Clustered Planarity: Embedded Clustered Graphs with Two-Component Clusters
(Extended Abstract)

Vít Jelínek[1,⋆], Eva Jelínková[1], Jan Kratochvíl[1,2], and Bernard Lidický[1]

[1] Department of Applied Mathematics[⋆⋆]
[2] Institute for Theoretical Computer Science[⋆⋆⋆]
Charles University
Malostranské nám. 25, 118 00 Praha, Czech Republic
{jelinek,eva,honza,bernard}@kam.mff.cuni.cz

Abstract. We present a polynomial-time algorithm for c-planarity testing of clustered graphs with fixed plane embedding and such that every cluster induces a subgraph with at most two connected components.

1 Introduction

Clustered planarity (or shortly, c-planarity) has recently become an intensively studied topic in the area of graph and network visualization. In many situations one needs to visualize a complicated inner structure of graphs and networks. Clustered graphs provide a possible model of such a visualization, and as such they find applications in many practical problems, e.g., management information systems, social networks or VLSI design tools [5]. However, from the theoretical point of view, the computational complexity of deciding c-planarity is still an open problem and it is regarded as one of the challenges of contemporary graph drawing.

A *clustered graph* is a pair $(G, \mathcal{C})$, where $G = (V, E)$ is a graph and $\mathcal{C}$ is a family of subsets of V (called *clusters*), with the property that each two clusters are either disjoint or in inclusion. We always assume that the vertex set V is in $\mathcal{C}$, and we call it *the root cluster*. We say that a clustered graph $(G, \mathcal{C})$ is *clustered-planar* (or shortly *c-planar*), if the graph G has a planar drawing such that we may assign to every cluster $X \in \mathcal{C}$ a compact simply connected region of the plane which contains precisely the vertices of X and whose boundary crosses every edge of G at most once (see Sect. 2 for the precise definition).

It is well known that planar graphs can be recognized in polynomial (even linear) time. For c-planarity, determining the time-complexity of the decision problem remains open; only partial results are known. If every cluster of $(G, \mathcal{C})$ induces a connected subgraph of G, then the c-planarity of $(G, \mathcal{C})$ can be tested in

⋆ Supported by the grant 201/05/H014 of the Czech Science Foundation.
⋆⋆ Supported by project MSM0021620838 of the Czech Ministry of Education.
⋆⋆⋆ Supported by grant 1M0545 of the Czech Ministry of Education.

I.G. Tollis and M. Patrignani (Eds.): GD 2008, LNCS 5417, pp. 121–132, 2009.

linear time by an algorithm of Dahlhaus [3], which improves upon a polynomial algorithm of Feng et al. [5]. Several generalizations of this result are known: c-planarity testing is polynomial for clustered graphs in which all disconnected clusters form a single chain in the cluster hierarchy [7], for clustered graphs in which for every disconnected cluster X, the parent cluster and all the sibling clusters of X are connected [7], and for clustered graphs where every disconnected cluster X has connected parent cluster, with the additional assumption that each component of X is adjacent to a vertex not belonging to the parent of X [6].

Another approach to c-planarity testing is to consider *flat clustered graphs*, which are clustered graphs in which all non-root clusters are disjoint. Even in this restricted setting, the complexity of c-planarity testing is unknown. However, polynomial-time algorithms exist for special types of flat clustered graphs, e.g., if the underlying graph is a cycle and the clusters are arranged in a cycle [2], if the underlying graph is a cycle and the clusters are arranged into an embedded plane graph [1], or if the underlying graph is a cycle and the clusters contain at most three vertices [9]. Even for these very restricted settings, the algorithms are quite non-trivial.

Suppose an embedding of the underlying graph is fixed. Does the c-planarity testing become easier? This question was already addressed in [4], who provide a linear algorithm for flat clustered graphs with a prescribed embedding in which all faces have size at most five.

In this paper, we also deal with clustered graphs $(G, \mathcal{C})$, for which the embedding of G is fixed. In this setting, we obtain a polynomial algorithm for c-planarity of clustered graphs in which each cluster induces a subgraph with at most two connected components.

Theorem 1. *There is a polynomial time algorithm for deciding c-planarity of a clustered graph $(G, \mathcal{C})$, where G is a plane graph and every cluster of $\mathcal{C}$ induces a subgraph of G with at most two connected components.*

In this extended abstract, we present a simplified version of the algorithm which assumes that the cluster hierarchy is flat. We also omit some of the proofs.

2 Preliminaries

We follow standard terminology on finite simple loopless plane graphs. A *plane graph* is an ordered pair $G = (V, E)$, where V is a finite set of points in the plane (called *vertices*) and E is a set of Jordan arcs (called *edges*), such that every edge connects two distinct vertices of G and avoids any other vertex, every pair of vertices is connected by at most one edge, and no two edges intersect, except in a possible common endpoint.

If $G = (V, E)$ is a plane graph and $X \subseteq V$ is a set of vertices, we let $\overline{X}$ denote the set $V \setminus X$ and we let $G[X]$ denote the subgraph of G induced by X.

Two plane graphs $G = (V, E)$ and $G' = (V', E')$ are *isomorphic* if there is a continuous bijection f of the plane with continuous inverse such that $V' = \{f(v) \colon v \in V\}$ and $E' = \{f[e] \colon e \in E\}$ (where $f[e]$ is the set $\{f(x) \colon x \in e\}$).

The algorithm we will present in this paper expects a representation of a plane graph as part of its input. Since the algorithm does not need to make a distinction between isomorphic plane graphs, we may represent a plane graph G by a data structure which identifies G uniquely up to isomorphism. We may identify the isomorphism class of G by specifying, for every vertex of G, the cyclic order of edges and faces incident to v, and by specifying the outer face of G. The isomorphism class of a plane graph can be thus represented by a data structure whose size is polynomial in $|V|$.

Let $G = (V, E)$ be a plane graph. A *cluster set* on G is a set $\mathcal{C} \subseteq \mathcal{P}(V(G))$ such that for all $X, Y \in \mathcal{C}$, either X and Y are disjoint or they are in inclusion; the pair $(G, \mathcal{C})$ is called a *plane clustered graph*. The elements of $\mathcal{C}$ are called *clusters*. We assume that the set $V(G)$ is always in $\mathcal{C}$, and we call it the *root cluster*. A cluster that does not contain any other cluster as a subset is called *minimal*.

Clusters are naturally ordered by inclusion. The set $V(G)$ is the maximum of this ordering. A cluster is called *connected* if it induces in G a connected subgraph and *disconnected* otherwise. A *component* of a cluster $X \in \mathcal{C}$ is a maximal set $X_1 \subseteq X$ such that $G[X_1]$ is a connected subgraph of $G[X]$.

We say that a plane clustered graph $(G, \mathcal{C})$ is *connected* (or *2-connected*, or *disconnected*) if the graph G is connected (or 2-connected, or disconnected). Let us remark that some earlier papers use the term 'connected clustered graph' to denote a clustered graph in which every cluster is connected; we break with this convention for the sake of consistency of our definitions.

In this paper, we consider clustered graphs $(G, \mathcal{C})$ in which every disconnected cluster in $\mathcal{C}$ has exactly two components. We will call such a pair $(G, \mathcal{C})$ a *2-component clustered graph*.

For a plane clustered graph $(G, \mathcal{C})$, a *clustered planar embedding* is a mapping emb_c that assigns to every cluster $X \in \mathcal{C}$ a compact simply connected planar region $emb_c(X)$ (called *the cluster region of* X) whose boundary $\gamma(X)$ is a closed Jordan curve (called *the cluster boundary of* X), such that

- for each vertex $v \in V$ and each cluster $X \in \mathcal{C}$, v is in $emb_c(X)$ if and only if $v \in X$,
- for each cluster $X \in \mathcal{C}$, the cluster boundary $\gamma(X)$ does not contain any vertex from V,
- for every two clusters X and Y, the regions $emb_c(X)$ and $emb_c(Y)$ are disjoint (in inclusion) if and only if X and Y are disjoint (in inclusion, respectively), and
- for every edge $e \in E$ and every cluster $X \in \mathcal{C}$, the edge e crosses the cluster boundary of X at most once.

A plane clustered graph is called *clustered planar* (shortly *c-planar*) if it allows a clustered planar embedding.

When testing c-planarity, we adopt the approach first used in [5] of adding extra edges to the underlying graph in order to make each cluster connected.

Definition 1. *Let $(G, \mathcal{C})$ be a plane clustered graph. Let c be a cycle in G whose vertices all belong to a cluster $X \in \mathcal{C}$. We say that c is a* hole *of the cluster X, if the interior region of c contains a vertex not belonging to X.*

Clearly, a plane clustered graph with a hole is not c-planar. On the other hand, it is known [5] that a plane clustered graph without holes whose clusters are all connected is c-planar. For a given plane clustered graph $(G, \mathcal{C})$ the existence of a hole can be determined in polynomial time [5].

Definition 2. *Let G be a plane graph. A* candidate edge *of G is a simple curve $e \notin E$ such that $(V, E \cup \{e\})$ is a plane graph. A* candidate set *is a set S of candidate edges of G such that $(V, E \cup S)$ is a plane graph. We use the notation $G \cup e$ and $G \cup S$ as a shorthand for $(V, E \cup \{e\})$ and $(V, E \cup S)$ respectively.*

We say that two candidate edges e and e' are isomorphic *if $G \cup e$ and $G \cup e'$ are isomorphic plane graphs.*

Note that a pair of vertices u, v of a plane graph G may be connected by two distinct non-isomorphic candidate edges. On the other hand, it is not hard to see that a plane graph on n vertices has at most $O(n^2)$ non-isomorphic candidate edges.

The following theorem reduces c-planarity testing to searching for a specific set of candidate edges. It was proved in an equivalent version by Feng et al. [5].

Theorem 2. *A plane clustered graph $(G, \mathcal{C})$ is c-planar if and only if there exists a candidate set S with the following properties:*

1. *$(G \cup S, \mathcal{C})$ has no hole,*
2. *every cluster X of $\mathcal{C}$ induces a connected subgraph in $G \cup S$.*

A set S of candidate edges satisfying the above conditions is called a *saturator*[1]. A set S that satisfies the first condition will be called a *partial saturator*. We say that a candidate edge e *saturates* a cluster X, if e connects a pair of vertices belonging to different components of X. A saturator S is *minimal* if no proper subset of S is a saturator. Note that every candidate edge from a minimal saturator S saturates a cluster from $\mathcal{C}$. Moreover, if X is a cluster with two components that does not contain any disconnected subcluster, then a minimal saturator S has exactly one candidate edge saturating X.

Definition 3. *If e is a candidate edge of a plane clustered graph $(G, \mathcal{C})$ such that $(G, \mathcal{C})$ is c-planar if and only if $(G \cup e, \mathcal{C})$ is c-planar, then the edge e is called* harmless. *Similarly, a candidate set S is* harmless *provided $(G, \mathcal{C})$ is c-planar if and only if $(G \cup S, \mathcal{C})$ is c-planar.*

Note that if $(G, \mathcal{C})$ is a c-planar clustered graph, then a candidate set is harmless if and only if it is a subset of a saturator of $(G, \mathcal{C})$. On the other hand, if $(G, \mathcal{C})$ is not c-planar, then any candidate set is harmless.

Let us now present several simple but useful lemmas, whose proofs are omitted due to space constraints.

[1] Note that this definition of saturator differs slightly from that of some other papers—here, candidate edges are already embedded.

Lemma 1. *Let $(G, \mathcal{C})$ be a plane clustered graph without holes, let $X \in \mathcal{C}$ be a cluster which is minimal and connected. Then $(G, \mathcal{C})$ is c-planar if and only if $(G, \mathcal{C} \setminus \{X\})$ is c-planar.*

The next lemma shows that c-planarity testing of 2-component graphs can be reduced to c-planarity testing of 2-component connected plane clustered graphs.

Lemma 2. *If there is a polynomial time algorithm for deciding c-planarity for connected 2-component plane clustered graphs, then there is a polynomial time algorithm for deciding c-planarity for arbitrary 2-component plane clustered graphs.*

The following lemma allows us to reduce c-planarity testing of a connected graph to an equivalent instance of c-planarity where the underlying graph is 2-connected.

Lemma 3. *Let $(G, \mathcal{C})$ be a connected plane clustered graph with at least three vertices which is not 2-connected. There is a polynomial-time transformation which constructs a plane clustered graph $(G', \mathcal{C}')$ such that G' is connected, G' has fewer components of 2-connectivity than G, $(G', \mathcal{C}')$ is c-planar if and only if $(G, \mathcal{C})$ is c-planar, and there is a bijection f between $\mathcal{C}$ and $\mathcal{C}'$ such that for every cluster $X \in \mathcal{C}$, the graph $G[X]$ has the same number of components as the graph $G'[f(X)]$.*

Thanks to Lemma 3, a connected 2-component plane c-planarity instance $(G, \mathcal{C})$ can be polynomially transformed into an equivalent 2-connected 2-component instance $(G', \mathcal{C}')$. To achieve this, we simply perform repeatedly the transformation described in Lemma 3, until the resulting graph has only one 2-connected component.

Combining Lemma 2 and Lemma 3, we see that to decide the c-planarity of 2-component plane graphs, it is sufficient to provide an algorithm that decides c-planarity of 2-connected 2-component plane graph. This is an important technical simplification, because in a 2-connected plane graph, the boundary of every face is a cycle, and a candidate edge in every inner face is uniquely determined (up to isomorphism) by its end-vertices and the face where it should be drawn.

Unfortunately, if F is the outer face of G, a pair of vertices of F may still be connected by two non-isomorphic candidate edges belonging to F (see Fig. 1). To avoid this technical nuisance, we will restrict the set of candidate edges. Let $(G, \mathcal{C})$ be a 2-connected plane clustered graph, let $f \in E(G)$ be an edge which connects a pair of vertices $u, v \in V(G)$, with the following properties:

- f appears on the boundary of the outer face of G,
- every non-root cluster contains at most one of the two vertices u, v.

Such an edge f exists, otherwise the boundary of the outer face would be a hole of a non-root cluster. We say that a candidate edge e of G is *properly drawn* if f is on the boundary of the outer face of $G \cup e$. Note that every candidate edge in an inner face of G is properly drawn, while a pair of non-adjacent vertices on the boundary of the outer face may be connected by two non-isomorphic candidate

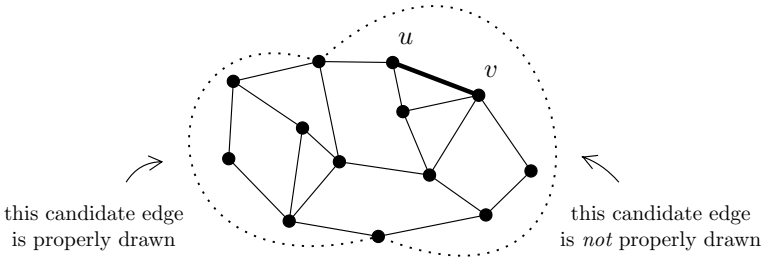

Fig. 1. Two candidate edges connecting the same pair of vertices in the outer face

edges, exactly one of which is properly drawn. Thus, a properly drawn candidate edge is uniquely determined (up to isomorphism) by its pair of endpoints and the face where it should be embedded.

It can be shown that if a 2-connected plane clustered graph is c-planar, then it has a saturator that only contains properly drawn candidate edges.

3 The Algorithm

In this section, we present our algorithm deciding the c-planarity of 2-component plane clustered graphs. As mentioned in the introduction, we will only deal with the restricted setting of *flat* clustered graph, i.e., the clustered graphs where all the non-root clusters are minimal.

Our aim is to find a polynomial algorithm deciding the c-planarity of plane 2-connected 2-component flat clustered graph $(G, \mathcal{C})$.

To achieve this, we will present a polynomial-time procedure FIND-EDGE which, when presented with a 2-component 2-connected hole-free plane clustered graph $(G, \mathcal{C})$ as an input, will either determine that $(G, \mathcal{C})$ is not c-planar, or it will output a harmless candidate edge e that saturates a cluster $X \in \mathcal{C}$. Observe that such a candidate edge e cannot create a hole in $G \cup e$, because both its endpoints belong to different components of X by assumption, and there is no other non-root cluster containing the endpoints of e. This is the main reason why the flat clustered graphs are much easier to deal with than general clustered graphs.

If the procedure FIND-EDGE outputs a harmless candidate edge e, it does not necessarily mean that $(G, \mathcal{C})$ is c-planar. However, since e is harmless, we know that $(G, \mathcal{C})$ is c-planar if and only if $(G \cup e, \mathcal{C})$ is c-planar. We may then call FIND-EDGE again on the input $(G \cup e, \mathcal{C})$, to determine that $(G \cup e, \mathcal{C})$ (and hence also $(G, \mathcal{C})$) is not c-planar, or to find another harmless edge. Since every candidate edge output by the FIND-EDGE procedure saturates a cluster from $\mathcal{C}$, after at most $|\mathcal{C}|$ invocations of FIND-EDGE we will either obtain a saturator of $(G, \mathcal{C})$ or determine that $(G, \mathcal{C})$ is not c-planar.

The FIND-EDGE algorithm maintains a set P of *permitted edges*. In the beginning, the set P is initialized to contain all the properly drawn candidate edges that saturate a cluster from $\mathcal{C}$. In the first phase of the algorithm, called *the*

pruning phase, the algorithm iteratively removes some candidate edges from P, using a set of *pruning rules*, which will be described in Subsection 3.1. The pruning rules guarantee that if $(G, \mathcal{C})$ has a saturator, then it also has a saturator which is a subset of P.

When the set P cannot be further pruned, the algorithm performs the following *triviality checks*, described in detail in Subsection 3.2:

- if there a disconnected cluster that cannot be saturated by any of the permitted edges, then $(G, \mathcal{C})$ is not c-planar,
- if there is a disconnected cluster saturated by a unique permitted edge $e \in P$, then e is harmless,
- if there is a permitted edge e that does not cross any other permitted edge, then e is harmless.

If any of the above conditions is satisfied, the algorithm outputs the corresponding solution and stops. Otherwise, it distinguishes two cases:

1. If there is a disconnected cluster $X \in C$ and a face F of G such that every permitted edge saturating X appears in the face F, then the algorithm performs a subroutine LOCATE-IN-FACE, which will output a harmless permitted edge inside F and stop. This subroutine, together with a brief sketch of its proof, is presented in Subsection 3.3.
2. If the previous case does not apply, it can be shown that any permitted edge is harmless. The algorithm then performs a subroutine called OUTPUT-ANYTHING which outputs an arbitrary permitted edge and stops. The proof of its correctness is sketched in Subsection 3.4.

Before we describe the main parts of the algorithm in greater detail, we need some more terminology.

Let G be a 2-connected plane graph. Let a, b, c, d be a quadruple of distinct vertices on the boundary of a face F of G. We say that the pair ab *crosses* the pair cd in F, if the four vertices appear on the boundary of F in the cyclic order $acbd$. If e and f are two candidate edges of a 2-connected clustered graph $(G, \mathcal{C})$, we say that e *crosses* f if the two candidate edges belong to the same face F of G and the endpoints of e cross with the endpoints of f. For two sets of vertices X and Y, we say that X *crosses* Y *in face* F, if there are vertices $a, b \in X$ and $c, d \in Y$ such that ab crosses cd in the face F.

Most of our arguments rely on the following basic properties of connected subgraphs of 2-connected plane graphs:

- If G is a 2-connected plane graph, and X and Y are disjoint sets of vertices such that $G[X]$ and $G[Y]$ are both connected, then X and Y do not cross in any face of G.
- Let G be a 2-connected plane graph. Let X, Y and Z be disjoint sets of vertices, each of them inducing a connected subgraph of G. Then G has at most two faces that contain vertices of all the three sets on their boundary.

The proof of these properties are omitted from this extended abstract.

3.1 The Pruning Phase

In the pruning phase, the algorithm FIND-EDGE iteratively restricts the set P of permitted candidate edges. In the beginning of the pruning phase, the set P is initialized to contain all the properly drawn candidate edges that saturate at least one cluster. Note that every permitted edge $e \in P$ saturates a unique cluster $X \in \mathcal{C}$, since we assume that $\mathcal{C}$ is flat. A permitted edge that saturates X will be called an *X-edge*.

If X is a minimal cluster, and if e and e' are two X-edges, we say that e and e' are *equivalent*, if for every permitted edge $f \in P$ that is not an X-edge, the edge f crosses e if and only if it crosses e'.

Throughout the pruning phase, the set P will satisfy the following three invariants.

- For each cluster X and each face F, all the X-edges that belong to F form a vertex-disjoint union of complete bipartite subgraphs; these complete bipartite subgraphs will be called *X-bundles* (or just *bundles*, if X is clear from the context). Two X-edges from different bundles do not cross (see Fig. 2).
- If X and Y are distinct clusters, then if an X-edge e crosses two Y-edges f and f', then f and f' belong to the same bundle.
- If $(G, \mathcal{C})$ is c-planar, then it has a saturator that is a subset of P.

In the beginning, when P contains all the properly drawn candidate edges that saturate some cluster from C, the three invariants above are satisfied. In fact, if F is a face that contains at least one X-edge, then all the X-edges in F form a complete bipartite graph. Thus, each face has at most one X-bundle.

To prune the set P, we apply the following two rules.

- If, for a cluster X, there is a permitted edge that crosses all the X-edges, then remove from P each edge that crosses all the X-edges.
- Let $e = uv$ and $e' = u'v$ be two X-edges that belong to the same face F and that share a common vertex v. If e and e' are equivalent, remove from P all the X-edges in F incident to u'.

It can be proven that an arbitrary application of one of the rules above preserves all the invariants. The algorithm applies the pruning rules in arbitrary order, reducing the number of permitted edges in each step, until it reaches the situation when none of the rules is applicable. Let us remark that in the general (i.e., non-flat) situation, the pruning is slightly more complicated: there are four pruning rules instead of two, and the rules have assigned priorities which are taken into account when the algorithm selects which rule to apply.

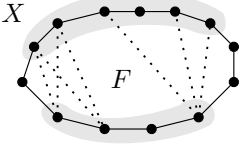

Fig. 2. A face F with two bundles of X-edges

3.2 Triviality Checks

When there is no rule applicable to the set P of permitted edges, the pruning phase ends. The FIND-EDGE algorithm then proceeds with three types of triviality checks, described below.

First, the algorithm checks whether there is a cluster X that is not saturated by any permitted edge. If this is the case, the algorithm concludes that the clustered graph $(G, \mathcal{C})$ is not c-planar and stops. This is a correct conclusion, since if $(G, \mathcal{C})$ were c-planar, then by the last invariant there would have to be a saturator made of permitted edges, which is clearly impossible.

As the next triviality check, the algorithm tries to find a cluster X, such that the set P contains a single X-edge e. If such a cluster X is found, the algorithm outputs e as a harmless edge and stops. This is again a correct output, since by the last invariant, if G is c-planar, then it has a saturator S which is a subset of P. Necessarily, S contains the edge e. This implies that e is harmless.

In the last type of triviality check, the algorithm looks for a permitted edge e that does not cross any permitted edge belonging to a different cluster. If such an edge e is found, the algorithm outputs e as a harmless edge and stops. This is again easily seen to be a correct output.

If none of the triviality checks succeeds, the algorithm counts, for each cluster X, the number of faces of G that contain at least one X-edge. We will say that a cluster X is *one-faced* if all the X-edges belong to a single face of G, X is *two-faced* if all the X-edges appear in the union of two distinct faces, and X is *many-faced* otherwise.

If there is a one-faced cluster X whose permitted edges belong to a face F, then the algorithm performs a subroutine LOCATE-IN-FACE to find a harmless permitted edge in F. This subroutine is described in the next subsection.

If there is no one-faced cluster, it can be shown that all the clusters are two-faced, and that any permitted edge is harmless. The algorithm then outputs an arbitrary permitted edge and stops. The main arguments involved in proving the correctness of this step are sketched in Subsection 3.4.

3.3 LOCATE-IN-FACE

Assume that we are given a set P of permitted edges satisfying all the invariants described in Subsection 3.1. Assume furthermore than none of the pruning rules is applicable to P, and none of the triviality checks has succeeded.

For a face F, we say that a cluster X is an *F-cluster*, if all the X-edges belong to F. We say that a vertex of X is *active*, if it is incident to at least one X-edge.

Assume that F is a face with at least one F-cluster. Using our assumptions about P, we are able to deduce the following facts:

- If X is an F-cluster, and Y is a cluster that has a permitted edge which crosses a permitted edge of X, then Y is also an F-cluster.
- If X is an F-cluster with two components X_1 and X_2, then each component X_i has at most two active vertices. It follows that X has either four permitted

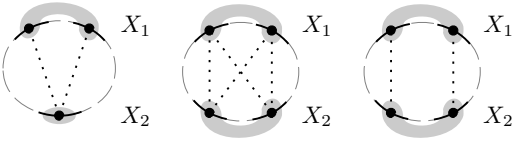

Fig. 3. Possible configurations of permitted edges of an F-cluster X

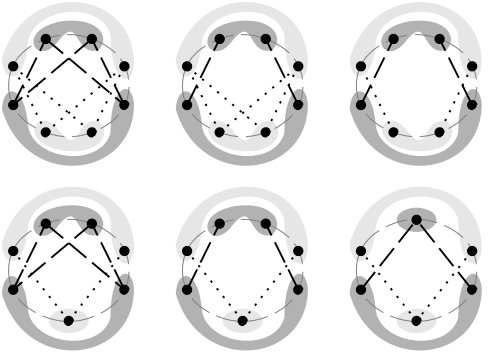

Fig. 4. Mutual positions of permitted edges of two crossing F-clusters

edges which all belong to a single bundle, or X has exactly two permitted edges (see Fig. 3; recall that due to the triviality checks, each cluster has at least two permitted edges).

Let X be an arbitrary F-cluster, let X_1 and X_2 be its two components. From the triviality checks, we know that every X-edge is crossed by a permitted edge of another cluster. Let $Y \neq X$ be a cluster whose permitted edge crosses an X-edge, and let Y_1 and Y_2 be its two components. Note that a set Y_i may not cross with the set X_j on the boundary of F, because these two sets induce connected subgraphs of G. Recall also, that no Y-edge may intersect all the X-edges (and vice versa), because it would have been pruned.

Putting all these facts together, we conclude that the mutual position of the X-edges and Y-edges corresponds to one of the situations depicted on Fig. 4.

Note that all the configurations of Fig. 4 exhibit a 'mirror symmetry'. To make this observation rigorous, we define a 'symmetry mapping' σ on the set of all the F-active vertices as follows: let X be an arbitrary F-cluster, with components X_1 and X_2. If a component X_i contains two active vertices x and x', then we define $\sigma(x) = x'$ and $\sigma(x') = x$. If X_i contains only one active vertex x, then we put $\sigma(x) = x$. We then extend the mapping σ to the set of X-edges in a natural way: for an X-edge e with endpoints x and y, we define $\sigma(e)$ to be the X-edge with endpoints $\sigma(x)$ and $\sigma(y)$.

The mapping σ has the following properties:

– For an F-cluster X and an X-edge e, $\sigma(e)$ is an X-edge different from e.

- If X and Y are F-clusters, an X-edge e crosses a Y-edge f if an only if $\sigma(e)$ crosses $\sigma(f)$.
- An X-edge e is harmless if and only if $\sigma(e)$ is harmless.

From these properties, it can be easily deduced that if an F-cluster X has only two permitted edges, then both these edges are harmless.

Furthermore, it is possible to show that if there is at least one F-cluster in a face F, then there is also an F-cluster that has only two permitted edges.

The procedure LOCATE-IN-FACE is then easy to describe: as an input, the procedure expects a face F for which there is at least one F-cluster. The procedure then finds an F-cluster X that has only two permitted edges, and outputs any X-edge as a harmless edge.

3.4 OUTPUT-ANYTHING

If, after the end of the pruning phase, each cluster has permitted edges in at least two distinct faces, and if none of the triviality checks is applicable, we can show that the set P of permitted edges has the following properties:

- For each cluster X, there are exactly two faces of G that contain the X-edges.
- All the X-edges that appear in the same face are equivalent.
- If X and Y are distinct clusters, and if an X-edge crosses a Y-edge, then all the X-edges and all the Y-edges appear in the same pair of faces, and every Y-edge crosses all the X-edges in its face.
- Let $S \subseteq P$ be a minimal saturator of permitted edges. For each edge $e \in S$ find an arbitrary permitted edge $\overline{e}$ that saturates the same cluster as e and appears in a different face than e. The set $\overline{S} = \{\overline{e}\colon e \in S\}$ is another minimal saturator of permitted edges.

From these properties, we may deduce that every permitted edge $e \in P$ is harmless. The procedure OUTPUT-ANYTHING simply outputs an arbitrary permitted edge and stops.

This completes the description of the simplified version of the FIND-EDGE algorithm. It is clear that the algorithm runs in polynomial time.

4 Concluding Remarks

We have shown that c-planarity of 2-component plane clustered graphs can be determined in polynomial time. This result raises several related open problems.

Problem 1. What is the complexity of the c-planarity problem for 2-component graphs $(G, \mathcal{C})$ if the embedding of G is not prescribed?

Problem 2. What is the complexity of deciding the c-planarity of clustered graphs with $O(1)$ components per cluster?

Problem 3. What if we relax the 2-component assumption by allowing the graph G to have arbitrarily many components, and only restricting the number of components of the non-root clusters?

References

1. Cortese, P.F., Di Battista, G., Patrignani, M., Pizzonia, M.: On embedding a cycle in a plane graph. In: Healy, P., Nikolov, N.S. (eds.) GD 2005. LNCS, vol. 3843, pp. 49–60. Springer, Heidelberg (2006)
2. Cortese, P.F., Di Battista, G., Patrignani, M., Pizzonia, M.: Clustering cycles into cycles of clusters. Journal of Graph Algorithms and Applications 9(3), 391–413 (2005); special issue In: Pach, J. (ed.) GD 2004. LNCS, vol. 3383, pp. 100–110. Springer, Heidelberg (2005)
3. Dahlhaus, E.: A linear time algorithm to recognize clustered planar graphs and its parallelization. In: Lucchesi, C.L., Moura, A.V. (eds.) LATIN 1998. LNCS, vol. 1380, pp. 239–248. Springer, Heidelberg (1998)
4. Di Battista, G., Frati, F.: Efficient C-planarity testing for embedded flat clustered graphs with small faces. In: Hong, S.-H., Nishizeki, T., Quan, W. (eds.) GD 2007. LNCS, vol. 4875, pp. 291–302. Springer, Heidelberg (2008)
5. Feng, Q.W., Cohen, R.F., Eades, P.: Planarity for clustered graphs. In: Spirakis, P.G. (ed.) ESA 1995. LNCS, vol. 979, pp. 213–226. Springer, Heidelberg (1995)
6. Goodrich, M.T., Lueker, G.S., Sun, J.Z.: C-planarity of extrovert clustered graphs. In: Healy, P., Nikolov, N.S. (eds.) GD 2005. LNCS, vol. 3843, pp. 211–222. Springer, Heidelberg (2006)
7. Gutwenger, C., Jünger, M., Leipert, S., Mutzel, P., Percan, M., Weiskircher, R.: Advances in c-planarity testing of clustered graphs. In: Goodrich, M.T., Kobourov, S.G. (eds.) GD 2002. LNCS, vol. 2528, pp. 220–235. Springer, Heidelberg (2002)
8. Gutwenger, C., Jünger, M., Leipert, S., Mutzel, P., Percan, M., Weiskircher, R.: Subgraph induced planar connectivity augmentation. In: Bodlaender, H.L. (ed.) WG 2003. LNCS, vol. 2880, pp. 261–272. Springer, Heidelberg (2003)
9. Jelínková, E., Kára, J., Kratochvíl, J., Pergel, M., Suchý, O., Vyskočil, T.: Clustered planarity: Small clusters in eulerian graphs. In: Hong, S.-H., Nishizeki, T., Quan, W. (eds.) GD 2007. LNCS, vol. 4875, pp. 303–314. Springer, Heidelberg (2008)