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/

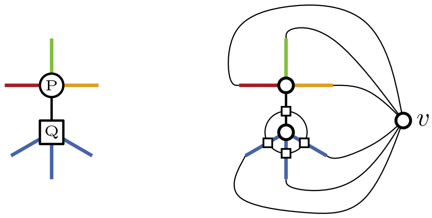

Fig. 1. Construction of a graph G where the edge-orderings of v in embeddings of G are represented by a given PQ-tree T. The boundary vertices of the wheel representing the Q-node are shown as squares.

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