

# Computing Maximum C-planar Subgraphs

Markus Chimani, Carsten Gutwenger, Mathias Jansen,  
Karsten Klein, and Petra Mutzel

Technische Universität Dortmund, Germany  
{markus.chimani, carsten.gutwenger, mathias.jansen,  
karsten.klein, petra.mutzel}@cs.tu-dortmund.de

**Abstract.** Deciding c-planarity for a given clustered graph  $C = (G, T)$  is one of the most challenging problems in current graph drawing research. Though it is yet unknown if this problem is solvable in polynomial time, latest research focused on algorithmic approaches for special classes of clustered graphs. In this paper, we introduce an approach to solve the *general* problem using integer linear programming (ILP) techniques. We give an ILP formulation that also includes the natural generalization of c-planarity testing—the *maximum c-planar subgraph problem*—and solve this ILP with a branch-and-cut algorithm. Our computational results show that this approach is already successful for many clustered graphs of small to medium sizes and thus can be the foundation of a practically efficient algorithm that integrates further sophisticated ILP techniques.

## 1 Introduction

Drawing clustered graphs is a prevalent problem in practical applications of graph drawing, e.g., to group nodes into departments, as well as in graph theory, since the occurring graph theoretical problems are in particular challenging, even in simplified special cases. A clustered graph  $C = (G, T)$  is formally defined as a graph  $G = (V, E)$  together with a rooted tree  $T$ , the *inclusion tree* of  $C$ , where the leaves of  $T$  are the vertices of  $G$ . Each node  $\nu$  of  $T$  represents a *cluster* of the vertices  $V(\nu)$  of  $G$  that are leaves of the subtree rooted at  $\nu$ .

In a drawing of a clustered graph, the clusters themselves are drawn as simple regions, e.g., rectangles, and special aesthetic criteria on the drawing need to be met to guarantee readability. In particular, we call a drawing *c-planar*, if there are neither edge–edge nor edge–region crossings and the drawing of a cluster  $\nu$  is contained in the interior of the region of a cluster  $\mu$  if and only if  $\mu$  lies on the path from  $\nu$  to the root of  $T$ . A *c-planar* clustered graph is a clustered graph for which a c-planar drawing exists.

Though c-planarity has been intensively studied in the past years, the complexity of deciding c-planarity is still unknown. Instead of considering the general problem, latest research focused on special classes of clustered graphs. Besides the well known results by Feng et al. [8] and Dahlhaus [4] for *c-connected* clustered graphs, i.e., clustered graphs where the vertices  $V(\mu)$  of each cluster  $\mu$  induce a connected graph, various classes of non-c-connected clustered graphs

have been studied [10, 3, 9, 6, 11]. In contrast to this, we tackle the general c-planarity problem in this paper by presenting the foundation of an ILP-based approach consisting of an ILP formulation and a branch-and-cut algorithm.

In order to draw not necessarily c-planar clustered graphs, Di Battista et al. [5] adapted the topology-shape-metrics approach to clustered graphs and described a planarization-based method for crossing minimization. This method first computes a c-planar subgraph  $C'$ , and then reinserts the deleted edges successively into a c-planar embedding of  $C'$ , so that only a small number of crossings is produced. Our ILP approach also solves the first problem of this c-planarization approach, i.e., finding a c-planar subgraph of maximum size:

**Definition 1 (Maximum C-planar Subgraph Problem (MCPSP)).** *Given a clustered graph  $C = (G = (V, E), T)$  find a c-planar clustered graph  $C' = (G' = (V, E'), T)$  with  $E' \subseteq E$  such that  $E'$  has maximum cardinality.*

Obviously, MCPSP is NP-hard, since the maximum planar subgraph problem is already NP-hard. This paper is organized as follows. Section 2 presents our ILP formulation for MCPSP and a branch-and-cut algorithm for solving the ILP; an experimental evaluation of this algorithm is given in Sect. 3.

## 2 ILP and Branch-and-Cut

In the following, let  $C = (G = (V, E), T)$  be the given clustered graph with edge set  $E$ . For a cluster  $\nu$  in  $C$  let  $E(\nu)$  denote the edge set induced by the vertices  $V(\nu)$  in cluster  $\nu$ , and let  $E(\bar{\nu})$  denote the edge set induced by the vertices in  $V(\bar{\nu}) = V \setminus V(\nu)$ .

We say a c-connected clustered graph is *completely connected* if, for each non-root cluster  $\nu$ , the subgraph by  $V(\bar{\nu})$  is connected. For our formulation we need the following result by Cornelsen and Wagner [2]:

**Theorem 1.** *A clustered graph is c-planar if and only if it is a subgraph of a c-planar completely connected clustered graph. A completely connected clustered graph  $C = (G, T)$  is c-planar if and only if its underlying graph  $G$  is planar.*

Our central concept for the formulation then is to (a) augment the given clustered graph such that it becomes completely connected, and (b) to ensure that the resulting graph, disregarding the cluster structure, is planar:

**Corollary 1.**  $C^*$  is a maximum c-planar subgraph of  $C$  if and only if it is the largest subgraph with the property that there exists a completely connected clustered graph  $C'$  such that (a)  $C^*$  is its subgraph and (b) the underlying graph of  $C'$  is planar. If  $C^* = C$ ,  $C$  is c-planar.

In the following we will hence concentrate on finding such a completely connected *solution graph*  $C' = (G' = (V, E'), T)$ .

## 2.1 The ILP Formulation

We define the set  $F$  as the complement of  $E$ , i.e.,  $F$  are the potential edges for the augmentation. This allows us to introduce our two variables

$$x_e, y_f \in \{0, 1\} \quad \forall e \in E, f \in F \quad (1)$$

which are 1 if the corresponding edge is contained in the solution graph, and 0 otherwise. Then we can write the objective function as

$$\max \sum_{e \in E} x_e - \varepsilon \sum_{f \in F} y_f. \quad (2)$$

We want to maximize the number of original edges in the solution and use as few augmenting edges as possible. In order for the latter criterion to not interfere with the main optimization goal, we restrict its influence by the introduction of  $\varepsilon := \frac{0.1}{3n}$ ; due to Euler's formula this guarantees that the second term in (2) does not grow larger than 0.1.

We have two sets of constraints: the first set guarantees that the solution graph  $C'$  is completely connected; the second set ensures planarity of  $G'$ .

*Connectivity Constraints.* A *cut set*  $W|A$  with  $W \subseteq V$  and  $A \subseteq E$  in the graph  $G = (V, E)$  is defined as the set of edges in  $A$  that are incident to exactly one vertex of  $W$ . A graph is connected if and only if the cardinality of  $W|E$  is at least 1 for any  $\emptyset \neq W \subset V$ . We define the *connectivity constraints* as:

$$\sum_{e \in W|E(\xi)} x_e + \sum_{f \in W|F(\xi)} y_f \geq 1 \quad \forall \nu \in T, \forall \xi \in \{\nu, \bar{\nu}\}, \forall \emptyset \neq W \subseteq V(\xi) \setminus \{w_\xi\} \quad (3)$$

While the case  $\xi \in T$  only guarantees  $c$ -connectivity, the additional constraints with  $\xi \notin T$  are necessary to ensure complete connectivity. We use  $W \subseteq V(\xi) \setminus \{w_\xi\}$  for some fixed  $w_\xi \in V(\xi)$  instead of  $W \subset V(\xi)$  to avoid redundancy.

*Kuratowski Constraints.* In order to guarantee that the solution graph is planar we use Kuratowski constraints as introduced for the maximum planar subgraph problem [12]. These constraints are based on Kuratowski's theorem [14] which states that a graph is planar if and only if it does not contain a subdivision of  $K_5$  or  $K_{3,3}$ . We call these subdivisions *Kuratowski subdivisions*, and represent them by their edge sets. Let  $\mathcal{K}$  be the set of all Kuratowski subdivisions in  $(V, E \cup F)$ . For any  $K \in \mathcal{K}$ , the solution graph will not contain all edges of  $K$ , as this would contradict its planarity. We hence formulate the Kuratowski constraints as

$$\sum_{e \in K} x_e + \sum_{e \in K} y_e \leq |K| - 1 \quad \forall K \in \mathcal{K}. \quad (4)$$

**Theorem 2.** *The ILP*

$$\left\{ \max \sum_{e \in E} x_e - \varepsilon \sum_{e \in F} y_e, \text{ subject to (1), (3), and (4)} \right\}$$

*solves the maximum  $c$ -planar subgraph problem. If  $x_e = 1$  for all  $e \in E$ , the given clustered graph is  $c$ -planar.*

## 2.2 Branch-and-Cut

Both constraint sets contain an exponential number of constraints and hence it is not applicable to generate all constraints in advance. We solve the ILP within a branch-and-cut framework: we start with a small subset of constraints, drop the integrality constraints, and apply cutting-plane algorithms to add additional constraints as required. The problem of identifying such cuts after obtaining a fractional solution of the partial LP-relaxation is called *separation problem*.

*Separation.* Separating the connectivity constraints can be done in polynomial time by computing minimum cuts on the graph, using the fractional solution as edge capacities. On the other hand, there are no known polynomial algorithms for the Kuratowski constraint separation, and we have to resort to a heuristic routine, similar to the ones described in [12]: we round the fractional solution to an integer solution, which we can interpret as our *support graph*  $S$ , and search for Kuratowski subdivisions in  $S$ . For any such subdivision  $K$  we can test whether the current fractional solution violates the constraint induced by  $K$ .

Traditional planarity test algorithms can extract a single Kuratowski subdivision per run; in our experiments we use the extended test algorithm presented in [1] which extracts multiple different subdivisions in linear time. Note that we separate all cut constraints before separating any Kuratowski constraints.

*Branching and Primal Heuristic.* If we have a fractional solution, but cannot find any violated constraints, we have to resort to branching. In such cases good LP-based heuristics become crucial, to prune nodes early in the branch-and-bound tree. Our heuristic works as follows: We start by computing a spanning tree recursively for each cluster in a bottom-up scheme on  $T$ , using the fractional solution as negative weights. Merging all these minimum spanning trees, we obtain a  $c$ -connected and  $c$ -planar spanning tree  $R$ . We sort the remaining edges based on their fractional values, and iteratively try to add them to  $R$  in decreasing order. This can be done in polynomial time, since planarity testing of a  $c$ -connected clustered graph is polynomial. We obtain a maximal  $c$ -connected,  $c$ -planar subgraph  $R$  that implies a  $c$ -planar subgraph of  $C$ .

## 3 Computational Experiments and Discussion

We report on the results of our experimental evaluation. The main intention of this short study is to point out the feasibility of our approach, without giving attention to speed-up techniques like strong preprocessing and heuristics, column generation, etc. We implemented our approach within the Open Graph Drawing Framework ([www.ogdf.net](http://www.ogdf.net)) using the branch-and-cut framework ABACUS [13] with CPLEX 9.0 as LP-solver. The experiments were run on a 2.33GHz Intel Xeon with 2GB RAM per process and a time limit of 30 minutes per instance.

In addition to solving the MCPSP, we also experimented with a variant where only  $c$ -planarity is tested; in this case no maximum  $c$ -planar subgraph needs to be computed and subproblems are pruned as soon as their dual bound proves that an original edge would have to be deleted.

*Benchmark Set.* We created a benchmark set based on the Rome graphs [7] by generating cluster hierarchies on top of each graph of the library. The library contains planar and non-planar graphs; key properties are shown in Table 1(top).

We create a cluster structure by randomly picking vertices in a cluster  $\nu$ , starting with the root cluster, and after each pick, a random decision is made if a new cluster is generated with the vertices picked so far, up to a maximum number of 9 clusters. We restrict the maximum cluster tree depth to two levels (in addition to the root cluster), the number of edges to 30, and divide the created clustered graphs into two groups depending on the planarity of the underlying graph. The benchmark set can be found at [ls11-www.cs.uni-dortmund.de/people/klein/clusterbenchmarks08.zip](http://ls11-www.cs.uni-dortmund.de/people/klein/clusterbenchmarks08.zip).

*Results and Discussion.* Figure 1(bottom-left) shows the resulting running times required by our approach, relative to the graph size; the table on the bottom right summarizes the runtime performance of the instances, depending on the planarity of the underlying graph. We see that restricting the computation to pure c-planarity testing by pruning leads to decreases in the overall average computation time, but does not necessarily need to speed up the computation for each instance, because subproblems containing the maximum c-planar subgraph may be pruned, which extends the search in the branch tree.

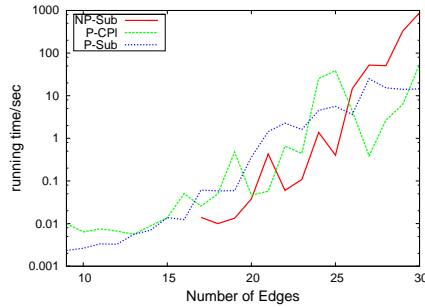
Our main observation is that the performance on most of the test graphs is promising: only 2 non-planar and 17 planar graphs could not be solved within the time limit; the 95%-percentile shows that long running cases are extremely rare. The average running time of the c-connected clustered graphs is below 0.02 seconds, indicating that the ILP performs well on this polynomial time solvable class. We therefore conjecture that the ILP may be useful as a tool when developing c-planarity tests for special graph classes, as the ILP may give hints on the classes' hardness.

*Conclusion and Future Work.* We introduced the Maximum C-planar Subgraph Problem and presented an ILP formulation together with a branch-and-cut approach to solve it to optimality. Our brief experimental evaluation showed the general feasibility of the concept. We believe that our branch-and-cut approach can be improved to also cope with harder instances, which is part of our future work, especially by using stronger heuristics and preprocessing to reduce the search space, as well as pricing instead of adding all possible variables in advance. Encouraged by the results on the c-connected graphs, we also plan to perform a closer investigation of the behavior of our branch-and-cut approach with regard to other polynomial time solvable classes of clustered graphs.

## References

1. Chimani, M., Mutzel, P., Schmidt, J.M.: Efficient extraction of multiple kuratowski subdivisions. In: GD '07. LNCS, vol. 4875, pp. 159–170. Springer (2007)
2. Cornelsen, S., Wagner, D.: Completely connected clustered graphs. *J. Discrete Algorithms* 4(2), 313–323 (2006)

	# inst.	compl.			Clusters			Vertices	Edges
		c-plan.	c-con.	con.	min	avg	max	max	max
Planar graphs	1815	1494	25	2	3	4	9	29	30
Non-planar graphs	116	0	3	0	3	5.2	9	26	30



	Running time (sec)			
	min	avg	95%	max
P-Sub	0.01	4.9	9.6	1460.9
NP-Sub	0.01	40.9	18.7	1456.5
P-CPI	0.01	4.3	6.1	249.9

**Fig. 1: (top)** Properties of the benchmark instances. **(bottom)** Average runtime performance of the branch-and-cut algorithm. P-Sub and NP-Sub are running times for solving the MCPSP on the (non-)planar graphs, respectively. P-CPI denotes the runtime for the c-planarity test on the instances with underlying planar graphs. 95% denotes the 95%-percentile.

- Cortese, P.F., Di Battista, G., Patrignani, M., Pizzonia, M.: Clustering cycles into cycles of clusters. In: GD '04. LNCS, vol. 3383, pp. 100–110. Springer (2004)
- Dahlhaus, E.: A linear time algorithm to recognize clustered planar graphs and its parallelization. In: LATIN '98. LNCS, vol. 1380, pp. 239–248. Springer (1998)
- Di Battista, G., Didimo, W., Marcandalli, A.: Planarization of clustered graphs. In: GD '01. LNCS, vol. 2265, pp. 60–74. Springer (2001)
- Di Battista, G., Frati, F.: Efficient c-planarity testing for embedded flat clustered graphs with small faces. In: GD '07. LNCS, vol. 4875, pp. 291–302. Springer (2007)
- Di Battista, G., Garg, A., Liotta, G., Tamassia, R., Tassinari, E., Vargiu, F.: An experimental comparison of four graph drawing algorithms. *Comput. Geom. Theory Appl.* 7(5-6), 303–325 (1997)
- Feng, Q.W., Cohen, R.F., Eades, P.: Planarity for clustered graphs. In: ESA '95. LNCS, vol. 979, pp. 213–226. Springer (1995)
- Goodrich, M.T., Lueker, G.S., Sun, J.Z.: C-planarity of extrovert clustered graphs. In: GD '05. LNCS, vol. 3843, pp. 211–222. Springer (2005)
- Gutwenger, C., Jünger, M., Leipert, S., Mutzel, P., Percan, M., Weiskircher, R.: Advances in c-planarity testing of clustered graphs. In: GD '02. LNCS, vol. 2528, pp. 220–235. Springer (2002)
- Jelínková, E., Kára, J., Kratochvíl, J., Pergel, M., Suchý, O., Vyskocil, T.: Clustered planarity: Small clusters in eulerian graphs. In: GD '07. LNCS, vol. 4875, pp. 303–314. Springer (2007)
- Jünger, M., Mutzel, P.: Maximum planar subgraphs and nice embeddings: Practical layout tools. *Algorithmica* 16(1), 33–59 (1996)
- Jünger, M., Thienel, S.: The ABACUS system for branch-and-cut-and-price algorithms in integer programming and combinatorial optimization. *Software: Practice and Experience* 30, 1325–1352 (2000)
- Kuratowski, K.: Sur le problème des courbes gauches en topologie. *Fundamenta Mathematicae* 15, 271–283 (1930)