On the Computational Complexity of Upward and Rectilinear Planarity Testing*

Ashim Garg
ag@cs.brown.edu

Roberto Tamassia
rt@cs.brown.edu

Department of Computer Science
Brown University
Providence, RI 02912–1910, USA

Abstract

A directed graph is said to be upward planar if it can be drawn in the plane such that every edge is a monotonically increasing curve in the vertical direction, and no two edges cross. An undirected graph is said to be rectilinear planar if it can be drawn in the plane such that every edge is a horizontal or vertical segment, and no two edges cross. Testing upward planarity and rectilinear planarity are fundamental problems in the effective visualization of various graph and network structures. For example, upward planarity is useful for the display of order diagrams and subroutine-call graphs, while rectilinear planarity is useful for the display of circuit schematics and entity-relationship diagrams. In this paper we show that upward planarity testing and rectilinear planarity testing are NP-complete problems. We also show that it is NP-hard to approximate the minimum number of bends in a planar orthogonal drawing of an n-vertex graph with an $O(n^{1-\epsilon})$ error, for any $\epsilon > 0$.

Key Words: Graph drawing, planar drawing, upward drawing, rectilinear drawing, orthogonal drawing, layout, ordered set, planar graph, algorithm, computational complexity, NP-complete problem, approximation algorithm.

*Research supported in part by the National Science Foundation, by the U.S. Army Research Office, and by the Advanced Research Projects Agency. An extended abstract of this paper was presented at *Graph Drawing '94*, Princeton, New Jersey, October 10–12, 1994.

1 Introduction

Graph drawing addresses the problem of constructing geometric representations of abstract graphs and networks [7]. It is an emerging area of research that combines flavors of topological graph theory and computational geometry. The automatic generation of drawings of graphs has important applications in key computer technologies such as software engineering, database design, visual interfaces, and computer-aided-design.

Various graphic standards have been proposed for the representation of graphs in the plane. Usually, vertices are represented by points, and each edge (u, v) is represented by a simple open Jordan curve joining the points associated with the vertices u and v. A *straight-line* drawing maps each edge into a straight-line segment. A drawing is *planar* if no two edges cross. A graph (or digraph) is planar if it admits a planar drawing. A drawing of a digraph is *upward* if every edge is monotonically nondecreasing in the y-direction. A digraph is *upward planar* if it admits a planar upward drawing. Fig. 1(a) shows a planar straight-line upward drawing. An *orthogonal* drawing maps each edge into a chain of horizontal and vertical segments. A *rectilinear* drawing is an orthogonal straight-line drawing, i.e., a drawing where every edge is either a horizontal or a vertical segment. A graph is *rectilinear planar* if it admits a planar rectilinear drawing. Fig. 1(b) shows a planar rectilinear drawing.

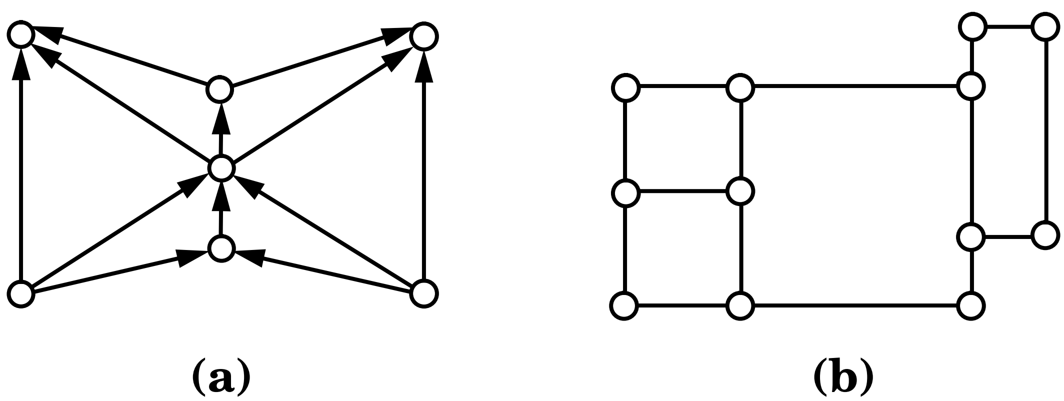

Figure 1: Examples of **(a)** a planar straight-line upward drawing of a digraph; and **(b)** a planar rectilinear drawing of a graph.

Testing upward planarity and rectilinear planarity are fundamental problems in the effective visualization of various graph and network structures. For example, upward planarity is useful for the display of order diagrams and subroutine-call graphs, while rectilinear planarity is useful for the display of circuit schematics and entity-relationship diagrams. In this paper we show that the following two problems are NP-complete:

Upward planarity testing: testing whether a digraph is upward planar.

Rectilinear planarity testing: testing whether a graph is rectilinear planar.

These problems have challenged researchers in order theory, topological graph theory, computational geometry, and graph drawing for many years. Our intractability results motivate the following observations:

- Testing whether a graph admits a planar drawing or an upward drawing can be done in linear time. Combining the two properties makes the problem NP-hard.
- Every planar graph admits a planar straight-line drawing. Hence, we can say that planarity is equivalent to straight-line planarity, and both properties can be verified in linear time. We

can view upward and rectilinear planarity as derived from straight-line planarity by adding further constraints, which make the problem become apparently much more difficult.

We also show that it is NP-hard to approximate the minimum number of bends in a planar orthogonal drawing of an n-vertex graph with an $O(n^{1-\epsilon})$ error, for any $\epsilon > 0$.

Previous results on upward and rectilinear planarity testing are summarized below. In the rest of this section, we denote with n the number of vertices of the graph being considered.

Combinatorial results on upward planarity of covering digraphs of lattices were first given in [17, 24]. Further results on the interplay between upward planarity and ordered sets are surveyed by Rival [25, 26, 27]. Lempel, Even, and Cederbaum [18] relate the planarity of biconnected undirected graphs to the upward planarity of *st*-digraphs. A combinatorial characterization of upward planar digraphs is provided in [10, 16]: namely, a digraph is upward planar if and only if it is a spanning subgraph of a planar *st-digraph*. This characterization implies that upward planarity testing is in NP.

Di Battista, Liu, and Rival [9] show that every planar bipartite digraph is upward planar. Papakostas [23] gives a polynomial-time algorithm for upward planarity testing of outerplanar digraphs. Bertolazzi, Di Battista, Liotta, and Mannino [3, 4] give a polynomial-time algorithm for testing upward planarity of triconnected digraphs and of digraphs with a fixed embedding. Concerning single-source digraphs, Thomassen [33] characterizes upward planarity in terms of forbidden circuits. Hutton and Lubiw [14] combine Thomassen's characterization with a decomposition scheme to test upward planarity of a single-source digraph in $O(n^2)$ time. Bertolazzi, Di Battista, Mannino, and Tamassia [5] show that upward planarity testing of a single-source digraph can be done optimally in $O(n)$ time. They also give a parallel algorithm that runs in $O(\log n)$ time on a CRCW PRAM with $n \log \log n / \log n$ processors.

Di Battista, Tamassia, and Tollis [10, 11] give algorithms for constructing upward planar drawings of planar *st*-digraphs, and investigate area bounds and symmetry display. Tamassia and Vitter [32] show that the above drawing algorithms can be efficiently parallelized. Upward planar drawings of series-parallel digraphs are studied ind [1, 2].

Regarding rectilinear planarity testing, Shiloach [28] and Valiant [34] show that any planar graph of degree at most 4 admits a planar orthogonal drawing. Vijayan and Wigderson [35] study structural properties of rectilinear planar drawings. From their results, the membership of rectilinear planarity testing in NP is easy to establish. Storer [29], Tamassia and Tollis [31], Liu, Marchioro, Morgana, Petreschi, and Simeone [20, 21, 22, 19], Even and Granot [12], and Biedl and Kant [15, 6] give various techniques for constructing planar orthogonal drawings with $O(n)$ bends. Tamassia [30] gives an $O(n^2 \log n)$-time algorithm that constructs a planar orthogonal drawing with the minimum number of bends for an embedded planar graph. Di Battista, Liotta, and Vargiu [8] give polynomial time algorithms for minimizing bends in planar orthogonal drawings of series-parallel and cubic graphs. The latter two results show that rectilinear planarity testing can be done in polynomial time for a fixed embedding or for special classes of graphs.

Our proof techniques are based on a two-phase reduction from the known NP-complete problem NOT-ALL-EQUAL-3-SAT. In the first phase, we reduce NOT-ALL-EQUAL-3-SAT to an auxiliary undirected flow problem. In the second phase, we reduce this undirected flow problem to the upward (or rectilinear) planarity testing of a special class of digraphs. The latter reduction is interesting in its own and provides new insights on the characterization by flow networks of the angles formed by the edges of upward planar drawings [3, 4] and orthogonal drawings [8, 30].

The rest of this paper is organized as follows. Preliminary definitions and results are provided in Section 2. The reduction from NOT-ALL-EQUAL-3-SAT to the auxiliary flow problem is given in Section 3. Sections 4 and 5 describe the reductions from the auxiliary flow problem to upward and rectilinear planarity testing, respectively. Conclusive remarks are given in Section 6.

2 Preliminaries

We assume standard concepts and definitions on NP-completeness [13]. Our results use reductions from the following well-known NP-complete problem:

NOT-ALL-EQUAL-3-SAT Given a set of clauses with three literals each, is there a truth assignment such that each clause has at least one true literal and one false literal?

An *embedding* of a planar graph is the collection of circular permutations of the edges incident upon each vertex in a planar drawing of the graph. An *embedded graph* is a planar graph equipped with an embedding. We do not distinguish between a graph and its embedding if the embedding is unique and the meaning is clear from the context.

We denote by $G - \{v\}$ the subgraph of graph G obtained by removing vertex v and its incident edges from G.

The *angles* of an embedded graph are the pairs of consecutive edges incident on the same vertex. Such angles are mapped to geometric angles in a straight-line drawing of the graph.

A *rectilinear embedding* of a graph G is an embedding of G where each angle is assigned a label in the set $\{1, 2, 3, 4\}$, such that there exists a rectilinear drawing of G where each angle labeled ℓ in the embedding measures $\ell\pi/2$ in the drawing. Each rectilinear embedding has a unique external face.

The following definitions are from [3, 4]. An *upward embedding* of a digraph $\vec{G}$ is an embedding of $\vec{G}$ where each angle formed by pairs of incoming or outgoing edges is assigned a label in the set $\{$*small*, *large*$\}$, such that there exists a planar straight-line upward drawing of $\vec{G}$ where each angle labeled *small* has measure $< \pi$ and each angle labeled *large* has measure $> \pi$. Each upward embedding has a unique external face.

A *source* of a a digraph $\vec{G}$ is a vertex with all outgoing edges and a *sink* of $\vec{G}$ is a vertex with all incoming edges. A *switch* of $\vec{G}$ is a source or sink of $\vec{G}$. A source or sink of a face f of $\vec{G}$ is called a *local* source or sink of f. Note that a local switch of f may or may not be a switch of $\vec{G}$.

2.1 Upward and Rectilinear Embeddings

In this section we give some lemmas on upward and rectilinear embeddings that will be extensively used in the proofs.

First, some definitions. In an embedding of a digraph $\vec{G}$, a vertex has the *bimodal property* if the cyclic order of the edges incident on it can be partitioned into two sets of consecutive edges, one (possibly empty) consisting entirely of its incoming edges and the other (also possibly empty) consisting entirely of its outgoing edges. An embedding of $\vec{G}$ is *bimodal* if each vertex in it has the bimodal property.

Consider an assignment of labels from the set $\{$*small*, *large*$\}$ to the angles between incoming or outgoing edges of an embedding of $\vec{G}$. For a face f of the embedding, let $L(f)$ and $S(f)$ be the number of angles of f with label *large* and *small*, respectively. Face f is said to be *consistently assigned* if:

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \text{ is an internal face,} \\ +2 & \text{if } f \text{ is the external face.} \end{cases}$$

An assignment of labels to the angles of an embedding of digraph $\vec{G}$ is a *consistent assignment* if:

- all the angles at a nonsource or nonsink vertex of $\vec{G}$ are assigned label *small*;
- exactly one angle at a source or sink vertex of $\vec{G}$ is assigned label *large*; and
- each face is consistently assigned.

We paraphrase a result of [3, 4] in the following lemma:

Lemma 1 *An embedding of a digraph $\vec{G}$ can be extended to an upward embedding if and only if it is bimodal and admits a consistent assignment of labels to its angles.*

Let G be an undirected graph of degree at most 4. Consider an assignment of labels from the set $\{1, 2, 3, 4\}$ to the angles of an embedding of G. For a face f of the embedding, let $N_i(f)$ be the number of angles of f with label i. Face f is said to be *consistently rectilinearly assigned* if:

$$2 \cdot N_4(f) + N_3(f) - N_1(f) = \begin{cases} -4 & \text{if } f \text{ is an internal face,} \\ +4 & \text{if } f \text{ is the external face.} \end{cases}$$

An assignment of labels to the angles of an embedding of a graph G is a *consistent rectilinear assignment* if:

- the sum of the labels of the angles around each vertex is 4; and
- each face is consistently rectilinearly assigned.

The following lemma is an immediate consequence of the results in [30, 35].

Lemma 2 *An embedding of a graph G can be extended to a rectilinear embedding if and only if it admits a consistent rectilinear assignment of labels to its angles.*

2.2 Tendrils and Wiggles

We now define several graphs that will be used as gadgets in our reductions.

We show in Fig. 2(a) *tendril* T_k ($k \geq 1$), which is an acyclic digraph with $k+1$ sources and $k+1$ sinks. We also define tendril T_0 as a a digraph consisting of a single edge. Tendril T_k ($k \geq 0$) has a designated source and a designated sink, called the *poles* of T_k. We shall consider transformations where a directed edge (u, v) of a digraph is replaced with a tendril T_k, where the source is identified with u and the sink with t.

Lemma 3 *Tendril T_k is upward planar and admits a unique upward embedding.*

Proof: We use induction on k. Tendril T_1 is shown in Fig. 3(a). Tendril T_k can be constructed by adding to T_{k-1} the digraph H_k shown in Fig. 3(b) and by identifying edges e_{k-1} of T_{k-1} and f_k of H_k, as shown in Fig. 3(c). It is straightforward to verify that T_1 and H_k have a unique upward embedding by applying Lemma 1 to their $O(1)$ planar embeddings. This proves the base of the induction. For the inductive step, since T_{k-1} and H_k each have a unique upward embedding, we only need to consider the four planar embeddings of T_k obtained by flipping the upward embeddings of T_{k-1} and H_k around their common edge f_k. Applying Lemma 1 to the four faces containing both endpoints of f_k shows that only one of the above planar embeddings of T_k can be extended to an upward embedding. □

In the upward planar embedding of T_k, the external face consists of two paths between s and t. One such path, called *outer path*, has $2k$ large angles and no small angles, and the other path, called *inner path*, has $2k$ small angles and no large angles. When a tendril replaces an edge of an embedded planar digraph, the outer path becomes a subpath of a face, and we say that the *contribution* of the outer path to the face is $+2k$. Similarly, we say that the contribution of the inner path to its face is $-2k$.

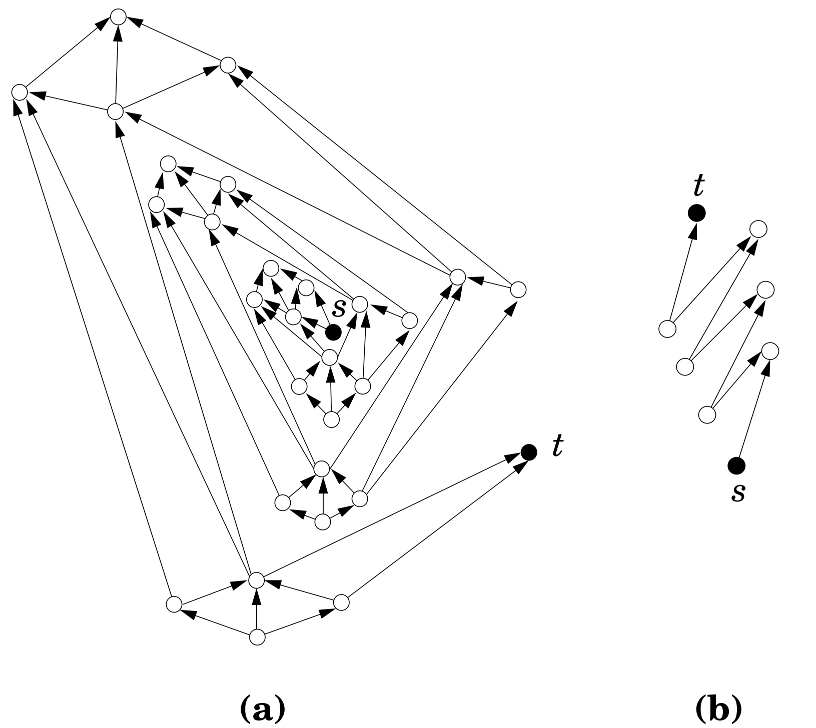

Figure 2: Examples of: **(a)** Tendril T_3; and **(b)** Wiggle W_3.

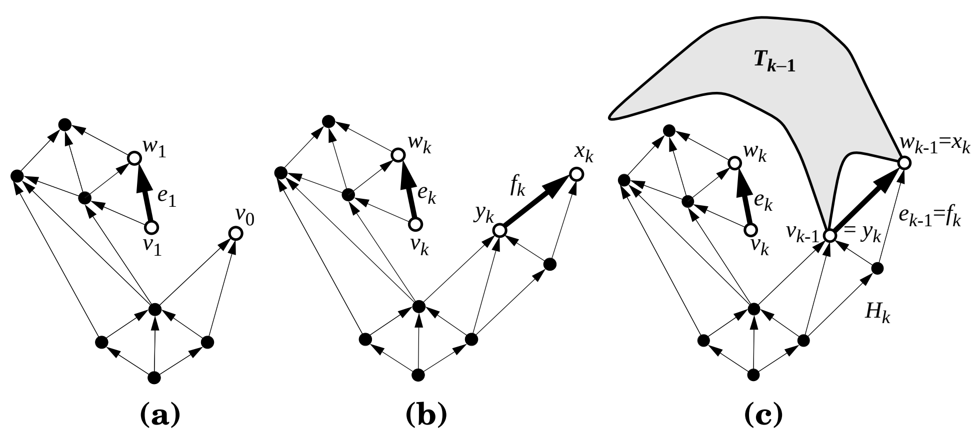

Figure 3: **(a)** Tendril T_1. **(b)** Graph H_k. **(c)** Constructing T_k from T_{k-1}.

Figure 2(b) shows a *wiggle* W_k, which is an acyclic digraph consisting of a chain of $2k+1$ edges whose orientation alternates along the chain. The extreme vertices of W_k, a source s and a sink t, are called the *poles* of W_k. An *arrangement* of wiggle W_k is an upward embedding of W_k. We shall consider transformations where a directed edge (u, v) of an embedded digraph is replaced with wiggle W_k, where s is identified with u and t with v, and W_k becomes a subpath of two faces. Given an arrangement of W_k, we say that the *contribution* of W_k to a face f containing W_k is the number of large angles minus the number of small angles of W_k in f. Clearly, W_k can be arranged to give to a face any contribution c such that c is an even number between 0 and $2k$. Note that if G_k gives contribution c to a face, it gives contribution $-c$ to the other face it belongs to.

We show in Fig. 4 an undirected graph called *rectilinear tendril* T_k, which also has two designated poles denoted by s and t. We also define rectilinear tendril T_0 as a a graph consisting of a single edge.

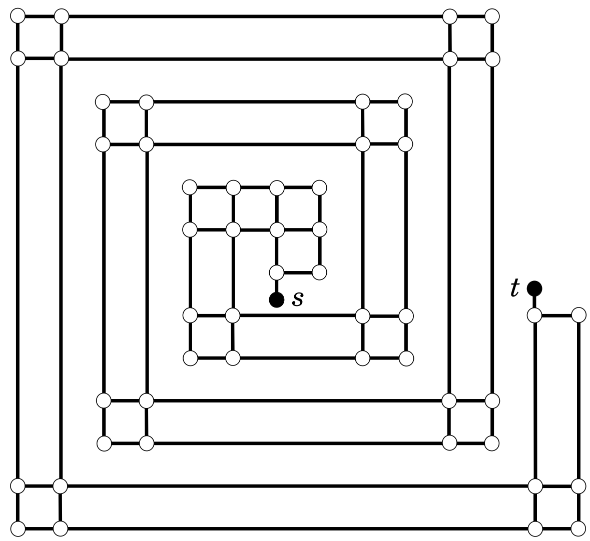

Figure 4: Example of rectilinear tendril T_3.

Lemma 4 *Rectilinear tendril T_k is rectilinear planar and admits exactly four rectilinear embeddings.*

Proof: The proof has the same flavor as the proof of Lemma 3. We use induction on k and apply Lemma 2. Rectilinear tendril T_1 is shown in Fig 5(a). As shown in Fig. 5(c), T_k can be constructed from T_{k-1} by removing vertex s_{k-1} and identifying its edge e_{k-1} with the edge f_k of the rectilinear planar graph H_k shown in Fig. 5(b). □

We show in Fig. 7 *rectilinear wiggle* W_k, which is a chain of $4k+1$ edges.

The *contribution* of a rectilinear tendril (or wiggle) to a face containing it is the number of angles of the tendril (or wiggle) labeled 3 minus the number of angles labeled 1 that lie in the face. Rectilinear tendril T_k contributes $4k$, $4k+1$, or $4k+2$ to one face, and the opposite value to the other face. Rectilinear wiggle W_k contributes to one face any value between 0 and $4k$, and the opposite value to the other face.

3 An Auxiliary Undirected Flow Problem

In this section we define two auxiliary flow problems and show that they are equivalent to NOT-ALL-EQUAL-3-SAT under polynomial-time reductions.

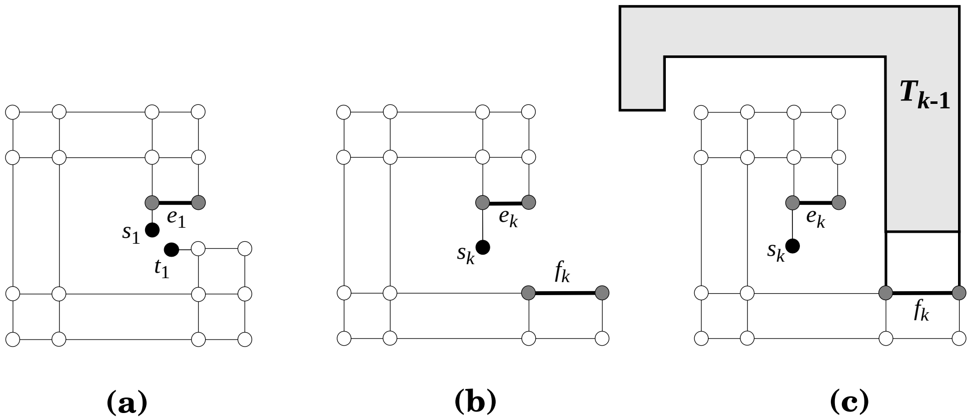

Figure 5: **(a)** Rectilinear Tendril T_1. **(b)** Graph H_k. **(c)** Constructing T_k from T_{k-1}.

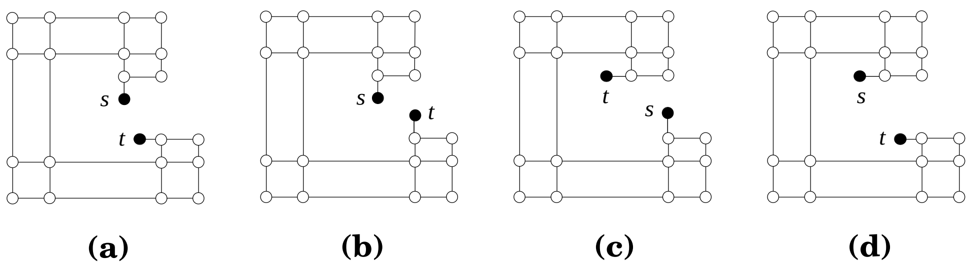

Figure 6: The four different rectilinear embeddings of rectilinear tendril T_1.

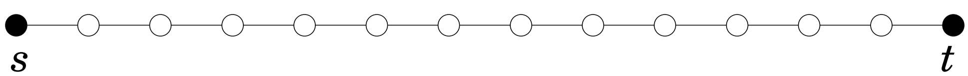

Figure 7: Example of a rectilinear wiggle W_3.

A *switch-flow network* is an undirected flow network $\mathcal{N}$ where each edge is labeled with a range $[c' \cdots c'']$ of nonnegative integer values, called the *capacity range* of the edge. For simplicity, we denote the capacity range $[c \cdots c]$ with $[c]$. A *flow* for a switch-flow network is an orientation of and an assignment of integer "flow" values to the edges of the network. A *feasible flow* is a flow that satisfies the following two properties:

Range property: the flow assigned to an edge is an integer within the capacity range of the edge.

Conservation property: the total flow entering a vertex from the incoming edges is equal to the total flow exiting the vertex from the outgoing edges.

Starting from an instance $\mathcal{S}$ of NOT-ALL-EQUAL-3-SAT, we construct a switch-flow network $\mathcal{N}$ as follows (see Figs. 8–9). Let the literals of $\mathcal{S}$ be denoted with $x_1, y_1, \cdots, x_n, y_n$, where $y_i = \overline{x_i}$, and the clauses of $\mathcal{S}$ be denoted with $c_1, \cdots, c_m$. Let θ be a positive integer parameter. We denote with α_i and β_i ($i = 1, \cdots, n$) the number of occurrences of literals x_i and y_i in the clauses of $\mathcal{S}$, respectively. Note that $\sum_{i=1}^{n}(\alpha_i + \beta_i) = 3m$. Also, we define $\gamma_i = (2i-1)\theta$ and $\delta_i = 2i\theta$ ($i = 1, \cdots, n$). Network $\mathcal{N}$ has a *literal vertex* for each literal of $\mathcal{S}$ and a *clause vertex* for each clause of $\mathcal{S}$, plus a special dummy vertex z. There are three types of edges in $\mathcal{N}$ (see Fig. 9):

Literal edges joining pairs of literals associated with the same boolean variable. The capacity range of literal edge (x_i, y_i) is $[\alpha_i\gamma_i + \beta_i\delta_i]$.

Clause edges joining each literal to each clause. The capacity range of clause edge (x_i, c_j) is $[\gamma_i]$ if $x_i \in c_j$, and $[0]$ otherwise. The capacity range of clause edge (y_i, c_j) is $[\delta_i]$ if $y_i \in c_j$, and $[0]$ otherwise.

Dummy edges joining each literal and each clause to the dummy vertex. The capacity ranges of dummy edges (z, x_i) and (z, y_i) are $[\beta_i\delta_i]$ and $[\alpha_i\gamma_i]$, respectively. The capacity range of dummy edge (z, c_j) is $[0 \cdots \eta_j - 2\theta]$, where η_j is the sum of the capacities of the clause edges incident on c_j.

The construction of network $\mathcal{N}$ from $\mathcal{S}$ is straightforward, and we have:

Lemma 5 *Given an instance $\mathcal{S}$ of* NOT-ALL-EQUAL-3-SAT *with n variables and m clauses, the associated switch-flow network $\mathcal{N}$ has $O(n+m)$ vertices and $O(nm)$ edges, and can be constructed in $O(nm)$ time.*

A feasible flow in network $\mathcal{N}$ corresponds to a satisfying truth assignment for $\mathcal{S}$. Namely, we have that a literal is true whenever its incident literal edge is incoming in the feasible flow (see Fig. 8(b)) and its incident clause edges with nonzero capacity range are outgoing. We formalize this correspondence in the following lemma.

Lemma 6 *An instance $\mathcal{S}$ of* NOT-ALL-EQUAL-3-SAT *is satisfiable if and only if the associated switch-flow network $\mathcal{N}$ admits a feasible flow. Also, given a feasible flow for $\mathcal{N}$, a satisfying truth assignment for $\mathcal{S}$ can be computed in time $O(nm)$, where n and m are the number of variables and clauses of $\mathcal{S}$, respectively.*

Proof:

$\Rightarrow$ Given a feasible flow in $\mathcal{N}$, we construct a truth assignment A by setting a literal true if and only if its incident literal edge is incoming in the flow. We now show that this is a satisfying assignment. Clearly, the two literals x_i and y_i associated with the same boolean variable consistently receive opposite truth assignments.

Because of the conservation property all the incident clause edges with nonzero capacity range of a true literal are outgoing, and the amount of flow in each of them is equal to the capacity. Conversely ,if a literal is false, then because of the conservation property all its incident clause edges with nonzero capacity range are incoming and the amount of flow in each of them is equal to the capacity. The three clause edges with nonzero capacity range incident on a clause vertex c_j cannot be all incoming or all outgoing because of the conservation property at vertex c_j and the choice of the capacity range for the dummy edge incident on c_j. Therefore, three literals in c_j can not be all true or all false. Hence, A is a satisfying truth assignment for $\mathcal{S}$. Also, A can be constructed in time linear in the number of edges of $\mathcal{N}$, which is $O(nm)$.

$\Leftarrow$ Let A be a satisfying truth assignment of $\mathcal{S}$. We construct a feasible flow f in $\mathcal{N}$ from A. In this flow,

- The amount of flow through the literal, clause and the dummy edges incident on literal vertices is equal to their capacities. The flow therefore satisfies the range property in these edges.
- If a literal is true then its incident literal edge is incoming and its incident clause edges with nonzero capacity range and its dummy edge are outgoing. If a literal is false then its incident literal edge is outgoing and its incident clause edges with nonzero capacity and its dummy edge are incoming. In either case the amount of flow coming into a literal l_i is $\alpha_i\gamma_i + \beta_i\delta_i$

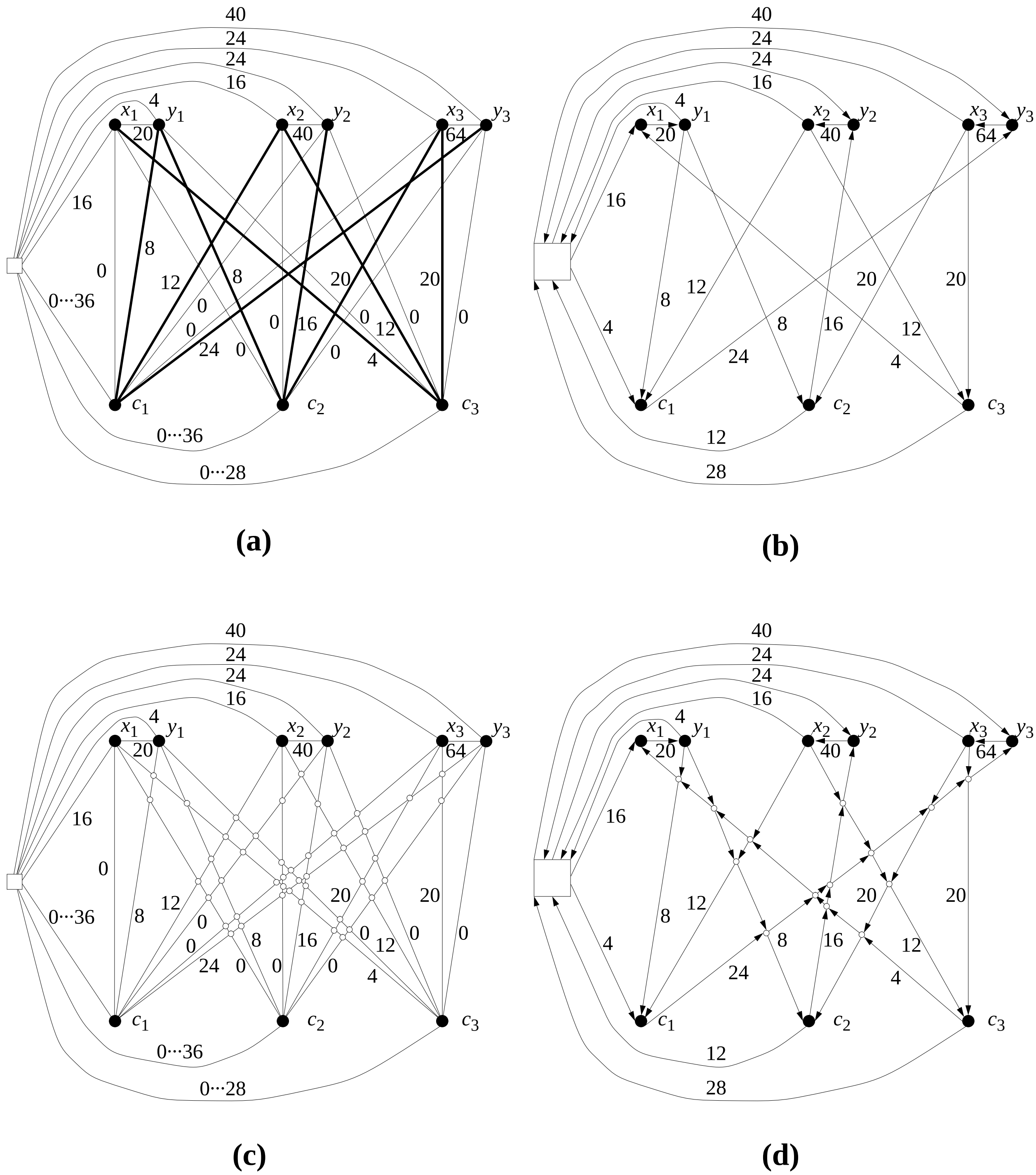

Figure 8: **(a)** Switch-flow network $\mathcal{N}$ with parameter $\theta = 4$ associated with the the NOT-ALL-EQUAL-3-SAT instance $\mathcal{S}$ with clauses $c_1 = y_1x_2y_3$, $c_2 = y_1y_2x_3$, and $c_3 = x_1x_2x_3$. The clause edges with nonzero capacity range are shown with thick lines. **(b)** Feasible flow for $\mathcal{N}$ corresponding to the satisfying truth assignment (y_1, x_2, x_3) for $\mathcal{S}$. Only the edges with nonzero flow are shown. **(c)** Planar switch-flow network $\mathcal{P}$ associated with $\mathcal{S}$. **(d)** Feasible flow for $\mathcal{P}$ corresponding to the satisfying truth assignment (y_1, x_2, x_3) for $\mathcal{S}$. Only the edges with nonzero flow are shown.

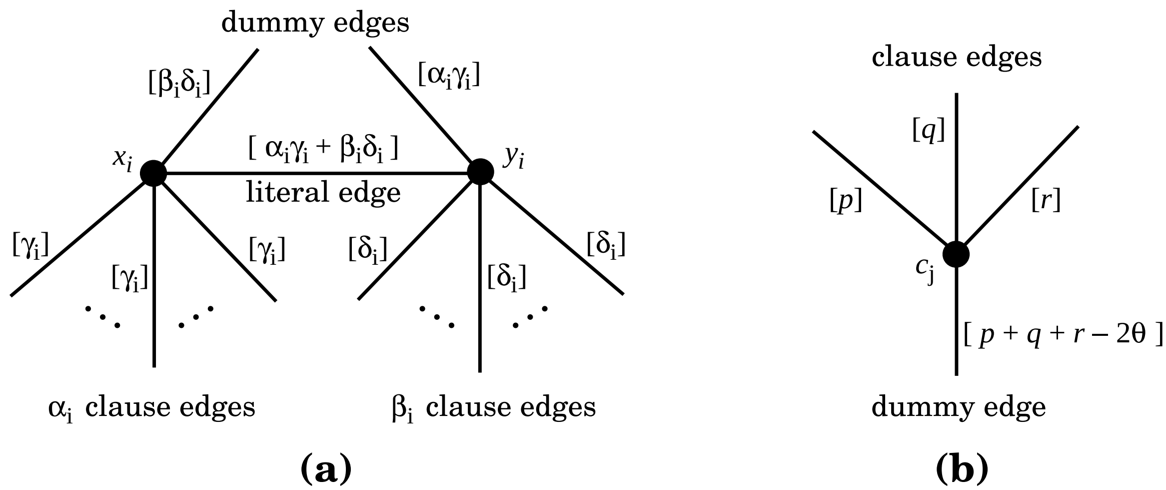

Figure 9: Schematic illustration of the edges incident on the literal and clause vertices of network N: **(a)** literal vertices x_i and y_i; **(b)** clause vertex c_j.

and the amount of flow leaving it is $\alpha_i\gamma_i + \beta_i\delta_i$. Therefore the flow satisfies the conservation property at these vertices.

- If the net flow entering a clause vertex through the clause edges is ν_j, then the flow leaving it through its dummy edge is ν_j. Conversely, if the net flow leaving a clause through the clause edges is ν_j, then the flow entering it through its dummy edge is ν_j. Since A is a satisfying assignment, the three clause edges with nonzero capacity range incident on a clause vertex can not be all incoming or all outgoing. A positive flow in a clause edge has is greater than or equal to θ. Therefore, the net flow coming into or leaving a clause vertex c_j through its clause edges is at most $\eta_j - 2\theta$. Hence, the conservation property at c_j and the range property in its dummy edge edge are both satisfied by the flow.
- The conservation property holds at the dummy vertex because of the following argument. By successive mergers of two nondummy vertices of $\mathcal{N}$ and elimination of the resultant self loops and the flow through them, we can reduce $\mathcal{N}$ into a graph with multiple edges between two vertices– the dummy vertex and another vertex u for which the conservation property holds. The net flow between these two vertices is same as in f. Since the conservation property holds for u and the dummy vertex is neither a source nor a sink, the conservation property holds for the dummy vertex also.

□

Now, starting from $\mathcal{S}$, we construct a planar switch-flow network $\mathcal{P}$ (see Fig. 8). We first construct a *layered drawing* $\psi_{\mathcal{N}}$ of $\mathcal{N}$ as follows (see Fig. 8(a)):

- Each literal and clause edge is drawn as a straight-line. The dummy edges are drawn as continuous curves.
- The clause vertices are horizontally aligned and ordered $c_1.c_2, \ldots, c_m$ from left to right.
- The literal vertices horizontally aligned above the clause vertices, and ordered $x_1, x_2, y_1, y_2, \ldots, x_m, y_m$ from left to right.
- There are crossings only between the clause edges. However, no more than two clause edges cross at the same point.

We next replace the crossings of $\psi_{\mathcal{N}}$ with vertices called the *crossing* vertices, thus splitting the clause edges at the crossing vertices. We call *fragment edges* or simply *fragments*, the edges originated by the splitting of the clause edges. Each fragment edge inherits the capacity range of the originating clause edge. We define the *facial degree* of a vertex as the total number of edges in its incident faces.

Lemma 7 *Given network $\mathcal{N}$ representing an instance $\mathcal{S}$ of* NOT-ALL-EQUAL-3-SAT *with $n \geq 3$ variables and $m \geq 3$ clauses, the associated planar switch-flow network $\mathcal{P}$ is triconnected, has $O(n^2m^2)$ vertices and edges, and can be constructed in $O(n^2m^2)$ time. Also, in the unique embedding of $\mathcal{P}$, the facial degree of each vertex is at most $7nm$.*

Proof: Let $\psi_{\mathcal{N}}$ be the layered drawing of $\mathcal{N}$ that is used to construct $\mathcal{P}$ (see Fig. 8(a)). There are $O(n^2m^2)$ crossings in $\psi_{\mathcal{N}}$. Therefore, $\mathcal{P}$ has $O(n^2m^2)$ crossing vertices and fragment edges. Consequently, $\mathcal{P}$ has $O(n^2m^2)$ vertices and edges.

Now we show that $\mathcal{P}$ is triconnected. We denote by u_k, the crossing vertex that corresponds to the crossing between the line joining c_k and y_n, and the line joining c_{k+1} and x_1. We denote by v_k, the crossing vertex that corresponds to the crossing between the line joining y_k and c_m, and the line joining x_{k+1} and c_1. We call the vertices of type u_k and v_k as the *bounding* crossing vertices of $\mathcal{P}$. No two bounding vertices are identical because they correspond to crossing between different pairs of lines. Hence there are two vertex disjoint paths $p = x_1y_1v_1 \dots x_ky_kv_k, x_{k+1} \dots y_n$ and $q = c_1u_1c_2 \dots c_ku_kc_{k+1} \dots c_m$ in $\mathcal{P}$. From each vertex there are two vertices disjoint (possibly empty) paths, one to a clause vertex and the other to a literal vertex. For the dummy vertex z, these paths consist of the dummy edges; For every nondummy vertex, these paths consist of the fragment edges of a clause edge whose fragment edges are incident on it. We have that between any two nondummy vertices there are two vertex disjoint paths not containing z. One such path contains a (possibly empty) subpath of p, and the other path contains a (possibly empty) subpath of q. Hence graph $\mathcal{P} - \{z\}$ is biconnected. Now suppose, for a contradiction, that $\mathcal{P}$ is not triconnected. Since the graph $\mathcal{P} - \{z\}$ is biconnected, z is not a member of a separating pair of $\mathcal{P}$. Let C_1 and C_2 be two connected components of $\mathcal{P}$ obtained after removing a separating pair. Suppose z is in C_1. Hence there are at most two vertex disjoint paths between z and the vertices in C_2. But between a vertex w of $\mathcal{P}$ and z there are at least four vertex disjoint paths (e.g., for a crossing vertex v, the four paths go through the endpoints of the two clause edges of $\mathcal{N}$ whose crossing is associated with v). Thus, we obtain a contradiction. Thererfore $\mathcal{P}$ is triconnected.

Now we show that the facial degree of any vertex is at most $7nm$. All the faces incident on the dummy vertex z have at most four vertices: one is z, two of them are clause and/or literal vertices and the fourth (if present) is a bounding crossing vertex. There are $n-1+m-1$ bounding crossing vertices in $\mathcal{P}$. A simple counting argument shows that the facial degree of z is $3n + m + 2(n-1+m-1) + 2 = 5n + 3m - 2$. Let f be a face that does not contain z. Face f contains at least one crossing vertex, and at most two fragments of clause edges incident on the same clause or literal vertex. Hence, the number of edges in face f is at most $\min\{4n, 2m\}$. The degree of each nondummy vertex is at most $\max\{2n+1, m+2, 4\}$. Consequently, the facial degree of each vertex is at most $7nm$, for $n, m \geq 3$.

Finally we show how to construct $\mathcal{P}$ from $\mathcal{N}$. Suppose the literals are numbered from 1 to $2n$ so that literal $l_{2k-1} = x_k$ and $l_{2k} = y_k$. We inductively construct a layered drawing $\psi(k, m)$ of the subgraph of $\mathcal{N}$ induced by literals $l_1, \cdots, l_k$ and clauses $c_1, \cdots, c_m$ (see Fig. 10). Drawing $\psi(2, m)$ is shown in Fig. 10(a). Suppose we have already constructed drawing $\psi(k-1, m)$ (see Fig. 10(b)). We place literal l_k at the same height as l_{k-1} and sufficiently far to its right so that that edge (l_k, c_1) intersects each clause edge of $\psi(k-1, m)$ below its lowest crossing. Replacing the crossings of $\psi(2n, m)$ by crossing vertices gives us the planar network $\mathcal{P}$.

The construction of the network $\mathcal{P}$ does not require computing the exact coordinates of the vertices and crossings of $\psi(2n, m)$. In the actual implementation, $\mathcal{P}$ is constructed directly by maintaining and updating at each inductive step a list of the lowest crossing vertices of the clause edges. The manipulation of this list takes constant time per crossing vertex and hence can be done in total $O(n^2m^2)$ time. The rest of the construction can also be carried out in $O(n^2m^2)$ time.

□

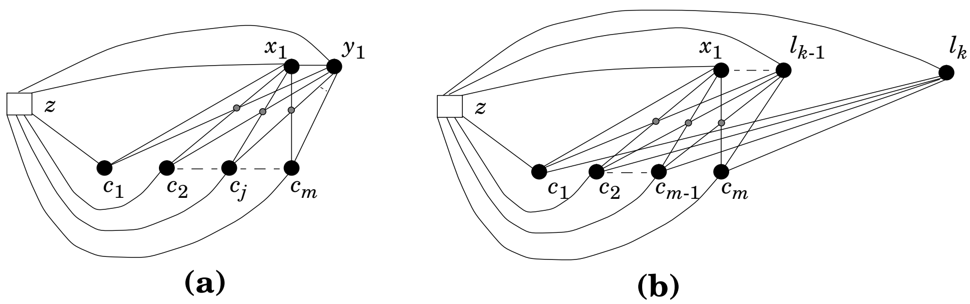

Figure 10: Proof of Lemma 7: **(a)** drawing $\psi(2, m)$; **(b)** drawing $\psi(k, m)$. The small circles show the lowest crossings of the clause edges in the drawing of $\psi(k-1, m)$.

Lemma 8 *Network $\mathcal{N}$ admits a feasible flow if and only if network $\mathcal{P}$ admits a feasible flow, and a feasible flow for $\mathcal{N}$ can be computed from a feasible flow for $\mathcal{P}$ in $O(n^2m^2)$ time.*

Proof: The *only if* part is trivial. Given a feasible flow in $\mathcal{N}$, we can get a feasible flow in $\mathcal{P}$ in which the flow through each fragment edge is same as in the corresponding clause edge in $\mathcal{N}$, and the flow through the dummy and literal edges is same as in the corresponding edges in $\mathcal{N}$.

For proving the *if* part suppose that $\mathcal{P}$ admits a feasible flow f. In f, at any crossing vertex, of the two fragment edges of a clause edge incident on it, one is incoming and the other is outgoing. This is so because the fragment edges of different clause edges have different capacities and the conservation property is satisfied at the vertex. Consequently all the fragment edges of a clause edge have the same flow. We can get a feasible flow in $\mathcal{N}$ in which the flow through a clause edge is same as the flow in its fragment edges in f, and the flow through the dummy edges and the literal edges is same as the flow in the corresponding edges in $\mathcal{P}$. □

By combining Lemmas 6, 7 and 8, we obtain the main result of this section.

Theorem 1 *Given an instance $\mathcal{S}$ of* NOT-ALL-EQUAL-3-SAT *with $n \geq 3$ variables and $m \geq 3$ clauses, the associated planar switch-flow network $\mathcal{P}$ is triconnected, has $O(n^2m^2)$ vertices and edges, has facial degree at most $7nm$, and can be constructed in $O(n^2m^2)$ time. Instance $\mathcal{S}$ is satisfiable if and only if network $\mathcal{P}$ admits a feasible flow. Also, given a feasible flow for $\mathcal{P}$, a satisfying truth assignment for $\mathcal{S}$ can be computed in time $O(n^2m^2)$.*

4 Upward Planarity Testing

In this section we show how to reduce the problem of computing a feasible flow in the planar switch-flow network associated with a NOT-ALL-EQUAL-3-SAT instance to the problem of testing the upward planarity of a suitable digraph.

Let $\mathcal{P}$ be the planar switch-flow network with parameter $\theta = 4$ associated with a NOT-ALL-EQUAL-3-SAT instance $\mathcal{S}$. Now, we construct on orientation $\vec{\mathcal{P}}$ of $\mathcal{P}$ as follows (see Fig. 11):

- Every literal edge (x_i, y_i) is oriented from x_i to y_i.

- Every fragment edge is oriented "away from" the clause vertex and "towards" the literal vertex.
- Every dummy edge incident on a literal vertex is oriented towards the dummy vertex, and every dummy edge incident on a clause vertex is oriented towards the clause vertex.

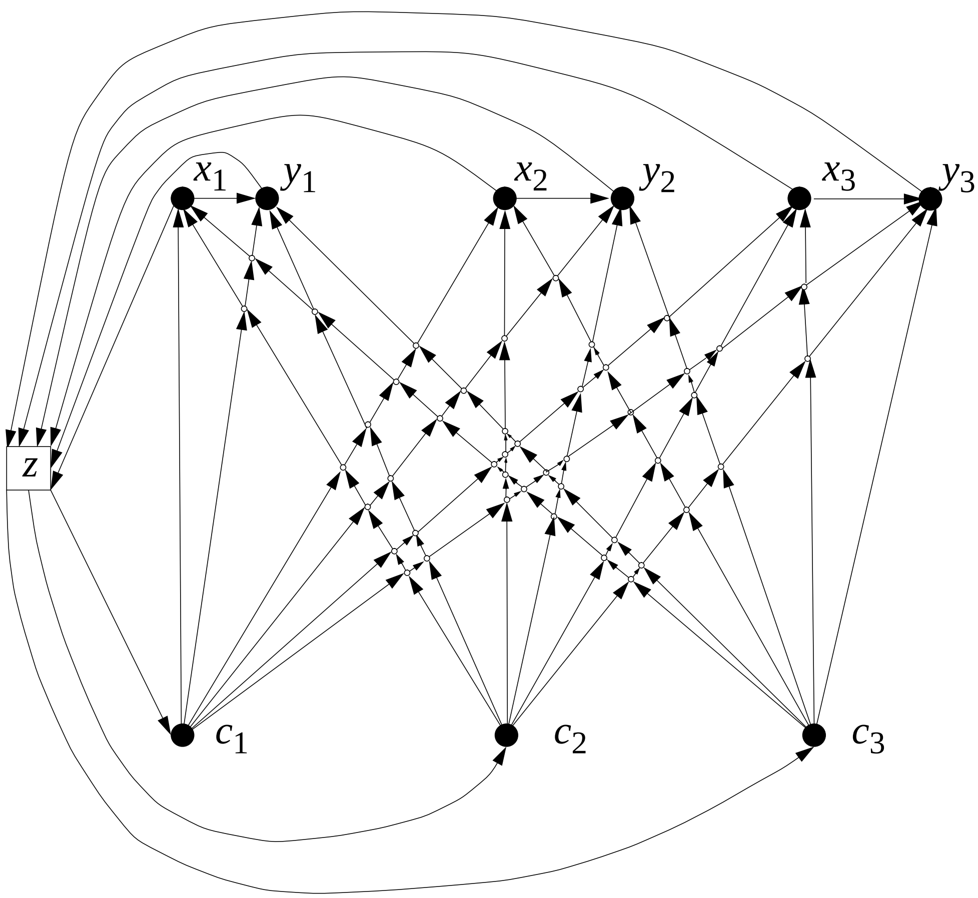

Figure 11: Orientation $\vec{\mathcal{P}}$ of the network $\mathcal{P}$ shown in Fig. 8(c).

Lemma 9 *In digraph $\vec{\mathcal{P}}$, every vertex has at least one incoming and one outgoing edge, every directed cycle contains the dummy vertex, and there are exactly two faces that are directed cycles. Also, $\vec{\mathcal{P}}$ is bimodal and each face of $\vec{\mathcal{P}}$ consists of at most two directed paths.*

Proof: Let m be the number of clauses and n be the number of variables in $\mathcal{S}$. Let z be the dummy vertex of $\vec{\mathcal{P}}$.

Each crossing vertex has two incoming and two outgoing fragment edges. Each literal vertex y_i has $m+1$ incoming edges and one outgoing edge (to z). Each literal vertex x_i has m incoming edges and two outgoing edges (to z and y_i). Each clause vertex has $2n$ outgoing edges and one incoming edge (from z). The dummy vertex has $2n$ incoming edges and m outgoing edges. Therefore, each vertex has at least one incoming and one outgoing edge. It can be easily verified that each vertex of $\vec{\mathcal{P}}$ has the bimodal property and hence $\vec{\mathcal{P}}$ is bimodal.

The digraph $\vec{\mathcal{P}} - \{z\}$ is upward planar with m sources, each being a clause vertex and n sinks, each being a literal vertex y_i. Therefore every directed cycle in $\vec{\mathcal{P}}$ contains z. There are exactly two faces in $\vec{\mathcal{P}}$ that that are directed cycles — one consisting of vertices x_1, z and c_1, and the other

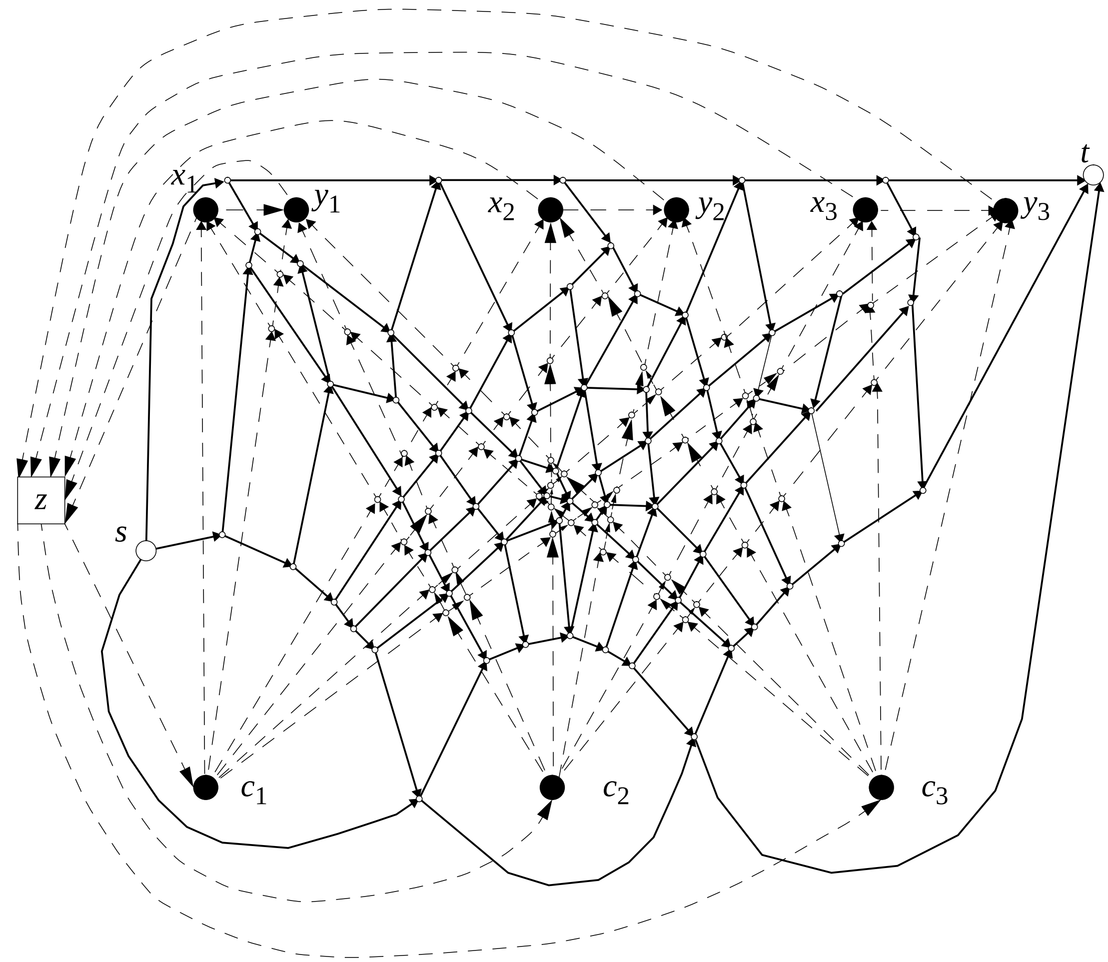

Figure 12: Dual digraph $\vec{\mathcal{D}}$ of network $\vec{\mathcal{P}}$ shown in Fig. 11.

consisting of vertices y_n, z and c_m. It can be easily verified that the other faces of $\vec{\mathcal{P}}$ each consist of exactly two directed paths. □

Since $\mathcal{P}$ is triconnected (see Theorem 1), the planar embedding of $\mathcal{P}$ and the dual graph of $\mathcal{P}$ are unique. We construct the dual digraph $\vec{\mathcal{D}}$ of $\vec{\mathcal{P}}$, by taking the dual graph $\mathcal{D}$ of $\mathcal{P}$ and orienting every dual edge from the face on the left to the face on the right of the primal edge (see Fig. 12).

Lemma 10 *The dual digraph $\vec{\mathcal{D}}$ of $\vec{\mathcal{P}}$ is upward planar, triconnected, acyclic and has exactly one source and one sink, denoted with s and t, both of which are on the same face. Also, each face of $\vec{\mathcal{D}}$ has exactly one source and one sink.*

Proof: Because $\vec{\mathcal{P}}$ is planar and triconnected (Theorem 1), so is its dual $\vec{\mathcal{D}}$. By Lemma 9, exactly two faces of $\vec{\mathcal{P}}$ are directed cycles. Hence $\vec{\mathcal{D}}$ has exactly two switches, denoted by s and t (see Fig. 12), corresponding to these two switches, s which is a source and t which is a sink. s corresponds to the face of $\vec{\mathcal{P}}$ consisting of the literal vertex x_1, dummy vertex z and the clause vertex c_1. t corresponds to the face of $\vec{\mathcal{P}}$ consisting of the vertices y_n, z and c_m. Both s and t are on the face that is the dual of z. From Lemma 9, each face of $\vec{\mathcal{P}}$ consists of at most two directed paths and hence each vertex of $\vec{\mathcal{D}}$ has the bimodal property. Therefore $\vec{\mathcal{D}}$ is bimodal. Again from Lemma 9, each vertex of $\vec{\mathcal{P}}$ has the bimodal property and none of them is a source or a sink. Hence each face of $\vec{\mathcal{D}}$ has exactly one source and one sink. Since s and t are the only switches of $\vec{\mathcal{D}}$ and both of them are on the same face, it follows that there is a consistent assignment of labels to the angles of $\vec{\mathcal{D}}$ in which exactly two angles are labeled large, namely the angles at s and t in their common face. Therefore, by Lemma 1, $\vec{\mathcal{D}}$ has an upward embedding and hence is upward planar. □

Starting from digraph $\vec{\mathcal{D}}$ we construct a new digraph $\vec{\mathcal{G}}$ by replacing the edges of $\vec{\mathcal{D}}$ with subgraphs (tendrils or wiggles), as follows (see Fig. 13):

- Every edge of $\vec{\mathcal{D}}$ that is the dual of a literal edge, fragment edge, or dummy edge incident on a literal vertex is replaced with tendril T_c, where $[c]$ is the capacity range of the dual edge. Note that c is a multiple of parameter θ.
- Every edge of $\vec{\mathcal{D}}$ that is the dual of a dummy edge incident on a clause vertex is replaced with wiggle W_c, where $[0 \cdots c]$ is the capacity range of the dual edge.

We distinguish two types of vertices in digraph $\vec{\mathcal{G}}$: we call *primary* vertices, the vertices that are also vertices of $\vec{\mathcal{D}}$, and *secondary* vertices the remaining vertices. There is no directed path from the sink the pole to the source pole of a tendril or a wiggle. Therefore $\vec{\mathcal{G}}$ is also acyclic. The following lemma is immediate.

Lemma 11 *Digraph $\vec{\mathcal{G}}$ is planar and acyclic. Also, the only primary source vertex is s and the only primary sink vertex is t. Both s and t are on the same face.*

By Lemma 10 and the construction of digraph $\vec{\mathcal{G}}$, all the embeddings of $\vec{\mathcal{G}}$ are obtained by choosing one of the two possible flips for each tendril. In an embedding of $\vec{\mathcal{G}}$, the face that corresponds to the dummy vertex of $\mathcal{P}$ is called the *dummy face* of the embedding. The other faces are called the *nondummy* faces.

Lemma 12 *Digraph $\vec{\mathcal{G}}$ is upward planar if and only if the tendrils can be flipped and the wiggles can be arranged such that for every nondummy face the total contribution of the tendrils and wiggles is zero.*

Proof:

$\Rightarrow$ By Lemma 10, $\mathcal{D}$ has an upward embedding. Let A be the assignment of labels to the angles in this embedding. A is a consistent assignment. Therefore if the tendrils can be flipped and the wiggles can be arranged to get an embedding ψ in which, for every face, the total contribution of the tendrils and wiggles is zero, we obtain from ψ an upward embedding of $\vec{\mathcal{G}}$ by assigning to the angles of the primary vertices of $\vec{\mathcal{G}}$ that are not angles of the internal faces of the tendrils to the same label as in assignment A.

$\Leftarrow$ Suppose $\vec{\mathcal{G}}$ has an upward embedding ψ. Let B be the assignment of labels to the angles of ψ. By Lemma 1, B is a consistent assignment, so that, denoting with $L(f)$ and $S(f)$, the number of angles of face f with label *large* and *small*, respectively, we have that $|L(f) - S(f)| = 2$ for every face f of ψ.

Let f be a nondummy face of ψ. By Lemma 10, f has exactly one primary source vertex and one primary sink vertex. Let us denote them by u and v. Let $\tau(f)$ be the contribution of the tendrils to f. Recall from Section 2.2, that the contribution of a tendril T_k is either $2k$ or $-2k$. Therefore, $\tau(f)$ is a multiple of $2\theta = 8$.

We have two cases:

Case 1: f corresponds to a literal or a crossing vertex of $\mathcal{P}$. Face f has no wiggles and exactly two angles between incoming or outgoing edges that are not angles of the tendrils of f. Thus, $L(f) - S(f) = \tau(f) + t$, with $|t| \leq 2$. By Lemma 1, $|L(f) - S(f)| = 2$. Hence, $|\tau(f) + t| = 2$, with $|t| \leq 2$. Since $\tau(f)$ is a multiple of 8, this implies $\tau(f) = 0$.

Case 2: f corresponds to a clause vertex of $\mathcal{P}$. Face f has three tendrils, one wiggle, and exactly two angles between incoming or outgoing edges that are not angles of the tendrils or wiggle

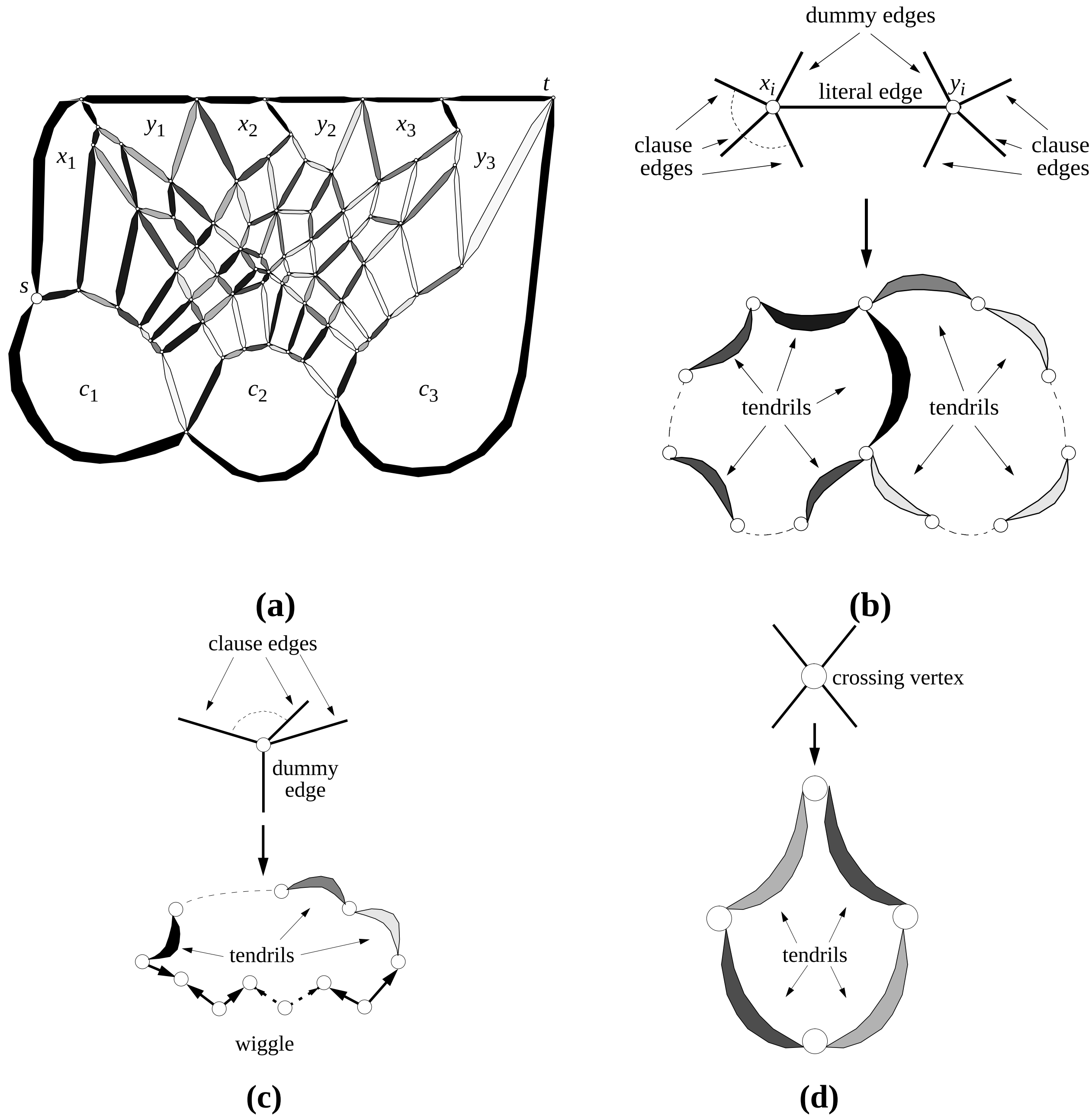

Figure 13: (**a**) Schematic illustration of digraph $\vec{\mathcal{G}}$ obtained from $\vec{\mathcal{D}}$ by replacing edges with tendrils and wiggles. (**b**) The two faces of $\vec{\mathcal{G}}$ associated with literal vertices x_i and y_i of $\mathcal{P}$. (**c**) The face of $\vec{\mathcal{G}}$ associated with a clause vertex of $\mathcal{P}$. (**d**) The face of $\vec{\mathcal{G}}$ associated with a crossing vertex of $\mathcal{P}$.

of f. Let $\omega(f)$ be the contribution of the wiggle. We have $L(f) - S(f) = \tau(f) + \omega(f) + t$, with $|t| \leq 2$. By Lemma 1, $|L(f) - S(f)| = 2$. Hence, $|\tau(f) + \omega(f) + t| = 2$, with $|t| \leq 2$. Let η_j be the sum of the absolute values of the contributions of the tendrils to f. By construction, we have $|\omega(f)| \leq \eta_j - 8$. We claim that $|\tau(f)| \leq \eta_j - 8$. Indeed, since $\tau(f)$ is a multiple of 8, $|\tau(f)| > \eta_j - 8$ implies $|\tau(f)| = \eta_j$ and thus $|\tau(f) + \omega(f)| \geq 8$, a contradiction. Since the wiggle of f can be rearranged so that $\omega(f)$ takes any even value between $-\eta_j + 8$ and $\eta_j - 8$, ψ can be modified so that $\tau(f) + \omega(f) = 0$ without affecting the other faces of the embedding.

□

We establish the following correspondence between digraph $\vec{\mathcal{G}}$ and network $\mathcal{P}$ (see Fig. 13):

- The faces of $\vec{\mathcal{G}}$ correspond to the vertices of $\mathcal{P}$.
- The tendrils and wiggles of $\vec{\mathcal{G}}$ correspond to the the edges of $\mathcal{P}$.
- Flipping a tendril T_k of $\vec{\mathcal{G}}$ corresponds to orienting an edge e of $\mathcal{P}$, where e is the dual of the edge replaced by T_k in constructing $\vec{\mathcal{G}}$ from $\vec{\mathcal{D}}$. Edge e is oriented towards the dual vertex of a face f of $\vec{\mathcal{G}}$ if and only if T_k contributes $2k$ to f.
- The contribution of a tendril or wiggle U of $\vec{\mathcal{G}}$ corresponds to the flow in an edge e of $\mathcal{P}$. e is the dual of the edge which is replaced by U in constructing $\vec{\mathcal{G}}$ from $\vec{\mathcal{D}}$. The contribution of U to a face f is equal to the amount of the flow coming into the dual vertex of f through edge e.
- The balance of the contribution of the tendrils and wiggles to the faces of $\vec{\mathcal{G}}$ corresponds to the conservation of flow at the vertices of $\mathcal{P}$, i.e., the total contribution of the tendrils and wiggles to a face f is zero if and only if there is a conservation of flow at the dual vertex of f.

Theorem 2 *Given an instance $\mathcal{S}$ of* NOT-ALL-EQUAL-3-SAT *with n variables and m clauses and the associated planar switch-flow network $\mathcal{P}$, digraph $\vec{\mathcal{G}}$ associated with $\mathcal{S}$ and $\mathcal{P}$ has $O(n^3m^2)$ vertices and edges, and can be constructed in $O(n^3m^2)$ time. Instance $\mathcal{S}$ is satisfiable and network $\mathcal{P}$ admits a feasible flow if and only if digraph $\vec{\mathcal{G}}$ is upward planar. Also, given an upward planar embedding for $\vec{\mathcal{G}}$, a feasible flow for $\mathcal{P}$ and a satisfying truth assignment for $\mathcal{S}$ can be computed in time $O(n^3m^2)$.*

Proof: The number of vertices and edges in a tendril or a wiggle is $O(n)$. Since $\mathcal{P}$ has $O(n^2m^2)$ vertices and edges, $\mathcal{S}$ has $O(n^3m^2)$ vertices and edges. $\mathcal{P}$ can be constructed from $\mathcal{S}$ in $O(n^2m^2)$ time and $\vec{\mathcal{G}}$ can be constructed from $\mathcal{P}$ in $O(n^3m^2)$ time giving a total time complexity of $O(n^3m^2)$ for the construction of $\vec{\mathcal{G}}$ from $\mathcal{S}$.

From Lemma 12 and the correspondence established above between a feasible flow in $\mathcal{P}$ and the upward planarity of $\vec{\mathcal{G}}$, we have that instance $\mathcal{S}$ is satisfiable and network $\mathcal{P}$ admits a feasible flow if and only if digraph $\vec{\mathcal{G}}$ is upward planar. We also obtain that given an upward planar embedding for $\vec{\mathcal{G}}$, a feasible flow for $\mathcal{P}$ and a satisfying truth assignment for $\mathcal{S}$ can be computed in time $O(n^3m^2)$.

□

From Theorems 1 and 2 we conclude:

Corollary 1 *Upward planarity testing is NP-complete.*

5 Rectilinear Planarity Testing

In this section we show how to reduce the problem of computing a feasible flow in the planar switch-flow network $\mathcal{P}$ associated with a NOT-ALL-EQUAL-3-SAT instance $\mathcal{S}$ to the problem of testing the rectilinear planarity of a suitable graph $\mathcal{G}$. The construction of graph $\mathcal{G}$ is carried out in several stages, where at each stage an intermediate graph is produced.

Given an instance $\mathcal{S}$ of NOT-ALL-EQUAL-3-SAT with n variables and m clauses, let $\mathcal{P}$ be the associated planar switch-flow network with parameter $\theta = 32nm$.

Let $\mathcal{D}$ be the dual graph of $\mathcal{P}$. The edges of $\mathcal{D}$ are classified as literal, clause, and dummy edges according to the type of their dual edge in $\mathcal{P}$. Starting from $\mathcal{D}$, we construct a planar graph $\mathcal{F}$ with vertices of maximum degree 3 by first replacing every vertex of degree d with a binary tree with d leaves, and then replacing every edge of the resulting graph with a chain of five edges and edges.

We classify the edges of $\mathcal{F}$ as expansion, literal, clause, and dummy edges, where the edges forming the binary trees replacing the former vertices of $\mathcal{D}$ are *expansion edges*, and the remaining edges of $\mathcal{F}$ are classified according to the type of the edge of $\mathcal{D}$ that originated them. Note that each edge e of $\mathcal{P}$ is thus associated with exactly five edges of $\mathcal{F}$ forming a path, and we call the middle edge the *representative* of e in $\mathcal{F}$.

Lemma 13 *Graph $\mathcal{F}$ has a unique planar embedding and admits a rectilinear embedding.*

Proof: Each embedding of $\mathcal{F}$ corresponds to exactly one embedding of $\mathcal{D}$, namely the embedding obtained by first replacing each chain of five edges by a single edge between the end points of the chain, and then replacing each binary tree induced by the expansion edges into a single vertex. Since $\mathcal{D}$ has a unique embedding, we have that $\mathcal{F}$ also has a unique embedding.

Every planar graph with vertices of degree at most 4 admits an orthogonal drawing with at most 4 bends per edge, which can be constructed in linear time (see, e.g., [31]). Hence, graph $\mathcal{F}$ admits a planar rectilinear drawing.. □

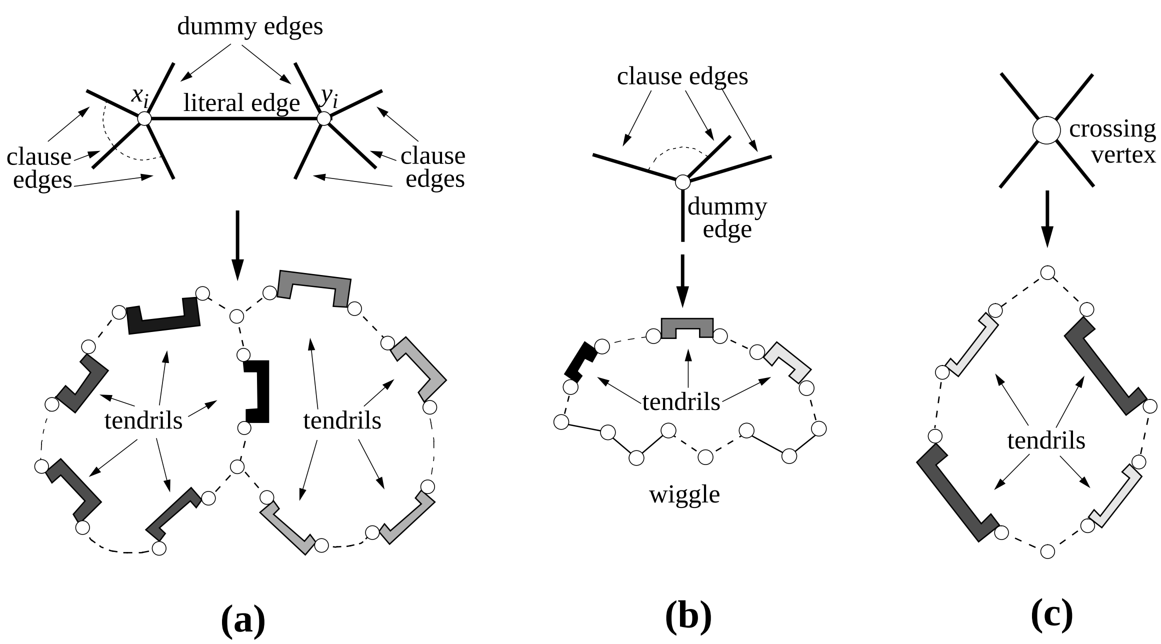

Figure 14: Schematic illustration of graph $\mathcal{G}$ obtained from $\mathcal{F}$ by replacing edges with rectilinear tendrils and wiggles: **(a)** the two faces of $\mathcal{G}$ associated with literal vertices x_i and y_i of $\mathcal{P}$. **(b)** the face of $\mathcal{G}$ associated with a clause vertex of $\mathcal{P}$. **(c)** the face of $\mathcal{G}$ associated with a crossing vertex of $\mathcal{P}$.

Finally, we construct graph $\mathcal{G}$ as follows (see Fig. 14). Let e be an edge of $\mathcal{P}$. If e is a dummy clause edge with capacity range $[0 \cdots c]$, we replace the representative of e in $\mathcal{F}$ with rectilinear wiggle W_c. Else, e is a literal, fragment, or dummy literal edge with capacity range $[c]$, and we replace the representative of e in $\mathcal{F}$ with rectilinear tendril T_c. The vertices of $\mathcal{G}$ that are also vertices of $\mathcal{F}$ are called its *primary* vertices.

By Lemma 13 and the construction of graph $\mathcal{G}$, all the embeddings of $\mathcal{G}$ are obtained by choosing one of the two possible flips for each rectilinear tendril. In an embedding of $\mathcal{G}$, the face that corresponds to the dummy vertex of $\mathcal{P}$ is called the *dummy face* of the embedding. The other faces are called the *nondummy* faces.

Recall that a rectilinear tendril T_k contributes one of $4k$, $4k+1$, $4k+2$, $-4k$, $-(4k+1)$, and $-(4k+2)$ to a face of $\mathcal{G}$. We say that the *significant* contribution of a rectilinear tendril T_k to a face is $4k$ if its contribution is positive and is $-4k$ if it is negative. The *significant* contribution of a rectilinear wiggle to a face is simply its contribution to the face. A face of an embedding of $\mathcal{G}$ is called *balanced* if the total significant contribution of its rectilinear tendrils and wiggles to it is zero. The *contribution* of primary vertices to a face f is equal to the number of angles at them in f labeled 3 minus the number of angles at them in f labeled 1.

Lemma 14 *If $n \geq 3$ and $m \geq 3$, then in any rectilinear embedding of $\vec{\mathcal{G}}$, the contribution of primary vertices to each face is at most $40nm$.*

Proof: Each face of $\mathcal{D}$ has $\max\{m+2, 2n+1, 4, m+2n\} \leq nm$ vertices for $n, m \geq 3$. The expansion of a vertex with degree d of $\mathcal{D}$ by a binary tree adds at most d vertices to an incident face. Hence, after we expand each vertex of $\mathcal{D}$ by a binary tree, we increase the number of vertices in a face by at most the facial degree of its dual vertex in $\mathcal{P}$, which is at most $7nm$ by Theorem 1. We further replace each edge by a chain of five edges resulting in a further increase of at most $4(7nm + nm) = 32nm$ vertices in each face. Hence each face of $\mathcal{F}$ has at most $32nm + 7nm + nm = 40nm$ vertices. Therefore, the contribution of the primary vertices to each face is at most $40nm$. □

Lemma 15 *Graph $\mathcal{G}$ admits a rectilinear planar drawing if and only if the rectilinear tendrils can be flipped and the rectilinear wiggles can be arranged such that every nondummy face is balanced.*

Proof:

$\Rightarrow$ By Lemma 10, $\mathcal{D}$ has a rectilinear embedding. Let A be the assignment of labels to the angles in this embedding. A is a consistent assignment. Therefore if the tendrils can be flipped and the wiggles can be arranged to get an embedding ψ in which, for every face, the total significant contribution of the tendrils and wiggles is zero, we obtain from ψ a rectilinear embedding of $\mathcal{G}$ by assigning to the angles of the primary vertices of $\mathcal{G}$ the same label as in assignment A, and making the contribution of each tendril to a face equal to its significant contribution.

$\Leftarrow$ Suppose $\mathcal{G}$ has a rectilinear embedding ψ. Let B be the assignment of labels to the angles of ψ. By Lemma 2, B is a consistent assignment, so that, denoting with $N_3(f)$ and $N_1(f)$, the number of angles of face f with label 3 and 1, respectively, we have that $|N_3(f) - N_1(f)| = 4$ for every face f of ψ.

Let f be a nondummy face of ψ. By Lemma 14, f has at most $40nm$ primary vertices. Let $\tau(f)$ be the total contribution of the tendrils to f. Let $\sigma(f)$ be the total significant contributions of the tendrils to f. Recall from Section 2.2, that the contribution of a tendril T_k is one of $4k$, $4k+1$, $4k+2$, $-4k$, $-4k-1$, and $-4k-2$. Therefore, $\sigma(f)$ is a multiple of $4\theta = 128nm$. Also, because each face has at most nm tendrils, $|\sigma(f) - \tau(f)| \leq 2nm$. We have two cases:

Case 1: f corresponds to a literal or a crossing vertex of $\mathcal{P}$. Face f has no wiggles. $N_3(f) - N_1(f) = \tau(f) + t$, with $|t| \leq 40nm$. By Lemma 2, $|N_3(f) - N_1(f)| = 4$. Hence, $|\tau(f) + t| = 4$, with $|t| \leq 40nm$. Consequently, $|\sigma(f) + t| \leq 4 + 2nm$. Since $\sigma(f)$ is a multiple of $128nm$, this implies that $\sigma(f) = 0$.

Case 2: f corresponds to a clause vertex of $\mathcal{P}$. Face f has three tendrils and one wiggle. Let $\omega(f)$ be the contribution of the wiggle. We have $N_3(f) - N_1(f) = \tau(f) + \omega(f) + t$, with $|t| \leq 40nm$. By Lemma 2, $|N_3(f) - N_1(f)| = 4$. Hence, $|\tau(f) + \omega(f) + t| = 4$, with $|t| \leq 40nm$. Since $|\sigma(f) - \tau(f)| \leq 2nm$, we have $|\sigma(f) + \omega(f) + t| \leq 4 + 2nm$.

Let η_j be the sum of the absolute values of the significant contributions of the tendrils to f. By construction, we have $|\omega(f)| \leq \eta_j - 64nm$. We claim that $|\sigma(f)| \leq \eta_j - 128nm$. Indeed, since $\sigma(f)$ is a multiple of $128nm$, $|\sigma(f)| > \eta_j - 128nm$ implies $|\sigma(f)| = \eta_j$ and thus $|\sigma(f) + \omega(f)| \geq 64nm > 4 + 2nm + 40nm$, a contradiction.

Since the wiggle of f can be rearranged so that $\omega(f)$ takes any value between $-\eta_j + 64nm$ and $\eta_j - 64nm$, ψ can be modified so that $\sigma(f) + \omega(f) = 0$ without affecting the other faces of the embedding.

□

We establish the following correspondence between graph $\mathcal{G}$ and network $\mathcal{P}$ (see Fig. 14):

- The faces of $\mathcal{G}$ correspond to the vertices of $\mathcal{P}$.
- The rectilinear tendrils and wiggles of $\mathcal{G}$ correspond to the the edges of $\mathcal{P}$.
- Flipping a rectilinear tendril T_k of $\mathcal{G}$ corresponds to orienting an edge e of $\mathcal{P}$. e is the dual of the edge which is replaced by T_k in constructing $\mathcal{G}$ from $\mathcal{F}$. Edge e is oriented towards the dual vertex of a face f if and only if the significant contribution of T_k to f is $4k$.
- The significant contribution of a rectilinear tendril or wiggle U of $\mathcal{G}$ corresponds to the flow in an edge e of $\mathcal{P}$, where e is the dual of the edge replaced by U in constructing $\mathcal{G}$ from $\mathcal{F}$. The significant contribution of U to a face f is equal to the amount of the flow coming into the dual vertex of f through edge e.
- The sum of the significant contributions of the rectilinear tendrils and wiggles to the faces of $\mathcal{G}$ corresponds to the conservation of flow at the vertices of $\mathcal{P}$, i.e., the total significant contribution of the rectilinear tendrils and wiggles to a face f is zero if and only if there is a conservation of flow at the dual vertex of f.

Theorem 3 *Given an instance $\mathcal{S}$ of* NOT-ALL-EQUAL-3-SAT *with n variables and m clauses, graph $\mathcal{G}$ associated with $\mathcal{S}$ has $O(n^4m^3)$ vertices and edges, and can be constructed in $O(n^4m^3)$ time. Instance $\mathcal{S}$ is satisfiable if and only if graph $\mathcal{G}$ is rectilinear planar. Also, given a rectilinear planar embedding for $\mathcal{G}$, a satisfying truth assignment for $\mathcal{S}$ can be computed in time $O(n^4m^3)$.*

Proof: From Theorem 1 and the fact that each rectilinear tendril and wiggle has $O(n^2m)$ vertices and edges, it follows that $\mathcal{G}$ has $O(n^4m^3)$ vertices and edges, and can be constructed in $O(n^4m^3)$ time.

From Lemma 15 and the correspondence established above between graph G and network P it follows that instance $\mathcal{S}$ is satisfiable if and only if graph $\mathcal{G}$ is rectilinear planar. Also, given a rectilinear planar embedding for $\mathcal{G}$, a satisfying truth assignment for $\mathcal{S}$ can be computed in time $O(n^4m^3)$. □

From Theorem 3 we conclude:

Corollary 2 *Rectilinear planarity testing is NP-complete.*

Corollary 3 *Computing a planar orthogonal drawing with the minimum number of bends is NP-hard.*

We can strengthen Corollary 3 as follows:

Corollary 4 *Let G be an n-vertex planar graph whose minimum number of bends in any planar orthogonal drawing is b^*. Computing a planar orthogonal drawing of G with $O(b^* + n^{1-\epsilon})$ bends is NP-hard for $\epsilon > 0$.*

Proof: Suppose there is a polynomial-time algorithm A that computes a planar orthogonal drawing of G with at most $c(b^* + n^{1-\epsilon})$ bends, where c is some constant. We can then use algorithm A to test in polynomial time whether graph G is rectilinear planar as follows: Construct a graph G' consisting of $K = \left\lceil (cn^{1-\epsilon})^{1/\epsilon} \right\rceil + 1$ copies of G and give G' as an input to algorithm A. Clearly, G is rectilinear planar if and only if G' has a planar orthogonal drawing with less than K bends. Since G' has Kn vertices and $K > c(Kn)^{1-\epsilon}$, algorithm A computes a drawing of G' with less than K bends if and only if G is rectilinear planar. □

6 Conclusions

Finding efficient algorithms for upward and rectilinear planarity testing had been open problems for many years. In this paper we have shown that a polynomial time algorithm for either of these problems is unlikely to exist by proving that both problems are NP-complete. NP-completeness of rectilinear planarity testing also implies that the bend-minimization problem for planar orthogonal drawings is NP-hard.

Acknowledgements

We would like to thank Giuseppe Di Battista for useful discussions, and Ivan Rival for comments on a preliminary version of this paper.

References

[1] P. Bertolazzi, R. F. Cohen, G. Di Battista, R. Tamassia, and I. G. Tollis. How to draw a series-parallel digraph. In *Proc. 3rd Scand. Workshop Algorithm Theory*, volume 621 of *Lecture Notes in Computer Science*, pages 272–283. Springer-Verlag, 1992.

[2] P. Bertolazzi, R. F. Cohen, G. Di Battista, R. Tamassia, and I. G. Tollis. How to draw a series-parallel digraph. *Internat. J. Comput. Geom. Appl.*, to appear.

[3] P. Bertolazzi and G. Di Battista. On upward drawing testing of triconnected digraphs. In *Proc. 7th Annu. ACM Sympos. Comput. Geom.*, pages 272–280, 1991.

[4] P. Bertolazzi, G. Di Battista, G. Liotta, and C. Mannino. Upward drawings of triconnected digraphs. *Algorithmica*, to appear.

[5] P. Bertolazzi, G. Di Battista, C. Mannino, and R. Tamassia. Optimal upward planarity testing of single-source digraphs. In *1st Annual European Symposium on Algorithms (ESA '93)*, volume 726 of *Lecture Notes in Computer Science*, pages 37–48. Springer-Verlag, 1993.

[6] T. Biedl and G. Kant. A better heuristic for orthogonal graph drawings. In *2nd Annu. European Symp. on Algorithms (ESA '94)*, volume 855 of *Lecture Notes in Computer Science*, pages 24–35. Springer-Verlag, 1994.

[7] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. Algorithms for drawing graphs: an annotated bibliography. Preprint, Dept. Comput. Sci., Brown Univ., Providence, RI, November 1993. To appear in *Comput. Geom. Theory Appl.* Preliminary version available via anonymous `ftp` from `wilma.cs.brown.edu`, `gdbiblio.tex.Z` and `gdbiblio.ps.Z` in `/pub/papers/compgeo`.

[8] G. Di Battista, G. Liotta, and F. Vargiu. Spirality of orthogonal representations and optimal drawings of series-parallel graphs and 3-planar graphs. In *Proc. Workshop Algorithms Data Struct.*, volume 709 of *Lecture Notes in Computer Science*, pages 151–162. Springer-Verlag, 1993.

[9] G. Di Battista, W. P. Liu, and I. Rival. Bipartite graphs upward drawings and planarity. *Inform. Process. Lett.*, 36:317–322, 1990.

[10] G. Di Battista and R. Tamassia. Algorithms for plane representations of acyclic digraphs. *Theoret. Comput. Sci.*, 61:175–198, 1988.

[11] G. Di Battista, R. Tamassia, and I. G. Tollis. Area requirement and symmetry display of planar upward drawings. *Discrete Comput. Geom.*, 7:381–401, 1992.

[12] S. Even and G. Granot. Rectilinear planar drawings with few bends in each edge. Technical Report 797, Computer Science Dept., Technion", 1994.

[13] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, NY, 1979.

[14] M. D. Hutton and A. Lubiw. Upward planar drawing of single source acyclic digraphs. In *Proc. 2nd ACM-SIAM Sympos. Discrete Algorithms*, pages 203–211, 1991.

[15] G. Kant. Drawing planar graphs using the *lmc*-ordering. In *Proc. 33th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 101–110, 1992.

[16] D. Kelly. Fundamentals of planar ordered sets. *Discrete Math.*, 63:197–216, 1987.

[17] D. Kelly and I. Rival. Planar lattices. *Canad. J. Math.*, 27(3):636–665, 1975.

[18] A. Lempel, S. Even, and I. Cederbaum. An algorithm for planarity testing of graphs. In *Theory of Graphs: Internat. Symposium (Rome 1966)*, pages 215–232, New York, 1967. Gordon and

Breach.

[19] Y. Liu, P. Marchioro, and R. Petreschi. A single bend embedding algorithm for cubic graphs. Manuscript, 1994.

[20] Y. Liu, P. Marchioro, R. Petreschi, and B. Simeone. Theoretical results on at most 1-bend embeddability of graphs. Technical report, Dipartimento di Statistica, Univ. di Roma "La Sapienza", 1990.

[21] Y. Liu, A. Morgana, and B. Simeone. General theoretical results on rectilinear embeddability of graphs. *Acta Math. Appl. Sinica*, 7:187–192, 1991.

[22] Y. Liu, A. Morgana, and B. Simeone. A linear algorithm for 3-bend embeddings of planar graphs in the grid. Manuscript, 1993.

[23] A. Papakostas. Upward planarity testing of outerplanar dags. In *Proc. Graph Drawing '94*, 1994.

[24] C. Platt. Planar lattices and planar graphs. *J. Combin. Theory Ser. B*, 21:30–39, 1976.

[25] I. Rival. The diagram. In I. Rival, editor, *Graphs and Orders*, pages 103–133. Reidel Publishing, 1985.

[26] I. Rival. Graphical data structures for ordered sets. In I. Rival, editor, *Algorithms and Order*, pages 3–31. Kluwer Academic Publishers, 1989.

[27] I. Rival. Reading, drawing, and order. In I. G. Rosenberg and G. Sabidussi, editors, *Algebras and Orders*, pages 359–404. Kluwer Academic Publishers, 1993.

[28] Y. Shiloach. *Arrangements of Planar Graphs on the Planar Lattice.* PhD thesis, Weizmann Institute of Science, 1976.

[29] J. A. Storer. On minimal node-cost planar embeddings. *Networks*, 14:181–212, 1984.

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

[31] R. Tamassia and I. G. Tollis. Planar grid embedding in linear time. *IEEE Trans. on Circuits and Systems*, CAS-36(9):1230–1234, 1989.

[32] R. Tamassia and J. S. Vitter. Parallel transitive closure and point location in planar structures. *SIAM J. Comput.*, 20(4):708–725, 1991.

[33] C. Thomassen. Planar acyclic oriented graphs. *Order*, 5(4):349–361, 1989.

[34] L. Valiant. Universality considerations in VLSI circuits. *IEEE Trans. Comput.*, C-30(2):135–140, 1981.

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