Theoretical Computer Science 609 (2016) 306–315

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

A new perspective on clustered planarity as a combinatorial embedding problem ☆,☆☆

Thomas Bläsius *, Ignaz Rutter *,[1]

Karlsruhe Institute of Technology (KIT), Germany

ARTICLE INFO

Article history:
Received 26 March 2015
Received in revised form 14 August 2015
Accepted 6 October 2015
Available online 22 October 2015
Communicated by A. Marchetti-Spaccamela

Keywords:
Graph drawing
Clustered planarity
Constrained planar embedding
Characterization
Algorithms

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 (complement of a 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* (the number of crossings in a drawing of a graph is a major aesthetic criterion). A natural way to represent a set of hierarchically aggregated vertices, usually called *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, namely if only one of its endpoints lies inside the cluster. Such a drawing is called *c-planar*; see Section 2 for a formal definition. Testing a clustered graph for *c-planarity* (i.e., testing whether it admits a c-planar drawing) is a fundamental open problem in the field of Graph Drawing.

C-planarity was first considered by Lengauer [1] in a completely different context. He gave an efficient algorithm for the case that every cluster is connected. Feng et al. [2], who coined the name c-planarity, rediscovered the problem and gave a similar algorithm. Cornelsen and Wagner [3] showed that c-planarity is equivalent to planarity when every cluster and every co-cluster is connected.

☆ Partly done within GRADR – EUROGIGA project no. 10-EuroGIGA-OP-003.

☆☆ A preliminary version of this work has been published as T. Bläsius and I. Rutter: A New Perspective on Clustered Planarity as a Combinatorial Embedding Problem, *Proceedings of the 22th International Symposium on Graph Drawing (GD'14)*, pages 440–451, LNCS, Springer, 2014.

* Corresponding authors.
E-mail addresses: blaesius@kit.edu (T. Bläsius), rutter@kit.edu (I. Rutter).

[1] Supported by a fellowship within the Postdoc-Program of the German Academic Exchange Service (DAAD).

http://dx.doi.org/10.1016/j.tcs.2015.10.011
0304-3975/

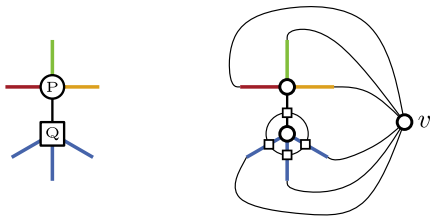

Fig. 1. Construction of a graph G where the edge-orderings of v in embeddings of G are represented by a given PQ-tree T. The boundary vertices of the wheel representing the Q-node are shown as squares.

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. [4,5] for the case that every cluster consists of at most two connected components while the planar embedding of the underlying graph is fixed. Another efficient algorithm by Jelínek et al. [6] solves the case that every cluster has at most four outgoing edges.

A popular restriction is to require a *flat* hierarchy, i.e., every pair of clusters has empty intersection. For example, Di Battista and Frati [7] solve the case where the clustering is flat, the graph has a fixed planar embedding and the size of the faces is bounded by five. Section 4.1.1 and Section 4.2.1 contain additional related work viewed from the new perspective.

1.1. Contribution and outline

We first present the cd-tree data structure (Section 3), which is similar to a data structure used by Lengauer [1]. We use the cd-tree to characterize c-planarity in terms of a combinatorial embedding problem. We believe that our definition of the cd-tree together with this characterization provides a very useful perspective on the c-planarity problem and significantly simplifies some previous results.

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

In Section 5, we show how the cd-tree characterization together with results on the problem Simultaneous PQ-Ordering [8] leads 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. [6], 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*, i.e., they do not have multiple edges or loops. Sometimes we allow multiple edges (we never allow loops). We indicate this with the prefix *multi-*, e.g., a multi-cycle is a graph 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. A *(planar) embedding* of G consists of an edge-ordering for every vertex such that G admits a planar drawing with these edge-orderings.

A *PQ-tree* [9] is a tree T (in our case unrooted) with leaves L such that every inner node is either a *P-node* or a *Q-node*. When embedding T, one can choose the (cyclic) 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. It is not hard to see that the valid edge-orderings of non-cutvertices in planar graphs are PQ-representable (e.g., [8]). Conversely, adding wheels around the Q-nodes of a PQ-tree T and connecting all leaves with a vertex v yields a planar graph G where the edge-orderings of v in embeddings of G are represented by T (e.g., [1]); see Fig. 1.

2.1. C-planarity on the plane and on the sphere

A *clustered graph* (G, T) is a graph G together with a rooted tree T whose leaves are the vertices of G (we also say that G itself is a clustered graph). Let μ be a node of T. The tree T_μ is the subtree of T consisting of all successors of μ together with the root μ. The graph induced by the leaves of T_μ is a *cluster* in G. We identify this cluster with the node μ. We call a cluster *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 in the plane 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 in

its interior. (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 and only if it admits a c-planar drawing.

The above definition of c-planarity relies on embeddings in the plane using the term "contains", and thus relies on concepts like "outside" and "inside". Instead, one can consider drawings on the sphere by considering the tree T to be unrooted instead of rooted, using cuts instead of clusters, and simple closed curves instead of simple regions. Let ε be an edge in T. Removing ε splits T in two connected 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) Each edge of G cross C_ε at most once.

Note that using clusters instead of cuts corresponds to orienting the cuts, using one side as the cluster and the other side as the cluster's complement (the *co-cluster*). This notion of c-planarity on the sphere is equivalent to the one on the plane; one simply has to choose an appropriate point on the sphere to lie in the outer face. The unrooted view has the advantage that it is more symmetric (i.e., there is no difference between clusters and co-clusters), which is sometimes desirable. We use the rooted and unrooted view interchangeably.

To clarify the interplay between the rooted and the unrooted view, consider an edge $\{\mu_1, \mu_2\}$ in the tree T and let (V_1, V_2) be the corresponding cut. Assume for a moment that T is rooted such that μ_2 is the parent of μ_1. Then the leaves that are descendants of μ_1 in T are exactly the vertices V_1. Thus, the cluster μ_1 and the cut (V_1, V_2) do more or less the same thing: they both separate V_1 from V_2 (with V_1 being the interior, i.e., the cluster, and V_2 being the exterior, i.e., the co-cluster). Now assume that we choose a different root such that μ_1 is the parent of μ_2. Then the cluster μ_2 includes the vertices V_2 and V_1 forms the corresponding co-cluster. Thus, which of the clusters μ_1 or μ_2 matches the cut corresponding to $\{\mu_1, \mu_2\}$ depends on the root. This makes sense as changing the root potentially changes the subtree T_μ below a node μ. Note that the root of T plays a special role. Let μ be an arbitrarily chosen root of T. As μ has no parent, there is no edge in T whose corresponding cut separates the vertices in the cluster μ from the vertices in the co-cluster. However, this also makes sense, as μ is the root cluster containing the whole graph (as every vertex of G is a leaf in T and thus a descendant of μ).

3. The CD-tree

Let (G, T) be a clustered graph. We introduce the *cd-tree (cut- or cluster-decomposition-tree)* by enhancing each node of T with a multi-graph that represents the decomposition of G along its cuts corresponding to edges in T; see Figs. 2a–b for an example. We note that Lengauer [1] uses a similar structure. Our notation is inspired by SPQR-trees [10].

Let μ be a node of T with neighbors $\mu_1, \dots, \mu_k$ and incident edges $\varepsilon_i = \{\mu, \mu_i\}$ (for $i = 1, \dots, k$). Removing μ separates T into k subtrees $T_1, \dots, T_k$. Let $V_1, \dots, V_k \subseteq V$ be the vertices of G represented by leaves in these subtrees. The *skeleton* $\text{skel}(\mu)$ of μ is the multi-graph obtained from G by contracting each subset V_i into a single vertex ν_i (the resulting graph has multiple edges but we remove loops). These vertices ν_i are called *virtual vertices*. 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 virtual vertex ν_i in $\text{skel}(\mu)$ *corresponds* to μ_i; this node represents the set V_i. Likewise, there is a virtual node ν in $\text{skel}(\mu_i)$ *corresponding* to μ; this node represents the set of vertices $V \setminus V_i$. We say that ν and ν_i are *twins*, and we write $\text{twin}(\nu) = \nu_i$ and $\text{twin}(\nu_i) = \nu$. 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. As edges might cross a linear number of cuts (but obviously not more), the cd-tree has at most quadratic size in the number of vertices of G, i.e., $c \in O(n^2)$.

Note that the above definition does not rely on a root. However, sometimes a root is useful as it serves as reference point in the tree. Assume the cd-tree is rooted. Recall that in this case every node μ represents a cluster of G. In analogy to the notion for SPQR-trees, we define the *pertinent graph* $\text{pert}(\mu)$ of the node μ to be 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 $\text{skel}(\mu)$. As mentioned before, we use the rooted and unrooted points of view interchangeably.

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.

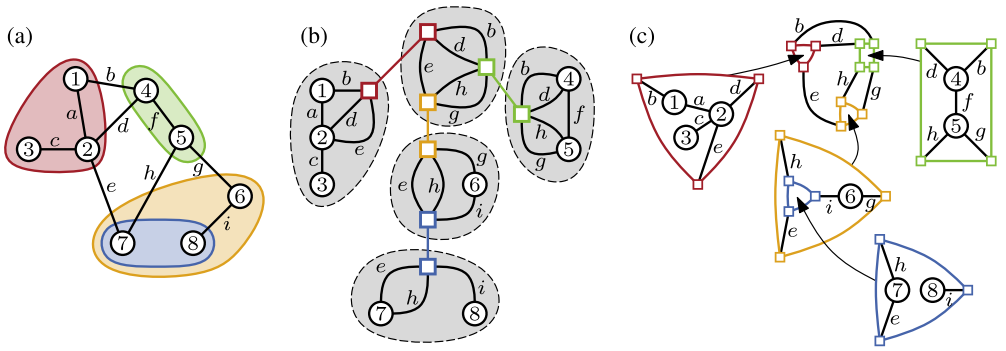

Fig. 2. (a) A c-planar drawing of a clustered graph. (b) The corresponding (rooted) cd-tree (without leaves). The skeletons are drawn inside their corresponding (gray) nodes. Note that each cluster/cut in the graph corresponds to an edge of the cd-tree as indicated by the colors. Every pair of twins (boxes with the same color) has the same edge-ordering. (c) Construction of a c-planar drawing from the cd-tree. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

3.1. 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 planarly 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 $\varepsilon_1, \dots, \varepsilon_k$ connecting μ to its neighbors $\mu_1, \dots, \mu_k$, 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 sets 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 $i = 1, \dots, k$. 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 the skeleton $\text{skel}(\mu)$ together with a planar embedding. Moreover, the edge-ordering of the vertex ν_i is the same as the order in which the edges cross the curve C_i, when traversing C_i in clockwise direction. 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-orderings of every virtual vertex and its twin are the reverse of each other. 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 ν_i of $\text{skel}(\mu)$ with edge-ordering $e_1, \dots, e_\ell$. We replace ν_i by a cycle $C_i = (\nu_i^1, \dots, \nu_i^\ell)$ and attach the edge e_j to the vertex ν_i^j; see Fig. 2c. Recall that $\text{twin}(\nu_i)$ has in $\text{skel}(\mu_i)$ the same incident edges $e_1, \dots, e_\ell$ and they also appear in this order around $\text{twin}(\nu_i)$. We also replace $\text{twin}(\nu_i)$ by a cycle of length ℓ. We say that this cycle is the *twin* of C_i and denote it by $\text{twin}(C_i) = (\text{twin}(\nu_i^1), \dots, \text{twin}(\nu_i^\ell))$ where $\text{twin}(\nu_i^j)$ 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 as a subgraph. Second, for every cut corresponding to an edge $\varepsilon = \{\mu, \mu_i\}$ in T, the graph G^+ contains the cycle C_i, whose vertices are the subdivision vertices of exactly those edges of G whose endpoints are separated by the cut corresponding to ε. 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. □

3.2. Cutvertices in skeletons

We show that cutvertices in skeletons correspond to different connected components in a cluster or in a co-cluster. More precisely, a cutvertex directly implies disconnectivity, while the opposite is not true. Consider the example in Fig. 2. The cutvertex in the skeleton containing the vertices 7 and 8 corresponds to the two connected components in the blue cluster (containing 7 and 8). However, the two connected components in the orange cluster (containing 6–8) do not yield a cutvertex in the skeleton containing the vertex 6. The following lemma in particular shows that requiring every cluster to be connected implies that the parent vertices of skeletons cannot be cutvertices.

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, the removal of the cutvertex v yields (at least) one different connected component for each of the blocks incident to v. □

While the converse of Lemma 1 is generally not true, it holds if the condition is satisfied for all parent vertices in all skeletons simultaneously.

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 a subset of O_S (only the orders in the subset are *allowed*, e.g., all orders given by a fixed PQ-tree). A *family of S-order-constraints* is a set of order constraints for the set S, i.e., a subset of the power set of O_S (e.g., all order-constraints that can be described by PQ-trees with leaf set S). A *family of order-constraints* (independent of the set S) contains for each set S a family of S-order constraints (e.g., for each set S all orders that can be specified by PQ-trees with leaf set S). We say that a family of order-constraints has a *compact representation*, if for every set S one can specify every order-constraint in its family of S-order-constraints with polynomial space in $|S|$ (e.g., because it can be described by a PQ-tree). In the following we describe families of order-constraints with compact representations.

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. Given a partition $S = S_1 \cup \dots \cup S_k$, a *partition-constraint* is an order-constraint composed of all elements o of O_S such that no two partitions S_i and S_j alternate in o. Note that a PQ-tree can specify an order only up to reversal and thus a full-constraint is not a special case of a PQ-constraint. Moreover, a partition-constraint of a set of six elements partitioned into three sets of size two each cannot be represented by a PQ-tree. Thus partition-constraints and PQ-constraints are different concepts.

It is further possible to combine these types of constraints. 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 combination of partition-constraints with other order-constraints can be defined for any type of order-constraints. Note that the families of order constraints of all given examples have compact representations as they can be described by giving an ordering (full-constraint), a partition (partition-constraint), a PQ-tree (PQ-constraint) or a partition together with either orders of total linear size (partitioned full-constraints) or a set of PQ-trees with total linear size (partitioned PQ-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 $v_1, \dots, v_k$ be the virtual vertices in $\text{skel}(\mu)$ corresponding to the children $\mu_1, \dots, \mu_k$ of μ. Note that $\text{skel}(\mu_i)$ contains exactly one virtual vertex, namely $\text{twin}(v_i)$. The possible ways to embed $\text{skel}(\mu_i)$ restrict the possible edge-orderings of $\text{twin}(v_i)$ and thus, by the characterization in Theorem 1, the edge-orderings of v_i in $\text{skel}(\mu)$. Hence, the graph $\text{skel}(\mu_i)$ essentially yields an order-constraint for v_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 $\text{skel}(\mu_i)$.

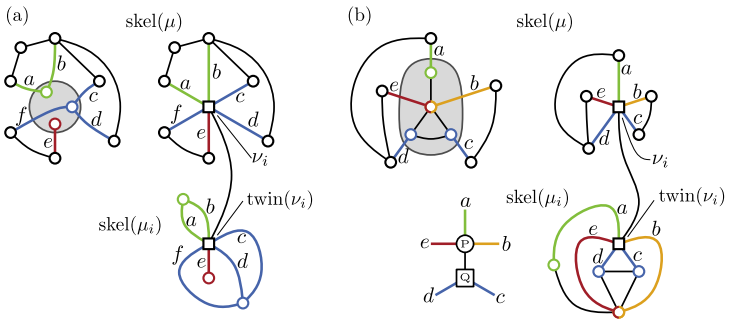

Fig. 3. (a) A graph with a single cluster consisting of isolated vertices together with an illustration of its cd-tree. An edge-ordering of twin(ν_i) corresponds to a planar embedding of skel(μ_i) if and only if no two partitions of the partitioning $\{\{a, b\}, \{c, d, f\}, \{e\}\}$ alternate. (b) A graph with a single connected cluster and its cd-tree. The valid edge-orderings of twin(ν_i) are represented by the shown PQ-tree.

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. 3a. Consider a flat-clustered graph G and let μ_i be one of the leaves of the cd-tree. As pert(μ_i) is a proper cluster, it consists of isolated vertices. Thus, skel(μ_i) is a set of vertices $v_1, \dots, v_\ell$, each connected (with multiple 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. Note that for flat-clustered graphs the cd-tree has size linear in the size of G.

Conversely, given a planar multi-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 cluster 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 to the same vertex if and only if they belong to the same partition. Clearly, this construction can be performed in linear time.

Case (ii) is illustrated in Fig. 3b. 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 of the theorem follows from the fact that the possible edge-orderings of non-cutvertices are 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 Section 2).

In case (iii) the embedding of G is fixed. 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. Moreover, the fixed embedding of G fixes the edge-ordering of non-virtual vertices and thus it 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. □

Recall that we assume the embedding of a graph to be fixed if the edge-ordering of every vertex is given. For disconnected graphs, this deviates from the commonly used notion of an embedding, where not only the edge-orderings but also the relative positions of different connected components to each other are fixed. Assume that the graph G itself is disconnected and not only the edge-orderings but also the relative positions are fixed. Consider a leaf μ_i of the cd-tree. Then the fixed relative positions for the components of G may prescribe how two blocks incident to the virtual vertex in skel(μ_i) have to be nested. Thus, the partitioned full-constraints we get in case (iii) of the above theorem are further restricted.

4.1.1. Related work

Biedl [11] 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. [12] show that one can test for the existence of HH-drawings in linear time. Hong and Nagamochi [13] 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 only two virtual vertices. By Theorem 2, this is equivalent to testing planarity with partitioned PQ-constraints for multi-graphs with only two vertices. 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 (with an absolutely non-trivial algorithm) on a set of parallel edges. This indicates that we are still far away from solving the c-planarity problem even for flat-clustered graphs.

Cortese et al. [14] 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. The requirement that G is a cycle implies that the skeleton of each non-root node μ of T has the property that in $\text{skel}(\mu)$ the blocks incident to the parent vertex are simple cycles or double edges. 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. [15] 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. [16] 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 always be 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 that 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. We now discuss the relationship between our model and these results.

In case G is 3-connected, its planar embedding is fixed and thus the edge-ordering of non-virtual vertices is fixed. Hence, given that each cluster has size three, 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 this kind of constraints seem pretty simple to handle, the algorithm by Jelínková et al. is quite involved. It seems like one barrier where constrained embedding problems become 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.

Chimani et al. [17] give a polynomial-time algorithm for embedded flat-clustered graphs with the additional requirement that each face is incident to at most two vertices of the same cluster. This basically solves planarity with partitioned full-constraints with some additional requirements. We do not describe how these additional requirements exactly restrict the possible instances of constrained planarity. However, we give some properties that shed a light on why these requirements make planarity with partitioned full-constraints tractable.

Consider the skeleton $\text{skel}(\mu)$ of a (non-root) node μ of the cd-tree. As G is a flat-clustered graph, $\text{skel}(\mu)$ has a single virtual vertex. Assume we choose planar embeddings with consistent edge orderings for all skeletons (i.e., we have a c-planar embedding of G). Two non-virtual vertices u and v in $\text{skel}(\mu)$ that are incident to the same face of $\text{skel}(\mu)$ are then also incident to the same face of G. Note that the converse is not true, as two vertices that share a face in G may be separated in $\text{skel}(\mu)$ due to the contraction of disconnected subgraphs. As the non-virtual vertices of $\text{skel}(\mu)$ belong to the same cluster μ, at most two of them can be incident to a common face of $\text{skel}(\mu)$. Thus, every face of $\text{skel}(\mu)$ has at most two non-virtual vertices on its boundary.

This implies that the possible ways how the blocks incident to the virtual vertex of $\text{skel}(\mu)$ can be nested into each other is heavily restricted. In particular, embedding multiple blocks next to each other into the same face of another block is not possible, as this would result in a face of $\text{skel}(\mu)$ with more than two non-virtual vertices on its boundary. In a sense, this enforces a strong nesting of the blocks. Thus, one actually obtains a variant of planarity with partitioned full-constraints, where the way how the partitions can nest is restricted beyond forbidding two partitions to alternate. These and similar restrictions on how partitions are allowed to be nested lead to a variety of new constrained planarity problems. We believe that studying these restricted problems can help to deepen the understanding of the more general partitioned full-constraints or even partitioned PQ-constraints.

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; also see Fig. 4. 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*.

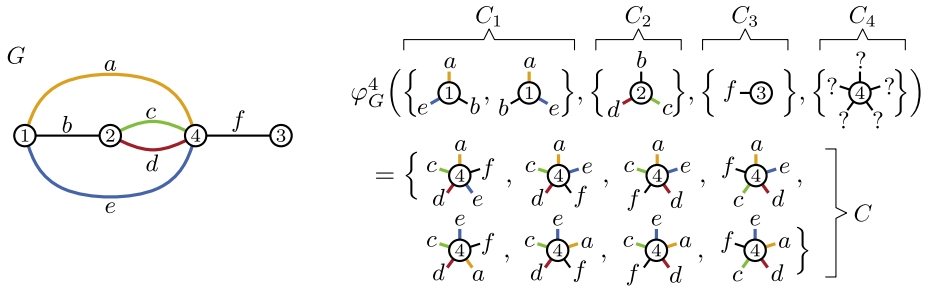

Fig. 4. A graph G with vertices $\{1,\dots,4\}$ and edges $\{a,\dots,f\}$ (left). The constrained-embedding operation φ_G^4 (for the graph G with the designated vertex 4) evaluated for the order constraints $C_1,\dots C_4$, resulting in the order constraint C of the designated vertex 4 (right). The constraint C_1 allows both possible edge-orderings for the vertex 1; for vertex 2, only one of the two orders is allowed; vertex 3 has a unique edge-ordering; C_4 allows all $4! = 24$ edge-orderings of the vertex 4. When embedding G, respecting the order constraints $C_1,\dots,C_4$, the edges a, c, d, and e have either the cyclic (clockwise) order $aedc$ or $eadc$. The edge f can be added at an arbitrary position in these orders, resulting in the eight possible edge-orderings for the vertex 4 allowed by C.

The issue mentioned above (with constraints iteratively handed to the 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^v_G 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, and*
- *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 the input of φ^v_G consists of only k order constraints (whose sizes are bounded by a function of d), then φ^v_G 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 [18] 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. We choose $\mathcal{C}$ as the PQ-constraints, which are closed under $\mathcal{R}$-restricted constrained-embedding operations as the valid edge-orderings of non-cutvertices are PQ-representable and planarity with PQ-constraints is basically equivalent to planarity (one can model a PQ-tree with a simple gadget; see Section 2). Thus, Theorem 3 directly implies that c-planarity can be solved in polynomial time if each cluster is connected.

4.2.1. Related work

The above algorithm for clustered graphs with connected clusters resulting from Theorem 3 is more or less the one described by Lengauer [1]. The algorithm was later rediscovered by Feng et al. [2] 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 [19] improves the running time to $O(n)$. Cortese et al. [20] give a characterization that also leads to a linear-time algorithm.

Goodrich et al. [21] 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 pert(μ) by 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. [22]. 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. [5,4] 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 [8]. 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 [8]. 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) [23].

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 ν be a virtual vertex in skel(μ). By the above properties, the embedding representation of μ contains the embedding tree $T(\nu)$ representing the valid edge-orderings of ν. Moreover, for twin(ν) there is an embedding tree $T(\text{twin}(\nu))$ in the embedding representation of the skeleton containing twin(ν). To ensure that ν and twin(ν) have the same edge-ordering, one can simply add a PQ-tree as a common child of $T(\nu)$ and $T(\text{twin}(\nu))$. 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. Conversely, every such set of embeddings yields a solution for D. It thus follows by the characterization in Theorem 1 that solving c-planarity is equivalent to solving D. The size of D is linear in the size c of the cd-tree T. Moreover, solving Simultaneous PQ-Ordering for 2-fixed instances can be done in quadratic time [8], yielding the running time $O(c^2)$. □

Theorem 4 includes the following interesting cases. The latter extends the result by Jelínek et al. [6] 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.*

Proof. Note that the expansion graphs of nodes in skeletons of T are exactly the clusters and co-clusters. Thus, the expansion graphs consist of at most two connected components. By Lemma 1 the cutvertices in skeletons of T are incident to at most two different blocks. Thus, we can simply apply Theorem 4. □

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

Proof. Let μ be a node with virtual vertex ν in its skeleton. The edges incident to ν in skel(μ) are exactly the edges that separate exp(ν) from the rest of the graph exp(twin(ν)). Thus, if every cluster has at most five outgoing edges, the virtual vertices in skeletons of T have maximum degree 5. With five edges incident to a vertex ν, one cannot get more than two non-trivial blocks incident to ν. It follows from Theorem 4 that we can test c-planarity in $O(c^2)$ time. As we have a linear number of cuts, each of constant size (at most 5), we get $c \in O(n)$. □

6. Conclusion

In this paper we have introduced the cd-tree and we have shown that it can be used to reformulate the classic c-planarity problem as a constrained embedding problem. Afterwards, we interpreted several previous results on c-planarity from this new perspective. In many cases the new perspective simplifies these algorithms or at least gives a better intuition why the imposed restrictions are helpful towards making the problem tractable. In some cases, the new view allowed us to generalize and extend previous results to larger sets of instances.

We believe that the constrained embedding problems we defined provide a promising starting point for further research, e.g., by studying restricted variants to further deepen the understanding of the c-planarity problem.

References

[1] T. Lengauer, Hierarchical planarity testing algorithms, J. ACM 36 (3) (1989) 474–509.
[2] Q.-W. Feng, R.F. Cohen, P. Eades, Planarity for clustered graphs, in: P. Spirakis (Ed.), Proceedings of the 3rd Annual European Symposium on Algorithms, ESA'95, in: Lecture Notes in Computer Science, vol. 979, Springer, Berlin/Heidelberg, 1995, pp. 213–226.
[3] S. Cornelsen, D. Wagner, Completely connected clustered graphs, Discrete Appl. Math. 4 (2) (2006) 313–323.
[4] V. Jelínek, E. Jelínková, J. Kratochvíl, B. Lidický, Clustered planarity: embedded clustered graphs with two-component clusters (extended abstract), in: I.G. Tollis, M. Patrignani (Eds.), Proceedings of the 16th International Symposium on Graph Drawing, GD'08, in: Lecture Notes in Computer Science, vol. 5417, Springer, Berlin/Heidelberg, 2009, pp. 121–132.
[5] V. Jelínek, E. Jelínková, J. Kratochvíl, B. Lidický, Clustered planarity: embedded clustered graphs with two-component clusters, manuscript, http://kam.mff.cuni.cz/~bernard/pub/flat.pdf, 2009.
[6] V. Jelínek, O. Suchý, M. Tesař, T. Vyskočil, Clustered planarity: clusters with few outgoing edges, in: I.G. Tollis, M. Patrignani (Eds.), Proceedings of the 16th International Symposium on Graph Drawing, GD'08, in: Lecture Notes in Computer Science, vol. 5417, Springer, Berlin/Heidelberg, 2009, pp. 102–113.
[7] G. Di Battista, F. Frati, Efficient c-planarity testing for embedded flat clustered graphs with small faces, J. Graph Algorithms Appl. 13 (3) (2009) 349–378.
[8] T. Bläsius, I. Rutter, Simultaneous PQ-ordering with applications to constrained embedding problems, in: Proceedings of the 24th Annual ACM–SIAM Symposium on Discrete Algorithms, SODA'13, Society for Industrial and Applied Mathematics, 2013, pp. 1030–1043.
[9] K.S. Booth, G.S. Lueker, Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms, J. Comput. System Sci. 13 (3) (1976) 335–379.
[10] G. Di Battista, R. Tamassia, On-line maintenance of triconnected components with SPQR-trees, Algorithmica 15 (4) (1996) 302–318.
[11] T. Biedl, Drawing planar partitions I: LL-drawings and LH-drawings, in: Proceedings of the 14th Annual Symposium on Computational Geometry, SoCG'98, ACM Press, 1998, pp. 287–296.
[12] T. Biedl, M. Kaufmann, P. Mutzel, Drawing planar partitions II: HH-drawings, in: J. Hromkovič, O. Sýkora (Eds.), Proceedings of the 24th Workshop on Graph-Theoretic Concepts in Computer Science, WG'98, in: Lecture Notes in Computer Science, vol. 1517, Springer, Berlin/Heidelberg, 1998, pp. 124–136.
[13] S.-H. Hong, H. Nagamochi, Simpler algorithms for testing two-page book embedding of partitioned graphs, in: Z. Cai, A. Zelikovsky, A. Bourgeois (Eds.), Proceedings of the 20th International Symposium on Computing and Combinatorics, COCOON'14, in: Lecture Notes in Computer Science, vol. 8591, Springer, 2014, pp. 477–488.
[14] P.F. Cortese, G. Di Battista, M. Patrignani, M. Pizzonia, Clustering cycles into cycles of clusters, J. Graph Algorithms Appl. 9 (3) (2005) 391–413.
[15] P.F. Cortese, G. Di Battista, M. Patrignani, M. Pizzonia, On embedding a cycle in a plane graph, Discrete Math. 309 (7) (2009) 1856–1869.
[16] E. Jelínková, J. Kára, J. Kratochvíl, M. Pergel, O. Suchý, T. Vyskočil, Clustered planarity: small clusters in cycles and Eulerian graphs, J. Graph Algorithms Appl. 13 (3) (2009) 379–422.
[17] M. Chimani, G. Di Battista, F. Frati, K. Klein, Advances on testing c-planarity of embedded flat clustered graphs, in: C. Duncan, A. Symvonis (Eds.), Proceedings of the 22nd International Symposium on Graph Drawing, GD'14, in: Lecture Notes in Computer Science, vol. 8871, Springer, Berlin/Heidelberg, 2014, pp. 416–427.
[18] M. Chimani, K. Klein, Shrinking the search space for clustered planarity, in: W. Didimo, M. Patrignani (Eds.), Proceedings of the 20th International Symposium on Graph Drawing, GD'12, in: Lecture Notes in Computer Science, vol. 7704, Springer, Berlin/Heidelberg, 2013, pp. 90–101.
[19] E. Dahlhaus, A linear time algorithm to recognize clustered planar graphs and its parallelization, in: C.L. Lucchesi, A.V. Moura (Eds.), Proceedings of the 3rd Latin American Symposium, LATIN'98, in: Lecture Notes in Computer Science, vol. 1380, Springer, Berlin/Heidelberg, 1998, pp. 239–248.
[20] P.F. Cortese, G. Di Battista, F. Frati, M. Patrignani, M. Pizzonia, c-Planarity of c-connected clustered graphs, J. Graph Algorithms Appl. 12 (2) (2008) 225–262.
[21] M.T. Goodrich, G.S. Lueker, J.Z. Sun, c-Planarity of extrovert clustered graphs, in: P. Healy, N.S. Nikolov (Eds.), Proceedings of the 13th International Symposium on Graph Drawing, GD'05, in: Lecture Notes in Computer Science, vol. 3843, Springer, Berlin/Heidelberg, 2006, pp. 211–222.
[22] C. Gutwenger, M. Jünger, S. Leipert, P. Mutzel, M. Percan, R. Weiskircher, Advances in c-planarity testing of clustered graphs, in: M.T. Goodrich, S.G. Kobourov (Eds.), Proceedings of the 10th International Symposium on Graph Drawing, GD'02, in: Lecture Notes in Computer Science, vol. 2528, Springer, Berlin/Heidelberg, 2002, pp. 220–235.
[23] T. Bläsius, I. Rutter, Simultaneous PQ-ordering with applications to constrained embedding problems, Computing Research Repository, arXiv:1112.0245 [abs], 2011, 46 pp.