TECHNISCHE UNIVERSITÄT WIEN
VIENNA UNIVERSITY OF TECHNOLOGY

DIPLOMARBEIT

Cluster Planarity Testing for the Case of Not Necessarily Connected Clusters

ausgeführt am

Institut für Computergraphik und Algorithmen
der Technischen Universität Wien

unter der Anleitung von

Univ.-Prof. Dipl.-Ing. Dr.techn. Günther Raidl
Dr.-Ing. René Weiskircher
Dipl.-Math. Merijam Percan

durch

Robert Beibl
Matrikelnr. 9226194
Polletstraße 15
1220 Wien

Datum

Unterschrift

Abstract

The central topic of this thesis are criteria and tests which reveal whether a given clustered graph allows an embedding in the plane for which no edges and clusters intersect.

Together with their definition in 1996, a notion of planarity was presented for clustered graphs, as well as an algorithm which tests this planarity for a given clustered graph in linear time. The algorithm however expects each cluster to be connected. For general clustered graphs, no efficient algorithm is yet known, neither is the computational complexity of the problem.

This work presents algorithms which extend the class of clustered graphs for which planarity can be tested in polynomial time.

A second part considers a weak form of planarity, and shows that a polynomial time test for this form also yields a polynomial time test for the classical definition.

Furthermore, an attempt is made, by means of a characterization of the weak realizability problem in terms of forbidden subgraphs, to gain a similar characterization of the weak form of cluster planarity.

Zusammenfassung

Das zentrale Thema dieser Diplomarbeit sind Kriterien und Tests, die angeben, ob es für bestimmte Clustergraphen Einbettungen in die Ebene gibt, für die sich keine Kanten und Cluster überschneiden.

Zugleich mit der Einführung des Begriffs des Clustergraphen im Jahr 1996 wurde eine Definition von Planarität für Clustergraphen vorgestellt, sowie ein Algorithmus, der für einen gegebenen Clustergraphen in linearer Zeit prüft, ob er planar ist. Dieser Algorithmus setzt jedoch voraus, daß die einzelnen Cluster zusammenhängend sind. Für allgemeine Clustergraphen ist derzeit weder ein effizienter Algorithmus zur Durchführung eines solchen Tests bekannt, noch, welche Komplexität das Problem hat.

Die vorliegende Arbeit präsentiert Algorithmen, die die Klasse der Clustergraphen, für die Planarität in polynomieller Zeit testbar ist, um neue Typen erweitert.

In einem zweiten Teil wird ein abgeschwächter Planaritätsbegriff für Clustergraphen untersucht, und gezeigt, daß ein polynomieller Planaritätstest für diesen auch einen für den klassischen Planaritätsbegriff liefert.

Weiters wird versucht, mithilfe einer Charakterisierung des “Weak Realizability Problem” anhand von verbotenen Teilgraphen auch den abgeschwächten Clusterplanaritäts-Begriff durch eine solche Formulierung zu charakterisieren.

Herzlichen Dank...

... all jenen, die entscheidend zum Gelingen dieser Arbeit beigetragen haben.

In besonderer Weise gilt dieser meinem Betreuer René Weiskircher, der mein Interesse für die Thematik geweckt und in sinnvolle Bahnen gelenkt hat, und dessen Unterstützung sich auch von längeren berufsbedingten Unterbrechungen meinerseits und räumlicher Distanz nicht hat stören lassen.

Nicht minder meiner Betreuerin Merijam Percan, die mich am aktuellen Stand der Entwicklung im Umfeld der Arbeit teilhaben ließ, und deren fachlichem Rat und Beistand es zuzuschreiben ist, daß aus all den in den letzten Jahren zusammengetragenen Bruchstücken und Bausteinen letztendlich eine Diplomarbeit entstanden ist.

Ebenso bedanken möchte ich mich bei Professor Günther Raidl, ohne dessen bereitwillige Unterstützung die Fertigstellung dieser Arbeit nicht möglich gewesen wäre. Weiters bei Harald Goldenits, mithilfe dessen aufmerksamen Auge es hoffentlich gelungen ist, Fehler jeglicher Art auf ein Minimum zu beschränken.

Nicht zuletzt gebührt natürlich Dank für Unterstützung und ertragene Unbill meinen Eltern, die mir Bildung und Studium ermöglicht haben, und generell ein entscheidender Faktor dafür sind, daß diese Arbeit entstehen konnte.

In gleicher Weise auch meiner Freundin Vroni, die als moralische Stütze einen diplomarbeitsbedingt oft nur nur physisch Präsenten zu ertragen hatte, sowie all jenen, deren teils aufmunternde Bemerkungen die Motivation zur Beendigung des Studiums ausreichend hoch haben werden lassen.

Contents

Part I

Introduction

Chapter 1

Motivation

Graph drawing is a rather young branch of mathematics. Brought on by increasing availability of computers, it has become interesting to study how large graphs can be visualized, manually or automatically, respecting criteria of aesthetics and readability.

Graphs themselves are structures studied for a much longer time, which allow to model relationships between objects, and to analyze these relationships in a formal way. Graphs are used to investigate problems in a variety of areas, as different as:

- Social sciences
- Computer sciences (information retrieval, knowlegde bases, workflows)
- Electrical engineering (VLSI design)
- Natural sciences (geographic information systems)
- Construction (cabling, piping)
- ...

Basically, a graph consists of *nodes* and *edges* joining nodes. By reducing a real-life problem to this abstract model, tools developed generally for graphs can be applied to answer questions to the actual problem (e.g. an algorithm finding the shortest path from one node to another can find the cheapest way to fly from one city to another).

As much as graphs can be used to analyze the structure of given real-life problems, they can also be used to visualize these problems, in order to reveal properties otherwise not immediately evident to the human eye.

One of the standard visualizations of graphs draws nodes as points in the plane, and edges as lines joining the points. Several criteria influence the readability of such a drawing, such as:

- size of the drawing
- proportions
- number of line crossings
- number of bends in the lines

In some cases, the real-life problem even consists of checking certain drawing properties: For VLSI design, a circuit can be manufactured as a one-layer plate only if the graph constructed from its electronical elements and the paths joining them can be drawn without any edge-crossings. Such a graph drawing without any edge-crossing is called *planar*. Correspondingly, a graph which has such a drawing is also called *planar*.

The central topic of this work is a special kind of planarity defined on an extension of the traditional graph model, called *clustered graphs*: This extension allows to additionally group nodes together in *clusters*, which are usually visualized by drawing them inside some closed region, possibly with a border around it.

Assigning nodes to a cluster could e.g. be used to convey that certain persons in a social relation map belong to a specific group, that certain computers are located in the same building, that airports are in the same country or belong to the same company, or that some electronic elements should be placed next or near to each other.

Therefore, the question whether a certain clustered graph is cluster planar can influence its readability, or even decide whether an electronic circuit can or cannot be printed on a one-sided plate, respecting certain proximity constraints.

Some algorithms have been devised to test cluster planarity, however, they only apply to a restricted set of graphs. Even more, it has not yet been established whether a polynomial time algorithm exists, or whether the problem is $\mathcal{NP}$-hard for arbitrary clustered graphs.

This work tries to take some new approaches to the topic, both in terms of a characterization of cluster planar graphs, and in providing cluster planarity test algorithms for a wider range of graphs.

Chapter 2

Preliminaries

2.1 General

In graph theory, unfortunately there are some terms which are not consistently defined in literature (examples include whether "graphs" are simple, directed, of whether embeddings are implicitly required to be planar, or even what a subgraph is meant to be). Therefore, this chapter shall state what definitions were used in this work, and also define other terms used in the following.

Let $P_2(S)$ denote the set of all 2-element subsets of S. An element of $P_2(S)$ representing $\{s_1, s_2\}$ with $s_1, s_2 \in S$ is denoted by $\langle s_1, s_2 \rangle$ or $\langle s_2, s_1 \rangle$.

2.2 Graphs

Definition 2.2.1. An *undirected graph* $G = (V, E)$ is a pair of sets V and E with $E \subseteq P_2(V)$. The elements of V are called the *nodes* or *vertices* of G, or $V(G)$, and the elements of E the *edges* of G, or $E(G)$. An undirected graph is called *finite* if both V and E are finite.

Note that by this definition there is at most one edge for any two nodes (which other authors would refer to as *simple undirected graphs*), and that there cannot be edges joining a node to itself (often called *self-loops*).

In this work, unless stated otherwise, all mentions of "graph" pertain to finite undirected graphs according to the above definition.

Definition 2.2.2. In a graph $G = (V, E)$ a node $v \in V$ is called *incident* with an edge $e \in E$ if $e = \langle v, w \rangle$ for some $w \in V$. Two nodes v_1, v_2 are called *adjacent* if $\langle v_1, v_2 \rangle \in E$, and two edges e_1, e_2 are called *adjacent* if they are incident with a common $v \in V$. The set of all edges incident with a node v is denoted by $inc(v)$.

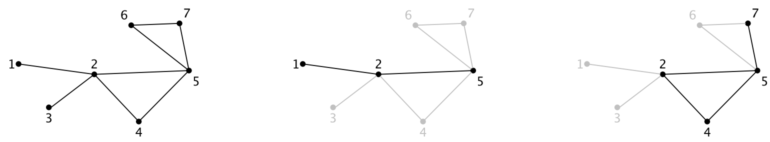


Figure 2.1: A graph G, a subgraph S of G, and $G[\{2, 4, 5, 7\}]$

Definition 2.2.3. A graph $S = (V', E')$ is a *subgraph* of a graph G, denoted $S \subseteq G$, if $V' \subseteq V$, and $E' \subseteq E(G) \cap P_2(V')$. $G[V']$ is the graph G *restricted to* V', $G[V'] = (V', E \cap P_2(V'))$, and $G \setminus X := G[V(G) \setminus V(X)]$. A *cut edge* of a subgraph S of G is an edge $\langle x, y\rangle$ with $x \in S, y \in G \setminus S$.

Definition 2.2.4. A *path* is a graph $W = (V, E)$ with $V = \{x_0, x_1, \ldots, x_k\}$, $k > 0$, and $E = \{\langle x_0, x_1\rangle, \ldots, \langle x_{k-1}, x_k\rangle\}$. x_0 and x_k are called the *endpoints* of W. A path *from x to y* is a path with endpoints x and y, and a X-path is a path for which exactly its endpoints are in X, $W \cap X = \{x_0, x_k\}$.

Definition 2.2.5. A *cycle* is a graph $C = (V, E)$ with $V = \{x_0, x_1, \ldots, x_k\}$, $k > 0$, and $E = \{\langle x_0, x_1\rangle, \ldots, \langle x_{k-1}, x_k\rangle, \langle x_k, x_0\rangle\}$.

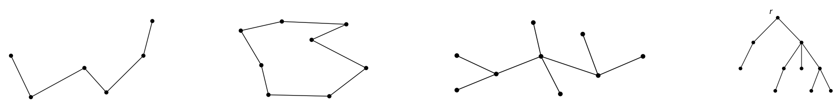


Figure 2.2: In this ordering: A path, a cycle, a tree, and a rooted tree.

Definition 2.2.6. A graph $G = (V, E)$ is called *connected* if for any nodes $v_1, v_2 \in V, v_1 \neq v_2$, there is a path in G from v_1 to v_2. G is called *k-connected* if $|V(G)| > k$ and $G \setminus X$ is connected for all $X \subseteq G$ with $|V(X)| < k$. A 2-connected graph is also called *biconnected*, and a 3-connected graph also *triconnected*. The maximal connected subgraphs of a graph are called its *components*, and the maximal biconnected subgraphs its *blocks*.

Definition 2.2.7. A *tree* T is a connected graph which does not contain a cycle (is *acyclic*). The nodes incident with at most one edge are called *leaves*, the others *interior nodes* of T. A *rooted tree* is a tree in which one node is designated *root* of the tree. In a rooted tree, all edges are interpreted to be directed, leading away from the root.

2.3 Plane Graphs

A *region* O of $\mathbb{R}^2$ is an open subset of $\mathbb{R}^2$. A *closed region* R is a region together with its border, $R = O \cup \partial O$. A (closed) region X is called *simple* if for any $x, y \in X$, x and y can be connected by a polyline in X.

Definition 2.3.1. Let $e = P(v_1, v_2)$ denote a polyline in $\mathbb{R}^2$ from v_1 to v_2, and e^o the interior of the polyline. $\mathcal{G} = (V, E)$ with $V \subseteq \mathbb{R}^2$, $|V| = n \in \mathbb{N}$, $E \subseteq \{P(v_1, v_2) | v_1, v_2 \in V, v_1 \neq v_2\}$ is called a *plane graph* if

1. $\forall e_1, e_2 \in E, e_1 \neq e_2 : \{v_1^{e_1}, v_2^{e_1}\} \neq \{v_1^{e_2}, v_2^{e_2}\}$
2. $\forall e_1, e_2 \in E, e_1 \neq e_2 : e_1^o \cap e_2 = \emptyset$

The elements of V are called *nodes* and the elements of E *edges* of the plane graph $\mathcal{G}$. The maximal simple regions of $\mathbb{R}^2 \setminus \mathcal{G}$ are called the *faces of* $\mathcal{G}$, $F(\mathcal{G})$. Edges and nodes are called incident with a face if they are contained in its border.

Each plane graph $\mathcal{G}$ immediately gives rise to a graph G by identifying nodes and edges in $\mathcal{G}$ and G, allowing to use graph definitions and properties (such as incidence, adjacence, paths, connectivity, ...) also for plane graphs.

Definition 2.3.2. Let S^n denote the n-dimensional unit sphere, and $\pi : S^2 \setminus \{(0, 0, 1)\} \mapsto \mathbb{R}^2$ a fixed homeomorphism from the 2-dimensional unit sphere with a "hole" to the plane. Plane graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ are called *topologically equivalent* if there exists an isomorphism $\sigma : \mathcal{G}_1 \mapsto \mathcal{G}_2$ which respects the incidences of nodes and edges, and a homeomorphism ϕ on S^2 such that $\pi \circ \phi \circ \pi^{-1}$ induces σ on $\mathcal{G}_1$.

This definition formalizes the most evident form of equivalence, allowing arbitrary homeomorphisms to be performed against a plane graph without leaving the equivalence class. Moreover, choosing another face as the outer face does not change the equivalence class either, due to the construction via S^2.

Definition 2.3.3. Plane graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_1, E_1)$ with faces F_1 rsp. F_2 are called *combinatorially equivalent* if there exists an isomorphism $\sigma : (V_1, E_1, F_1) \mapsto (V_2, E_2, F_2)$ which respects not only incidences of nodes with edges, but also of nodes and edges with faces.

Lemma 2.3.4. *If two plane graphs $\mathcal{G}_1, \mathcal{G}_2$ are topologically equivalent, they are combinatorially equivalent.*

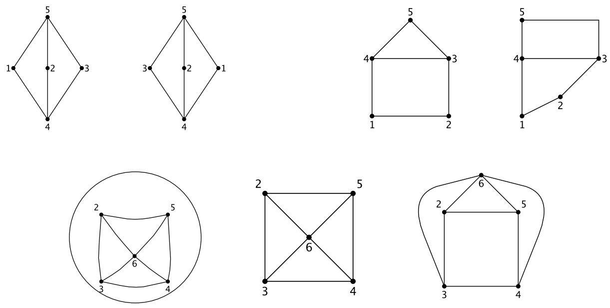


Figure 2.3: Equivalent plane graphs. For the third example, the two plane graphs are obtained from the drawing on S^2 by choosing $\{2, 3, 4, 5\}$ rsp. $\{3, 4, 6\}$ as the outer face.

Lemma 2.3.5. *If two 2-connected plane graphs $\mathcal{G}_1, \mathcal{G}_2$ are combinatorially equivalent, they are topologically equivalent. 2-connected plane graphs which are combinatorially equivalent, are called* equivalent*.*

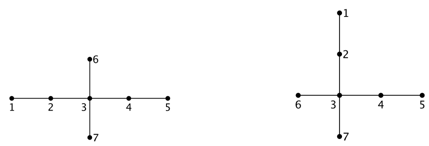


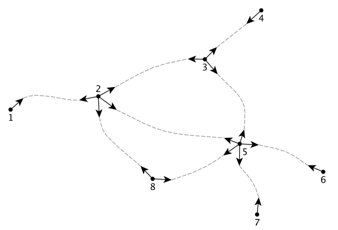
Figure 2.4: Plane graphs which are combinatorially equivalent, but not topologically equivalent.

2.4 Embeddings, Drawings

The canonical way to construct a plane graph from a graph is to map nodes to points, and edges to polylines connecting these points. Such a mapping is called an *embedding* of the graph, and the representation a *drawing* of the graph. However, such an embedding does not necessarily yield a plane graph: while condition 1) for a plane graph is trivially fulfilled by any drawing, condition 2) is not. So whether a drawing is a plane graph depends on whether $e_1^o \cap e_2 = \emptyset$ for all $e_1 \neq e_2 \in E$, i.e. whether there are no edge crossings.

A drawing for which this condition holds is called *planar*, as well as the embedding creating it. Correspondingly, a graph is called *planar* if it has a planar embedding.

Definition 2.4.1. A *combinatorial embedding* $\mathcal{E} = (V, E, \omega)$ of a graph $G = (V, E)$ is a graph together with an ordering of edges around each node, $\omega(v) \in Sym^*(inc(v))$ (with $Sym^*(X)$ denoting the permutation group of set X in which a given element is fixed). A combinatorial embedding $\mathcal{E}$ of a graph G is called planar if there is a planar drawing of G which is consistent with the orderings given in $\omega(\mathcal{E})$.



1: $\langle 1, 2\rangle$
2: $\langle 2, 1\rangle, \langle 2, 3\rangle, \langle 2, 5\rangle, \langle 2, 8\rangle$
3: $\langle 3, 5\rangle, \langle 3, 2\rangle, \langle 3, 4\rangle$
4: $\langle 4, 3\rangle$
5: $\langle 5, 3\rangle, \langle 5, 6\rangle, \langle 5, 7\rangle, \langle 5, 8\rangle, \langle 5, 2\rangle$
6: $\langle 6, 5\rangle$
7: $\langle 7, 5\rangle$
8: $\langle 8, 5\rangle, \langle 8, 2\rangle$

Figure 2.5: A combinatorial embedding. For each node, the ordering of the incident edges is fixed.

Let the set of combinatorial embeddings of G be called $\bar{\mathcal{E}}(G)$, and let $-\mathcal{E}$ denote the combinatorial embedding obtained from $\mathcal{E}$ by reversing the ordering of the edges around each $v \in V(G)$.

Lemma 2.4.2. *For connected graphs, there exists a bijection between the topological equivalence classes of planar drawings of a graph G and the set* $\{\{\mathcal{E}, -\mathcal{E}\} | \, \mathcal{E} \in \bar{\mathcal{E}}\}$*, such that the planar drawings are consistent with either* $\mathcal{E}$ *or* $-\mathcal{E}$*.*

Each homeomorphism ϕ in the definition of topological equivalence either keeps the ordering of edges around each node the same, or reverses it for all nodes.

Lemma 2.4.3. *If a graph G is 3-connected, all its planar combinatorial embeddings are equivalent.*

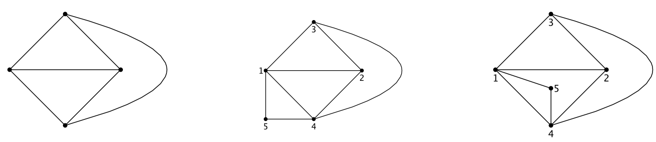


Figure 2.6: The left figure shows the only embedding (but for equivalence) allowed by its underlying triconnected graph. The figures on the right show two non-equivalent embeddings of a biconnected graph.

2.5 Minors

When testing a graph for certain properties, one would often like to know which "part" or "substructure" of the graph is responsible for a certain property. While the usual definition of a "part" is the one of a *subgraph*, for some properties other definitions of "substructures" of a graph have proven valuable.

Definition 2.5.1. If in a graph X, some edges $\langle x_1, x_2 \rangle$ are replaced by a path from x_1 to x_2, and the inners of the paths have no nodes in common with the other paths or X, the resulting graph G is called a *subdivision* of X, or $G = TX$. For any $Y \supseteq G$, X is called a *topological minor of* Y. The nodes in G corresponding to nodes in X are called the *branch vertices* of G.

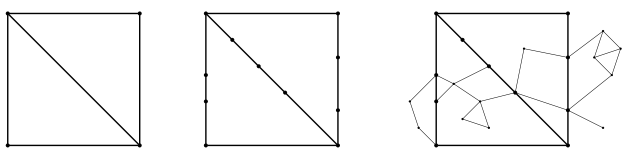


Figure 2.7: X, G, and Y where $G = TX$, and X is a topological minor of Y.

Definition 2.5.2. If $G = (V, E)$ is a graph, $P = \{V_1, V_2, ..., V_n\}$ a partition of V for which $G[V_i]$ is connected for each i, and $X = (P, E')$ is a graph with $E' = \{\langle V_i, V_j \rangle | i \neq j, \exists v_1 \in V_i, v_2 \in V_j : \langle v_1, v_2 \rangle \in E\}$, then G is called an MX (or $G = MX$), and the V_i are called the *branch sets* of G. For any $Y \supseteq G$, X is called a *minor of* Y.

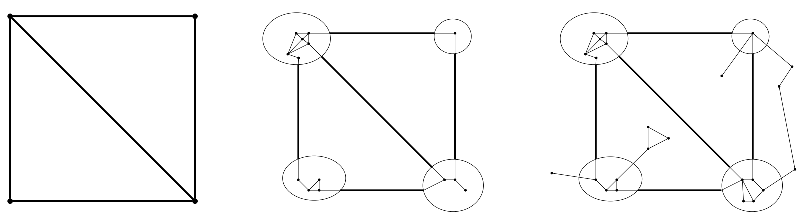


Figure 2.8: X, G, and Y where $G = MX$, and X is a minor of Y.

Lemma 2.5.3. *G is an MX if and only if X is obtained from G by successively contracting edges.*

Lemma 2.5.4. *If $G = TX$, then $G = MX$. Therefore any topological minor of Y is also a minor of Y.*

Lemma 2.5.5. *Being a minor is a partial order on the set of finite graphs. The same holds for being a topological minor.*

Therefore, if A is a (topological) minor of B, and B is a (topological) minor of C, then A is also a (topological) minor of C. Moreover, if A is a (topological) minor of B, and B is a (topological) minor of A, $A = B$. Every graph is a (topological) minor of itself, and any subgraph A of B is a (topological) minor of B.

Chapter 3

Graph Extensions for Clustering

3.1 Overview

Even though the classical graph model is already powerful, for some applications it does not suffice to consider relations between two objects, but to additionally store the information that a group of nodes belongs together or shares some common property.

To model such information, various extensions to the classical graph model have been created, with different aims and different possibilities.

3.2 Hypergraphs

Hypergraphs [Ber73] allow to assign more than two nodes to an edge, thereby allowing arbitrary connections between several nodes. Based on this model, more general ones such as *higraphs* [Har88] have been designed. While suitable for a wide range of problems, they are also hard to draw automatically.

3.3 Compound Graphs

In [MS93], an extension to graphs was introduced which allows to add hierarchical information, by designating some nodes to be "included" in others.

Definition 3.3.1. A *compound graph* is a triple $D = (V, E, I)$ such that $D_a = (V, E)$ is a graph and $D_c = (V, I)$ is a directed graph. The elements of E are called *adjacency edges*, those of I *inclusion edges*.

Usually, it is also expected that D_c contains no cycles, so that interpreting $\langle v, w\rangle \in I$ as an inclusion relation makes sense.

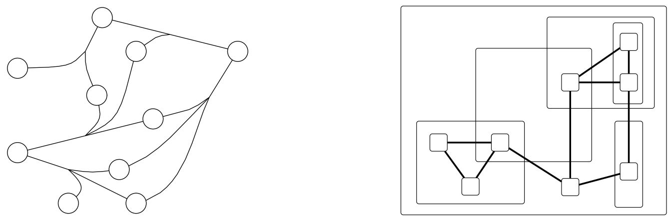


Figure 3.1: A hypergraph, and a compound graph.

3.4 Clustered Graphs

Clustered graphs have been introduced in [Feng96]: in addition to a classical graph, a clustered graph contains entities called *clusters* which contain nodes and possibly other clusters. This delivers a hierarchical structure on top of the original graph, which is most easily associated with the principle of proximity of objects: putting nodes inside a cluster can convey that these objects should be placed next to each other, preferably inside some box.

As opposed to higraphs, efficient drawing algorithms are available for some classes of clustered graphs, [Feng96], [Dah98].

Formal definition of clustered graphs:

Definition 3.4.1. Let (G, T) be called a *clustered graph* with G a graph, and T a rooted tree with leaves $V(G)$. Each non-leaf element ν of T defines a cluster C^ν consisting of all leaves having ν as ancestor.

A clustered graph can be seen as a special case of a compound graph, where D_c is a rooted tree, and adjacency edges are only incident to leaves. The V in the definition of the clustered graph is the set of leaves of D_c.

In addition to the definition of clustered graphs, [Feng96] also introduces a definition of *cluster planarity* for clustered graphs:

Definition 3.4.2. A clustered graph (G, T) is called *c-planar* if G can be drawn in the plane such that for each cluster C, there exists a simple closed region $\mathcal{G}^C \subseteq \mathbb{R}^2$ such that all $v \in C$ and all $e \in P_2(C)$ are contained in $\mathcal{G}^C$, each cut edge of C crosses the border of $\mathcal{G}^C$ exactly once, and all $v \in G \setminus C$ and $e \in P_2(G \setminus C)$ are contained in $\mathbb{R}^2 \setminus \mathcal{G}^C$.

This definition matches the picture of all contents of a cluster (and nothing else) being drawn inside some border.

Definition 3.4.3. A clustered graph (G, T) is called *c-connected* if $G[C^\nu]$ is connected for all $\nu \in T$.

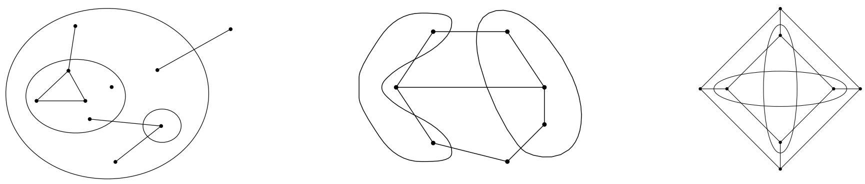


Figure 3.2: Examples of clustered graphs (straight lines are edges, other curves cluster borders). The rightmost one is not c-planar.

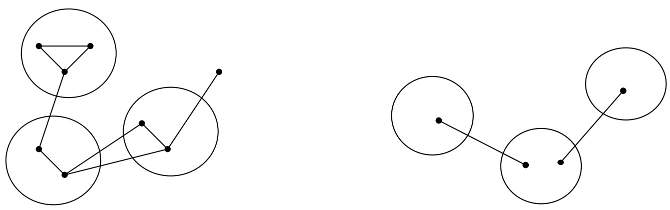


Figure 3.3: Examples of a c-connected and a non-c-connected clustered graph.

Part II

Previous Results

Chapter 4

Planarity Tests

Since this work aims at investigating the (cluster) planarity of clustered graphs, some algorithms or criteria which test planarity on classical graphs shall be presented. Attempts to extend them to clustered graphs will be made in the following chapters.

4.1 Basic Criteria

Lemma 4.1.1. *A graph is planar if and only if its blocks are planar.*

If a graph is split into its maximal biconnected subgraphs, then for each block B, B is connected to $G \setminus B$ by at most one edge, e. Therefore, any planar drawings of B and $G \setminus B$ can be combined to generate a planar drawing of G. In the case of e present, the outer faces of each drawing should first be chosen such that the corresponding endpoint of e is on the outer face.

Let a plane graph $\mathcal{G}$ be called triangular if each face of $\mathcal{G}$ is incident with exactly 3 nodes of $\mathcal{G}$.

Lemma 4.1.2. *Each triangular plane graph has exactly* $3n - 6$ *edges.*

This follows directly from Euler's Polyeder Formula, which states that for a connected plane graph $\mathcal{G}$ with $n \geq 1$ nodes, m edges and l faces, $n - m + l = 2$. Observing that for a triangular graph, $2m = 3l$, delivers the result.

Corollary 4.1.3. *A plane graph with* $n \geq 3$ *nodes has at most* $3n - 6$ *edges.*

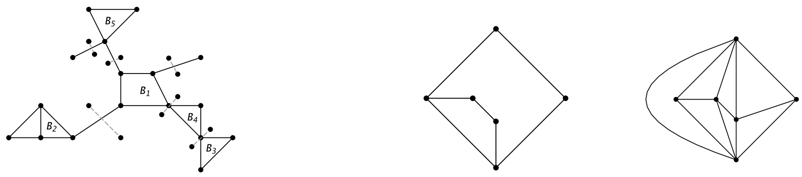


Figure 4.1: On the left: A graph is planar if and only if its blocks B_i are planar. On the right: A graph and a triangular graph which contains it.

4.2 Kuratowski's Theorem

In 1930, Kuratowski established the central characterization of planar graphs in [Kur30] by means of the *Kuratowski graphs*: K^5 is the complete graph with 5 nodes (all 5 nodes are connected pairwise), and $K_{3,3}$ is the complete bipartite graph on 6 nodes (two groups of three nodes each, where exactly every two nodes from different groups are connected).

Theorem 4.2.1. *(Kuratowski's Theorem) A graph G is planar if and only if neither K^5 nor $K_{3,3}$ is a minor of G. Equivalently, a graph G is planar if and only if neither K^5 nor $K_{3,3}$ is a topological minor of G.*

While Kuratowski's theorem gives a very comprehensible criterion for planarity, it is not easy to use for testing planarity of large graphs, since the number of minors of a graph grows exponentially.

However, it is indirectly used in most other planarity testing algorithms, since the result of such an algorithm is usually either a combinatorial embedding (which can be verified to be planar by drawing it), or pointing out a minor of the graph which is a Kuratowski graph.

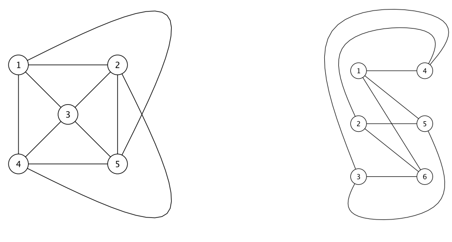


Figure 4.2: The Kuratowski subgraphs, K^5 and $K_{3,3}$.

4.3 Linear Time Planarity Tests

Starting in the 1970s, several planarity tests with linear running time have been developed. They all share that their speed is bought by complexity, which makes it tedious to prove their correctness, and to implement them correctly. They also share that they are usually defined on 2-connected graphs, since the general result can be obtained using Lemma 4.1.1.

The first one, presented in 1974, is that of Hopcroft and Tarjan [HT74], which is based on a depth first search on the graph, first creating a spanning tree for the graph, and then trying to embed the other edges .

Some corrections to the algorithm were published in [Deo76] in 1976, and several other papers deal with making the algorithm more accessible and easier to understand (e.g. [Mut96], [MM96]).

In 1976, Booth and Lueker in [BL76] presented a linear variant of a planarity test designed in 1967 by Lempel, Even and Cederbaum [LEC67], using a datastructure called PQ-trees which had been introduced to solve the *consecutive ones problem* efficiently. In short, a PQ-tree can represent all possible orderings of nodes on the outer face of a (sub)graph, and can be efficiently manipulated to respect the planarity constraints imposed by additional nodes. The algorithm will be explained in slightly more detail in chapter 4.4, since it will serve as a basis for the c-planarity test extension presented in this work.

Also for this algorithm, additional papers were needed for clarification, most of all how it should be used to yield an actual planar embedding of the graph. This was presented by Chiba et al in [CNA85].

For quite a long time, these two were basically the only linear time planarity tests available, with the exception of the algorithm by Fraysseix and Rosenstiehl [FR82] added in 1982, until only recently new algorithms were published: The algorithm of Boyer and Myrvold [BM99] and that of Shih and Hsu [SH99], which both claim to be simpler than the "classical" ones.

4.4 PQ-Tree Planarity Test

The planarity test presented in [BL76] makes use of a data structure called *PQ-trees*. A PQ-tree can be used to store all permutations of a set in which certain subsets are contiguous, and to add such constraints efficiently.

This is achieved by the following structure: A PQ-tree T on a set S is a directed rooted tree made of P-nodes, Q-nodes and leaves. The P-nodes and Q-nodes have ordered lists of children. The leaves are exactly the elements of S, and a drawing of T has all leaves horizontally aligned at the bottom, the root at the top, the children of P-nodes and Q-nodes ordered according to their lists, and no line crossings. Each such drawing corresponds to exactly one ordering of S, called a *frontier* of T.

The special properties of the PQ-tree are now obtained by allowing the following operations: The children of a P-node (which is drawn as a circle) can be arbitrarily permuted; the list of children of a Q-node (which is drawn as a rectangle) can be mirrored.

The orderings of leaves obtained from all drawings of a PQ-tree after such operations give the set of *consistent permutations* of S for T, denoted CONSISTENT(T). [BL76] describes an operation $T' = \text{REDUCE}(T, X)$, which yields a new PQ-tree T' for which the consistent permutations are reduced to those in which all elements of $X \subseteq S$ are consecutive. This operation works by detecting certain patterns in the tree, and replacing them with corresponding templates. While this operation is tedious and error-prone to implement, it is very efficient.

There are two special PQ-trees: the universal tree, consisting of a single P-node with all leaves as children (not imposing any restrictions on the set of permutations, $\text{CONSISTENT}(T) = 2^S$), and the empty tree, with $\text{CONSISTENT}(T) = \emptyset$.

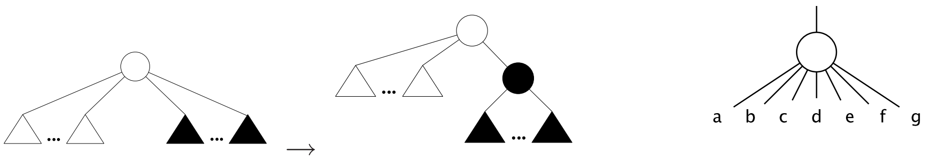


Figure 4.3: A pattern used in the PQ-tree REDUCE operation. On the right, the universal PQ-tree on the set $\{a, b, c, d, e, f, g\}$.

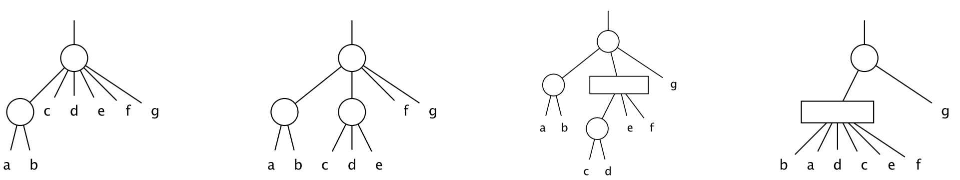


Figure 4.4: PQ-tree obtained by reducing by the sets $\{a, b\}, \{c, d, e\}, \{e, f\}, \{a, d\}$. Any front of this PQ-tree will have all elements of each set consecutive, and the permutations represented by these fronts are the only ones with this property.

The planarity test using PQ-trees now basically tries to draw the graph as a PQ-tree: each node represents some subgraph, with mandatory orderings of edges imposed by Q-nodes. It works by starting with a one-node subgraph, and extending it by repeatedly adding vertices until all of G is covered (called a *vertex addition* approach).

First, a numbering $o : V \mapsto \{1, \ldots, |V|\}$ of the nodes of G is retrieved, such that each node is connected to nodes preceding it and to nodes following it (with the exception of the first and the last node, which are required to be connected by an edge). Such a numbering is called an *st-numbering* ($s := v_1, t := v_{|V|}$), and it can always be constructed for biconnected graphs G, in linear time (for general graphs G, the PQ-tree planarity test is done on the blocks of G, with Lemma 4.1.1 yielding the overall result).

Now, subgraphs of G are considered, constructed by vertex addition according to the st-numbering: The first graph is made of s and the edges incident with it (but the one connecting it with t). This is represented by a P-node with edges to leaves labeled with the names of the nodes adjacent to s.

In each of the following steps, the next node i according to the st-numbering is considered: the PQ-tree is manipulated such that all leaves with label i (representing edges leading to this node) are always adjacent in all permutations allowed by the PQ-tree (if there were another leaf j in between, the drawing could not be planar, since the corresponding edge would have to go "through" v_i or one of the edges incident with it, since it leads to a v_j with $j > i$). When this is done, all these leaves are replaced by a single P-node for v_i and all edges leading from v_i to later nodes according to the st-numbering.

If at any time, it is not possible to perform the step (i.e., the empty tree is reported when trying to make leaves consecutive), the graph is nonplanar; if the final node t is reached, it is planar.

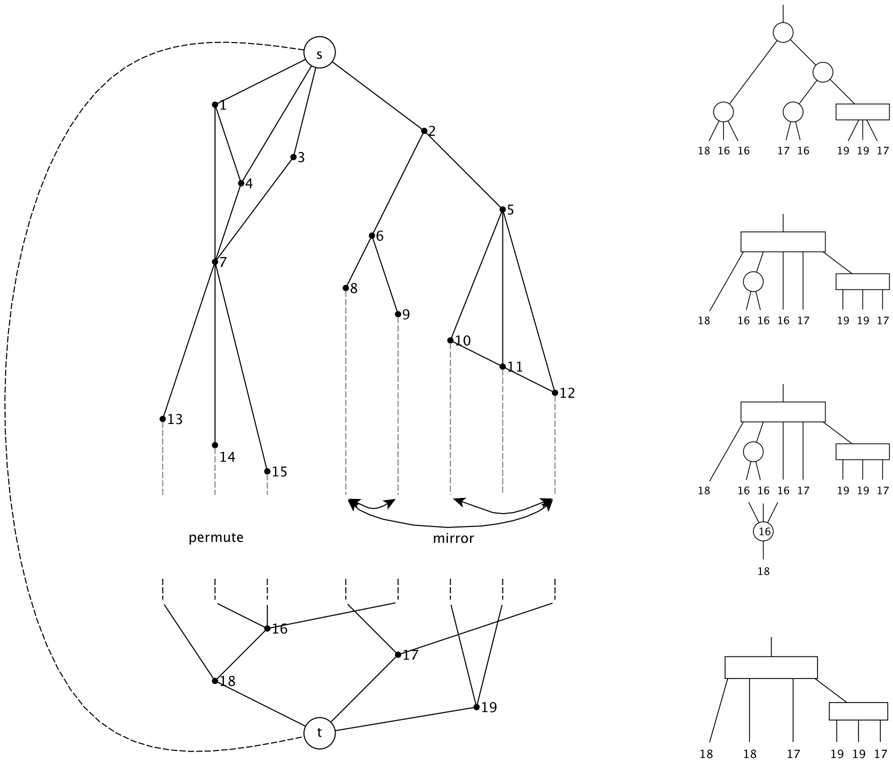


Figure 4.5: In each step, the PQ-tree holds the information which parts of the already laid out subgraph can still be mirrored or permuted, i.e. it stores all the planar embeddings of the subgraph. To obtain a planar embedding of this subgraph together with the next node (16 in this example), all edges leading to it must be consecutive (otherwise there would be an edge crossing). Therefore a REDUCE operation is performed on the PQ-tree to allow only such frontiers. If this yields the empty PQ-tree, the graph is nonplanar. Otherwise, all such edges are replaced by a P-node and all edges leading from the current node to later nodes, and the process advances to the next node.

Chapter 5

c-Planarity Tests

5.1 Overview

Many of the planarity tests presented in chapter 4 are not easily extendable to clustered graphs. The main reason is that for clustered graphs, there are two independent relations (incidence between nodes and edges; assignment to clusters) with very different properties and behaviour.

This rules out any arguments based solely on node/edge/... counts, and also makes a characterization in terms of forbidden subgraphs hard. However, the PQ-tree algorithm shown in chapter 4.4 lends itself to extension to clustered graphs, as shown in [Feng96]. The algorithm presented there works for c-connected clustered graphs, and has a linear running time. Some details are given in the following, since it is the basis for the extended c-planarity test presented in chapter 9.

In the last years, cluster planarity testing has again received some attention, and Gutwenger et al. presented an algorithm using SPQR-trees which can handle some cases of non-connected clusters in polynomial time [GJ02].

Also in 2002, Jünger, Leipert and Percan established that every planar clustered graph which is completely connected (meaning that for every cluster also its complement is connected) is c-planar [JLP02], a result which was also published in 2003 by Cornelsen and Wagner in [CW03].

In the year 2004, a polynomial time algorithm was presented by Cortese, Battista et al. to test cluster planarity on clustered graphs for which the underlying graph is a cycle (and therefore typically non-c-connected), [CBPP04].

All these results show that there has been interest and investigation in the field of cluster planarity for non-c-connected clustered graphs; a general result, whether there is a polynomial time algorithm for all clustered graphs, however, has not yet been obtained.

5.2 Feng's Algorithm

This c-planarity test was presented in [Feng96] along with the definition of clustered graphs. The main criterion upon which the test relies reads as follows:

Theorem 5.2.1. *A c-connected clustered graph $C = (G, T)$ is c-planar if and only if G is planar, and there exists a planar drawing of G, such that for each node ν of T, all the vertices and edges of $G \setminus C^\nu$ are in the external face of the drawing of C^ν.*

The test is done recursively using a depth-first search in the tree T of a clustered graph (G, T). Starting with the "smallest" clusters, a planarity test is performed for each cluster ν, with the additional restriction, that all nodes connected to $G \setminus C^\nu$ must border a common face of C^ν.

This restriction is enforced by connecting all such nodes to a new node v_o representing $G \setminus C^\nu$, via so-called "virtual nodes" v_i (representing the cut-edges of C^ν). The PQ planarity test is invoked with some v_i as s, and $t = v_o$ the last node to be considered. By stopping the PQ planarity test immediately before the reduce step for v_o (the REDUCE step for v_o will trivially always be successful), a PQ-tree is obtained which holds all possible permutations of the v_i around v_o for which a planar embedding of C^ν is possible.

(Actually, the algorithm contracts v_o and one of the v_i, and uses the other vertex connected to v_i as s, but this doesn't change the result.)

If the planarity test fails, this means that there is no planar embedding of the cluster C^ν with all nodes connected to the outside adjacent to the same face. In this case, G is not c-planar by theorem 5.2.1.

If the planarity test succeeds, the obtained PQ-tree can be used to construct a representant graph C'^ν in which all the nodes connected to $G \setminus C^\nu$ always are on the same face, and which allows the same orderings during a round-trip along the border of the face as given by the PQ-tree. These are exactly the orderings that allow a planar embedding of C^ν with $G \setminus C^\nu$ in a single face of C^ν.

The representant graph is built from "wheel graphs" and paths joining them: every Q-node is replaced by a wheel graph, with each anchor of a tree edge represented by a node on the rim of the wheel, each one joined to its two neighbours, and all joined to an additional node, called axis. Every P-node is replaced by a single node.

All embeddings of this representant graph have all nodes connected to some v_i on a single face, without constraints from cluster structure, only using adjacency, and allows exactly the same ordering of the cut edges of C^ν

as C^{ν} together with the cluster condition. So replacing C^{ν} by C'^{ν} doesn't change the (cluster) planarity of G, and eliminates one cluster constraint.

The overall algorithmic complexity of this c-planarity testing algorithm is $\mathcal{O}(n \cdot |T|)$. A corresponding embedding algorithm was also presented in the same work, with a running time of $\mathcal{O}(n^2)$. This was improved to a linear running time by Dahlhaus in 1998, [Dah98].

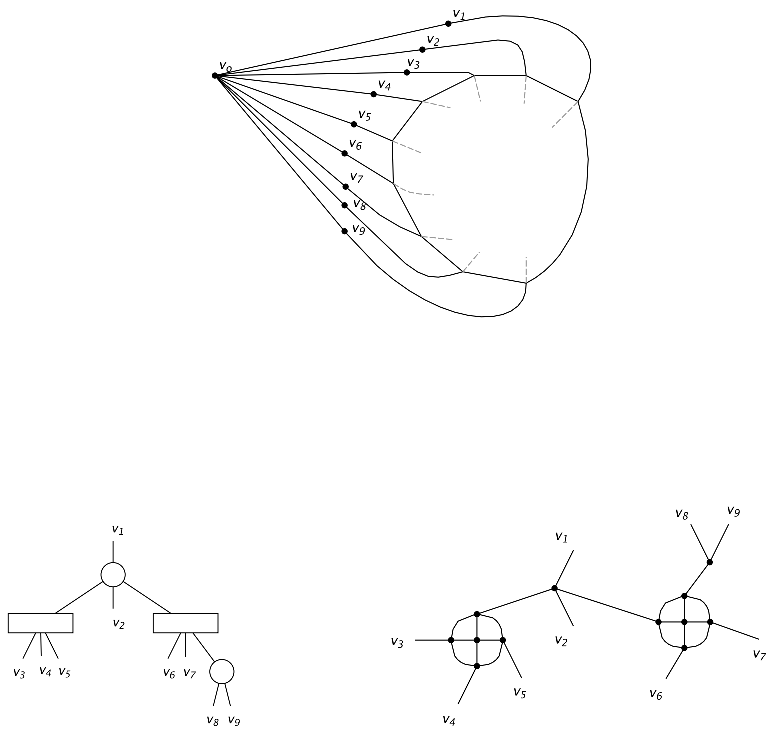


Figure 5.1: Representant node v_o for $G \setminus C^{\nu}$, and virtual nodes v_i around C^{ν}. The second figure shows some sample PQ tree reflecting the possible ordering of the virtual nodes around v_o, and the third one the corresponding representant graph made of wheel graphs.

Part III

New Results

Chapter 6

Introduction

Although there are fast algorithms for checking c-planarity for c-connected clustered graphs, and since recently, also for “almost c-connected” clustered graphs [GJ02], no such algorithm has yet been presented for arbitrary clustered graphs.

Moreover, when considering clustered graphs with non-connected clusters, the constraints demanded by c-planarity seem too restrictive for certain purposes (e.g. why not have “holes” in the drawing of a cluster for some topological map?).

This work now tries to find c-planarity testing algorithms which work for a wider range of graphs, and to investigate other forms of planarity for clustered graphs.

Chapter 7 presents such an alternative form of planarity, *c*-planarity*, and shows its relation to c-planarity. Chapter 8 tries to establish a characterization of c*-planar graphs in terms of forbidden subgraphs, and in chapter 8.2 includes some results on forbidden subgraphs for graphs in which at most given edges may intersect, which could be interesting on their own.

Chapter 9 once again deals with the “classical” c-planarity (as specified in definition 3.4.2), and presents algorithms for c-planarity testing for some special cases of non-connected clusters.

Chapter 7

An Alternative Concept of Cluster Planarity

This chapter presents an alternative form of cluster planarity, which poses less restrictive constraints than c-planarity, but has the property that the problem of testing a graph for c-planarity can be transformed into the problem of testing this alternative form of cluster planarity.

7.1 Definition of c*-Planarity

Definition 7.1.1. Let a clustered graph (G, T) be called *c*-planar* if for each cluster C, there exists a set of edges E^C such that $G \cup \bigcup_C E^C$ is planar, and $C \cup E^C$ is connected.

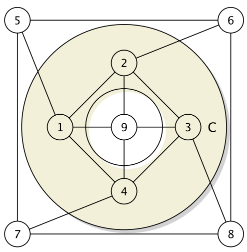


Figure 7.1: A graph which is c*-planar, but not c-planar.

See figure 7.1 for an example of a c*-planar graph. This form of planarity suffices to create comprehensible charts of computer networks or social relation maps; clearly, every planar clustered graph with connected clusters is c*-planar.

7.2 Relation to c-Planarity

The following theorems expose a relation between c-planarity and c*-planarity, which allows to use any c*-planarity test also for testing c-planarity.

(A similar result has independently been obtained by Jünger, Leipert and Percan in [JLP02], and by Cornelsen and Wagner in [CW03], from a slightly different point of view, the latter stating that a planar completely connected clustered graph is c-planar. Their papers however do not investigate whether it is possible to make a non-c-connected clustered graph completely connected while preserving planarity.)

Lemma 7.2.1. *For a graph G, if $C \subseteq G$ is connected and $G \setminus C$ is connected, then in any planar drawing of G, $G \setminus C$ is contained in a single face of C.*

Proof. If there exists a planar drawing of G, and $G \setminus C$ is contained in more than one face of C, then there must exist an edge $e \in E(G \setminus C)$ connecting these parts, and thereby crossing an edge in $E(C)$, contradicting the planarity of the drawing.

Theorem 7.2.2. *Let (G, T) be a clustered graph, let r denote the root of T, and $p(\nu)$ the parent of a node $\nu \in T$; let $L(\nu)$ denote the set of leaves which are direct children of ν, and $\bar{L}(\nu)$ the set of leaves which are descendents of ν.*

Let each edge $\langle x_1, x_2 \rangle \in E(G)$ have $p(x_1) = p(x_2) \vee p(x_1) = p(p(x_2)) \vee p(p(x_1)) = p(x_2)$ (i.e., no edge crosses more than one cluster border). Further, let $\bar{L}(\nu)$ be connected to $G \setminus \bar{L}(\nu)$ for all $\nu \in T \setminus \{r\}$, and S connected to $G \setminus S$ for all components S of $L(\nu)$, $\nu \in T$.

Let T' denote the tree obtained from T by replacing, for each ν which has non-leaf children, all leaf children by a cluster l_ν which has exactly these leaves as children. (G, T) is c-planar if and only if (G, T') is c-planar.*

Proof. "$\Leftarrow$": If (G, T') is c*-planar, there exist sets $E^{\nu'}$ of edges which make ν' connected for all $\nu' \in T'$, and a planar embedding of (G', T') with $G' = G \cup \bigcup_{\nu' \in T'} E^{\nu'}$. Specifically, l_ν and therefore $L(\nu)$ is made connected for each $\nu \in T$. Furthermore, $L(\nu)$ is adjacent to each $L(\mu)$ with $p(\mu) = \nu$, and is adjacent to $L(p(\nu))$. It needs to be shown that each $\nu \in T$ can be drawn in a simple closed region. This is definitely true for $\nu = r$, therefore consider ν with $p(\nu) = \nu_p$. $G' \setminus L(\nu_p)$ has the components ν_i with $p(\nu_i) = \nu_p$ and $G' \setminus \bar{L}(\nu_p)$ (if not empty), and each of these components is adjacent to $L(\nu_p)$. Therefore ν and $G' \setminus \nu$ are connected for each ν, and according to Lemma 7.2.1, $G' \setminus \nu$ is contained in a single face of ν in any planar drawing of G'.

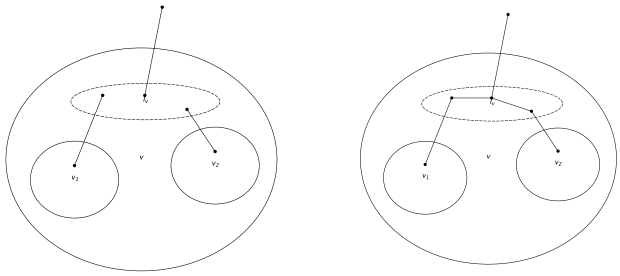


Figure 7.2: All edges cross at most one cluster border. Therefore, all connections from ν_i to $G \setminus \nu_i$ go to $L(\nu)$. Since G is c*planar, for each ν_i, $L(\nu_i)$ and therefore ν_i and $G \setminus \nu_i$ are made connected by adding E^ν. This shows that $G \setminus \nu_i$ can be drawn in one face of ν_i, proving that G is c-planar.

"⇒": Consider the plane graph made of a cluster planar drawing of G, including cluster borders drawn around each cluster, $\mathcal{E}(\mathcal{G}) \cup \bigcup_\nu \partial\nu$. For a given ν, first ignore all edges which have at least one endpoint in $L(\nu)$. This gives a single face (not necessarily simple) which contains $L(\nu)$, since no edge is allowed to cross more than one cluster border. Now add edges step by step. Every time a new edge divides a face (necessarily into two), the two new faces share a common $x \in L(\nu)$.

Therefore, after all edges are added, a set of faces $\bar{F} = (f_i)_i$ is obtained, in which any subset $S \subseteq \bar{F}$ shares a common x with $\bar{F} \setminus S$. Hence, connecting all $x \in L(\nu) \cap \partial f_i$ for all $f_i \in \bar{F}$ (e.g. to a new node x_{f_i}) makes $L(\nu)$ connected, and preserves planarity (adding a star into a face does not need any crossings), fulfilling the requirements for c*-planarity (see figure 7.3).

Corollary 7.2.3. *For a clustered graph (G, T), any c*-planarity test can be used to test c-planarity.*

Proof. The assumptions on (G, T) made in Theorem 7.2.2 can be overcome: If a clustered graph (G, T) does not have all edges cross at most one cluster border, i.e. $p(x_1) \neq p(x_2) \wedge p(x_1) \neq p(p(x_2)) \wedge p(p(x_1)) \neq p(x_2)$, inserting nodes belonging to the clusters on the path in T from $p(x_1)$ to $p(x_2)$, in this order, creates a graph (G', T') which is c-planar if and only if (G, T) is: If (G, T) is c-planar, consider any c-planar drawing of (G, T) including cluster borders, and insert nodes at the crossings of edges with the cluster borders. The graph stays connected and is a (G', T') as described above. On the other hand, if (G', T') is c-planar, consider a c-planar drawing of (G', T'), and replace the inserted nodes by a single edge again. The conditions on the new edges imposed by c-planarity are fulfilled: they have endpoints in

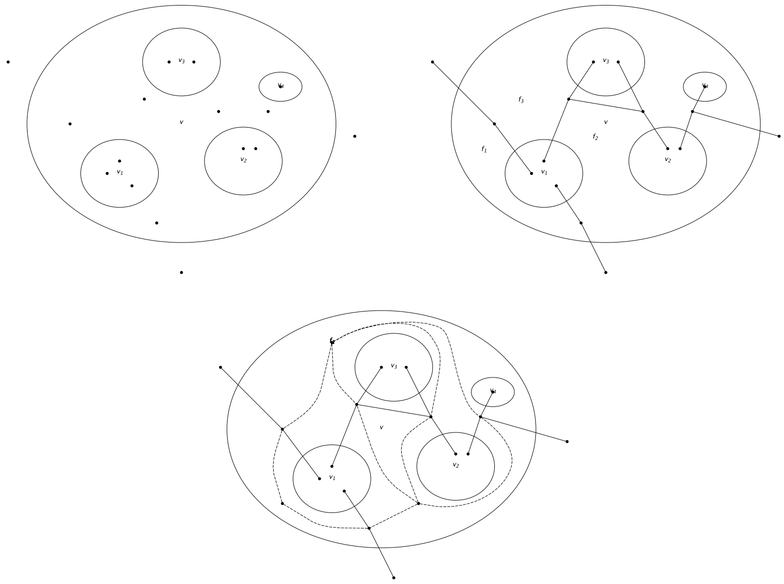


Figure 7.3: Inserting the edges incident with $L(\nu)$ one by one shows that all $x \in L(\nu)$ can be connected without disrupting planarity.

different clusters, and cross each cluster border at most once (since at most one node was inserted for each cluster, due to choosing a path in T).

If $\bar{L}(\nu)$ is not connected to $G \setminus \bar{L}(\nu)$, any c-planar embedding of $G[\bar{L}(\nu)]$ can be used together with a c-planar embedding of $G[G \setminus \bar{L}(\nu)]$ to yield a c-planar embedding of (G, T). The same is valid for a component S of $L(\nu)$ not connected to $G \setminus S$. So in both cases, the proof of Theorem 7.2.2 can be applied to the subproblems, yielding the result for G.

Chapter 8

Forbidden Subgraphs for c*-Planarity

8.1 Overview

Any test algorithm for c*-planarity needs to be able to indicate that a given graph is c*-planar or not c*-planar. The more explicit the indication, the easier the algorithm or an implementation of it can be verified.

So the wish-list for such an algorithm would first contain an embedding option for the algorithm, by which the result "is c*-planar" can be verified. The next item on the list would be a possibility to verify a negative result, such as pointing out a rather small or simple substructure of the graph which is known to be non-c*planar. In other words, a criterion in terms of forbidden subgraphs (this would also help in finding partitions of a graph, where the graph is c*-planar only if the subgraphs indicated by the partition are).

Such a criterion, however, is not yet known for neither c-planarity nor c*-planarity. This chapter takes aim at finding one; as of now, the full statement of the general characterization presented remains a conjecture, since some links in the proofs are not yet established. Parts of it (one direction of the criterion) however are proven, and even if the conjecture turns out wrong, the ideas used could eventually lead to a correct criterion.

Moreover, the results in section 8.2, preparing tools for the characterization of c*-planarity, could be interesting on their own.

The proofs of the individual theorems are rather technical and lengthy, since dealing with forbidden subgraphs and the Kuratowski graphs often requires case differentiation of many individual situations (at least, no higher-level arguments presented themselves to give relief...). An effort was taken to illustrate the proofs as far as possible.

8.2 Planarity with Allowed Crossings

This chapter investigates graphs which allow an embedding in which at most some given edge pairs may intersect. This property is also known as *weak realizability*, and its algorithmic complexity has been explored in [Kra98].

In this work, the focus is on a characterization of weak realizability in terms of forbidden subgraphs, which shall yield a tool for the characterization of c*planar graphs.

For a graph G and edges $e_1, e_2 \in E(G)$, let $G \otimes \{e_1, e_2\}$ denote the graph obtained from G by "crossing" e_1 and e_2, i.e. by adding a node v_S and replacing $e_1 = \langle x_1, x_2 \rangle$ by $e_{11} = \langle x_1, v_S \rangle$ and $e_{12} = \langle v_S, x_2 \rangle$, and $e_2 = \langle y_1, y_2 \rangle$ by $e_{21} = \langle y_1, v_S \rangle$ and $e_{22} = \langle v_S, y_2 \rangle$.

Furthermore, for a sequence of edge pairs $S = (s_1, s_2, ..., s_n)$, and $S_j = (s_1, ..., s_j)$ for $0 \leq j \leq n$, let $G \otimes S := ((G \otimes s_1) \otimes s_2) ... \otimes s_n$, with $s_j \in P_2(E(G \otimes S_{j-1}))$. By setting $\psi(e) = e$ if $e \in E(G)$, and $\psi(e_{lm}) = \psi(e_l)$ in each crossing operation, $\psi(e)$ is defined as the original edge in G of which e is a part of. Let $\psi(\{e_1, e_2\}) := \{\psi(e_1), \psi(e_2)\}$.

A sequence S is said to be consistent with a set $R \subseteq P_2(E(G))$ if $\forall s_j \in S : \psi(s_j) \in R$, and ψ is injective on S.

Definition 8.2.1. For $R \subseteq P_2(E(G))$, let a graph G be called *R-planar* if G can be drawn in the plane without any edge intersections but between edge pairs listed in R, with at most one intersection per pair.

Remark 8.2.2. *G is R-planar if and only if for some S consistent with R, $G \otimes S$ is planar.*

For $X = TY$, i.e. X a subdivision of Y, and an edge e of X, let $\phi(e) \in E(Y)$ denote the edge in Y whose subdivision in X contains e. Likewise, for a node x of X which is not a branch vertex of X, let $\phi(x)$ denote the edge in Y whose subdivision in X contains x. For branch vertices, $\phi(x)$ is the node corresponding to x in Y. A *d-edge* (subdivision edge) $\bar{e}$ is the concatenation of all $e \in X$ which have the same $\phi(e)$: $\bar{e}(e_y) = \bigcup \phi^{-1}(e_y)$; let $\bar{E}(X) := \{\bar{e} | \bar{e} = \bigcup \phi^{-1}(e_y), e_y \in E(Y)\}$. Let $\phi(\bar{e}) = \phi(e)$ for arbitrary $e \in \bar{e}$. The endpoints of a d-edge $\bar{e}$ are the nodes in $\bar{e}$ which are incident with only one $e \in \bar{e}$. Two d-edges $\bar{e}_1, \bar{e}_2$ are called adjacent if $\phi(\bar{e}_1), \phi(\bar{e}_2)$ are.

When using ϕ or $\bar{e}$, if Y is not explicitly named, it is always assumed to be the graph obtained from X by contracting every two edges connected to a common node of degree 2 to a single edge.

Let K^5_s denote the graph obtained from K^5 by splitting one of the vertices x of degree 4 into two vertices x_1, x_2 joined by a new edge $\langle x_1, x_2 \rangle$, which both have degree 3 (see figure 8.1), and let K^{5*} denote the class of graphs obtained from K^5 by splitting zero or more vertices in this way (clearly, $K^5, K^5_s \in K^{5*}$). For any $Y \in K^{5*}$, let $SP(Y)$ be defined as the set of all edges added by the splitting operations. For $X = TY$, $SP(X) := \{\bar{e} \in \bar{E}(X) | \phi(\bar{e}) \in SP(Y)\}$. For all other graphs, $SP(X) = \emptyset$.

Two edges $e_1, e_2 \in Y$ are called *s-adjacent* if $e_1 = e_2$, are adjacent to one another, or are both adjacent to a common $s \in SP(Y)$. Accordingly, $\bar{e}_1, \bar{e}_2 \in \bar{E}(X)$ are s-adjacent if $\phi(\bar{e}_1), \phi(\bar{e}_2)$ are.

Every $Y \in K^{5*}$ is a $TK_{3,3}$ or TK^5, since it is a MK^5, and therefore nonplanar. X shall be called a TK^{5*} if it is a TY for any $Y \in K^{5*}$.

Theorem 8.2.3. *G is $\{e_1, e_2\}$-planar if and only if there exists no $X \subseteq G$ such that $X = TK^5$, $TK_{3,3}$ or TK^5_s with $e_1 \notin X \vee e_2 \notin X \vee \bar{e}(e_1)$ s-adjacent to $\bar{e}(e_2)$.*

Remark 8.2.4. *A $TK_{3,3}$ or TK^5_s in which $\bar{e}(e_1)$ and $\bar{e}(e_2)$ are not s-adjacent, does not necessarily prove that G is non-$\{e_1, e_2\}$-planar, see figure 8.2.*

Proof of Theorem 8.2.3. "⇒": Let $G' := G \otimes \{e_1, e_2\}$ with node v_s and edges e_{lm} added by the crossing operation. If an X with the given properties exists, it clearly makes G nonplanar. Moreover, it also gives rise to an X' which makes G' nonplanar: If $e_1 \notin X \wedge e_2 \notin X$, then $X \subseteq G'$ and X can be used as X'. If $e_1 \in X \wedge e_2 \notin X$, then X' is obtained by replacing e_1 by $(v_S, \{e_{11}, e_{12}\})$; similar for $e_1 \notin X \wedge e_2 \in X$. If both are in X, then in case they are on the same d-edge $\bar{e}$, X' can be constructed by joining the two nodes nearest to the endpoints of $\bar{e}$ via v_S (see figure 8.3). If they are on different adjacent d-edges $\bar{e}_1$, $\bar{e}_2$ (joined in v_B), then in the case of $X = TK_{3,3}$, $X' = TK_{3,3}$ is obtained by replacing v_B by v_S as a branch vertex (see figure 8.4); in the case of $X = TK^5$, $X' = TK^{5*}$ is obtained according to figure 8.5. When X is a TK^5_s, the same arguments can be used in case $\bar{e}_1$ and $\bar{e}_2$ are adjacent (yielding a TK^{5*}); if they are separated by $\bar{s} \in SP(X)$, then a TK^5 is obtained by the operation depicted in figure 8.6.

In all cases $X' \subseteq G'$ is not planar, and according to Remark 8.2.2, G is not $\{e_1, e_2\}$-planar.

"⇐": Now it needs to be shown that if G is not $\{e_1, e_2\}$-planar, such an X exists. According to Remark 8.2.2, there must exist $X \subseteq G, X = TK_{3,3}$ or TK^5 and $X' \subseteq G', X' = TK_{3,3}$ or TK_5. If for X, $e_1 \notin X \vee e_2 \notin X \vee \bar{e}(e_1)$ adjacent to $\bar{e}(e_2)$, the result is shown. Likewise, for $v_S \notin X'$, $X' \subseteq G$ and has the properties to be shown. Therefore, in the following assume that $e_1 \in X \wedge e_2 \in X \wedge \bar{e}(e_1)$ not adjacent to $\bar{e}(e_2)$, and that $v_S \in X'$. The result will be shown by starting with $Y \subseteq G', Y = X \otimes \{e_1, e_2\}$ and then showing that any existing X' leads to a $Z \subseteq G, Z = TK_{3,3}$ or TK^{5*} which has the properties demanded by the theorem.

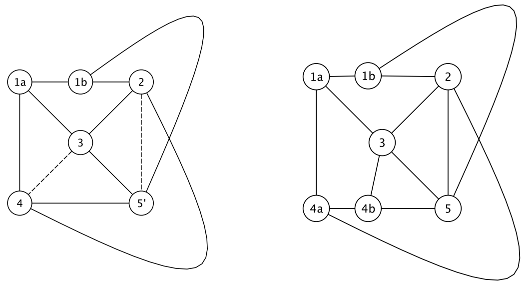


Figure 8.1: Left figure: K_s^5 (contains a $TK_{3,3}$ consisting of only the solid lines), $SP(X) = \{\langle 1a, 1b\rangle\}$; right figure: a K^{5*}, $SP(X) = \{\langle 1a, 1b\rangle, \langle 4a, 4b\rangle\}$.

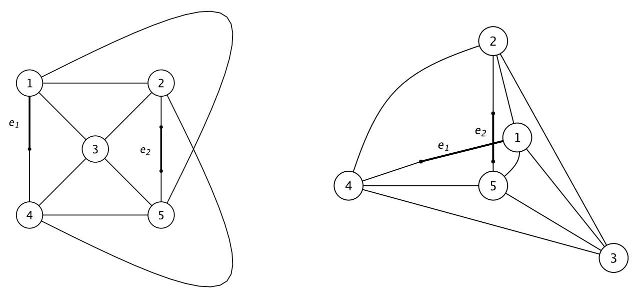


Figure 8.2: For $X = TK^5$ with nonadjacent edges $\bar{e}(e_1)$, $\bar{e}(e_2)$, a $\{e_1, e_2\}$-planar embedding can be constructed as shown on the right (obtained from the left figure by moving nodes 1 and 3).

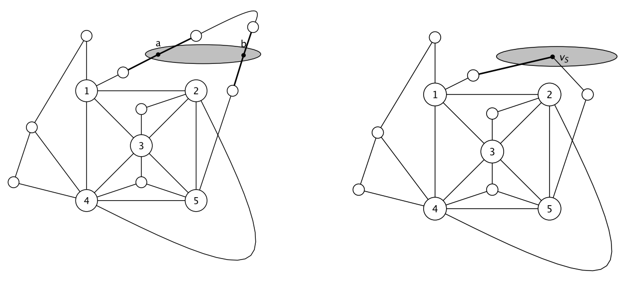


Figure 8.3: e_1 and e_2 are on the same edge, X' is obtained by replacing the path from a to b in G by the one through v_S in G'.

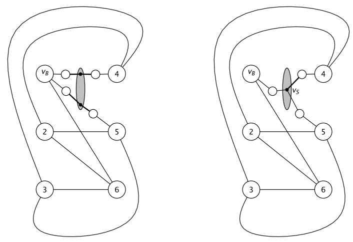


Figure 8.4: $X = TK_{3,3}$, e_1 and e_2 are on adjacent edges. X' is constructed by replacing v_B by v_S as a branch node.

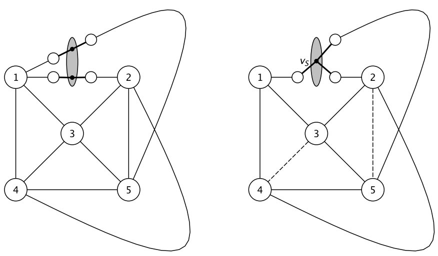


Figure 8.5: $X = TK^5$, e_1 and e_2 are on adjacent edges. The graph created by the construction is a TK^{5*}, and contains a $TK_{3,3}$ (only the solid lines in the graph on the right).

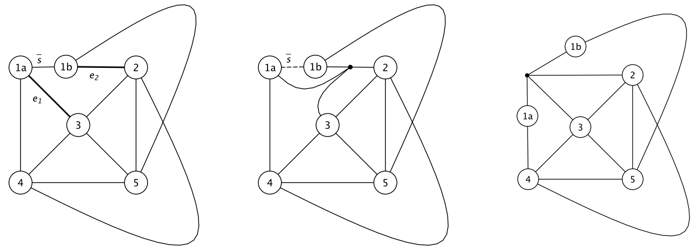


Figure 8.6: $X = TK^5_s$, $\bar{e}(e_1)$ and $\bar{e}(e_2)$ are separated by $\bar{s} \in SP(X)$. Performing the crossing operation and deleting $\bar{s}$ results in a K^5.

1. First, consider $X = TK^5$ (figure 8.7). Since $\phi(Y)$ is 3-connected, this is the only planar embedding (but for choosing the outer face) of Y. If G contains an X-path W from a to b such that a and b are not on the border of a common face f of Y, then $(G \setminus \{e_i\}) \cup W$ already contains a $Z = TK_{3,3}$ or TK^5 for $i = 1$ or $i = 2$ which therefore meets the conditions:

   (a) a, b both are branch vertices of X. This implies that $(a, b) = (1, 5)$ or $(a, b) = (2, 4)$. In each of these cases, Z can be obtained from X by replacing either $\bar{e}(e_1)$ or $\bar{e}(e_2)$ by W (see figure 8.8).

   (b) a is a branch vertex of X, b from the interior of a d-edge $\bar{e}$. This implies that b lies on one of the four d-edges of Y adjacent to the one vertex x of Y with $\phi(x)$ not adjacent to $\phi(a)$. In the case that $a = 3$, construct Z as in figure 8.8, otherwise as in figure 8.9.

   (c) a and b are from the interior of d-edges $\bar{e}_a$ and $\bar{e}_b$. If both a and b lie on some $\bar{e}(e_{ij})$, then i must be the same for both, and j different. Therefore Z can be constructed by replacing the (a, b)-path through e_i by W, see figure 8.10. Otherwise, let a lie on some edge $\bar{e}$ but $\bar{e}(e_{ij})$. Set a' to an endpoint of $\bar{e}$ which is not on the border of a face f of Y adjacent to b (this is always possible, since if both endpoints have a face in common with b, b and a would share a common face, or the dual graph of Y would have a cycle of length 3). By contracting a and a', the previous case is created for a minor G_1 of G resp. G'_1 of G'. Since this yields a Z which is contained in $G_1 \setminus \{e_i\}$, this Z also delivers the wanted result for G (figure 8.10).

   If G does not contain such an X-path, then each component of $G' \setminus Y$ falls in one of the following categories (let $att^Y(S)$ denote the points of attachment of S in Y):

   - $\bar{A} = \{A \mid |att^Y(A)| > 1, \exists \bar{e} : att^Y(A) \subseteq \bar{e}\}$
   - $\bar{B} = \{B \mid |att^Y(B)| > 1, B \notin \bar{A}\}$
   - $\bar{C} = \{C \mid |att^Y(C)| = 1\}$
   - $\bar{D} = \{D \mid att^Y(D) = \emptyset\}$

   Since a graph is planar if and only if all its blocks are planar, and since we assumed that there is no X' with $v_S \notin X'$, $\bar{C}$ and $\bar{D}$ can be omitted for further considerations, as well as all blocks of $G'[Y \cup \bar{B}]$ and $G'[Y \cup \bar{A}]$ which do not contain elements of Y.

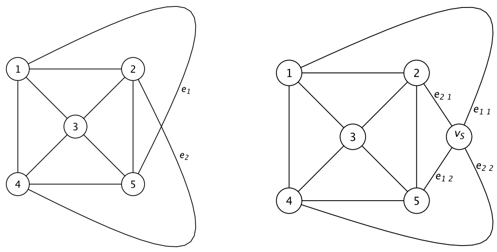


Figure 8.7: X and Y for the case $X = TK^5$.

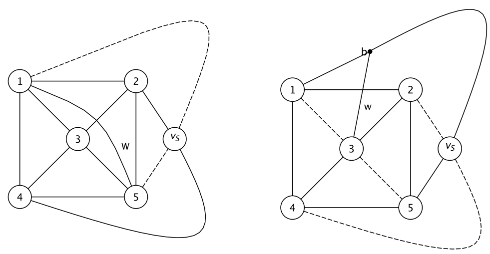


Figure 8.8: On the left: Case 1a; on the right: Case 1b, $a = 3$. Y contains a $TK_{3,3}$ with the needed properties (consisting of only the solid lines).

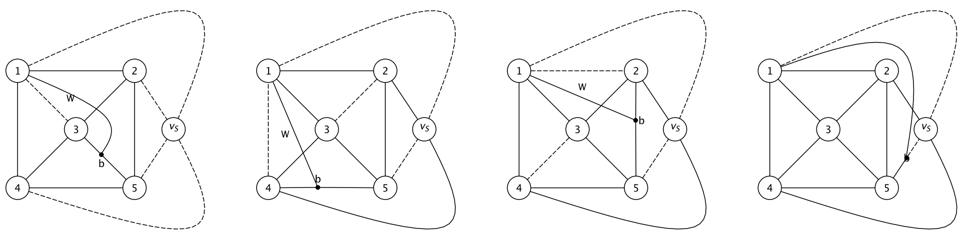


Figure 8.9: Case 1b, $a \neq 3$. In the first three cases, X contains a $TK_{3,3}$, in the last a TK^5 with the needed properties.

Since two d-edges of Y border at most one common face, each B can only be embedded in a single face of X, denoted $f(B)$. Therefore $G'[Y \cup B]$ is planar if and only if $G'[\partial f(B) \cup B]$ is planar. If $\partial f(B)$ does not contain any e_{ij}, any $Z = TK_{3,3}$ or TK^5, $Z \subseteq G'[\partial f(B) \cup B]$ fulfills the requirements. If not, then it contains exactly two e_{ij}, and $G'[\partial f(B) \cup B]$ is a minor of the graph obtained from $G'[\partial f(B) \cup B]$ by replacing each e_{ij} by a path in G to a common branch vertex of X not contained in ∂f (see figure 8.11). In this graph, any subdivision Z of K^5 or $K_{3,3}$ has e_1 and e_2 on the same d-edge, therefore Z fulfills the requirements.

So let assume that $G'[Y \cup B]$ is planar for all B. With $\bar{B}(f) := \{B|f(B) = f\}$, $G'[Y \cup \bar{B}]$ is planar if and only if for all f, $G'[Y \cup \bar{B}(f)]$ is planar (since $\phi(Y)$ is 3-connected). So if $G'[Y \cup \bar{B}(f)]$ is nonplanar, this is due to $\bar{B}(f)$ not "fitting" into a single face of ∂f. Then, however, $U := G[\partial f \cup \bar{B}(f)] \cup (v_f, \{\langle v_f, x\rangle | x \text{ branch vertex of } \partial f\})$ also is not planar ($\phi(U \setminus \bar{B}(f))$ is 3-connected, and therefore forces $\bar{B}(f)$ into a single face). If $v_S \notin \partial f$, v_f can be chosen arbitrarily from the remaining two branch vertices of X, and a subdivision of U is contained in $G \setminus \{e_1, e_2\}$ (see figure 8.12). If $v_S \in \partial f$, choose the branch vertex of X not adjacent to v_S as v_f. Further create U' by replacing the two e_{ij} by a path through the corresponding e_i and the remaining two branch vertices of X. Since the induced plane graph does not change ($U' \setminus B(f)$ to $U \setminus B(f)$), this does not change planarity. So U' is nonplanar, and $U' \subseteq G$. Moreover, since any d-edge that contains either e_1 or e_2 must contain a fixed branch vertex (node 1 in figure 8.13), they must be on adjacent d-edges of any $Z = TK^5$ or $TK_{3,3} \subseteq U'$. So in both cases, the requirements are met (figure 8.13).

Finally let assume that $G'[Y \cup \bar{B}]$ is planar. In order for G' to be nonplanar, it must be impossible to fit the $\bar{A}(\bar{e})$ into the faces bounded by Y and the $\bar{B}(f_1)$,$\bar{B}(f_2)$ on the both sides of an edge $\bar{e}$ of Y adjacent to the faces f_1, f_2 . This is equivalent to $U := G'[\bar{A}(\bar{e}) \cup \bar{B}(\bar{e}) \cup \partial f_1 \cup \partial f_2 \cup \langle v_{\bar{e}1}, v_{\bar{e}2}\rangle]$ being nonplanar, with $\bar{B}(\bar{e}) := \{B|att^Y(B) \cap \bar{e} \neq \emptyset, B \in \bar{B}(f_1) \cup \bar{B}(f_2)\}$, and $v_{\bar{e}k}$ being the two branch vertices of ∂f_1 and ∂f_2 not adjacent to $\bar{e}$, since $\phi(U \cap Y)$ is 3-connected. See figure 8.14. (If $\bar{B}(\bar{e}) = \emptyset$, it suffices to see that $\bar{e}$ always joins exactly 2 different faces. If $\bar{B}(\bar{e}) \neq \emptyset$, then for a single path W in a B joining $\bar{e}$ and $Y \setminus \bar{e}$, $\phi(G'[\partial f_1 \cup \partial f_2 \cup W])$ is 3-connected, and therefore only allows a single planar embedding.)

In case that none of f_1, f_2 borders v_S the $v_{\bar{e}k}$ can be connected via the fifth branch vertex. In the case that v_S borders one of the faces (i.e.

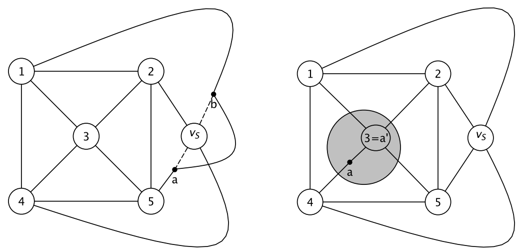


Figure 8.10: Case 1c. Construction for a and b both on $\bar{e}(e_{ij})$ on the left, for at least one not on $\bar{e}(e_{ij})$ on the right.

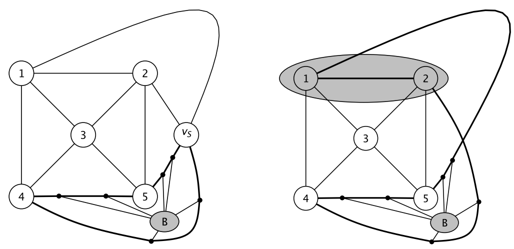


Figure 8.11: $G'[Y \cup B]$ is nonplanar for some B, and $e_{ij} \subseteq \partial f(B)$. Replacing the e_{ij} by paths to 1 (e_2 via 2) yields an $X \subseteq G$ with e_1 on the same d-edge as e_2.

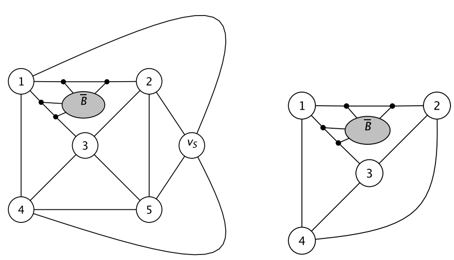


Figure 8.12: If $v_S \notin \partial f$, U is a minor of $G \setminus \{e_1, e_2\}$.

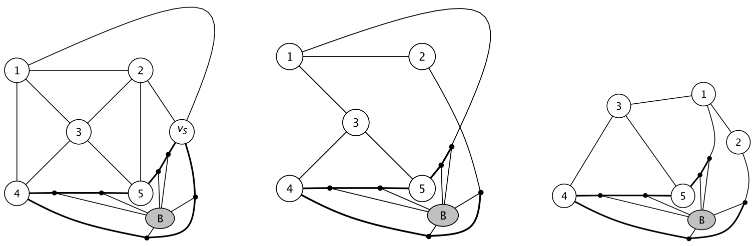


Figure 8.13: U' can be obtained from Y by removing some edges when $v_S \in \partial f$.

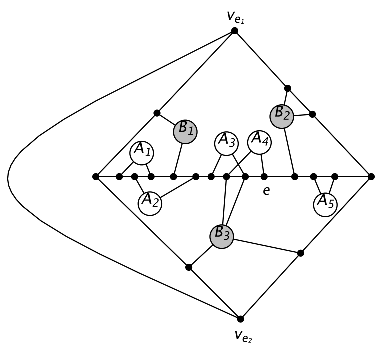


Figure 8.14: planarity checking for $\bar{A}(\bar{e}) \cup \bar{B}(\bar{e})$.

$v_S = v_{\bar{e}k}$), form U' by replacing the v_S and the three e_{ij} by one of the remaining two branch vertices of X and the corresponding paths in X, see figure 8.15. Whenever both e_i are contained in a $Z \subseteq U, Z = TK^5$ or $TK_{3,3}$, they are always on the same or adjacent d-edges.

In the case that v_S is one of the endpoints of $\bar{e}$, and $e_{i_0 j_0} \in \bar{e}$, construct U' by replacing $e_{i_0 j_0}$ and v_S and the two other e_{ij} by e_{i_0} and paths in X to the fourth branch vertex x of X with $\phi(x)$ adjacent to $\phi(v_S)$. In this case, U contains at most one e_i, and is nonplanar by construction (figure 8.16).

2. Now, consider $X = TK_{3,3}$ (figure 8.17). If G contains an X-path W from a to b such that a and b are not on the border of a common face f of Y, then $(G \setminus \{e_i\}) \cup W$ already contains a $Z = TK_{3,3}$ or TK^{5*} for $i = 1$ or $i = 2$ which meets the conditions:

   (a) a, b both are branch vertices of Y. This implies that $(a, b) = (5, 3)$ or $(a, b) = (2, 6)$. So W can be used as a replacement for e_1 or e_2 to yield a Z as needed.

   (b) a is a branch vertex of X, b is from the interior of a d-edge $\bar{e}$. Z can be constructed as shown in figure 8.18.

   (c) a and b lie on the interior of d-edges $\bar{e}_a$ resp. $\bar{e}_b$. At least one of the endpoints of $\bar{e}_a$, $\bar{e}_b$ which is not adjacent to a face adjacent to the other d-edge is different from v_S. Let this be one of $\bar{e}_a$, and denote it by a'. Contracting Y by (a, a') yields the previous case for a minor Y' of G'. Any resulting Z also shows the result for G (because it is contained in a minor of either $G \setminus \{e_i\}$ or of a graph in which the e_i must always be on adjacent d-edges for any Z).

If G does not contain such an X-path, the same process is performed as for $X = TK^5$ up to the case of $G'[Y \cup B]$ being nonplanar for some B. So let assume that $G'[Y \cup B]$ is planar for all B, and that $G'[Y \cup \bar{B}(f)]$ is nonplanar for some f.

For this to happen, there must be some components B_1, B_2 crossing one another, i.e. when assigning numbers to the attachment points of $\bar{B}(f)$ in the order they occur during a round trip along ∂f, there must be some $n_{11} < n_{21} < n_{12} < n_{22}$ with $x_{n_{1j}}$ attached to B_1 and $x_{n_{2j}}$ attached to B_2. Moreover, for at least one i, the n_{ij} can be chosen so that f is the only face shared by $x_{n_{i1}}$ and $x_{n_{i2}}$. The graph constructed by adding edges $e_{B_1} = \langle x_{n_{11}}, x_{n_{12}} \rangle$ and $e_{B_2} = \langle x_{n_{21}}, x_{n_{22}} \rangle$ is a minor of G' (resp. of G, when adding to X).

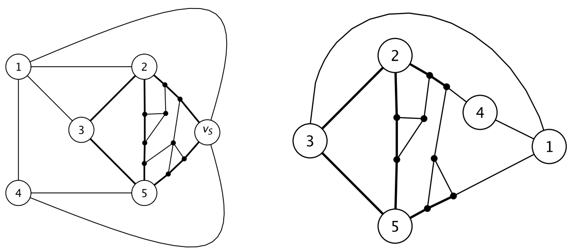


Figure 8.15: $\bar{A}(\bar{e})$ in case $v_S = v_{e_i}$.

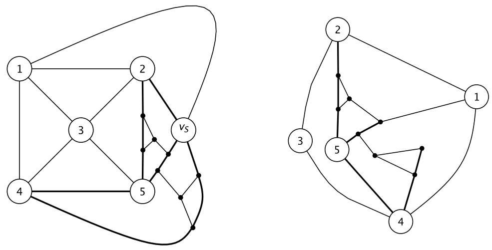


Figure 8.16: $\bar{A}(\bar{e})$ in case $v_S \in \bar{e}$.

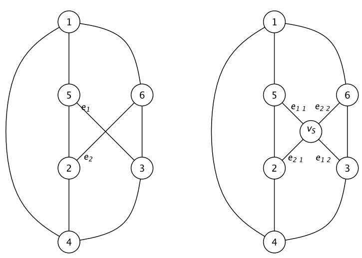


Figure 8.17: X and Y for $K_{3,3}$.

For each of the faces of Y, this yields a Z as required: For the 4-node faces adjacent to v_S, a Z can be constructed according to figure 8.19, for the 3-node faces (adjacent to v_S) according to figure 8.20, and for the 4-node faces not adjacent to v_S, according to figures 8.21, 8.22, and 8.23. In all cases either at least one of the $e_i \notin Z$ or e_1 and e_2 are on s-adjacent d-edges of Z.

It remains to consider the case $G'[Y \cup \bar{B}]$ planar. Use the same construction as for TK^5. For each $\bar{e}$ it suffices to find a subgraph U of G in which every B incident with $\bar{e}$ can only be placed in a single face of U, and $\bar{e}^o$ borders exactly two faces of U. Therefore whenever $G'[Y \cup \bar{B}(\bar{e}) \cup \bar{A}(\bar{e})]$ is nonplanar, so is $U' = U \cup \bar{B}(\bar{e}) \cup \bar{A}(\bar{e})$. See figure 8.24 for the construction of U when $\bar{e}$ joins a face f with $v_S \notin \partial f$ with another, and figure 8.25 for the construction of U when $\bar{e}$ joins two faces f_1, f_2 with $v_S \in \partial f_1 \wedge v_S \in \partial f_2$.

Summing up, all possibilities to render G' nonplanar result in a Z with the properties required by the theorem. □

Corollary 8.2.5. *G is $\{e_1, e_2\}$-planar if and only if there exists no $X \subseteq G$ such that $X = TK_{3,3}$ or TK^{5*} with $e_1 \notin X \vee e_2 \notin X \vee \bar{e}(e_1)$ s-adjacent to $\bar{e}(e_2)$.*

Proof. The existence of an appropriate X for a non-$\{e_1, e_2\}$-planar G follows from the above theorem, since a TK^5 and a TK^5_s both are TK^{5*}. For the other direction, it suffices to observe that for $X = TK^{5*}$ with $\bar{e}(e_1)$ s-adjacent to $\bar{e}(e_2)$, $X \otimes \{e_1, e_2\}$ is also a TK^{5*}: for $\bar{e}(e_1) = \bar{e}(e_2)$ and $\bar{e}(e_1)$ adjacent to $\bar{e}(e_2)$, the proof is the same as for TK^5 resp. $TK_{3,3}$, and for $\bar{e}(e_1)$ and $\bar{e}(e_2)$ separated by $\bar{s} \in SP(X)$ see figure 8.26.

Conjecture 8.2.6. *G is R-planar if and only if there exists no $X \subseteq G$ such that $X = TK_{3,3}$ or $X = TK^{5*}$, and $e_1 \notin X \vee e_2 \notin X \vee \bar{e}(e_1)$ s-adjacent to $\bar{e}(e_2)$ for all $\{e_1, e_2\} \in R$.*

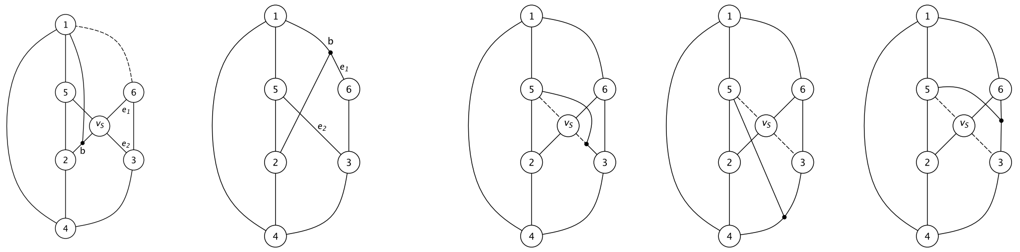


Figure 8.18: If $a = 1$ and $b \in \langle 2, v_S \rangle$, a $TK_{3,3}$ where e_1 and e_2 are on adjacent d-edges can be constructed by removing the edge $\langle 1, 6 \rangle$ and taking b as branch point instead of 6. $b \in \langle 3, v_S \rangle$ and cases with $a = 4$ can be handled analogously for symmetry reasons. The other figures show $a = 5, b \in \langle 3, v_S \rangle$ resp. $a = 5, b \in \langle 3, 4 \rangle$ resp. $a = 5, b \in \langle 6, 3 \rangle$. Here e_1 can be removed while still preserving a $K_{3,3}$, yielding an appropriate Z. The remaining cases are obtained by symmetry.

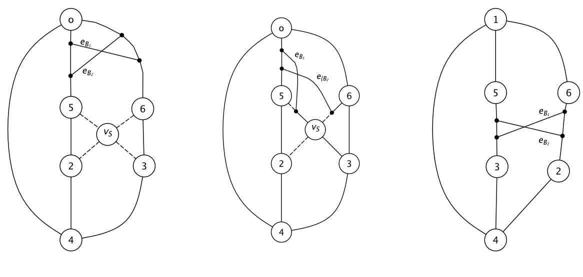


Figure 8.19: 4-node face with $v_S \in \partial f$: Let the branch vertex on ∂f not adjacent to v_S be called o. If o is one of the $x_{n_{ij}}$ or an $x_{n_{ij}}$ lies on a d-edge adjacent to o, then set $Z := X \cup \{e_{B_1}, e_{B_2}\} \setminus \{e_l | x_{n_{i2-j}} \notin \bar{e}(e_l)\}$ (if multiple $x_{n_{ij}}$ are on d-edges adjacent to o, choose one "nearest" to o). In case both e_l were removed, Z is a $TK_{3,3}$ with $e_1, e_2 \notin Z$. Otherwise, $\phi(Z \setminus \{e_{B_i}\})$ is 3-connected, $e_{B_{2-i}}$ connects two nodes which are not adjacent to a common face, and therefore Z is nonplanar with $e_l \in Z$ for only one l. If no $x_{n_{ij}}$ is adjacent to o, $Z = TK_{3,3}$ is obtained according to the figure on the right, and has e_1 and e_2 on adjacent d-edges.

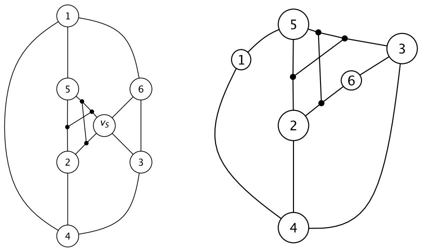


Figure 8.20: 3-node face with $v_S \in \partial f$: By removing the edge $\langle 1, 6\rangle$ from X, a graph U with 3-connected $\phi(U)$ is obtained; in U, every pair of edges has a single face which they both border. Therefore, $U' = U \cup \{e_{B_j}\}$ is nonplanar. Moreover, any Z contained in U' has $\bar{e}(e_1)$ and $\bar{e}(e_2)$ adjacent.

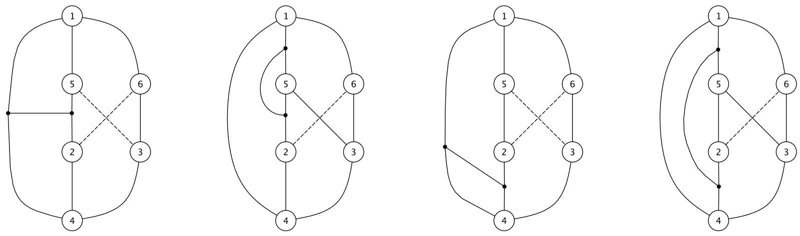


Figure 8.21: 4-node face with $v_S \notin \partial f$, and $\exists e_{B_i}$ with $x_{n_{ij}}$ on the interiors of different d-edges. By adding e_{B_i} and removing at least one e_{l_0}, a graph $Z = X \cup \{e_{B_1}, e_{B_2}\} \setminus \{e_{l_0}\}$ with 3-connected $\phi(Z)$ is obtained, for which the $x_{n_{2-ij}}$ do not share a common face. Therefore Z is nonplanar and contains only one e_l.

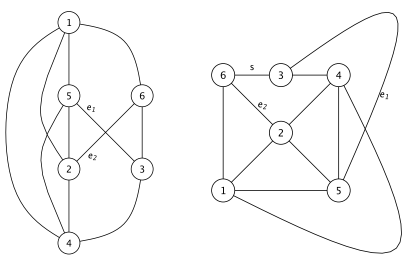


Figure 8.22: 4-node face with $v_S \notin \partial f$, and $\nexists e_{B_i}$ with an $x_{n_{ij}}$ on the interior of a d-edge. $Z = X \cup \{e_{B_1}, e_{B_2}\}$ is a K_s^5 in which $\bar{e}_1$ and $\bar{e}_2$ are separated by $\bar{s} \in SP(X)$, i.e. s-adjacent.

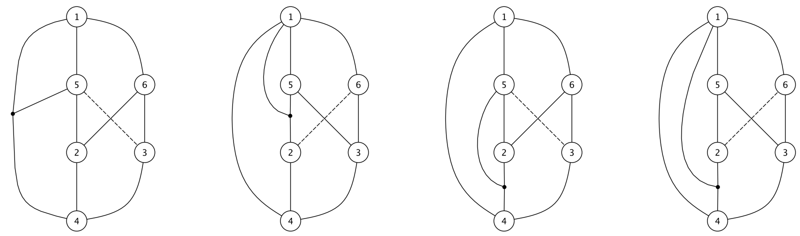


Figure 8.23: 4-node face with $v_S \notin \partial f$, and $\exists e_{B_i}$ with one $x_{n_{ij}}$ on the interior of a d-edge, and the other a branch point of X. By adding e_{B_i} and removing at least one e_l, a graph $Z = X \cup \{e_{B_1}, e_{B_2}\} \setminus \{e_k\}$ with 3-connected $\phi(Z)$ is obtained, for which the $x_{n_{2-ij}}$ do not share a common face. Therefore Z is nonplanar for appropriate k and contains only one e_l.

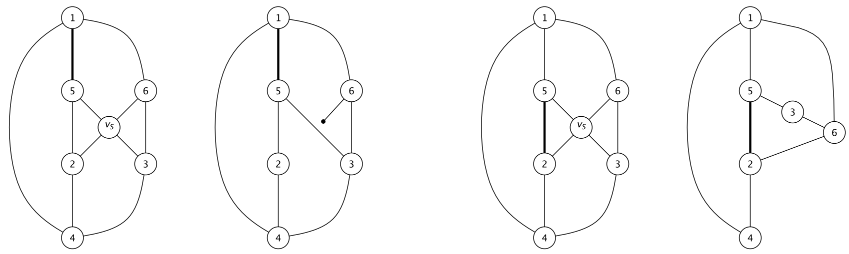


Figure 8.24: Construction of U for the cases involving an f (wlog f_1) which has not got v_S on its border. If also $v_S \notin \partial f_2$, set $U = \partial f_1 \cup \partial f_2 \cup \bar{e}(e_i)$ with arbitrary i. The other cases are shown in the figure: on the left side f_2 a 4-node face adjacent to f_1 and v_S: (1,5,3,4) is 3-connected, therefore there is only one possible embedding (but for choosing the outer face). All B connected to $\bar{e}$ can only be placed in a single face, and $\bar{e}^o$ borders exactly two faces. U contains only one of the e_i. On the right hand side, the case of f_2 being the 3-node face adjacent to f_1: U fulfills the requirements, and e_1 and e_2 always are on adjacent d-edges for any Z found in U'.

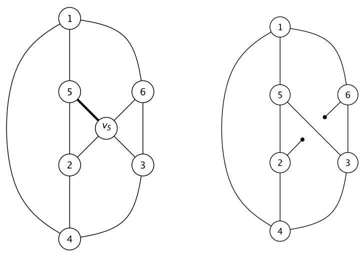


Figure 8.25: Construction of U for the case of $\bar{e}$ joining two f adjacent to v_S. Let assume $\bar{e} = (5, v_S)$. $\phi(U)$ is 3-connected, and therefore U allows only one embedding, each B attached to $\bar{e}$ can only be placed in one face, and the faces on both sides of $\bar{e}$ are different. Moreover, each U only contains one e_i, thereby fulfilling the requirements.

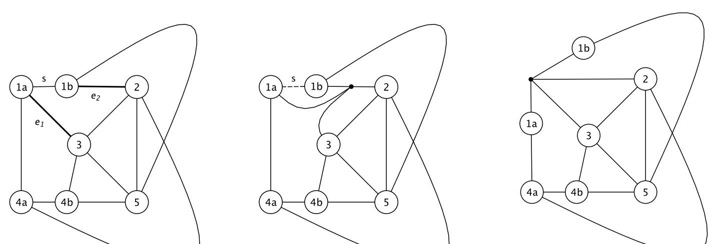


Figure 8.26: When crossing e_1, e_2 in $X = TK^{5*}$ for which $\bar{e}(e_1), \bar{e}(e_2)$ are separated by $\bar{s} \in SP(X)$, an X' can be obtained which has a node of degree 4 instead of the two endpoints of $\bar{s}$ by removing $\bar{s}$.

8.3 Graphs with a Single Cluster

The results from chapter 8.2 can now be used to give a characterization of c*-planar graphs in terms of forbidden subgraphs.

In the following, let G be a clustered graph containing a single cluster C, let K_i denote the components of C, and $\bar{K}$ the set of all K_i. Wlog let each node in $G \setminus C$ adjacent to C have degree 2, and call these nodes *virtual nodes*. The set of virtual nodes adjacent to a set S is denoted by $H(S)$. Set $E_C := E(C \cup \{v | v \in H(C)\})$.

Theorem 8.3.1. *G is c*-planar if and only if*

$$\hat{G} := G \cup (\{v_C\}, \bigcup_{K \in \bar{K}} \{\langle v_C, v_r^K \rangle | v_r^K \in K \textit{ arbitrarily chosen}\})$$

is R-planar with $R = \bigcup_{K \in \bar{K}, e \in E_C} \{\{\langle v_C, v_r^K \rangle, e\}\}$.

Proof. "⇒": Create a straight-line planar drawing of $G \cup E^C$ (a straight-line planar drawing is possible for any planar graph). Choose an arbitrary node $v_1 \in C$, and add a node v_C into a face adjacent to v_1, and the edge $\langle v_C, v_1 \rangle$. Choose a v_r^K for each K, and for each of these determine the shortest path $p(K)$ from v_C to v_r^K in C. This shall be done in a consistent way, i.e. whenever a node v occurs in a path: $p(K_i) = (v_C = v_0, v_1, ..., v_j, v, ..., v_n = v_r^{K_i})$, its predecessors shall be the same (i.e., $(v_C, v_1, ..., v_j)$ shall precede it in any path). Obviously, any node occurs at most once in a single path. Sorted by decreasing path length $|p(K)|$, add a line l^K an ϵ to the left or right of $p(K)$ for each K, so that it crosses no other l^{K_i}. This is always possible since $\bigcup_{K \in \bar{K}} p(K)$ is a tree, and none of the intermediary nodes $v_1, ..., v_{n-1}$ is an endpoint of any previous l^{K_i}. The ordering of the l^{K_i} around v_C is determined by the ordering of $v_j^{K_{i_1}}$ and $v_j^{K_{i_2}}$ around $v_{j-1}^{K_{i_1}}$ for every $K_{i_1}, K_{i_2} \in \bar{K}$, where j is the first index for which the $v_j^{K_i}$ are different. By removing E^C and inserting nodes at the crossings of l^K and $e \in E_C$, a $G \otimes S$ as required is obtained.

"⇐": If $\hat{G}$ is R-planar for the R given in the theorem, then there exists a $G^S = (\hat{G} \otimes \{e_1, \langle v_C, v_r^K \rangle\}) \otimes \dots$ with a set of intersection nodes x_j added by the crossing operations such that G^S is planar. Contracting v_C and an intersection node x_0 adjacent to it, and then contracting every intersection node with a node $v(x)$ of G next to it yields a graph G'^S. G'^S is planar since it is a minor of G^S, $G \subseteq G'^S$, and it also makes C connected (in G^S, every component of C was connected to v_C, hence every component of C is connected to $v(x_0)$ via zero or more other nodes of C). Therefore,

$G'^S \setminus (G \setminus C) \setminus E(C)$ can be used for E^C in the sense of Definition 7.1.1. See figure 8.27. □

Definition 8.3.2. Let $G_K(C) := (G \setminus C) \cup (\{v_C\} \cup \{v_K | K \in \bar{K}\}, \{\langle v_K, v \rangle | v \in H(K), K \in \bar{K}\} \cup \{\langle v_C, v_K \rangle | K \in \bar{K}\})$. A clustered graph G with a single cluster C is called *pseudo-c*-planar* if

a) G is planar,

b) $G^C := (G \setminus C) \cup (\{v_O\}, \{\langle v_O, v \rangle | v \in H(C)\})$ is planar, and

c) $G^K := (G \setminus C) \cup G_K(C)$ contains no $X = TK_{3,3}$ or $X = TK^{5*}$ with v_C not a branch vertex of X, and $v_C \notin X$ or $\bar{e}(e)$ s-adjacent to $\bar{e}(v_C)$ for all e incident with a v_K.

See figure 8.28 for an example of G, G^C, G^K.

Remark 8.3.3. *The three conditions in Definition 8.3.2 are independent.*

Proof. For each one of the conditions a),b),c), there exists a graph which violates only this condition, but not the two others. See figure 8.29.

Theorem 8.3.4. *If a graph G with a single cluster C is c*-planar, it is pseudo-c*-planar.*

Proof. Let assume G is not pseudo-c*-planar. If a) is violated (and therefore G not planar), then G is clearly not c*-planar. If b) is violated, it suffices to see that G^C is a minor of $G \cup E^C$ for any E^C (since $C \cup E^C$ is connected, it can be contracted to a single node which delivers a graph isomorphic to G^C), and therefore $G \cup E^C$ is nonplanar for any E^C.

For a violation of c) (G^K contains an X with the listed properties), consider for each given E^C the graph $X' \subseteq G \cup E^C$ created from X by finding nodes in K for each v_K (for $X = TK_{3,3}$, one node can always be found in K; for $X = TK^5$, it may be needed to choose two nodes in K as the two parts of a "split" node of a TK^{5*}). If $v_C \notin X$, then $X' \subseteq G$, and therefore $G \cup E^C$ is nonplanar for any E^C. Otherwise, with b_1, b_2 being the branch vertices of X incident with $\bar{e}(v_C)$, all $v_K \in X$ are either one of b_1, b_2, on the inners of d-edges s-adjacent to those, or branch vertices separated from b_i by $\bar{s}_i \in SP(X)$. Let $S_i := \{v_K | v_K \in X, v_K \text{ s-adjacent to } v_C \text{ in } X \setminus \{\langle v_C, b_{2-i} \rangle\}\}$, and S'_i the set of corresponding nodes in X'. $X \cap \{v_K | K \in \bar{K}\} = S_1 \cup S_2$. Replacing the d-edge containing v_C by any path joining S'_1 with S'_2 results again in a $TK_{3,3}$ or TK^{5*}. Finding such a path is always possible: E^C connects all $K, K \in \bar{K}$, so there is a tree T^{E^C} of connections between the $K, K \in \bar{K}$ through E^C, and it only needs to be provided that on the path between the

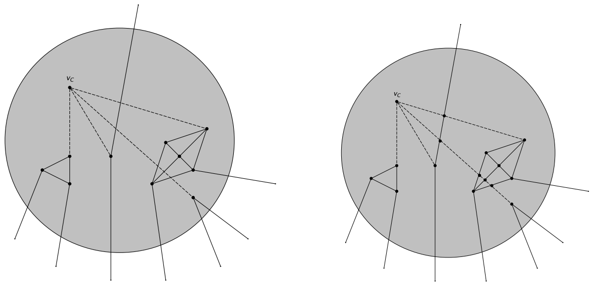


Figure 8.27: G'^S contains a subdivision of G, and $G'^S \setminus (G \setminus C)$ is connected.

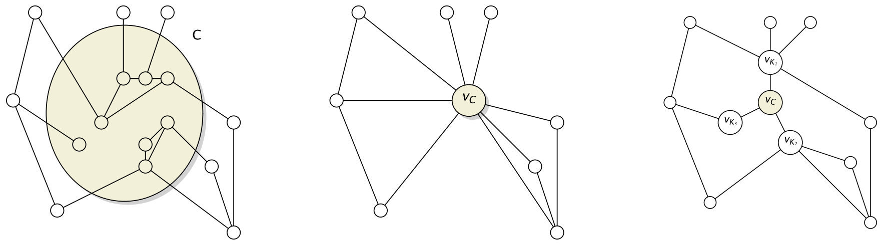


Figure 8.28: Corresponding G, G^C, G^K from Theorem 8.3.1.

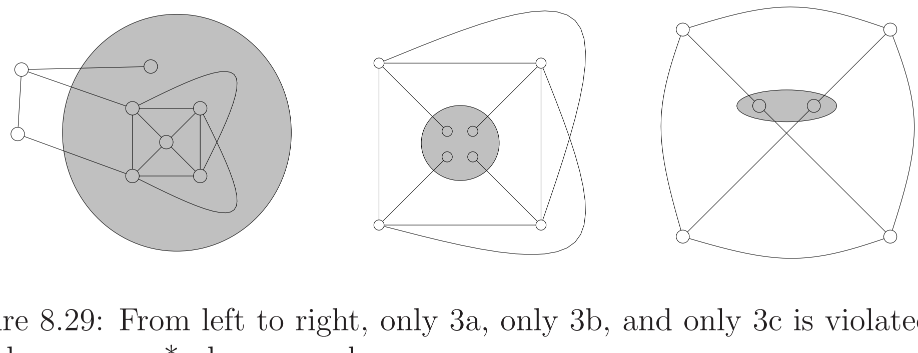


Figure 8.29: From left to right, only 3a, only 3b, and only 3c is violated by the shown non-c*-planar graphs.

chosen K_1 and K_2 there is no other $K \in S_1 \cup S_2$. This can be achieved by choosing K_1 and K_2 "nearest" to each other in T^{E^C} (i.e. with no other $K \in S_1 \cup S_2$ in between), see figure 8.30. Choosing x_1 from the corresponding nodes in K_1 and x_2 from the corresponding nodes in K_2, and creating connections from x_1, x_2 to the endpoints of E^C in each K yields the wanted path. Therefore for each E^C, a $TK_{3,3}$ or TK^{5*} is contained in $G \cup E^C$, as required.

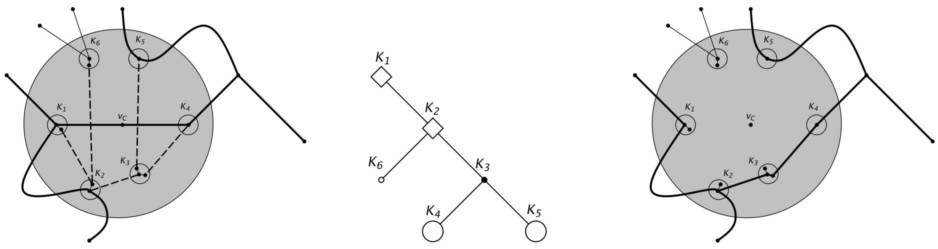


Figure 8.30: In the graph on the left, the thick lines are a part of X', the dashed lines are those in E^C. A T^{E^C} is shown in the middle, with the elements of S_1 shown as rectangles, and of S_2 as circles. As candidates either (K_2, K_4) or (K_2, K_5) can be chosen. On the right, see the final graph created by replacing the d-edge through v_C by a path through E^C.

Theorem 8.3.5. *If Conjecture 8.2.6 holds, then if a graph G with a single cluster C is pseudo-c*-planar, it is also c*-planar.*

Proof. If G is not c*-planar, then according to Theorem 8.3.1 there exist $\hat{G}$ and R, $\hat{G}$ not R-planar. By conjecture 8.2.6 there exists $X \subseteq \hat{G}, X = TK_{3,3}$ or $X = TK^{5*}$ in which for each edge pair $(\langle v_C, v_r^K \rangle, e)$ in R with $e \in E(C)$ and $K \in \bar{K}$, only one is contained in X, or they are s-adjacent. Now if $v_C \notin X$, then $X \subseteq G$, and a) is violated.

Otherwise, if v_C is a branch vertex of X, then in case of $TK_{3,3}$ or a v_C with degree 4, no other branch vertex of X may lie in C; contracting the d-edges starting in v_C to single edges delivers a non-planar minor of G^C, thereby violating b). In case v_C is a node of degree 3 in a TK^{5*}, additionally contract by s connecting v_C with its corresponding node of degree 3 to get the same result.

Lastly, if $v_C \in X$ and v_C is not a branch vertex of X, then no $e \in E(C)$ can lie on an edge not s-adjacent to $\bar{e}(v_C)$. Let a "path through K" denote a path $(w_0, w_1, \ldots, w_{n-1}, w_n)$ with $w_0, w_n \in G \setminus C$, and $w_j \in K \, \forall j = 1, \ldots, n-1$. In order to have c) violated, it only needs to be shown that there exists such an X in which there is at most one disjoint path through each component

K of C, so that the paths can be contracted to single edges, and $K \cap X$ to a single node without changing planarity, yielding a graph isomorphic to a subgraph of G^K. Such an X will be constructed iteratively by the following process: Let assume that for X there exists a component K of C with two disjoint paths W_1, W_2 of X through K. Moreover, let assume that they can be connected by a path in K which does not contain other nodes of X (this can always be assumed, since K is connected). In each step, the total number of disjoint paths through the K_i is decreased by at least one, or a X is obtained which violates b). Therefore the iteration terminates, and delivers the result as required. Iteration step:

1. If W_1, W_2 lie on the same d-edge of X, construct the next X by connecting W_1 and W_2 by a path in K and removing the "loop" (see figure 8.31).

2. If W_1, W_2 lie on different d-edges $\bar{k}_1, \bar{k}_2$ of X on opposite sides of v_C, an X can be constructed which does not contain v_C (see figure 8.32), and therefore violates b). In this case, the iteration can be aborted.

3. If W_1, W_2 lie on different d-edges $\bar{k}_1, \bar{k}_2$ of X on the same side of v_C, the next X is obtained according to figure 8.33.

Therefore, in all cases it is possible to construct an X as required. □

Corollary 8.3.6. *If Conjecture 8.2.6 holds, then a graph G with a single cluster C with two components is c*-planar if and only if G is planar and $(G \setminus C) \cup G_K(C)$ is planar.*

Proof. "⇒": Assume G is c*-planar. By Theorem 8.3.4, G is planar. Moreover, G^K contains no $X = TK_{3,3}$ or $X = TK^{5*}$ with v_C not a branch vertex of X, and $v_C \notin X$ or $\bar{e}(e)$ s-adjacent to $\bar{e}(v_C)$ for all e incident with a

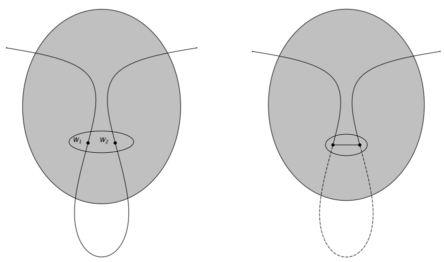


Figure 8.31: K contains two disjoint paths on the same d-edge of X.

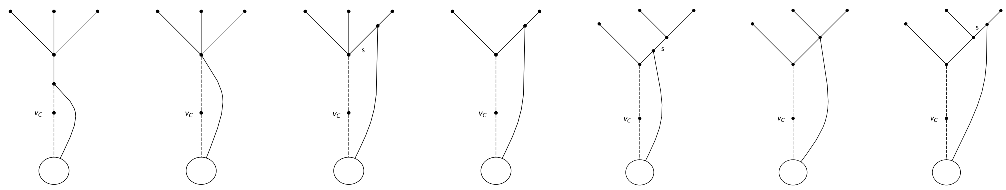


Figure 8.32: K contains two disjoint paths which lie on d-edges on opposite sides of $\bar{e}(v_C)$. Let x_1, x_2 denote the branch vertices on either side of $\bar{e}(v_C)$, and w_1, w_2 the endpoints of the path $W(w_1, w_2)$ connecting the two paths in K, and consider w_j for $j = 1, 2$. If w_j is adjacent to v_C, x_j stays branch vertex; if w_j is the branch vertex separated from v_C by $\bar{s} \in SP(X)$, x_j is not a branch vertex anymore, but w_j becomes a node of degree 4. Otherwise, w_j becomes a new branch vertex, either replacing x_j, or as partner of the split degree-4-node x_j. In all cases, a part of $\bar{e}(v_C)$ containing v_C can be eliminated while still keeping an $X = TK_{3,3}$ or $X = TK^{5*}$; this shows that b) is violated.

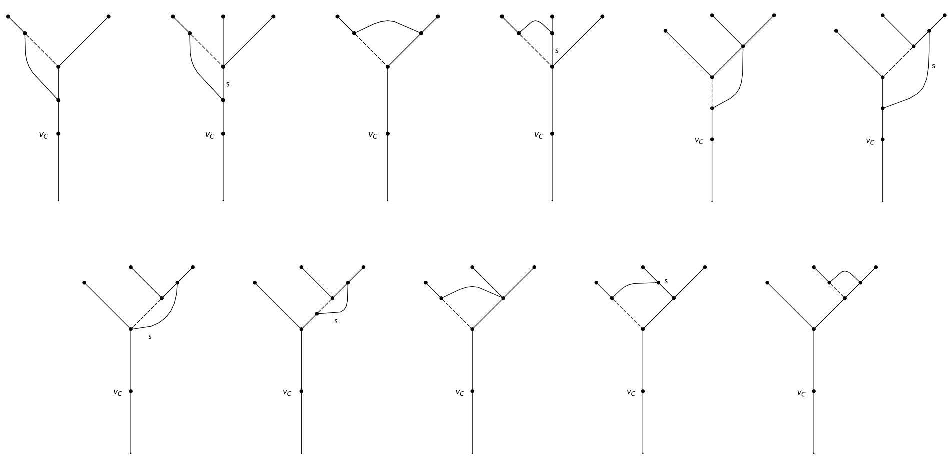


Figure 8.33: K contains two disjoint paths which lie on d-edges on the same side of $\bar{e}(v_C)$. By adding $W(w_1, w_2)$, a new X is obtained which has one disjoint path less; moreover, all edges which were s-adjacent to $\bar{e}(v_C)$, still are, and the newly added edges in K also are.

v_K. However, G^K can also contain no other $X = TK_{3,3}$ or $X = TK^{5*}$, since v_C has degree 2 (and therefore cannot be a branch vertex of X), and since all edges incident with a v_K are adjacent to the one containing v_C if $v_C \in X$ (there are only two v_K in X: v_{K_1}, v_{K_2}, and v_C is connected to only these two nodes).

"$\Leftarrow$": Consider that G is planar and $(G \setminus C) \cup G_K(C)$ is planar; condition a) of pseudo-c*-planarity is trivially fulfilled. G^K contains no $TK_{3,3}$ or TK^5 at all, therefore c) also holds. Lastly, G^C is a minor of G^K, and must therefore be planar. □

8.4 Graphs with Multiple Clusters on One Level

The results from the last chapter are now extended to graphs with multiple clusters on a single level (i.e. no cluster may have nonempty intersection with another).

In the following, let G be a clustered graph containing clusters $\bar{C} = \{C_i | i = 0..n\}$, and let $\bar{K}(C_i)$ denote the components of C_i. Wlog let each node in $G \setminus C_i$ adjacent to C_i have degree 2, and call these nodes *virtual nodes*. The set of virtual nodes adjacent to a set S is denoted by $H(S)$. Set $E_{C_i} := E(C_i \cup \{v | v \in H(C_i)\})$.

Theorem 8.4.1. *G is c*-planar if and only if*

$$\hat{G} := G \cup \bigcup_{C \in \bar{C}} (\{v_C\}, \bigcup_{K \in \bar{K}(C)} \{\langle v_C, v_r^K \rangle | v_r^K \in K \text{ arbitrarily chosen}\})$$

is R-planar with $R = \bigcup_{C \in \bar{C}} \bigcup_{K \in \bar{K}(C), e \in E_C} \{\{\langle v_C, v_r^K \rangle, e\}\}$.

Proof. Perform the steps given in the proof of Theorem 8.3.1 for each cluster $C \in \bar{C}$.

Definition 8.4.2. Define replacements for $G[C_i] \cup E_{C_i}$ as follows:

$$\begin{aligned} F_0(C_i) &:= G[C_i] \cup E_{C_i} \\ F_1(C_i) &:= (\{v_O^{C_i}\}, \{\langle v_O^{C_i}, v \rangle | v \in H(C_i)\} \\ F_2(C_i) &:= G_K(C_i) \end{aligned}$$

Further, let $\mathcal{L}(G)$ denote the set of graphs L obtained from G by replacing $G[C_i] \cup E_{C_i}$ by one of the $F_j(C_i)$ for all clusters C_i, i.e.

$$\mathcal{L}(G) := \bigcup_{\delta \in [0,1,2]^{\bar{C}}} \{(G \setminus \bigcup_{C_i \in \bar{C}} C_i) \cup \bigcup_{C_i \in \bar{C}} F_{\delta(C_i)}(C_i)\}$$

A graph G with clusters $\bar{C}$ on one level is called *pseudo-c*-planar* if for all $L \in \mathcal{L}$, L does not contain a $X = TK_{3,3}$ or $X = TK^{5*}$ with $v_C^{C_i}$ not a branch vertex of X, and $v_C^{C_i} \notin X$ or $\bar{e}(e)$ s-adjacent to $\bar{e}(v_C^{C_i})$ for all e incident with a $v_K^{C_i}$, for all $C_i \in \bar{C}$.

Theorem 8.4.3. *If a graph G with clusters $\bar{C}$ on one level is c*-planar, it is pseudo-c*-planar.*

Proof. Basically the proof can be done by performing the steps as in the proof of theorem 8.3.4 for each $C \in \bar{C}$. This yields the result that for any set $\bar{E} := \{E^C | C \in \bar{C}\}$ making all C connected, $G \cup \bar{E}$ is nonplanar, showing that G is not c*-planar.

However, a different construction has to be used instead of the one shown in figure 8.30 for a violation of condition c) of pseudo-c*-planarity with $v_C \in X$, since applying the construction given there for one cluster can destroy the adjacency constraints for another cluster (see figure 8.34).

In case constraint c) of pseudo-c*-planarity is violated with $v_C \in X$, $X \cap C$ can contain at most one (possibly split) branch vertex of X, together with parts of d-edges adjacent to it. Denote by X' the graph obtained from X by inserting into each d-edge between $x_1 \in C$ and $x_2 \in X \setminus C$ a node p, and denote by X'_1 the component of $X' \setminus C$ which contains more than two branch vertices. X'_1 is a $K_{3,3}$ or K^{5*} with one branch vertex missing between $\{p | p \in X'_1\}$. Therefore, in order to prove $G \cup E^C$ nonplanar, it suffices to have all $p \in X'_1$ connected in $(G \cup E^C) \setminus X'_1$. This is always possible, since all p are adjacent to C, and $C \cup E^C$ is connected.

This construction can be used iteratively for several clusters, since in each step, all d-edges of $X \setminus C$ which were s-adjacent before, are s-adjacent afterwards (or have been eliminated - specifically the parts of d-edges in $X \setminus C$ between two nodes of C).

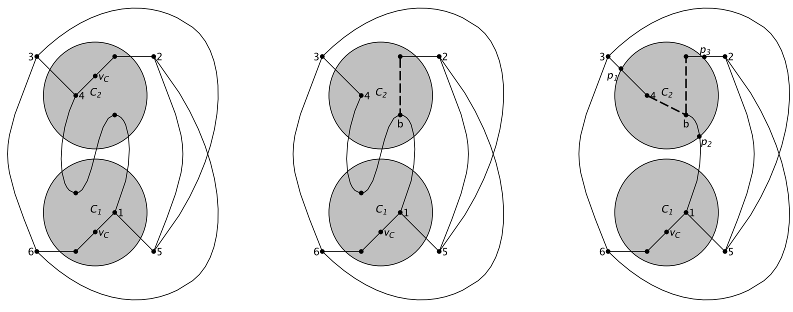


Figure 8.34: If in C_2, the path through E^C has to be chosen via node b instead of node 4, then for C_1, the d-edge between nodes 1 and b is not any more s-adjacent to the d-edge containing v_{C_1}. The alternative construction eliminates the "loop" through $X \setminus C_2$.

Conjecture 8.4.4. *If a clustered graph G with clusters $\bar{C}$ on one level is pseudo-c*-planar, it is c*-planar.*

The attempts to prove this conjecture not only rely on Conjecture 8.2.6, but face additional difficulties, since the proof of Theorem 8.3.5 cannot be trivially extended to the case of multiple clusters: When performing the iteration steps to eliminate multiple disjoint paths through a component of a cluster C_1, it can happen that nodes of another cluster C_2 are not adjacent to one another after the iteration step. To illustrate this, see figure 8.35.

So in order to extend the proof of Theorem 8.3.5, it must be ensured that there is always at least one possibility to perform an iteration step which does not break the requirements on X.

Corollary 8.4.5. *If Conjecture 8.4.4 holds, then a graph G with clusters $\bar{C}$ on one level, with at most two components per cluster, is c*-planar if and only if every graph obtained from G by replacing zero or more clusters C by $G_K(C)$ is planar.*

Proof. Apply the proof of Corollary 8.3.6 for each C.

Corollary 8.4.6. *If Conjecture 8.4.4 holds, then a graph G with clusters $\bar{C}$ on one level, with at most two components per cluster, can be tested for c*-planarity in $O(n \cdot 2^c)$ steps, where n is the number of nodes of G, and c the number of non-connected clusters of G.*

Proof. For a connected cluster C, the checks for graphs U which contain $G_K(C)$ instead of C are not needed, since U is a minor of $(U \setminus G_K(C)) \cup C$, and must therefore be planar if $(U \setminus G_K(C)) \cup C$ is. Performing each planarity test is linear in the number of nodes.

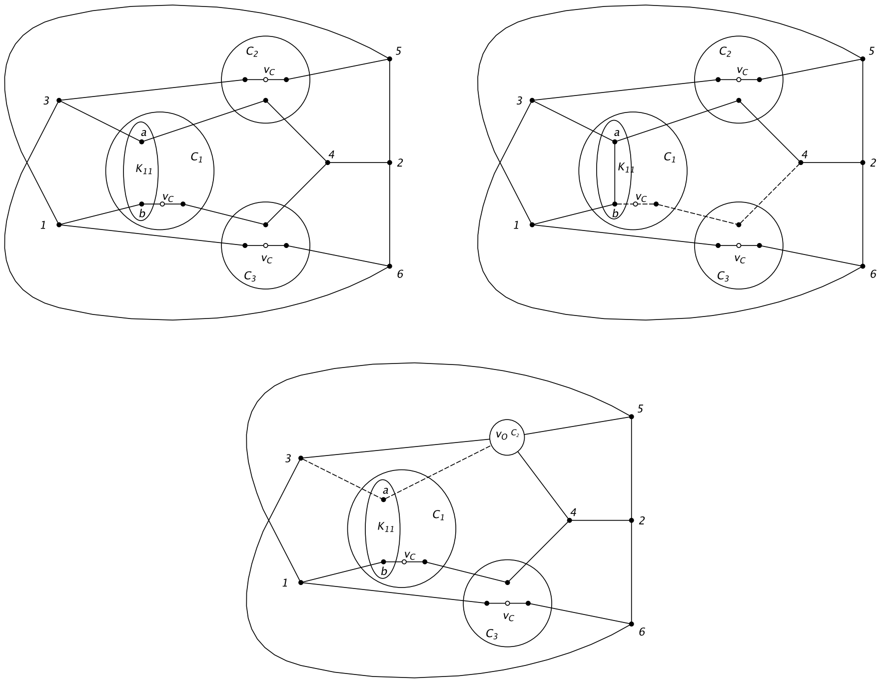


Figure 8.35: If C_1 is processed first, the graph resulting from inserting $\langle a, b \rangle$ and deleting one of the paths joining node 4 and C_1 has an edge not adjacent to that containing v_C through either C_2 or C_3. Therefore the conditions maintained throughout the proof of Theorem 8.3.5 are violated. However, if C_2 is replaced by $v_O^{C_2}$, an X as required is obtained immediately; so what needs to be shown is whether there is always a step which leads to a valid next X.

Chapter 9

c-Planarity Tests for Non-Connected Clusters

9.1 Overview

This chapter presents a polynomial-time algorithm for testing of c-planarity in the sense of definition 3.4.2 for some special cases of non-connected clusters, along with the construction of an embedding for these graphs.

Whereas [GJ02] extended the class of graphs which can be tested in polynomial time to that of "almost c-connected" clustered graphs, this algorithm considers some clustered graphs in which the layout would be rather evident for the human eye, but which are not handled by the traditional planarity test or embedding algorithms.

The algorithm works on clustered graphs for which the non-connected clusters are connected to rather rigid (as to how they can be embedded) structures of the rest of the graph: It requires that the nodes of $G \setminus C$ to which a non-connected cluster C is connected, are contained in a biconnected component of $G \setminus C$, or that there are at most two such nodes.

9.2 Challenges for General c-Planarity Tests

When trying to cope with non-connected clusters, several problems arise:

- The algorithm in [Feng96] works by replacing each cluster C by a representant graph C', for which all planar embeddings have all points corresponding to attachment points of $G \setminus C$ in C bordering a common face of C' (called the "outer face"). The representant graph is made of wheel graphs, in which every face but one is bordered by exactly 3 nodes, and the axis is not adjacent to any $x \in G \setminus C$. Together with the fact that C' is connected, this construction ensures, that for any planar embedding for which parts of $G \setminus C'$ are in a face of C' different from the outer face, there is also an embedding where $G \setminus C'$ is contained in the outer face of C', i.e. an embedding, where C' is contained within a simple closed region. Now, if the cluster C is not connected, the construction yields an unconnected representant graph C'. Replacing C by this representant graph only ensures that each connected component of the cluster will fit into a single simple closed region, but not that the same holds for the cluster as a whole. Moreover, the set of permutations of virtual nodes around a non-connected cluster allowing planar embeddings can not be represented by a single PQ tree.

- It does not suffice to check for each cluster C, whether it is possible to find embeddings G such that the components of C are "connected" via faces, i.e., that for each partition of the components of C into C_1 and C_2, there is at least one face F with $K_1 \in C_1$ and $K_2 \in C_2$ bordering F. This condition is only sufficient if there is e.g. only one unconnected cluster. In the case of more unconnected clusters, it is possible that a face is "needed" by more than one cluster, and possibly in a way that the clusters resp. edges added to connect the clusters have to intersect.

- When trying to stack the components of C around some representant graph of $G \setminus C$, the representant usually only conveys the restrictions imposed by the graph structure or the cluster structure of $G \setminus C$, but not both; therefore, choosing from some stacking order allowed by the representant might not be valid (in the special cases presented below, this problem is met by requiring that the graph structure already imposes very strong restrictions which cannot be narrowed further by the cluster structure without making the graph non-cluster-planar).

9.3 Clusters with Biconnected Attachment

Let an attachment point of A in B be defined as a node $b \in B$ which is adjacent to at least one node a in A, and let $A(C)$ denote the set of all attachment points of C in $G \setminus C$. A cluster C is defined to have *biconnected attachment* if $A(C)$ is contained in a biconnected component of $G \setminus C$.

The algorithm presented in this chapter extends the c-planarity test in [Feng96] to non-connected clusters with biconnected attachment.

9.3.1 Outline of the Algorithm

Let $S(C)$ denote the cut-edges of C, i.e. the edges having one endpoint in C and one in $G \setminus C$. Further, let G' denote the graph obtained from G by replacing each $e = \langle x, r \rangle, e \in S(C)$ by a so called *virtual node* v_e ($v_e \notin C$) and two edges $\langle x, v_e \rangle, \langle v_e, r \rangle$ connecting v_e to the original endpoints of e. Let H be the set of all virtual nodes.

Clearly, each planar embedding of G' corresponds to exactly one planar embedding of G. The planar embeddings of G' in which C can be drawn within a simple closed region are exactly those obtained from a planar embedding $\mathcal{E}(G' \setminus C)$ for which all $v_e \in H$ border the same face, and a planar embedding $\mathcal{E}(C \cup H)$ for which all $v_e \in H$ border the same face, and the ordering of the virtual nodes v_e around C and around $G \setminus C$ is the same.

Now let $\mathcal{O}(U, T)$ denote the set of orderings of the elements of T around U assigned by the individual embeddings of $U \cup T$ in which all elements of T are on a single face of $\mathcal{E}(U)$. G' has a planar embedding with C contained within a simple closed region if and only if $\mathcal{O}(C, H) \cap \mathcal{O}(G \setminus C, H) \neq \emptyset$. Therefore, the result is not changed if C is replaced by any other graph C' for which $\mathcal{O}(C, H) \cap \mathcal{O}(G \setminus C, H) = \mathcal{O}(C', H) \cap \mathcal{O}(G \setminus C', H)$.

The algorithm tries to find a connected representant graph C' for each non-connected cluster C, such that $\mathcal{O}(C, H) \cap \mathcal{O}(G \setminus C, H) = \mathcal{O}(C', H) \cap \mathcal{O}(G \setminus C', H)$. For all cases where it can be constructed, the planarity testing algorithm presented in [Feng96] can be immediately applied by replacing the representant graph generation step by the construction for non-connected clusters. If it cannot be constructed, G is not c-planar.

9.3.2 Step 1: Obtain a Representant for $G' \setminus C$

For a biconnected graph B and nodes $S \subseteq B$, the ordering of S around each common face is the same. If virtual nodes H are attached to S, and each virtual node is also connected to a new common node v_C, the possible orderings of H around v_C (and, equivalently, around B) can be obtained by performing a PQ tree planarity test on the graph $K = (V(B \cup H) \cup \{v_c\}, E(B \cup H) \cup \{\langle v_c, v_e \rangle | v_e \in H\})$ with $s = v_e$ arbitrary and $t = v_c$, stopping just before the reduce step for v_c.

An example is shown in figures 9.1, 9.2, 9.3; figure 9.4 shows the representant graph for the PQ-tree obtained as described in [Feng96], which allows the same orderings of the virtual nodes as the original graph.

If applied to $G' \setminus C$, a connected graph representing $\mathcal{O}(G' \setminus C, H)$ is obtained. This graph is made up of a single wheel $R \cup \{x_R\}$ (x_R being the axis) and the virtual nodes H each connected to one $r \in R$ by a single edge.

The elements of R are labeled $r_1, \ldots, r_n$, starting with an arbitrary element of R, and proceeding along the circle formed by R. Let $\hat{R} := R \cup \{x_R\}$, $h(K_j, r_l) := \{v_e | e = \langle y, r_l \rangle \wedge y \in K_j\}$, $h(K_j) := \bigcup_{r_l \in R} h(K_j, r_l)$, $h(r_l) := \bigcup_{K_j \subseteq C} h(K_j, r_l)$. Further, let $A_R(K_j) := \{r_l | h(K_j, r_l) \neq \emptyset\}$.

9.3.3 Step 2: Find Possible Ordering of H around C

First, for each component K_j of C, a planarity test is performed on $K_j \cup h(K_j) \cup \hat{R}$. If such a test fails, G is not c-planar. Otherwise, select an ordering $\pi_j \in \mathcal{O}(K_j, h(K_j))$ as obtained from the planarity test, and let $\pi_j^{r_l}$ refer to π_j restricted to the virtual nodes connected to r_l.

These π_j must now be combined to yield a $\pi \in \mathcal{O}(C, H)$. First, see that it suffices to consider orderings in which $h(K_j, r_l)$ is consecutive within π for all j, l: If there is a planar embedding in which virtual nodes from $h(K_m, r_l)$ are "between" two from $h(K_j, r_l)$, then all virtual nodes from $h(K_m)$ must appear between those two (otherwise there would be a crossing). So $A_R(K_m) = \{r_l\}$, and there also exists a planar embedding in which $h(K_m, r_l)$ is not between the two virtual nodes from $h(K_j, r_l)$ (one could e.g. regard the embedding where all $v \in h(K_m, r_l)$ are consecutive at the end of the ordering of $h(r_l)$; in this embedding, they won't appear anywhere "between" other virtual nodes belonging to the same K_r).

So what is left to do is to find a "stacking order" of the K_j around $\hat{R}$ which does not introduce any crossings rsp. show that there is no such stacking order. This is done using the following process:

For each r_l, define lists $down(r_l)$ and $up(r_l)$, and $sep(r_l)$, initially empty. Let $l_{min}(K_j) := \min\{l | r_l \in A_R(K_j)\}$ and $l_{max}(K_j) := \max\{l | r_l \in A_R(K_j)\}$.

Then, for all K_j with $|A_R(K_j)| = 1, A_R(K_j) = \{r_l^{K_j}\}$, append $\pi_j^{r_l}$ to $up(r_l)$, and mark K_j. Define the operation $AssignOrdering(lower, upper)$ as follows:

- Determine the set M of yet unmarked components K_j with $l = l_{min}(K_j)$ minimal in $[lower, upper)$, and with $u = l_{max}(K_j)$ maximal among these. Since all K_j with $|A_R(K_j)| = 1$ are already processed, $u > l$.
- If $M = \emptyset$, return.
- If $u > upper$ then the graph is not c-planar, abort (see figure 9.5).
- If $|M| = 1$, then use this single component for the following operations, else:
  - If at least two $K_j \in M$ have $|A_R(K_j)| \geq 3$, then the graph is not c-planar, abort (see figure 9.6).

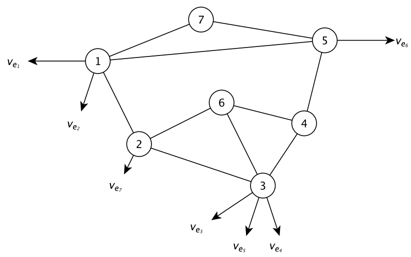


Figure 9.1: $B \cup H$.

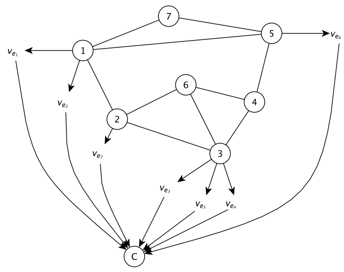


Figure 9.2: $K = B \cup H \cup \{v_c\}$.

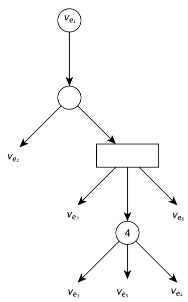


Figure 9.3: PQ-tree for K with $s = v_{e1}$ and $t = v_c$.

  - If only one $K_j \in M$ has $|A_R(K_j)| \geq 3$, then the only possibility to obtain a c-planar embedding is to stack K_j inside all other components $K_r \in M$. Therefore, choose any $K_r \in M$ with $r \neq j$ for the following operations (figure 9.7).
  - Else all $K_j \in M$ have $|A_R(K_j)| = 2$. These components can be stacked in any order around $\hat{R}$, so choose any of these components for the following operations.

- Let K_j be the component chosen above. Append $\pi_j^{r_l}$ to $up(r_l)$, prepend $\pi_j^{r_u}$ to $down(r_u)$, and for all $r \in A_R(K_j) \setminus \{r_l, r_u\}$, set $sep(r) = \pi_j^r$.
- Mark the component.
- Define $n_1, ..., n_{|A_R(K_j)|}$ as the indices of the elements of $A_R(K_j)$ in ascending order (with $r_{n_1} = l_{min}(K_j), r_{n_{|A_R(K_j)|}} = l_{max}(K_j)$). Invoke $AssignOrdering(n_i, n_{i+1})$ for all $i = 1, \ldots, |A_R(K_j)| - 1$.

Invoking $AssignOrdering(1, n)$ returns either that the graph is not c-planar, or it returns all components marked, and $down, up, sep$ filled. In this case, the ordering $\pi \in \mathcal{O}(C, H)$ of the virtual nodes can be obtained by concatenating the lists $up(r_l), sep(r_l), down(r_l))$ for each $r_l, 1 \leq l \leq n$ in this order.

9.3.4 Step 3: Connected Representant Graph for C

Since a unique ordering of the virtual nodes adjacent to R around R is now known, a wheel graph can be constructed by forming a cycle from the virtual nodes (in the order given by π) and an axis. The virtual nodes must be connected to the nodes to which they were connected in G', i.e., to $x \in G \setminus C$ for all v_e with $e = \langle y, x \rangle$.

9.3.5 Step 4: Construct an Embedding

An embedding for a c-planar graph can be constructed in the same way as in [Feng96], only now the embeddings for the representant graphs constructed by the above algorithm must be replaced by embeddings of the original clusters.

The steps performed in the construction already give the information necessary: The planarity tests done for subgraphs corresponding to the components give planar embeddings for each component and the virtual nodes connecting it to R. These embeddings have to be assembled in the stacking

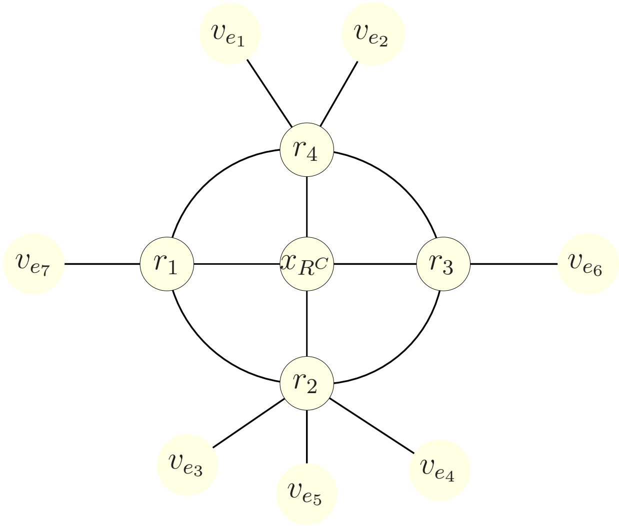


Figure 9.4: Representant graph R for PQ-tree.

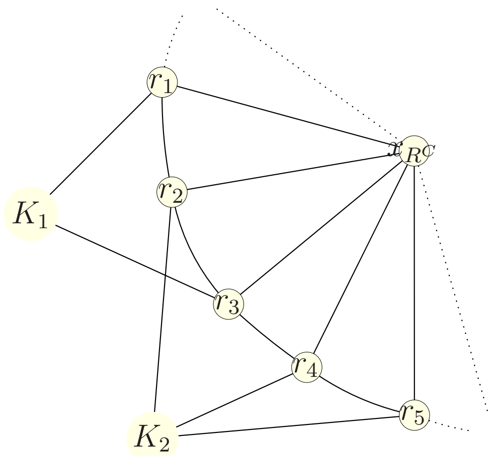


Figure 9.5: *AssignOrdering* invoked with $lower = 1$, $upper = 3$, $M = \{K_2\}$. $l_{max}(K_2) = 5 > upper$, no planar embedding possible.

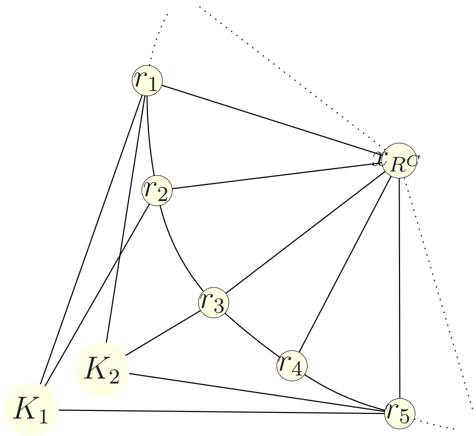


Figure 9.6: *AssignOrdering* invoked with *lower* = 1, *upper* = 5. More than one $K_j \in M$ has $|A_R(K_j)| > 2$, no planar embedding possible.

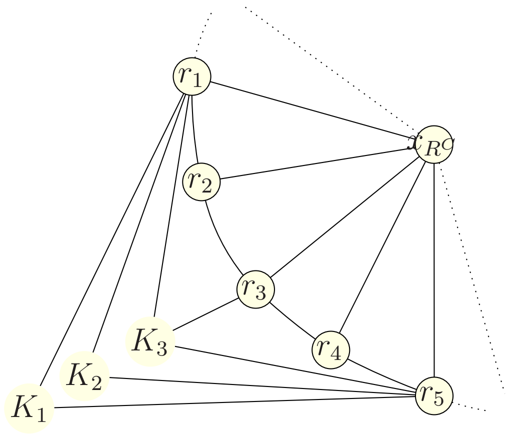


Figure 9.7: *AssignOrdering* invoked with *lower* = 1, *upper* = 5. Only K_3 has $|A_R(K_j)| > 2$, must be stacked inside the others. The ordering of K_1 and K_2 is irrelevant.

order determined, which yields a planar embedding of C together with its virtual nodes.

By contracting the virtual nodes into the nodes of $G \setminus C$, an embedding of the original graph G is obtained.

9.3.6 Alternative Construction of an Embedding

If an embedding is to be created, the existing algorithms can more easily be employed if some edges are inserted into G which make C connected while preserving planarity. These can be removed again after creating the embedding. So instead of steps 3 and 4, the following procedure can be applied to obtain a connected $C' = C \cup E^C$:

- Set v_{prev} = undefined
- $\forall r_l \in \hat{R}$ sorted by l: Let $\pi(r_l)$ denote the set of virtual nodes connected to r_l, in the order determined by π as obtained in step 2, and let $v_1^l, ..., v_{m^l}^l$ denote the elements of $\pi(r_l)$, in this order.
  - move to next r_l if $\pi(r_l) = \emptyset$
  - insert an edge $\langle v_{prev}, v_1^l \rangle$ into E^C if v_{prev} is defined
  - $\forall j = 1, ..., m^l - 1$: insert an edge $\langle v_j^l, v_{j+1}^l \rangle$ into E^C
  - set $v_{prev} = v_{m^l}^l$

By this construction, $C' = C \cup H$ is connected in $G' \cup E^C$, and G' is c-planar with cluster C'. Therefore, the standard algorithm can be applied to C' and G', and any drawing created for it can be transformed into a c-planar drawing of G and C by deleting the edges from E^C and contracting the $v_e \in H$ with one of their endpoints.

9.3.7 Complexity

It takes $O(|A_R(K_j)| \cdot \log |A_R(K_j)|)$ steps to sort the virtual nodes belonging to each component, and $O(|\bar{K}| \cdot \log |\bar{K}|)$ steps to sort the components by l_{min}, l_{max}, and $|A_R(K_j)|$. *AssignOrdering* performs actions other than determining M once for every K_j; it is additionally called at most $|R|$ times. Since $|A_R(K_j)|$, $|\bar{K}|$ and $|R|$ are bounded by $|H| = |S(C)|$, the running time of finding a connected representant for C is bounded by $O(|S(C)| \cdot \log |S(C)|)$.

With $n = |V(G)|, m = |E(G)|, c = |\{C | C \text{ non-connected cluster of } G\}|$, the overall complexity of the c-planarity testing algorithm from [Feng96] extended to non-connected clusters with biconnected attachment is therefore bounded by $O(n^2 + c \cdot m \cdot \log m)$.

9.4 Clusters with $|A(C)| \leq 2$

If a cluster C has only one or two attachment points in $G \setminus C$, the same process as for clusters with biconnected attachment can be used to create a connected representant graph.

This is possible since the ordering of R around C is completely determined (for ≤ 2 elements, there is only one).

Chapter 10

Conclusion

The initial rather ambitious aim of this work was to find out the computational complexity of cluster planarity testing. While this aim was not reached, at least some steps were taken in this direction. Part of the results however depend on conjecture 8.2.6 to prove true:

- A new notion of cluster planarity has been introduced, which can deliver readable drawings while being less restrictive. Moreover, tests for this property yield tests for the classical definition.
- A characterization of c*planarity in terms of forbidden subgraphs has been established (depends on conjecture 8.2.6). This characterization could be employed to identify subgraphs of a clustered graph which can be tested individually to yield an overall result (in the same way as for basic graphs, planarity is equivalent to planarity of its blocks). Further, if proven, it could help devising new tests, by exposing what makes a graph non-cluster-planar.
- An algorithm polynomial in the number of vertices, and exponential in the number of nonconnected clusters has been given for clustered graphs with at most two components per cluster.
- An algorithm polynomial in the number of vertices and clusters has been given for clustered graphs in which each non-connected cluster has biconnected attachment, or is connected to at most two vertices.
- A litte off-topic, a characterization of weak realizability has been established (depends on conjecture 8.2.6). This could be useful also in other areas of graph theory, probably most in those related to planarity.

Additionally, during research for this work, many attempts were done to reduce $\mathcal{NP}$-complete problems to cluster-planarity tests, however, none of the constructions proved suitable.

So even though some results have been achieved (and the class of graphs that can be tested for cluster planarity efficiently has been extended), the initial question remains open. Moreover, for a part of the results, a proof needs completion, so what remains to be done after this work is even more than there was in the first place:

- Prove conjecture 8.2.6.
- Reveal the computational complexity of cluster planarity testing, either by giving efficient algorithms for all clustered graphs, or by showing e.g. $\mathcal{NP}$-hardness.

Part IV

Appendix

Bibliography

[Ber73] Berge, C., *Graphs and Hypergraphs*. North-Holland Publishing, 1973

[BL76] Booth, K.S., Lueker, G.S., *Testing for the consecutive ones property, interval graphs and graph planarity using PQ-tree algorithms*. Journal of Computer and Systems Sciences, 13:335-379, 1976

[BM99] Boyer, J., Myrvold, W., *Stop minding your P's and Q's: A simplified $\mathcal{O}(n)$ planar embedding algorithm*. Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, 140-146, 1999

[CBPP04] Cortese, P.F., Battista, G., Patrignani, M., Pizzonia, M., *Clustering Cycles into Cycles of Clusters*. Graph Drawing 2004, LNCS 3383:100-110, 2004

[CNA85] Chiba, N., Nishizeki, T., Abe, S., Ozawa, T., *A linear algorithm for embedding planar graphs using PQ-trees*. Journal of Computer and Systems Sciences, 30(1):54-76, 1985

[CW03] Cornelsen, S., Wagner, D., *Completely connected clustered graphs*. Proceedings of the 29th International Workshop on Graph Theoretic Concepts in Computer Science (WG 2003), LNCS 2880:168-179, 2003

[Dah98] Dahlhaus, E., *Linear time algorithm to recognize clustered planar graphs and its parallelization (extended abstract)*. Lecture Notes in Computer Science, 1380:239-248, 1998

[Deo76] Deo, N., *Note on Hopcroft and Tarjan's planarity algorithm*. Journal of the ACM, 23(1):74-75, 1976

[Die00] Diestel, R., *Graphentheorie*. 2. Auflage, Springer-Verlag, 2000

[Feng96] Feng, Q., *Algorithms for Drawing Clustered Graphs*. Doctoral Thesis submitted to the University of Newcastle, 1997

[FR82] Fraysseix, H. de, Rosenstiehl, P., *A depth-first-search characterization of planarity*. Annals of Discrete Mathematics, 13:75-80, 1982

[GJ02] Gutwenger, C., Jünger, M., Leipert, S., Mutzel, P., Percan, M., Weiskircher, R., *Advances in C-Planarity Testing of Clustered Graphs*, Graph Drawing 2002, LNCS 2528:220-232, 2002

[Har88] Harel, D., *On visual formalisms*. Communications of the ACM, 31(5):514-530, 1988

[HT74] Hopcroft, J., Tarjan, R., *Efficient planarity testing*. Journal of the Association for Computing Machinery, 21:549-568, 1974

[JLP02] Jünger, M., Leipert, S., Percan, M., *Triangulating clustered graphs*. Technical report, Zentrum für Angewandte Informatik Köln, 2002

[Kra98] Kratochvil, Jan, *Crossing Number of Abstract Topological Graphs*. Graph Drawing 98

[Kur30] Kuratowski, C., *Sur le problème des courbes gauches en topologie*. Fundations of Mathematics, 15:271-283, 1930

[LEC67] Lempel, A., Even, S., Cederbaum, I., *An algorithm for planarity testing of graphs*. Theory of Graphs: Int Symposium (Rome 1966), 215-232, 1967

[MM96] Mehlhorn, K., Mutzel, P., *On the embedding phase of the Hopcroft and Tarjan planarity testing algorithm*. Algorithmica 16:(2):233-242, 1996

[MS93] Misue, K., Sugiyama, K., *An overview of diagram based idea organizer: D-ABDUCTOR*. Technical Report IIAS-RR-03-3E, ISIS, Fujitsu Laboratories, 1993

[Mut96] Mutzel, P., *An implementation of the Hopcroft and Tarjan planarity test*. Technical Report MPI-1-93-151, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1996

[SH99] Shih, W.K., Hsu, W.L., *A new planarity test*. Theoretical Computer Sciences, 223:179-191, 1999