and the algorithm has $O(n)$ time complexity.

# 7 Polyline Drawings

In the following section, we describe an algorithm that constructs an upward planar polyline drawing of a planar $st$-graph $G$ starting from a visibility representation of G. The algorithm consists of the following general steps. We draw each vertex $v$ of $G$ at an arbitrary point of its vertex-segment, and each edge $(u, v)$ of $G$ as a three-segment polygonal chain, whose middle segment is a subset of the edge segment of $(u, v)$.

Following we describe the algorithm that computes a polyline drawing. Initially, we compute a visibility representation $\Gamma$ of $G$. Then, for each vertex $v$, we replace $\Gamma(v)$ with an arbitrary point $P(v) = (x(v), y(v))$ on $\Gamma(v)$. Then, for each edge $(u, v)$ we distinguish the following two cases:

1. **if** $y(v) - y(u) = 1$ (short edge), then replace the edge segment $\Gamma(u, v)$ with the segment with endpoints $P(u)$ and $P(v)$

2. **if** $y(v) - y(u) > 1$ (long edge), then replace the edge segment $\Gamma(u, v)$ with the polygonal line from $P(u)$ to $P(v)$ with origin $(x(\Gamma(u, v)), y(u) + 1)$ and $(x(\Gamma(u, v)), y(v) - 1)$

A possible choice for the placement of $P(v)$ is the middle point of vertex-segment $\Gamma(v)$. An example of polyline drawing obtained with this "median positioning" from the visibility representation of Figure 11. Following we give some theoretical results

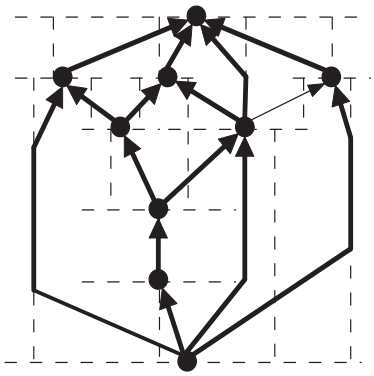

Figure 11: Construction of a polyline drawing.

**Theorem 7.1.** *Let $G$ be a planar st-graph with $n$ vertices. The algorithm described constructs in $O(n)$ time a planar upward polyline grid drawing of $G$ with the following properties:*

- *The number of bends is at most $6n - 12$*
- *Every edge has at most two bends*

*Proof.* The correctness of the algorithm can be proved with simple geometric considerations. Each edge segment $\Gamma(u, v)$ is replaced by either a segment or a polygonal line with at most two bends. Since a planar graph has at most $3n - 6$ edges, the total number of bends is at most $6n - 12$. □

# References


[1] S. Even, R. Tarzan, Theoretical Computer Science 2 (1976) 339-344, *Computing an st - numbering*

[2] G. Battista, P. Eades, R. Tamassia, I. Tollis, *Graph Drawing - Algorithms for the Visualization of Graphs*, (1999)