Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 14, no. 2, pp. 337–365 (2010)

Applications of Parameterized *st*-Orientations

Charalampos Papamanthou[1] *Ioannis G. Tollis*[2]

[1]Brown University, Providence RI, USA
[2]University of Crete and ICS-FORTH, Heraklion, Greece

Abstract

An *st*-orientation of a biconnected undirected graph defines a directed graph with no cycles, a single source s and a single sink t. Given an undirected graph G as input, linear-time algorithms have been proposed for computing an *st*-orientation of G. Such an orientation is useful especially in graph drawing algorithms which use it at their first stage [23]. Namely, before they process the original undirected graph they receive as input, they transform it into an *st*-DAG, by computing an *st*-orientation of it. In this paper we observe that using *st*-orientations of different longest path lengths in various applications can result in different solutions, each one having its own merit. Guided by this intuition, we present results concerning applications of proposed algorithms for longest path parameterized *st*-orientations. Specifically, we show how to achieve considerable space savings (e.g., $O(n)$) for visibility representations of planar graphs by using *st*-orientations computed by algorithms that can control the length of the longest path. Also, we apply our results to the graph coloring problem, where we use an *st*-orientation as an intermediate step to compute a good coloring of a graph, and to other problems, such as computing space-efficient orthogonal drawings and longest paths.

Submitted:	Reviewed:	Revised:	Accepted:
December 2008	May 2009	December 2009	April 2010
	Final:	Published:	
	June 2010	June 2010	

Article type:
Regular paper

Communicated by:
U. Brandes

E-mail addresses: cpap@cs.brown.edu (Charalampos Papamanthou) tollis@ics.forth.gr (Ioannis G. Tollis)

1 Introduction

An st-orientation is an orientation of an undirected graph that satisfies certain criteria, i.e., it defines no cycles and has exactly one source s and one sink t. Starting with an undirected biconnected graph $G = (V, E)$, many graph algorithms, such as graph drawing algorithms [5, 17, 23] and network routing algorithms [1, 2], use an st-orientation of G as a first step.

Close to the notion of st-orientations are st-numberings, which were first introduced in 1967 [14]. An st-numbering of G is a numbering of its nodes such that s receives number 1, t receives number n and every other node except for s and t is adjacent to at least one lower-numbered and at least one higher-numbered node, where n is the number of nodes of the graph. An st-orientation of an undirected graph can be easily computed using an st-numbering of the respective graph G and orienting the edges of G from *low* to *high*.

There has been a lot of work on efficient algorithms for the computation of an st-orientation (or an st-numbering). The first existential result is given in [14] where it is proved that given any edge $\{s, t\}$ of a biconnected undirected graph G, we can define an st-numbering. The proof of a theorem in [14] gives a recursive algorithm that runs in time $O(nm)$. However, in 1976 Even and Tarjan proposed an algorithm that computes an st-numbering of an undirected biconnected graph in $O(n + m)$ time [7]. Ebert [6] presented a slightly simpler algorithm for the computation of such a numbering, which was further simplified by Tarjan [24]. The planar case has been extensively investigated in [20] where a linear-time algorithm is presented which may reach any st-orientation of a planar graph. Additionally, in [16] a parallel algorithm is described (running in $O(\log n)$ time using $O(m)$ processors) and finally in [3] another linear-time algorithm for the problem is presented. An overview of the work concerning st-orientations is presented in [4].

However, all developed linear-time algorithms compute an st-numbering at random, without expecting any specific properties of the oriented graph. Algorithms for longest path parameterized st-orientations—namely, algorithms that apart from computing a correct st-orientation also manage to effectively control the length of the longest path of the produced st-orientation—, that, however, do not run in linear time and the application of which we study in this paper, firstly appeared in [19]. Towards this direction, but in a more theoretical context, Zhang and He have recently improved on various theoretical bounds on the longest path length of st-orientations for the case of plane triangulations only [11, 26, 27, 28]. They also provided an algorithm to construct an st-orientation of a plane triangulation of minimum longest path length, equal to $2n/3 + O(1)$ [29], by essentially matching a lower bound presented in [28]. Finally, computing an st-orientation of minimum longest path length was recently proved to be NP-hard for the case of 2-connected plane graphs [21] (in this paper we give a more general result).

This work presents extensive applications of new techniques that produce orientations with specific properties (parameterized longest path length). We apply parameterized st-orientations in several problems, such as the computation of visibility representations [23] of minimal (variable) area, graph coloring and orthogonal drawings [17]. For visibility representations, we exhibit some classes of planar graphs for which using *st*-orientations of different longest path lengths results in significant space savings, i.e., of a factor of $O(n)$, where n is the number of the nodes of the graph. For graph coloring, we derive an algorithm that uses an *st*-orientation to produce a graph coloring of a graph and we present some experiments for a well-known benchmark set of instances, that justify the effectiveness of our approach. We finally note that an *st*-orientation can also be used as a heuristic solution for the well-known *longest path problem*, which is NP-hard [9]. To achieve that, we can compute an *st*-orientation of maximal longest path and output the longest path of the orientation as a solution.

The paper is organized as follows. In Section 2 we present some preliminary definitions and briefly describe the algorithm that is used for the presented applications. In Section 3 we comment on applications of longest path parameterized *st*-orientations in graph drawing problems, graph coloring and longest path. Section 4 presents experimental results. Finally, some conclusions and discussion are presented in Section 5.

2 Preliminaries

In this section, we outline the algorithms [19] used for the presented applications.

2.1 Terminology

Throughout the paper, $N_G(v)$ denotes the set of neighbors of node v in graph G, s is the source, t is the sink of the graph, n is the number of nodes of G and m is the number of edges of G. Additionally, $\ell(G)$ is the length of the longest path of the *st*-oriented graph G (i.e., the length of the longest path from s to t)[1].

Let now $G = (V, E)$ be a *one-connected* undirected graph, i.e., a graph that contains at least one node whose removal causes the initial graph to disconnect. The nodes that have this property are called *cutpoints* [13]. Every *one-connected* graph is composed of a set of *blocks* (biconnected components) and *cutpoints* that form a tree structure. This tree is called the *block-cutpoint tree* of the graph and its nodes are the blocks and cutpoints of the graph. Suppose now that G consists of a set of blocks B and a set of cutpoints C. Each edge $(i, j) \in U$ of the block-cutpoint tree connects a block to a cutpoint that belongs to its node set [13]. The respective block-cutpoint tree $T = (B \cup C, U)$ has $|B| + |C|$ nodes and $|B| + |C| - 1$ edges.

The block-cutpoint tree is a free tree, i.e., it has no distinct root. In order to transform this free tree into a rooted tree, we define the t-rooted block-cutpoint

[1] Most of the times the source and the sink vertices of the *st*-orientation will be obvious and therefore there is no need to explicitly mention them in the notation.

tree with respect to the sink t. Consequently, the root of the block-cutpoint tree is the block that contains t.

Finally, we define the leaf-blocks of the t-rooted block-cutpoint tree to be the blocks, except for the root, of the block-cutpoint tree that contain a single cutpoint. The block-cutpoint tree can be computed in $O(n+m)$ time with an algorithm similar to DFS [13].

2.2 The algorithm

The idea behind the algorithm is as follows. Given an undirected biconnected graph G and two of its nodes s, t, we repeatedly remove a node v_i (different from t), orienting at the same time all its incident edges from v_i to its neighbors. Thus we build a directed graph F. The first node removed is the source s of the desired st-orientation. Thus, the problem of computing a correct st-orientation is reduced to this of removing the nodes of the graph in a correct order $v_1, v_2, \ldots, v_n$ with $v_1 = s$ and $v_n = t$ and simultaneously maintaining a data structure (basically correctly updating the t-rooted block-cutpoint tree) that will allow us to compute such a correct order. We now have the following lemma:

Lemma 1 (Block-cutpoint tree [19]) *Let $G = (V, E)$ be an undirected biconnected graph and let s, t be two of its nodes. Suppose we remove s and all its incident edges. Then there is at least one neighbor of s lying in a leaf-block of the t-rooted block-cutpoint tree of $G - \{s\}$. Moreover, this neighbor is not a cutpoint.*

The algorithm proceeds by successively removing nodes and simultaneously updating the t-rooted block-cutpoint tree that corresponds to the new graph. We call each such node a source, because at the time of its removal it is effectively chosen to be a source of the remainder of the graph. We initially remove s, the first source, which is the source of the desired st-orientation and give direction to all its incident edges from s to all its neighbors. Note that after this removal, the graph either remains biconnected or is decomposed into several biconnected components. This procedure continues until all the nodes of the graph are removed, except for the desired sink t of the st-orientation. At each step of the algorithm, the updated biconnectivity structure gives us information about the choice of the next source. Actually, the biconnectivity maintenance allows us to remove nodes and simultaneously maintain a "map" of possible nodes whose future removal may or may not cause dramatic changes to the structure of the tree.

At every step of the algorithm there will be a set of potential sources to choose from. According to the choice we make every time, we can efficiently influence the length of the longest path. A parameter $0 \leq p \leq 1$ is introduced as an input to the algorithm that provides control on the length of the longest path, i.e., for $p = 1$ the algorithm explores the graph in a "DFS" mode, producing st-oriented graphs of long longest path (in that case we say that we apply MAX-STN) whereas for $p = 0$ the algorithm explores the graph in a "BFS" mode, producing *st*-oriented graphs of small longest path (in that case we say that we apply MIN-STN). For other values of p ($0 < p < 1$) the algorithm is defined as a hybrid execution of BFS and DFS and in that case we say that we apply PAR-STN.

In terms of time complexity, since we have to remove n nodes and at each removal we have to run an $O(n+m)$ algorithm that updates the t-rooted block-cutpoint tree, the total complexity of the algorithm is not linear, i.e., it is $O(nm)$. By using techniques from [12], we can reduce the complexity to $O(m \log^5 n)$ time ($O(\log^5 n)$ amortized cost per edge removal). Furthermore, for planar graphs, for which the block-cutpoint tree can be updated in $O(\log n)$ amortized time per edge removal [15], we have that the complexity of the algorithm is $O(m \log n)$. For a more detailed description of the algorithm, please refer to [19].

3 Various applications

During the past decades, many algorithms have been proposed for drawing biconnected graphs by firstly *st*-orienting them, using various *st*-orientation algorithms. Hierarchical drawings [5], visibility representations [23] and orthogonal drawings [23] are areas in Graph Drawing where the *st*-orientation (of a certain longest path length) used can have a considerable impact on the aesthetics of the final drawing.

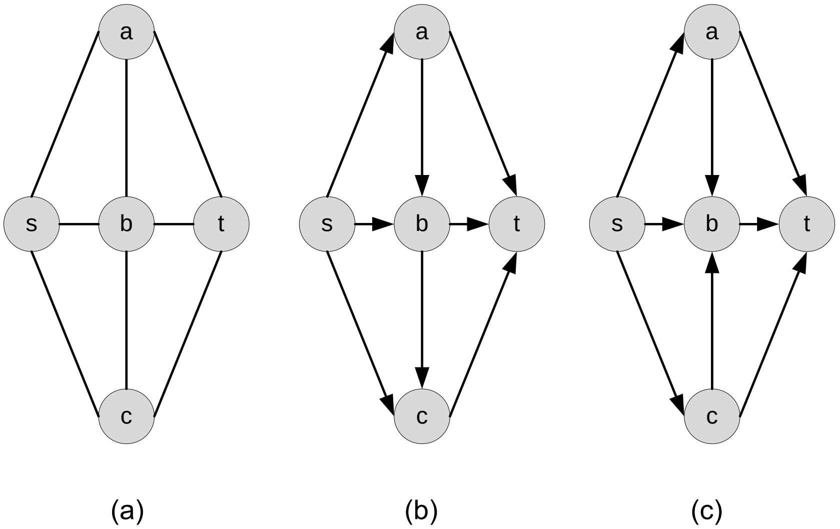

Figure 1: An undirected graph (a) and two *st*-orientations of it ((b),(c)).

For example, Figure 1 depicts an undirected graph G (Figure 1a) and two different *st*-orientations of it. The length of the longest path (from s to t) of the first *st*-orientation (Figure 1b) is equal to 4, while the second *st*-orientation (Figure 1c) has length of longest path from s to t equal to 3. Figure 2 shows two different longest path and visibility representation layouts for the two different *st*-orientations (1b), (1c) of the same graph (1a). The drawings have different characteristics, for example drawings in Figure 2a are "longer" and "thinner" whereas drawings in Figure 2b are "shorter" and "wider". Additionally, in the visibility representations layout, the required area is different. In the following we give an extensive description of these applications. We also specify in more detail how the properties of the output drawings depend on the choice of the specific *st*-ordering.

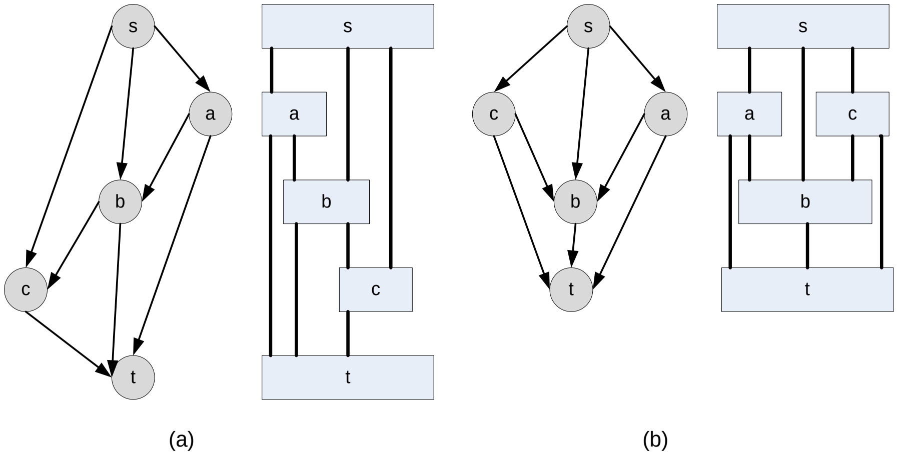

Figure 2: Longest path layering and visibility representation layouts for the *st*-orientation of Figure 1b (a) and for the one of Figure 1c (b).

3.1 Primal and dual *st*-orientations

st-planar graphs G are undirected planar graphs admitting a planar embedding that has two distinct nodes s, t on the outer face. If we *st*-orient such a graph, we can define a single orientation for the dual graph G^* which is also an s^*t^*-orientation (see Figure 3). The dual graph G^* can be computed from the primal graph G by replacing every *face* f of G with a node $v(f)$ in G^*. Edges that bound face f in G are adjacent edges of $v(f)$ in G^*. Moreover, the direction of edges in G uniquely defines a direction for edges in G^*. Details on the construction a dual s^*t^*-planar graph can be found in [5].

Both orientations are used in the visibility representation algorithms [23] in order to compute the coordinates of the segments that correspond to nodes of the primal and the dual graph. Actually, the length of the longest path of the dual s^*t^*-oriented graph determines the width of the geometric representation. Specifically, we have the following [5]:

Fact 1 *Let G be an st-planar graph. Let O be an st-orientation of G and O^* be the respective dual s^*t^*-orientation. Then there is a visibility representation*

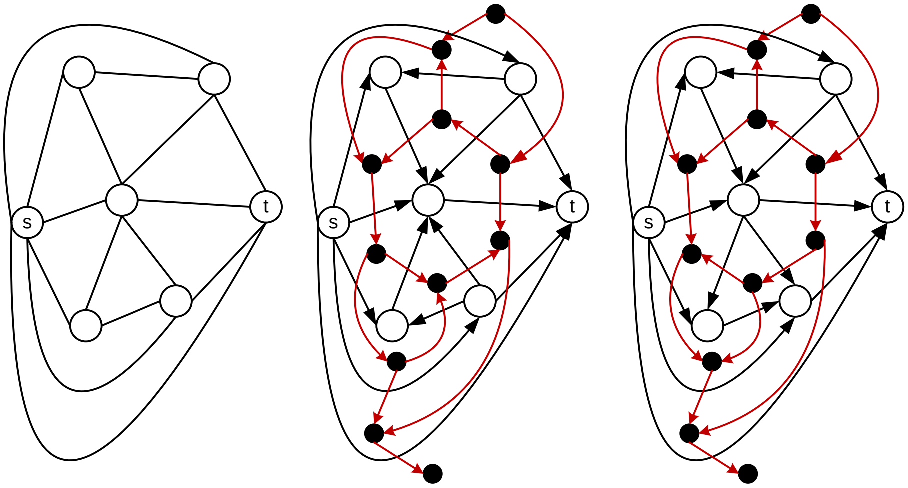

Figure 3: Constructing the dual graph for different values of parameter p ($p = 0, 1$), by running PAR-STN(p) on a triangulated undirected planar graph G_1 (first graph on the left). The second graph G_2 is the st-oriented graph (together with the dual G_2^*) produced after running MIN-STN ($p = 0$) where the longest path lengths are $\ell(G_2) = 4$ and $\ell(G_2^*) = 10$. Finally, the third graph G_3 is the st-oriented graph (together with the dual G_3^*) produced after running MAX-STN ($p = 1$) where the longest path lengths are $\ell(G_3) = 6$ and $\ell(G_3^*) = 8$.

of G of area $\ell(O) \times \ell(O^*)$.

Detailed definitions of visibility representations, st-planar graphs and the construction of their respective dual s^*t^*-planar graphs can be found in [5].

The questions that arise now are natural. What is the impact of parameter p on the length of the longest path of the dual s^*t^*-oriented graph G^* of an st-planar graph G, which (the graph G) has been st-oriented with PAR-STN(p)? Intuitively, we would expect that the length of the longest path of the dual graph G^* will grow inversely proportional to the longest path length of the primal graph G (see Figure 3). As we will see, this is not always the case.

In Figure 3, we demonstrate the impact of parameter p on the longest path length of the dual graph of a produced st-orientation. We compute two st-orientations, G_2 and G_3, of a triangulated planar graph, and also the respective dual graphs G_2^* and G_3^*. G_2 is computed by using PAR-STN(0), while G_3 is computed by using PAR-STN(1). Note that $\ell(G_2) + \ell(G_2^*) = \ell(G_3) + \ell(G_3^*) = 14 = 2n$.

In the following, we present a special class of planar graphs where this impact can be quantified in a more formal way and be translated into a considerable saving of area of the respective visibility representations, computed by using different longest path length st-orientations.

3.2 A special class of planar graphs

Definition 1 *Let $n \geq 5$. We define an n-path planar graph $G_n = (V, E)$ to be the planar graph that consists of a path $P = v_2, v_3, \ldots, v_{n-1}$ of $n-2$ nodes and two other nodes v_1, v_n such that $\{v_1, v_i\} \in E$, $\{v_i, v_n\} \in E$, for all $i = 2, \ldots, n-1$ and $\{v_1, v_n\} \in E$.*

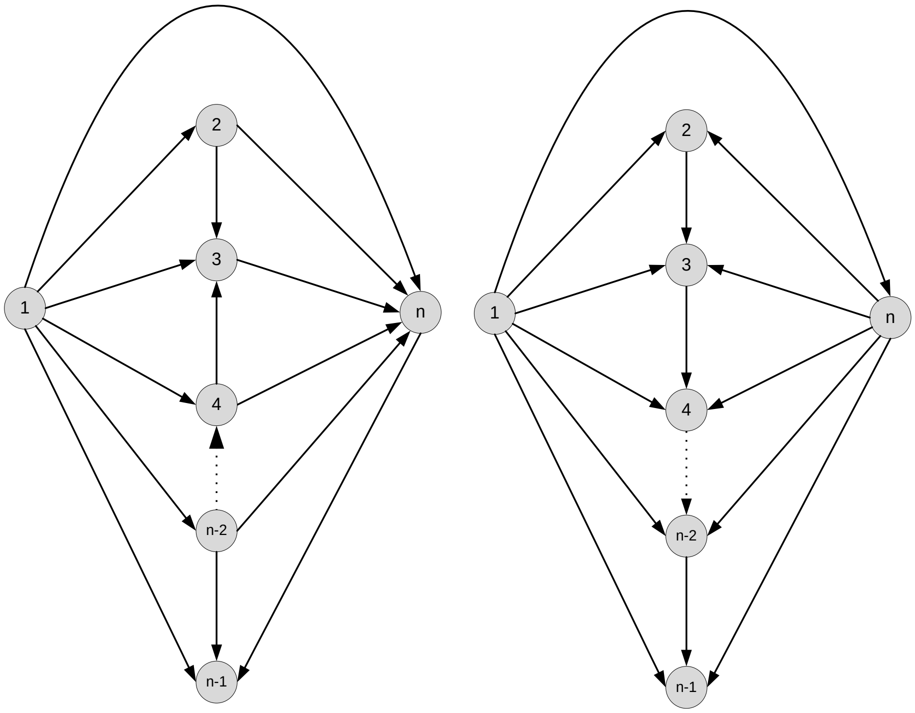

Figure 4: Two v_1v_{n-1}-orientations of an n-path planar graph G_n, G_n^{min} (left) and G_n^{max} (right). G_n^{min} is the minimum longest path length v_1v_{n-1}-orientation while G_n^{max} is the maximum longest path length v_1v_{n-1}-orientation.

In Figure 4, an n-path planar graph is depicted (actually an n-path planar graph is defined as the underlying undirected graph of Figure 4). Its source is node v_1 and its sink is node v_{n-1}. Note that an $(n+1)$-path planar graph G_{n+1} can be obtained from an n-path planar graph G_n by adding a new node and connecting it with nodes v_1, v_2 and v_n (nodes v_1 and v_n are the rightmost and leftmost nodes of G_n's embedding in Figure 4). In Figure 4, two v_1v_{n-1}-orientations of G_n are depicted. On the left, the orientation of the minimum longest path length is depicted while on the right the orientation of the maximum longest path length is depicted. We call those two v_1v_{n-1}-orientations G_n^{min} (see Lemma 2) and G_n^{max} (see Lemma 3) respectively.

Note the difference between the two orientations of Figure 4: All edges belonging to the middle path of G_n^{min} successively change their direction. On the other hand, all edges belonging to the middle path of G_n^{max} have an orientation towards the sink of the graph, v_{n-1}. In the following lemmas, we say that a graph is a minimum (maximum) st-oriented graph if it is st-oriented and the length of the longest path from s to t is the minimum (maximum) over all possible st-orientations of the respective undirected graph.

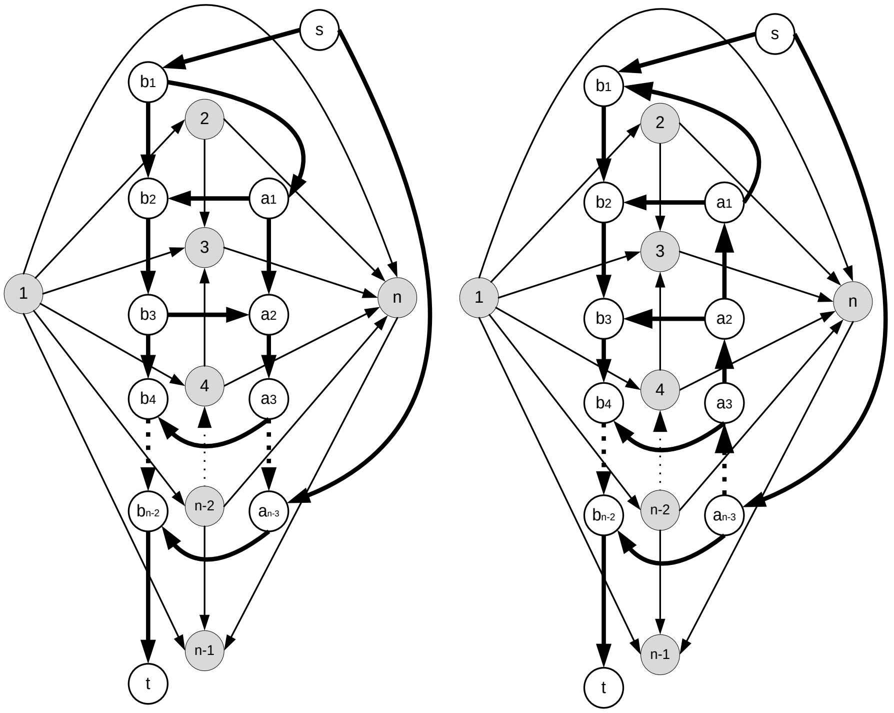

Figure 5: Dual orientations of G_n^{min} (left) and G_n^{max} (right), $G_n^{\mathsf{min},*}$ and $G_n^{\mathsf{max},*}$ respectively. $G_n^{\mathsf{min},*}$'s longest path is $s, b_1, a_1, b_2, b_3, a_2, a_3, b_4, b_5, a_4, a_5, \ldots, b_{n-2}, t$ whereas $G_n^{\mathsf{max},*}$'s longest path is $s, a_{n-3}, a_{n-2}, \ldots, a_1, b_1, b_2, \ldots, b_{n-2}, t$, which are both equal to $2n-4$ (see Theorem 2). Note that the dual source and the dual sink are embedded in the external primal face, following standard techniques described in the literature [5].

Before proceeding to prove some results for n-path planar graphs, we prove Lemma 2 that refers to minimum st-oriented n-path planar graphs, by using a well-known result by Gallai [8], which is along the same lines of the result given by Vitaver in [25] (note that this result is also used later in the paper to prove the NP-hardness of the problem of computing a minimum longest path length st-orientation):

Theorem 1 (Graph coloring and acyclic orientations) *Let G be an undirected biconnected graph. The length of the longest path of an acyclic orientation* *achieving the minimum longest path length over all acyclic orientations equals* $\chi(G)-1$*, where* $\chi(G)$ *is the chromatic number of* G*.*

Now we have the following:

Lemma 2 *For all* $n \geq 5$*,* G_n^{min} *is a minimum* v_1v_{n-1}*-oriented graph of an* n*-path planar graph* G_n*. Moreover, for this orientation, the length of the longest path is* $\ell(G_n^{\mathsf{min}}) = 4$*.*

Proof: Let R be the graph that is produced from G_n by removing nodes v_1 and v_{n-1} (and all their incident edges). The remaining graph has chromatic number 3, namely we color the nodes $v_2, \ldots, v_{n-2}$ with two different colors and we use a third color to color v_n, since v_n is incident to all nodes $v_2, \ldots, v_{n-2}$. Therefore, by Theorem 1 the acyclic orientation of R of minimum longest path length has length of longest path $2 = 3 - 1$. There are many acyclic orientations of R that achieve this longest path length, and here we give one of them: $(v_2, v_3), (v_4, v_3), (v_4, v_5), \ldots, (v_{n-2}, v_{n-3})$ (note that if $n-2$ is odd then the direction is (v_{n-3}, v_{n-2})) and (v_i, v_n) for all $i = 2, \ldots, n-2$. This directed graph is a subgraph of G_n^{min} and has a directed path of length 2 ending in v_n. Note that however, for any v_1v_{n-1}-orientation, the orientation of the prior-removed edges (i.e., the edges (v_1, v_i), (v_{n-2}, v_{n-1}) and (v_n, v_{n-1})) is pre-determined since v_1 and v_{n-1} are the source and the sink of the desired v_1v_{n-1}-orientation. Moreover, these oriented edges increase the longest path length by 2. Thus the length of the minimum longest path length *st*-orientation is $2 + 2 = 4$. Since G_n^{min} achieves that length, it follows that G_n^{min} is a minimum v_1v_{n-1}-oriented graph of an n-path planar graph G_n. □

We now continue with a similar lemma for the maximum *st*-oriented n-path planar graph.

Lemma 3 *For all* $n \geq 5$*,* G_n^{max} *is a maximum* v_1v_{n-1}*-oriented graph of an* n*-path planar graph* G_n*. Moreover, for this orientation, the length of the longest path is* $\ell(G_n^{\mathsf{max}}) = n - 1$*.*

Proof: For every n, G_n^{max} has a Hamiltonian path from v_1 to v_{n-1}, namely the path $v_1, v_n, v_2, v_3, \ldots, v_{n-1}$. Note that this directed path also creates no cycles. Therefore, G_n^{max} is a maximum v_1v_{n-1}-oriented graph of an n-path planar graph G_n. □

Let now $G_n^{\mathsf{min},*}$ (dual graph on the left in Figure 5), $G_n^{\mathsf{max},*}$ (dual graph on the right in Figure 5) be the dual graphs of G_n^{min} and G_n^{max} respectively. We have the following result:

Theorem 2 *Let* G_n *be an* n*-path planar graph and* G_n^{min} *and* G_n^{max} *be the minimum and the maximum* v_1v_{n-1}*-oriented graphs of* G_n*. Let also* $G_n^{\mathsf{min},*}$ *and* $G_n^{\mathsf{max},*}$ *be the dual graphs of* G_n^{min} *and* G_n^{max} *respectively. Then for all* $n \geq 5$*, it holds that* $\ell(G_n^{\mathsf{min},*}) = \ell(G_n^{\mathsf{max},*}) = 2n - 4$*.*

Proof: Note that the number of nodes of the dual graphs in Figure 5 is $2n-3$. For the case of $G_n^{\mathsf{min},*}$ of Figure 5, denote with $a_1, a_2, \ldots, a_{n-3}$ the sequence of dual nodes that lie on the right of the middle path of G_n^{min} and with $b_1, b_2, \ldots, b_{n-2}$ (the numbering is from the top to the bottom) the sequence of dual nodes that lie on the left of the middle path of G_n^{min} (see Figure 5). Also denote with s and t the dual source node and the dual sink node respectively. Then, for all n, one can always output a Hamiltonian path as the longest path of the dual orientation, namely the path

$$s, b_1, a_1, b_2, b_3, a_2, a_3, b_4, b_5, a_4, a_5, \ldots, b_{n-2}, t\,.$$

Therefore $\ell(G_n^{\mathsf{min},*}) = 2n-3-1 = 2n-4$. For the case of $G_n^{\mathsf{max},*}$, one can again output a different Hamiltonian path

$$s, a_{n-3}, a_{n-2}, \ldots, a_1, b_1, b_2, \ldots, b_{n-2}, t\,.$$

Therefore $\ell(G_n^{\mathsf{max},*}) = 2n-3-1 = 2n-4$. □

The above findings indicate that there is a significant impact of a different st-orientation of an n-path planar graph on the area of its visibility representation. By using the minimum st-orientation, we will need an area equal to

$$\ell(G_n^{\mathsf{min}})\ell(G_n^{\mathsf{min},*}) = 4(2n-4) = 8n-16 = \Omega(n)\,. \tag{1}$$

If we use the maximum st-orientation, we will need an area equal to

$$\ell(G_n^{\mathsf{max}})\ell(G_n^{\mathsf{max},*}) = (n-1)(2n-4) = 2n^2-6n+4 = \Omega(n^2)\,. \tag{2}$$

Thus we can reduce the area by a factor of n. Note that while $\ell(G_n^{\mathsf{max}}) + \ell(G_n^{\mathsf{max},*}) = 3n-5 > 2n$, it is $\ell(G_n^{\mathsf{min}}) + \ell(G_n^{\mathsf{min},*}) = 2n \leq 2n$.

An observation on n-path planar graphs. As we saw in the computation of the st-orientations of n-path planar graphs, it is the case that the st-orientation of the primal graph does not seem to significantly influence the longest path length of the dual st-orientation. Namely, both for constant (4) and linear $(n-1)$ length of longest path of the primal graph, the length of the dual longest path is the same, i.e., equal to $2n-4$. Moreover, we can derive a linear lower bound on the length of the longest path of the dual st-orientation. This is mostly due to the large degree $(O(n))$ of the source of the primal graph, which translates into long dual directed paths. We summarize our discussion with the following theorem.

Theorem 3 *Let G_n be an n-path planar graph and $\mathcal{G}_n$ be any v_1v_{n-1}-oriented graph of G_n. Let also $\mathcal{G}_n^*$ be the respective dual graph of $\mathcal{G}_n$. Then it holds $\ell(\mathcal{G}_n^*) \geq n-2$.*

Proof: The source v_1 of G_n has degree $n-2 = O(n)$. Therefore, for every st-orientation of G_n, the edges adjacent to v_1 are always oriented (fixed) from v_1 to v_j for $j = 2, \ldots, n-1$. This causes the creation of a dual path that contains at least $n-2$ edges, starting from the dual source and ending to the dual sink, i.e., the path $s, b_1, b_2, \ldots, b_{n-2}, t$ in Figure 5. Thus every dual longest path is at least as long as this directed path and therefore $\ell(\mathcal{G}_n^*) \geq n-2$. □

3.3 Graph coloring and longest path

In this section we investigate the application of longest path parameterized *st*-orientations in two NP-hard problems, graph coloring and longest path. We see that computing *st*-orientations of certain longest path length is equivalent to producing heuristic solutions to these difficult problems. Based on Theorem 1 we can prove that computing an *st*-orientation with minimum longest path length is NP-hard.

Theorem 4 *Let G be an undirected biconnected graph and s, t be two of its nodes. Computing an st-orientation with minimum longest path length from s to t is NP-hard.*

Proof: We reduce the graph coloring problem, a well-known NP-hard problem [9], to the problem of computing an *st*-orientation with minimum longest path length from s to t. Let G be an instance of the graph coloring problem. We give a polynomial-time reduction that produces a graph G' such that computing a minimum longest path length *st*-orientation from some s to some t of G' gives an algorithm for computing the chromatic number of G. To produce G', introduce two new nodes s, t and connect them with every node of G. These new nodes will serve as the source and the sink of G' (see Figure 6). An *st*-orientation of G' that has the minimum longest path length implies an acyclic orientation of G (which is not necessarily an *st*-orientation of G) that has the minimum longest path length, say l, over all acyclic orientations of G (otherwise the computed *st*-orientation of G' would not have the minimum longest path length). Therefore, by Theorem 1, the chromatic number of G is equal to $l+1$. Therefore computing an *st*-orientation with minimum longest path length is NP-hard. □

Theorem 4 shows a connection of graph coloring and minimum longest path length *st*-orientations. By producing a good solution for the minimum longest path length *st*-orientation problem we may obtain a good solution for the graph coloring problem. To see that, suppose we are given a graph $G = (V, E)$ and we want to compute a coloring of G. We use the reduction in the proof of Theorem 4 as an algorithm to do that: We produce the graph $G' = (V', E')$ by adding two extra nodes s, t and edges from s to all the nodes of G and from t to all the nodes of G. We apply an algorithm that computes an orientation to G' with source s and sink t, resulting in an *st*-oriented graph F with longest path length l. Then, by labelling the nodes of the graph with the respective longest path length (i.e., the label of each node v is the length of the longest path from the source s to v) and by assigning different colors to different longest path lengths, we can color G using $l-1$ colors, since two neighboring nodes with the same color would imply a cycle.

Here we note that, from a practical viewpoint, if we were to use an *st*-orientation to compute a good coloring of a graph, we would better use an *st*-orientation of small longest path length. We illustrate this thought with an example. Suppose we want to compute a coloring of a ring $G = (V, E)$ consisting of 6 nodes. Clearly $\chi(G) = 2$. If we add the nodes s, t, the undirected edges (s, i), (t, i), for all $i \in V$ and apply MIN-STN to it, we produce the st-oriented graph of Figure 6.

Note that all nodes lying on the ring have a longest path length from s either 1 or 2. The longest path length from s to t is 3, and thus we need $3 - 1 = 2$ colors to color G. Actually, this is the chromatic number of G. Hence, we have computed the chromatic number of G by applying MIN-STN to G'.

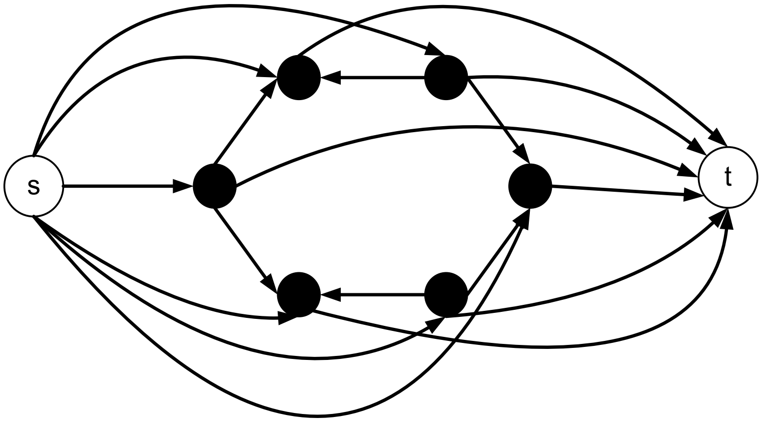

Figure 6: Combining graph coloring and st-orientations.

Recently, computing an st-orientation of minimum longest path length was proved to be NP-hard also for the case of 2-connected plane graphs [21]. This shows that the problem has an inherent difficulty and does not get easier even for the plane case. Therefore the need for heuristics for solving the problem is now more prominent, since most of real life applications refer to planar graphs.

Analogously, computing an st-orientation for general graphs of the maximum longest path length is NP-hard [19].

Theorem 5 *Let G be an undirected biconnected graph and s, t be two of its nodes. Computing an st-orientation of the maximum longest path length from s to t is NP-hard.*

Therefore, by using MAX-STN we can produce an st-orientation of high longest path length which will also be a heuristic solution to the undirected graph longest path problem, a well known NP-hard problem [9]. The use of st-orientations in solving these difficult problems in practice will be presented in Section 4.

3.4 Orthogonal drawings

Finally, we present some applications of the st-orientations in orthogonal drawings. An orthogonal drawing is a drawing of a graph G such that the nodes of G

are placed on the points of the grid and the edges of G are drawn as a sequence of horizontal and vertical segments. There has been a considerable amount of work on orthogonal drawings. To name few, [17] explores the case of graphs of maximum degree-4 and [18] gives algorithms for embedding high degree graphs on the orthogonal grid. Both works use an st-orientation as a preprocessing step. In this section we explore the impact that different st-orientations can have in the area of the orthogonal drawing of a maximum degree-four graph, embedded on the orthogonal grid by using algorithms in [17], which are area-efficient algorithms producing drawings of area at most $0.76n^2$.

After an initial st-orientation on the input graph is computed, the algorithm in [17] divides the vertices of the graph into four categories (according to their incoming and outgoing edges): column pairs (one column pair contains two nodes), unassigned degree-2 nodes, unassigned degree-3 nodes and row pairs (one row pair contains two nodes). This categorization, which is a function of the st-orientation, is used by the algorithm in order to construct the final drawing. Let now p_1, p_2, p_3 and k_2 be the number of column pairs, unassigned degree-2 nodes, unassigned degree-3 nodes and row pairs respectively. Then it can be proved [17] that the area occupied by the final orthogonal drawing is

$$\left(n + 1 - p_1 - p_2 - \frac{p_3}{2}\right) \times (n + 1 - k_2)\,.$$

Therefore the area of the drawing is dependent on the initial st-orientation. This means that given the same graph to be embedded on the orthogonal grid, if we use different st-orientations in the beginning we end up with correct drawings (i.e., drawings that satisfy certain properties) which however occupy different areas. In Section 4, we experimentally investigate the impact of different st-orientations on the area of orthogonal drawings computed with algorithms in [17].

4 Experimental results

In this section we present experimental results concerning the execution of the algorithm for producing solutions mainly for visibility representations and for the graph coloring problem.

We also show implications of the algorithm in orthogonal drawings [17]. The algorithms were implemented in Java, using the Java Data Structures Library (`http://www.jdsl.org`) [10].

4.1 Dual st-orientations of planar graphs

In this section we present some results for low-density and maximum density (triangulated) st-planar graphs.

Table 1: Primal and dual longest path length for low density st-planar graphs.

	p=0		**p=0.5**		**p=1**		$l \times l^*$		
n	l	l^*	l	l^*	l	l^*	**p=0**	**p=0.5**	**p=1**
100	54	31	62	30	75	19	1674	1860	1425
200	111	36	138	26	173	14	3996	3588	2422
300	149	39	199	32	251	20	5811	6368	5020
400	190	112	257	81	346	19	21280	20817	6574
500	165	129	339	73	454	16	21285	24747	7264
600	302	118	378	120	462	32	35636	45360	14784
700	412	208	502	130	626	17	85696	65260	10642
800	447	156	565	156	717	19	69732	88140	13623
900	396	178	501	108	664	32	70488	54108	21248
1000	619	188	757	118	884	41	116372	89326	36244
1100	438	287	649	221	841	31	125706	143429	26071
1200	596	283	832	196	1014	43	168668	163072	43602
1300	756	361	970	182	1150	34	272916	176540	39100
1400	599	497	1010	315	1260	29	297703	318150	36540
1500	835	345	1047	281	1281	46	288075	294207	58926
1600	617	599	865	313	1407	36	369583	270745	50652
1700	671	327	963	296	1100	44	219417	285048	48400
1800	926	499	1258	292	1635	32	462074	367336	52320
1900	681	685	1241	333	1536	35	466485	413253	53760
2000	1147	337	1503	239	1803	44	386539	359217	79332
2500	1010	712	1471	511	2146	33	719120	751681	70818
3000	1652	683	2114	555	2608	44	1128316	1173270	114752
3500	1486	695	2804	695	2804	49	1032770	1948780	137396
4000	1500	1115	2271	745	3619	50	1672500	1691895	180950
5000	2101	1358	2500	763	3482	59	2853158	1907500	205438

Low-density ($m = 1.5n$) st-planar graphs of n nodes are constructed as follows: Initially a node is chosen at random to be the root of the tree, which is also the *source* of the orientation we are going to compute. Then we connect the current tree (initially it only consists of the root) by inserting an edge between a randomly chosen node of the current tree and a node that does not belong to the current tree and which is again chosen at random. We execute the same procedure until all nodes are inserted into the tree. Then we connect the leaves of the tree following a preorder numbering so that all crossings are avoided. Note that all the non-tree edges lie on the outer face of the graph embedding. The sink of the orientation is set to be the last node that the aforementioned procedure encounters (after connecting the leaves of the tree).

Maximum density ($m = 3n - 6$) st-planar graphs were computed with a certain software for graph algorithms and visualization called PIGALE[2], which offers planar graph generators that are based on the algorithms by Schaeffer [22].

From Table 1, it is clear that the primal and the dual longest path length are inversely proportional for various values of the parameter p. We have used the

[2] `http://pigale.sourceforge.net/`.

Table 2: Primal and dual longest path length for max-density st-planar graphs.

	p=0		**p=0.5**		**p=1**		$l \times l^*$		
n	l	l^*	l	l^*	l	l^*	**p=0**	**p=0.5**	**p=1**
109	31	167	75	95	100	74	5177	7125	7400
222	42	374	105	216	151	129	15708	22680	19479
310	44	503	186	319	280	163	22132	59334	45640
436	100	524	248	412	397	178	52400	102176	70666
535	98	785	240	534	402	293	76930	128160	117786
678	80	1019	382	449	625	195	81520	171518	121875
763	144	1114	385	780	691	241	160416	300300	166531
863	105	1286	453	791	767	270	135030	358323	207090
998	83	1419	425	862	846	340	117777	366350	287640
1117	109	1561	551	902	1013	208	170149	497002	210704
1302	134	2024	704	1154	1173	451	271216	812416	529023
1410	122	2120	730	835	1291	298	258640	609550	384718
1501	119	2203	784	1073	1403	224	262157	841232	314272
1638	110	2487	833	1436	1477	263	273570	1196188	388451
1719	131	2550	856	1661	1555	515	334050	1421816	800825
1825	180	2729	886	1391	1618	353	491220	1232426	571154
1990	208	2339	1013	1581	1773	400	486512	1601553	709200
2089	136	3095	1002	1648	1789	347	420920	1651296	620783
2159	142	3238	930	1816	1823	445	459796	1688880	811235
2213	162	3400	1093	2082	2008	551	550800	2275626	1106408
2268	148	3136	952	1666	1887	336	464128	1586032	634032
2413	154	3033	971	1968	1631	513	467082	1910928	836703
4323	356	5852	2238	3589	3957	841	2083312	8032182	3327837
5102	525	7155	2597	4473	4582	1139	3756375	11616381	5218898

values $p = 0, 0.5, 1$, as the most representative ones. Additionally, it seems that for low density st-planar graphs the sum $l + l^*$ is no more that n (the number of the primal graph nodes), something that does not hold in general, as will be discussed later in this section.

The last three columns of Table 1 show the product $l \times l^*$. This is actually the area that is needed in order to construct a visibility representation of the given graph using the algorithms proposed in [23]. The impact of the parameter p on the area is very evident. The savings in the area for different values of the parameter p is clear and actually for low density it is preferable to use the parameter $p = 1$. In Figures 7, 8, we present a plot of the products $l \times l^*$ as a function of the size of the graph and the value of the parameter p for both low density graphs and maximum density graphs. In Table 2, we show the same results for the other class of planar graphs, the triangulated planar graphs. In this case the sum $l + l^*$ is higher but always less than $2n$, n being the number of nodes of the primal planar graph.

4.2 Visibility representations

In this section we present results concerning various visibility representations that can be output by using our algorithm for different values of parameter p. We recall that the area of a visibility representation is totally dependent on

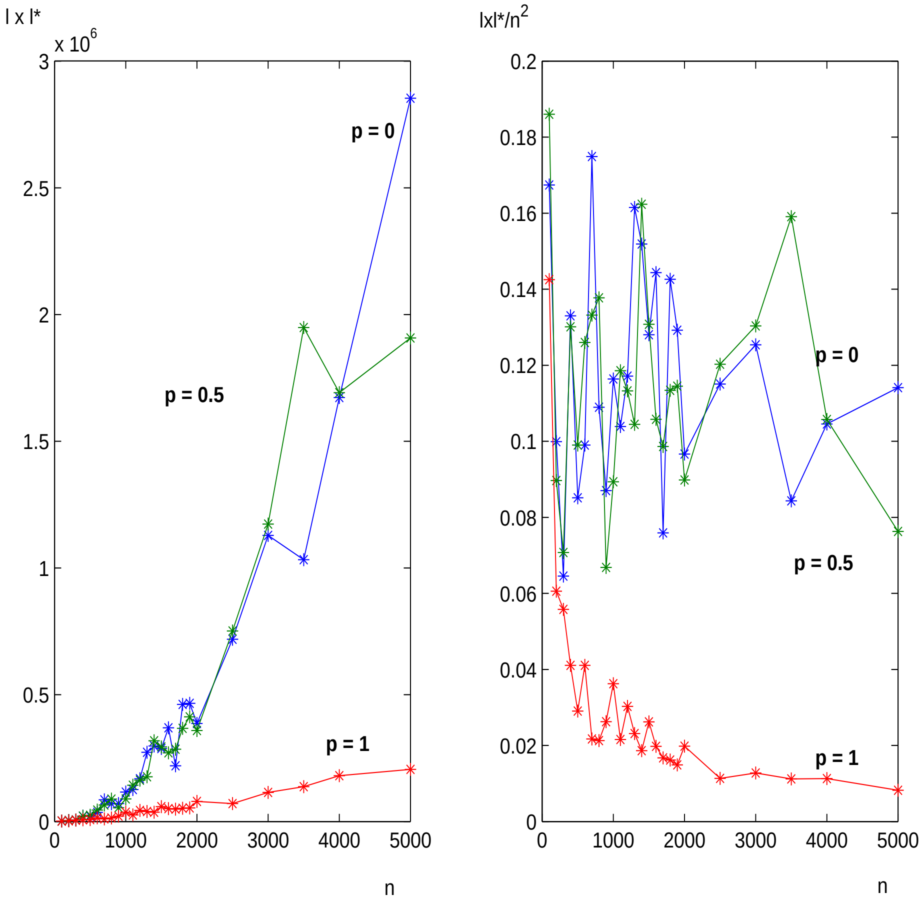

Figure 7: Absolute (left) and normalized (divided by n^2) (right) results for visibility representation area requirement for different values of the parameter p and low density planar graphs.

the *st*-orientation used (see Fact 1). When the input is an undirected graph G, we run PAR-STN(p) for different values of p, producing the respective *st*-orientations of the graph. Then we use the implementation of a visibility representation algorithm offered by the software PIGALE[3] to compute and visualize the respective visibility representation.

In Figure 9, we show 3 visibility representation frames of a 21-path planar graph. We use 3 different *st*-orientations computed for $p = 0, 0.5, 1$. Note that the length of the dual longest path of all three *st*-orientations is consistent

[3] We are indebted to Hubert de Fraysseix for kindly offering to provide us with a version of the visibility representation software PIGALE (`http://pigale.sourceforge.net/`) that uses as input not only the undirected graph, but also an *st*-orientation of it. The initial version of PIGALE would use an *st*-numbering with no specific properties.

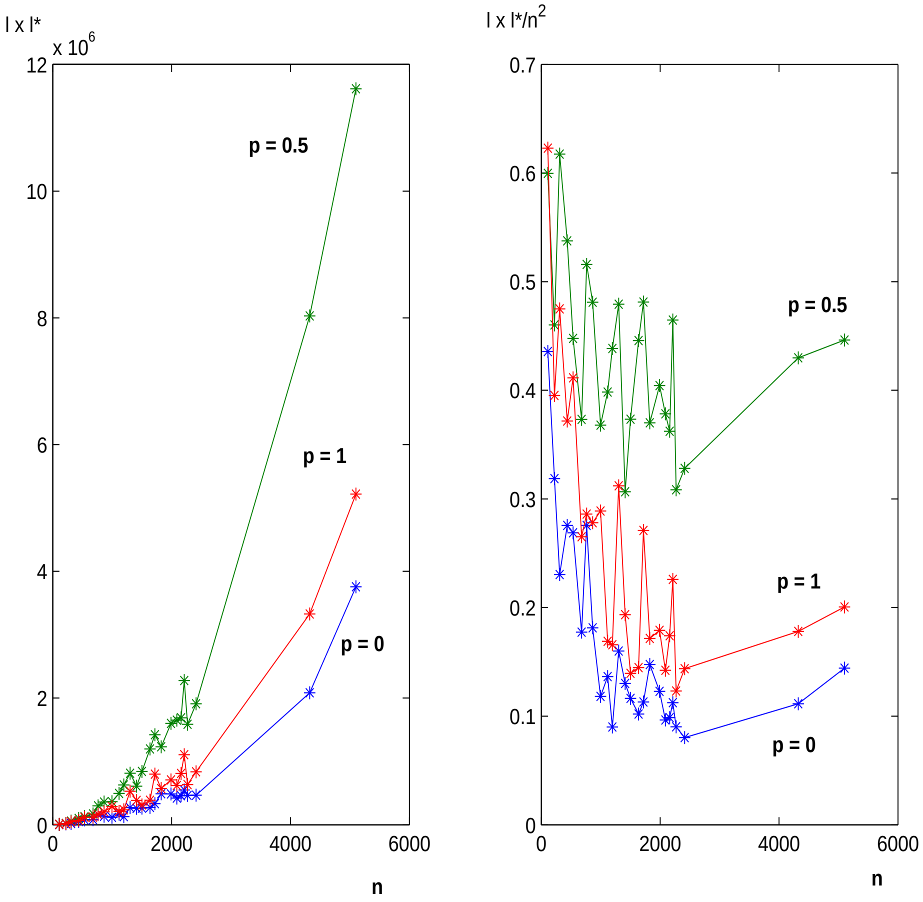

Figure 8: Absolute (left) and normalized (divided by n^2) (right) results for visibility representation area requirement for different values of the parameter p and maximum density planar graphs. The parameter $p = 0$ (low longest path *st*-oriented graphs) is clearly preferable.

with Theorem 3, since 21, the degree of the primal source is always less than the length of the dual longest paths, i.e., $21 \leq 38, 29, 38$ for $p = 0, 0.5, 1$ respectively. Also, the length of the longest path of the primal graph varies significantly for different values of p. Namely, and in consistence with experimental results on Hamiltonian graphs presented in [19], it increases as parameter p gets larger. This notable variance of the length of the primal longest path (and also the smaller variance of the length of the dual longest path) causes the area of the different visibility representations to vary significantly, namely being equal to $152, 348, 760$ for $p = 0, 0.5, 1$ respectively. This means that the area of the visibility representations increases in proportion with the length of the primal longest path—in theory, this can be the case only when the length of the dual longest path remains proportional to n, while parameter p changes, since the area equals the product of the two—. This is in accordance with the observation made in Section 3.2 (Equations 1 and 2), where it is shown that the *st*-orientations of small longest path length produce more area-efficient visibility representations, for the case of n-path planar graphs. Finally note that the dual longest path length does not change in a monotone way as p increases (i.e., at some point it reaches a minimal value and then it increases again).

In Figure 10, visibility representations of low-density *st*-planar graphs are presented. Here, parameter p does not have a big influence on the length of the longest path. The length of the dual longest path has a more notable variance (especially for the case of $p = 0.5, 1$). We believe that the primal longest path length does not vary due to the small degree (due to the small density) of the nodes of the graph: This results in a smaller number of existent *st*-orientations which in turn does not give enough flexibility to PAR-STN(p), forcing it to orient most of the edges in the same way. Although the variance of the primal longest path length is not that big, there is no obvious way to predict (and argue about) the variance of the respective dual longest path length—even if a direction change of a primal edge implies a direction change of dual edge—, since our *st*-orientation algorithm is executed on the *primal* graph. This is why the dual longest path length varies more significantly than the primal longest path length. Finally, for these graphs only, it is interesting that the minimum area for visibility representations is achieved for $p = 1$. This is due to the non-variance of the primal longest path and therefore the area becomes a function of the dual longest path which, in this case, decreases in a monotone way as p increases.

In Figure 11, we present the visibility representations ($p = 0, 0.5, 1$) of an 85-node triangulated *st*-planar graph. This triangulated graph was produced with PIGALE. Again here the dual longest path length does not change in a monotone way as p increases. However, the most important observation that can be made in this case is the fact that, unlike the case of the n-path planar graphs, the “minimal” *st*-orientation here does not achieve such a low longest path length (e.g., $O(1)$). We believe this is due to the fact that the embedding of those random planar graphs span a wider area—which enforces longer directed paths—than the one occupied by n-path planar graphs, and also, due to the lack of symmetries. Finally, note that again, the “minimal” (i.e., $p = 0$) *st*-orientation produces the most area-efficient visibility representation.

Finally, in Figure 12, we present some visibility representations frames produced by *st*-orienting a grid graph. As we can see here the parameter has a considerable influence on the length of the primal longest path, which grows analogously with parameter p. Actually, for $p = 1$, the algorithm computes the Hamiltonian path. We believe that this is the case because the graph has certain symmetries and structure, which is also observed in other graphs with well defined structure such as n-path planar graphs and *st*-Hamiltonian graphs [19]. Again, the parameter that gives the most area-efficient visibility representation is $p = 0$.

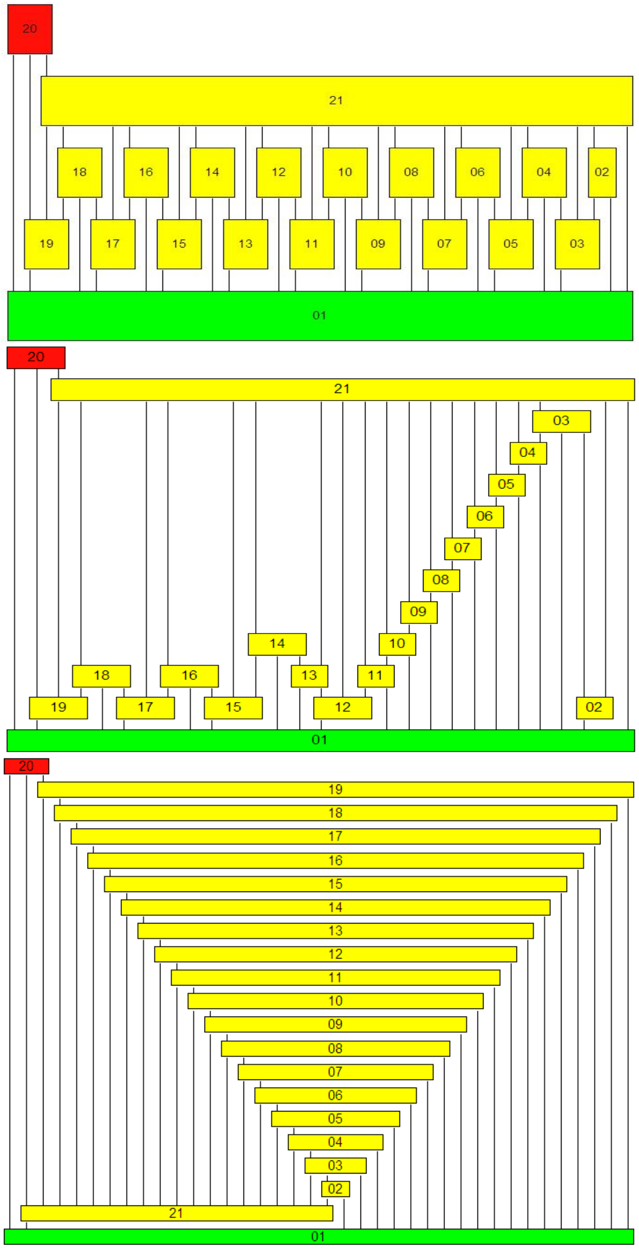

Figure 9: Visibility representations of a 21-path planar graph for different *st*-orientations produced with $p = 0, 0.5, 1$. The respective areas ($\mathsf{height} \times \mathsf{width}$) are $4 \times 38 = 152$, $12 \times 29 = 348$ and $20 \times 38 = 760$.

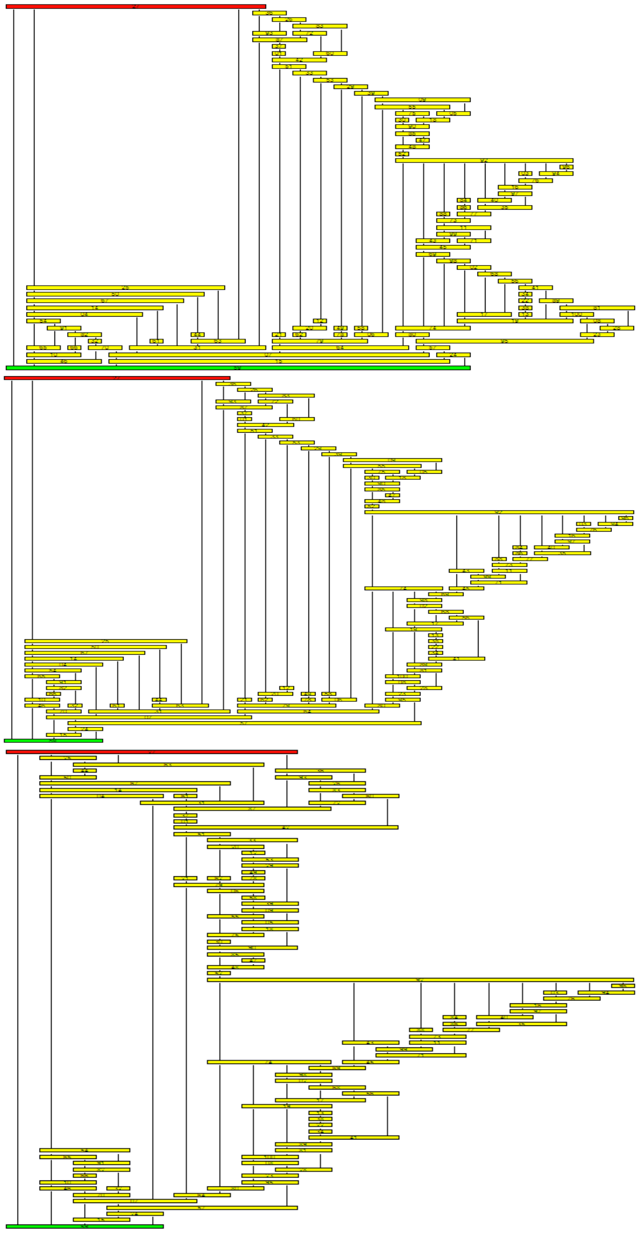

Figure 10: Visibility representations of an 100-node st-planar graph of low density ($m = 1.5n$) for different st-orientations produced with PAR-STN(p) ($p = 0, 0.5, 1$). The respective areas (height × width) are $54 \times 31 = 1674$, $62 \times 30 = 1860$ and $75 \times 19 = 1425$.

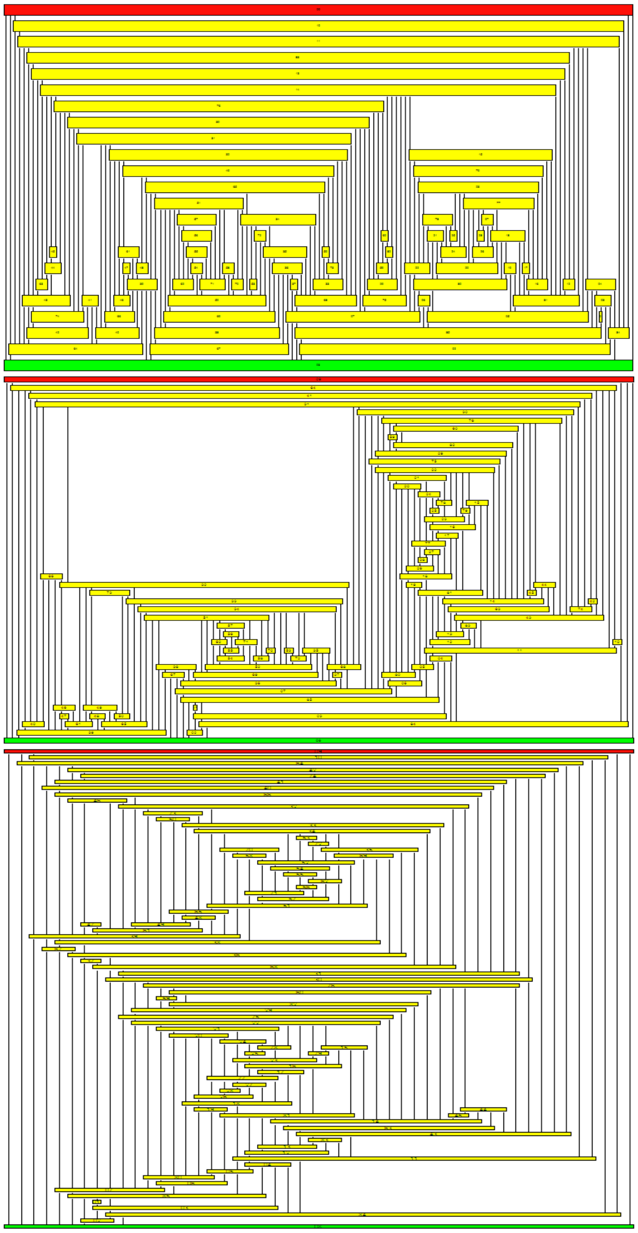

Figure 11: Visibility representations of an 85-node triangulated st-planar graph for different st-orientations produced with PAR-STN(p) ($p = 0, 0.5, 1$). The respective areas ($\mathsf{height} \times \mathsf{width}$) are $22 \times 49 = 1078$, $49 \times 61 = 2989$ and $66 \times 49 = 3234$.

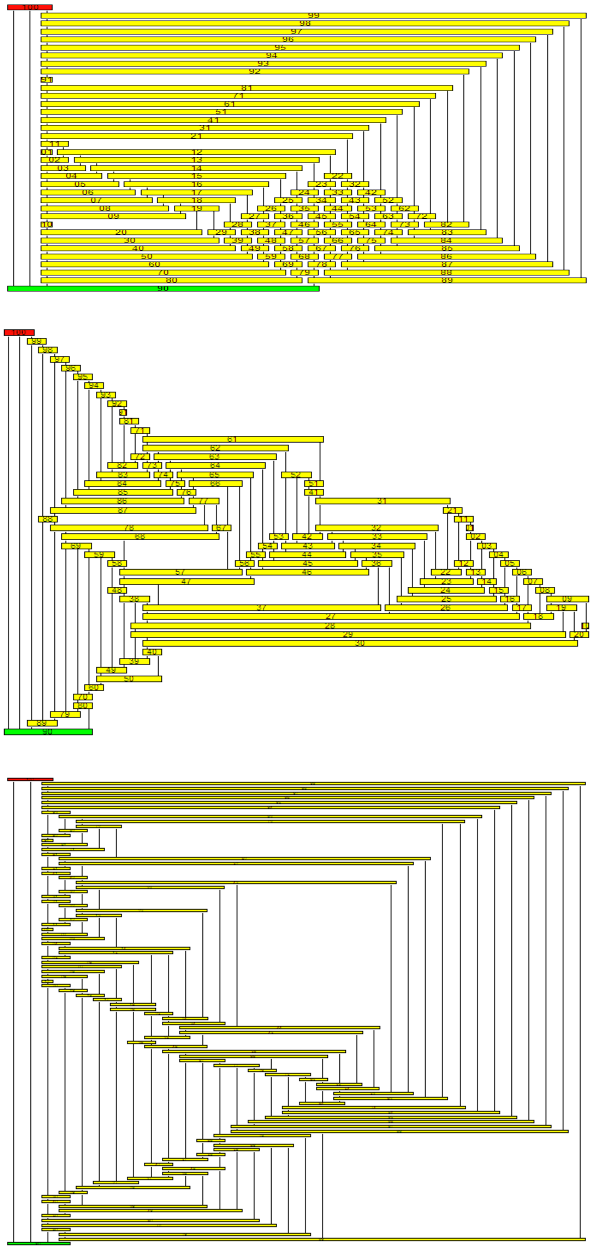

Figure 12: Visibility representations of a 10 × 10 grid graph for different *st*-orientations produced with PAR-STN(p) ($p = 0, 0.25, 1$). The respective areas (height × width) are $35 \times 34 = 1190$, $46 \times 50 = 2300$ and $99 \times 34 = 3366$.

4.3 Graph coloring

In this section we present some experimental results concerning the use of MIN-STN in coloring graphs using the method described in Section 3.3.

Table 3: Almost optimal coloring computed by MIN-STN.

file name	**n**	**m**	**optimal coloring**	**MIN-STN coloring**
myciel6.col	95	755	7	7
myciel5.col	47	236	6	6
myciel4.col	23	71	5	5
myciel3.col	11	20	4	4
games120.col	120	368	9	9
jean.col	80	254	10	10
huck.col	74	301	11	11
zeroin.i.1.col	211	4100	49	49
mulsol.i.5.col	186	3973	31	31
mulsol.i.4.col	185	3946	31	31
mulsol.i.3.col	184	3916	31	31
mulsol.i.2.col	188	3885	31	31
mulsol.i.1.col	197	3925	49	49
inithx.i.3.col	621	13969	31	31
inithx.i.1.col	864	18707	54	54
fpsol2.i.3.col	425	8688	30	30
fpsol2.i.1.col	496	11654	65	65
myciel7.col	191	2360	8	9
miles250.col	128	387	8	9
david.col	87	406	11	12
anna.col	138	493	11	12
zeroin.i.3.col	206	3540	30	31
zeroin.i.2.col	211	3541	30	31
inithx.i.2.col	645	13979	31	32

We test the algorithm's performance on known benchmarks which are available at `http://mat.gsia.cmu.edu/COLOR/instances.html`. For some graphs (see Table 3) we obtained very good results, computing essentially an almost optimum coloring. Actually, for the first 17 benchmark graphs G of Table 3, MIN-STN computes the chromatic number $\chi(G)$. Note that graphs `myciel*.col` are difficult to solve[4]: They are based on the Mycielski transformation, they are triangle free (clique number 2) but the chromatic number increases with the problem size. Still, our algorithm computes an optimum coloring for them. For the last 7 benchmark graphs of Table 3, MIN-STN computes a coloring equal to $\chi(G)+1$.

In Table 4, we show the results for some benchmark graphs for which MIN-STN did not perform so well. Some of these graphs, e.g., the graphs `queen*.col`,

[4] See `http://mat.gsia.cmu.edu/COLOR/instances.html#XXMYC`.

are constructed as follows. Given an $n \times n$ chessboard, a queen graph is a graph on n^2 nodes, each corresponding to a square of the board. Two nodes are connected by an edge if the corresponding squares are in the same row, column, or diagonal. Although these graphs have certain structure and symmetry, which seemed to be an advantage for graphs we studied in Section 4.2, our algorithm fails to compute optimum colorings. We believe this is due to the large degree (i.e., $O(n)$) of the nodes of these graphs.

Table 4: Relatively good coloring computed by MIN-STN.

file name	**n**	**m**	**optimal coloring**	**MIN-STN coloring**
queen8_12.col	96	1368	12	15
queen7_7.col	49	476	7	10
queen6_6.col	36	290	7	9
queen5_5.col	25	160	5	7
miles500.col	128	1170	20	23
homer.col	561	1629	13	15
fpsol2.i.2.col	451	8691	30	32

4.4 Orthogonal drawings

The pairing technique of [17] described in Section 3.4 has been implemented. We used the parameterized *st*-orientations for $p = 0, 0.5, 1$ and we present some experimental results in Table 5. The impact of the different *st*-orientations is not very clear in orthogonal drawings, as indicated in Table 5. For example the width and the height of the orthogonal drawings are not considerably influenced for various values of parameter p. However, for the algorithm described in [17], where the area upper bound is roughly $0.76n^2$, we are able to produce *st*-orientations that produce drawings of an area equal to $0.68n^2$ or less. The test graphs that we use are constructed by inserting random edges to a node set of n nodes, making sure every time that the degree of every node does not exceed four.

From the results in Table 5, we observe that the most area-efficient orthogonal drawings are achieved for $p = 1$. Note that this is the case since, although $p = 1$ produces *st*-orientations of large longest path length, the area of the drawing is not an immediate function of the length of directed paths in the *st*-oriented graph but of the number of outgoing and incoming edges of each node in the produced *st*-oriented graph. Therefore we cannot report some direct connection of length of longest paths with the area of the orthogonal drawings apart from this experimental evidence. However, it would be interesting to see, from a theoretical point of view, in which way different longest path length *st*-orientations influence the number of row and column pairs produced with the pairing algorithm in [17].

Table 5: Area bounds for orthogonal drawings and different st-orientations. The values of the parameter p that is used for these experiments are $p = 0, 0.5, 1$.

n	**width** w			**height** h			$\frac{wh}{n^2}$		
	0	0.5	1	0	0.5	1	0	0.5	1
200	174	156	152	157	167	169	0.68	0.65	0.64
400	317	310	303	332	335	337	0.66	0.65	0.64
600	478	467	444	493	501	511	0.65	0.65	0.63
800	627	618	600	661	668	669	0.65	0.65	0.63
1000	790	742	728	819	848	850	0.65	0.63	0.62
1200	939	903	874	985	1009	1021	0.64	0.63	0.62
1400	1099	1052	1012	1146	1172	1191	0.64	0.63	0.61
1600	1240	1204	1166	1319	1346	1360	0.64	0.63	0.62
1800	1402	1363	1308	1479	1507	1525	0.64	0.63	0.62
2000	1527	1512	1444	1662	1673	1667	0.63	0.63	0.60

5 Conclusions and discussion

In this paper we present applications of a newly developed algorithm that manages to efficiently control the length of the longest path of an st-orientation. We especially study these applications from the viewpoint of Graph Drawing but also give evidence of the importance of parameterized st-orientations in producing heuristics for difficult problems such as graph coloring and longest path. There are two main conclusions that have been drawn, by carefully examining the experimental results.

First of all, the experimental results indicate that the length of the longest path of a graph G that is st-oriented with MAX-STN is always greater than or equal to the length of the longest path of G that is st-oriented with MIN-STN. Of course, this is something to be expected, since MAX-STN explores the graph in a "DFS" mode, thus producing "long" paths, while MIN-STN explores the graph in a "BFS" mode, thus producing "short" paths. We point out that, in our experiments, there was not a single case where this property was violated. It would be nice to find a formal proof for that property of our algorithm and we conjecture here that it should hold.

Secondly, the experimental results indicate that the length of the longest path of the primal st-oriented graph G grows inversely proportional to the length of the longest path of the respective dual st-oriented graph G^*. Moreover their sum never exceeds $2n$, where n is the number of the nodes of the graph (see Tables 1 and 2). However, this is not the case for all st-orientations of a given graph G. A counterexample is graph G_n^{max} with dual graph $G_n^{\mathsf{max},*}$ (as shown in Figure 5). This leads us to the following conjecture:

Conjecture 1 *For every n-node planar biconnected graph G, there exists an st-orientation O with a respective dual st-orientation O^* such that $\ell(O) + \ell(O^*) \leq 2n + c$, where c is a small constant.*

Acknowledgements

The authors thank Philip Klein, Franco P. Preparata, Roberto Tamassia and Mihalis Yannakakis for useful comments and discussions. They also thank Hubert de Fraysseix for his help on the visualization frames produced with his software PIGALE and the reviewers of JGAA for providing useful feedback that considerably improved the quality of this paper. This research was supported in part by INFOBIOMED code IST-2002-507585, the Greek General Secretariat for Research and Technology under program "ARISTIA" code 1308/B1/3.3.1/317/12.04.2002, the U.S. National Science Foundation under grant 0830149 and the Center for Geometric Computing and the Kanellakis Fellowship at Brown University. Most of the work was performed while the first author was with the University of Crete and ICS-FORTH. The views in this paper do not necessarily reflect the views of the sponsors.

References

[1] M. M. Akon, S. Asaduzzaman, M. S. Rahman, and M. Matsumoto. Proposal for *st*-routing. *Telecommunication Systems*, 25(3-4):287–298, 2004.

[2] F. Annexstein and K. Berman. Directional routing via generalized *st*-numberings. *SIAM Journal on Discrete Mathematics*, 13(2):268–279, 2000.

[3] U. Brandes. Eager *st*-ordering. In *Proc. European Symposium on Algorithms (ESA)*, pages 247–256, 2002.

[4] H. de Fraysseix, P. de Mendez, and P. Rosenstiehl. Bipolar orientations revisited. *Discrete Applied Mathematics*, 56:157–179, 1995.

[5] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs.* Prentice Hall, 1999.

[6] J. Ebert. *st*-ordering the nodes of biconnected graphs. *Computing*, 30(1):19–33, 1983.

[7] S. Even and R. Tarjan. Computing an *st*-numbering. *Theoretical Computer Science*, 2:339–344, 1976.

[8] T. Gallai. On directed paths and circuits. In *Proc. Theory of Graphs: International Symposium*, pages 215–232, 1967.

[9] M. Garey and D. Johnson. *Computers and intractability: A guide to the theory of NP-completeness.* W. H. Freeman and Company, 1979.

[10] M. T. Goodrich and R. Tamassia. *Data structures and algorithms in JAVA*. John Wiley & Sons, 2005.

[11] X. He and H. Zhang. Nearly optimal visibility representations of plane graphs. In *Proc. Colloquium on Automata, Languages and Programming (ICALP)*, pages 407–418, 2006.

[12] J. Holm, K. de Lichtenberg, and M. Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge and biconnectivity. *Journal of the ACM*, 48(4):723–760, 2001.

[13] J. Hopcroft and R. Tarjan. Efficient algorithms for graph manipulation. *Communications of the ACM*, 16:372–378, 1973.

[14] A. Lempel, S. Even, and I. Cederbaum. An algorithm for planarity testing of graphs. In *Proc. Theory of Graphs: International Symposium*, pages 215–232, 1967.

[15] T. Lukovszki and W.-B. Strothmann. Decremental biconnectivity on planar graphs. *Technical report* TR-RI-97-186, University of Paderborn, 1997.

[16] Y. Maon, B. Schieber, and U. Vishkin. Parallel ear decomposition search (EDS) and *st*-numbering in graphs. *Theoretical Computer Science*, 47:277–298, 1986.

[17] A. Papakostas and I. G. Tollis. Algorithms for area-efficient orthogonal drawings. *Computational Geometry: Theory and Applications*, 9:83–110, 1998.

[18] A. Papakostas and I. G. Tollis. Efficient orthogonal drawings of high degree graphs. *Algorithmica*, 26(1):100–125, 2000.

[19] C. Papamanthou and I. G. Tollis. Algorithms for computing a parameterized *st*-orientation. *Theoretical Computer Science*, 408:224–240, 2008.

[20] P. Rosenstiehl and R. Tarjan. Rectilinear planar layout and bipolar orientation of planar graphs. *Discrete and Computational Geometry*, 1:343–353, 1986.

[21] S. Sadasivam and H. Zhang. NP-completeness of *st*-orientations for plane graphs. In *Proc. Fundamentals of Computation Theory (FCT)*, pages 298–309, 2009.

[22] G. Schaeffer. Bijective census and random generation of eulerian planar maps with prescribed degrees. *Electronic Journal of Combinatorics*, 4(1), 1997.

[23] R. Tamassia and I. G. Tollis. A unified approach to visibility representations of planar graphs. *Discrete and Computational Geometry*, 1:321–341, 1986.

[24] R. Tarjan. Two streamlined depth-first search algorithms. *Fundamenta Informaticae*, 9:85–94, 1986.

[25] L. Vitaver. Determination of minimal coloring of vertices of a graph by means of boolean powers of the incidence matrix. *Dokl. Akad. Nauk. SSSR.*, 147:758–759, 1962.

[26] H. Zhang and X. He. On visibility representation of plane graphs. In *Proc. Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 477–488, 2004.

[27] H. Zhang and X. He. Improved visibility representation of plane graphs. *Computational Geometry*, 30(1):29–39, 2005.

[28] H. Zhang and X. He. Visibility representation of plane graphs via canonical ordering tree. *Information Processing Letters*, 96(2):41–48, 2005.

[29] H. Zhang and X. He. Optimal *st*-orientations for plane triangulations. *Journal of Combinatorial Optimization*, 17(4):367–377, 2009.