Università degli Studi di Roma Tre
Dipartimento di Informatica e Automazione
Via della Vasca Navale, 79 – 00146 Roma, Italy

Complexity Results for Three-dimensional Orthogonal Graph Drawing

Maurizio Patrignani

RT-DIA-94-2005 **Febbraio 2005**

Dipartimento di Informatica e Automazione,
Università di Roma Tre,
Rome, Italy.
patrigna@dia.uniroma3.it

Work partially supported by European Commission – Fet Open project COSIN – CO-evolution and Self-organisation In dynamical Networks – IST-2001-33555, by European Commission – Fet Open project DELIS – Dynamically Evolving Large Scale Information Systems – Contract no 001907, by MIUR under Project ALGO-NEXT (Algorithms for the Next Generation Internet and Web: Methodologies, Design, and Experiments), and by "The Multichannel Adaptive Information Systems (MAIS) Project", MIUR Fondo per gli Investimenti della Ricerca di Base.

ABSTRACT

In this paper we consider the problem of finding three-dimensional orthogonal drawings of maximum degree six graphs from the computational complexity perspective. We introduce the 3SAT reduction framework which can be used to prove the NP-hardness of finding three-dimensional orthogonal drawings with specific constraints. By using the framework we show that, given a three-dimensional orthogonal shape of a graph (a description of the sequence of axis parallel segments of each edge) finding the coordinates for nodes and bends such that the drawing has no intersection is NP-hard. Conversely, we show that if node coordinates are fixed, finding a shape for the edges that is compatible with a non-intersecting drawing is a feasible problem, which becomes NP-hard if a maximum of two bends per edge is allowed. We comment the impact of these results on the two open problems of determining whether a graph always admits a drawing with at most two bends per edge and of characterizing orthogonal shapes admitting an orthogonal drawing without intersections.

1 Introduction

Because of its impact on applications and because of its theoretical appeal three-dimensional orthogonal graph drawing attracted a constant research interest through the last decade [1, 4, 9, 10, 11, 12, 14, 15, 20, 21, 23]. Nevertheless, some basic questions still lack an answer. It is open, for example, whether a graph of maximum degree six always admits a drawing with at most two bends per edge ([11, 12, 23], and [5], problem #46). A positive answer to this question could provide a drawing algorithm of unprecedented effectiveness for three-dimensional information visualization. In fact, two bends would be the best possible, since any drawing of K_5 uses at least two bends on at least one edge [22]. In [11] is conjectured that the answer to this question is false, but the graph that was thought to require three bends, the K_7 graph, was drawn with two bends per edge by Wood [20], along with the other 6-regular complete multi-partite graphs $K_{6,6}$, $K_{3,3,3}$, and $K_{2,2,2,2}$ [22].

Also, a characterization of the orthogonal shapes admitting an orthogonal drawing without intersections (called *simple orthogonal shapes*) is still missing in the general case ([13] and [3], problem 20). Such a characterization would allow to separate the task of defining the shape of the drawing from the task of computing its coordinates, extending to three-dimensions a well studied and widely adopted two-dimensional approach [17, 18, 6]. In 2D, the topology-shape-metrics approach consists of three main steps. In the first step a planar embedding of the input graph G is defined. In the second step a two-dimensional orthogonal representation of G is computed. An orthogonal representation is an equivalence class of orthogonal drawings of G all having the same shape and such that no two edges intersect. It can be described by labeling each edge (u, v) of G with a sequence of labels in the set {x+, x–, y+, y–}. In the third step, the coordinates for the nodes and for the bends along the edges are found.

A key component of the 2D topology-shape-metrics approach is a characterization of those properties that must be satisfied by the labeling in order to guarantee the existence of an orthogonal drawing without intersections. Such a characterization can be found in the works by Vijaian and Widgerson and by Tamassia [17, 19], and a 3D counterpart of it consists of the solution to the following problem (also called *Simplicity Testing Problem*): Let G be a graph whose edges are directed and labeled with a sequence of labels in the set {x+, x–, y+, y–, z+, z–}. Does a 3D orthogonal drawing of G exist such that each edge has a shape "consistent" with its labeling and no two edges intersect? For example, Figure 1 shows two graphs, G_1 and G_2, along with a labeling for their edges. For G_1 there exists a 3D orthogonal drawing without intersections and such that every edge has a shape consistent with the sequence of labels associated with it. For G_2, such a drawing does not exist.

Only very preliminary results toward the recognition of simple orthogonal shapes are provided in [7, 8] where paths (with further additional constraints) and cycles are considered, respectively. In [13] it is shown that the known characterization for cycles does not immediately extend to even seemingly simple graphs such as theta graphs (simple graphs consisting of three cycles).

In this paper we consider three-dimensional orthogonal drawing of a maximum degree six graph from the computational complexity perspective. The main contributions of this paper are the following:

- We introduce a framework which can be used for reducing an NP-hard problem

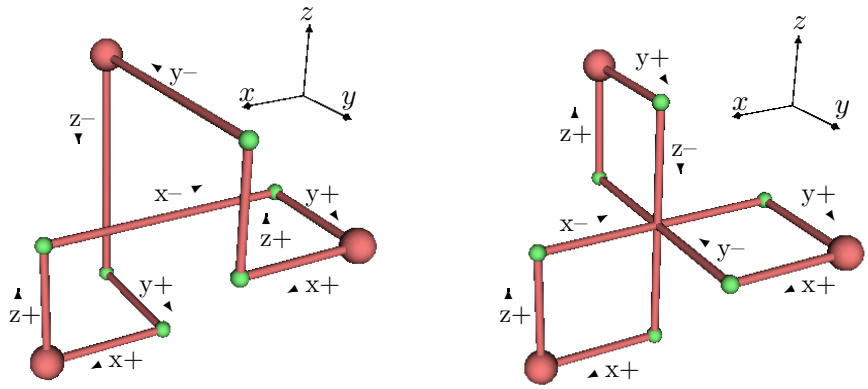


Figure 1: Graph G_1 (a) admits a non-intersecting 3D orthogonal drawing such that the edges are "consistent" with the directions and labels associated with them. Graph G_2 (b) does not admit such a drawing.

(namely, the 3SAT problem) to a three-dimensional geometric problem. The 3SAT reduction framework can be used to show that it is NP-hard to decide if an orthogonal 3D drawing of a graph satisfying some constraints exists.

- By using such a framework we show that the *simplicity testing* problem is NP-hard.
- Conversely, we show that the opposite problem of finding a shape for the edges that is compatible with a non-intersecting drawing where the position of the nodes is fixed is a feasible problem. In fact we produce an algorithm for computing such drawings which works in $O(|V| \log |V|)$ time, where $|V|$ is the number of nodes of the input graph.
- By using the 3SAT reduction framework, we show that if a maximum of two bends per edge is allowed the latter problem becomes NP-hard.
- We comment the impact of these results on the two open problems of determining whether a graph always admits a drawing with at most two bends per edge and of characterizing orthogonal shapes admitting an orthogonal drawing without intersections.

The paper is organized as follows. In Section 2 some preliminary definitions are given. Section 3 introduces the 3SAT reduction framework. Section 4 is devoted to the problem of finding a drawing with a given shape, while Section 5 is devoted to the reverse problem of finding a drawing with nodes at fixed points. Finally, Section 6 contains a discussion of the results and some open problems.

2 Background

We assume familiarity with basic graph drawing, graph theory, and computational geometry terminology (see, e.g. [6, 2, 16]).

A *3D orthogonal drawing* of a graph is such that nodes are mapped to distinct points of the three dimensional space and edges are chains of axis-parallel segments. A *bend* is a point shared between two subsequent segments of the same edge. An *intersection* in a three dimensional orthogonal drawing is a pair of edges that overlap in at least one point that does not correspond to a common end-node. A *k-bend (3D orthogonal) drawing* of a graph, where k is a non-negative integer, is a non-intersecting orthogonal drawing such that each edge has at most k bends.

For the sake of brevity in the remainder of this paper we call *drawing* a 3D orthogonal drawing. An x-plane (y-plane, z-plane, respectively) is a plane perpendicular to the x axis (y axis, z axis, respectively). Given a drawing Γ of a graph G and two nodes u and v, we write $u >_x v$ if the x coordinate of u is greater than the x coordinate of v in Γ. Also, we write $u >_x >_y v$ if $u >_x v$ and $u >_y v$.

A *direction label* is a label in the set {x+, x–, y+, y–, z+, z–}. Let G be a graph and Γ be a 3D orthogonal drawing of G. Let e be an undirected edge of G whose end-nodes are u and v. Select one of the two possible orientations (u, v) and (v, u) of e and call $p_1, p_2, \ldots, p_m$ be the end points of the orthogonal segments corresponding to edge e in Γ as they are encountered while moving along e from u to v. The *shape of e in Γ* is the sequence of the direction labels corresponding to the directions of vectors $\overrightarrow{p_i, p_{i+1}}$, $i = 1, \ldots, m-1$. For example, consider an edge (u, v) drawn with a single bend and such that $u <_x <_y v$. The shape of e consists of the orientation from u to v and the sequence of labels x+, y+. We also write $u \xrightarrow{\text{x+}} \xrightarrow{\text{y+}} v$.

When producing a 3D orthogonal drawing of a graph one can ask if the positions of the vertices and the shapes of the edges can be computed separately. For example, it can be asked if it is always possible to find a drawing of a graph whose vertex positions are fixed.

Problem: **Routing**
Instance: A graph $G(V, E)$ and a mapping between nodes and distinct points of the three-dimensional space.
Question: Does a non-intersecting 3D orthogonal drawing of G exist such that the nodes have the specified coordinates?

Conversely, it can be asked what is the complexity of deciding if a graph admits a drawing such that its edges have a specified shape. We call *shape graph* a graph where a shape (an orientation and a sequence of direction labels) is specified for each one of its edges. A shape graph γ is *simple* if it admits a non-intersecting drawing Γ such that each edge has the specified shape. Formally, the Simplicity Testing problem is as follows.

Problem: **Simplicity Testing**
Instance: A shape graph γ, that is, a graph $G(V, E)$ and a shape for each edge $e \in E$, consisting of an orientation of e and a sequence of labels in the set {x+, x–, y+, y–, z+, z–}.
Question: Does a non-intersecting drawing of G exist such that each edge has the specified shape?

Since the existence of a 2-bend drawing for every graph of maximum degree six is an open problem, it could be interesting to investigate the complexity of the Routing problem when restricted to 2-bend drawings.

Problem: **2-Bend Routing**
Instance: A graph $G(V, E)$ and a mapping between nodes and distinct points of the three-dimensional space.
Question: Does a non-intersecting 2-bend drawing of G exist such that the nodes have the specified coordinates?

3 The 3SAT Reduction Framework

The 3SAT reduction framework introduced in this section can be used to show that it is NP-hard finding a 3D drawing of a graph within the orthogonal standard that satisfies some constraints. By using this framework it is shown, in Sections 4 and 5, respectively, the NP-hardness of Simplicity Testing and of 2-Bend Routing. Throughout this section, the target problem is assumed to be as follows:

Problem: ***Target problem***
Instance: A graph $G(V, E)$ and a set S of constraints expressed with respect to its nodes and edges.
Question: Does a non-intersecting 3D drawing of G exist such that the constraints in S are satisfied?

The 3SAT problem is as follows:

Problem: **3-Satisfiability (3SAT)**
Instance: A set of clauses $\{c_1, c_2, \ldots, c_m\}$, each containing three literals from a set of boolean variables $\{v_1, v_2, \ldots, v_n\}$.
Question: Can truth values be assigned to the variables so that each clause contains at least one true literal?

Given a 3SAT instance ϕ, the 3SAT reduction framework specifies how to build an instance $I_\phi = (G_\phi(V_\phi, E_\phi), S_\phi)$ of the target problem such that ϕ admits a solution if and only if I_ϕ does. $G_\phi(V_\phi, E_\phi)$ is composed by different types of gadgets connected together. The bounding boxes of the gadgets are depicted in Fig. 2, while the interior components are not shown and depend on the specific target problem. The three basic gadgets are the following.

Variable gadget. Instance I_ϕ has a variable gadget V_i for each boolean variable v_i of ϕ. Fig. 2 shows the variable gadgets as tall vertical blocks placed in a row along the y axis in such a way that, if $i < j$, variable gadget V_i has lower y coordinates than variable gadget V_j.

Clause gadget. Instance I_ϕ has one clause gadget C_i for each clause $c_i = l_h \vee l_j \vee l_k$ of ϕ. Clause gadgets are represented in Fig. 2 as small cubes. Denoted with v_h, v_j, and v_k the variables of literals l_h, l_j, and l_k, respectively, and assumed that $h < j < k$, clause gadget C_i is placed directly in front of the variable gadget V_j.

Joint gadget. For each clause $c_i = l_h \vee l_j \vee l_k$ of ϕ, I_ϕ has two joint gadgets $J_{i,h}$ and $J_{i,k}$, depicted in Fig. 2 as flat blocks placed in front of the variable gadgets V_j and V_k, respectively.

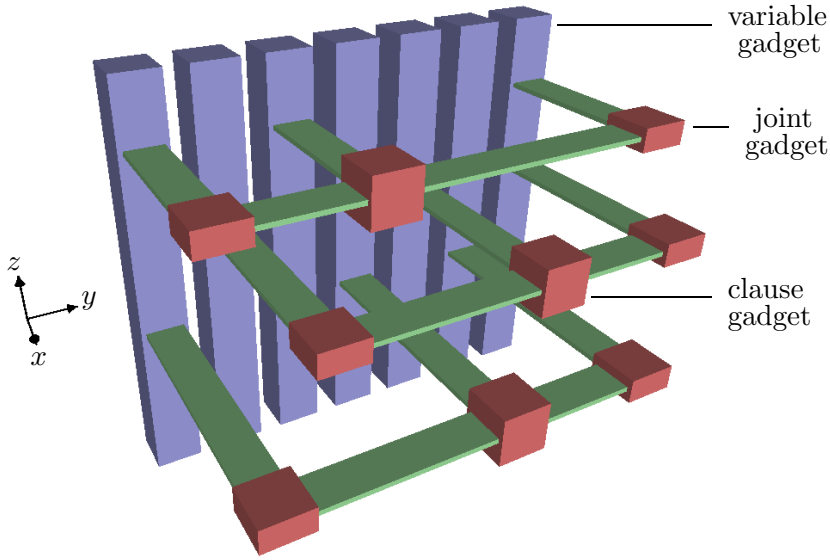


Figure 2: A representation of the basic blocks of an instance of the target problem built as specified by the 3SAT reduction framework.

In order to use the 3SAT reduction framework for the NP-hardness proof of a specific target problem a complete specification must be provided, where a *specification* for the 3SAT reduction framework is defined as follows.

- Construction rules describing how, starting from an instance ϕ of the 3SAT problem, variable gadgets, joint gadgets, and clause gadgets are built and connected together and an instance $I_\phi = (G_\phi(V_\phi, E_\phi), S_\phi)$ of the target problem is obtained.
- For each variable gadget V_i a bipartition of the non-intersecting drawings of $G_\phi(V_\phi, E_\phi)$ satisfying constraints S_ϕ into two sets, denoted T_{V_i} and F_{V_i}.
- For each joint gadget $J_{i,k}$ a bipartition of the non-intersecting drawings of $G_\phi(V_\phi, E_\phi)$ satisfying constraints S_ϕ into two sets, denoted $T_{J_{i,k}}$ and $F_{J_{i,k}}$.

A specification is said to be *compliant* if, for any 3SAT instance ϕ, the following three statements hold.

Statement 1 *Given an instance ϕ of* 3SAT *problem, the corresponding instance $I_\phi = (G_\phi(V_\phi, E_\phi), S_\phi)$ of the target problem can be constructed in polynomial time.*

Statement 2 *If a non-intersecting drawing of $G_\phi(V_\phi, E_\phi)$ satisfying S_ϕ exists, it belongs to $T_{J_{i,h}}$ ($T_{J_{i,k}}$) if and only if it belongs to T_{V_h} (T_{V_k}).*

Statement 3 *For each clause $c_i = l_h \vee l_j \vee l_k$, where l_h (l_j, l_k, respectively) is the positive or the negative literal of variable v_h (v_j, v_k, respectively), and for each non-intersecting drawing Γ of $G_\phi(V_\phi, E_\phi)$ satisfying S_ϕ at least one between the following conditions holds:*

1. *$\Gamma \in T_{J_{i,h}}$ ($\Gamma \in F_{J_{i,h}}$) and l_h is the positive (negative) literal of v_h.*
2. *$\Gamma \in T_{V_j}$ ($\Gamma \in F_{V_j}$) and l_j is the positive (negative) literal of v_j.*

3. $\Gamma \in T_{J_{i,k}}$ ($\Gamma \in F_{J_{i,k}}$) *and* l_k *is the positive (negative) literal of* v_k.

Statement 4 *Consider a truth assignment to the variables* $v_i, \ldots v_n$ *satisfying* ϕ. *The set* $\cap_{i=0}^{n} A_i$, *where* $A_i = T_{V_i}$ *if* v_i *is true and* $A_i = F_{V_i}$ *if* v_i *is false, is not empty.*

Theorem 1 *Given a target problem, whose instance is a graph* $G(V, E)$ *and a set* S *of constraints expressed with respect to its nodes and edges, if it admits a compliant specification for the* 3SAT *reduction framework, then finding a non-intersecting 3D orthogonal drawing of* G *satisfying the constraints in* S *is NP-hard.*

Proof: First we show that a 3SAT instance ϕ admits a solution if and only if $G_\phi(V_\phi, E_\phi)$ admits a non-intersecting drawing satisfying S_ϕ. Consider a non-intersecting drawing Γ of $G_\phi(V_\phi, E_\phi)$ satisfying S_ϕ. It is easy to find an assignment of truth values to the boolean variables that satisfies ϕ, by taking $v_i = true$ if $\Gamma \in T_{V_i}$ and $v_i = false$ if $\Gamma \in F_{V_i}$. In fact, because of Statements 2 and 3 we have that each clause $c_i = l_h \vee l_j \vee l_k$ has at least one true literal and thus ϕ is satisfied. Conversely, consider an assignment of truth values to the boolean variables that satisfies ϕ. Statement 4 guarantees the existence of a drawing of $G_\phi(V_\phi, E_\phi)$ satisfying S_ϕ. The proof is completed by showing that instance $I_\phi = (G_\phi(V_\phi, E_\phi), S_\phi)$ can be obtained in polynomial time, which is guaranteed by Statement 1. □

4 Fixing the Shape and Searching for Coordinates

In this section we consider the Simplicity Testing problem, that is the problem of finding a non-intersecting drawing for a graph whose orthogonal shape is fixed. We first prove the following lemma.

Lemma 1 Simplicity Testing *is in NP.*

Proof: Given an instance $I = (G(V, E), S)$ of the Simplicity Testing problem, the search for a 3D orthogonal drawing of $G(V, E)$ satisfying S can be restricted to those drawings where each axis-orthogonal plane intersecting the drawing hosts at least a node or a bend. Since the number of nodes and bends in any 3D drawing satisfying S can be easily computed, an upper bound for the sides of the bounding box of such drawings can be obtained. This gives an upper bound for the maximum length λ of any axis-aligned segment of the edges. A non-deterministic Turing machine can be devised that assigns all possible lengths between one and λ to the edge segments and then checks in polynomial time if a non-intersecting drawing is produced. □

In the remaining part of this section we show that Simplicity Testing is NP-hard. Therefore, the following theorem holds.

Theorem 2 Simplicity Testing *is NP-complete.*

4.1 Simplicity Testing is NP-hard

We use the framework introduced in Section 3 in order to reduce an instance of the 3SAT problem to an instance of the Simplicity Testing problem. In the following sections it is shown how the variable gadgets (Section 4.1.1), joint gadgets (Section 4.1.2), and clause gadgets (Section 4.1.3) are built. Section 4.1.4 contains the hardness proof.

4.1.1 The Variable Gadget

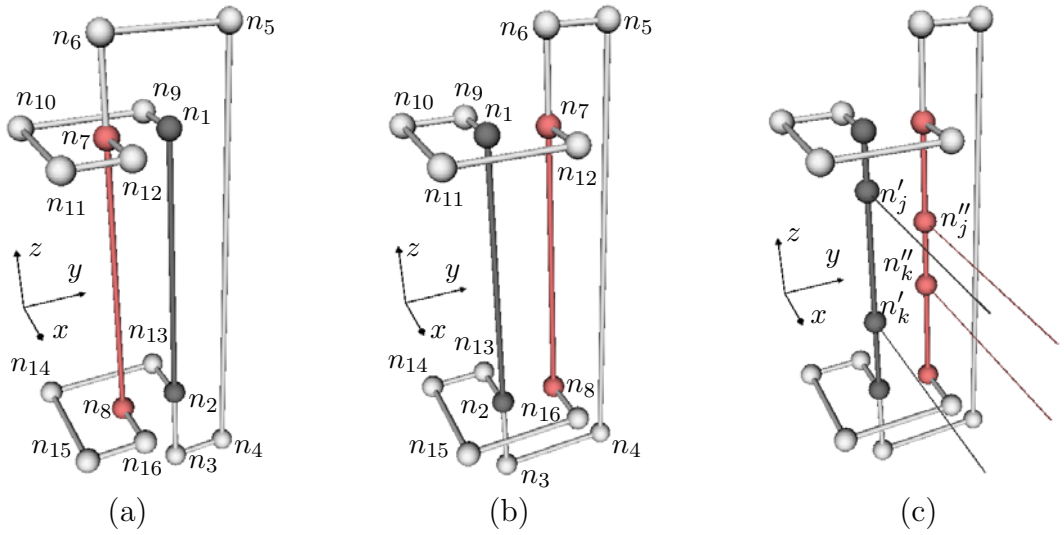


Figure 3: A drawing of the variable gadget V_i belonging to T_{V_i} (a) and a drawing belonging to F_{V_i} (b). Nodes n'_j, n''_j, n'_k, and n''_k are inserted in order to transmit the geometric constraints to the clause gadgets of clauses C_j and C_k, respectively (c).

The heart of the variable gadget V_i, depicted in Fig. 3, is the path $n_1 \xrightarrow{\text{z-}} n_2 \xrightarrow{\text{z-}} n_3 \xrightarrow{\text{y+}} n_4 \xrightarrow{\text{z+}} n_5 \xrightarrow{\text{y-}} n_6 \xrightarrow{\text{z-}} n_7 \xrightarrow{\text{z-}} n_8$, whose nodes lie on the same x-plane. Further, the path $n_1 \xrightarrow{\text{x-}} n_9 \xrightarrow{\text{y-}} n_{10} \xrightarrow{\text{x+}} n_{11} \xrightarrow{\text{y+}} n_{12} \xrightarrow{\text{x-}} n_7$ constraints nodes n_1 and n_7 to share the same z-plane. Analogously, path $n_2 \xrightarrow{\text{x-}} n_{13} \xrightarrow{\text{y-}} n_{14} \xrightarrow{\text{x+}} n_{15} \xrightarrow{\text{y+}} n_{16} \xrightarrow{\text{x-}} n_8$ constraints nodes n_2 and n_8 to share the same z-plane. We define T_{V_i} as the set of non-intersecting drawings of $G_\phi(V_\phi, E_\phi)$ satisfying the directions constraints and such that $n_1 >_y n_7$ (as in Fig. 3.a). Analogously, we define F_{V_i} as the set of non-intersecting drawings of $G_\phi(V_\phi, E_\phi)$ satisfying the direction constraints and such that $n_1 <_y n_7$ (as in Fig. 3.b). It is easy to show the following lemma that guarantees that T_{V_i} and F_{V_i} form a bipartition of the non-intersecting drawings of $G_\phi(V_\phi, E_\phi)$ satisfying the direction constraints.

Lemma 2 *For each $i = 1, \ldots n$, any non-intersecting drawing of $G_\phi(V_\phi, E_\phi)$ satisfying the direction constraints either belongs to T_{V_i} or to F_{V_i}.*

For each clause c_j of the 3SAT formula in which the variable participates we insert a node n'_j between nodes n_1 and n_2 and a node n''_j between nodes n_7 and n_8. In any drawing Γ of $G_\phi(V_\phi, E_\phi)$ satisfying the direction constraints, nodes n'_j and n''_j have the same relative position with respect to the y axis as n_1 and n_7, i.e., $n'_j >_y n''_j$ if $\Gamma \in T_{V_i}$ and $n'_j <_y n''_j$ if $\Gamma \in F_{V_i}$. Suitable edges attached to the nodes n'_j and n''_j along the protruding lines shown in Fig. 3.c transmit the above constraints from V_i to the clause gadget C_j (possibly via joint gadget $J_{i,j}$). Note that nodes n'_j and n''_j do not need to lie on the same z-plane.

4.1.2 The Joint Gadget

Given a clause $c_i = l_h \vee l_j \vee l_k$, the joint gadget $J_{i,k}$ is the reflected image with respect to the y axis of the joint gadget $J_{i,h}$. Thus, in the following we will only describe the joint gadget $J_{i,h}$, which is depicted in Fig. 4 and composed by two cycles

$\alpha = n_1 \xrightarrow{\text{y+}} n_2 \xrightarrow{\text{x–}} n_3 \xrightarrow{\text{y–}} n_4 \xrightarrow{\text{x+}} n_5 \xrightarrow{\text{y–}} n_6 \xrightarrow{\text{x+}} n_7 \xrightarrow{\text{y+}} n_8 \xrightarrow{\text{x–}} n_1$, and $\alpha' = n'_1 \xrightarrow{\text{y–}} n'_2 \xrightarrow{\text{x+}} n'_3 \xrightarrow{\text{y+}} n'_4 \xrightarrow{\text{x–}} n'_5 \xrightarrow{\text{y+}} n'_6 \xrightarrow{\text{x–}} n'_7 \xrightarrow{\text{y–}} n'_8 \xrightarrow{\text{x+}} n'_1$. Nodes n_1 and n'_1 are connected by a path $n_1 \xrightarrow{\text{z–}} n''_1 \xrightarrow{\text{z–}} n'_1$ while nodes n_5 and n'_5 are connected by the path $n_5 \xrightarrow{\text{z–}} n''_5 \xrightarrow{\text{z–}} n'_5$.

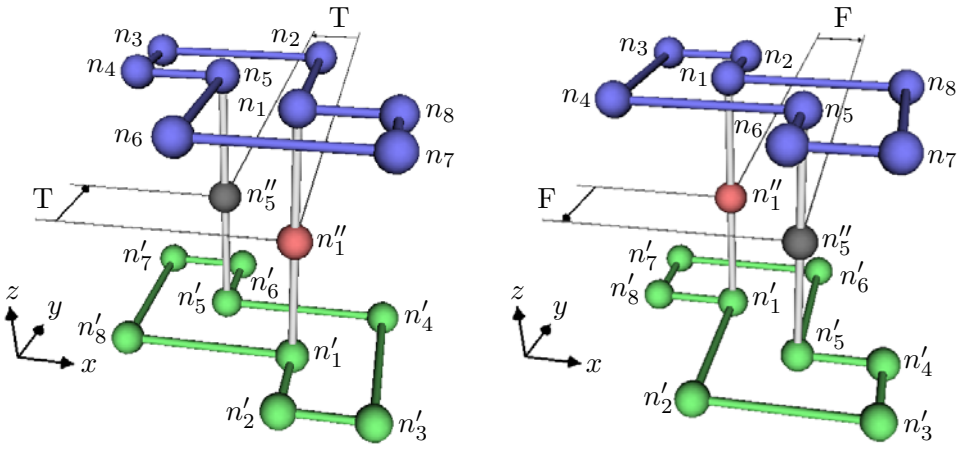


Figure 4: A drawing of the joint gadget $J_{i,h}$ belonging to $T_{J_{i,h}}$ (a) and a drawing belonging to $F_{J_{i,h}}$ (b).

We define $T_{J_{i,h}}$ as the set of non-intersecting drawings of $G_\phi(V_\phi, E_\phi)$ satisfying the directions constraints and such that $n''_5 <_x n''_1$ (as in Fig. 4.a). Analogously, we define F_{V_i} as the set of non-intersecting drawings of $G_\phi(V_\phi, E_\phi)$ satisfying the direction constraints and such that $n''_5 >_x n''_1$ (as in Fig. 4.b). It is easy to show the following lemma that guarantees that $T_{J_{i,h}}$ and $F_{J_{i,h}}$ form a bipartition of the non-intersecting drawings of $G_\phi(V_\phi, E_\phi)$ satisfying the direction constraints.

Lemma 3 *For each $i = 1, \ldots m$, any non-intersecting drawing of $G_\phi(V_\phi, E_\phi)$ satisfying the direction constraints either belongs to $T_{J_{i,h}}$ or to $F_{J_{i,h}}$.*

The following lemma shows how a geometric constraint on the relative position of nodes n''_5 and n''_1 with respect to the x axis has an effect on their relative position with respect to the y axis in any non-intersecting drawing of the joint gadget.

Lemma 4 *In any non-intersecting drawing of $G_\phi(V_\phi, E_\phi)$ satisfying the directions constraints either $n''_5 >_y n''_1$ and $n''_5 <_x n''_1$ or $n''_5 <_y n''_1$ and $n''_5 >_x n''_1$.*

Proof: A drawing without intersections of the joint gadget where $n''_5 >_y n''_1$ and $n''_5 <_x n''_1$ is shown in Fig. 5.a. Analogously, a drawing without intersections of the joint gadget where $n''_5 <_y n''_1$ and $n''_5 >_x n''_1$ is shown in Fig. 5.d. From Fig. 5.b it is easy to see that that if $n''_5 >_y n''_1$ and $n''_5 >_x n''_1$ then α necessarily intersects. Analogously, from Fig. 5.c it is easy to see that if $n''_5 <_y n''_1$ and $n''_5 <_x n''_1$ then α' necessarily intersects. □

4.1.3 The Clause Gadget

The clause gadget is depicted in Fig. 6. Its main component is the path $\alpha = n_1 \xrightarrow{\text{y+}} n_2 \xrightarrow{\text{x–}} n_3 \xrightarrow{\text{y–}} n_4 \xrightarrow{\text{x+}} n_5 \xrightarrow{\text{y+}} n_6 \xrightarrow{\text{x+}} n_7$, whose nodes lie on the same z-plane. Attached to α are the paths $n'_1 \xrightarrow{\text{z–}} n_1 \xrightarrow{\text{z–}} n''_1$, $n'_2 \xrightarrow{\text{z–}} n_2 \xrightarrow{\text{z–}} n''_2$, $n'_6 \xrightarrow{\text{z–}} n_6 \xrightarrow{\text{z–}} n''_6$, and $n'''_7 \xrightarrow{\text{x+}} n'_7 \xrightarrow{\text{z–}} n_7 \xrightarrow{\text{z–}} n''_7 \xrightarrow{\text{x+}} n_8 \xrightarrow{\text{y+}} n_9 \xrightarrow{\text{x–}} n''_2$.

The following lemma holds:

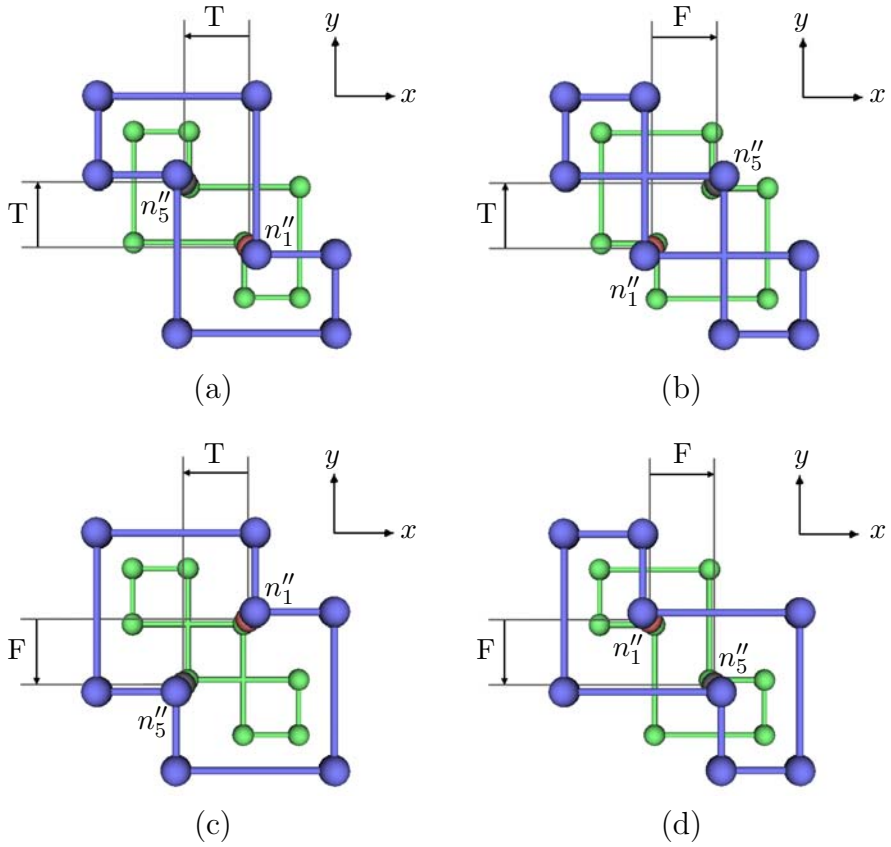


Figure 5: The Joint gadget admits a drawing without intersection if and only if $n_5'' >_y n_1''$ and $n_5'' <_x n_1''$ (a) or $n_5'' <_y n_1''$ and $n_5'' >_x n_1''$ (d).

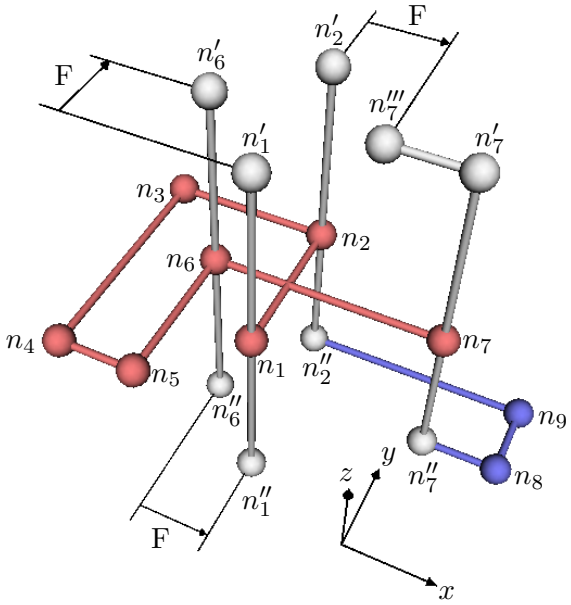


Figure 6: Clause gadget

Lemma 5 *Clause gadget C_i admits a drawing without intersections if and only if $n_1 <_x n_6$ or $n_1 >_y n_6$ or $n_7''' <_x n_2'$.*

Proof: Fig. 7 shows that α admits a drawing without intersections if at least one of the three following conditions holds: $n_1 <_x n_6$ (as shown in Figs. 7.a, 7.b, 7.c, and 7.d), $n_1 >_y n_6$ (as shown in Figs. 7.a, 7.b, 7.e, and 7.f), or $n_7''' <_x n_2'$ (as shown in Figs. 7.a, 7.c, 7.e, and 7.g).

Conversely, suppose that $n_6 \leq_x n_1$, $n_1 \leq_y n_6$, and $n_7''' \geq_x n_2'$ (as depicted in Figs 6 and 7.h). We have that α necessarily intersects. In fact, condition $n_7''' \geq_x n_2'$ implies $n_7' >_x n_2'$, which in turn implies $n_7 >_x n_2$, or, equivalently, $n_1 <_x n_7$. Also, because of path $n_7'' \xrightarrow{\text{x+}} n_8 \xrightarrow{\text{y+}} n_9 \xrightarrow{\text{x-}} n_2''$, we have $n_7'' <_y n_2''$, which implies $n_6 <_y n_2$. An intersection between edges $n_1 \xrightarrow{\text{y+}} n_2$ and $n_6 \xrightarrow{\text{x+}} n_7$ follows from $n_6 \leq_x n_1 <_x n_7$ and $n_1 \leq_y n_6 <_y n_2$ and from the fact that the nodes of α lie on the same z-plane. □

4.1.4 The Hardness Proof

We now describe how the various gadgets are connected together, and we show that the whole construction is a compliant specification for the 3SAT reduction framework.

Joint gadget $J_{i,h}$ is connected to both variable gadget V_h and clause gadget C_i. In particular, n_1'' of $J_{i,h}$ is connected to n_i' of V_h with the edge $n_i' \xrightarrow{\text{x+}} n_1''$ and n_5'' of $J_{i,h}$ is connected to n_i'' of V_h with the edge $n_i'' \xrightarrow{\text{x+}} n_5''$. Due to the above described connections the following lemma holds:

Lemma 6 *Statement 2 holds, that is, if a non-intersecting drawing of $G_\phi(V_\phi, E_\phi)$ exists such that the edges have the prescribed shape, it belongs to $T_{J_{i,h}}$ if and only if it belongs to T_{V_h}.*

Proof: The statement follows by considering that n_1'' and n_5'' of $J_{i,h}$ share their y coordinates with n_i' and n_i'' of V_h, respectively, and by applying Lemma 4. □

Each clause $c_i = l_h \vee l_j \vee l_k$ is connected to joint gadget $J_{i,h}$, variable gadget V_j, and joint gadget $J_{i,k}$. If l_h is the positive (negative) literal of variable v_h, we attach nodes n_1'' and n_5'' of the joint gadget $J_{i,h}$ to nodes n_6'' and n_1'' (n_1'' and n_6''), respectively. If l_j is the positive (negative) literal of variable v_j, we attach nodes n_i' and n_i'' of the variable gadget V_j to n_6' and n_1' (n_1' and n_6'), respectively. If l_k is the positive (negative) literal of variable v_k, we attach nodes n_1 and n_4 of the joint gadget $J_{i,k}$ to nodes n_2' and n_7''' (n_7''' and n_2'), respectively.

Lemma 7 *Statement 3 holds, that is, for each clause $c_i = l_h \vee l_j \vee l_k$ and for each non-intersecting drawing Γ of $G_\phi(V_\phi, E_\phi)$ such that the edges have the prescribed shape, at least one between the following conditions holds: (i) $\Gamma \in T_{J_{i,h}}$ ($\Gamma \in F_{J_{i,h}}$) and l_h is the positive (negative) literal of v_h. (ii) $\Gamma \in T_{V_j}$ ($\Gamma \in F_{V_j}$) and l_j is the positive (negative) literal of v_j. (iii) $\Gamma \in T_{J_{i,k}}$ ($\Gamma \in F_{J_{i,k}}$) and l_k is the positive (negative) literal of v_k.*

Proof: Due to Lemma 5 each clause gadget C_i admits a drawing without intersections if and only if: (i) $n_1 <_x n_6$, or (ii) $n_1 >_y n_6$, or (iii) $n_7''' <_x n_2'$. Considering the connection rules described above the three conditions can be rewritten as: (i) l_h is the positive (negative) literal of v_h and $n_1'' <_x n_5''$ ($n_5'' <_x n_1''$), or (ii) l_j is the positive (negative) literal of v_j and $n_i'' >_y n_i'$ ($n_i' >_y n_i''$), or (iii) l_k is the positive (negative) literal of v_k and $n_1' <_x n_5''$ ($n_5' <_x n_1''$). Recalling the definition of true variable gadget and true joint gadget, the statement follows. □

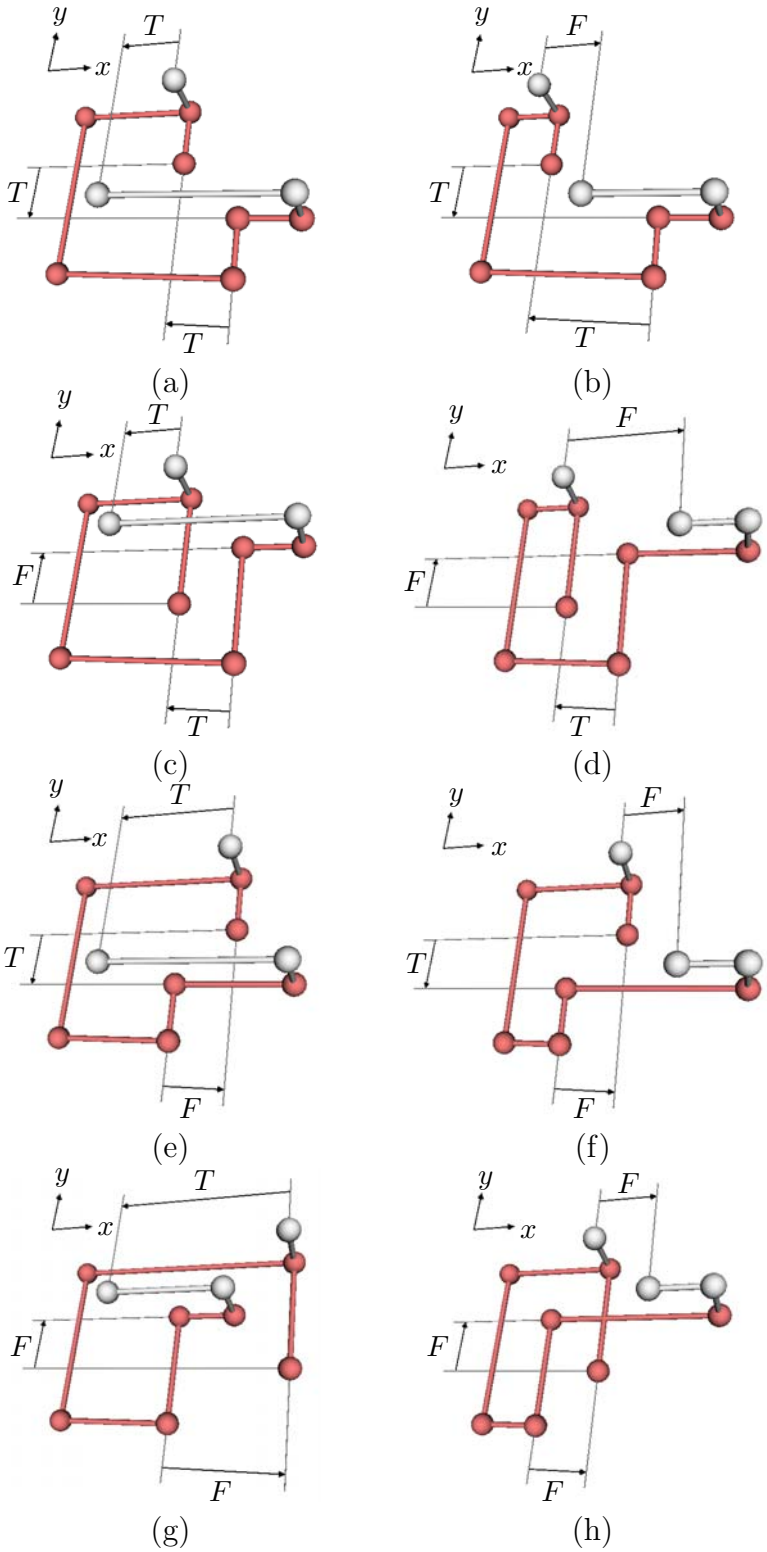


Figure 7: The cases used in the proof of Lemma 5.

Lemma 8 *Statement 4 holds, that is, if ϕ admits a solution, then $G_\phi(V_\phi, E_\phi)$ admits a non-intersecting drawing whose edges have the prescribed shape.*

Proof: Consider a truth assignment satisfying ϕ. If variable v_i is true (false) we can use the true (false) drawing of variable gadget V_i depicted in Fig. 3.a (Fig. 3.b). Also, given

a clause $c_i = l_h \vee l_j \vee l_k$, if variable v_h is true (false) we can use the true (false) drawing of variable gadget $J_{i,h}$ depicted in Fig. 4.a (Fig. 4.b). In order to draw the clause gadget C_i without intersection a suitable drawing can be selected between the ones depicted in Fig. 7.a-g. □

Now we can prove the following lemma.

Lemma 9 Simplicity Testing *is NP-hard*

Proof: The proof is based on the fact that a compliant specification can be found for the 3SAT reduction framework introduced in Section 3. Lemmas 6, 7, and 8 prove that Statements 2, 3, and 4 hold, respectively. Also, given a 3SAT instance ϕ, the corresponding Simplicity Testing instance I_ϕ can be built in polynomial time (Statement 1 holds). Therefore, the construction rules described in Sections 4.1.1, 4.1.2, and 4.1.3 correspond to a compliant specification for the 3SAT reduction framework, and Theorem 1 applies. □

5 Fixing the Coordinates and Searching for a Shape

In this section we tackle the reverse problem with respect to the one addressed in Section 4, that is, the problem of finding a routing for the edges when the position of the nodes is fixed. We first show in Section 5.1 that Routing is feasible and then in Section 5.2 that the same problem where only two bends per edge are allowed (2-Bend Routing) is NP-hard. It is easy to show the following lemma.

Lemma 10 2-Bend Routing *is in NP.*

Proof: First note that a graph admits a 3D orthogonal drawing with nodes at prescribed positions if and only if it admits an orthogonal drawing such that each pair of nodes have the same relative position with respect to the x, y, and z axes as the one prescribed. It follows that in order to search for a solution of 2-Bend Routing we may restrict to consider orthogonal grid drawings, searching for one that has the nodes in the same relative position. Analogously to the proof of Lemma 1 an upper bound for the bounding box of such drawings can be easily found. A non-deterministic Turing machine could generate all grid drawings of the graph within the computed bounding box and then check in polynomial time whether each pair of nodes has the same relative position as the one specified by the 2-Bend Routing instance. □

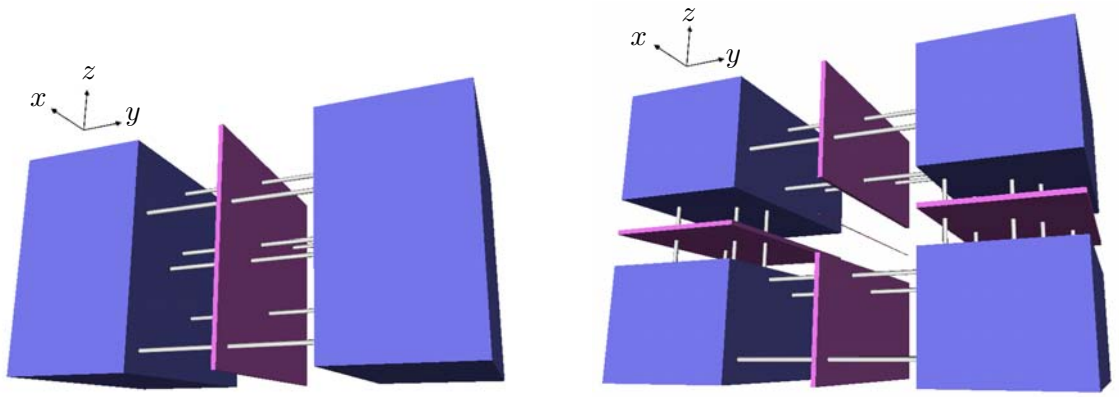
Due to Lemma 10 and due to the NP-hardness of 2-Bend Routing which is proved in Section 5.2 (Lemma 16 of Section 5.2.5) the following theorem holds.

Theorem 3 2-Bend Routing *is NP-complete.*

5.1 Routing is Feasible

In order to show that Routing is feasible, we produce an algorithm that computes a non-intersecting orthogonal drawing of a graph $G(V, E)$ with nodes at prescribed positions in $O(|V| \log |V|)$ time. In accordance with the considerations in the proof of Lemma 10, we

describe an algorithm that produces a grid drawing where the relative positions of the nodes are coherent with the prescribed positions, assuming that the drawing requested by ROUTING can be easily obtained from it. The drawing algorithm takes advantage of the relative coordinates scenario [15], where it is possible to insert a grid plane in the drawing in constant time. It starts by sorting the nodes in ascending order according to their x, y, and z coordinates. This is needed to build a drawing where only the nodes are present. Finally, edges are added one at a time, routing them without introducing intersections.



(a) (b)

Figure 8: Inserting an orthogonal plane in a drawing in the relative coordinates scenario (a). If two perpendicular planes are inserted (b) the grid line common to the two planes does not intersect any node or edge of the drawing.

In order to insert an edge in the drawing without introducing intersections, consider the operation of inserting a plane perpendicular to the y axis (see Fig. 8.a). After the insertion, no node or edge will lay on the newly inserted plane, although some edges may perpendicularly intersect the plane. Now consider a second insertion of a plane perpendicular to the z axis (see Fig. 8.b). After this second insertion, the x parallel grid line that is common to both the two inserted planes is not intersected by any edge or node of the drawing. The algorithm uses this strategy of plane insertion in order to create the space to route each edge from the source node to the target node.

Suppose that edges $e_1, e_2, \ldots, e_{i-1}$ have been inserted and that edge e_i needs to be added to the drawing, connecting node n_s to node n_t. In Fig. 9 it is shown how edge e_i could be drawn in the case in which $n_t >_x >_y <_z n_s$, node n_s has no edge leaving in direction $y+$ and node n_t has no edge leaving in direction $y-$. Observe that, at most four planes need to be inserted in the drawing in order to create the three non-intersected grid lines needed for the new edge to be routed.

Since all other cases can be analogously handled by inserting at most six bends for each edge, the following theorem holds.

Theorem 4 *There exists an algorithm that solves* ROUTING *in* $O(|V| \log |V|)$ *time. The algorithm introduces a maximum of* 6 *bends per edge.*

5.2 2-BEND ROUTING is NP-hard

In this section we take advantage of the 3SAT reduction framework in order to show that 2-BEND ROUTING is NP-hard. Namely, Sections 5.2.1, 5.2.2, 5.2.3, and 5.2.4 contain the

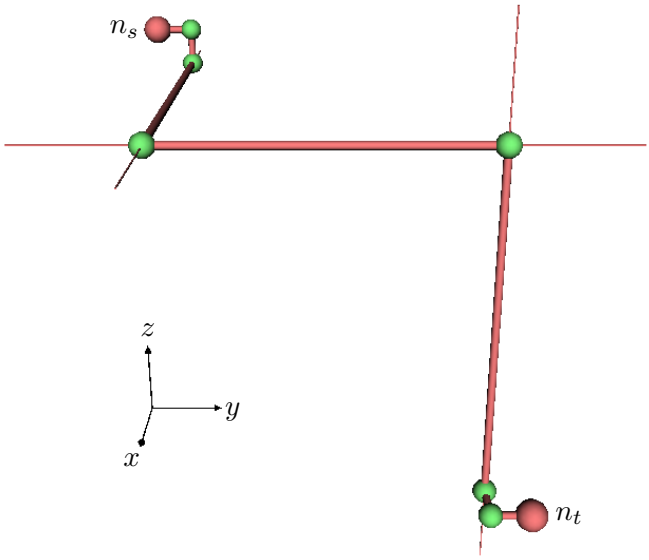


Figure 9: The insertion of an edge from n_s to n_t with six bends.

construction rules for the 2-Bend Routing instance I_ϕ corresponding to a given 3SAT instance ϕ, while Section 5.2.5 shows that the described rules correspond to a compliant specification for the 3SAT reduction framework and provides the NP-hardness proof.

5.2.1 The Basic Gadget

The basic gadget (see Fig. 10) is used as a building block of several parts of the 2-Bend Routing instance. Fig. 10.a shows its nodes and how they are connected, while Fig. 10.b shows nodes prescribed positions. The basic gadget is composed by ten nodes. Node n_1 is connected to the three nodes n_2, n_3 and n_4. Analogously, node n_5 is connected to the three nodes n_6, n_7 and n_8. Nodes n_1 and n_5 are connected both with the single edge (n_1, n_5) and with the path of three edges $(n_1, n_{1,5})$, $(n_{1,5}, n_{5,1})$ and $(n_{5,1}, n_5)$.

As for nodes prescribed positions, they are placed in such a way that $n_1 <_x <_y <_z n_2 =_x =_y <_z n_3 =_x =_y <_z n_4$, $n_1 =_x >_y >_z n_{1,5} =_x >_y >_z n_{5,1} =_x >_y >_z n_5$, and $n_5 <_x >_y >_z n_6 =_x =_y >_z n_7 =_x =_y >_z n_8$.

Fig. 10.b shows also some lines and points to help understanding the node prescribed positions and their mutual relationships. Points $p_{t,1}, p_{t,2}$ and $p_{t,3}$ are defined as follows. Point $p_{t,1}$ has the same coordinates of $n_{1,5}$ with the exception of the z coordinate which is shared with n_1. Point $p_{t,2}$ has the same coordinates of n_1 with the exception of the y coordinate which is shared with n_5. Point $p_{t,3}$ has the same coordinates of $n_{5,1}$ with the exception of the y coordinate which is shared with n_5. Analogously, nodes $p_{f,1}, p_{f,2}$ and $p_{f,3}$ can be defined by replacing n_1 with n_5 and $n_{1,5}$ with $n_{5,1}$.

Lemma 11 *In any non-intersecting 2-bend drawing of the basic gadget, edge (n_1, n_5) has exactly one bend placed either in $p_{t,2}$ or in $p_{f,2}$.*

Proof: Since n_1 and n_5 share the same x-plane, but do not share any axis-parallel line, edge (n_1, n_5) must lie on the x-plane in order to be drawn with a maximum of two bends. Also, due to the prescribed positions of nodes n_2, n_3, and n_4 with respect to node n_1, the

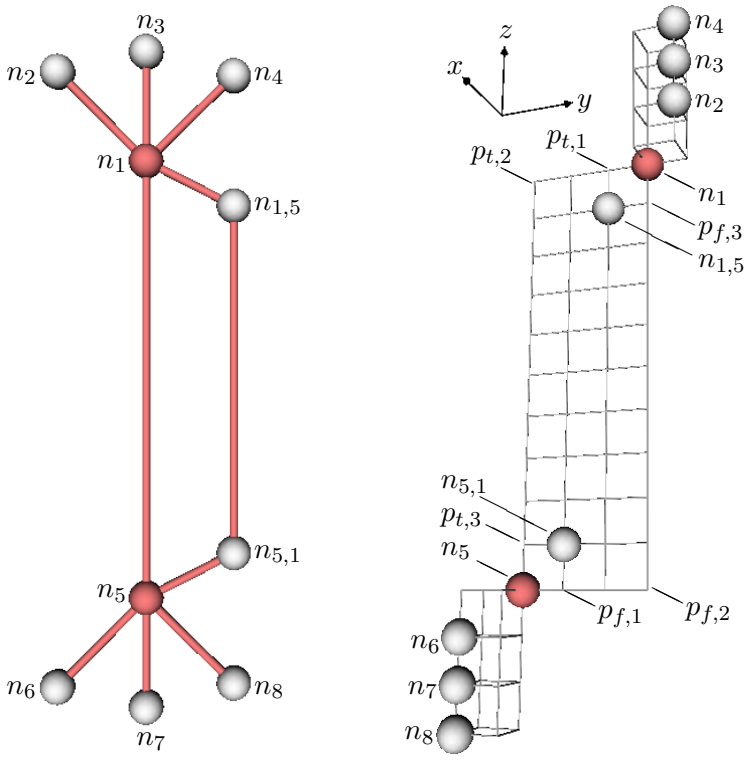


Figure 10: The basic gadget is composed by ten nodes, joined by ten edges as represented in (a). In (b) the prescribed node positions are represented.

three directions $x+$, $y+$, and $z+$ of n_1 are used by edges (n_1, n_2), (n_1, n_3), and (n_1, n_4) (not necessarily in this order, see Fig. 11.a and 11.b). Analogously, the three directions $x+$, $z-$, and $y-$ of n_5 are used by edges (n_5, n_6), (n_5, n_7), and (n_5, n_8). It follows that edge (n_1, n_5) must use the y– or z– direction of node n_1 and the y+ or z+ direction of node n_5. Due to the path of three edges $(n_1, n_{1,5})$, $(n_{1,5}, n_{5,1})$ and $(n_{5,1}, n_5)$, edge (n_1, n_5) can not be routed with two bends, but must have a single bend placed in such a way to share its z coordinate with n_1 and its y coordinate with n_5 (point $p_{t,2}$), or to share its z coordinate with n_5 and its y coordinate with n_1 (point $p_{f,2}$). □

Given a 2-bend drawing of the basic gadget, we call *true* the basic gadget when it is drawn with the bend of edge (n_1, n_5) placed in $p_{t,2}$ (see Fig. 10.a) and *false* the basic gadget when it is drawn with the bend of edge (n_1, n_5) placed in $p_{f,2}$ (see Fig. 10.b). Also, in what follows we use the graphic representation of the basic gadget shown in Fig. 10.c, where the nodes n_1, n_2, n_3, n_4, and $n_{1,5}$ are replaced by their bounding box, and analogously for the nodes n_5, n_6, n_7, n_8, and $n_{5,1}$. In this representation only edge (n_1, n_5) is shown, and it is assumed to have its bend in $p_{t,2}$.

Lemma 12 *In any non-intersecting 2-bend drawing of the basic gadget such that the internal points of the segments $\overline{p_{t,1}p_{t,2}}$ and $\overline{p_{t,2}p_{t,3}}$ are not used by any edge of the gadget segments $\overline{p_{f,1}p_{f,2}}$ and $\overline{p_{f,2}p_{f,3}}$ are used by edge (n_1, n_5).*

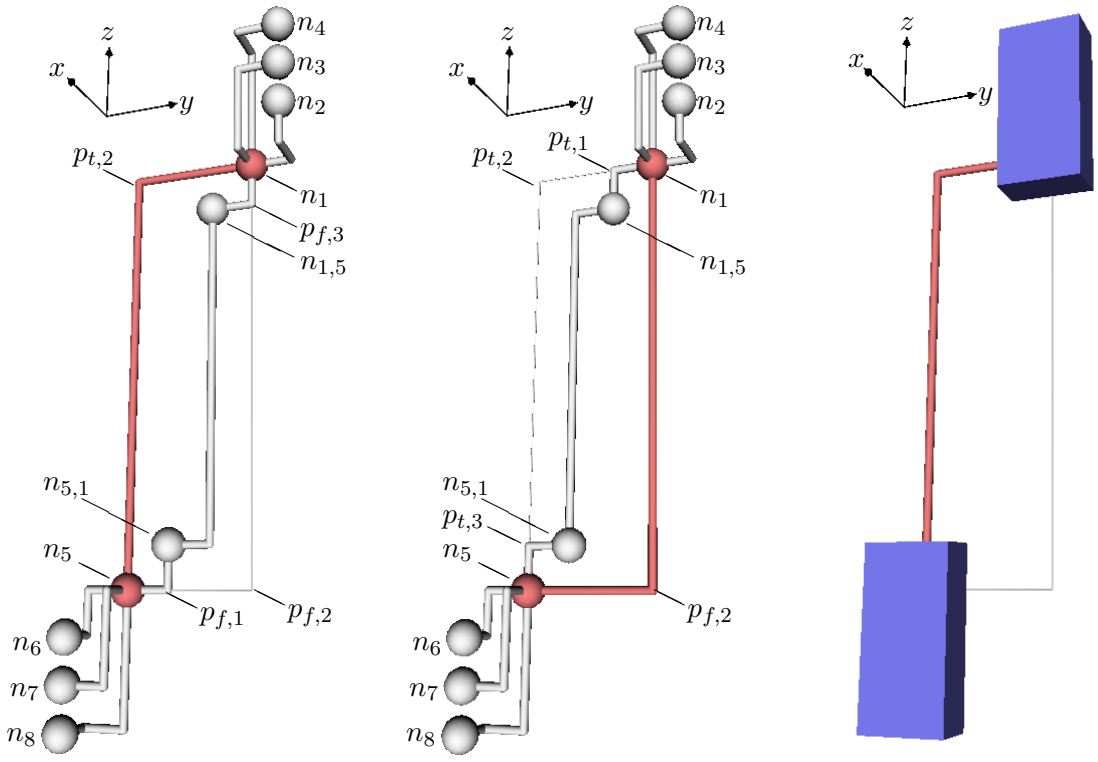


Figure 11: A true drawing (a) and a false drawing (b) of the basic gadget. In (c) it is shown the schematic representation of the basic gadget that is used in the remaining part of the paper.

5.2.2 The Variable Gadget

The variable gadget V_i is composed by a single basic gadget. Given a variable gadget V_i, we define as T_{V_i} (F_{V_i}) the set of non-intersecting 2-bend drawings of $G_\phi(V_\phi, E_\phi)$ such that the basic gadget is true (false). Lemma 11 guarantees that sets T_{V_i} and F_{V_i} correspond to a bipartition of the non-intersecting 2-bend drawings of G_ϕ.

5.2.3 The Joint Gadget

Fig. 12 shows how basic gadgets can be interleaved together. In fact, a basic gadget can be suitably rotated with respect to another basic gadget, and node positions can be chosen in such a way that if one basic gadget is true the other also need to be true. In particular, a variable gadget V_i can be intersected by a suitable number of basic gadgets, one for each clause in which the variable v_i participates, in order to transfer the geometric constraints determined by the drawing of V_i to the clause gadgets.

Given a clause $c_i = l_h \vee l_j \vee l_k$, where l_h (l_j, l_k, respectively) is a literal of the variable v_h (v_j, v_k, respectively) and $h < j < k$, the joint gadget $J_{i,k}$ is the reflected image with respect to the y axis of the joint gadget $J_{i,h}$. Thus, in the following we only describe the joint gadget $J_{i,h}$, which is depicted in Fig. 13 and built by interleaving four basic gadgets B_1, B_2, B_3, and B_4 as follows. B_1 intersects the variable gadget (not shown in Fig. 13.a). B_2 is placed on an orthogonal plane as shown in Fig. 13.a. B_3 intersects only B_2 and is placed on a plane orthogonal to the first two (see Fig. 13.b). Finally, B_4 is placed on a

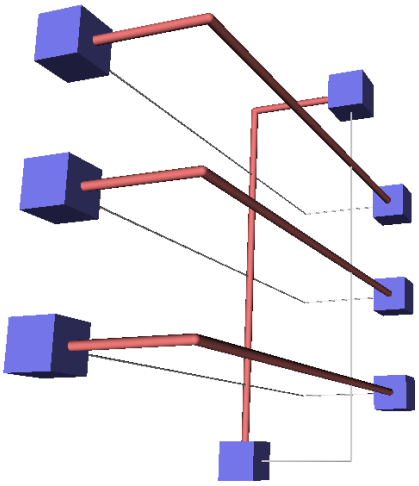


Figure 12: Four basic gadgets interleaved in such a way that in any 2-bend drawing of them they are all true or all false.

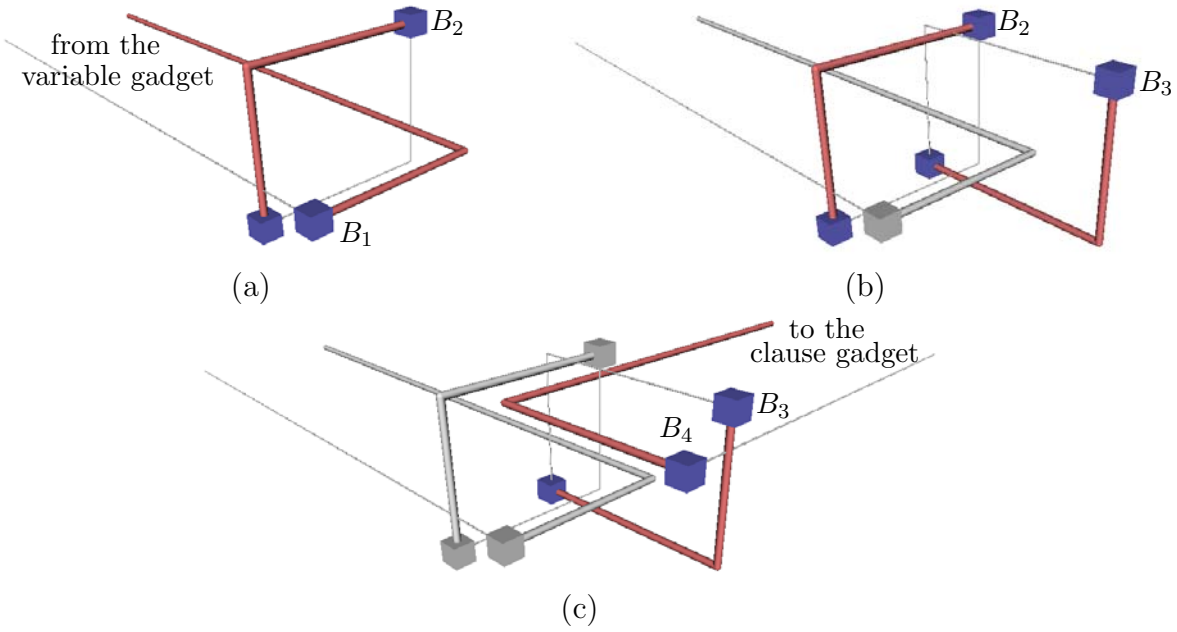


Figure 13: Joint gadget $J_{i,h}$ is composed by four interleaved basic gadgets.

plane parallel to the first one and intersects B_3 only as shown in Fig. 13.c. We define $T_{J_{i,h}}$ ($F_{J_{i,h}}$) as the set of non-intersecting 2-bend drawings of G_ϕ satisfying S_ϕ such that B_4 is true (false).

5.2.4 The Clause Gadget

The clause C_i for clause $c_i = l_h \vee l_j \vee l_k$ is shown in Fig. 14. It is composed by two nodes n_1 and n_2 placed at the opposite vertices of a cube. The two nodes are joined by edge (n_1, n_2) (not shown in Fig. 14). In any 2-bend drawing of the clause gadget edge (n_1, n_2) uses one of the four vertical edges of the cube. The basic gadget B_4 of joint gadgets $J_{i,h}$ and $J_{i,k}$ and the basic gadget coming coming from V_j suitably intersect the vertical edges

of the cube such that only if one literal is true the clause gadget admits a non-intersecting drawing.

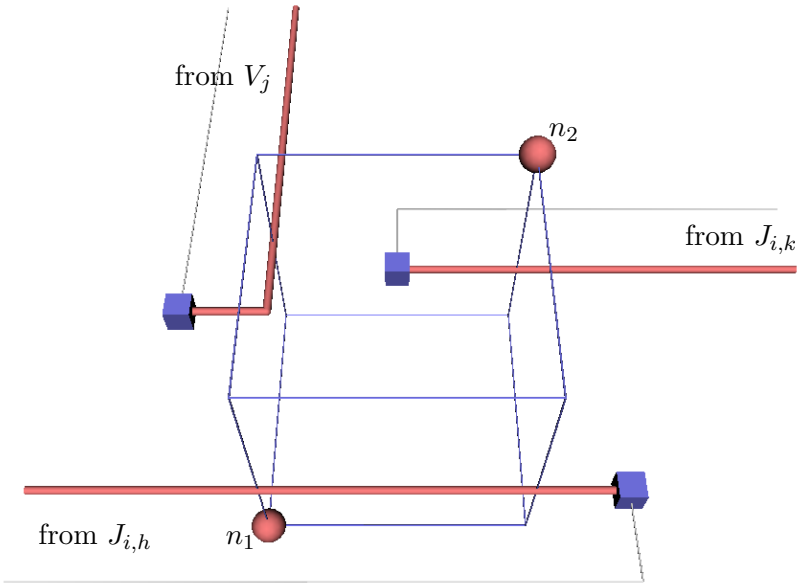


Figure 14: The clause C_i for clause $c_i = l_h \vee l_j \vee l_k$ in the case in which l_h, l_j and l_k are negative literals: if the three variables v_h, v_h and v_k are true, there is no way of adding edge (n_1, n_2) with at most two bend.

5.2.5 The Hardness Proof

By using Lemmas 11 and 12 it is easy to show the following lemma.

Lemma 13 *Statement 2 holds, that is, if a non-intersecting 2-bend drawing of G_ϕ satisfying exists with nodes at the prescribed positions, it belongs to $T_{J_{i,h}}$ if and only if it belongs to T_{V_h}.*

Lemma 14 *Statement 3 holds, that is, for each clause $c_i = l_h \vee l_j \vee l_k$ and for each non-intersecting drawing Γ of $G_\phi(V_\phi, E_\phi)$ with the nodes at the prescribed positions, at least one between the following conditions holds: (i) $\Gamma \in T_{J_{i,h}}$ ($\Gamma \in F_{J_{i,h}}$) and l_h is the positive (negative) literal of v_h. (ii) $\Gamma \in T_{V_j}$ ($\Gamma \in F_{V_j}$) and l_j is the positive (negative) literal of v_j. (iii) $\Gamma \in T_{J_{i,k}}$ ($\Gamma \in F_{J_{i,k}}$) and l_k is the positive (negative) literal of v_k.*

Proof: There is a way to route edge (n_1, n_2) with only two bends only if one of the four vertical edges of the cube of clause gadget C_i is not intersected by a basic gadget. If a drawing Γ of $G_\phi(V_\phi, E_\phi)$ satisfying S_ϕ exists, and edge (n_1, n_2) is routed with two bends, one of the edges is not blocked, and one of the three conditions in the statements is verified. □

Lemma 15 *Statement 4 holds, that is, if ϕ admits a solution, then $G_\phi(V_\phi, E_\phi)$ admits a non-intersecting drawing with nodes at the prescribed positions.*

Proof: Consider a truth assignment satisfying ϕ. If variable v_i is true (false) we can use the true (false) drawing of variable gadget V_i depicted in Fig. 11.a (Fig. 11.b). Also, for each clause $c_i = l_h \vee l_j \vee l_k$, at least one of its literals is true. This implies that one of the vertical edges of the clause gadget C_i is not blocked, and edge (n_1, n_2) can be routed with two bends without intersection. □

Lemma 16 2-Bend Routing *is NP-hard*

Proof: The proof is based on the fact that a compliant specification can be found for the 3SAT reduction framework introduced in Section 3. Lemmas 13, 14, and 15 prove that Statements 2, 3, and 4 hold, respectively. Since the 2-Bend Routing instance I_ϕ corresponding to a 3SAT instance ϕ can be built in polynomial time, Statement 1 also holds. Therefore, the construction rules described in Sections 5.2.1, 5.2.2, 5.2.3, and 5.2.4 correspond to a compliant specification for the 3SAT reduction framework, and Theorem 1 applies. □

6 Discussion and Open Problems

This paper shows that Simplicity Testing is NP-hard, while the reverse problem, Routing, is feasible. This asymmetry may explain why most three-dimensional drawing algorithms in the literature determine edge shapes as a consequence of node relative positions and not vice-versa.

6.1 Characterization of Simple Orthogonal Shapes

With respect to the problem of characterizing simple orthogonal shapes, deciding whether a shape graph is simple is shown here to be NP-hard. Of course, the problem of characterizing simple orthogonal shapes remains open, although we now know that in the general case it implies a heavy computation.

As a consequence of the complexity of the Simplicity Testing problem in the general case, in any hypothetical 3D drawing process in which the definition of the shape of the drawing is followed by the actual computation of its coordinates, the first step should be very carefully conceived in order for the second step to be efficiently computable. In fact, focusing on peculiar classes of shape graphs seems to be an obliged strategy for practical applications. Are there non trivial families of shape graph for which the simplicity testing is feasible? In particular, is there a "universal" set of shape graphs such that any graph is represented and such that the simplicity testing is guaranteed to be polynomial and to have a positive answer?

6.2 2-Bend Drawings

With respect to the problem of determining if a graph of degree six always admits a 2-bend drawing, this paper shows the NP-hardness of two problems related with finding such drawings. Namely, it is NP-hard when node positions are fixed (Section 4) and it is NP-hard when edge shapes are fixed (Section 5). Some other 3D drawing problems involving the number of the bends are known to be NP-hard, as, for example, finding a 2-bend drawing when vertices are placed on the diagonal of a cube [24] (provided that the graph

admits such a drawing). The number of NP-hard problems related with the computation of a 2-bend drawing rises the following question: What is the complexity of finding a 2-bend drawing of a graph? If finding such a drawing was also NP-hard, then any attempt to prove that the conjecture in [11] is false, should produce an algorithm for an intractable problem, which is hard to conceive without resorting to an enumerative approach (which, in turn, assumes the existence of a solution). However both the conception of such an algorithm and the description of a graph not admitting a 2-bend drawing appear to be elusive goals.

Acknowledgments

We would like to thank Giuseppe Di Battista and Giuseppe Liotta for constant encouragement and interesting conversations.

References

[1] T. C. Biedl. Heuristics for *3d*-orthogonal graph drawings. In *Proc. 4th Twente Workshop on Graphs and Combinatorial Optimization*, pages 41–44, 1995.

[2] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. Macmillan, London, 1976.

[3] F. Brandenburg, D. Eppstein, M. T. Goodrich, S. Kobourov, G. Liotta, and P. Mutzel. Selected open problems in graph drawing. In G. Liotta, editor, *Graph Drawing (Proc. GD 2003)*, volume 2912 of *Lecture Notes Comput. Sci.*, pages 515–539. Springer-Verlag, 2004.

[4] M. Closson, S. Gartshore, J. Johansen, and S. K. Wismath. Fully dynamic 3-dimensional orthogonal graph drawing. *Journal of Graph Algorithms and Applications*, 5(2):1–34, 2001.

[5] E. D. Demaine, J. S. B. Mitchell, and J. O'Rourke, (eds.). The Open Problems Project. `http://cs.smith.edu/~orourke/TOPP/Welcome.html`.

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

[7] G. Di Battista, G. Liotta, A. Lubiw, and S. Whitesides. Orthogonal drawings of cycles in 3d space. In Joe Marks, editor, *Graph Drawing (Proc. GD '00)*, volume 1984 of *Lecture Notes Comput. Sci.* Springer-Verlag, 2001.

[8] G. Di Battista, G. Liotta, A. Lubiw, and S. Whitesides. Embedding problems for paths with direction constrained edges. *J. of Theor. Comp. Sci.*, 289:897–917, 2002.

[9] G. Di Battista, M. Patrignani, and F. Vargiu. A split&push approach to 3D orthogonal drawing. *Giuseppe Liotta and Sue Whitesides, Guest editors, Journal of Graph Algorithms and Applications*, 4(3):105–133, 2000.

[10] P. Eades, C. Stirk, and S. Whitesides. The techniques of Kolmogorov and Bardzin for three dimensional orthogonal graph drawings. *Inform. Process. Lett.*, 60:97–103, 1996.

[11] P. Eades, A. Symvonis, and S. Whitesides. Two algorithms for three dimensional orthogonal graph drawing. In S. North, editor, *Graph Drawing (Proc. GD '96)*, volume 1190 of *Lecture Notes Comput. Sci.*, pages 139–154. Springer-Verlag, 1997.

[12] P. Eades, A. Symvonis, and S. Whitesides. Three dimensional orthogonal graph drawing algorithms. *Discrete Applied Math.*, 103(1-3):55–87, 2000.

[13] E. Di Giacomo, G. Liotta, and M. Patrignani. A note on 3d orthogonal drawings with direction constrained edges. *Information Processing Letters*, 90:97–101, 2004.

[14] B. Y. S. Lynn, A. Symvonis, and D. R. Wood. Refinement of three-dimensional orthogonal graph drawings. In Joe Marks, editor, *Graph Drawing (Proc. GD '00)*, volume 1984 of *Lecture Notes Comput. Sci.*, pages 308–320. Springer-Verlag, 2001.

[15] A. Papakostas and I. G. Tollis. Algorithms for incremental orthogonal graph drawing in three dimensions. *Journal of Graph Algorithms and Applications*, 3(4):81–115, 1999.

[16] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 3rd edition, October 1990.

[17] R. Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987.

[18] R. Tamassia, G. Di Battista, and C. Batini. Automatic graph drawing and readability of diagrams. *IEEE Trans. Syst. Man Cybern.*, SMC-18(1):61–79, 1988.

[19] G. Vijayan and A. Wigderson. Rectilinear graphs and their embeddings. *SIAM J. Comput.*, 14:355–372, 1985.

[20] D. R. Wood. On higher-dimensional orthogonal graph drawing. In J. Harland, editor, *Proc. Computing: the Australasian Theory Symposimum (CATS '97)*, volume 19, pages 3–8. Australian Computer Science Commission, 1997.

[21] D. R. Wood. An algorithm for three-dimensional orthogonal graph drawing. In S. H. Whitesides, editor, *Graph Drawing (Proc. GD '98)*, volume 1547 of *Lecture Notes Comput. Sci.*, pages 332–346. Springer-Verlag, 1998.

[22] D. R. Wood. *Three-Dimensional Orthogonal Graph Drawing*. PhD thesis, School of Computer Science and Software Engineering, 2000.

[23] D. R. Wood. Optimal three-dimensional orthogonal graph drawing in the general position model. *Theoretical Computer Science*, 299:151–178, 2003.

[24] D. R. Wood. Minimising the number of bends and volume in 3-dimensional orthogonal graph drawings with a diagonal vertex layout. *Algorithmica*, 39:235–253, 2004.