Computational Geometry 24 (2003) 95–114

Computational Geometry
Theory and Applications

www.elsevier.com/locate/comgeo

Planarity-preserving clustering and embedding for large planar graphs [☆]

Christian A. Duncan [a], Michael T. Goodrich [b], Stephen G. Kobourov [c,*]

[a] *Department of Computer Science, University of Miami, USA*
[b] *Department of Information and Computer Science, University of California – Irvine, USA*
[c] *Department of Computer Science, University of Arizona, USA*

Received 1 April 2000; received in revised form 2 January 2001; accepted 15 January 2001

Communicated by P. K. Agarwal and L. Arge

Abstract

In this paper we present a novel approach for cluster-based drawing of large planar graphs that maintains planarity. Our technique works for arbitrary planar graphs and produces a clustering which satisfies the conditions for compound-planarity (c-planarity). Using the clustering, we obtain a representation of the graph as a collection of $O(\log n)$ layers, where each succeeding layer represents the graph in an increasing level of detail. At the same time, the difference between two graphs on neighboring layers of the hierarchy is small, thus preserving the viewer's mental map. The overall running time of the algorithm is $O(n \log n)$, where n is the number of vertices of graph G.
© 2002 Elsevier Science B.V. All rights reserved.

1. Introduction

In the lead article of its recent March/April 1999 issue, *SIAM News* highlighted computations involving large graphs as a grand challenge, and it listed several applications of such computations, particularly in the networking and telecommunications areas. While such application areas typically give rise to non-planar graphs, there are nevertheless several application areas that give rise to large graphs that are planar. Examples of such planar graph applications include computations arising in computational

[☆] A preliminary version of this paper appeared in the Proceedings of the 7th Annual Symposium on Graph Drawing (GD '99), 1999, pp. 186–196. This research was partially supported by NSF under grant CCR-9732300 and ARO under grant DAAH04-96-1-0013.

[*] Corresponding author.
E-mail addresses: duncan@cs.miami.edu (C.A. Duncan), goodrich@ics.uci.edu (M.T. Goodrich), kobourov@cs.arizona.edu (S.G. Kobourov).

cartography and geographic information systems (GIS). In this paper we are therefore concerned with the visualization of large planar graphs.

There are several approaches to the visualization of planar graphs, each of which must address the fact that the resolution of most display technologies (and possibly even the human eye) is simply limited to a few million pixels. Moreover, no matter how many pixels a display technology has, these pixels must display not just the vertices of a graph of interest, but also, and more importantly, the edges connecting these vertices. One approach to drawing large planar graphs is the so-called fish-eye or hyperbolic view approach [9,10,15], which shows a single "cursor" vertex neighborhood in high detail while showing the rest of the graph in decreasing detail (inversely proportional to the distance from the cursor).

A competing approach, however, and the one that is the focus of this paper, is based on representing the graph by a hierarchical clustering in which the graph is represented by a collection of layers, where each succeeding layer represents the graph in a decreasing level of detail. That is, together with G one gives a tree T such that the leaves of T coincide with the vertices of G, and each internal node v of T represents the *cluster* defined by the vertices of G associated with the descendent leaves of v in T. In this case G can be drawn in a "layered" manner, where we draw each cluster on the same layer of T as a region of the plane and connect adjacent clusters by segments. It is desired that each such layer be drawn planar, with no segments intersecting each other or intersecting the boundary of a non-incident cluster region. Thus, the general goal of clustered graph drawing is to preserve the global structure of a graph G by recursively clustering smaller subgraphs of G and drawing these subgraphs as single nodes or filled-in regions in a rendering of G. By grouping vertices together into clusters in this way one can recursively divide a given graph into layers of decreasing detail, which can then be viewed in a top-down fashion.

1.1. Prior related work on clustered graph drawing

If clusters of a graph are given as input along with the graph itself, then several authors give various algorithms for displaying these clusters in two or three dimensions [4,5,7,8,13]. Still, as will often be the case, if clusters of a graph are not given a priori, then various heuristics can be applied for finding clusters using properties such as connectivity, cluster size, geometric proximity, or statistical variation [12,14,18]. If no clusters are given and no special properties are known in advance, Duncan et al. [2] show how to create a hierarchical decomposition and a 3-dimensional drawing for general graphs. However, for planar graphs, it is possible to introduce edge-region crossings, in which edges can cross the cluster regions they are not part of. Even with no edge-edge crossings, the edge-region crossings are a serious drawback to the readability of a drawing.

Eades et al. [5] describe a drawing algorithm that draws a planar graph G, assuming that the clusters of G preserve certain recursive conditions, which they collectively call the *c-planarity* conditions. They show that if G and its clusters satisfy the c-planarity conditions, then one can produce a drawing of G such that each layer of the cluster hierarchy is drawn planar, with each vertex drawn as a convex region and each edge drawn as a straight line segment. This approach allows the graph to be represented by a sequence of drawings of increasing detail. As illustrated by Eades and Feng [4], this hierarchical approach to drawing large graphs can be very effective. However, we are not aware of any previous work for deterministically producing a clustering of an arbitrary planar graph so as to satisfy all the c-planarity conditions.

1.2. Our results

In this paper we describe an algorithm for constructing a clustering of any planar graph so as to satisfy the c-planarity conditions of Eades et al. [5]. Preliminary results can be found in Duncan et al. [3]. Our algorithm runs in $\mathrm{O}(n \log n)$ time, uses $\mathrm{O}(n)$ space, and can be implemented using simple "off-the-shelf" data structures. We also show that the clustering tree T, defined by our algorithm, has the additional property that the number of clusters at layer i of T (i.e., the clusters associated with the nodes of T at height i) is a constant fraction larger than the number of clusters at the next higher layer, $i+1$. Thus, T has $\mathrm{O}(\log n)$ height. This in turn implies faster drawing times when T is used in a clustered graph drawing algorithm, such as that of Eades et al. [5].

This logarithmic height result also implies some nice properties of the clustered drawing itself. For example, had we instead produced a clustering tree T of depth $\Theta(n)$, which is possible if one uses a different clustering algorithm, then we would have a hierarchy that takes an extraordinarily long time to traverse for large planar graphs. At the same time, an $\mathrm{o}(\log n)$ height for T would imply drastic changes between consecutive layers in the hierarchy.

In addition to this logarithmic height result, our algorithm produces a clustering such that the changes between the graphs in consecutive layers of the hierarchy T are "local". In order to preserve the viewer's mental map of the graph when moving from one layer to another, the changes in the graph should be minimal. Given the graph in layer i in T, to obtain the graph associated with the next higher layer $i+1$ in T, we need to group certain sets of vertices together and replace them by new vertices. In this paper, we consider only changes that affect pairs of vertices, so that the tree T is in fact a *binary* tree.

Thus we restrict our clustering operation so as to allow only the combining of two adjacent clusters, which is an operation typically referred to as an *edge contraction.* Through a sequence of such edge contractions, we obtain the layer graphs $G_0, G_1, \ldots, G_k$, where $G_0 = G$ and G_k is a singleton graph. If the changes necessary to obtain layer $i+1$ from layer i are to be local, then the following three *locality conditions for edge contraction* must be met:

(1) A vertex can participate in at most one edge contraction.
(2) Changes in the drawing of the graph that result from the contraction of an edge (u, v) should only affect edges with endpoint u or v.
(3) A contraction of edge (u, v) results in the creation of vertex w. The placement of w in the drawing should be "close" to the edge (u, v). Optimally, we would like that w lie along the line segment defined by (u, v).

We provide a clustering method that satisfies the above locality conditions. One of the main challenges in creating the layers in a cluster hierarchy of a planar graph is to define clusters and the drawing algorithm associated with G's clustering in such a way that no edge crossings are introduced in the drawing of each layer. We provide a drawing algorithm which makes use of our clustering method to produce a drawing that has neither edge-edge crossings nor edge-region crossings. In addition, we show that one can use our clustering as input to the clustered planar graph drawing algorithm of Eades et al. [5].

2. Hierarchical embedding of planar graphs

A graph is *maximally planar* if it is planar and adding any new edge results in a non-planar graph. Maximally planar graphs are also called *fully triangulated* as every face of a maximally planar graph (including the exterior face) is a triangle. Let us assume, without loss of generality, that all the graphs that we are dealing with are maximally planar. If a particular graph is not maximally planar then we can fully triangulate it. Let $G = (V, E)$ be a maximally planar graph, where $|V| = n$. $V(G)$ and $E(G)$ as usual refer to the set of G's vertices and edges, respectively and the degree of a vertex v in graph G is $d_G(v)$. Let $l_G(f_i)$ be the length of a face f_i in G, where by the length of the face we mean the number of vertices on that face. Further, Δuvw will refer to the triangle defined by vertices u, v, w and by the edges (u, v), (v, w), (w, u).

Similar to the definition in [4] we define the *clustered graph* $C = (G, T)$ to be the graph G and a tree T such that the vertices of G coincide with the leaves of T; see Fig. 1. An internal node of T represents a cluster, which consists of all the vertices in its subtree. All the nodes of T at a given height i represent the clusters of that level. A *view at level* i, $G_i = (V(G_i)_J, E(G_i))$, consists of the nodes of height i in T and a set of representative edges. The edge (u, v) is in $E(G_i)$ if there exists an edge between a and b in G, where a is in the subtree of u and b is in the subtree of v. Each node $u \in T$ has an associated region, corresponding to the partition given by T.

We create the graphs G_i in a bottom-up fashion, starting with $G_0 = G$ and going all the way up to G_k, where $k = \mathtt{height}(T)$. We obtain G_{i+1} from G_i by contracting a carefully chosen set of edges of G_i in a certain order. The z-coordinate of a vertex $v \in V(G_i)$ is equal to i, that is, all the vertices in G_i are embedded in the plane given by $z = i$. The edges of T are defined by the edge contractions. More precisely, if $(u, v) \in E(G_i)$ is contracted to a vertex $w \in G_{i+1}$, then edges (w, u) and (w, v) are added to T.

The problem of embedding planar graphs with straight lines and no crossings is well studied [1,6,16, 17,19]. Embedding clustered graphs without crossings poses additional difficulties. To embed the layers, we reverse the sequence of graph contractions: we start with embedding of G_k (which has only one vertex). To obtain an embedding for G_{i-1} from an embedding for G_i we consider the set of edges of G_{i-1} whose contraction resulted in G_i. We then reverse the process by carefully expanding and embedding one edge from that set at a time. Throughout this process we maintain the three locality conditions for edge expansions/contractions.

2.1. Edge contraction and separating triangles

Contracting an edge is a standard operation on planar graphs; see [11]. We say that an edge $e = (u, v)$ of G is *contracted* when its endpoints, u and v, are replaced by a new vertex w such that all resulting multiple edges are removed. Formally,

- edge $e = (u, v)$ is removed from G.
- $\forall x \in V(G)$: (x, u) and $(x, v) \in E(G)$, we remove edges (x, u) and (x, v) and add edge (x, w); see Fig. 2.
- $\forall x \in V(G)$: $(x, u) \in E(G)$ and $(x, v) \notin E(G)$ (or $(x, u) \notin E(G)$ and $(x, v) \in E(G)$), we remove edge (x, u) (or (x, v)) and add edge (x, w).

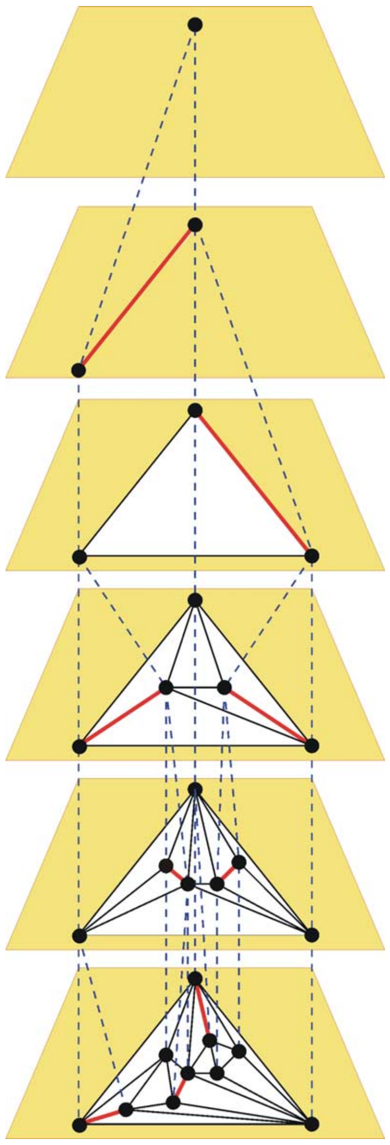

Fig. 1. A clustered graph $C = (G, T)$. The underlying graph G is shown on the lowest level. The tree T is represented by the dashed edges that connect the layers. The thick edges on each level are contracted to vertices in the next layer.

Edge contractions always produce parallel edges. If parallel edges are not eliminated, further edge contractions result in more parallel edges and even self loops. Ideally, we would like to perform edge contractions in a straight-line drawing that can be continuously animated so as to preserve planarity. Furthermore, so as to preserve the viewer's mental map, we prefer that only the endpoints of the contracted edge move, resulting in only minimal changes in the drawing.

It is well-known that contracting an edge in a planar graph results in a planar graph [11]. Note that this does not imply that contracting an edge in a straight line planar drawing of a graph results in a straight

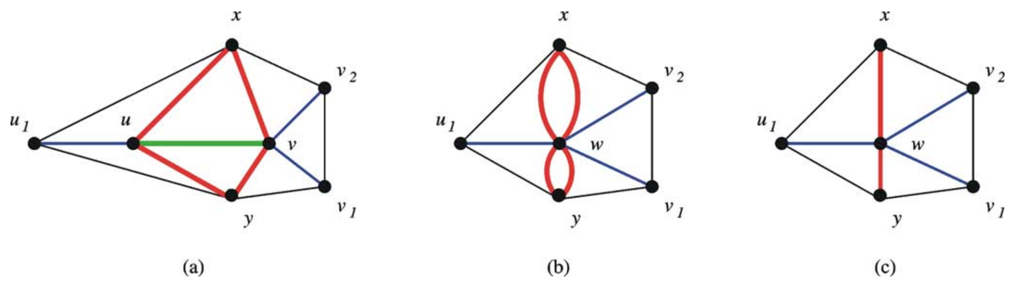

Fig. 2. Contracting an edge (u, v) down to a vertex w subject to the locality conditions.

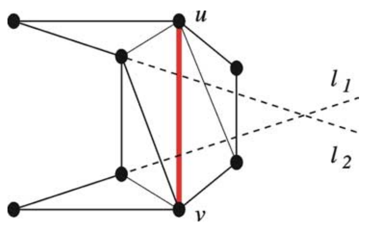

Fig. 3. A subgraph of an embedded fully triangulated graph. Edge (u, v) cannot be contracted without introducing a crossing, if we are to keep all other vertices fixed. Note that the resulting vertex should lie below line l_1 and above line l_2.

line planar drawing! More precisely, consider a straight-line planar drawing of a graph and an edge to be contracted. Suppose we are not allowed to move any other vertices in the drawing except the two involved. Then there exist drawings in which the contraction of some such edge introduces a crossing. We show this with an example in Fig. 3.

2.2. Simple edges and separating triangles

A well-known result in the theory of planar graphs states that if a graph G is maximally planar, then it has a unique combinatorial embedding on a sphere. This implies that the clockwise (counterclockwise) order of the neighbors around every vertex is the same in every drawing of G on the sphere. As a result, all embeddings of G in the plane are the same, up to the choice of the outer face of G. It is important to note that the statement is not true for general planar graphs, hence, maximal planarity is a necessary condition.

We have seen one of the problems that occur when an edge in an embedded graph is contracted. Another problem can occur even if we do not have a fixed embedding. When the contracted edge is a part of a separating triangle, the resulting graph is not fully triangulated and in fact may have many different embeddings. We call a triangle in G a *separating triangle* if the removal of its vertices and their adjacent edges disconnects G; see Fig. 4.

Thus, we can divide the edges of G into two categories depending on the effect their contraction has on the resulting graph. We say that an edge is *simple* if it is not a part of a separating triangle. Edges that are part of separating triangles we call *non-simple.* Let G' be the graph obtained from G by removing all non-simple edges. We refer to G' as the *simple skeleton* or just the *skeleton* of G. Non-simple edges present problems when contracted, so we will be contracting only simple edges, for their contraction can be continuously animated while preserving planarity using straight lines. Moreover, eliminating the

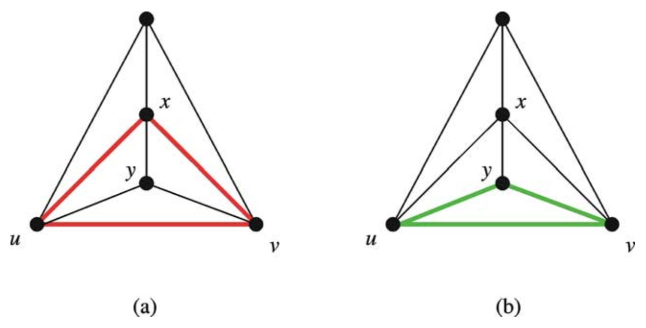

Fig. 4. Triangle Δuvx in part (a) is separating triangle. Contrast that with the non-separating triangle Δuvy in part (b). Note that although Δuvy is not a separating triangle, edge (u, v) belongs to a separating triangle.

parallel edges after contracting a simple edge in a maximally planar graph results in a maximally planar graph, as the following lemma shows:

Lemma 1. *Contracting a simple edge in a maximally planar graph results in a maximally planar graph.*

Proof. Suppose $e = (u, v)$ does not belong to any separating triangle. Since every edge is a part of two triangles, u and v have at least two neighbors in common. Suppose there are three or more neighbors in common. Then e is a part of three or more triangles. One of those triangles must be a separating triangle. Hence u and v have exactly two neighbors in common.

We also need to show that after the contraction of e the remaining graph is still fully triangulated. Let x and y be the two neighbors of u and v; see Fig. 2. When we contract e we replace the pair of vertices (u, v) with a new node w. In the process we remove edges (u, x), (u, y), (v, x), (v, y), (u, v) and replace them with edges (w, x), (w, y). Since contracting an edge in a planar graph results in a planar graph, we know that the new graph is still planar. Also, if the original graph had n vertices and $3n - 6$ edges, the new graph has $n - 1$ vertices and $3n - 6 - 5 + 2 = 3(n - 1) - 6$ edges. Thus the resulting graph is maximally planar.

Note that a similar claim was stated in [16, Lemma 4.1]. □

We showed that contracting a simple edge in a maximally planar graph results in a maximally planar graph. This property is not true for non-simple edges. Suppose we were to contract edge (u, v) in Fig. 5 and replace it with a new node w. The resulting graph has more than one different embeddings. Note that the difference in the graphs on Fig. 5 (b) and (c) is not just in the selection of the outer face: the order of the neighboring vertices around vertices x and w is different.

We first show that if Δuvw is a separating triangle in G then there exist two disjoint paths between u and v in the skeleton G' of G. Next we show that G' is biconnected.

Lemma 2. *Let G and G' be a maximally planar graph and its skeleton. If Δuvw is a separating triangle in G, then there exist two disjoint paths between u and v in G' using only vertices from inside the triangle Δuvw for one and outside the triangle for the other.*

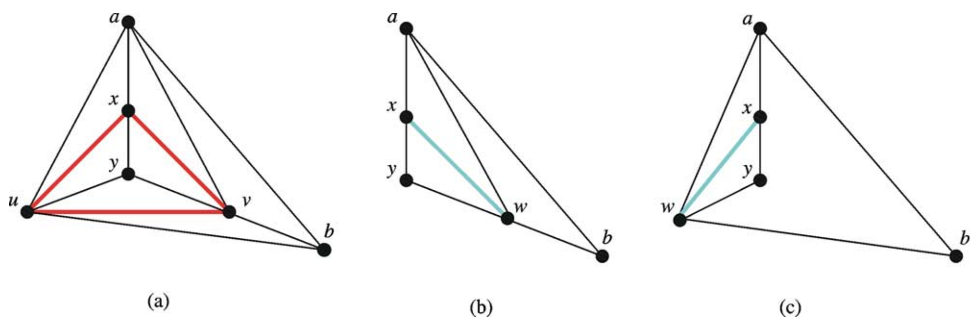

Fig. 5. The graph in part (a) contains a separating triangle Δuvx. If we contract edge (u, v) and replace it with a node w we can get combinatorially different graphs, depending on where we place w. For example, the order of the neighboring vertices around vertex x in (b) and in (c) is different.

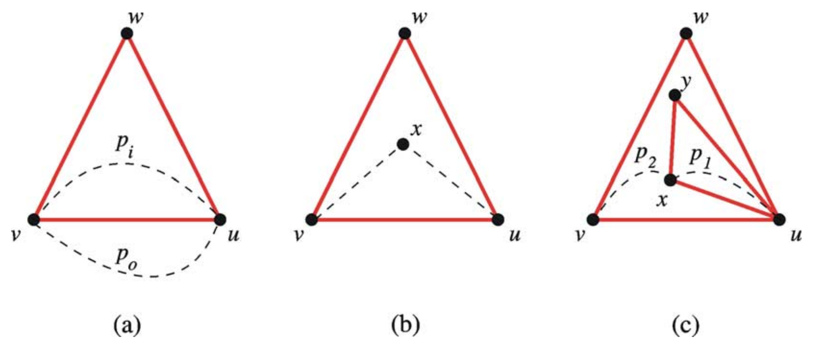

Fig. 6. If Δuvw is a separating triangle then there exist two disjoint paths p_i and p_o connecting u and v. (b) If $|V_i| = 1$ then $p_i = uxv$ for some vertex $x \in V_i$. (c) If $|V_i| > 1$ then p_i is made of two paths p_1 and p_2.

Proof. Let Δuvw be a separating triangle in G. Every separating triangle divides the vertices of G into two classes, V_i (vertices inside the triangle) and V_o (vertices outside the triangle), such that $V_i \cup V_o \cup \{u, v, w\} = V(G)$ and $V_i \cap V_o = \emptyset$. We claim that there exist paths p_i and p_o in G' which connect u and v such that p_i uses only vertices in V_i and p_o uses only vertices in V_o; see Fig. 6(a). We first prove the existence of p_i by induction on the number of vertices in V_i. Note that since Δuvw is a separating triangle, we have $|V_i| \geqslant 1$. Assume that $|V_i| = 1$. Then $V_i = \{x\}$, for some vertex x, and since G is fully triangulated, edges (x, u), (x, v) exist and are simple which implies that they belong to G'. Then $p_i = uxv$; see Fig. 6(b).

Suppose the claim holds for $|V_i| \leqslant k$ and consider the case $|V_i| = k + 1$. Let x be the first vertex before v in the counterclockwise order around u. If (u, x) and (x, v) are simple edges then $p_i = uxv$. The other possibility is that one or both are non-simple. If (u, x) is a non-simple edge then it is a part of a separating triangle. Let y be the third vertex of a separating triangle which uses u and x; see Fig. 6(c).

We claim that $y \notin V_o$. Otherwise the edge (x, y) must cross one of the edges of Δuvw which is a contradiction to planarity. Thus, x and y do not belong to V_o. Consider Δuxy. Since it is inside Δuvw and does not contain v it contains at most k vertices. Then the inductive hypothesis applies and there exists a path p_1 which connects u and x and uses only vertices in V_i.

If (x, v) is a simple edge, then the path p_1 together with edge (x, v) is the path p_1 we are looking for. Otherwise (x, v) is a non-simple edge. We just showed that if (u, x) is a non-simple edge, there exists a path p_i which connects u to x using only simple edges with endpoints in V_i. By a similar argument, if (v, x) is a non-simple edge, there exists a path p_2 which connects x to v using only simple edges with endpoints in V_i. Then the two paths p_1 and p_2 form the path p_i from u to v of only simple edges with endpoints in V_i.

The above argument shows that there exits a path p_i which connects u and v using only simple edges with endpoints in V_i. A similar argument shows that there exists path p_o which connects u and v using only simple edges with endpoints in V_o. □

Lemma 3. *If G is a maximally planar graph and G' is its skeleton, then G' is biconnected.*

Proof. Let us first prove that G' is connected. Since G is fully triangulated, there exists a path between any two vertices u and v. Let p be one such path. It is possible that some of the edges along that path were non-simple and so the same path does not necessarily exist in G'. However, for any non-simple edge (x_i, x_{i+1}) along that path we can find a path p_i which uses only simple edges (from Lemma 2). We can replace all non-simple edges with paths of simple edges to get a path in G' which connects u and v.

Suppose that G' is not biconnected. Since we know that G' is connected, if it is not biconnected there must exist a cut vertex v (a vertex whose removal disconnects G'). Since v is a cut vertex, in the counterclockwise traversal about v, there must exist two neighboring vertices u and w which are not connected in $G' - v$. Since (u, w) is therefore a non-simple edge in G, from Lemma 2, we know that there exist two disjoint paths between u and w in G'. At least one of these paths cannot use v. Therefore, there must exist a path from u to w in $G' - v'$. Since u and w are not connected in $G' - v$, we have a contradiction, and G' must be biconnected. □

The next two corollaries follow trivially from Lemma 3.

Corollary 1. *Let G be a maximally planar graph and G' be the graph obtained from G by removing all the non-simple edges. For every vertex $v \in G'$, we have $d_{G'}(v) \geqslant 2$, where $d_{G'}(v)$ is the degree of v in G'.*

Corollary 2. *If a vertex v is a part of a separating triangle Δvab in a maximally planar graph G, then there exist at least two simple edges (v, x_i) and (v, y_j) adjacent to it, such that $x_i \in V_i$ and $y_j \in V_o$, where V_i and V_o are the inside and outside vertices of G with respect to Δuab.*

3. Finding a simple matching in a maximally planar graph

In this section we show that any maximal matching that uses only simple edges contains a constant fraction of all the edges in G, provided G is maximally planar. Next we show how to find a matching that can be used to contract the graph so that the resulting graph is maximally planar. Furthermore, if the size of that matching is $O(n)$, then after repeating this process $O(\log n)$ times we are left with only a constant number of vertices. Thus, we need to show that we can construct a maximal matching with $O(n)$ edges such that their contraction results in a maximally planar graph.

Let G' be the skeleton of G. Recall that we construct G' from G by removing all the non-simple edges. We start by showing that any maximal matching in G' contains at least $n/12$ edges. To prove this claim we construct a maximal matching in G' and consider faces of different lengths. Recall that the length of

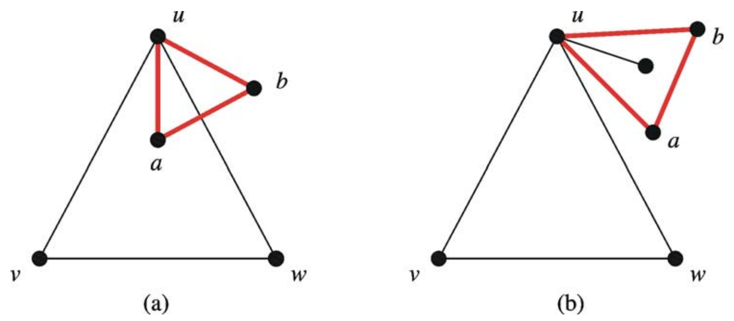

Fig. 7. Let Δuab be a separating triangle, (a) If a and b are on opposite sides with respect to Δuvw there must be an edge crossing. (b) If a and b are on the same side, then there exists a third simple edge incident to u.

a face refers to the number of vertices on that face. We break the faces of G' into three classes, A, B, C, which contain faces of length 3, faces of length 4, and faces of length 5 or more, respectively. We then count the number of unmatched vertices in faces of the different classes. Finally, when we factor in over-counting we show that any maximal matching must contain at least $n/12$ edges.

Lemma 4. *Let G and G' be a maximally planar graph and its skeleton. If f_i is a face of length* 3 ($f_i \in A$), *then there is at most one unmatched vertex in f_i and this vertex has degree at least* 3.

Proof. Let M be the set of vertices that are incident to a matching edge in an arbitrary maximal matching of G'. We first show that there can be at most one unmatched vertex in a triangular face of G'. Consider such a triangular face in G' and suppose there are two or more unmatched vertices u, v on that face. Then $u, v \notin M$ which implies that edge $(u, v) \in G'$ on that face can be safely added to the matching (recall that all edges in G' are simple and if both endpoints are unmatched then it is an eligible candidate for the matching). Adding edge (u, v) results in a larger matching which contradicts the maximality of the existing matching. Thus there can be at most one unmatched vertex in a triangular face of G'.

We now show that the degree of a vertex on a face in A is at least 3. Let u be an unmatched vertex in a triangular face, defined by u, v, w in G'. We claim that $d_{G'}(u) \geqslant 3$ in G'. Clearly, the degree of u is at least 2 (from Corollary 1). Suppose $d_{G'}(u) = 2$. We know that $d_G(u) \geqslant 3$ since G is fully triangulated. Then u participates in at least one separating triangle in G. There exist at least two vertices, a and b, such that Δuab is a separating triangle in G.

Note that a and b must be on the same side of the triangle, since otherwise edge (a, b) would cross one of the edges of Δuvw; see Fig. 7. Since G' is connected, it is not possible that both V_i and V_o are non-empty (otherwise Δuvw is a separating triangle). Without loss of generality, let $V_i = \emptyset$. Then $a, b \in V_o$ and there must exist a simple edge adjacent to u and inside Δuab (from Corollary 2). Thus $d_{G'}(u) \geqslant 3$. □

Lemma 5. *Let G and G' be a maximally planar graph and its skeleton. If f_i is a face of length* 4 *or more* ($f_i \in B$ *or* $f_i \in C$), *then there exist at most $l(f_i)/2$ unmatched vertices in f_i.*

Proof. Consider a maximal matching in G' and a face of length more than 3 in G'. Suppose there are more than $l(f_i)/2$ unmatched vertices on that face. Since G' is biconnected, at least two of them must be adjacent. The edge that connects them is simple, so by adding the edge to the maximal matching its

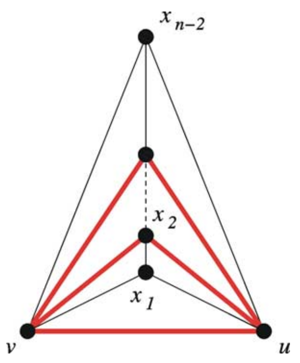

Fig. 8. Graphs in the class $\mathcal{H}$ consist of $n-2$ nested triangles, all of which share one edge.

size is increased by one. This would contradict the maximality of the matching. Thus there are at most $l(f_1)/2$ unmatched vertices. □

For the unmatched vertices on faces in A we were able to show that they have degree at least 3 in G'. This is not necessarily true for the unmatched vertices on faces in B or C. We are able to show, however, that if a pair of unmatched vertices on a face in B have degrees 2 in the skeleton then G belongs to a special class of graphs $\mathcal{H}$. The class of nested triangles $\mathcal{H}$ is defined as the class of maximally planar graphs in which there exist two adjacent vertices, u, v such that every other vertex in the graph is adjacent to both u and v; see Fig. 8. If $G \notin \mathcal{H}$ then for any face $f_i \in B$, at most one vertex on that face has degree 2.

Lemma 6. *If H is a maximally planar graph in the class $\mathcal{H}$, then any maximal matching that uses only simple edges contains $n/12$ or more edges.*

Proof. Let H be a graph in the $\mathcal{H}$ class such that the vertices of H are $V(H) = u, v, x_1, x_2, \ldots, x_{n-2}$ and the edges of H are $E(H) = (u, v) \cup E_1 \cup E_2$, where $E_1 = \{(u, x_i), (v, x_i),$ for $1 \leqslant i \leqslant n-2\}$ and $E_2 = \{(x_{i-1}, x_i),$ for $1 < i \leqslant n-2\}$. Let the counterclockwise order of the vertices around v be $u, x_1, x_2, \ldots, x_{n-2}$; see Fig. 8.

Let H' be the graph obtained from H by removing all non-simple edges. Then $E(H') = \{(u, x_1), (u, x_{n-2}), (v, x_1), (v, x_{n-2}), (x_{i-1}, x_i),\ \forall i\colon\ 1 < i \leqslant n-2\}$. The cycle $u, x_1, x_2, \ldots, x_{n-2}, u$ has $n-1$ edges so any maximal matching must use at least $(n-1)/2$ edges. Since $(n-1)/2 > n/12$ for all $n > 1$, this completes the proof of the lemma. □

Lemma 7. *Let G and G' be a maximally planar graph and its skeleton. If there exists a face of length four in G' with more than one vertex of degree 2, then $G \in \mathcal{H}$.*

Proof. Let f be a face of length four in G' with more than one vertex of degree 2 on that face. Assume that $G \notin \mathcal{H}$. Let the four vertices of f be a, b, c, d; see Fig. 9. Since G is fully triangulated, there exists an edge connecting two opposite vertices. Without loss of generality, let b and d be connected with an edge in G. But the edge (b, d) is not in G' so it must be on a separating triangle in G. Then there exists a vertex u such that Δbdu is a separating triangle in G. Since the edges (a, b) and (b, c) are in G', vertex u cannot be a or c.

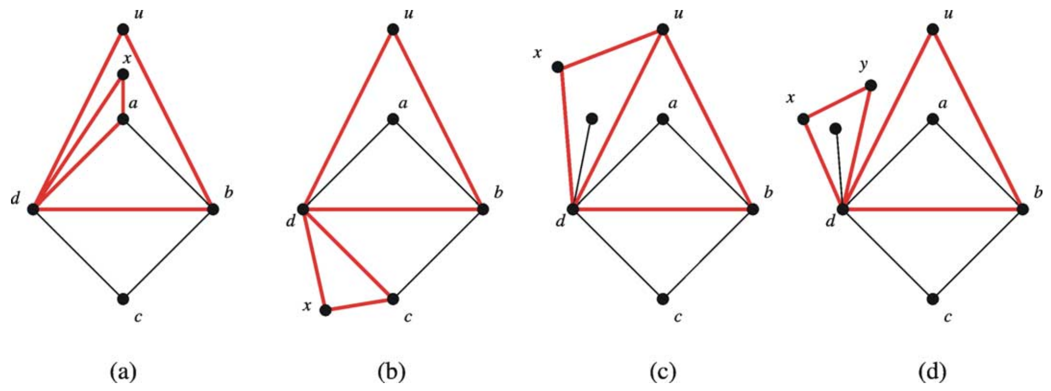

Fig. 9. A 4-cycle in G' defined by vertices a, b, c, d. If both vertices b and d are of degree 2, then $G \in \mathcal{H}$.

Let us now consider the possible vertices of degree 2 in f. Assume $d_{G'}(a) = 2$. Since G is fully triangulated, $d_G(a) \geqslant 3$. Since G' is connected and a, b, c, d is a face in G', there cannot be any non-simple edges in the counterclockwise order between d and b in G. If a has degree 2 in G' then one incident edge of a is not simple, hence there must exist a separating triangle which uses a outside of Δabd. But this implies that there exists at least one more simple edge with a as one of its endpoints (from Corollary 2). Hence, $d_{G'}(a) \geqslant 3$, which is a contradiction.

Similarly, we can show that $d_{G'}(c) \geqslant 3$. Since there are at least two vertices of degree two on f, it must be that $d_{G'}(b) = 2$ and $d_{G'}(d) = 2$. Let x be any neighbor of d other than a, b, c, d, u. Since $d_{G'}(d) = 2$, $(d, x) \notin E(G')$. Therefore, (d, x) is an edge of a separating triangle Δdxy. Consider the different possibilities for y:

(a) Assume $y = a$ then (a, d) must be a non-simple edge. But $(a, d) \in G'$, and so $y \neq a$; see Fig. 9(a).
(b) Assume $y = c$ then (c, d) must be a non-simple edge. But $(c, d) \in G'$, and so $y \neq c$; see Fig. 9(b).
(c) Assume $y = u$. Then either u follows x in the clockwise order around d or the other way around; see Fig. 9(c). Without loss of generality, let u follow x. From Corollary 2 there must exist a simple edge (d, z) between (d, u) and (d, x) in the counterclockwise order of the edges around d. But then $d_{G'}(d) \geqslant 3$, which is a contradiction. Thus $y \neq u$.
(d) Assume $y \notin \{a, b, c, d, u, x\}$. Similar to the previous case, there exists a separating triangle Δdxy such that a and c are on the same side of Δdxy; see Fig. 9(d). Then there is at least one more simple edge (d, z) where $z \notin \{a, b, c, d, u, x, y\}$ implying that $d_{G'}(d) \geqslant 3$, which is a contradiction.

The only possibility left is $y = b$, which implies that every neighbor of d is also a neighbor of b. A similar argument shows that every neighbor of b must be a neighbor of d.

Since $G \notin \mathcal{H}$, there must exist at least one vertex v which is not a neighbor of b or d. In particular, since G is connected, let us look at the vertex closest to d or b in G which is not a neighbor of b or d. Without loss of generality let v be closer to b. Let p be the shortest path from b to v. Since b is not a neighbor of v, the path has at least three vertices. In fact, since v is the closest non-neighbor to b or d, p has exactly three vertices, b, u, v, where u is some neighbor of b and v. Recall that from the arguments above u must also be a neighbor of d. In the traversal about u from b to v to d, there may be multiple vertices. Since the path lengths are equivalent, without loss of generality let v be the vertex closest to d

in the traversal. Since G is fully triangulated, v must also be a neighbor of d and hence a neighbor of b. However, this contradicts our choice of v. Therefore, G must be in $\mathcal{H}$. $\square$

We have shown that if G' has a face of length four with more than two vertices of degree 2, then $G \in \mathcal{H}$ and hence any maximal matching in G' contains at least $n/12$ edges (from Lemma 6). Finally we show that the same result holds for all maximally planar graphs.

Theorem 1. *Let G be a maximally planar graph and let M be the set of matched vertices in a maximal matching which uses only simple edges. Then $|M| \geqslant n/6$, where n is the number of vertices of G.*

Proof. If $G \in \mathcal{H}$, from Lemma 6 we know that the theorem holds. Then suppose $G \notin \mathcal{H}$. Let M and U be the set of matched and unmatched vertices, respectively, in any maximal matching in G'. In order to count the number of matched edges in G, we will count the number of unmatched vertices. Let us define a function

$$u(v) = \begin{cases} 1 & \text{if } v \notin M \ (v \text{ is unmatched}), \\ 0 & \text{otherwise.} \end{cases}$$

Note that $\sum_{v \in G} u(v) = |U| = n - |M|$. We are going to count the number of unmatched vertices in each face, so we want to make sure we know how much we over-count. With this in mind, let $u_i(v) = u(v)/d(v)$ if v is on a face f_i. Then $\sum_{f_i \in F} u_i(v) = u(v)$ and $u(f_i) = \sum_{v \in f_i} u_i(v) = \sum_{v \in f_i} u(v)/d(v)$. Recall that A, B, C are the sets of faces of length 3, 4 and 5 or more in G', respectively. Then

$$|U| = \sum_{v \in V} u(v) = \sum_{f_i \in F} u(f_i) = \sum_{f_i \in A} u(f_i) + \sum_{f_i \in B} u(f_i) + \sum_{f_i \in C} u(f_i).$$

Next we look at each of the three classes of faces:

(a) Consider the faces of length 3 in G'. Recall from Lemma 4 that if $f_i \in A$ then there is at most one unmatched vertex in f_i and it has degree at least 3. Hence,

$$\sum_{f_i \in A} u(f_i) \leqslant |A|/3.$$

(b) Consider the faces of length 4 in G'. Recall we assumed that $G \notin \mathcal{H}$, and then any face in B has at most one vertex of degree 2, and there are no more than 2 unmatched vertices per face (from Lemma 7). It is possible then to have one of the unmatched vertices of degree 2 but if there is a second unmatched vertex it has degree at least 3. Then

$$\sum_{f_i \in B} u(f_i) \leqslant (1/2 + 1/3)|B| = 5|B|/6.$$

(c) Finally, consider the faces of length 5 or more in G'. From Lemma 5 we know that if $f_i \in C$, then there are at most $l(f_i)/2$ unmatched vertices in f_i. Putting this together with the fact that every vertex in every face of G' has degree at least 2 gives us that

$$\sum_{f_i \in C} u(f_i) \leqslant \sum_{f_i \in C} \frac{l(f_i)}{4}.$$

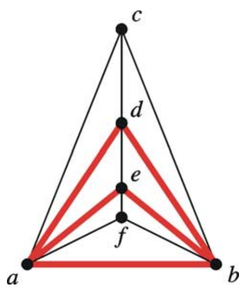

Fig. 10. Consider a graph G, with its simple and non-simple edges drawn with thin and thick lines, respectively. Since G' has no faces of length 3, $|A| = 0$. G' has one face of length 4, defined by vertices a, c, b, f and hence $|B| = 1$. G' has two faces of length 5 or more, defined by vertices a, c, d, e, f and b, c, d, e, f, respectively, hence $|C| = 2$. G has six faces of length 3 which end up as part of faces of length 5 or more in G'. These are defined by the triangles Δafe, Δaed, Δadc, Δbef, Δbde, Δbcd. Hence $|D| = 6$.

We next calculate $\sum_{f_i \in C} \frac{l(f_i)}{4}$. Let D be the set of faces in G which end up as a part of a face of length 5 or more in G'. From Euler's formula for G, we have $|A| + 2|B| + |D| = 2n - 4$, since every face of length 4 in G' (recall that B is the set of faces of length 4 in G') corresponds to exactly two faces in G. We can then express D in terms of A, B,

$$|D| = 2n - 4 - |A| - 2|B|. \tag{1}$$

Sets A, B, C, D are illustrated with an example in Fig. 10.

Consider a face f_i of length $l(f_i) \geqslant 3$. There are exactly $l(f_i) - 2$ triangular faces in G which merged to form f_i. Then

$$|D| = \sum_{f_i \in C} \bigl(l(f_i) - 2\bigr) = \sum_{f_i \in C} l(f_i) - 2|C|.$$

From the last equation we can express the sum of the lengths in terms of C, D: $\sum_{f_i \in C} l(f_i) = |D| + 2|C|$. Putting the last two results together we get that $\sum_{f_i \in C} l(f_i) = 2n - 4 - |A| - 2|B| + 2|C|$ and we obtain the bound

$$\sum_{f_i \in C} u(f_i) \leqslant \frac{n}{2} - 1 - \frac{|A|}{4} - \frac{|B|}{2} + \frac{|C|}{2}.$$

Combining the results from parts (a), (b), (c) we get

$$|U| \leqslant \frac{|A|}{3} + \frac{5|B|}{6} + \frac{n}{2} - \frac{|A|}{4} - \frac{|B|}{2} + \frac{|C|}{2} - 1 = \frac{n}{2} - 1 + \frac{|A|}{12} - \frac{|B|}{3} + \frac{|C|}{2}. \tag{2}$$

We need one more observation before we conclude the proof. It is easy to see that $|D| \geqslant 3|C|$, since every face in C corresponds to at least 3 faces in G. Using this observation, together with Eq. (1) gives us a bound on the size of $|C|$ in terms of the number of faces in A and in B: $|C| \leqslant |D|/3 = (2n - |A| - 2|B| - 4)/3$. Substituting for C in (2) we obtain

$$|U| \leqslant \frac{n}{2} + \frac{|A|}{12} + \frac{|B|}{3} + \frac{2n - |A| - 2|B| - 4}{6} - 1 = \frac{5n}{6} - \frac{|A|}{12} - \frac{5}{3} \leqslant \frac{5n}{6}.$$

Since $|U| \leqslant 5n/6$, $|M| \geqslant n/6$, which concludes the proof of the theorem. $\square$

4. Algorithm and analysis

Before we can consider a particular embedding we must show how to obtain all the graphs in the hierarchy, $G_0, G_1, \ldots, G_k$. Recall that $G_0 = G$ is a fully triangulated planar graph on n vertices. To construct G_{i+1} from G_i we find a matching E_i of G_i and perform the graph contraction using the edges in E_i. We repeat this process until G_{i+1} is a singleton graph; see Fig. 11.

Set E_i for $0 \leqslant i < k$ contains a maximal matching on the edges of G_i with some added constraints. It is important that after the contraction of the edges in E_i the resulting graph G_{i+1} remains fully triangulated. In order to preserve the mental map, the three locality conditions must be maintained. Finally, in order to maintain a small hierarchical height, $|E_i|$ must be a constant fraction of the edges in G_i. Thus, the constraints that we have on E_i are as follows:

(1) E_i is a matching of simple edges.
(2) After the contraction of all the edges in E_i, subject to the locality conditions, the resulting graph G_{i+1} is maximally planar.
(3) $|E_i| \geqslant |V(G_i)|/c$, for some constant $c > 1$.

Note that condition (1) does not imply condition (2); see Fig. 12. Before we proceed we show how to produce a set E_i which satisfies the above three conditions. Suppose we have graph G_i and we want to create set E_i so that when all the edges in E_i are contracted, we get G_{i+1}. We will contract simple edges of G_i one at a time. When an edge (u, v) is contracted, it is replaced by a vertex w. The next time an edge is contracted, it cannot have w as an endpoint. Let W_i be the set of vertices that were created as a result of contractions in phase i. The edges that we place in E_i must be a matching, and so when a new

```
create_hierarchy($G$)
  $i \leftarrow 0$
  $G_i \leftarrow G$
  **while** $|V_i(G_i)| > 3$
    $G_{i+1} \leftarrow$ match($G_i, E_i$)
    $i \leftarrow i + 1$
```

Fig. 11. Creating the hierarchy of graphs $G_0, G_1, \ldots, G_k$.

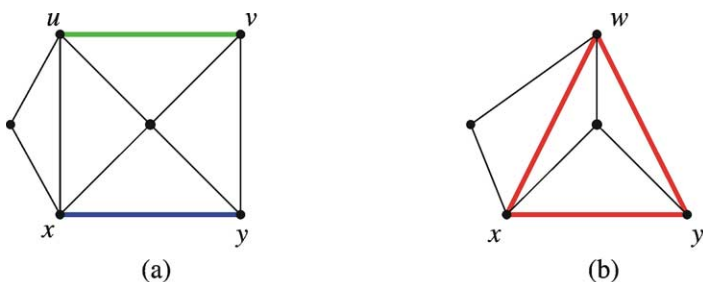

Fig. 12. Edges (u, v) and (x, y) in part (a) are both simple, do not share an endpoint, and can be contracted as a part of a matching. After (u, v) is contracted to w in part (b), edge (x, y) becomes a part of a separating triangle and so it should not be contracted.

```
match($G_i, E_i$)
  $j \leftarrow 0$
  $G_{i,j} \leftarrow G_i$
  $W_i \leftarrow \emptyset$
  while ($S_i \neq \emptyset$)
    Let $u_j \in S_i$
    $S_i \leftarrow S_i \setminus \{u_j\}$
    if $\exists (u_j, v_j) \in G_{i,j}$, s.t. $v_j \notin W_i$ and $(u_j, v_j)$ is simple
      $E_i \leftarrow E_i \cup \{(u_j, v_j)\}$
      Contract $(u_j, v_j)$ to $w_j$ to get $G_{i,j+1}$
      $W_i \leftarrow W_i \cup \{w_j\}$
      $j \leftarrow j + 1$
  return($G_{i,j-1}$)
```

Fig. 13. Create G_{i+1} from G_i by contracting a sequence of edges in E_i.

edge is considered for contraction, it cannot have an endpoint in W_i. Finally, let S_i be the set of vertices of G_i of small degrees. More precisely, let $S_i = \{v \in V(G_i)\colon d_{G_i}(v) < 39\}$.

In general, G_i is transformed into G_{i+1} one edge contraction at a time using the edges in E_i *in the order they were chosen*. We select edges for contraction by first finding a vertex u_j of G_i with small degree (vertices of small degree are members of the set S_i). We look for a simple edge (u_j, v_j), such that v_j is not a vertex obtained from a contraction in the previous phase ($v_j \notin W_i$). Call the intermediate graphs from G_i to G_{i+1}, $G_i = G_{i,0}, G_{i,1}, \ldots, G_{i,j} = G_{i+1}$, and consider the algorithm on Fig. 13.

Lemma 8. *Let $|V(G_i)| = n_i$. Then $|E_i| \geqslant n_i/50$.*

Proof. For G_i with more than 3 vertices, $|E_i| \geqslant 1$. Then consider the sequence of intermediate graphs $G_{i,0}, G_{i,1}, \ldots, G_{i,j}$ and let $G_{i,j}$ have no more edges that could be added to E_i. Observe that we have contracted exactly j edges of G_i and so $|V(G_{i,j})| = n_i - j$. Then from Theorem 1 there are $(n_i - j)/12$ edges in any maximal matching of $G_{i,j}$ which uses only simple edges. Consider such a matching M. Recall that W_i is the set of vertices created as a result of edge contractions in phase i. We are not allowed to add to M vertices from W_i. But since $W_i = j$, at most j of the edges with matched endpoints in M can have endpoints in W_i. Also note that if both endpoints of a simple edge in the matching have degrees greater than or equal to 39 in G_i they cannot be added to M. If there exist at most k vertices of degree greater than or equal to 39, then there are at most $k/2$ such edges. It is easy to show that $k < n_i/12$: Suppose there are k vertices of degrees 39 or more in G_i. Since G_i is fully triangulated, every vertex has degree at least 3 and since G_i is maximally planar, the sum of the degrees is twice the sum of the edges. Then $39k + 3(n_i - k) \leqslant 6n_i - 12$. From this we get that $k < n_i/12$.

We stopped selecting good edges from G_i when we got to graph $G_{i,j}$ in which we could not find a simple edge to contract. The only other types of edges that might be available in $G_{i,j}$ but which we cannot take are those that were at some point non-simple, but later became simple. Also, there can be at most j such edges. Then $(n_i - j)/12 - 2j - n_i/24 \leqslant 0$ which implies $j \geqslant n_i/50$. Thus, if we cannot find another edge to add to the matching, we must have $|E_i| = j \geqslant n_i/50$ which completes the proof. □

We next argue that one call to `match`(G_i, E_i) takes $O(n_i \log n_i)$ time, and since n_{i+1} is a constant fraction of n_i, the $O(\log n)$ calls to `match`(G_i, E_i) take $O(n \log n)$ time overall thus yielding the desired theorem:

Theorem 2. *The clustering algorithm runs in* $O(n \log n)$ *time and produces a sequence of graphs* $G_0, G_1, \ldots, G_k$ *such that* G_i *is maximally planar for all* $0 \leqslant i \leqslant k$ *and* $k = O(\log n)$.

Proof. Let us consider a graph G_i, $0 \leqslant i < k$ in the hierarchical decomposition. From Lemma 8 above we have $|E_i| > n_i/50$, where $n_i = |V(G_i)|$. Then $|V(G_{i+1})| \leqslant 49n_i/50$, that is, the number of vertices of G_{i+1} is a constant fraction of the vertices of G_i.

Recall that G_i is a maximally planar graph and the call to `match`(G_i, E_i) completes when we have contracted G_i to G_{i+1} by constructing the intermediate graphs $G_i = G_{i,0}, G_{i,1}, \ldots, G_{i,j} = G_{i+1}$. Assume that the vertices of V are uniquely labeled. Assume that the graph is maintained as an adjacency list (an adjacency matrix will give us better running time but $O(n^2)$ space complexity). For every vertex $v \in V(G_i)$ we have a list of its neighbors in counterclockwise order. We can trivially sort these lists according to the labels of the neighbors in $O(n_i \log n_i)$ time.

Recall from the algorithm in Fig. 13 that we examine only vertices of small degree in G_i (S_i is the set of such small degree vertices in G_i), and once a vertex is processed, it is not considered again in this call to `match`. Thus we perform $O(n_i)$ vertex examinations. Consider an intermediate graph $G_{i,l}$ where $0 \leqslant l < j$. For every vertex $u_l \in S_i$ we are looking for an adjacent edge (u_l, v) in $G_{i,l}$ such that $v \notin W_i$ and (u_l, v) is simple. To find out if v belongs to W_i takes $O(\log n_i)$ time, provided we maintain the set W_i sorted. To determine whether (u_l, v) is simple we need to check the number of common neighbors of u_l and v. Recall that if u_l and v have exactly two neighbors in common, then (u_l, v) is simple; otherwise it is non-simple. Since $u_l \in S_i$, we have $d_{G_{i,l}}(u_l) \leqslant d_{G_i}(u_l) < 39$. Then we can check in $O(\log n_i)$ time if a neighbor of u_l is also a neighbor of v using binary search in the list of sorted neighbors of v). There is a constant number of neighbors of u_l, so to check all still takes $O(\log n_i)$ time. Thus processing one vertex of S_i takes $O(\log n_i)$ time. Since $|S_i| < n_i$, it takes $O(n_i \log n_i)$ time to process the whole set S_i. □

5. Constructing the embedding

After we obtain the combinatorial graphs $G_0, G_1, \ldots, G_k$ we have to embed them in planes $z = 0$, $z = 1, \ldots, z = k$. While constructing the combinatorial graphs is a bottom up process, constructing the embedding is a top-down one. The first graph to be embedded is G_k, which only has one vertex. We then expand the edges in E_{k-1} one at a time, in the reverse order of their insertion. We then argue that this can be done in a way which guarantees that no crossings are introduced. We need the following lemma.

Lemma 9. *Let* G *be a maximally planar graph embedded in the plane without crossings. For any* $v \in V(G)$, *there exists a ball of radius* $\varepsilon > 0$ *such that if* v *is placed anywhere inside that ball, the embedding has no crossings.*

Proof. The main idea is to consider the visibility region around vertex v. Any point inside that region can "see" all the neighbors of v. It is not hard to show that this region cannot be empty. This would imply the existence of $\varepsilon > 0$ for which the ball of size ε fits inside the visibility region. □

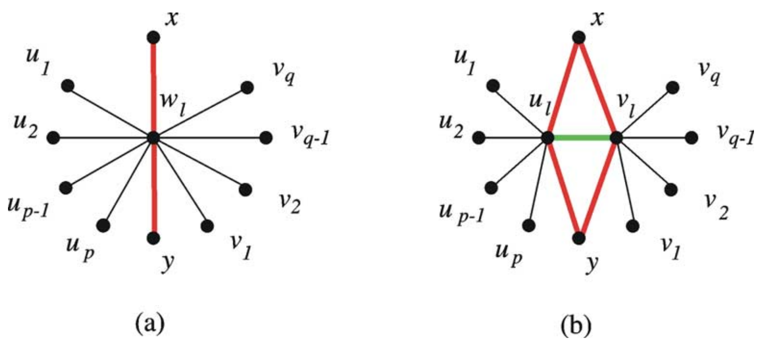

Fig. 14. Vertex w_l and its neighbors in $G_{i,l+1}$ (a) before expansion, (b) after expansion.

Theorem 3. *Given combinatorial representations of graphs $G_k, G_{k-1}, \ldots, G_0$ we can embed them in the planes $z = k, z = k-1, \ldots, z = 0$ so that there are no crossings in any of the drawings.*

Proof. We first embed G_k in the plane $z = k$ without crossings using any straight-line drawing method. Suppose we have embedded $G_k, G_{k-1}, \ldots, G_i$. We will show how to embed G_{i-1} given an embedding for G_i. Recall that we obtained G_i from G_{i-1} through a series of edge contractions from the edge set $E_{i-1} = \{(u_0, v_0), (u_1, v_1), \ldots (u_j, v_j)\}$ which produced graphs $G_{i-1,0}, G_{i-1,1}, \ldots, G_{i-1,j} = G_i$. We now reverse the process and expand G_i back to G_{i-1} through the exact opposite sequence of expansions. Since we have an embedding for G_i in the plane $z = i$, we can embed $G_{i-1,j}$ in the plane $z = i - 1$. Next we expand edge (u_j, v_j) by replacing vertex w_j by the pair u_j, v_j. The resulting graph is $G_{i,j-1}$ and we embed it without a crossing. We proceed until we get to $G_{i,0}$. We next show how to embed $G_{i,l}$ given an embedding for $G_{i,l+1}$, for $0 \leqslant l < j$.

Assume we have a straight-line embedding for $G_{i,l+1}$ without crossings on the plane $z = i$. To get $G_{i,l}$ we must expand vertex w_l back to edge (u_l, v_l). Consider the subgraph on Fig. 14. Let x and y be the neighbors in common for u_l and v_l. We then consider the ball of maximal radius around w_l which sees all neighbors (we know it is of radius $\varepsilon > 0$ from Lemma 9). Consider a diagonal in this ball which is perpendicular to the line connecting x and y. Place u_l and v_l on the two ends of the diagonal. $\square$

We define the *drawing of a clustered graph* $C = (G, T)$ as in [5]. Graph G is drawn as usual, while for every node $v \in T$ the cluster is drawn as a simple closed region R such that:

- All sub-cluster regions of R are completely contained in the interior of R.
- All other cluster regions are completely contained in the exterior of R.
- If there is an edge e between two vertices contained in a cluster v, then the drawing of e is completely contained in R.

Following the definitions of Eades et al. the drawing of edge e and region R have an *edge crossing* if the drawing of e crosses the boundary of R more than once. A drawing of a clustered graph is *c-planar* if there are no edge crossings or edge-region crossings. Graphs with c-planar drawings are c-planar.

Theorem 4. *The clustered graph $C = (G, T)$ produced by our algorithm is c-planar and a c-planar embedding can be obtained in* $O(n^2)$ *time.*

Proof. It suffices to show that there exists a drawing of C which has no edge crossings and no edge-region crossings. Let us embed G using any planar embedding algorithm. Define the region of a cluster, ν to be the simple closed curve around the subgraph of G induced by the cluster, $G(\nu)$. By the definition of the clustering in our algorithm, the subgraph $G(\nu)$ is connected.

If u is a vertex not in cluster ν, then u cannot be contained inside the region R. Assume that u is contained in R. If we contract the edges of ν in the order defined by our algorithm, eventually u will be inside a triangular face. But then none of the edges on that face can be contracted. This is a contradiction since ν is eventually contracted to one vertex.

Finally, since G is embedded in the plane without crossings and the regions are connected there can be neither edge crossings nor edge-region crossings. Therefore C is c-planar and from [5] it follows that the c-planar embedding can be produced in $O(n^2)$ time. □

6. Conclusion and open problems

We have shown how to decompose a large planar graph G into a hierarchy of graphs $G_0, G_1, \ldots, G_k$ such that:

- The height of the hierarchy is $O(\log n)$.
- Graph G_i, $0 \leqslant i \leqslant k$, is embedded in the plane $z = i$ with straight lines and no crossings.
- The transition from G_i to G_{i+1} preserves the mental map.
- The running time of the algorithm is $O(n \log n)$.

Several open problems remain. In Theorem 3 we show that each of the graphs G_i is embedded in the plane with straight lines and no crossings. However, we have not obtained lower or upper bounds on the area necessary for each of the drawings.

Recall that we create a drawing for the original graph $G = G_0$ in a top-down fashion, by first drawing $G_k, G_{k-1}, \ldots, G_1$ and then G_0. A natural question is whether we can achieve similar results in a bottom-up fashion. That is, if we are given a drawing of a planar graph, what conditions are needed so that we can create a hierarchy of graphs which has the above four properties.

Acknowledgements

We would like to thank the anonymous referees for many helpful comments and suggestion.

References

[1] H. de Fraysseix, J. Pach, R. Pollack, How to draw a planar graph on a grid, Combinatorica 10 (1) (1990) 41–51.
[2] C.A. Duncan, M.T. Goodrich, S.G. Kobourov, Balanced aspect ratio trees and their use for drawing very large graphs, in: Proceedings of the 6th Symposium on Graph Drawing, 1998, pp. 111–124.
[3] C.A. Duncan, M.T. Goodrich, S.G. Kobourov, Planarity-preserving clustering and embedding for large planar graphs, in: Proceedings of the 7th Symposium on Graph Drawing, 1999, pp. 186–196.

[4] P. Eades, Q.-W. Feng, Multilevel visualization of clustered graphs, in: Proceedings of the 4th Symposium on Graph Drawing (GD ’96), 1996, pp. 101–112.
[5] P. Eades, Q.-W. Feng, X. Lin, Straight-line drawing algorithms for hierarchical graphs and clustered graphs, in: Proceedings of the 4th Symposium on Graph Drawing (GD ’96), 1996, pp. 113–128.
[6] I. Fáry, On straight lines representation of planar graphs, Acta Scientiarum Mathematicarum 11 (1948) 229–233.
[7] Q.-W. Feng, R.F. Cohen, P. Eades, How to draw a planar clustered graph, in: Proceedings of the 1st Annual International Conference on Computing and Combinatorics (COCOON ’95), 1995, pp. 21–31.
[8] Q.-W. Feng, R.F. Cohen, P. Eades, Planarity for clustered graphs, in: Third Annual European Symposium on Algorithms (ESA ’95), 1995, pp. 213–226.
[9] G.W. Furnas, Generalized fisheye views, in: Proceedings of ACM Conference on Human Factors in Computing Systems (CHI ’86), 1986, pp. 16–23.
[10] K. Kaugars, J. Reinfelds, A. Brazma, A simple algorithm for drawing large graphs on small screens, in: Graph Drawing (GD ’94), 1995, pp. 278–281.
[11] R.J. Lipton, R.E. Tarjan, A separator theorem for planar graphs, SIAM J. Appl. Math. 36 (1979) 177–189.
[12] F.J. Newbery, Edge concentration: A method for clustering directed graphs, in: Proceedings of the 2nd International Workshop on Software Configuration Management, 1989, pp. 76–85.
[13] S.C. North, Drawing ranked digraphs with recursive clusters, in: Graph Drawing, ALCOM International Workshop PARIS on Graph Drawing and Topological Graph Algorithms (GD ’93), 1993.
[14] R. Sablowski, A. Frick, Automatic graph clustering, in: Proc. of 4th Symposium on Graph Drawing (GD ’96), in: Lecture Notes in Computer Science, Vol. 1190, 1996, pp. 395–400.
[15] M. Sarkar, M.H. Brown, Graphical fisheye views, Comm. ACM 37 (12) (1994) 73–84.
[16] W. Schnyder, Embedding planar graphs on the grid, in: Proceedings of the 1st ACM-SIAM Symposium on Discrete Algorithms (SODA), 1990, pp. 138–148.
[17] S.K. Stein, Convex maps, Proc. Amer. Math. Soc. 2 (3) (1951) 464–466.
[18] K. Sugiyama, K. Misue, Visualization of structural information: Automatic drawing of compound digraphs, IEEE Trans. Systems Man Cybernet. 21 (4) (1991) 876–892.
[19] K. Wagner, Bemerkungen zum vierfarbenproblem, Jahresbericht der Deutschen Mathematiker-Vereinigung 46 (1936) 26–32.