$\mathcal{X}$-graphs of $\mathcal{Y}$-graphs and their Representations*

Vladimir Batagelj[1], Franz J. Brandenburg[2], Walter Didimo[3], Giuseppe Liotta[3], and Maurizio Patrignani[4]

[1] University of Ljubljana, Department for Theoretical Computer Science
1000 Ljubljana, Jadranska 19, Slovenia
vladimir.batagelj@fmf.uni-lj.si
[2] Universität Passau, Fakultät für Mathematik und Informatik
94030 Passau, Innstraße 33, Germany
brandenb@informatik.uni-passau.de
[3] University of Perugia, Dipartimento di Ingegneria Elettronica e dell'Informazione
06125 Perugia, Via G. Duranti 93, Italy
{didimo,liotta}@diei.unipg.it
[4] Università Roma Tre, Dipartimento di Informatica e Automazione
00146 Roma, Via della Vasca Navale 79, Italy
patrigna@dia.uniroma3.it

Abstract. We address graph decomposition problems that help the hybrid visualization of large graphs, where different graphic metaphors (node-link, matrix, etc.) are used in the same picture. We generalize the $\mathcal{X}$-graphs of $\mathcal{Y}$-graphs model introduced in [1] to formalize the problem of automatically identifying dense subgraphs ($\mathcal{Y}$-graphs, clusters) that are prone to be collapsed and shown with a matricial representation when needed. We show that (planar, K_5)-recognition, that is, the problem of identifying K_5 subgraphs such that the graph obtained by collapsing them is planar, is NP-hard. On the positive side, we show that it is possible to determine the highest value of k such that G is a (planar,k-core)-graph in $O(m + n\log(n))$ time.

Keywords. graph drawing, $\mathcal{X}$-graphs of $\mathcal{Y}$-graphs, visualization of large graphs.

1 Introduction

Diverse research fields as bioinformatics and social sciences point out that the visualization and exploration of large graphs are key steps for the analysis of the large amount of data that is nowadays available in electronic form. Such a task, however, is still an elusive goal. In particular, it is difficult to accomplish with standard graph drawing techniques, since the usual node-link representation, where entities are mapped to nodes and relations are mapped to links between them, is often unable to convey both the high-level structure of the graph (context) and its atomic details (focus).

* Work on this problem began at Dagstuhl Seminar No. 08191 "Graph Drawing with Applications to Bioinformatics and Social Sciences", May 2008.

Dagstuhl Seminar Proceedings 08191
Graph Drawing with Applications to Bioinformatics and Social Sciences
http://drops.dagstuhl.de/opus/volltexte/2008/1556

A recent trend is that of recurring to hybrid representations, where part of the information is conveyed with the usual node-link metaphor and part of it is conveyed by some other means, such as adjacency matrices, which are more effective to represent very dense subgraphs [2]. This approach, although promising, urges for the automatic identification of those subgraphs that are prone to be represented in a succinct way, a task that would be otherwise left to the user.

In this paper we tackle from a rigorous graph theoretic perspective the problem of identifying subgraphs (clusters) that are responsible for the visual complexity of the drawing. We gather models and results from the literature and add new pieces to fill in the theoretic scenario. Given a graph $G = (V, E)$, we consider both a "strong" model where the whole vertex set V has to be partitioned into clusters [1], and a generalization of it, that we call "weak" model, where the vertex-disjoint clusters are not requested to be a partition of V.

In Section 2 we give basic definitions about $(\mathcal{X}, \mathcal{Y})$-graph decompositions and review known results. Section 3 is devoted to some special cases of the problem where graphs of bounded size are involved in the strong model decomposition. Section 4 contains a reduction from the Planar 3-Satisfiability problem [3] to prove that it is NP-hard deciding whether a graph becomes planar by collapsing some K_5 vertex-disjoint subgraphs. Section 5 describes how to find the maximum k such that there exists subgraphs of coreness k, which, when collapsed, yield a planar graph. Finally Section 6 contains our conclusions.

2 Background

Given two graph classes $\mathcal{X}$ and $\mathcal{Y}$, a graph $G = (V, E)$ is an *$\mathcal{X}$-graph of $\mathcal{Y}$-graphs* (or *$(\mathcal{X}, \mathcal{Y})$-graph*, for short) if a family $V_1, V_2, \ldots, V_h$ of disjoint subsets of V, called *clusters*, can be identified, such that:

1. every cluster induces a graph belonging to class $\mathcal{Y}$, and
2. the *reduced graph* G^* obtained from G by collapsing each cluster into a single vertex and replacing multiple edges with a single one is a graph of class $\mathcal{X}$.

If subset $V_1, V_2, \ldots V_h$ are requested to be a partition of V, that is, if we add the constraint that $V = V_1 \cup V_2 \cup \cdots \cup V_h$, then we call G a *strong* $(\mathcal{X}, \mathcal{Y})$*-graph*, otherwise we call G a *weak* $(\mathcal{X}, \mathcal{Y})$*-graph* or, simply, an $(\mathcal{X}, \mathcal{Y})$*-graph*. The strong model of $\mathcal{X}$-graph of $\mathcal{Y}$-graphs, also known as *two level clustered graphs* [4,5,6], was introduced in [1].

Both for the strong model and for the weak one, by considering different families for $\mathcal{X}$- and $\mathcal{Y}$-graphs one obtains different $(\mathcal{X}, \mathcal{Y})$-decomposition problems which have diverse importance with respect to applications or to the insight into graph-theoretic decomposition problems.

Only for the strong model it makes sense considering the case when $\mathcal{X}$-graphs are general graphs, that is, when they are not constrained, since with such a hypothesis any graph is an $(\mathcal{X}, \mathcal{Y})$-graph in the weak model. More generally, for their impact on applications it is worth exploring cases when $\mathcal{X}$-graphs are "low-density" graphs; planar graphs; connected graphs with bounded or specified

number of nodes; and directed acyclic graphs. Also, the cases when $\mathcal{X}$-graphs are trees, paths, cycles, and bounded size graphs are interesting from a more theoretic perspective.

Similarly, regarding the families of $\mathcal{Y}$-graphs, we signal as important for applications the cases when $\mathcal{Y}$-graphs are "high-density" graphs, as, for example, graphs with high clustering coefficient; k-cores; cliques; complete bipartite graphs; k-connected graphs; strongly connected digraphs; and stars. We recall that the clustering coefficient of a graph $G = (V, E)$ is defined as $\frac{2|E_i|}{|V_i|(|V_i|-1)}$, and that a graph has core k if all its vertices have degree at least k.

Again, from a more theoretic perspective, the cases when $\mathcal{Y}$-graphs are trees, paths, cycles, and bounded size graphs are interesting.

	$\mathcal{X}$-graphs			
$\mathcal{Y}$-graphs	tree	non-trivial path	single k-tree	single k-star
paths	NP-complete [5]	NP-complete [6]		
cycles	NP-complete [5]	NP-complete [6]		
bounded paths	Polynomial [5]			
bounded cycles	Polynomial [5]			

Table 1. Known results for paths and cycles $\mathcal{Y}$-graphs in the strong model.

Table 1 summarizes known results in the strong model when $\mathcal{Y}$-graphs are paths or cycles. In [5] it is shown that deciding if it is possible to collapse paths or cycles in such a way that the reduced graph is a tree is NP-complete. Conversely, if the length of the paths (cycles) is bounded by a constant the problem is polynomial [5]. In [6] it is proved that, for a given integer $k \geq 2$, it is NP-complete to decide whether or not a graph is a path of length $k-1$ of paths (cycles), and that it is NP-complete to decide whether or not a graph is a k-star or a k-clique of paths (cycles). In contrast, in [6] it is shown that k-graphs of paths (cycles) can be recognized in polynomial time when the inputs are restricted to graphs of bounded treewidth.

	$\mathcal{X}$-graphs			
$\mathcal{Y}$-graphs	general graph	planar graph	3-cycle	cycle
cliques		NP-hard [7,4]	NP-hard [1]	Polynomial if diameter > 3 [1]
3-cliques	NP-complete [8]			

Table 2. Known results for cliques $\mathcal{Y}$-graphs in the strong model.

Table 2 summarizes known results when $\mathcal{Y}$-graphs are cliques in the strong model. Recognizing planar graphs of cliques is NP-hard [7,4]. This problem stays NP-complete even if the planar graph is restricted to be a cycle of length three, although for cycles of length greater than 5 the problem is polynomial [1].

	$\mathcal{X}$-**graphs**	
$\mathcal{Y}$-**graphs**	tree	bounded size
cliques		NP-hard (Sec. 3)
bounded size	$O(n)$ (Sec. 3)	$O(1)$ (Sec. 3)

Table 3. Results discussed in this paper for the strong model.

Table 3 summarizes results discussed in this paper for the strong model. We show that while considering bounded size $\mathcal{Y}$-graph may yield to efficiently decidable $(\mathcal{X}, \mathcal{Y})$-graphs, for example when $\mathcal{Y}$ is the class of trees, on the other hand, having a bounded size $\mathcal{X}$-graph does not necessarily make the corresponding recognition problem easier (recognizing a bounded size graph of cliques is NP-hard).

Regarding the weak model introduced in this paper, we explored some cases when dense $\mathcal{Y}$-graph are involved (see Table 4). In particular, we observe that identifying cliques in large graphs may lead to a very effective strategy for information visualization. In fact, when the user is able to tell that a subset of nodes is a clique, its internal edges are understood and do not need to be explicitly displayed. Unfortunately, recognizing $(planar, K_5)$-graphs is NP-hard (Section 4). This result parallels the analogous result for the strong model [7,4].

The k-core of a graph $G(V, E)$ is the graph obtained by recursively removing vertices of degree less than k. Let n and m be the number of vertices and edges of G, respectively. In Section 5 we show that there exists an $O(m + n\log(n))$ algorithm to find the maximum k such that the reduced graph obtained by collapsing each connected component of the k-core of G is planar.

	$\mathcal{X}$-**graphs**
$\mathcal{Y}$-**graphs**	planar graph
5-cliques	NP-hard (Sec. 4)
k-core graphs (max k)	$O(m + n\log(n))$ (Sec. 5)

Table 4. Results discussed in this paper for the weak model.

3 Bounded-size Graphs

Both from a theoretical perspective and from a more practical one, it would be interesting to explore the case when the $\mathcal{X}$-graphs or the $\mathcal{Y}$-graphs involved have bounded size. In this section we show a positive and a negative result for the strong model. On the positive side, we show that it is easy to decide whether the vertices of a given graph can be partitioned into bounded-size clusters such that their collapse yields a tree.

Theorem 1. *There is an $O(n)$ algorithm to recognize whether or not the vertices of a graph can be partitioned into bounded-size clusters such that the reduced graph is a tree.*

Proof sketch: Let $G(V, E)$ be a graph such that a family of disjoint subsets $V_1, V_2, \ldots, V_h$ of V, with $|V_i| < k$, can be identified so that the reduced graph G^* is a tree. Observe that G has tree-width at most k. The proof of Theorem 2 in [5] can be adapted to this case. □

On the negative side, we show that by restricting to bounded size $\mathcal{X}$-graphs the recognition problem in the strong model can still be NP-hard for certain classes of $\mathcal{Y}$-graphs.

Theorem 2. *It is NP-hard to decide whether or not the vertices of a graph can be partitioned into cliques such that the reduced graph has bounded-size.*

Proof sketch: Consider problem Partition into cliques (see, for example, Problem GT15 of [8]). The instance of such a problem is a graph $G = (V, E)$ and a positive integer $K \leq |V|$. The question is whether the vertices of G can be partitioned into $k \leq K$ disjoints sets $V_1, V_2, \ldots, V_k$ such that, for $1 \leq i \leq k$, the subgraph induced by V_i is a complete graph. This problem, also called Clique cover, is proved to be NP-complete by a reduction of Graph k-colorability [8]. A reduction from Graph 3-colorability with the additional constraint that the graph is connected can be used to prove the statement. □

4 Planar Graphs of Cliques

Recognizing a planar graph of cliques is NP-hard in the strong model [7,4]. In this section we show that $(planar, K_5)$-recognition is NP-hard also in the weak model.

Theorem 3. *It is NP-hard to decide whether or not a graph can be made planar by collapsing vertex-disjoint K_5 subgraphs.*

To prove the theorem we use a reduction from the Planar 3-Satisfiability problem [3]:

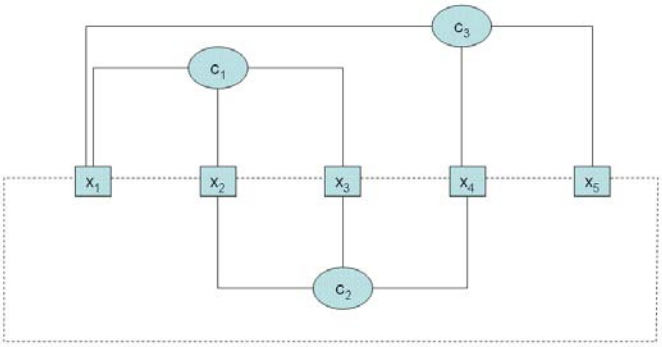

Fig. 1. An instance of PLANAR 3SAT is composed by a 3SAT formula with all positive literals, such as $(x_1 \vee x_2 \vee x_3) \wedge (x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee x_3 \vee x_5)$, and a corresponding planar graph, represented in the figure.

Problem: **Planar 3-Satisfiability (P3SAT)**
Instance: A set of clauses $c_1, \ldots, c_m$ each one having three literals from a set of Boolean variables $x_1, \ldots, x_n$. A plane bipartite graph $G(V_A, V_B, E)$ where nodes in V_A correspond to the variables while nodes in V_B correspond to the clauses (hence, $|V_A| = n$ and $|V_B| = m$). Edges connect clauses to the variables of the literals they contain. Moreover, $G(V_A, V_B, E)$ is drawn without intersections on a rectangular grid of polynomial size in such a way that nodes in V_A are arranged in a horizontal line that is not crossed by any edge (see Fig. 1).
Question: Can truth values be assigned to the variables $x_1, \ldots, x_n$ such that each clause has at least one true literal?

First, we first describe how to construct the instance of the (planar,K_5)-recognition problem starting from an instance of the P3SAT problem. Second, we show that the first problem admits a solution if and only if the second does.

Suppose I_{P3SAT} is an instance of the P3SAT problem with n Boolean variables and m clauses. The corresponding instance $I_{(planar,K_5)}$ is built by glueing together *variable gadgets* and *clause gadgets*, linked by *transmission gadgets* and *optional-switch gadgets*.

4.1 The Transmission Gadget

The *transmission gadget* is the basic tool of the whole construction and is also used to transmit a truth value from variable gadgets to clause gadgets. As such, it has the property of admitting two different states. Fig. 2.a shows a section of the transmission gadget, which is composed by a sequence of K_5's each sharing two vertices with the following K_5 and other two vertices with the preceding K_5.

The remaining forth vertex is used to form a path that links together all K_5's in even position (odd-position) of the sequence.

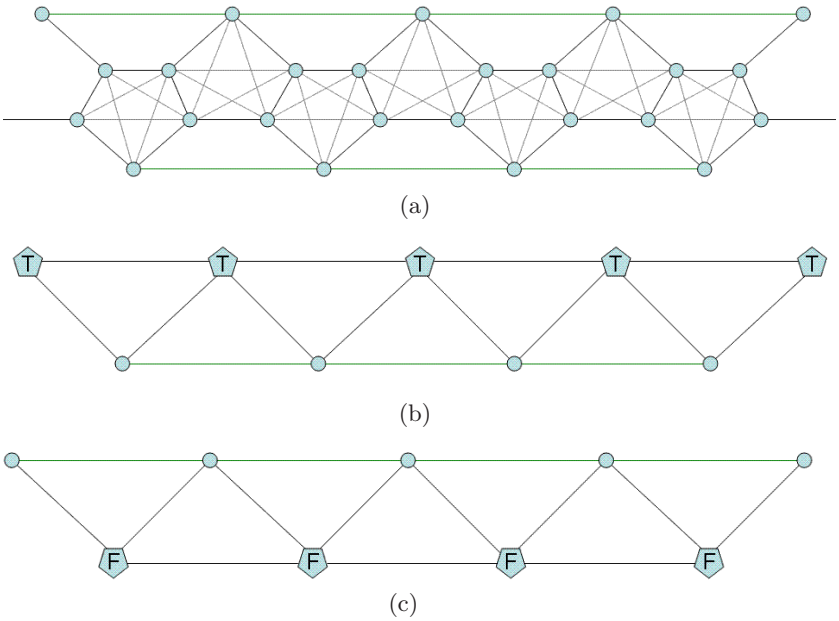

Fig. 2. The *transmission gadget* (a) transmitting a `true` value (b) and transmitting a `false` value (c).

Conventionally, we say that a transmission gadget *transmits a true value* (*transmits a false value*, respectively) when its K_5's in even position (odd position, respectively) are collapsed. Figs. 2.b and 2.c show the transmission gadget when transmitting a true value and a false value.

In order for the reduced graph to be planar, each K_5 must be either collapsed or adjacent to a collapsed K_5 in such a way that, after the adjacent K_5 is collapsed, the removal of multiple edges removes the K_5 itself from the graph. Fig. 3 shows that in order to obtain a planar reduced graph, either all K_5's in even position or all K_5's in odd position need to be collapsed (small pentagons in the figure represent collapsed K_5's). In other words, the property of transmitting a `true` or `false` value is a global property for the transmission gadget, that is, a transmitted `true` value can not be changed into a `false` value without introducing a non-planarity.

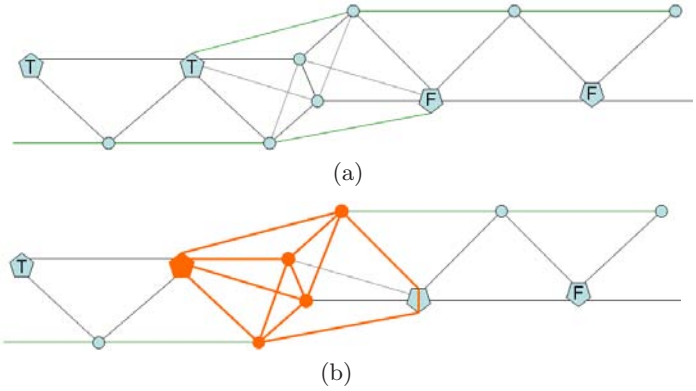

Fig. 3. The *transmission gadget* can not change the transmitted value from `true` into `false` or vice versa (a). In fact, in this case the reduced graph has a K_5 subdivision and hence is not planar (b).

4.2 The Variable Gadget

The *variable gadget* is built by closing a transmission gadget into a cycle (see Fig. 4). Since either K_5's in even position or K_5's in odd position can be collapsed to yield a planar reduced graph, the variable gadget admits two states, corresponding to the two truth values for the corresponding Boolean variable.

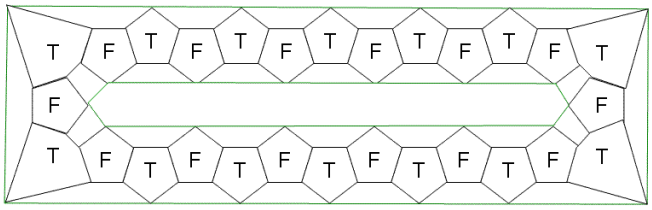

Fig. 4. The *variable gadget* is composed by a transmission gadget closed into a cycle. When T-labeled K_5 are collapsed its value is `true`. When F-labeled K_5 are collapsed its value is `false`.

4.3 The Junction Gadget

The *junction gadget* is used to attach to a variable gadget in order to extract its truth value and transmit it towards the clause gadgets. Fig. 5 shows the junction

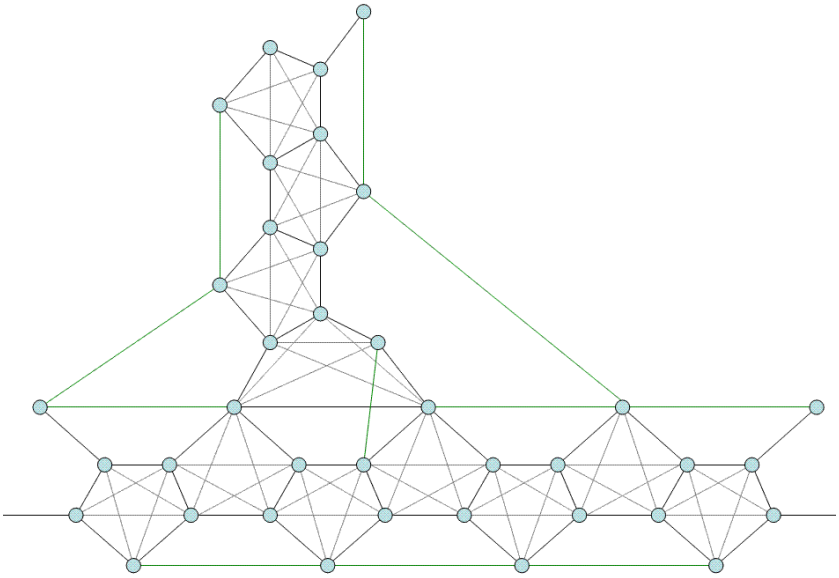

Fig. 5. The *junction gadget*.

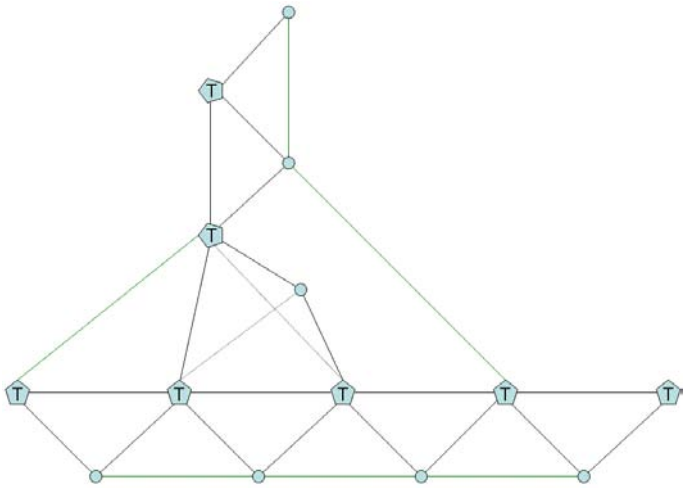

Fig. 6. The *junction gadget* transmitting a `true` value.

gadget, where the sequence of K_5's in the lower part of the figure corresponds to the portion of the variable gadget to which the junction gadget is attached, and

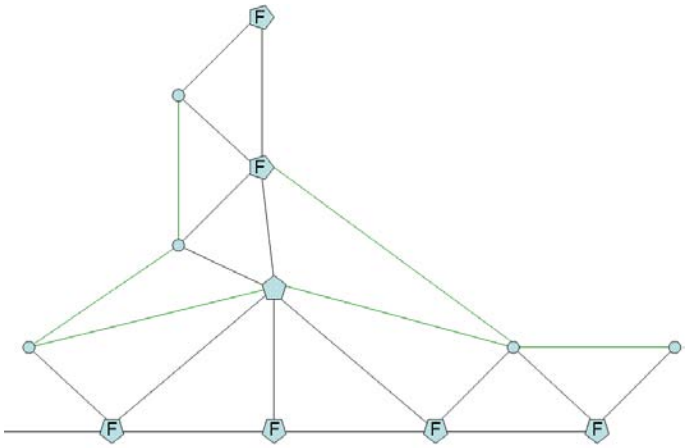

Fig. 7. The *junction gadget* transmitting a `false` value.

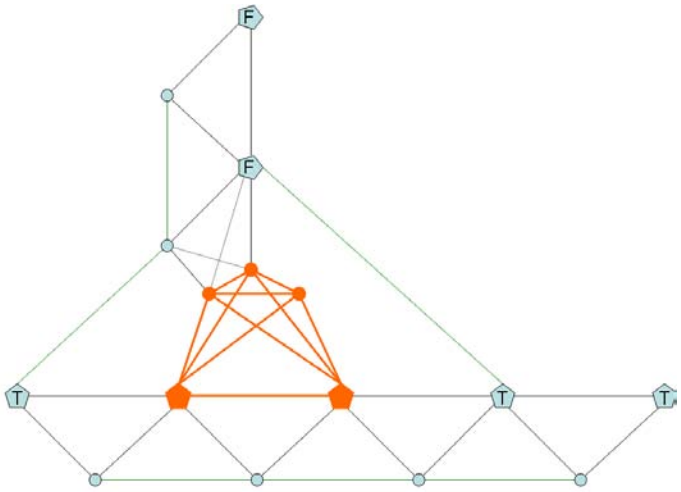

Fig. 8. The *junction gadget* changing the transmitted `true` value to a `false` value. A non-collapsible K_5 yields non-planarity.

the sequence of K_5's at the top of the figure is the starting point of a transmission gadget that heads towards the clause gadget.

Figs. 6 and 7 show that the reduced graph is planar if the truth value extracted from the variable gadget is coherent the the truth value of the transmission gadget heading towards the clause gadget. Conversely, Figs. 8 and 9 show that if such truth values are not coherent the reduced graph is not planar. In

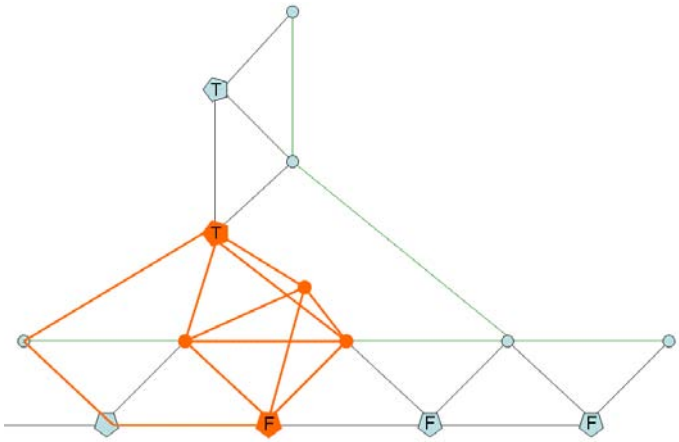

Fig. 9. The *junction gadget* changing the transmitted `false` value to a `true` value. A K_5 subdivision yields non-planarity.

fact, the K_5 drawn with thick lines in Figs. 8 can not be collapsed, since two of its vertices are, in their turn, collapsed K_5. Analogously, the K_5 subdivision of Fig. 9 can not be removed from the reduced graph, since subdivisions can not be collapsed.

4.4 The Optional-Switch Gadget

The *optional-switch gadget* is used to weaken the constraints on some transmission gadget. Namely, the role of optional-switch gadgets is that of allowing some transmission gadgets to switch from a `true` state to a `false` state, but not allowing the opposite, that is to switch from a `false` state to a `true` state. Details on where to place optional-switch gadgets are given in Section 4.5 when clause gadgets are described.

The optional-switch gadget, shown in Fig. 10.a, is obtained from the transmission gadget by removing some edges. While Fig. 10.b shows that it allows to change a transmitted `true` value into a `false` value, Figs. 11.a and 11.b show that a transmitted `false` value can not be changed into a `true` value without introducing a non-planarity in the reduced graph.

Of course, a similar optional-switch gadget can be built that allows a `false` value to be switched into a `true` value but not *vice versa*. In fact, it suffice flipping the optional-switch gadget of Fig. 10.a around the horizontal axis. We call this version of the gadget *negative optional-switch gadget*, and the basic version of the gadget *positive optional-switch gadget*.

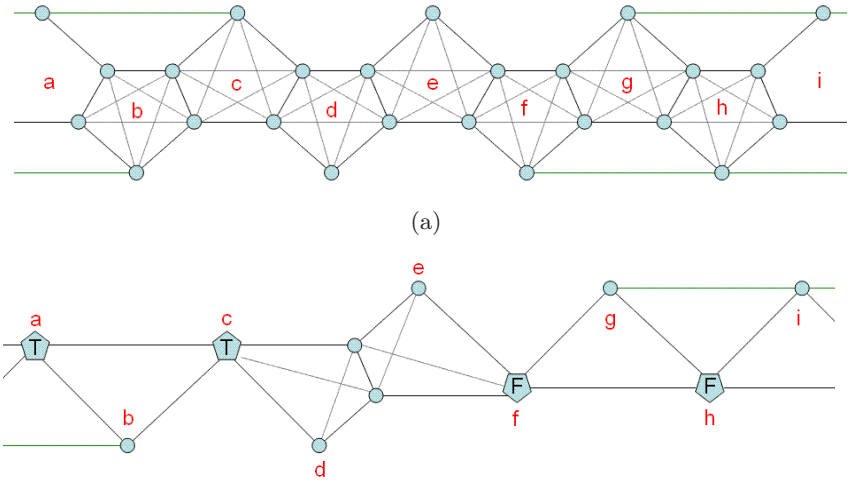

Fig. 10. The *optional switch gadget* (a) allows a transmitted `true` value to be changed into a `false` value (b). Observe that, although the drawing in (b) is not planar, the graph represented can be easily drawn without intersections.

4.5 The Clause Gadget

Consider a clause with three literals $(l_1 \wedge l_2 \wedge l_3)$ from three Boolean variables x_1, x_2, and x_3. By using junction gadgets, we attach three transmission gadgets to the variable gadgets corresponding to x_1, x_2, and x_3, respectively. If l_i, with $i \in \{1, 2, 3\}$, is a non-negated (negated, respectively) literal of x_i, we insert a positive optional-switch gadget (negative optional-switch gadget, respectively) into the transmission gadget before it reaches the clause gadget. Hence, we can assume that the clause gadget has at most a single `true` literal. If more than one transmission gadget corresponding to a `true` literal arrives a clause gadget, we can use the optional-switch gadgets to change the values of all but one `true` literals.

Fig. 12 show the construction for the clause gadget. The clause gadget consists of a K_9 which shares two vertices with the "last K_5" of each transmission gadget. Here, by "last K_5" of the transmission gadget of literal l_i, we mean a K_5 in odd position if l_i is the non-negated literal of x_i, and a K_5 in even position if l_i is the negated literal of x_i. Hence, if literal l_i is `true`, the last K_5 of its transmission gadget which attaches to the K_9 is not collapsed in any planar drawing of the reduced graph.

Fig. 13 shows that if all literals are `false`, that is, if all the K_5 attached to the K_9 are collapsed, then a non-collapsible K_6 is part of the reduced graph, which is therefore non-planar.

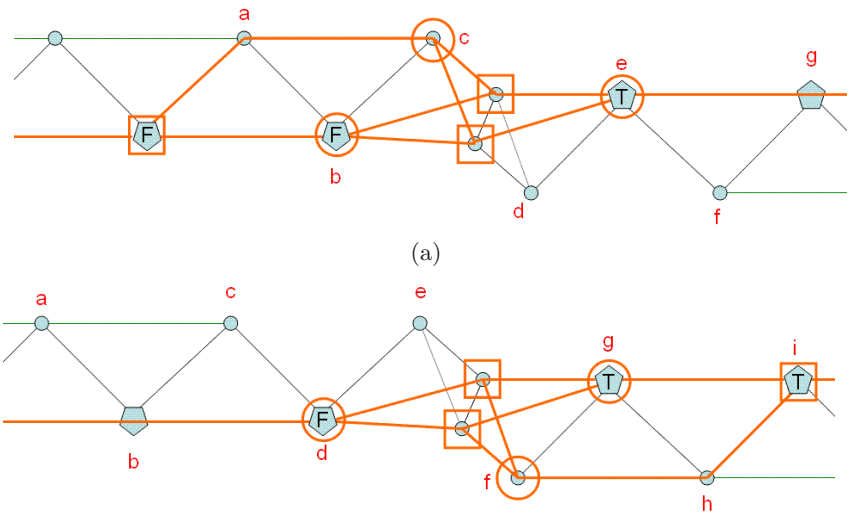

Fig. 11. The *optional switch gadget* does not allow a transmitted `false` value to be changed into a `true` value. When the change happens between the K_5 labeled b and the K_5 labeled e of Fig. 10(a), the $K_{3,3}$ subdivision identified by the vertices into squares and circles implies non-planarity (a). Also, when the change happens between the K_5 labeled d and that labeled g, a $K_{3,3}$ subdivision implies non-planarity of the reduced graph (b).

Conversely, Fig. 14 shows that if a single literal is `true`, that is, if the last K_5 of its transmission gadget is not collapsed, a K_5 can be collapsed to transform the clause gadget into a triangle yielding a planar reduced graph (see Fig. 15).

4.6 Proof of Theorem 3

By using the gadgets given in the previous sections, starting from an instance I_{P3SAT} of the P3SAT problem we build an instance $I_{(planar,K_5)}$ of (planar,K_5)-recognition. Based on such construction Theorem 3 can be proved by showing that the I_{P3SAT} instance admits a solution if and only if the $I_{(planar,K_5)}$ instance admits a solution and that the whole construction can be obtained in polynomial time.

Namely, given a truth assignment for the Boolean variables that satisfies instance I_{P3SAT}, we determine the state of each variable gadget, collapsing K_5's accordingly. We select a `true` literal for each clause, and collapse the last K_5's of the transmission gadgets corresponding to unselected literals of the same clause. Therefore, each clause gadget admits a final K_5 collapse that will transform it into a triangle. The truth values propagate from the variable gadgets towards

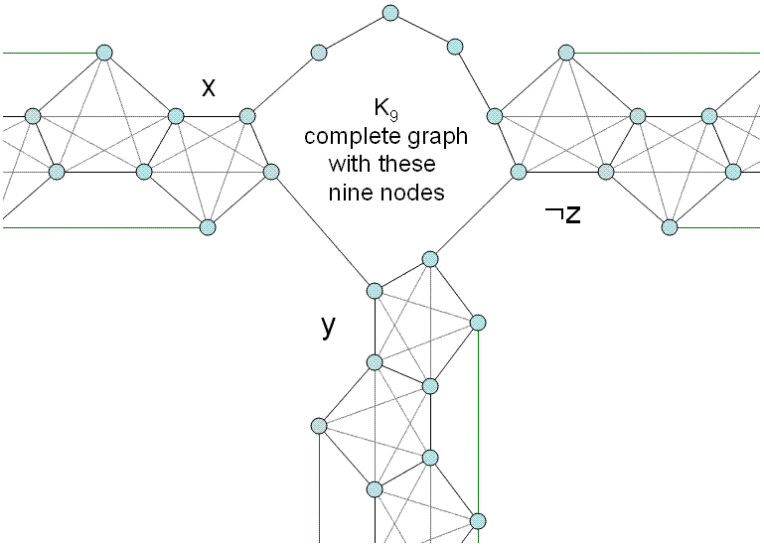

Fig. 12. The *clause gadget* for clause $(x \vee y \vee \neg z)$. Before arriving the clause gadget, the transmission gadgets for variables x and y have a positive optional-switch gadget inserted, and the transmission gadget for variable z has a negative optional-switch inserted. Therefore, if variables x and y are `true`, they may arrive the clause gadget with `true` or `false` values. If variable z is `false`, it may arrive the clause gadget with a `true` or a `false` value.

the clause gadgets possibly taking advantage of the optional-switch sections. The obtained reduced graph is planar.

Conversely, suppose to have decided that $I_{(planar,K_5)}$ is planar when certain K_5's are collapsed. Observe that each clause gadget is adjacent to a non-collapsed last K_5 corresponding to a `true` literal. Determine the corresponding truth assignment. The transmission gadgets (each one with its optional-switch gadget inserted) guarantees that if a literal is `true` for a clause, the corresponding variable gadget is coherently collapsed into a `true` variable gadget. Hence, the truth assignment determined by the `true` literals satisfies instance I_{P3SAT}.

5 Planar Graphs of *k*-Cores

A graph $G(V, E)$ is a (planar,k-core)-graph in the weak model if a family $V_1, V_2, \ldots, V_h$ of disjoint subsets of V can be identified, such that every cluster induces a graph with coreness k and the reduced graph obtained by collapsing the clusters is planar.

It is trivial to observe that any graph is a (planar,k-core)-graph for some k. In fact, any graph is a (planar,1-core)-graph. The following lemma holds.

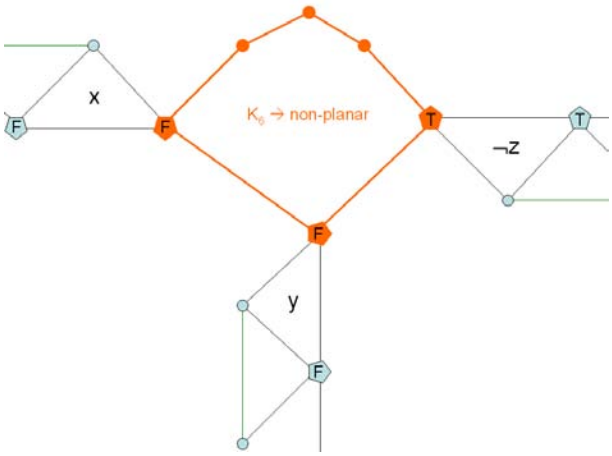

Fig. 13. The *clause gadget* when all literals are `false`, that is, when $x =$ `false`, $y =$ `false`, and $z =$ `true`. The K_6 yields non-planarity for the reduced graph.

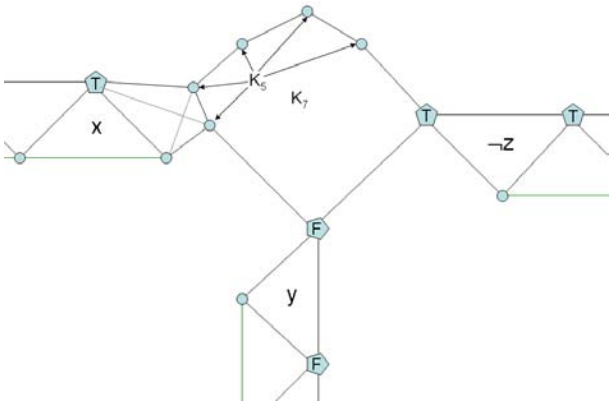

Fig. 14. The *clause gadget* when one literal is `true` ($x =$ `true`, $y =$ `false`, and $z =$ `true`). A further K_5 can be collapsed.

Lemma 1. *A (planar, k-core)-graph is also a (planar, (k-1)-core)-graph.*

Proof sketch: Let $G(V, E)$ be a (planar, k-core)-graph. Consider the graph G_k^* obtained by collapsing k-cores and removing multiple edges and the analogous graph G_{k-1}^* obtained by collapsing (k-1)-cores. Observe that G_{k-1}^* can be obtained from G_k^* by contracting some edges and removing self-loops. As edge-contraction does not increase the genus of a graph, the planarity of G_k^* implies the planarity of G_{k-1}^*. Hence, G is also a (planar, (k-1)-core)-graph. □

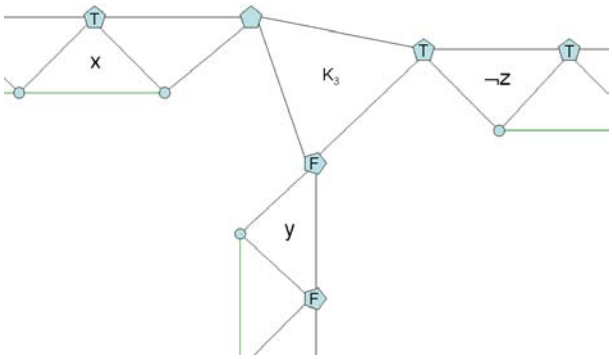

Fig. 15. The *clause gadget* when one literal is `true` ($x =$ `true`, $y =$ `false`, and $z =$ `true`). After contraction the resulting graph is planar.

Observe that higher values of k correspond to smaller clusters. Hence, it is interesting for applications searching for the maximum k such that $G(V, E)$ is a (planar,k-core)-graph, which corresponds to searching for the smaller clusters such that the reduced graph is planar.

Theorem 4. *Let $G(V, E)$ be any graph. It is possible to determine the highest value of k such that G is a (planar,k-core)-graph in $O(m + n\log(n))$ time, with $n = |V|$ and $m = |E|$.*

Proof sketch: Compute the core number of each vertex. This operation can be performed in $O(m)$ time. Rearrange the adjacency lists of all vertices based on decreasing core numbers. This operation can also be performed in $O(m)$ time by inserting all entries into a single array, sorting them in linear time, for example with a bucket sort, and reinserting them back into the adjacency lists. Perform a binary search for $k \in [1, n]$ combined with a planarity test on the reduced graph. The latter operation can be performed in $O(n\log(n))$ time. □

6 Conclusions

We addressed several decomposition problems in the $\mathcal{X}$-graph of $\mathcal{Y}$-graph model, where subgraphs of a specified type $\mathcal{Y}$ have to be identified with the purpose of collapsing them and yielding a reduced graph of type $\mathcal{X}$. We generalized the definition of $(\mathcal{X}, \mathcal{Y})$-graph given in [1] by considering both the case when clusters are a partition of the vertices of the input graph (strong model) and the case when clusters are only required to be disjoint subsets of the vertices (weak model).

We showed both positive and negative results.

On one hand, we showed that it is NP-hard to decide whether or not a graph is a bounded-size graph of cliques in the strong model, and that it is NP-hard to decide whether or not a graph is a planar graph of K_5 in the weak model.

On the other hand, we showed that there is a linear time algorithm to recognize whether or not a graph G is a tree of bounded-size graphs in the strong model and there is an $O(m + n\log(n))$ time algorithm to determine the highest value of k such that G is a (planar,k-core)-graph in the weak model. We think that the latter result is promising for applications that are based on hybrid visualization techniques.

References

1. Brandenburg, F.J.: Graph clustering I: Cycles of cliques. In Di Battista, G., ed.: Graph Drawing (Proc. GD '97). Volume 1353 of Lecture Notes Comput. Sci., Springer-Verlag (1997) 158–168
2. Henry, N., Fekete, J.D., McGuffin, M.J.: NodeTrix: A hybrid visualization of social networks. IEEE Trans. Visual. and Comp. Graphics **13**(6) (2007) 1302–1309
3. Lichtenstein, D.: Planar formulae and their uses. SIAM J. Comput. **11** (1982) 185–225
4. Kratochvìl, J., Goljan, M., Kučera, P.: String Graphs. Academia, Prague (1986)
5. Schreiber, F., Skodinis, K.: NP-completeness of some tree-clustering problems. In Whitesides, S.H., ed.: Graph Drawing (Proc. GD '98). Volume 1547 of Lecture Notes Comput. Sci., Springer-Verlag (1998) 288–301
6. Le, H.O., Le, V.B., Müller, H.: Splitting a graph into disjoint induced paths or cycles. Discr. Appl. Math. **131** (2003) 199–212
7. Kratochvìl, J.: String graphs II: Recognizing string graphs is NP-hard. J. of Combinatorial Theory, Series B **52** (1991) 67–78
8. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, New York, NY (1979)