2007－AL－113 (1)
2007／7／3

Orthogonal Drawings for Plane Graphs with Specified Face Areas

Akifumi Kawaguchi and Hiroshi Nagamochi

Department of Applied Mathematics and Physics,
Graduate School of Informatics, Kyoto University, Japan
kawaguti@amp.i.kyoto-u.ac.jp, nag@amp.i.kyoto-u.ac.jp

Abstract. We consider orthogonal drawings of a plane graph G with specified face areas. For a natural number k, a k-gonal drawing of G is an orthogonal drawing such that the outer cycle is drawn as a rectangle and each inner face is drawn as a polygon with at most k corners whose area is equal to the specified value. In this paper, we show that several classes of plane graphs have a k-gonal drawing with bounded k; A slicing graph has a 10-gonal drawing, a rectangular graph has an 18-gonal drawing and a 3-connected plane graph whose maximum degree is 3 has a 34-gonal drawing.

1 Introduction

Graph drawing has important applications in many areas in computer science such as VLSI design, information visualization and so on. Various graphic standards are studied for drawing graphs [5].

Orthogonal drawings, in which every edge is drawn as a sequence of alternate vertical and horizontal segments, have applications in circuit design, geometry and construction. Many aspects have been studied on orthogonal drawings. Studies of an orthogonal drawing with specified face areas have begun recently. For a natural number k, a k-gonal drawing of a graph is an orthogonal drawing such that the outer cycle of the graph is drawn as a rectangle and that each inner face is drawn as a polygon with k corners. Rahman, Miura and Nishizeki [6] proposed an 8-gonal drawing for a special class of plane graphs called a good slicing graph. Recently, de Berg, Mumford and Speckmann [2] proved that a general slicing graph admits a 12-gonal drawing. They also showed that a rectangular graph admits a 20-gonal drawing and a 3-connected plane graph whose maximum degree is 3 admits a 60-gonal drawing.

In this paper, we show that a general slicing graph has a 10-gonal drawing, a rectangular graph has an 18-gonal drawing and a 3-connected plane graph whose maximum degree is 3 has a 34-gonal drawing. Our approach for a general slicing graph is different from that by de Berg et al. [2].

The paper is organized as follows. Section 2 gives some definitions of graphs. Section 3 introduces outlines of the algorithm for a 10-gonal drawing of a slicing graph. Sections 4 and 5 discuss an 18-gonal drawing of a rectangular graph, and an orthogonal drawing of a 3-connected plane graph, respectively. Finally Section 6 concludes.

2 Preliminary

A plane graph is denoted by $G = (V, E, F, c_0)$, where V, E, F and c_0 denote a set of vertices, a set of edges, a set of inner faces and the outer face, respectively. Let $n = |V|, m = |E|$ and $f = |F|$. Since G is a plane graph, $m = O(n)$ and $f = O(n)$ hold. A vertex of degree k is called a *k-degree vertex*. We denote the maximum degree of a graph G by $\Delta(G)$. An *orthogonal drawing* of a plane graph G is a drawing such that each edge $e \in E$ is drawn as an alternate sequence of vertical and horizontal line segments, and any two edges do not intersect except at their common end. It is known [3] that a plane graph G admits an orthogonal drawing if and only if $\Delta(G) \leq 4$. For a natural number k, an orthogonal drawing is called a *k-gonal drawing* if the outer cycle of G is drawn as a rectangle, and each inner facial cycle c_i is drawn as a polygon with at most k corners.

We consider a plane graph G such that the area of each inner face $c_i \in F$ is specified by a real $a_i > 0$. Let A be a set of areas a_i, and we denote a plane graph with the specified face areas by (G, A).

For a plane graph (G, A), we consider an orthogonal drawing such that the area of each face c_i is equal to a_i. Figure 1 illustrates an example of a plane graph with specified face areas, and its 10-gonal drawing.

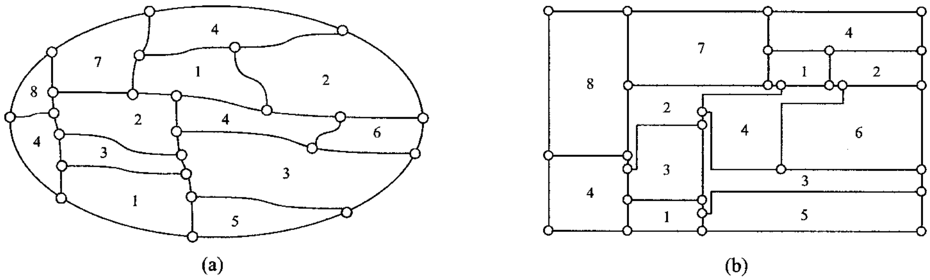

Fig. 1. (a) An example of a plane graph (G, A) with specified areas, where the number in each face represents the area specified for the face; (b) A 10-gonal drawing of (G, A)

Let G be a plane graph that has exactly four 2-degree vertices a, b, c and d in its outer cycle. We call these four vertices a, b, c and d *corner vertices.* The four corners a, b, c and d divide the outer cycle of G into four paths sharing end vertices; the *top path*, the *bottom path*, the *left path* and the *right path.* We call each of these four paths an *unit path.* A path π in G which does not pass through any other outer vertex is called a *vertical* (*horizontal*) path of G if one end of π is on the top (left) path and the other is on the bottom (right) path. Such a path π divides the interior of G into two areas, each of which is enclosed by a cycle and induces a subgraph of G (the subgraph consisting of edges and vertices in the area and the cycle). We say that π *slices* G into these two subgraphs of G.

A *slicing graph* G is a plane graph that is defined recursively as follows; a cycle G of length 4 with a single inner face is a slicing graph, and G has a vertical or horizontal path π such that each of the two subgraphs generated from G by slicing G with π is a slicing graph. Note that $\Delta(G) \leq 4$ for every slicing graph G. A vertical or horizontal path in slicing graph G is called a *slicing path* if two subgraphs generated by slicing G with π are slicing graphs.

A *slicing tree* T is a binary tree which represents a recursive definition of a slicing graph G. Each node u in T corresponds to a subgraph G_u of G. Let u be a non-leaf node in T, and v and w be the left and right child of u, respectively. Then we denote by π_u the slicing path that slices G_u into G_v and G_w; If π_u is vertical (horizontal), then G_v is the left (upper) subgraph of G_u, and G_w is the right (lower) subgraph of G_u. The node u is called a *V-node* if π_u is vertical, and u is called an *H-node* if π_u is horizontal. For a leaf u' of T, the corresponded subgraph $G_{u'}$ has one inner face c_i. Figure 2 illustrates an example of a slicing tree and a slicing graph corresponded to each node of T.

A *rectangular graph* is a plane graph whose outer face and each inner face can be drawn as a rectangle. Note that $\Delta(G) \leq 4$ for every rectangular graph G. A *3-connected plane graph* is a plane graph that remains connected even after removal of any two vertices together with edges incident to them.

In this paper, we show the following results, where a "combined decagon" is defined in the next section.

Theorem 1. *Every slicing graph with specified face areas has a 10-gonal drawing such that each inner face is drawn as a combined decagon. Such a drawing can be found in $O(n)$ time if its slicing tree and four corner vertices on the outer rectangle are given.* □

Theorem 2. *Every rectangular graph with specified face areas has an 18-gonal drawing. Such a drawing can be found in $O(n \log n)$ time if its outer rectangle and its four corner vertices are given.* □

Theorem 3. *Every 3-connected plane graph (G, A) with $\Delta(G) = 3$ has a 34-gonal drawing. Such a drawing can be found in $O(n \log n)$ time.* □

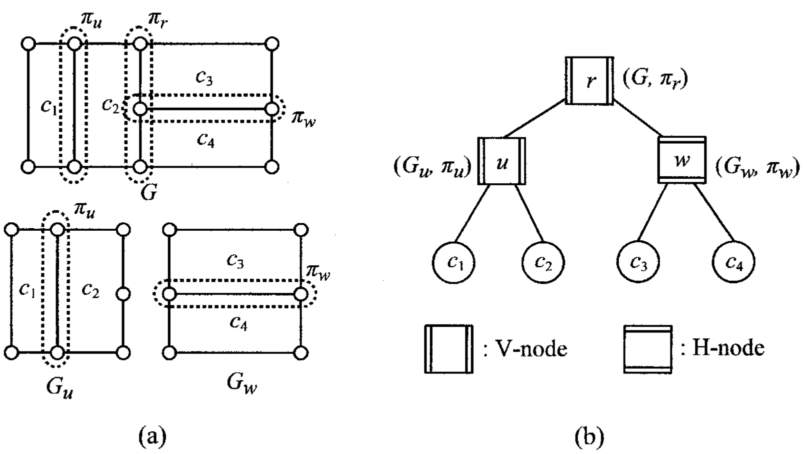

Fig. 2. (a) A slicing graph G and subgraphs G_u and G_w of G; (b) A slicing tree with nodes r, u and w

3 Drawings of Slicing Graphs

By definition, every inner face of a slicing graph can be drawn as a rectangle if we ignore the area constraint. To equalize the area of inner face to the specified value, we need to draw some edges with sequences of several straight-line segments.

We define a *step-line* as an alternate sequence of three vertical and horizontal straight-line segments. A step-line has two corners, which we call *bends*. Based on step-lines, we introduce a polygon called a "combined decagon," which plays a key role to find a 10-gonal drawing of a slicing graph.

3.1 Combined Decagon

We introduce how to draw a cycle with four corner vertices as a k-gon with $4 \leq k \leq 10$. We consider a plane graph G of cycle $G = (\{a, b, c, d\}, \{(a, b), (b, c), (c, d), (d, a)\})$. Note that path ab is the top path, dc is the bottom path, ad is the left path and bc is the right path of G. We call path dab the *top-left path* of G.

We consider a k-gon $(4 \leq k \leq 10)$ in which each path is drawn as a straight-line, a step-line or a pair of these. We use several types of combinations of lines for each of the top-left path, the right path and the bottom path; Five types for the top-left path (Fig. 3), three types for the right path (Fig. 4), and three types for the bottom path (Fig. 5).

We draw cycle (a, b, c, d) by choosing a drawing pattern $A_i\,(i = 1, 2, 3, 4, 5)$ for the top-left path, $B_j\,(j = 1, 2, 3)$ for the right path and $C_k\,(k = 1, 2, 3)$ for the bottom path. Note that the resulting polygon has at most 10 corners. A *combined decagon* P is defined as a polygon such that each unit path of P is drawn as a straight-line or a step-line and at least one of its top and left paths is drawn as a straight-line. Figure 6 illustrates examples of a combined decagon. We may let A_i denote the set of combined decagons such that the top-left path is drawn as a pattern in A_i. Similarly for B_j and C_k.

Let P be a combined decagon. A line segment in the top-left path is called *connectable* if it is incident to corner b or d. Similarly a line segment in the right (bottom) path is called *connectable* if it is incident to corner c. Other line segments are called *unconnectable*. In Figs. 3, 4 and 5, connectable segments are depicted by thick lines.

We denote the connectable segment in the top path, the left path, the right path and the bottom path of P by $\alpha_t(P)$, $\alpha_\ell(P)$, $\alpha_r(P)$ and $\alpha_b(P)$, respectively. An unconnectable line segment in the top-left path is called a *control segment* if it is incident to corner a. Similarly an unconnectable line segment in the right (bottom) path is called a *control segment* if it is incident to corner b (d). In

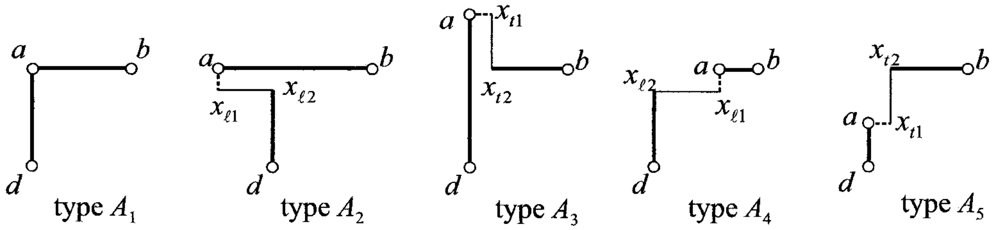

Fig. 3. Five types of drawing pattern for the top-left path *dab*

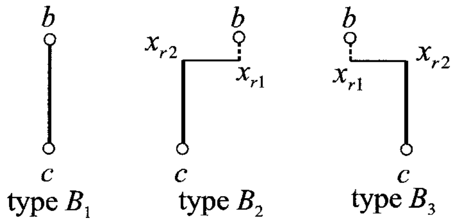

Fig. 4. Three types of drawing pattern for the right path *bc*

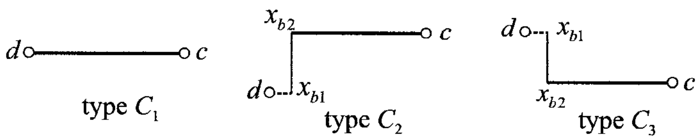

Fig. 5. Three types of drawing pattern for path *dc*

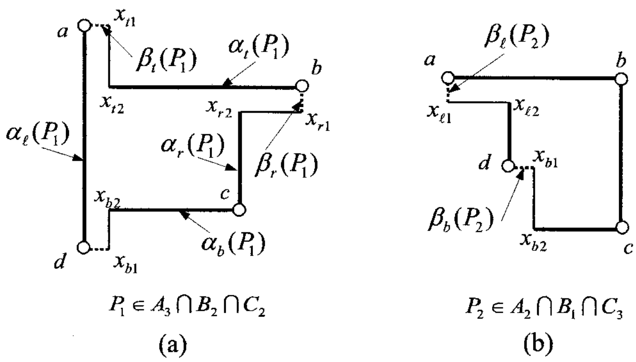

Fig. 6. Illustration of combined decagons P_1 and P_2

Figs. 3, 4 and 5, control segments are depicted by dashed lines. We denote the control segment in the top path, the left path, the right path and the bottom path of P by $\beta_t(P)$, $\beta_\ell(P)$, $\beta_r(P)$ and $\beta_b(P)$, respectively. Let $\beta_{\max}(P)$ be a control segment whose length is maximum in P. A control segment e is called *convex* if both of the two interior angles of P at the both ends of e are 90 degree.

The *width* $w(P)$ of P is the distance from the leftmost vertical segment to the rightmost one, and the *height* $h(P)$ of P is the distance from the top horizontal segment to the bottom one. We denote by xy the *line segment* with end points x and y. We denote the length of segment xy by $|xy|$, the area of a polygon P by $A(P)$, and the sum of the areas specified for all inner faces of a plane graph G by $A(G)$. For a node u of a slicing tree T, we call the following condition the *size condition* of combined decagon P_u; $A(P_u) = A(G_u)$.

3.2 Outline of Algorithm

This subsection outlines our algorithm for slicing graphs with specified areas. The algorithm is a divide-and-conquer based on slicing trees. We are given a slicing graph G with specified areas, its slicing tree T, and rectangle P_r with corner vertices for the outer cycle of G. At this point, the positions of all vertices have not been determined yet. A vertex whose position is determined during the algorithm is called *fixed.* We first draw the outer cycle of G as the specified rectangle P_r, fixing the corner vertices. We then visit all non-leaf nodes in T in preorder and slice P_r recursively to obtain an entire drawing of G. For a node u of T, suppose that the outer cycle of G_u is to be drawn as a combined decagon P_u which satisfies the size condition.

Let u be a V-node. Then G_u has the vertical slicing path π_u, and let z_t and z_b be end vertices of π_u on the top and bottom path of G_u, respectively. First, we try to slice P_u into two combined decagons which satisfy the size condition by choosing a (unique) vertical straight-line segment L as its slicing path π_u (see Fig. 7). If L can be drawn correctly, i.e., the end points z_t and z_b of L are on $\alpha_t(P_u)$ and $\alpha_b(P_u)$, respectively, then we slice P_u by L to obtain two combined decagons. Otherwise, we split P_u by choosing a step-line as its slicing path π_u (see Fig. 7). We can show that the existence of such a suitable step-line π_u is ensured if P_u satisfies the size condition and "boundary condition," which will be described later (the detail of the proof is omitted due to space limitation).

The slicing procedure for H-nodes u is analogous with that for V-nodes. An entire drawing of the given slicing graph G will be constructed by applying the above procedure recursively. We call the algorithm described above Algorithm *Decagonal-Draw.*

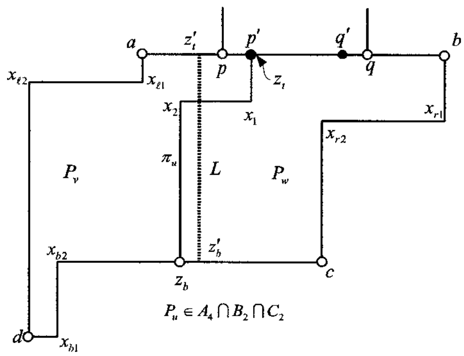

Fig. 7. Vertical slicing of P_u

To ensure that a combined decagon can be chosen as the polygon for the outer facial cycle of each subgraph G_u, the positions of end vertices z_t and z_b of π_u will be decided so that certain conditions are satisfied. We now describe these conditions.

For each node u of T, let f_u^t be the number of inner faces of G_u that are adjacent to the top path of G_u, and f_u^ℓ be the number of inner faces of G_u that are adjacent to the left path of G_u.

Let $a_{\min}$ be the minimum area of all areas for inner faces of G. Let W and H be the width and height of the rectangle specified for the outer facial cycle of a given slicing graph G. We define

$$\lambda = \frac{a_{\min}}{3f \cdot \max(W, H)}. \tag{1}$$

We define some conditions on combined decagon P_u.

A control segment e of P_u is called *(λ, f)-admissible* if one of the followings holds:

e is a convex and vertical segment, and $f_u^t\lambda \leq |e| < f\lambda$,
e is a convex and horizontal segment, and $f_u^\ell\lambda \leq |e| < f\lambda$,
e is a non-convex and vertical segment, and $|e| < (f - f_u^t)\lambda$,
e is a non-convex and horizontal segment, and $|e| < (f - f_u^\ell)\lambda$.

A combined decagon P_u is called *(λ, f)-admissible* if it satisfies the followings.

(a1) $|\alpha_t(P_u)| \geq f_u^t\lambda$,
(a2) $|\alpha_\ell(P_u)| \geq f_u^\ell\lambda$,
(a3) Every control segment of P_u is (λ, f)-admissible,
(a4) If $P_u \in A_1$, then $|\alpha_t(P_u)| \geq (f + f_u^t)\lambda$ or $|\alpha_\ell(P_u)| \geq (f + f_u^\ell)\lambda$,
(a5) If $P_u \in A_2 \cup A_4$, then $|\alpha_\ell(P_u)| + |\beta_\ell(P_u)| \geq (f + f_u^\ell)\lambda$,
(a6) If $P_u \in A_3 \cup A_5$, then $|\alpha_t(P_u)| + |\beta_t(P_u)| \geq (f + f_u^t)\lambda$,
(a7) If $P_u \in A_2 \cap B_3$, then $|\beta_\ell(P_u)| - |\beta_r(P_u)| \geq f_u^t\lambda$,
(a8) If $P_u \in A_3 \cap C_3$, then $|\beta_t(P_u)| - |\beta_b(P_u)| \geq f_u^\ell\lambda$,
(a9) If $P_u \in A_4 \cap B_2$, then $|\beta_r(P_u)| - |\beta_\ell(P_u)| \geq f_u^t\lambda$,
(a10) If $P_u \in A_5 \cap C_2$, then $|\beta_b(P_u)| - |\beta_t(P_u)| \geq f_u^\ell\lambda$.

By (λ, f)-admissibility of P_u, P_u is a simple polygon, and the distance of any pair of vertical line segments or any pair of horizontal line segments of P_u is at least λ.

For a combined decagon P_u, let a be the top-left corner vertex of P_u, b' be a fixed vertex which is the nearest to a on the top path of P_u, and d' be a fixed vertex which is the nearest to a on the left path of P_u. We call the following conditions the *boundary condition* of P_u.

(b1) If there exists fixed vertices on the top path of P_u, then these vertices are on $\alpha_t(P_u)$. The distance of any pair of fixed vertices on $\alpha_t(P_u)$ is at least $f_u^t\lambda$, and the distance from both ends of $\alpha_t(P_u)$ to any fixed vertex is at least $f_u^t\lambda$.
(b2) If there exists fixed vertices on the left path of P_u, then these vertices are on $\alpha_\ell(P_u)$. The distance of any pair of fixed vertices on $\alpha_\ell(P_u)$ is at least $f_u^\ell\lambda$, and the distance from both ends of $\alpha_\ell(P_u)$ to any fixed vertex is at least $f_u^\ell\lambda$.
(b3) If $P_u \in A_1$, then the distance from b' to the left path of P_u is greater than $(f + f_u^t)\lambda$ or the distance from d' to the top path of P_u is greater than $(f + f_u^\ell)\lambda$.
(b4) If $P_u \in A_2 \cup A_4$, then the distance from d' to the top path of P_u is greater than $(f + f_u^\ell)\lambda$.
(b5) If $P_u \in A_3 \cup A_5$, then the distance from b' to the left path of P_u is greater than $(f + f_u^t)\lambda$.

Let $\mathcal{D}$ be the set of all (λ, f)-admissible decagons that satisfy the boundary and size conditions. The following lemma guarantees the correctness of the algorithm, whose proof can be found in the full version of the paper.

Lemma 1. *For a decagon $P_u \in \mathcal{D}$, let P_v and P_w be combined decagons generated by slicing P_u in Decagonal-Draw. Then P_v and P_w belong to $\mathcal{D}$.* □

By this lemma, we can prove the existence of 10-gonal drawings in Theorem 1.

Lemma 2. *Algorithm Decagonal-Draw finds a 10-gonal drawing of a slicing graph G with specified face areas correctly.*

Proof. Let P_r be a rectangle given as the boundary of G. Clearly P_r has no control segments and satisfies the size condition. Hence, P_r satisfies (λ, f)-admissibility. Since P_r satisfies the boundary condition, we have $P_r \in \mathcal{D}$. By Lemma 1, every face of G is drawn as a decagon in $\mathcal{D}$ recursively. Hence, algorithm Decagonal-Draw finds a 10-gonal drawing of a slicing graph G with specified face areas. □

It is not difficult to observe the time complexity of the algorithm.

Lemma 3. *Algorithm Decagonal-Draw can be implemented to run in $O(n)$ time and space.* □

Lemmas 2 and 3 prove Theorem 1.

4 Drawing of Rectangular Graphs

We assume that a given rectangular graph G has no 2-degree vertex except for its corner vertices. To utilize Theorem 1, we convert G into a slicing graph. For this, we slice some inner faces in G by adding new vertices and edges. By the result due to F. d'Amore and P. G. Franciosa [1], we can convert a rectangular graph G into a slicing graph G' by slicing each inner face c of G into at most 4 inner faces so that each unit path of c contains at most 2 unit paths of inner faces generated in the interior of c (see Fig. 8). A cycle c' in G' which corresponds to the facial cycle of c contains at most 8 unit paths

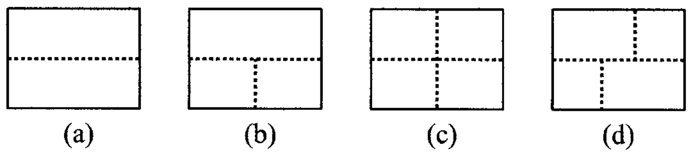

Fig. 8. Four slicing patterns of a rectangle

of inner faces generated in the interior of c'. Figure 9(a) illustrates a inner face c of G sliced into 4 inner faces c_1, c_2, c_3 and c_4. Also, we obtain a slicing tree of the resulting slicing graph from the way of slicing the region by their slicing algorithm. When an inner face c of G is sliced into two inner faces c_1 and c_2, the area of each c_i is set to be a half of the area of c.

We apply Theorem 1 to the resulting slicing graph G' to obtain a 10-gonal drawing D', and remove the edges and vertices added to G from the drawing D'. In the resulting drawing of G, each inner face c is drawn as a polygon with at most 7 step-line segments and some straight-line segments since either the top path or the left path of a combined decagon is always drawn as a straight-line segment (see Fig. 9(b)). Hence c is drawn as a polygon with at most 14 corners of step-lines and 4 corners of the original rectangle. Then we obtain an 18-gonal drawing as described in Theorem 2.

5 Drawing of 3-Connected Plane Graph

By the definition of a 3-connected plane graph, the degree of every vertex is at least three. First, we consider a 3-connected plane graph G with $\Delta(G) = 3$.

An *inner dual graph* G^* for G is a plane graph such that G^* has an embedded vertex in each inner face of G, and G^* has an edge (v, w) if and only if two inner faces c_v and c_w in G that correspond to the vertices v and w in G^* are adjacent. Then edge (v, w) in G^* intersects only one edge of G which separates faces c_v and c_w. Since the degree of each vertex of G is three, G^* is a triangulated plane graph. Note that G^* has no parallel edges since otherwise some two inner faces in G would share more than one edge as the boundary of each of them, contradicting the 3-connectivity of G.

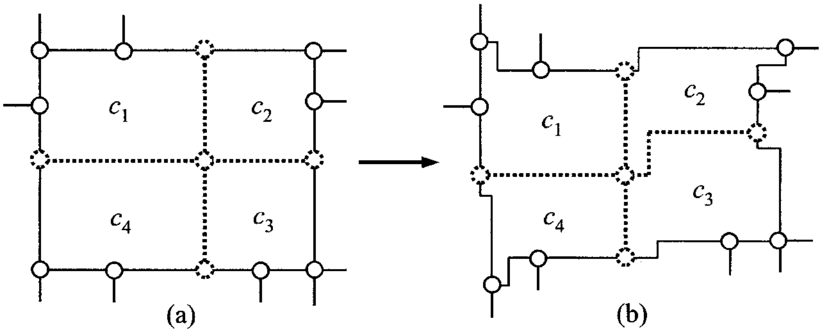

Fig. 9. (a) An inner face c sliced into four inner faces c_1, c_2, c_3 and c_4 in G'; (b) the drawing of c in D'

C.-C. Liao et al. [4] proved that a plane graph G whose inner dual graph G^* is a triangulated plane graph without parallel edges has an orthogonal drawing where outer face of G is drawn as a rectangle and each inner face is drawn as a rectangle, an L-shape or a T-shape. For each inner face c which is drawn as an L-shape or a T-shape, we can slice c into 2 inner faces c_1 and c_2 drawn as rectangles. Hence by the slicing procedure described in the previous section, we can convert a 3-connected plane graph G into a slicing graph G' by slicing each inner face c of G into at most 8 inner faces. Let c', c'_1 and c'_2 be cycles in G' which correspond to the facial cycle of c, c_1 and c_2, respectively. Both c'_1 and c'_2 contains at most 8 unit paths of inner faces generated in their interior, and two of them are always in the interior of c'. Hence c' contains 14 unit paths of inner faces generated in the interior of c'. We apply Theorem 1 to slicing graph G' to obtain a 10-gonal drawing D', and remove the edges and vertices added to G from the drawing D'. In the resulting drawing of G, each inner face c is drawn as a polygon with at most 13 step-line segments and some straight-line segments. Hence c is drawn as a polygon with at most 26 corners of step-lines and at most 8 corners of the original shape (a rectangle, an L-shape or a T-shape). Then we can obtain a 34-gonal drawing of G, as claimed in Theorem 3.

6 Conclusion

In this paper, we showed that every slicing graph has a 10-gonal drawing, every rectangular graph has an 18-gonal drawing, and every 3-connected plane graph whose maximum degree is three has a 34-gonal drawing. We also gave a linear time algorithm to find a 10-gonal drawing for a slicing graph.

It is left as a future work to derive lower bounds on the number k such that every slicing graph admits a k-gonal drawing.

References

1. F. d'Amore and P. G. Franciosa: On the optimal binary plane partition for sets of isothetic rectangles, *Information Proc. Letters*, Vol. 44, pp. 255-259, 1992.
2. M. de Berg, E. Mumford and B. Speckmann: On rectilinear duals for vertex-weighted plane graphs. In *Proc 13th International Symposium on Graph Drawing*, pp. 61-72, 2005.
3. G. Di Battista, P. Eades, R. Tamassia and I. G. Tollis: Graph drawing: Algorithms for the visualization of graphs, Prentice hall, 1999.
4. C.-C. Liao, H.-I. Lu and H.-C. Yen: Floor-planning via orderly spanning trees, In *Proc. 9th International Symposium on Graph Drawing*, pp. 367-377, 2001.
5. T. Nishizeki, K. Miura and Md. S. Rahman: Algorithms for drawing plane graphs, *IEICE Trans. Electron.*, Vol.E87-D, No.2, pp.281-289, 2004.
6. M. S. Rahman, K. Miura and T. Nishizeki: Octagonal drawings of plane graphs with prescribed face areas, In *Graph Theoretic Concepts in Computer Science: 30th International Workshop*, Vol. 3353 of Lecture Notes in Computer Science, pp. 320-331, 2004.