A New Perspective on Clustered Planarity as a Combinatorial Embedding Problem*

Thomas Bläsius and Ignaz Rutter

Faculty of Informatics, Karlsruhe Institute of Technology (KIT), Germany
{blaesius,rutter}@kit.edu

Abstract. The clustered planarity problem (c-planarity) asks whether a hierarchically clustered graph admits a planar drawing such that the clusters can be nicely represented by regions. We introduce the cd-tree data structure and give a new characterization of c-planarity. It leads to efficient algorithms for c-planarity testing in the following cases. (i) Every cluster and every co-cluster has at most two connected components. (ii) Every cluster has at most five outgoing edges.

Moreover, the cd-tree reveals interesting connections between c-planarity and planarity with constraints on the order of edges around vertices. On one hand, this gives rise to a bunch of new open problems related to c-planarity, on the other hand it provides a new perspective on previous results.

1 Introduction

When visualizing graphs whose nodes are structured in a hierarchy, one usually has two objectives. First, the graph should be drawn nicely. Second, the hierarchical structure should be expressed by the drawing. Regarding the first objective, we require drawings without edge crossings, i.e., *planar drawings*. A natural way to represent a cluster is a simple region containing exactly the vertices in the cluster. To express the hierarchical structure, the boundaries of two regions must not cross and edges of the graph can cross region boundaries at most once (if only one of its endpoints lies inside the cluster). Such a drawing is called *c-planar*; see Sec. 2 for a formal definition. Testing a clustered graph for c-planarity is a fundamental open problem in the field of Graph Drawing.

C-planarity was first considered by Lengauer [21] (in a different context). He gave an efficient algorithm for the case that every cluster is connected. Feng et al. [13], who coined the name c-planarity, rediscovered the problem and gave a similar algorithm. Cornelsen and Wagner [7] showed that c-planarity is equivalent to planarity when additionally every co-cluster is connected.

Relaxing the condition that every cluster must be connected, makes testing c-planarity surprisingly difficult. Efficient algorithms are known only for very restricted cases and many of these algorithms are very involved. One example is the efficient algorithm by Jelínek et al. [17, 18] for the case that every cluster consists of at most two connected components while the planar embedding of the graph is fixed. Another algorithm by Jelínek et al. [19] solves the case that every cluster has at most four outgoing edges.

* Partly done within GRADR – EUROGIGA project no. 10-EuroGIGA-OP-003. Supported by a fellowship within the Postdoc-Program of the German Academic Exchange Service (DAAD).

C. Duncan and A. Symvonis (Eds.): GD 2014, LNCS 8871, pp. 440–451, 2014.

A popular restriction is to require a *flat* hierarchy, i.e., every pair of clusters has empty intersection. For example, Di Battista and Frati [12] solve the case where the clustering is flat, the graph has a fixed embedding and the faces have size at most 5. Sec. 4.1 and Sec. 4.2 contain additional related work viewed from the new perspective.

Contribution and Outline. We first present the cd-tree data structure (Sec. 3) and use it to characterize c-planarity in terms of combinatorial embeddings of planar graphs. This provides a useful new perspective and significantly simplifies some previous results.

In Sec. 4 we define different constrained-planarity problems. We show in Sec. 4.1 that they are equivalent to different variants of the c-planarity problem of flat-clustered graphs. We also discuss which cases of the constrained embedding problems are solved by previous results on c-planarity. Based on these insights we derive a generic algorithm for testing c-planarity in Sec. 4.2 and discuss previous work in this context.

In Sec. 5, we show how the cd-tree characterization together with results on the problem SIMULTANEOUS PQ-ORDERING [4] lead to efficient algorithms for the cases that (i) every cluster and every co-cluster consists of at most two connected components; or (ii) every cluster has at most five outgoing edges. The latter extends the result by Jelínek et al. [19], where every cluster has at most four outgoing edges.

2 Preliminaries

We denote graphs by G with vertex set V and edge set E. We implicitly assume graphs to be *simple* (no multiple edges or loops). We use the prefix *multi-* to indicate that a graph may have multiple edges (but no loops), e.g., a multi-cycle is obtained from a cycle by multiplying edges. A (multi-)graph G is *planar* if it admits a planar drawing (no edge crossings). The *edge-ordering* of a vertex v is the clockwise cyclic order of its incident edges in a planar drawing of G. An *embedding* of G consists of an edge-ordering for every vertex such that G has a planar drawing with these edge-orderings.

A *PQ-tree* [5] is an unrooted tree T with leaves L such that every inner node is either a *P-node* or a *Q-node*. When embedding T, one can choose the edge-orderings of P-nodes arbitrarily, whereas the edge-orderings of Q-nodes are fixed up to reversal. Every such embedding of T defines a cyclic order on the leaves L. The PQ-tree T *represents* the orders one can obtain in this way. A set of orders is *PQ-representable* if it can be represented by a PQ-tree. The valid edge-orderings of non-cutvertices in planar graphs are PQ-representable (e.g., [4]). Conversely, replace each Q-node of a PQ-tree T by a wheel (to fix its edge-ordering) and connect all leaves to a new vertex v. Then T represents the edge-orderings of v in embeddings of the resulting graph (e.g., [21]).

C-Planarity. A *clustered graph* (G,T) is a graph G together with a rooted tree T whose leaves are the vertices of G. Let μ be a node of T. The tree T_μ is the subtree of T consisting of the root μ and all its successors. The graph induced by the leaves of T_μ is a *cluster* in G. We identify this cluster with the node μ. A cluster is *proper* if it is neither the whole graph (root cluster) nor a single vertex (leaf cluster).

A *c-planar drawing* of (G,T) is a planar drawing of G together with a *simple* (= simply-connected) region R_μ for every cluster μ satisfying the following properties.

(i) Every region R_μ contains exactly the vertices of the cluster μ. (ii) Two regions have non-empty intersection only if one contains the other. (iii) Edges cross the boundary of a region at most once. A clustered graph is *c-planar* if it admits a c-planar drawing.

This definition relies on embeddings in the plane using terms like "outside" and "inside". Instead, one can consider drawings on the sphere by unrooting T, using cuts instead of clusters and simple closed curves instead of simple regions. Removing an edge ε of T splits T in two components. As the leaves of T are the vertices of G, this induces a *corresponding cut* $(V_\varepsilon, V'_\varepsilon)$ with $V'_\varepsilon = V \setminus V_\varepsilon$ on G. For a c-planar drawing of G on the sphere, we require a planar drawing of G together with a simple closed curve C_ε for every cut $(V_\varepsilon, V'_\varepsilon)$ with the following properties. (i) The curve C_ε separates V_ε from V'_ε. (ii) No two curves intersect. (iii) Edges of G cross C_ε at most once.

Using clusters instead of cuts corresponds to orienting the cuts, using one side as cluster and the other side as the cluster's complement (*co-cluster*). C-planarity on the sphere and in the plane are equivalent; one simply has to choose an appropriate point on the sphere to lie in the outer face. We use the rooted and unrooted view interchangeably.

3 The CD-Tree

The *cd-tree (cut- or cluster-decomposition-tree)* of a clustered graph (G,T) is the tree T together with a multi-graph associated with each node of T that represents the decomposition of G along its cuts corresponding to edges in T; see Fig. 1a and b for an example. Lengauer [21] uses a similar structure. Our notation is inspired by SPQR-trees.

Let μ be a node of T with neighbors $\mu_1, \dots, \mu_k$ and incident edges $\varepsilon_i = \{\mu, \mu_i\}$. Removing μ separates the leaves of T into k subsets and thus partitions the vertices of G into $V_1, \dots, V_k \subseteq V$. The *skeleton* $\text{skel}(\mu)$ of μ is the multi-graph obtained from G by contracting each subset V_i into a *virtual vertex* ν_i (we keep multiple edges but remove loops). Note that skeletons of inner nodes of T contain only virtual vertices, while skeletons of leaves consist of one virtual and one non-virtual vertex. The node μ_i is the neighbor of μ *corresponding* to ν_i and the virtual vertex in $\text{skel}(\mu_i)$ corresponding to μ is the *twin* of ν_i, denoted by $\text{twin}(\nu_i)$. Note that $\text{twin}(\text{twin}(\nu_i)) = \nu_i$.

The edges incident to ν_i are exactly the edges of G crossing the cut corresponding to the tree edge ε_i. Thus, the same edges of G are incident to ν_i and $\text{twin}(\nu_i)$. This gives a bound on the total size c of the cd-tree's skeletons (which we shortly call the *size of the cd-tree*). The total number of edges in skeletons of T is twice the total size of all cuts represented by T. Since T represents $O(n)$ cuts, each of size $O(n)$, it is $c \in O(n^2)$.

Assume the cd-tree is rooted. Recall that in this case every node μ represents a cluster of G. The *pertinent graph* $\text{pert}(\mu)$ of the node μ is the cluster represented by μ. Note that one could also define the pertinent graph recursively, by removing the virtual vertex corresponding to the parent of μ (the *parent vertex*) from $\text{skel}(\mu)$ and replacing each remaining virtual vertex by the pertinent graph of the corresponding child of μ. Clearly, the pertinent graph of a leaf of T is a single vertex and the pertinent graph of the root is the whole graph G. A similar concept, also defined for unrooted cd-trees, is the *expansion graph*. The expansion graph $\exp(\nu_i)$ of a virtual vertex ν_i in $\text{skel}(\mu)$ is the pertinent graph of its corresponding neighbor μ_i of μ, when rooting T at μ. One can think of the expansion graph $\exp(\nu_i)$ as the subgraph of G represented by ν_i in $\text{skel}(\mu)$.

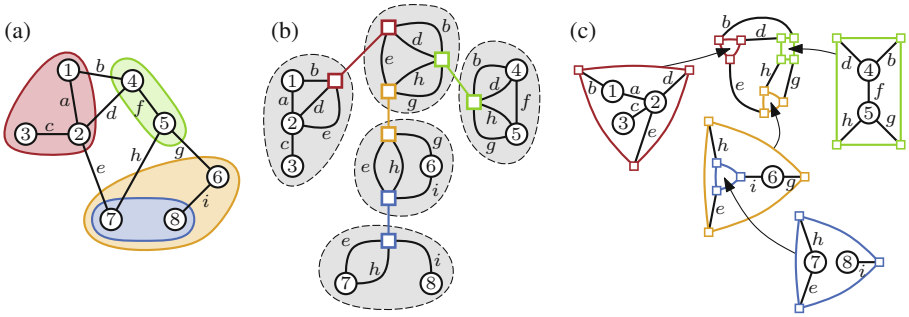

Fig. 1. (a) A c-planar drawing of a clustered graph. (b) The corresponding (rooted) cd-tree (without leaves). The skeletons are drawn inside their corresponding (gray) nodes. Every pair of twins has the same edge-ordering. (c) Construction of a c-planar drawing from the cd-tree.

The leaves of a cd-tree represent singleton clusters that exist only due to technical reasons. It is often more convenient to consider cd-trees with all leaves removed as follows. Let μ be a node with virtual vertex ν in $\text{skel}(\mu)$ that corresponds to a leaf. The leaf contains $\text{twin}(\nu)$ and a non-virtual vertex $v \in V$ in its skeleton (with an edge between $\text{twin}(\nu)$ and v for each edge incident to v in G). We replace ν in $\text{skel}(\mu)$ with the non-virtual vertex v and remove the leaf containing v. Clearly, this preserves all clusters except for the singleton cluster. Moreover, the graph G represented by the cd-tree remains unchanged as we replaced the virtual vertex ν by its expansion graph $\exp(\nu) = v$. In the following we always assume the leaves of cd-trees to be removed.

The CD-Tree Characterization. We show that c-planarity testing can be expressed in terms of edge-orderings in embeddings of the skeletons of T.

Theorem 1. *A clustered graph is c-planar if and only if the skeletons of all nodes in its cd-tree can be embedded such that every virtual vertex and its twin have the same edge-ordering.*

Proof. Assume G admits a c-planar drawing Γ on the sphere. Let μ be a node of T with incident edges $\varepsilon_1, \ldots, \varepsilon_k$ connecting μ to its neighbors $\mu_1, \ldots, \mu_k$, respectively. Let further ν_i be the virtual vertex in $\text{skel}(\mu)$ corresponding to μ_i and let V_i be the nodes in the expansion graph $\exp(\nu_i)$. For every cut (V_i, V_i') (with $V_i' = V \setminus V_i$), Γ contains a simple closed curve C_i representing it. Since the V_i are disjoint, we can choose a point on the sphere to be the outside such that V_i lies inside C_i for $i = 1, \ldots, k$. Since Γ is a c-planar drawing, the C_i do not intersect and only the edges of G crossing the cut (V_i, V_i') cross C_i exactly once. Thus, one can contract the inside of C_i to a single point while preserving the embedding of G. Doing this for each of the curves C_i yields $\text{skel}(\mu)$ together with a planar embedding. Moreover, the edge-ordering of ν_i is the same as the order in which the edges cross the curve C_i. Applying the same construction for the neighbor μ_i corresponding to ν_i yields a planar embedding of $\text{skel}(\mu_i)$ in which the edge-ordering of $\text{twin}(\nu_i)$ is the same as the order in which these edges cross the curve C_i, when traversing C_i in counter-clockwise direction. Thus, in the resulting embeddings

of the skeletons, the edge-ordering of a virtual vertex and its twin is the same up to reversal. To make them the same one can choose a 2-coloring of T and mirror the embeddings of all skeletons of nodes in one color class.

Conversely, assume that all skeletons are embedded such that every virtual vertex and its twin have the same edge-ordering. Let μ be a node of T. Consider a virtual vertex v_i of $\text{skel}(\mu)$ with edge-ordering $e_1, \dots, e_\ell$. We replace v_i by a cycle $C_i = (v_i^1, \dots, v_i^\ell)$ and attach the edge e_j to the vertex v_i^j; see Fig. 1c. Recall that $\text{twin}(v_i)$ has in $\text{skel}(\mu_i)$ the same incident edges $e_1, \dots, e_\ell$ appearing in this order around $\text{twin}(v_i)$. We also replace $\text{twin}(v_i)$ by a cycle of length ℓ. This cycle is the *twin* of C_i and denote it by $\text{twin}(C_i) = (\text{twin}(v_i^1), \dots, \text{twin}(v_i^\ell))$ where $\text{twin}(v_i^j)$ denotes the new vertex incident to the edge e_j. As the interiors of C_i and $\text{twin}(C_i)$ are empty, we can glue the skeletons $\text{skel}(\mu)$ and $\text{skel}(\text{twin}(\mu))$ together by identifying the vertices of C_i with the corresponding vertices in $\text{twin}(C_i)$ (one of the embeddings has to be flipped). Applying this replacement for every virtual vertex and gluing it with its twin leads to an embedded planar graph G^+ with the following properties. First, G^+ contains a subdivision of G. Second, for every cut corresponding to an edge $\varepsilon = \{\mu, \mu_i\}$ in T, G^+ contains the cycle C_i with exactly one subdivision vertex of an edge e of G if the cut corresponding to ε separates the endpoints of e. Third, no two of these cycles share a vertex. The planar drawing of G^+ gives a planar drawing of G. Moreover, the drawings of the cycles can be used as curves representing the cuts, yielding a c-planar drawing of G. □

Cutvertices in Skeletons. We show that cutvertices in skeletons correspond to different connected components in a cluster or in a co-cluster.

Lemma 1. *Let v be a virtual vertex that is a cutvertex in its skeleton. The expansion graphs of virtual vertices in different blocks incident to v belong to different connected components in* $\exp(\text{twin}(v))$.

Proof. Let μ be the node whose skeleton contains v. Recall that one can obtain the graph $\exp(\text{twin}(v))$ by removing v from $\text{skel}(\mu)$ and replacing all other virtual vertices of $\text{skel}(\mu)$ with their expansion graphs. Clearly, this yields (at least) one connected component for each of the blocks incident to v. □

Lemma 2. *Every cluster in a clustered graph is connected if and only if in every node μ of the rooted cd-tree the parent vertex is not a cutvertex in* $\text{skel}(\mu)$.

Proof. By Lemma 1, the existence of a cutvertex implies a disconnected cluster. Conversely, let $\text{pert}(\mu)$ be disconnected and assume without loss of generality that $\text{pert}(\mu_i)$ is connected for every child $\mu_1, \dots, \mu_k$ of μ in the cd-tree. One obtains $\text{skel}(\mu)$ without the parent vertex v by contracting in $\text{pert}(\mu)$ the child clusters $\text{pert}(\mu_i)$ to virtual vertices v_i. As the contracted graphs $\text{pert}(\mu_i)$ are connected while the initial graph $\text{pert}(\mu)$ is not, the resulting graph must be disconnected. Thus, v is a cutvertex in $\text{skel}(\mu)$. □

4 Clustered and Constrained Planarity

We first describe several constraints on planar embeddings, each restricting the edge-orderings of vertices. We then show the relation to c-planarity.

Consider a finite set S (e.g., edges incident to a vertex). Denote the set of all cyclic orders of S by O_S. An *order-constraint* on S is simply a subset of O_S (only the orders in the subset are *allowed*). A *family of order-constraints* for the set S is a set of different order constraints, i.e., a subset of the power set of O_S. We say that a family of order-constraints has a *compact representation*, if one can specify every order-constraint in this family with polynomial space (in $|S|$). In the following we describe families of order-constraints with compact representations.

A *partition-constraint* is given by partitioning S into subsets $S_1 \cup \ldots \cup S_k = S$. It requires that no two partitions *alternate*, i.e., elements $a_i, b_i \in S_i$ and $a_j, b_j \in S_j$ must not appear in the order a_i, a_j, b_i, b_j. A *PQ-constraint* requires that the order of elements in S is represented by a given PQ-tree with leaves S. A *full-constraint* contains only one order, i.e., the order of S is completely fixed.

A *partitioned full-constraint* restricts the orders of elements in S according to a partition constraint (partitions must not alternate) and additionally completely fixes the order within each partition. Similarly, *partitioned PQ-constraints* require the elements in each partition to be ordered according to a PQ-constraint. Clearly, this notion of partitioned order-constraints generalizes to arbitrary order-constraints.

Consider a planar graph G. By *constraining* a vertex v of G, we mean that there is an order-constraint on the edges incident to v. We then only allow planar embeddings of G where the edge-ordering of v is allowed by the order-constraint. By constraining G, we mean that several (or all) vertices of G are constrained.

4.1 Flat-Clustered Graph

Consider a flat-clustered graph, i.e., a clustered graph where the cd-tree is a star. We choose the center μ of the star to be the root. Let $\nu_1, \ldots, \nu_k$ be the virtual vertices in $\text{skel}(\mu)$ corresponding to the children $\mu_1, \ldots, \mu_k$ of μ. Note that $\text{skel}(\mu_i)$ contains exactly one virtual vertex, namely $\text{twin}(\nu_i)$. The possible ways to embed $\text{skel}(\mu_i)$ restrict the possible edge-orderings of $\text{twin}(\nu_i)$ and thus, by the characterization in Theorem 1, the edge-orderings of ν_i in $\text{skel}(\mu)$. Hence, the graph $\text{skel}(\mu_i)$ essentially yields an order constraint for ν_i in $\text{skel}(\mu)$. We consider c-planarity with differently restricted instances leading to different families of order-constraints. To show that testing c-planarity is equivalent to testing whether $\text{skel}(\mu)$ is planar with respect to order-constraints of a specific family, we have to show two directions. First, the embeddings of $\text{skel}(\mu_i)$ only yield order-constraints of the given family. Second, we can get every possible order-constraint of the given family by choosing an appropriate graph for $\text{skel}(\mu_i)$.

Theorem 2. *Testing c-planarity of flat-clustered graphs* (i) where each proper cluster consists of isolated vertices; (ii) where each cluster is connected; (iii) with fixed planar embedding; (iv) without restriction *is linear-time equivalent to testing planarity of a multi-graph with* (i) partition-constraints; (ii) PQ-constraints; (iii) partitioned full-constraints; (iv) partitioned PQ-constraints, *respectively.*

Proof. We start with case (i); see Fig. 2. Consider a flat-clustered graph G and let μ_i be one of the leaves of the cd-tree. As $\text{pert}(\mu_i)$ is a proper cluster, it consists of isolated vertices. Thus, $\text{skel}(\mu_i)$ is a set of vertices $v_1, \ldots, v_\ell$, each connected (with multiple

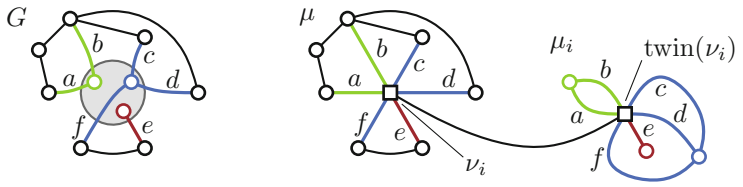

Fig. 2. A graph G with a single cluster consisting of isolated vertices together with an illustration of its cd-tree. An edge-ordering of twin(ν_i) corresponds to a planar embedding of skel(μ_i) if and only if no two partitions of $\{\{a,b\},\{c,d,f\},\{e\}\}$ alternate.

edges) to the virtual vertex twin(ν_i). The vertices $v_1,\dots,v_\ell$ partition the edges incident to twin(ν_i) into ℓ subsets. Clearly, in every planar embedding of skel(μ_i) no two partitions alternate. Moreover, every edge-ordering of twin(ν_i) in which no two partitions alternate gives a planar embedding of skel(μ_i). Thus, the edges incident to ν_i in skel(μ) are constrained by a partition-constraint, where the partitions are determined by the incidence of the edges to the vertices $v_1,\dots,v_\ell$. One can easily construct the resulting instance of planarity with partition-constraints problem in linear time in the size of the cd-tree, which is linear in the size of G for flat-clustered graphs.

Conversely, given a planar graph H with partition-constraints, we set skel(μ) $= H$. For every vertex of H we have a virtual vertex ν_i in skel(μ) with corresponding child μ_i. We can simulate every partitioning of the edges incident to ν_i by connecting edges incident to twin(ν_i) (in skel(μ_i)) with vertices such that two edges are connected with the same vertex if and only if they belong to the same partition.

Consider case (ii). By Lemma 2 the condition of connected clusters is equivalent to requiring that the virtual vertex twin(ν_i) in the skeleton of any leaf μ_i of the cd-tree is not a cutvertex. The statement follows from the fact that the possible edge-orderings of non-cutvertices is PQ-representable and that any PQ-tree can be achieved by choosing an appropriate planar graph in which twin(ν_i) is not a cutvertex (see Sec. 2).

Consider case (iii). As in case (i), the blocks incident to twin(ν_i) in skel(μ_i) partition the edges incident to ν_i in skel(μ) such that two partitions must not alternate. The fixed embedding of G fixes the edge-ordering of non-virtual vertices and thus fixes the embeddings of the blocks in skel(μ_i). Hence, we get partitioned full-constraints for ν_i. Conversely, we can construct an arbitrary partitioned full-constraint as in case (i).

For case (iv) the arguments from case (iii) show that we again get partitioned order-constraints, while the arguments from case (ii) show that these order-constraints (for the blocks) are PQ-constraints. □

Related Work. Biedl [1] proposes different drawing-models for graphs whose vertices are partitioned into two subsets. The model matching the requirements of c-planar drawings is called *HH-drawings*. Biedl et al. [2] show that one can test for the existence of HH-drawings in linear time. Hong and Nagamochi [16] rediscovered this result in the context of 2-page book embeddings. These results solve c-planarity for flat-clustered graphs if the skeleton of the root node contains two virtual vertices. This is equivalent to testing planarity with partitioned PQ-constraints for multi-graphs with only two

vertices (Theorem 2). Thus, to solve c-planarity for flat-clustered graphs, one needs to solve an embedding problem on general planar multi-graphs that is so far only solved on a set of parallel edges (with absolutely non-trivial algorithms). This indicates that we are still far away from solving the c-planarity problem even for flat-clustered graphs.

Cortese et al. [9] give a linear-time algorithm for testing c-planarity of a flat-clustered cycle (i.e., G is a simple cycle) if the skeleton of the cd-tree's root is a multi-cycle. Requiring that G is a cycle implies that the skeleton of each non-root node in T has the property that the blocks incident to the parent vertex are simple cycles. Thus, in terms of constrained planarity, they show how to test planarity of multi-cycles with partition-constraints where each partition has size two. The result can be extended to a special case of c-planarity where the clustering is not flat. However, the cd-tree fails to have easy-to-state properties in this case, which shows that the cd-tree perspective of course has some limitations. Later, Cortese et al. [10] extended this result to the case where G is still a cycle, while the skeleton of the root can be an arbitrary planar multi-graph that has a fixed embedding up to the ordering of parallel edges. This is equivalent to testing planarity of such a graph with partition-constraints where each partition has size two.

Jelínková et al. [20] consider the case where each cluster contains at most three vertices (with additional restrictions). Consider a cluster containing only two vertices u and v. If u and v are connected, then the region representing the cluster can be always added and we can omit the cluster. Otherwise, the region representing the cluster in a c-planar drawing implies that one can add the edge uv to G, yielding an equivalent instance. Thus, one can assume that every cluster has size exactly 3, which yields flat-clustered graphs. In this setting they give efficient algorithms for the cases that G is a cycle and G is 3-connected. Moreover, they give an FPT-algorithm for the case that G is an *Eulerian graph* with k nodes, i.e., a graph obtained from a 3-connected graph of size k by multiplying and then subdividing edges.

In case G is 3-connected, its planar embedding is fixed and thus the edge-ordering of non-virtual vertices is fixed. Thus, one obtains partitioned full-constraints with the restriction that there are only three partitions. Clearly, the requirement that G is 3-connected also restricts the possible skeletons of the root of the cd-tree. It is an interesting open question whether planarity with partitioned full-constraints with at most three partitions can be tested efficiently for arbitrary planar graphs. In case G is a cycle, one obtains partition constraints with only three partitions and each partition has size two. Note that this in particular restricts the skeleton of the root to have maximum degree 6. Although these kind of constraints seem pretty simple to handle, the algorithm by Jelínková et al. is pretty involved. It seems like one barrier where constrained embedding becomes difficult is when there are partition constraints with three or more partitions (see also Theorem 4). The result about Eulerian graphs in a sense combines the cases where G is 3-connected and a cycle. A vertex has either degree two and thus yields a partition of size two or it is one of the constantly many vertices with higher degree for which the edge-ordering is partly fixed.

4.2 General Clustered Graphs

Expressing c-planarity for general clustered graphs (not necessarily flat) in terms of constrained planarity problems is harder for the following reason. Consider a leaf μ in

the cd-tree. The skeleton of μ is a planar graph yielding (as in the flat-clustered case) partitioned PQ-constraints for its parent μ'. This restricts the possible embeddings of $\text{skel}(\mu')$ and thus the order-constraints one obtains for the parent of μ' are not necessarily again partitioned PQ-constraints.

One can express this issue in the following, more formal way. Let G be a planar multi-graph with vertices $v_1,\dots,v_n$ and designated vertex $v = v_n$. The map φ_G^v maps a tuple $(C_1,\dots,C_n)$ where C_i is an order-constraint on the edges incident to v_i to an order-constraint C on the edges incident to v. The order-constraint $C = \varphi_G^v(C_1,\dots,C_n)$ contains exactly those edge-orderings of v that one can get in a planar embedding of G that respects $C_1,\dots,C_n$. Note that C is empty if and only if there is no such embedding. Note further that testing planarity with order-constraints is equivalent to deciding whether φ_G^v evaluates to the empty set. We call such a map φ_G^v a *constrained-embedding operation*.

The issue mentioned above (constraints iteratively handed to parents) boils down to the fact that partitioned PQ-constraints are not closed under constrained-embedding operations. On the positive side, we obtain a general algorithm for solving c-planarity as follows. Assume we have a family of order-constraints $\mathcal{C}$ with compact representations that is closed under constrained-embedding operations. Assume further that we can evaluate the constrained embedding operations in polynomial time on order-constraints in $\mathcal{C}$. Then one can simply solve c-planarity by traversing the cd-tree bottom-up, evaluating for a node μ with parent vertex v the constrained-embedding operation $\varphi^v_{\text{skel}(\mu)}$ on the constraints one computed in the same way for the children of μ.

Clearly, when restricting the skeletons of the cd-tree or requiring properties for the parent vertices in these skeletons, these restrictions carry over to the constrained-embedding operations one has to consider. More precisely, let $\mathcal{R}$ be a set of pairs (G,v), where v is a vertex in G. We say that a clustered graph is *$\mathcal{R}$-restricted* if $(\text{skel}(\mu),v) \in \mathcal{R}$ holds for every node μ in the cd-tree with parent vertex v. Moreover, the *$\mathcal{R}$-restricted* constrained-embedding operations are those operations φ_G^v with $(G,v) \in \mathcal{R}$. The following theorem directly follows.

Theorem 3. *One can solve c-planarity of $\mathcal{R}$-restricted clustered graphs in polynomial time if there is a family $\mathcal{C}$ of order-constraints such that*

- *$\mathcal{C}$ has a compact representation,*
- *$\mathcal{C}$ is closed under $\mathcal{R}$-restricted constrained-embedding operations,*
- *every $\mathcal{R}$-restricted constrained-embedding operation on order-constraints in $\mathcal{C}$ can be evaluated in polynomial time.*

When dropping the requirement that $\mathcal{C}$ has a compact representation the algorithm becomes super-polynomial only in the maximum degree d of the virtual vertices (the number of possible order-constraints for a set of size d depends only on d). Moreover, if φ_G^v has only k order constraints (whose sizes are bounded by a function of d) as input, then φ_G^v can be evaluated by iterating over all combinations of orders, applying a planarity test in every step. This gives an FPT-algorithm with parameter $d+k$ (running time $O(f(d+k)p(n))$, where f is a computable function depending only on $d+k$ and p is a polynomial). In other words, we obtain an FPT-algorithm where the parameter is the sum of the maximum degree of the tree T and the maximum number of edges leaving

a cluster. Note that this generalizes the FPT-algorithm by Chimani and Klein [6] with respect to the total number of edges connecting different clusters.

Moreover, Theorem 3 has the following simple implication. Consider a clustered graph where each cluster is connected. This restricts the skeletons of the cd-tree such that none of the parent vertices is a cutvertex (Lemma 1). Thus, we have $\mathcal{R}$-restricted clustered graphs where $(G, v) \in \mathcal{R}$ implies that v is not a cutvertex in G. PQ-constraints are closed under $\mathcal{R}$-restricted constrained-embedding operations as the valid edge-ordering of non-cutvertices is PQ-representable and planarity with PQ-constraints is basically equivalent to planarity (one can model a PQ-tree with a simple gadget; see Sec. 2). Thus, Theorem 3 directly implies that c-planarity can be solved in polynomial time if each cluster is connected.

Related Work. The above algorithm resulting from Theorem 3 is more or less the one described by Lengauer [21]. The algorithm was later rediscovered by Feng et al. [13] who coined the term "c-planarity". The algorithm runs in $O(c) \subseteq O(n^2)$ time (recall that c is the size of the cd-tree). Dahlhaus [11] improves the running time to $O(n)$. Cortese et al. [8] give a characterization that also leads to a linear-time algorithm.

Goodrich et al. [14] consider the case where each cluster is either connected or *extrovert*. Let μ be a node in the cd-tree with parent μ'. The cluster $\text{pert}(\mu)$ is extrovert if the parent cluster $\text{pert}(\mu')$ is connected and every connected component in $\text{pert}(\mu)$ is connected to a vertex not in the parent $\text{pert}(\mu')$. They show that one obtains an equivalent instance by replacing the extrovert cluster $\text{pert}(\mu)$ with one cluster for each of its connected components while requiring additional PQ-constraints for the parent vertex in the resulting skeleton. In this instance every cluster is connected and the additional PQ-constraints clearly do no harm.

Another extension to the case where every cluster must be connected is given by Gutwenger et al. [15]. They give an algorithm for the case where every cluster is connected with the following exception. Either, the disconnected clusters form a path in the tree or for every disconnected cluster the parent and all siblings are connected. This has basically the effect that at most one order-constraint in the input of a constrained-embedding operation is not a PQ-tree.

Jelínek et al. [17, 18] assume each cluster to have at most two connected components and the underlying (connected) graph to have a fixed planar embedding. Thus, they consider $\mathcal{R}$-restricted clustered graphs where $(G, v) \in \mathcal{R}$ implies that v is incident to at most two different blocks. The fixed embedding of the graph yields additional restrictions that are not so easy to state within this model.

5 Cutvertices with Two Non-trivial Blocks

The input of the SIMULTANEOUS PQ-ORDERING problem consists of several PQ-trees together with child-parent relations between them (the PQ-trees are the nodes of a directed acyclic graph) such that the leaves of every child form a subset of the leaves of its parents. SIMULTANEOUS PQ-ORDERING asks whether one can choose orders for all PQ-trees *simultaneously* in the sense that every child-parent relation implies that the order of the leaves of the parent are an extension of the order of the leaves of the child.

In this way one can represent orders that cannot be represented by a single PQ-tree. For example, adding one or more children to a PQ-tree T restricts the set of orders represented by T by requiring the orders of different subsets of leaves to be represented by some other PQ-tree. Moreover, one can synchronize the orders of different trees that share a subset of leaves by introducing a common child containing these leaves.

SIMULTANEOUS PQ-ORDERING is NP-hard but efficiently solvable for so-called 2-fixed instances [4]. For every biconnected planar graph G, there exists an instance of SIMULTANEOUS PQ-ORDERING, the *PQ-embedding representation*, that represents all planar embeddings of G [4]. It has the following properties.

- For every vertex v in G there is a PQ-tree $T(v)$, the *embedding tree*, that has the edges incident to v as leaves.
- For every solution of the PQ-embedding representation, setting the edge-ordering of every vertex v to the order given by $T(v)$ yields a planar embedding. Moreover, one can obtain every embedding of G in this way.
- The instance remains 2-fixed when adding up to one child to each embedding tree.

A PQ-embedding representation still exists if every cutvertex in G is incident to at most two *non-trivial blocks* (blocks that are not just bridges) [3].

Theorem 4. *C-planarity can be tested in $O(c^2) \subseteq O(n^4)$ time if every virtual vertex in the skeletons of the cd-tree is incident to at most two non-trivial blocks.*

Proof. Let G be a clustered graph with cd-tree T. For the skeleton of each node in T, we get a PQ-embedding representation with the above-mentioned properties. Let μ be a node of T and let v be a virtual vertex in $\text{skel}(\mu)$. Let μ' be the node whose skeleton contains $\text{twin}(v)$. The embedding representations of $\text{skel}(\mu)$ and $\text{skel}(\mu')$ contain the embedding trees $T(v)$ and $T(\text{twin}(v))$ representing the edge-orderings of v and $\text{twin}(v)$, respectively. To ensure that v and $\text{twin}(v)$ have the same edge-ordering, one can simply add a PQ-tree as common child of $T(v)$ and $T(\text{twin}(v))$. We do this for every virtual node in the skeletons of T. Due to the last property of the PQ-embedding representations, the resulting instance remains 2-fixed and can thus be solved efficiently.

Every solution of this SIMULTANEOUS PQ-ORDERING instance D yields planar embeddings of the skeletons such that every virtual vertex and its twin have the same edge-ordering and vice versa. By Theorem 1, testing c-planarity is equivalent to solving D. The size of D is linear in the size c of T. Moreover, solving SIMULTANEOUS PQ-ORDERING for 2-fixed instances can be done in quadratic time [4], yielding the running time $O(c^2)$. □

Theorem 4 includes the following interesting cases. The latter extends the result by Jelínek et al. [19] from four to five outgoing edges per cluster.

Corollary 1. *C-planarity can be tested in $O(c^2) \subseteq O(n^4)$ time if every cluster and every co-cluster has at most two connected components.*

Corollary 2. *C-planarity can be tested in $O(n^2)$ time if every cluster has at most five outgoing edges.*

References

1. Biedl, T.: Drawing planar partitions I: LL-drawings and LH-drawings. In: SoCG 1998, pp. 287–296. ACM (1998)
2. Biedl, T., Kaufmann, M., Mutzel, P.: Drawing planar partitions II: HH-drawings. In: Hromkovič, J., Sýkora, O. (eds.) WG 1998. LNCS, vol. 1517, pp. 124–136. Springer, Heidelberg (1998)
3. Bläsius, T., Rutter, I.: Simultaneous PQ-ordering with applications to constrained embedding problems. CoRR abs/1112.0245, 1–46 (2011)
4. Bläsius, T., Rutter, I.: Simultaneous PQ-ordering with applications to constrained embedding problems. In: SODA 2013. SIAM (2013)
5. Booth, K.S., Lueker, G.S.: Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. J. Comput. System Sci. 13(3), 335–379 (1976)
6. Chimani, M., Klein, K.: Shrinking the search space for clustered planarity. In: Didimo, W., Patrignani, M. (eds.) GD 2012. LNCS, vol. 7704, pp. 90–101. Springer, Heidelberg (2013)
7. Cornelsen, S., Wagner, D.: Completely connected clustered graphs. J. of Disc. Alg. 4(2), 313–323 (2006)
8. Cortese, P.F., Di Battista, G., Frati, F., Patrignani, M., Pizzonia, M.: C-planarity of c-connected clustered graphs. J. Graph Alg. Appl. 12(2), 225–262 (2008)
9. Cortese, P.F., Di Battista, G., Patrignani, M., Pizzonia, M.: Clustering cycles into cycles of clusters. J. Graph Alg. Appl. 9(3), 391–413 (2005)
10. Cortese, P.F., Di Battista, G., Patrignani, M., Pizzonia, M.: On embedding a cycle in a plane graph. Disc. Math. 309(7), 1856–1869 (2009)
11. 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)
12. 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)
13. 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)
14. 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)
15. 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)
16. Hong, S.H., Nagamochi, H.: Two-page book embedding and clustered graph planarity. Tech. Rep. 2009-004, Kyoto University, Depart. Appl. Math. & Phys. (2009)
17. Jelínek, V., Jelínková, E., Kratochvíl, J., Lidický, B.: Clustered planarity: Embedded clustered graphs with two-component clusters (2009), `http://kam.mff.cuni.cz/~bernard/pub/flat.pdf` (manuscript)
18. Jelínek, V., Jelínková, E., Kratochvíl, J., Lidický, B.: Clustered planarity: Embedded clustered graphs with two-component clusters (extended abstract). In: Tollis, I.G., Patrignani, M. (eds.) GD 2008. LNCS, vol. 5417, pp. 121–132. Springer, Heidelberg (2009)
19. Jelínek, V., Suchý, O., Tesař, M., Vyskočil, T.: Clustered planarity: Clusters with few outgoing edges. In: Tollis, I.G., Patrignani, M. (eds.) GD 2008. LNCS, vol. 5417, pp. 102–113. Springer, Heidelberg (2009)
20. Jelínková, E., Kára, J., Kratochvíl, J., Pergel, M., Suchý, O., Vyskočil, T.: Clustered planarity: Small clusters in cycles and eulerian graphs. J. Graph Alg. Appl. 13(3), 379–422 (2009)
21. Lengauer, T.: Hierarchical planarity testing algorithms. J. ACM 36(3), 474–509 (1989)