Minimum-Width Grid Drawings of Plane Graphs

Marek Chrobak *　　Shin-ichi Nakano †

UCR-CS-94-5

*Department of Computer Science, University of California, Riverside, CA 92521. Research partially supported by NSF grant CCR-9112067.

†Department of System Information Sciences, Tohoku University, Sendai 980-77, Japan. This work was done while the author was visiting University of California, Riverside.

Abstract

Given a plane graph G, we wish to draw it in the plane, according to the given embedding, in such a way that the vertices of G are drawn as grid points, and the edges are drawn as straight-line segments between their endpoints. An additional objective is to minimize the size of the resulting grid. It is known that each plane graph can be drawn in such a way in a $(n-2) \times (n-2)$ grid (for $n \geq 3$), and that no grid smaller than $(2n/3-1) \times (2n/3-1)$ can be used for this purpose, if n is a multiple of 3. In fact, it can be shown that, for all $n \geq 3$, each dimension of the resulting grid needs to be at least $\lfloor 2(n-1)/3 \rfloor$, even if the other one is allowed to be infinite. In this paper we show that this bound is tight, by presenting a grid drawing algorithm that produces drawings of width $\lfloor 2(n-1)/3 \rfloor$. The height of the produced drawings is bounded by $4\lfloor 2(n-1)/3 \rfloor - 1$.

Contents

1 Introduction

The problem of automatic graph drawing has attracted recently a lot of attention, due to its numerous practical applications and challenging mathematical and algorithmic questions that arise in this area. Generally, given a graph G, the task is to produce an *æsthetic* drawing of G, one that accurately reflects the topological structure of G in a graphical form. Many versions of this problem have been considered, and there is a variety of techniques, algorithms, and software packages that are currently available. (See the survey in [BETT] for more information.)

For planar graphs, this problem is especially interesting. In this case, we typically require that vertices are represented by points in the plane, and edges are drawn as non-intersecting straight-line segments between their endpoints. Additionally, we are often given a *plane graph*, that is a planar graph with a given planar embedding, represented combinatorially by cyclic orderings of edges incident to all vertices. Then the drawing needs to be consistent with that given planar embedding, in the sense that for each vertex v, the given cyclic ordering of edges incident to v needs to be the same as their clockwise ordering in the drawing.

In this paper we deal with the following version: given a plane graph G, we want to map its vertices into integer grid points in such a way that the edges between them can be drawn as straight, non-intersecting line segments. The resulting drawing has to be consistent with the planar embedding of G. We call such mappings *grid drawings.*

Restricting vertex coordinates to integer values has been motivated by the fact that using arbitrary real values leads to problems with rounding errors. Also, integer vertex coordinates facilitate the display of the drawing on raster graphics devices.

It has been proven that each plane graph has a straight-line drawing [Fa48, Wa36, St51], which implies that it also has a grid drawing, since we can approximate real vertex coordinates by rational numbers, and then use appropriate scaling. The grids obtained by following this method are, unfortunately, of exponential size.

The question whether smaller, polynomial-size, grids can be used for this purpose was open until 1988, when de Fraisseix, Pach and Pollack [FPP88, FPP90] proved that each plane graph with n vertices can be embedded into a $(2n-4) \times (n-2)$ grid. (Throughout the paper we assume that $n \geq 3$.) We will refer to their method as the *shift method.* Their paper initiated intensive research in this area, and led to new results and implementations. Chrobak and Payne [CP89] gave a simple, linear-time implementation of the shift method. Schnyder [Sc90] presented a different technique, based on so called *barycentric representations*, that led to smaller grid drawings of size $(n-2) \times (n-2)$. He also pointed out later (personal communication) that there is a close relationship between the shift method and barycentric representations, and that the grid size in [CP89] can be reduced to $(n-2) \times (n-2)$ without affecting time complexity.

In the work mentioned above, it is usually assumed that the given plane graph is triangulated.

Otherwise, one can always triangulate a given graph and remove the added edges after constructing the drawing. This approach, however, leads to poor quality drawings, so the question was raised whether æsthetic drawings can be constructed efficiently without a prior triangulation. For non-triangulated plane graph, one criterion of æstheticity is that the faces should be represented by convex polygons. This can be always achieved when the graph is 3-connected [Tu63]. G. Kant, working in this direction, proved that each 3-connected plane graph has a convex drawing in a $(2n-4) \times (n-2)$ grid, and the grid size was recently improved to $(n-2) \times (n-2)$ by Schnyder and Trotter [ST92] and Chrobak and Kant [CK93], independently. All these algorithms can be implemented in linear time.

The obvious question is: what is the minimum size of grid drawings? In their paper, [FPP88] proved that, in the worst case, no grid smaller than $(2n/3-1) \times (2n/3-1)$ is possible for n-vertex plane graphs, if n is a multiple of 3. The simple argument they presented can be easily modifed to show that, for all $n \geq 3$, each dimension of the grid needs to be at least $\lfloor 2(n-1)/3 \rfloor$, even if the other one is unbounded.

In this paper we show that this bound is tight, by presenting an algorithm that embeds each n-vertex plane graph into a grid of width at most $\lfloor 2(n-1)/3 \rfloor$. The height of the resulting drawings is at most $4\lfloor 2(n-1)/3 \rfloor - 1 \leq 8n/3 - 3$.

It is important to note that the distinction between planar and plane graphs is essential. If we are given a *planar* graph G on input, and we are allowed to choose its planar embedding, then the known $\lfloor 2(n-1)/3 \rfloor$ lower bound proof does not apply. We discuss this issue in Section 7.

In Section 2 and Section 3 we introduce our notation and terminology. In Section 4 we present a generic shift method for grid drawings, of which algorithms from [FPP88, CP89] and the restriction of [CK93] to maximal plane graphs are special instances.

Our algorithm is based on the shift method as well. In Section 5, for the sake of presentation, we introduce a simplified version of our algorithm, called Algorithm $\mathcal{A}$, that illustrates the main idea for reducing grid width. Algorithm $\mathcal{A}$ produces drawings of width $\lfloor 2(n-1)/3 \rfloor$, but the height can be quadratic (it is bounded by $n^2/4$). Later, in Section 6, we present Algorithm $\mathcal{B}$ that uses the same width but reduces the height to $4\lfloor 2(n-1)/3 \rfloor - 1$. In Section 7 we discuss possible modifications of Algorithm $\mathcal{B}$ (including one improving the grid height to $4\lfloor 2(n-1)/3 \rfloor - 4$).

2 Preliminaries

Let $G = (V, E)$ be an arbitrary maximal (triangulated) plane graph with n vertices, where $n \geq 3$, and $\pi = v_1, \ldots, v_n$ an ordering of V such that (v_1, v_2, v_n) is the external face of G. Define G_k to be the subgraph of G induced by $v_1, \ldots, v_k$ and C_k to be its external face. We say that π is a *canonical ordering of* G if the following conditions are satisfied for each $k = 3, \ldots, n$:

(co1) Each G_k is 2-connected and internally triangulated (that is, all internal faces of G_k are triangles).

(co2) C_k contains (v_1, v_2).

(co3) If $k < n$, then v_{k+1} is in the exterior face of G_k, and all neighbors of v_{k+1} in G_k belong to C_k.

It is easy to see that Conditions (co1) and (co3) imply that the neighbors of v_{k+1} must, in fact, be consecutive in C_k. The existence of canonical orderings was proven by de Fraisseix, Pach and Pollack in [FPP88] (See also [Ka93]).

Lemma 1 *Let G be a maximal plane graph, and (v_1, v_2) an edge on its external face. Then there exists a canonical ordering $\pi = v_1, v_2, \ldots, v_n$ of G, and π can be constructed in linear time.*

We will use symbol $\prec$ to denote the linear order given by the canonical ordering, that is, if $i < j$, then we will write $v_i \prec v_j$.

In Fig. 1 an example of a canonical ordering is given. Canonical orderings (and its extensions to 3-connected graphs) were used in [FPP88, CP89, Ka92, CK93] for graph drawing algorithms.

By an *ordered plane* graph (G, π) we will understand a plane graph G with a given canonical ordering $\pi = v_1, \ldots, v_n$. By the *contour of G_k* we mean its external face written as

$$C_k = (w_1 = v_1, w_2, \ldots, w_m = v_2).$$

Let $k \geq 3$, and let the neighbors of $v = v_{k+1}$ in G_k be $w_p, w_{p+1}, \ldots, w_q$. The *in-degree of* v, denoted $deg^-(v)$, is the number of neighbors of v in G_k, that is $deg^-(v) = q - p + 1$. For $i = p, \ldots, q$, we denote $ind_v(w_i) = i - p + 1$ and call it the *index of w_i with respect to v*. For $k = 1, 2, 3$, $deg^-(v_k)$ and ind_{v_k} are undefined.

By a and b we will usually denote, respectively, the number of vertices of in-degree 2, and the number of vertices of in-degree ≥ 3, among $v_4, \ldots, v_n$. If $n \geq 4$, clearly, $deg^-(v_k) \geq 2$ for each $k = 4, \ldots, n-1$, $deg^-(v_n) \geq 3$, and $n = a + b + 3$.

3 Grid Drawings

Let N denote the set of non-negative integers, and G a given plane graph with vertex set V. Let $P = (x, y) : V \to N \times N$ be a function that maps V into an integer grid, where $x(v)$ and $y(v)$ represent the x and y coordinates of a vertex v.

Given two points A, B in the plane, by $[A, B]$ we denote the closed line segment joining A and B. We will say that edges (u, v), (s, t) *intersect* (or *cross*) in P if

$$[P(u), P(v)] \cap [P(s), P(t)] \neq \{u, v\} \cap \{s, t\}.$$

In other words, segments $[P(u), P(v)]$, $[P(s), P(t)]$ share points other than common endpoints of edges (u, v), (s, t). We will omit the phrase "in P", when P is understood from context.

Given a mapping $P = (x, y)$, we say that P is a *grid drawing* of G, if it satisfies the following conditions:

(gd1) If $u \neq v$, then $P(u) \neq P(v)$.

(gd2) No two edges of G intersect in P.

(gd3) For each vertex v, the clockwise ordering of the segments $[P(v), P(u)]$, where u is a neighbor of v, is identical to their cyclic ordering in the given planar embedding of G.

The *width* of a given drawing is defined as the distance between its leftmost and rightmost vertices, that is $\max_{u,v} |x(u) - x(v)|$. The *height* is defined similarly: $\max_{u,v} |y(u) - y(v)|$.

By a minor modification of the construction in [FPP88], we obtain the following theorem.

Theorem 1 *For each $n \geq 3$ there is an n-vertex plane graph H_n, such that each grid drawing of H_n has width at least $\lfloor 2(n-1)/3 \rfloor$.*

Proof: H_3 is the triangle (v_1, v_2, v_3) and, for $n \geq 4$, H_n is obtained by adding vertex v_n to the outer face of H_{n-1} and connecting it to $v_{n-3}, v_{n-2}, v_{n-1}$ in such a way that the outer face of H_n is (v_n, v_{n-1}, v_{n-2}).

First, notice that for $n = 3, 4, 5$, H_n requires width $1, 2, 2$, respectively, which equals $\lfloor 2(n-1)/3 \rfloor$. The theorem follows by induction, since adding v_{n+1}, v_{n+2} and v_{n+3} to H_n forces us to use at least two more x-coordinates. $\square$

4 The Shift Method for Grid Drawings

In this method, as well as in Algorithms $\mathcal{A}$ and $\mathcal{B}$ we will assume that the plane graph given on input is maximal (triangulated). If the input graph is not maximal, it can be easily triangulated in linear time (see, for example, [Ka93]), and after the grid drawing is found, the added edges can be removed.

Let (G, π) be a given ordered maximal plane graph, where $\pi = v_1, \ldots, v_n$ and $n \geq 3$. Our general strategy is similar to the methods from [FPP88, CP89]: we add vertices one at a time, in canonical order. At every time step, the contour C_k satisfies a certain invariant that involves restrictions on the slopes of contour edges. When adding a vertex v_{k+1} we determine its location in the grid and, if necessary, shift some parts of G_k to the right in order to preserve the invariant. The difficult part is to determine which internal vertices of G_k can be shifted to the right without violating planarity. We will describe such a method in this section.

The U-sets. We will maintain a set $U(v)$ for each vertex v. This set will contain vertices located "under" v that need to be shifted whenever v is shifted. Initially, $U(v_i) = \{v_i\}$ for $i = 1, 2, 3$. Suppose that $3 \leq k \leq n-1$ and that we are about to add v_{k+1} to G_k. Let the contour of G_k be

$$C_k = (w_1 = v_1, w_2, \ldots, w_m = v_2)$$

and that $w_p, \ldots, w_q$ are the neighbors of v_{k+1} in G_k. Then we set

$$U(v_{k+1}) := \{v_{k+1}\} \cup \bigcup_{i=p+1}^{q-1} U(w_i).$$

The shift operation. Shifting a contour vertex w_j is achieved by operation *shift*(w_j), that increases the x-coordinate of each $u \in \bigcup_{i=j}^{m} U(w_i)$ by 1.

In the original shift method [FPP88], all slopes in the contour were either 1 or -1. Preserving this invariant required shifting w_{p+1} and w_q at each step, resulting in the drawing of width $2 + 2(n-3) = 2n-4$ and height $n-2$. This can be improved by using arbitrary non-negative upward slopes, but restricting downward slopes to -1 (see [CK93]). In this method, each step involves one shift only. This leads to drawings of width $2 + (n-3) = n-1$ and height $n-1$, which can be improved to $(n-2) \times (n-2)$ by handling the last vertex in a special manner. Our method will in fact avoid making any shifts in at least $n/3 + O(1)$ steps.

Generic Shift Algorithm: Initially, v_1, v_2, v_3 are mapped into different grid points so that $x(v_2) > x(v_1) \geq 0$, and v_3 is located at a point above the line segment joining v_1, v_2 and satisfying $x(v_1) \leq x(v_3) \leq x(v_2)$.

Inductively, suppose that $3 \leq k \leq n-1$, that G_k has already been embedded, and that we are about to add $v = v_{k+1}$. Let $C_k = (w_1, \ldots, w_m)$ be the contour of G_k, and $w_p, \ldots, w_q$ be the neighbors of v in G_k. Apply *shift*(w_i) to some of $w_1, \ldots, w_m$ (possibly none), so that afterwards there exists at least one point (x', y') such that

(gsm1) $x(w_p) \leq x' \leq x(w_q)$,

(gsm2) (x', y') is located *above* C_k in the following sense: (a) the half line $\{(x', z) : z \geq y'\}$ does not intersect C_k, and (b) $y' \geq y(w_{p+1}), y(w_{q-1})$,

(gsm3) all vertices $w_p, \ldots, w_q$ are visible from (x', y').

Pick an arbitrary such point (x', y') and set $(x, y)(v) = (x', y')$.

In (gsm3), the term "visible" means that the edges from v to all $w_p, \ldots, w_q$ do not intersect each other, nor the edges in C_k.

Lemma 2 *For all choices of shift operations and vertex coordinates, as long as (gsm1), (gsm2) and (gsm3) are satisfied, the Generic Shift Algorithm produces a correct grid drawing.*

Proof: All vertices are mapped into grid points, so we only need to show that the edges will not cross. The proof is by induction.

The inductive claim is as follows: Let $3 \le k \le n$, and $C_k = (w_1, \ldots, w_m)$. Then

(i) G_k is correctly embedded,

(ii) $x(w_1) \le x(w_2) \le \ldots \le x(w_m)$, and

(iii) executing an arbitrary number of operations *shift*(w_j) does not introduce edge crossings.

First, note that G_3 satisfies (i)–(iii). For the inductive step, we need to show that adding $v = v_{k+1}$ to G_k preserves conditions (i)–(iii). When we install v, we may execute a number of operations *shift*(w_j). By the inductive assumption (iii), the correctness of the drawing of G_k is preserved, and now (i) follows from (gsm3). The new contour is $C_{k+1} = (w_1, \ldots, w_p, v, w_q, \ldots, w_m)$. Property (ii) is invariant under shifting, and therefore the inductive assumption (ii) and the choice of $x(v)$ in (gsm1) imply that (ii) is preserved as well.

It remains to show that (iii) holds after adding v. Recall that the new contour is $C_{k+1} = (w_1, \ldots, w_p, v, w_q, \ldots, w_m)$. By the definition of the U-sets, *shift*(w_j) in G_{k+1}, for $j > q$, is equivalent to *shift*(w_j) in G_k, since only sets $U(w_i)$, for $i \ge j$ are shifted; vertex v and the edges incident to v are not affected.

Executing *shift*(w_q) in G_{k+1} is also equivalent to *shift*(w_q) in G_k, but in G_{k+1} it also stretches the edge (v, w_q). No edge crossings are introduced in G_k, by induction. In the triangle (w_{q-1}, v, w_q) we have $x(v), x(w_{q-1}) \le x(w_q)$ and $y(w_{q-1}) \le y(v)$, so moving w_q to the right does not introduce edge intersections.

Executing *shift*(v) is equivalent to *shift*(w_{p+1}) in G_k and increasing $x(v)$ by 1. From the inductive assumption (iii), no edge crossings are introduced in G_k. But v moves rigidly with all its neighbors except w_p, and thus no edge crossings can be introduced between vertices v, $w_{p+1}, \ldots, w_q$. By (gsm1), we have $x(w_p) \le x(v)$. By (gsm1) and (gsm3), $x(w_p) < x(w_{p+1})$, since otherwise w_p or w_{p+1} would not be visible from v. By (gsm2) we also have $y(v) \ge y(w_{p+1})$. Then, since the edge (v, w_{p+1}) moves rigidly to the right, no edge intersections will be introduced in the triangle (w_p, w_{p+1}, v).

Finally, if we do *shift*(w_j) for $j \le p$, the inductive assumption (iii) and the fact that v moves rigidly with all $w_p, \ldots, w_q$, imply that no edge intersections can be introduced. □

5 Minimum-Width Grid Drawings

Let (G, π) be a given ordered, maximal plane graph, where $\pi = v_1, \ldots, v_n$. For a given $3 \le k \le n-1$, let $w_p, \ldots, w_q$ be the neighbors of $v = v_{k+1}$ in C_k. When we add v to G_k, its leftmost and rightmost edges (w_p, v), (v, w_q) become contour edges. We call (w_p, v) a *forward edge* and (v, w_q) a *backward edge*. All vertices and edges that disappear from the contour when we add v are said to be *covered* by v.

A vertex $v \neq v_1, v_2, v_3$ of in-degree 2 is called *forward-oriented (backward-oriented)* if it covers a forward (backward) edge. Let $a_{\rm f}$ and $a_{\rm b}$ be the numbers of forward-oriented and backward-oriented vertices. Note that the values of $a_{\rm f}$, $a_{\rm b}$ depend on the canonical ordering.

Assume now that $n \geq 4$. Each vertex $v \neq v_1, v_2$ will be classified as *stable* or *unstable*. Also, with each such vertex we will associate a sequence of vertices called its *domino chain*, $DC(v)$, and a vertex $dom(v)$ called the *dominator* of v.

These concepts are defined as follows: For $v = v_n$, we define $DC(v_n) = (v_n)$, $dom(v_n)$ is undefined, and v_n is stable.

Suppose now $2 \leq k \leq n-2$, $v = v_{k+1}$, and let u be the leftmost neighbor of v in G_k, that is $ind_v(u) = 1$ (for $k = 2$ we assume $u = v_1$). Let also z be the vertex that covers edge (u, v). Such z must exist because $v \neq v_n$. Then:

(dc1) If $ind_z(v) = 2$, then $DC(v) := (v)$, $dom(v) := z$ and v is unstable.

(dc2) If $ind_z(v) \geq 4$, then $DC(v) := (v)$, $dom(v) := z$ and v is stable.

(dc3) If $ind_z(v) = 3$ and $DC(z) = (z_1, \ldots, z_i, z)$, then $DC(v) := (z_1, \ldots, z_i, z, v)$ and $dom(v) := dom(z)$. Also, v is stable iff z is stable.

An unstable vertex of in-degree 2 is called a *room-shift vertex.*

The intuition is that a stable vertex v can be placed above its leftmost neighbor u, saving one x-coordinate, while an unstable one may need to be placed one x-coordinate to the right. In particular, if v is a room-shift vertex, then this can result in putting v directly above its right neighbor and violating our invariant. Thus, in that case, we also need to shift v's right neighbor to the right in order to "make room" for v.

Note that, for $n \geq 4$, $DC(v_3)$ contains only vertex v_3, and the dominator of v_3 is the vertex z that covers edge (v_1, v_3); thus $ind_z(v_3) = 2$. For this reason, v_3 is considered unstable, although it is not relevant to the algorithm. The dominator of v_3 will only play a role in the width estimate.

Example: Consider the ordered graph in Fig 1. We have $DC(10) = (13, 11, 10)$, $dom(10) = 19$, $DC(8) = (13, 11, 10, 9, 8)$, $dom(8) = 19$, $DC(15) = (16, 15)$, $dom(15) = 17$, $DC(12) = (18, 12)$ and $dom(12) = 19$. Since $ind_{18}(17) = 4$, vertex 17 is stable. Since $ind_{17}(16) = 2$, vertices 16, 15 are unstable. Vertices 4, 5, 7, 8, 15 are room-shift vertices, and 14 has in-degree 2 but is not a room-shift vertex. Note that domino chains in this example are either disjoint or one is a prefix of another. Also, no two unstable vertices share a dominator. ♠

Algorithm $\mathcal{A}$: Given a maximal plane graph G, with $n \geq 3$ vertices, pick any edge (v_1, v_2) on the external face of G, and find its canonical ordering $\pi = v_1, \ldots, v_n$. If the number of forward-oriented vertices in π is greater than the number of backward-oriented vertices, then we modify π by swapping v_1 and v_2.

At this point, we are given an ordered maximal plane graph (G, π). If $n = 3$, we define $(x, y)(v_1) = (0, 0)$, $(x, y)(v_2) = (1, 0)$ and $(x, y)(v_3) = (0, 1)$, and the algorithm terminates.

Assume now that $n \geq 4$. We first embed v_1, v_2, v_3, as follows: $(x, y)(v_1) = (0, 0)$, $(x, y)(v_2) = (2, 0)$ and $(x, y)(v_3) = (1, 1)$.

After this initialization, we add vertices in order $v_4, \ldots, v_n$. Suppose $3 \leq k \leq n - 1$, and that we are about to add $v = v_{k+1}$. As usual, let $C_k = (w_1, \ldots, w_m)$ and $w_p, \ldots, w_q$ be the neighbors of v in G_k. If v is stable then $x(v) := x(w_p)$. Otherwise, $x(v) := x(w_p) + 1$ and, additionally, if $deg^-(v) = 2$ then we do $shift(w_q)$.

In both cases the $y(v)$ is chosen to be the smallest integer such that $(x', y') = (x(v), y(v))$ satisfies requirements (gsm2) and (gsm3).

If we swapped v_1, v_2 at the beginning of the algorithm, the clockwise ordering of edges incident to each vertex will be reversed with respect to the one in the given planar embedding. This can be easily modified, if desired, by using the left-right reflection: set $x_0 := x(v_2)$ and then $x(v_k) := x_0 - x(v_k)$ for all k.

This completes the description of Algorithm $\mathcal{A}$. Now we will prove its correctness and estimate the grid size.

Lemma 3 *Assume $n \geq 4$, and let $u, v \neq v_1, v_2$. Then*

(a) *If $u \in DC(v)$ then $DC(u)$ is a prefix of $DC(v)$.*

(b) *If $u \notin DC(v)$ and $v \notin DC(u)$ then:*

- (b1) *$DC(u) \cap DC(v) = \emptyset$.*
- (b2) *If u, v are unstable, then $dom(u) \neq dom(v)$.*

Proof: Part (a) follows directly from the definition of domino chains, since the predecessor of each vertex in a domino-chain is uniquely defined.

We now prove Part (b). Suppose that $u \notin DC(v)$ and $v \notin DC(u)$. The only way to violate (b1) is when two domino-chains have a common prefix. We will show that this cannot occur. Let z be the last vertex in this prefix, and let z' and z'' be its successors in $DC(u)$ and $DC(v)$, respectively. By the definition of domino chains, we then have that $z', z'' \prec z$, $ind_z(z') = 3$ and $ind_z(z'') = 3$, reaching a contradiction. Thus (b1) holds.

Now we prove (b2). Suppose that $u \notin DC(v)$, $v \notin DC(u)$, and u, v are unstable. Let z' and z'' be the first vertices in $DC(u)$ and $DC(v)$, respectively. By (b1), $z' \neq z''$. Suppose

$z = dom(u) = dom(v)$. Then also $z = dom(z') = dom(z'')$. We have $z', z'' \prec z$. Since u is unstable, all vertices in $DC(u)$, including z', are unstable. Similarly, z'' is unstable. We conclude that $ind_z(z') = ind_z(z'') = 2$ – a contradiction. □

Theorem 2 *If G is a given maximal plane graph with $n \geq 3$ vertices, then Algorithm $\mathcal{A}$ produces a grid drawing of G of width $\lfloor 2(n-1)/3 \rfloor$ and height $n^2/4$.*

Proof: The theorem holds obviously for $n = 3$, so we assume that $n \geq 4$.

Correctness: We will show that the following invariant holds at each step $k = 3, \ldots, n$:

(I): Let $C_k = (w_1, \ldots, w_m)$ be the current contour. Then

- (I1): For each $j = 1, \ldots, m-1$, we have
  - (a) $x(w_j) \leq x(w_{j+1})$, and
  - (b) $x(w_j) = x(w_{j+1})$ iff $w_j \prec w_{j+1}$ and w_{j+1} is stable. Also, $x(w_j) = x(w_{j+1})$ only if $y(w_j) < y(w_{j+1})$.
- (I2) If $k < n$, and $w_p, \ldots, w_q$ are the neighbors of $v = v_{k+1}$ in G_k, then after adding v we have $x(w_p) \leq x(v) < x(w_q)$, v is above C_k (as defined in (gsm2)), and all $w_p, \ldots, w_q$ are visible from v.

Since (I2) implies that the choices made by Algorithm $\mathcal{A}$ satisfy conditions (gsm1)–(gsm3), the correctness follows directly from Lemma 2.

Thus it is sufficient to show that (I) holds at each step. Invariant (I) is true, trivially, for $k = 3$. Assume (I1) holds for some $3 \leq k < n$. In the inductive step we will show that (I1) implies (I2), and that (I1) is preserved after adding $v = v_{k+1}$.

Claim A: $x(w_{p+1}) > x(w_p)$.

If $w_{p+1} \prec w_p$ then $x(w_{p+1}) > x(w_p)$ by the inductive assumption (I1). Suppose that $w_p \prec w_{p+1}$. Since $ind_v(w_{p+1}) = 2$, we have $v = dom(w_{p+1})$ and w_{p+1} is unstable. Claim A follows now from the inductive assumption.

Claim B: If v is unstable and $\deg(v) \geq 3$, then $x(w_{p+2}) > x(w_{p+1})$.

If $w_{p+2} \prec w_{p+1}$, then Claim B follows from the inductive assumption (I1). Suppose that $w_{p+1} \prec w_{p+2}$. Since v covers edge (w_{p+1}, w_{p+2}), and $ind_v(w_{p+2}) = 3$, the fact that v is unstable implies that w_{p+2} is unstable, and now Claim B follows from inductive assumption (I1).

Suppose now that v is stable. By Claim A and (I1), if we set $x(v) = x(w_p)$ and choose $y(v)$ to be large enough, then v will be located above C_k and all vertices $w_p, \ldots, w_q$ will be visible from v. We also have $x(w_p) = x(v) < x(w_{p+1}) \leq x(w_q)$, completing the proof of (I2). Since the new contour is $C_{k+1} = (w_1, \ldots, w_p, v, w_q, \ldots, w_m)$, these inequalities, together with the inductive assumption imply that (I1) is preserved.

The second case is when v is unstable and $\deg(v) \geq 3$. Claims A and B imply that $x(w_p) < x(w_{p+1}) < x(w_{p+2})$. Therefore, using (I1), after setting $x(v) = x(w_p) + 1$ and choosing $y(v)$ large enough, v will be located above C_k and all vertices $w_p, \ldots, w_q$ will be visible from v. Since we also have $x(w_p) < x(v) \leq x(w_{p+1}) < x(w_{p+2}) \leq x(w_q)$, (I2) follows. These inequalities, together with the inductive assumption, imply also that (I1) is preserved.

Finally, consider the case when v is unstable and $deg^-(v) = 2$, that is v is a room-shift vertex. By Claim A, after executing the shift operation, we have $x(w_{p+1}) \geq x(w_p) + 2$. Therefore, by setting $x(v) = x(w_p) + 1$, and choosing $y(v)$ so that v is above the line segment joining w_p and $w_q = w_{p+1}$, we can assure that w_p and w_q will be visible from v. Since we also have $x(w_p) < x(v) < x(w_q)$, (I2) follows. These last inequalities, together with the inductive assumption imply that (I1) is preserved.

Width estimate: Notice first that if π is an arbitrary canonical ordering of G and π' is its "left-right mirror copy" obtained by swapping v_1 with v_2, then each vertex of degree 2 is forward-oriented in π iff it is backward-oriented in π'. Therefore in the canonical ordering π computed in $\mathcal{A}$ we have $a_{\mathrm{f}} \leq a_{\mathrm{b}}$.

Let ω be the width of the drawing constructed by algorithm $\mathcal{A}$, and denote by a^{rs} the number of room-shift vertices (other than v_1, v_2, v_3). Then $\omega = a^{\mathrm{rs}} + 2$. Observe that a dominator cannot be a backward-oriented vertex of in-degree 2. By Lemma 3, dominators of room-shift vertices are distinct, and they are distinct from the dominator of v_3. This implies that $a^{\mathrm{rs}} \leq a_{\mathrm{f}} + b - 1$, and we get

$$\omega \;=\; a^{\mathrm{rs}} + 2 \;\leq\; (a_{\mathrm{f}} + b - 1) + 2 \;\leq\; a/2 + b + 1 \;=\; n - a/2 - 2.$$

On the other hand, $\omega \leq a + 2$. Therefore we get

$$\omega \leq \min(a + 2, n - a/2 - 2) \;\leq\; 2(n-1)/3,$$

as required.

Height estimate: The slope of a contour edge (w_i, w_{i+1}) is defined as

$$\mathit{slope}(w_i, w_{i+1}) \;=\; \frac{y(w_{i+1}) - y(w_i)}{x(w_{i+1}) - x(w_i)},$$

where for $x(w_{i+1}) = x(w_i)$ we assume that the above value is ∞. (By (I1), $x(w_{i+1}) = x(w_i)$ implies that $y(w_{i+1}) > y(w_i)$.)

Let $-\gamma$ be the smallest slope among the edges in the current contour. By the invariant (I1), $\gamma < \infty$. After we add a room-shift vertex to the contour, the smallest contour slope is at least $-\gamma$. After we add a vertex which is not a room-shift vertex the smallest contour slope will be at least $-\gamma - 1$.

Thus, if G has a^{rs} room-shift vertices, then the width of the drawing is exactly $a^{\mathrm{rs}} + 2$ and the edge (v_n, v_2) has slope at least $-(n - 2 - a^{\mathrm{rs}})$. Therefore the height of the drawing is at most $(a^{\mathrm{rs}} + 2) \cdot (n - a^{\mathrm{rs}} - 2) \leq n^2/4$. □

6 Reducing Height

In this section we will show how we can reduce the grid height to $4\lfloor 2(n-1)/3 \rfloor - 1$.

As usual, let $C_k = (w_1 = v_1, w_2, \ldots, w_m = v_2)$ be the contour of G_k. Then each contour edge (w_i, w_{i+1}) belongs to one of the following four types:

vertical: when $x(w_i) = x(w_{i+1})$.

horizontal: when $y(w_i) = y(w_{i+1})$.

upward: when $y(w_i) < y(w_{i+1})$.

downward: when $y(w_i) > y(w_{i+1})$.

In Algorithm $\mathcal{A}$ vertical edges were always forward, but horizontal, upward and downward edges could be either forward or backward. Algorithm $\mathcal{B}$ will have the same property.

In the proof of Theorem 2 we defined the slope of contour edges. In general, given two vertices u, v, the *slope* of the segment from $(x,y)(u)$ to $(x,y)(v)$ is defined in a standard fashion:

$$slope(u,v) \quad = \quad \frac{y(v) - y(u)}{x(v) - x(u)},$$

where for $x(v) = x(u)$ we assume that $slope(u,v)$ is $\pm\infty$, depending on the sign of $y(v) - y(u)$. If (u,v) is an edge, we will sometimes say that the slope of edge (u,v) is $slope(u,v)$, if the grid drawing functions (x,y) are understood from context.

We also define the *slack* between u and v by

$$slack(u,v) \quad = \quad y(v) + 4[x(v) - x(u)] - y(u).$$

Thus we have the following relationship between the slack and slope:

$$slope(u,v) \quad = \quad -4 + \frac{slack(u,v)}{x(v) - x(u)}.$$

We will distinguish two types of shifts. Let v be a vertex to be installed. As in Algorithm $\mathcal{A}$, a *room-shift* occurs when v is a room-shift vertex. A *slope-shift* occurs when we shift the rightmost neighbor w_q of v in order to reduce the absolute value of the slope of edge (v, w_q). We will call such v *slope-shift vertices.* No two shifts will occur simultaneously. A *shift vertex* is either a room-shift or a slope-shift vertex.

A vertex v is *slack-preserving* if either

(sp1) $deg^-(v) \geq 4$, or

(sp2) $deg^-(v) = 3$ and v is stable.

Also, v is *slack-reducing* if either

(sr1) $deg^{-}(v) = 3$ and v is unstable, or

(sr2) $deg^{-}(v) = 2$ and v is stable.

Note that the two above concepts are not complementary, since room-shift vertices are neither slack-preserving nor slack-reducing.

The main intuition behind our method can be explained as follows: Suppose that we are about to install a slack-preserving vertex $v = v_{k+1}$ and let w_q be its rightmost neighbor in C_k. Assume that the edge (w_{q-1}, w_q) is downward. In order for w_q to be visible from v, the slope of (v, w_q) must be smaller than that of (w_{q-1}, w_q). (Recall that both values are negative.) The important, though simple, observation is that in this case, because of the definition of slack-preserving vertices, we must have $x(v) < x(w_{q-1})$ and then, even though the slope decreases, the slack of (v, w_q) can still remain the same as that of (w_{q-1}, w_q).

In Algorithm $\mathcal{B}$ we follow, in general, the same strategy as in Algorithm $\mathcal{A}$, except that we place vertices more carefully in order to reduce the height. This will require, in some cases, making more shifts than in Algorithm $\mathcal{A}$. Nevertheless, in the worst case, Algorithm $\mathcal{B}$ does not use more x coordinates than Algorithm $\mathcal{A}$.

Algorithm $\mathcal{B}$: The choice of the canonical ordering $\pi = v_1, \ldots, v_n$ and the initialization are exactly the same as in Algorithm $\mathcal{A}$.

Assume now that $n \geq 4$, and suppose that we are now about to add a vertex $v = v_{k+1}$, for $3 \leq k \leq n-1$. As usual, we denote $C_k = (w_1, \ldots, w_m)$, and $w_p, \ldots, w_q$ are the neighbors of v in G_k.

Let $x(v) := x(w_p)$ if v is stable, otherwise $x(v) := x(w_p) + 1$. Then we consider three cases.

Case 1: If v is slack-preserving, then

$$y(v) := y(w_q) + 4[x(w_q) - x(v)] - slack(w_{q-1}, w_q).$$

Note that $slack(v, w_q) = slack(w_{q-1}, w_q)$.

Case 2: If v is slack-reducing, then

$$y(v) := y(w_q) + 4[x(w_q) - x(v)] - slack(w_{q-1}, w_q) + 1.$$

Note that $slack(v, w_q) = slack(w_{q-1}, w_q) - 1$. Then, if $slack(v, w_q) = 0$, we slope-shift w_q by executing $shift(w_q)$.

Case 3: If v is a room-shift vertex, then we room-shift w_q by executing $shift(w_q)$. If (w_p, w_q) is upward, then $y(v) := y(w_q)$. If (w_p, w_q) is horizontal, then $y(v) := y(w_p) + 1$. If (w_p, w_q) is downward, then $y(v) := y(w_p)$.

This completes the description of Algorithm $\mathcal{B}$. Now we will prove its correctness and the bound on the grid size.

Lemma 4 *Algorithm $\mathcal{B}$ produces a correct grid drawing, and the height of this drawing is less than 4 times its width, that is $y(v_n) < 4x(v_2)$.*

Proof: We prove the following invariant:

(J) Let $3 \leq k \leq n$, and $C_k = (w_1, \ldots, w_m)$. Then the drawing produced by Algorithm $\mathcal{B}$ satisfies the following conditions:

- (J1) For all $i = 1, \ldots, m-1$, we have
  - (a) $x(w_i) \leq x(w_{i+1})$ and,
  - (b) $x(w_i) = x(w_{i+1})$ iff $w_i \prec w_{i+1}$ and w_{i+1} is stable. Also, $x(w_i) = x(w_{i+1})$ only if $y(w_i) < y(w_{i+1})$.
  - (c) $\mathit{slack}(w_i, w_{i+1}) \geq 1$. Additionally, if (w_i, w_{i+1}) is forward (that is, $w_i \prec w_{i+1}$) but not vertical then $\mathit{slack}(w_i, w_{i+1}) \geq 2$.
- (J2) If $k < n$, and $w_p, \ldots, w_q$ are the neighbors of $v = v_{k+1}$ in G_k, then v is above C_k (as defined in (gsm2)), $x(w_p) \leq x(v) < x(w_q)$, and all $w_p, \ldots, w_q$ are visible from v.

The proof of Invariant (J) is by induction on k. (J1) holds for $k = 3$, by inspection. Suppose G_k satisfies (J1) for some $k \geq 3$. In the inductive step we will show that (J1) implies (J2) and that (J1) is preserved after adding $v = v_{k+1}$.

Assume that (J1) holds for G_k. Observe that for all $1 \leq i < j \leq m$ we have

$$\begin{aligned} \mathit{slack}(w_i, w_j) &= y(w_j) + 4[x(w_j) - x(w_i)] - y(w_i) \\ &= \sum_{\ell=i}^{j-1} \{y(w_{\ell+1}) + 4[x(w_{\ell+1}) - x(w_\ell)] - y(w_\ell)\} \\ &= \sum_{\ell=i}^{j-1} \mathit{slack}(w_\ell, w_{\ell+1}). \end{aligned}$$

Thus, by (c), for all $i \leq i' \leq j' \leq j$, if $i < i'$ or $j' < j$ then

$$\mathit{slack}(w_i, w_j) \quad > \quad \mathit{slack}(w_{i'}, w_{j'}).$$

Now we consider five cases.

Case A: v is stable and $\mathit{deg}^-(v) \geq 3$.

In this case v satisfies Case 1 of Algorithm $\mathcal{B}$. We prove first that (J1) implies (J2). We have $x(w_p) = x(v) < x(w_{p+1})$. If $p + 1 \leq j \leq q - 1$ then $x(w_j) > x(v)$ and $\mathit{slack}(w_j, w_q) \geq$

$slack(w_{q-1}, w_q)$. If $j = p$, then $slack(w_j, w_q) > slack(w_{q-1}, w_q)$, by the argument in the previous paragraph. Therefore, for all $j = p, \ldots, q-1$,

$$\begin{aligned} y(w_j) &= y(w_q) + 4[x(w_q) - x(w_j)] - slack(w_j, w_q) \\ &< y(w_q) + 4[x(w_q) - x(v)] - slack(w_{q-1}, w_q) \\ &= y(v), \end{aligned}$$

implies that v is above C_k. (Note, however, that $y(w_q) > y(v)$ is possible.) For each $j = p+1, \ldots, q-1$, edge (v, w_j) has $slope(v, w_j) < 0$. By the description of the algorithm, we have $slope(v, w_{q-1}) = -4 < slope(v, w_q)$ and $slope(v, w_p) = -\infty$.

For $j = p+1, \ldots, q-1$, let $\lambda_j = slack(w_j, w_q) - slack(w_{q-1}, w_q) = slack(w_j, w_q) - slack(v, w_q)$, $\alpha_j = x(w_j) - x(v)$ and $\beta_j = y(v) - y(w_j)$. Note that $\lambda_j \geq 0$, $\alpha_j > 0$, $\beta_j > 0$, and $slope(v, w_j) = -\beta_j/\alpha_j$. Then

$$\beta_j - \lambda_j = 4[x(w_j) - x(v)] = 4\alpha_j.$$

Therefore

$$\frac{\beta_j - \lambda_j}{\alpha_j} = 4 = \frac{\beta_{j+1} - \lambda_{j+1}}{\alpha_{j+1}}.$$

Then $\alpha_j \leq \alpha_{j+1}$ and $\lambda_j > \lambda_{j+1} \geq 0$ imply that $\beta_j/\alpha_j > \beta_{j+1}/\alpha_{j+1}$.

In summary, we have: (i) $x(v) = x(w_p) < x(w_j)$ for $j = p+1, \ldots, q$, and (ii) $slope(v, w_j) < slope(v, w_{j+1})$ and $y(v) > y(w_j)$ for $j = p, \ldots, q-1$. This implies that all $w_p, \ldots, w_q$ are visible from v. Thus (J2) holds.

Now we show that (J1) is preserved after adding $v = v_{k+1}$. The new contour is $C_{k+1} = (w_1, \ldots, w_p, v, w_q, \ldots, w_m)$. We only need to worry about contour edges (w_p, v) and (v, w_q), since other edges do not move, except possibly for a rigid shift to the right. Since $x(w_p) = x(v) < x(w_q)$ and v is stable, (a) and (b) are true. Edge (v, w_q) is backward, and we have $slack(v, w_q) = slack(w_{q-1}, w_q) \geq 1$, by induction. Edge (w_p, v) is forward but vertical, and $slack(w_p, v) = y(v) - y(w_p) \geq 1$. Thus (c) holds after adding v.

Case B: v is unstable and $deg^-(v) \geq 4$.

As in the previous case, v satisfies Case 1 of Algorithm $\mathcal{B}$. We show first that (J2) holds. Similarly as in the proof of Theorem 2, we have $x(w_p) < x(v) \leq x(w_{p+1}) < x(w_{p+2}) \leq x(w_q)$. By an argument analogous to the previous case, $y(v) > y(w_j)$ for $j = p+1, \ldots, q-1$. (But it may happen that $y(v) < y(w_p)$ and/or $y(v) < y(w_q)$.)

We also have $slope(v, w_{q-1}) = -4 < slope(v, w_q)$, and, by an argument analogous to the previous case, $slope(v, w_j) < slope(v, w_{j+1})$ for all $j = p+2, \ldots, q-2$. If $x(v) < x(w_{p+1})$ then, similarly, $slope(v, w_{p+1}) < slope(v, w_{p+2})$. Otherwise, if $x(v) = x(w_{p+1})$, then $slope(v, w_{p+1}) = -\infty$.

In particular, we have $slope(w_p, w_{p+1}) > -4 = slope(v, w_{q-1}) > slope(v, w_{p+1})$. Since also $x(w_p) < x(v) \leq x(w_{p+1})$, we obtain that v is above edge (w_p, w_{p+1}) and, consequently, above C_k.

In summary, we have (i) v is located above (w_p, w_{p+1}), (ii) $x(v) \leq x(w_{p+1}) < x(w_i)$ for $i = p+2, \ldots, q$, and (iii) $y(v) > y(w_j)$ and $slope(v, w_j) < slope(v, w_{j+1})$ for all $j = p+1, \cdots q-1$. This implies that all $w_p, \ldots, w_q$ are visible from v. Thus (J2) holds.

Now we show that (J1) is preserved. The new contour is $C_{k+1} = (w_1, \ldots, w_p, v, w_q, \ldots, w_m)$. As in Case A, we only need to consider edges (w_p, v) and (v, w_q). These edges satisfy (a) and (b) because $x(w_p) < x(v) < x(w_q)$ and v is unstable.

We now prove (c). By induction, $slack(v, w_q) = slack(w_{q-1}, w_q) \geq 1$. Consider (w_p, v). In this case we have $x(v) > x(w_p)$ and thus we have to show that $slack(w_p, v) \geq 2$. We get

$$\begin{aligned} slack(w_p, v) &= slack(w_p, w_q) - slack(v, w_q) \\ &= slack(w_p, w_q) - slack(w_{q-1}, w_q) \\ &= slack(w_p, w_{q-1}) \\ &\geq slack(w_p, w_{p+2}) \;\geq\; 2. \end{aligned}$$

Thus (c) is preserved.

Case C: $deg^-(v) = 3$ and v is unstable.

Now v satisfies Case 2 of Algorithm $\mathcal{B}$, and thus a slope-shift can occur. By the same argument as in the previous case, we can show that before the shift (if any) we have the following: (i) $x(w_p) < x(v) \leq x(w_{p+1}) < x(w_q)$ (ii) v is located above (w_p, w_{p+1}), and (iii) $slope(v, w_{p+1}) < slope(v, w_q)$. (Recall that $q = p + 2$.) This implies that (J2) holds before $shift(w_q)$. This is sufficient, because (J2) is preserved after $shift(w_q)$.

It remains to show that (J1) holds after adding $v = v_{k+1}$. Again, we need to consider only edges (w_p, v) and (v, w_q). Parts (a) and (b) are obvious, because $x(w_p) < x(v) < x(w_q)$ and v is unstable. We get

$$\begin{aligned} slack(w_p, v) &= slack(w_p, w_q) - slack(v, w_q) \\ &= slack(w_p, w_q) - slack(w_{q-1}, w_q) + 1 \\ &= slack(w_p, w_{q-1}) + 1 \\ &= slack(w_p, w_{p+1}) + 1 \;\geq\; 2. \end{aligned}$$

Thus (c) is preserved for (w_p, v). Consider now (v, w_q). If $slack(w_{q-1}, w_q) \geq 2$ then $slack(v, w_q) = slack(w_{q-1}, w_q) - 1 \geq 1$. If $slack(w_{q-1}, w_q) = 1$ then $slack(v, w_q) = 4$, because of the shift operation, and thus (c) holds for (v, w_q) as well.

Case D: $deg^-(v) = 2$ and v is stable.

Vertex v is slack-reducing, so it satisfies Case 2 of Algorithm $\mathcal{B}$. We have $x(w_p) = x(v) < x(w_q)$, and $y(v) > y(w_p)$ (by simple calculations), implying (J2).

Since $x(w_p) = x(v) < x(w_q)$, (a) and (b) are preserved. Edge (w_p, v) is vertical, so (c) is true for (w_p, v). Consider (v, w_q). If (w_p, w_q) is upward or horizontal then $\mathit{slack}(v, w_q) = \mathit{slack}(w_p, w_q) - 1 \geq 3$. If (v, w_q) is downward, then the argument is the same as in Case C.

Case E: $\mathit{deg}^-(v) = 2$ and v is unstable (room-shift vertex).

Vertex v satisfies Case 3 of Algorithm $\mathcal{B}$. By the algorithm, before the shift occurs, v is always above the line segment joining w_p and w_q, so w_p and w_q are visible from v. The shift operation does not destroy the visibility. Thus (J2) is true.

Similarly, (J1) Parts (a), (b) are preserved, directly from the algorithm. We now prove (c). If (w_p, w_q) is upward, then (w_p, v) is upward and (v, w_q) is horizontal, so both have positive slack. If (w_p, w_q) is horizontal, then (w_p, v) is upward, so it has a positive slack, and $\mathit{slack}(v, w_q) = y(w_q) + 4[x(w_q) - x(v)] - y(v) \geq -1 + 4 \cdot 1 > 0$ holds after executing $\mathit{shift}(w_q)$. The last case is when (w_p, w_q) is downward. Then $\mathit{slack}(w_p, v) > 0$ because (w_p, v) is horizontal. We also have $\mathit{slack}(v, w_q) = y(w_q) + 4[x(w_q) - x(v)] - y(v) = y(w_q) + 4[(x(w_q) - 1) - x(w_p)] - y(w_p)$ after executing $\mathit{shift}(w_q)$. The last expression is equal to the value of $\mathit{slack}(w_p, w_q)$ before executing $\mathit{shift}(w_q)$, and therefore $\mathit{slack}(v, w_q) > 0$, by induction. This proves that (c) is preserved.

This completes the proof of Invariant (J). The correctness of the drawing follows now directly from (J2) and Lemma 2. Invariant (J1), part (c), implies that $\mathit{slope}(v_n, v_2) > -4$, and therefore $y(v_n) < 4x(v_2)$, as required. □

Lemma 5 *If G is a n-vertex plane graph, then Algorithm $\mathcal{B}$ computes a grid drawing of G of width at most $\lfloor 2(n-1)/3 \rfloor$.*

Proof: The proof is more complicated than the one for Algorithm $\mathcal{A}$, as Algorithm $\mathcal{B}$ makes more shift operations and we need to count much more carefully.

Let $v = v_{k+1}$, for $k \geq 3$, be a slack-reducing vertex. Let w_q, w_{q-1} be the last and next-to-last neighbors of v in C_k, that is $\mathit{ind}_v(w_q) = \mathit{deg}^-(v)$ and $\mathit{ind}_v(w_{q-1}) = \mathit{deg}^-(v) - 1$. (Recall that $\mathit{deg}^-(v) = 2$ or 3.) We say that v is *slack-critical*, if edge (w_{q-1}, w_q) is downward. Note that all slope-shift vertices are slack-critical.

If v is a backward-oriented room-shift-vertex, we will assign to v its mate $\nu(v)$. Vertex $\nu(v)$ will be either

(mr1) a forward-oriented stable, but not slack-critical, vertex of in-degree 2, or

(mr2) an unstable, but not slack-critical, vertex of in-degree 3, or

(mr3) a slack-preserving vertex.

If v is a slope-shift vertex, we will assign to it two mates $\mu_1(v)$ and $\mu_2(v)$. In this case, each $\mu_i(v)$ will be (ms) a slack-critical, but not slope-shift, vertex of in-degree 2 or 3.

The important properties of mates are:

(ma1) All mate vertices are different.

(ma2) No mate vertex is a shift vertex.

Suppose that we can find such mate vertices. We show that this implies the $\lfloor 2(n-1)/3 \rfloor$ upper bound on the grid width. The method for constructing mates will be given later.

As before, let a and b be the numbers of vertices of in-degree 2, and in-degree at least 3, respectively. By a_{f} and a_{b} we denote the numbers of forward and backward-oriented vertices of in-degree 2.

For $\xi = a, b$, by ξ^{rs}, ξ^{ss} and ξ^{ns} we denote the number of ξ-vertices which are room-shift, slope-shift and no-shift, respectively.

By a^{mr} and b^{mr} we denote the number of vertices of in-degree 2, and at least 3, respectively, that satisfy conditions (mr1)–(mr3) above. By a^{ms} and b^{ms} we denote the number of vertices of in-degree 2 and 3, respectively, that satisfy condition (ms).

We will also combine subscripts and superscripts, with an obvious interpretation. For example, $a_{\mathrm{f}}^{\mathrm{ns}}$ is the number of no-shift forward-oriented vertices of in-degree 2. Some such combinations will be void, for example, by the definition of room-shift vertices, we have $b^{\mathrm{rs}} = 0$. Also, by Invariant (J1.c) in the proof of Lemma 4, forward edges must have slack at least 2, implies that $a_{\mathrm{f}}^{\mathrm{ss}} = 0$.

Let ω be the width of the grid drawing of G produced by Algorithm $\mathcal{B}$. First, we have the following equations:

$$\begin{aligned} \omega &= a^{\mathrm{rs}} + a^{\mathrm{ss}} + b^{\mathrm{ss}} + 2 && (1) \\ n &= a + b + 3 \\ &= a^{\mathrm{rs}} + a^{\mathrm{ss}} + b^{\mathrm{ss}} + a^{\mathrm{ns}} + b^{\mathrm{ns}} + 3 \\ &= \omega - 2 + a^{\mathrm{ns}} + b^{\mathrm{ns}} + 3 \\ &= \omega + (a^{\mathrm{ns}} + b^{\mathrm{ns}}) + 1 && (2) \end{aligned}$$

Since $a_{\mathrm{f}} \leq a_{\mathrm{b}}$ and $a_{\mathrm{f}}^{\mathrm{ss}} = 0$, we have

$$a_{\mathrm{f}}^{\mathrm{rs}} + a_{\mathrm{f}}^{\mathrm{ns}} \leq a_{\mathrm{b}}^{\mathrm{rs}} + a_{\mathrm{b}}^{\mathrm{ss}} + a_{\mathrm{b}}^{\mathrm{ns}} \quad (3)$$

By the existence of the mates satisfying the conditions described above we have

$$\begin{aligned} a_{\mathrm{b}}^{\mathrm{rs}} &\leq a_{\mathrm{f}}^{\mathrm{mr}} + b^{\mathrm{mr}} - 1 && (4) \\ 2(a_{\mathrm{b}}^{\mathrm{ss}} + b^{\mathrm{ss}}) &\leq a_{\mathrm{f}}^{\mathrm{ms}} + a_{\mathrm{b}}^{\mathrm{ms}} + b^{\mathrm{ms}} && (5) \end{aligned}$$

We can subtract 1 in inequality (4) because v_n satisfies (mr3) but it cannot be a mate of any backward-oriented room-shift vertex. Multiply inequality (4) by 2, and add it to (5). This yields

$$\begin{aligned} 2(a_{\mathrm{b}}^{\mathrm{rs}} + a_{\mathrm{b}}^{\mathrm{ss}} + b^{\mathrm{ss}}) &\leq 2a_{\mathrm{f}}^{\mathrm{mr}} + a_{\mathrm{f}}^{\mathrm{ms}} + a_{\mathrm{b}}^{\mathrm{ms}} + 2b^{\mathrm{mr}} + b^{\mathrm{ms}} - 2 \\ &\leq 2a_{\mathrm{f}}^{\mathrm{ns}} + a_{\mathrm{b}}^{\mathrm{ns}} + 2b^{\mathrm{ns}} - 2, \end{aligned} \tag{6}$$

where the second inequality follows from conditions (ma1) and (ma2). Now add (6) and (3), getting

$$a_{\mathrm{f}}^{\mathrm{rs}} + a_{\mathrm{b}}^{\mathrm{rs}} + a_{\mathrm{b}}^{\mathrm{ss}} + 2b^{\mathrm{ss}} \leq a_{\mathrm{f}}^{\mathrm{ns}} + 2a_{\mathrm{b}}^{\mathrm{ns}} + 2b^{\mathrm{ns}} - 2 \tag{7}$$

$$\leq 2(a^{\mathrm{ns}} + b^{\mathrm{ns}}) - 2 \tag{8}$$

and then, using (8) and (1), we obtain

$$\begin{aligned} \omega &= a^{\mathrm{rs}} + a^{\mathrm{ss}} + b^{\mathrm{ss}} + 2 \\ &\leq a_{\mathrm{f}}^{\mathrm{rs}} + a_{\mathrm{b}}^{\mathrm{rs}} + a_{\mathrm{b}}^{\mathrm{ss}} + 2b^{\mathrm{ss}} + 2 \\ &\leq 2(a^{\mathrm{ns}} + b^{\mathrm{ns}}). \end{aligned} \tag{9}$$

Finally, inequalities (2) and (9) imply that

$$\omega \leq 2(n-1)/3$$

as required.

Now we show how to find mate vertices. First, let v be a backward-oriented room-shift vertex, and t its left neighbor, that is $ind_v(t) = 1$. Note that, by Algorithm $\mathcal{B}$, (t, v) is horizontal or upward. Let z be latest vertex in the ordering $\prec$ such that $ind_z(t) = 1$. (It may happen that $z = v$.) If z is stable, then we set $\nu(v) = z$. If $deg^-(z) = 2$, then z is forward-oriented and not slack-critical, satisfying (mr1). If $deg^-(z) \geq 3$, z satisfies (mr3).

Otherwise, if z is unstable, there is z' that covers (t, z). Since z is unstable and (by the choice of z) $ind_{z'}(t) \neq 1$, we have $deg^-(z') \geq 3$ and $ind_{z'}(t) = 2$. Then we set $\nu(v) = z'$. Note that z' must be unstable. If $deg^-(z') \geq 4$ then z' satisfies (mr3). Otherwise, if $deg^-(z') = 3$, z' satisfies (mr2), since (t, z) cannot be a downward edge (because (t, v) is not downward).

Claim A: all $\nu(v)$ are different.

Pick two different backward-oriented room-shift vertices u, v. Let t^u and t^v denote, respectively, the left neighbors of u and v, as in the above construction.

We show first that $t^u \neq t^v$. Suppose that $t^u = t^v$ and that, without loss of generality, that $u \prec v$. In that case v cannot be backward-oriented, reaching a contradiction.

Suppose now that $\nu(u) = \nu(v) = z$. By the construction of mates, we have $t^u, t^v \prec z$. Furthermore, either $ind_z(t^u) = 1$ and z is stable, or $ind_z(t^u) = 2$ and z is unstable. But the same two choices hold for v as well, and since $ind_z(t^u) \neq ind_z(t^v)$, we reach a contradiction. Thus $\nu(u) \neq \nu(v)$, completing the proof of Claim A.

Now we show how to construct mates of slope-shift vertices. For convenience, all occurrences of $slack(u, v)$ in the remaining of this proof will be the values of the slack of edge (u, v) at the time when it was in the contour. (Algorithm $\mathcal{B}$ does not change slack values of contour edges, but it may change slacks of non-contour edges.)

Let now v be a slope-shift vertex and t its rightmost neighbor, that is $ind_v(t) = deg^-(v)$. We define a sequence $\{r_i\}$ as follows: Initially, $r_0 = v$ and r_1 is the vertex such that $ind_v(r_1) = deg^-(v) - 1$. In other words, edge (r_1, t) is covered by v. Such r_1 must exist, and we must have $slack(r_1, t) = 1$, otherwise we wouldn't have executed a shift when adding v.

Suppose that we already have found r_i, such that $r_i \succ t$, r_i is a neighbor of t satisfying $ind_{r_i}(t) = deg^-(r_i)$, and $slack(r_i, t) = 1$. If r_i is slack-reducing, then we set $\mu_1(v) = r_i$. Such $\mu_1(v)$ satisfies (ms), since it has to be slack-critical, and it is not slope-shift vertex.

Otherwise, if r_i is not slack-reducing, let $r_{i+1} \prec r_i$ be the neighbor of r_i such that $ind_{r_i}(r_{i+1}) = deg^-(r_i) - 1$. Thus r_i covers edge (r_{i+1}, t) when we add it to the graph. Since r_i is not slack-reducing, it can be either slack-preserving or a room-shift vertex. If r_i is slack-preserving, then by the algorithm, we have $slack(r_{i+1}, t) = slack(r_i, t) = 1$, and therefore also $r_{i+1} \succ t$. (This follows from Invariant (J1.c) in the proof of Lemma 4.) If r_i is a room-shift vertex, then we have $slack(r_{i+1}, t) = slack(r_i, t) = 1$, implying again that $r_{i+1} \succ t$ by (J1.c).

Having determined $u = \mu_1(v)$, we proceed similarly to find $\mu_2(v)$. We find a sequence $\{s_j\}$ as follows. Let $s_0 = u$, and pick s_1 such that u covers (s_1, t), that is $ind_u(s_1) = deg^-(u) - 1$. By the above construction, (s_1, t) was a contour edge before adding u, and $slack(s_1, t) = 2$. Unlike in the construction of $\{r_i\}$, we cannot assume that $s_1 \succ t$.

Suppose we have already found a neighbor s_j of t such that (s_j, t) was a contour edge at some time and $slack(s_j, t) = 2$. We either have $s_j \succ t$ and $ind_{s_j}(t) = deg^-(s_j)$, or $s_j \prec t$ and $ind_t(s_j) = 1$.

If $s_j \succ t$ and s_j is slack-reducing, set $\mu_2(v) = s_j$. Then s_j satisfies (ms), because if we denote by z the vertex such that $ind_{s_j}(z) = deg^-(s_j) - 1$, then $slack(z, t) = 3$, implying that (z, t) was a downward contour edge.

If $s_j \succ t$ but s_j is not slack reducing, then we continue the construction. Pick s_{j+1} such that $ind_{s_j}(s_{j+1}) = deg^-(s_j) - 1$. Then (s_{j+1}, t) is the contour edge covered by s_j, and $slack(s_{j+1}, t) = 2$.

The remaining case is when $s_j \prec t$. In this case $slack(s_j, t) = 2$ and (s_j, t) is forward and downward. Pick t' such that $ind_t(t') = 2$, that is (s_j, t') is the first edge covered by t. By simple calculations, similar to those in the proof of Lemma 4, this is possible only when $deg^-(t)$ is 3 or 4, and in both these cases we must have $slack(s_j, t') = 1$. At this point, we continue as in the construction of $\{r_i\}$: find the sequence $s_j = \bar{r}_0, \ldots, \bar{r}_l$ and take $\mu_2(v) = \bar{r}_l$.

Claim B: All $\mu_i(v)$ are different.

Let v be a slope shift vertex and t is its rightmost neighbor. Let also $\{r_i\}$, $\{s_j\}$ and $\{\bar{r}_l\}$ be the sequences from the above construction.

Pick any r_f. Since t is the rightmost neighbor of r_f, t is uniquely determined by r_f. This means also that, if $f \geq 1$, r_{f-1} is uniquely determined by r_f. Since $slack(r_i, t) = 1$ for $i \geq 1$ and $slack(r_0, t) = 4$ (recall that $r_0 = v$), vertex v is uniquely determined, given r_f. An analogous argument shows that s_0 can be uniquely determined from any s_g, and $\bar{r}_0$ from any $\bar{r}_h$. This implies Claim B.

By Claims A,B, and by the fact that the conditions (mr1)–(mr3) contradict condition (ms), no two mate vertices can be equal. Thus the proof is complete. □

Theorem 3 *Given a maximal n-vertex plane graph G, Algorithm $\mathcal{B}$ constructs a grid drawing of G into a $\lfloor 2(n-1)/3 \rfloor \times 4\lfloor 2(n-1)/3 \rfloor - 1$ grid.*

Theorem 4 *Algorithm $\mathcal{B}$ can be implemented in linear time.*

Proof: The implementation is very similar to the one in [CP89], so we only sketch it briefly here.

Canonical orderings can be computed in linear time, as described in [FPP88, Ka93]. To determine the final canonical ordering, we determine which edges are forward and which are backward, and compute the number of forward-oriented and backward-oriented vertices of in-degree 2. If necessary, we will interchange v_1 and v_2 in π. It all can be done easily in linear time.

The straighforward implementation of the construction of the drawing runs in $\Omega(n^2)$ time, since the shift operations can cost as much as $\Omega(n)$ time.

In order to speed it up, we need to install each v_k in time $O(deg^-(v_k))$, which adds up to $O(n)$. This is achieved by postponing the shift operations, and computing only relative x-distances between vertices whenever necessary.

Represent the structure of U-sets in a directed tree T. At each time step, T contains contour edges $w_i \rightarrow w_{i+1}$. Also, if v covers u, then T will contain edge $u \rightarrow v$. Vertex v_1 is the root of T.

For each vertex v, store $y(v)$. The y-coordinates do not change during the algorithm. For each edge $u \rightarrow v$ in T, store the offset value $\Delta x(u, v)$ which is equal to $x(v) - x(u)$. However, we do *not* store the x-coordinates. If (w_j, w_{j+1}) is a contour edge, then a shift operation $shift(w_{j+1})$ affects only one offset value, namely $\Delta x(w_j, w_{j+1})$. Also, given all $\Delta x(w_i, w_{i+1})$ for contour edges, we can determine, in $O(deg^-(v_k))$ time, the shape of the path $w_p, \ldots, w_q$ (but not its exact location in the plane), and this information is sufficient to determine $y(v_k)$, $\Delta x(w_p, v_k)$ and $\Delta x(v_k, w_j)$ for all $j = p+1, \ldots, q$. □

7 Final Comments

We have shown that plane graphs have grid drawings of width $\lfloor 2(n-1)/3 \rfloor$, which is optimal, and height $4\lfloor 2(n-1)/3 \rfloor - 1$.

Our height analysis is nearly tight, since there are examples of graphs on which Algorithm $\mathcal{B}$ uses grids of height $4\lfloor 2(n-1)/3 \rfloor - 3$. The most intriguing question is whether the grid height can be improved. A minor improvement can be achieved by treating v_n in a special way, and setting $x(v_n) = 1$ instead of 0. This reduces the height to $4\lfloor 2(n-1)/3 \rfloor - 4$ (or less).

Another possibility for improving the height is to use more a restrictive invariant on the slope, by replacing the bound of -4 by $-3, -2$ or -1. It is not hard to see that one obtains correct grid drawings with this change. Unfortunately, we have examples showing that such modified algorithms do not give optimum-width drawings.

Consider the slope bound of -3. The following example forces Algorithm $\mathcal{B}$ to use width $5n/7 + O(1)$. We start with the triangle (v_1, v_2, v_3). Identify $t_0 = v_3$, and add vertices $t_1, \ldots, t_m$, where each t_i, for $i > 0$, is connected to v_1 and t_{i-1}. Then, for each $i = 0, \ldots, m-1$ we add the component $D_i = \{t_i, c_i, d_i, e_i, f_i, g_i, h_i\}$ with edges:

- (c_i, t_i), (c_i, t_{i+1}),
- (d_i, c_i), (d_i, t_i),
- (e_i, d_i), (e_i, t_i),
- (f_i, t_{i+1}), (f_i, c_i),
- (g_i, f_i), (g_i, t_{i+1}),
- (h_i, t_{i+1}), (h_i, g_i), (h_i, f_i), (h_i, c_i), (h_i, d_i), (h_i, e_i) and (h_i, t_i).

We add, symmetrically, vertices t'_i, and components $D'_i = \{t'_i, c'_i, \ldots, h'_i\}$, on the other side of v_3. Finally, we add one more vertex and connect it to all vertices on the outer face. (See Fig. 2.) It is easy to see that is independent of the canonical ordering and we have $a_{\mathrm{f}} = a_{\mathrm{b}}$. In each component D_i, we make room-shifts for t_i, c_i, f_i, g_i and one slope-shift for e_i. Thus, for each group of 7 vertices, we make 5 shifts.

Heuristics. The previous discussion leads to some simple heuristics, with which the height of grid drawings can be reduced for most, if not all, graphs. For example, many slack-preserving vertices v_k can be installed below their location determined by Algorithm $\mathcal{B}$. Instead of using the location that preserves the slack, we can set $y(v_k)$ to be smallest possible, without violating conditions (gsm1)–(gsm3). Quite possibly, this can improve the height to $cn + O(1)$, for $c < 8/3$, even in the worst case.

Another possible heuristic is to use smaller slope bound, as indicated above. Even though this may increase the width, the resulting drawing may be lower than the one obtained with slope bound -4. This also provides additional flexibility; by experimenting with different slope bounds, one can choose a drawing that best suits his application.

References

[BETT] G. Di Battista, P. Eades, R. Tamassia, I.G. Tollis, *Automatic graph drawing: an annotated bibliography*, to appear in Computational Geometry: Theory and Applications.

[CP89] M. Chrobak, T. Payne, *A linear-time algorithm for drawing planar graphs on a grid*, Technical Report UCR-CS-89-1, Department of Mathematics and Computer Science, University of California at Riverside, 1989.

[CK93] M. Chrobak, G. Kant, *Convex grid drawings of 3-connected planar graphs*, Technical Report RUU-CS-93-45, Department of Computer Science, Utrecht University, 1993.

[CYN84] N. Chiba, T. Yamanouchi, T. Nishizeki, *Linear algorithms for convex drawings of planar graphs*, in Progress in Graph Theory, J.A. Bondy and U.S.R. Murty (eds.), Academic Press, 1984, pp. 153-173.

[CON85] N. Chiba, K. Onoguchi, and T. Nishizeki, *Drawing planar graphs nicely*, Acta Inform. 22 (1985) 187-201.

[Fa48] I. Fáry, *On straight lines representation of plane graphs*, Acta. Sci. Math. Szeged 11 (1948) 229-233.

[FPP88] H. de Fraysseix, J. Pach, R. Pollack, *Small sets supporting Straight-Line Embeddings of planar graphs*, Proc. 20th Annual Symposium on Theory of Computing, 1988, pp. 426-433.

[FPP90] H. de Fraysseix, J. Pach, R. Pollack, *How to draw a planar graph on a grid*, Combinatorica 10 (1990) 41-51.

[HT74] J. Hopcroft, R.E. Tarjan, *Efficient planarity testing*, Journal of ACM 21 (1974) 549-568.

[Jon93] S. Jones, P. Eades, A. Moran, N. Ward, G. Delott, R. Tamassia, *A note on planar graph drawing algorithms*, Technical Report 216, Dept. of Computer Science, Univ. of Queensland, 1991.

[Ka92] G. Kant, *Drawing planar graphs using the lmc-ordering*, Proc. 33rd Symp. on Foundations of Computer Science, Pittsburgh, 1992, pp. 101-110.

[Ka93] G. Kant, *Algorithms for Drawing Planar Graphs*, Ph.D. Dissertation, Department of Computer Science, University of Utrecht, 1993.

[Sc90] W. Schnyder, *Embedding planar graphs in the grid*, Proc. 1st Annual ACM-SIAM Symp. on Discrete Algorithms, San Francisco, 1990, pp. 138-147.

[ST92] W. Schnyder, W. Trotter, *Convex drawings of planar graphs*, Abstracts of the AMS, 13, 5 (1992), 92T-05-135.

[St51] S. K. Stein, *Convex maps*, Proc. Amer. Math. Soc., 2 (1951) 464-466.

[Tu63] W.T. Tutte, *How to draw a graph*, Proc. London Math. Soc. 13 (1963) 743-768.

[Wa36] K. Wagner, *Bemerkungen zum vierfarbenproblem*, Jahresbericht der Deutschen Mathematiker-Vereinigung, 46 (1936) 26-32.

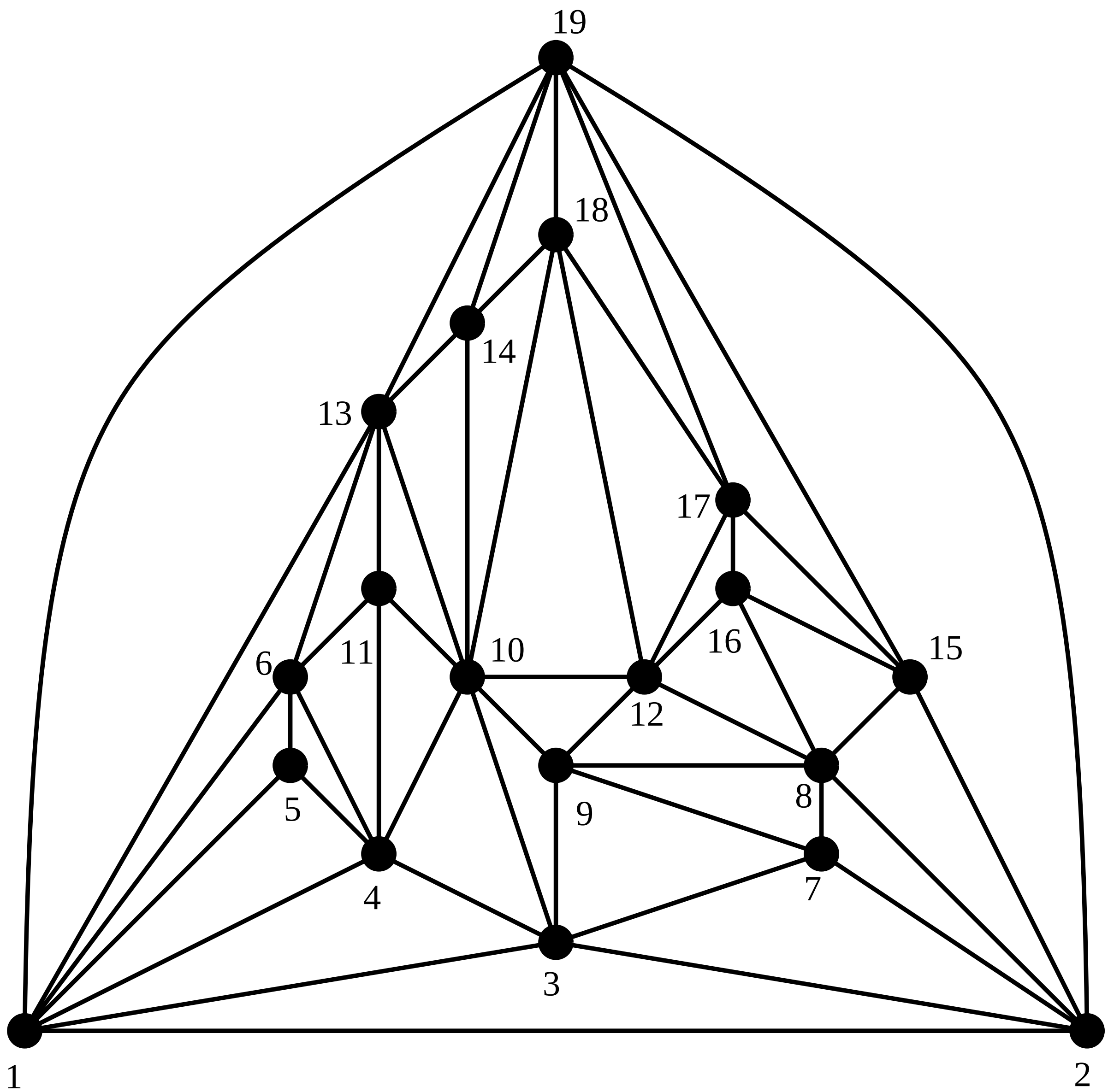

Figure 1: An example of a canonical ordering.

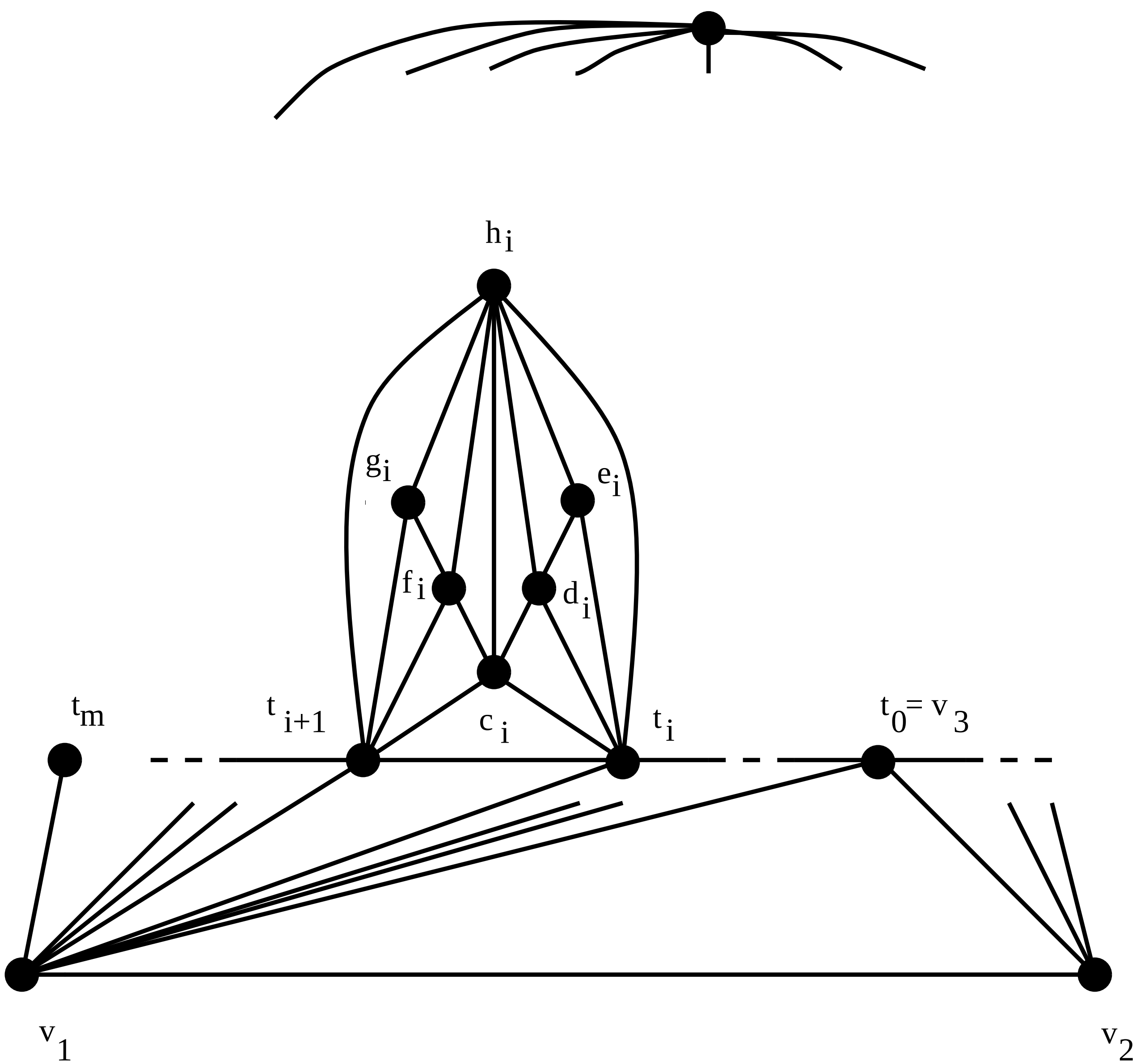

Figure 2: An example showing that replacing slope bound -4 by -3 can increase the width to $5n/7 + O(1)$.