On Stable Cutsets in Claw-Free Graphs and Planar Graphs

Van Bang Le[1], Raffaele Mosca[2], and Haiko Müller[3]

[1] Institut für Informatik, Universität Rostock, 18051 Rostock, Germany
le@informatik.uni-rostock.de
[2] Dipartimento di Scienze, Universitá degli Studi "G.D'Annunzio",
Viale Pindaro 42, Pescara 65127, Italy
r.mosca@unich.it
[3] School of Computing, University of Leeds, Leeds, LS2 9JT, UK
hm@comp.leeds.ac.uk

Abstract. To decide whether a line graph (hence a claw-free graph) of maximum degree five admits a stable cutset has been proven to be an **NP**-complete problem. The same result has been known for K_4-free graphs. Here we show how to decide this problem in polynomial time for (claw, K_4)-free graphs and for a claw-free graph of maximum degree at most four. As a by-product we prove that the stable cutset problem is polynomially solvable for claw-free planar graphs, and for planar line graphs. Now, the computational complexity of the stable cutset problem restricted to claw-free graphs and claw-free planar graphs is known for all bounds on the maximum degree.

Moreover, we prove that the stable cutset problem remains **NP**-complete for K_4-free planar graphs of maximum degree five.

1 Introduction

In a graph, a *clique* (*stable set*) is a set of pairwise (non-)adjacent vertices. A *cutset* (or *separator*) of a graph G is a set S of vertices such that $G - S$ is disconnected. A *clique cutset* (*stable cutset*) is a cutset which is also a clique (stable set).

Clique cutsets are a well-studied kind of separators in the literature, and have been used in divide-and-conquer algorithms for various graph problems, such as graph colouring and finding maximum stable sets; see [18,22]. Applications of clique cutsets in algorithm design use the fact that these cutsets (in arbitrary graphs) can be found in polynomial time [18,21,22].

The importance of stable cutsets has been demonstrated first in [6,20] in connection to perfect graphs. Tucker [20] proved that if S is a stable cutset in G and if no induced cycle of odd length at least five in G has a vertex in S then the colouring problem on G can be reduced to the same problem on the smaller subgraphs induced by S and the components of $G - S$.

Later, the papers [2,3,4,10,13,15] discussed the computational complexity of the problem Stable Cutset ("Does a given graph admit a stable cutset?").

D. Kratsch (Ed.): WG 2005, LNCS 3787, pp. 163–174, 2005.

Stable cutsets (in line graphs) have been also studied under other notion. A graph is *decomposable* (cf. [11]) if its vertices can be coloured red and blue in such a way that each colour appears on at least one vertex but each vertex v has at most one neighbour having a different colour from v. In other words, a graph is decomposable if its vertices can be partitioned into two nonempty parts such that the edges connecting vertices of different parts form an induced matching, a *matching-cut*. It turns out that matching-cuts in a graph correspond to stable cutsets in its line graphs. Matching-cuts have been studied in [1,5,8,9,15,16,17]. The papers [7,17] point out an application in graph drawing.

The relationship between decomposability and a stable cutset is (cf. [2]): If $L(G)$ has a stable cutset, then G is decomposable. If G is decomposable and has minimum degree at least two, then $L(G)$ has a stable cutset.

Chvátal [5] proved that recognising decomposable graphs is **NP**-complete, even for graphs with maximum degree four. Thus, in terms of stable cutsets in line graphs, Chvátal's result may be reformulated and improved as follows.

Theorem 1 (Chvátal [5]). Stable Cutset *is* ***NP****-complete, even if the input is restricted to line graphs with maximum degree six.*

Theorem 2 ([15]). Stable Cutset *remains* ***NP****-complete if restricted to line graphs with maximum degree five, and is polynomial solvable for line graphs of maximum degree at most four.*

Hence, the computational complexity of Stable Cutset for line graphs is completely characterised with respect to maximum degree constraints.

In particular, Stable Cutset is **NP**-complete for claw-free graphs with maximum degree five. In [15], it is shown that Stable Cutset is solvable in linear time for arbitrary graphs with maximum degree at most three. The complexity of Stable Cutset for graphs with maximum degree 4 is still open.

In this paper we will improve the second part of Theorem 2 to the larger class of claw-free graphs as follows: Stable Cutset becomes polynomial for claw-free graphs of maximum degree at most four. Thus the computational complexity of Stable Cutset for claw-free graphs is completely characterised with respect to maximum degree constraints.

Stable Cutset for K_3-free graphs is trivial. In [2] it was shown that Stable Cutset is **NP**-complete for K_4-free graphs. Our second result is that Stable Cutset can be solved in polynomial time for (claw, K_4)-free-graphs. As a by-product, we will show that Stable Cutset is polynomially solvable for claw-free planar graphs, and in particular for planar line graphs.

Finally, we show that Stable Cutset remains **NP**-complete on planar K_4-free graphs with maximum degree five.

2 Preliminaries

Let G be a graph. The vertex set and the edge set of G are denoted by $V(G)$ and $E(G)$, respectively. The neighbourhood of a vertex v in G, denoted by $N(v)$, is the set of all vertices in G adjacent to v. Let $\deg(v) = |N(v)|$ be the degree of the vertex v, and $\Delta(G) = \max\{\deg(v) \mid v \in V(G)\}$ the maximum degree of G. For a subset $W \subseteq V(G)$, $G[W]$ is the subgraph of G induced by W.

Let $\mathrm{scs}(G)$ denote the minimum size of a stable cutset of G. If G has no stable cutset we write $\mathrm{scs}(G) = \infty$.

When discussing the computational complexity of Stable Cutset we may assume that G is connected. Moreover we assume that no vertex v of G has a stable neighbourhood $N(v)$. Otherwise $N(v)$ or $\{v\}$ would be a stable cutset in G, or G has at most two vertices, and we are done. Thus, we have (cf. [15]):

Lemma 1. *If* $\mathrm{scs}(G) < \infty$, *then* $\mathrm{scs}(G - v) < \infty$ *for all* $v \in V(G)$.

Lemma 2. *Let C be a clique cutset in a graph G, $|C| \geqslant 2$. Then* $\mathrm{scs}(G) < \infty$ *if and only if there is a component $G[A]$ of $G - C$ such that* $\mathrm{scs}(G[A \cup C]) < \infty$.

Since a clique cutset can be found in polynomial time ([18,21]), and singletons are stable, Lemma 2 allows us to assume that G has no clique cutset.

3 Rigid Sets

A set $R \subseteq V$ is said to be *rigid* in $G = (V, E)$ if, for every stable set $S \subseteq V$, there is a connected component $G[A]$ of $G - S$ with $R \setminus S \subseteq A$. Rigid sets naturally come in because G has a stable cutset if and only if V is not rigid.

Clearly, every clique of G is rigid. Moreover, if Q and R are rigid sets such that $Q \cap R$ contains a pair of adjacent vertices, then $Q \cup R$ is rigid. However, further rigid sets exist, see Fig. 1 for examples.

By definition, a *chordal graph* has no induced cycle of length four or more.

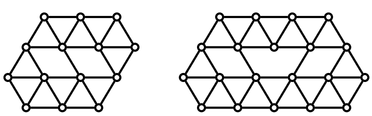

Fig. 1. Graphs without stable cutset

Lemma 3. *Let $H = (R, F)$ be a 2-connected chordal subgraph of $G = (V, E)$. Then R is rigid in G.*

Proof. The base step of the inductive proof is for complete H. In the inductive step we consider a minimal separator of H and use that it is a clique in G. □

4 Claw-Free Graphs of Maximum Degree Four

We are going to improve the second part of Theorem 2. We will show that Stable Cutset is polynomial solvable for claw-free graphs with maximum degree four by reducing the problem to line graphs.

Recall that the *line graph* $L(G)$ of a graph G has the edges of G as its vertices, and two distinct edges of G are adjacent in $L(G)$ if they are incident in G. Line graphs have been characterised in terms for forbidden induced subgraphs as follows: A graph is a line graph if and only if it does not contain any of the nine graphs listed in Fig. 2 as an induced subgraph (cf. [12]).

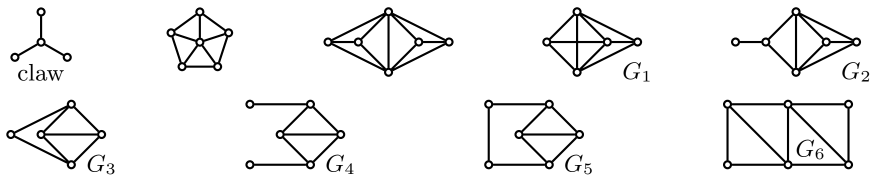

Fig. 2. Forbidden induced subgraphs for line graphs

Lemma 4. *Let G be a claw-free graph without clique cutset and $\Delta(G) = 4$.*

(i) *If G contains an induced G_1, then $G = G_1$ or* $\mathrm{scs}(G) \leqslant 2$.
(ii) *If G contains an induced G_2, then $|V(G)| \leqslant 8$ or* $\mathrm{scs}(G) \leqslant 3$.
(iii) *If G contains an induced G_3, then $|V(G)| \leqslant 8$ or* $\mathrm{scs}(G) \leqslant 3$.
(iv) *If* $\mathrm{scs}(G) < \infty$ *and G contains an induced G_4 or G_5, then* $\mathrm{scs}(G) \leqslant 3$.
(v) *If* $\mathrm{scs}(G) < \infty$ *and G contains an induced G_6 then* $\mathrm{scs}(G) \leqslant 2$.

Theorem 3. *Let G be a claw-free graph with $\Delta(G) = 4$ and without clique cutset. Assume that G is not a line graph and has at least 9 vertices. Then* $\mathrm{scs}(G) < \infty$ *if and only if* $\mathrm{scs}(G) \leqslant 3$.

Proof. As G is not a line graph, G must contain one of the nine forbidden induced subgraphs listed in Fig. 2. As G is claw-free and has maximum degree four, G therefore must contain one of the graphs $G_1, \ldots, G_6$ in Fig. 2 as an induced subgraph. Now the Theorem follows from Lemma 4. □

Theorem 4. Stable Cutset *can be solved in polynomial time for claw-free graphs with maximum degree at most four.*

Thus the computational complexity of Stable Cutset for claw-free graphs is completely characterised with respect to maximum degree constraints.

5 (Claw, K_4)-Free Graphs

This section shows that Stable Cutset can be solved efficiently for (claw, K_4)-free graphs by reducing the problem to claw-free graphs with maximum degree at most four. We observe first:

Lemma 5. *The maximum degree in a (claw, K_4)-free graph is at most five.*

Proof. Let v be a vertex of degree at least six in any graph G. By a Ramsey-argument, $G[N(v)]$ contains either a triangle or the complement thereof. That is, G contains a K_4 including v, or there is a claw with central vertex v. □

Let G be a (claw, K_4)-free graph on at least 11 vertices that contains neither clique cutsets nor vertices with stable neighbourhood. We will show that, for all vertices v of G with $\deg(v) = 5$, G has a stable cutset if and only if $G - v$ has a stable cutset. By Theorem 4, Stable Cutset is then solvable in polynomial time for (claw, K_4)-free graphs.

Let v be a vertex of degree five in G. By Lemma 1 it remains to show that if $G - v$ has a stable cutset then G has a stable cutset.

Assume to the contrary that G has no stable cutset, and consider an inclusion-minimal stable cutset S in $G - v$. By the minimality of S, every vertex in S has at least one neighbour in each connected component of $(G - v) - S$. Hence $(G - v) - S$ has exactly two connected components, otherwise there would be a claw in G. Moreover, $1 \leqslant |N(v) \cap S| \leqslant 2$, otherwise $S \cup \{v\}$ would be a stable cutset in G (if $N(v) \cap S = \varnothing$) or there would be a claw in G (if $|N(v) \cap S| \geqslant 3$).

Let A and B induce connected components of $(G - v) - S$. Then for all $u \in S \cup \{v\}, N(u) \cap A$ and $N(u) \cap B$ are cliques, each containing one or two vertices.

If $|N(v) \cap A| = 2 = |N(v) \cap B|$ then, as G is K_4-free, no vertex in $N(v) \cap S$ is adjacent to a vertex in $N(v) \cap A$ or $N(v) \cap B$. But then G admits a claw, a contradiction. Thus, $|N(v) \cap A| = 1$ or $|N(v) \cap B| = 1$, hence $|N(v) \cap S| = 2$. Let, without loss of generality, $N(v) \cap A = \{a_1, a_2\}$, $N(v) \cap B = \{b\}$, and $N(v) \cap S = \{s_1, s_2\}$.

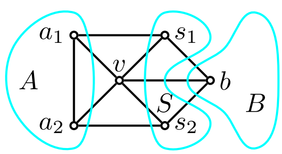

Fig. 3. Minimal stable cutset S in $G - v$ and the neighbourhood of v in $G - v$

Recall that a_1 and a_2 are adjacent. As G is K_4-free, we may assume $s_1 a_2 \notin E(G)$. Then s_2 and a_2 are adjacent (otherwise, v, s_1, s_2, and a_2 would form a claw) and hence s_2 and a_1 are nonadjacent, implying $s_1 a_1 \in E(G)$ (otherwise, v, s_1, s_2, and a_1 would form a claw). Finally, s_1 and s_2 both must be adjacent to b (otherwise there would be a claw), see also Fig. 3.

We complete the proof by case analysis according to the number of neighbours of s_i in A and B.

Theorem 5. Stable Cutset *is polynomial on* (*claw,* K_4)*-free graphs.*

6 Claw-Free Planar Graphs

In [4], it was shown that every graph with n vertices and $2n - 4$ edges contains a stable cutset (and, by the proof given there, such one can be found in polynomial time). Consequently one might ask the computational complexity of Stable Cutset in graphs with few edges. A natural candidate in this direction is the class of planar graphs. In this section we show that Stable Cutset can be solved efficiently for claw-free planar graphs.

It is well-known that planar graphs do not contain a K_5-minor.

Lemma 6. *Let G be a graph without clique cutset. If G contains no K_5-minor, then $G = K_4$ or G is K_4-free.*

Proof. We show that G cannot properly contain a K_4. Assume the contrary and consider four pairwise adjacent vertices a, b, c, and d in G. Then $H :=$

$G - \{a, b, c, d\}$ is non-empty and connected (otherwise, $\{a, b, c, d\}$ would be a clique cutset in G). Moreover, for each vertex $v \in \{a, b, c, d\}$, $N(v) \cap H \neq \varnothing$, otherwise $\{a, b, c, d\} \setminus \{v\}$ would be a clique cutset in G separating v and H. Thus, $\{a\}, \{b\}, \{c\}, \{d\}$, and H form a K_5-minor, a contradiction. □

Theorem 6. Stable Cutset *is polynomial on claw-free planar graphs.*

Proof. Theorem 6 directly follows from Lemma 6 and Theorem 5 since we may assume that our graphs do not contain any clique cutset. □

Corollary 1. Stable Cutset *becomes polynomial on planar line graphs.*

7 Planar Graphs of Degree at Most Five

In this section we prove that Stable Cutset remains **NP**-complete when restricted to partial subgraphs of the triangular grid without vertices of degree six. Since these graphs are K_4-free, this substantially improves the **NP**-completeness result in [2]. We use a reduction from a restricted version of planar 3SAT [14].

Let $\varphi = \bigwedge_{j=1}^{m} c_j$ be the conjunction of clauses. Each clause is the disjunction of literals. The literals are boolean variables or their negations. By X and C we denote the set of variables and clauses. For $x \in X$ and $c \in C$, $x \in c$ means that x or its negation $\overline{x}$ is a literal in c. We may assume the following restrictions:

- each variable appears (as x or $\overline{x}$) in at least three and at most four clauses,
- each clause consists of exactly three literals, and
- the graph $G = (V, E)$ is planar, where $V = X \cup C$ and $E = \{xc : x \in c\}$.

Note that these conditions ensure $|X| \leqslant |C| \leqslant \frac{4}{3}|X|$, i.e. $|V|$ is linear in $|X|$.

7.1 Construction

Let G' be a partial subgraph of a square grid such that each edge of G corresponds with a path in G', and the vertices having degree three or four in G' are in one-to-one correspondence with the vertices of G. Such an embedding G' of G into an $n \times n$-grid, $n = \mathcal{O}(|X|)$, can be constructed in quadratic time [19]. For each $e \in E$ let $\ell(e)$ be the number of horizontal edges on the path representing e in G'. We compute a ℓ-minimum spanning tree $T = (V, F)$ of G. Then each edge in $E \setminus F$ is represented by a path containing a horizontal edge because we cannot make a cycle of vertical edges only.

Starting from the embedding G', we construct a reduction graph as follows:

- each vertex in X is replaced by a truth assignment component,
- each vertex in C is replaced by a satisfaction test component, and
- each path corresponding with an edge in E is replaced by a channel.

Channels consist of three strips. The outer ones are banks and appear as double lines in the subsequent figures. The inner strip is the water, depicted in

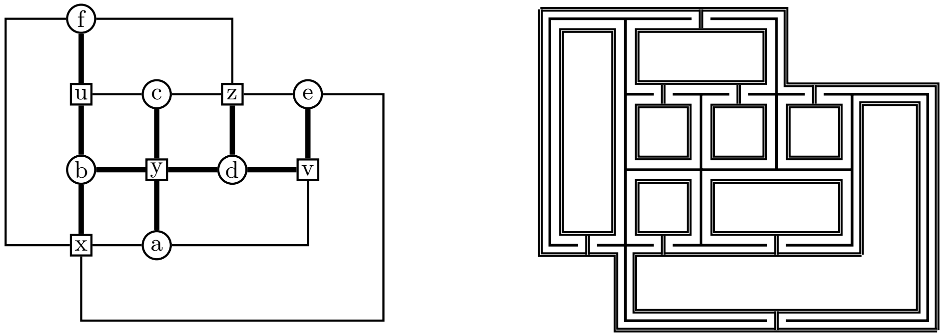

Fig. 4. Planar embedding and channel map

bold. Unlike edges in F, those in $E \setminus F$ contain a bridge in a horizontal part. The bridge interrupts the water and connects the two banks.

The water component is still connected because T is connected. Similarly, the bank component becomes connected via the bridges because all the water is surrounded by banks.

For example, let $X = \{u, v, x, y, z\}$ and $C = \{a, b, c, d, e, f\}$, where $\varphi = a \wedge b \wedge c \wedge d \wedge e \wedge f$ and

$$a = v \vee x \vee y \qquad b = u \vee \overline{x} \vee y \qquad c = u \vee \overline{y} \vee z$$
$$d = v \vee \overline{y} \vee \overline{z} \qquad e = \overline{v} \vee \overline{x} \vee \overline{z} \qquad f = \overline{u} \vee x \vee z$$

An grid embedding G' of the graph G corresponding with φ is shown on the left hand side of Fig. 4. A spanning tree T is indicated by bold edges. The right-hand side of this figure maps the channels and shows the bridges.

Now we are ready to describe building blocks in more detail. All the vertices are either bold (water) or double (bank), except four black vertices in the satisfaction test component. Edges are double (if both endpoints are double), bold (if both endpoints are bold), dotted (a double and a bold endpoint) for the reed between bank and water, and black (if one endpoint is black). A *monochrome component* is a maximal connected set of vertices of the same style (double or bold). All building blocks have the following properties:

- they are partial subgraphs of the triangular grid,
- they do not contain vertices of degree six (or more), and
- all monochrome components are rigid.

In the entire reduction graph, all double vertices (bank) will form one monochrome component, and all bold vertices (water) will form another one. If this graph has a stable cutset at all, then it separates bank from water. That is, each stable cutset will contain exactly one endpoint from each dotted edge.

Horizontal Channel. The horizontal channel is depicted in Fig. 5.

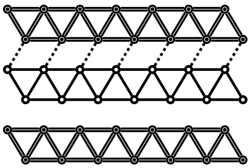

Fig. 5. Horizontal channel

Note that exactly two different stable cutsets exist which separate the upper monochrome component (bank) from the middle one (water). These cutsets are disjoint. That is, one endpoint of a dotted edge fixes the entire stable cutset. This way the truth values are propagated through the horizontal channel.

Vertical Channel. The vertical channel is depicted in Fig. 6.

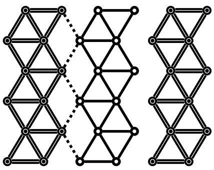

Fig. 6. Vertical channel

As in the horizontal channel, exactly two different stable cutsets exist which separate the left monochrome component (bank) from the middle one (water). Again, these cutsets are disjoint, and one endpoint of a dotted edge fixes the entire stable cutset. The truth values are propagated through the vertical channel in a similar way.

Bends. Two mini-bends are depicted in Fig. 7. At the hart of each bend in the channel we have one of them, or a reflection thereof.

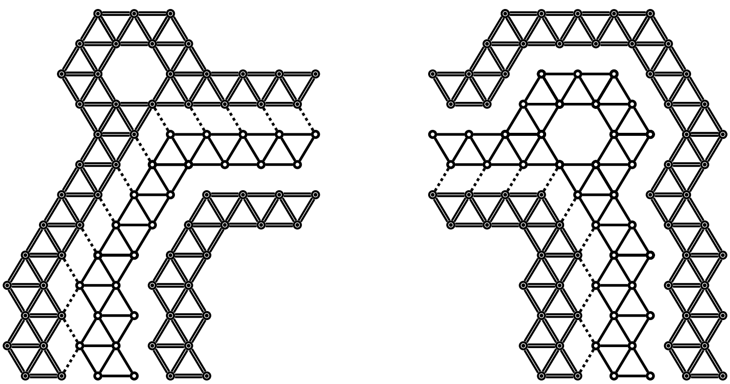

Fig. 7. Mini-bends

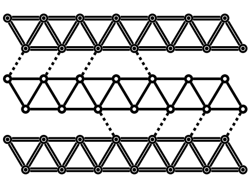

Fig. 8. Across

While the vertical part of a mini-bend always fits to a vertical channel, this is not the case for the horizontal part. The gadgets depicted in Fig. 8 and 9 their reflections will rectify. Note that all these building blocks propagate the truth values as the straight channels do.

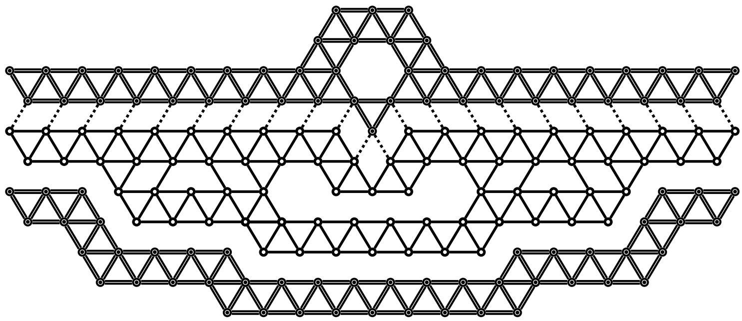

Fig. 9. Change of tilt

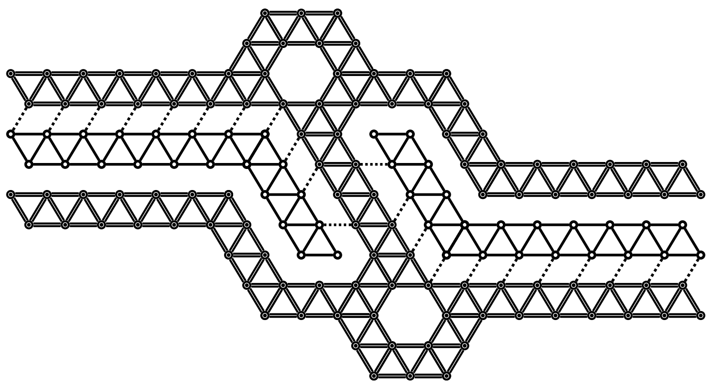

Fig. 10. Channel with bridge

Channel with Bridge. The bridge is depicted in Fig. 10.

The essential part in the centre resembles the idea of Fig. 8 with interchanged styles. The rest keeps the monochrome components rigid.

Truth Assignment Component. We give a mini-version with four horizontal outlets in Fig. 11. For a variable appearing in only three clauses cap one outlet.

The central part is known from Fig. 8, serving four outlets rather than two. The remaining parts are struts to keep the monochrome parts rigid.

Satisfaction Test Component. A mini-version of this component is given in Fig. 12. It has three inlets, on the top right, on the left, and bottom right. Let x, y and z be the literals whose truth values are fed in at these positions.

On the left we first split the y-channel into two, as in Figure 11. What follows is a strut to keep the water component rigid. The interesting part follows further to the right. The two black houses really test whether the clause is satisfied.

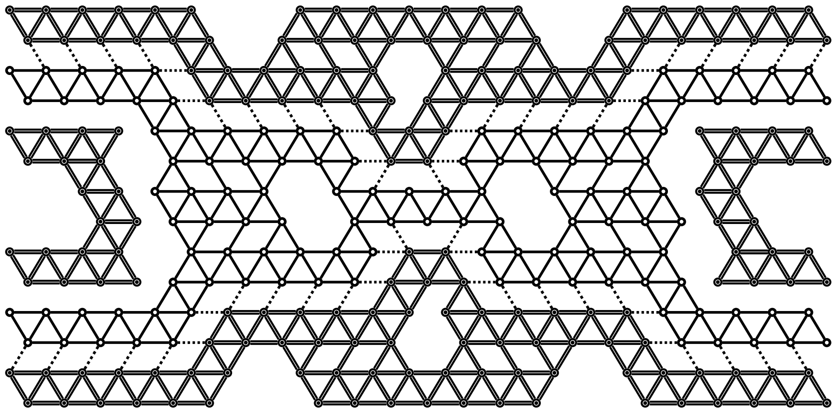

Fig. 11. Truth assignment component

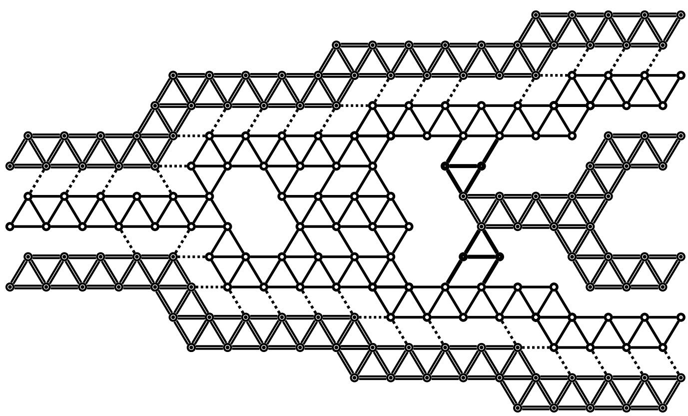

Fig. 12. Satisfaction test component

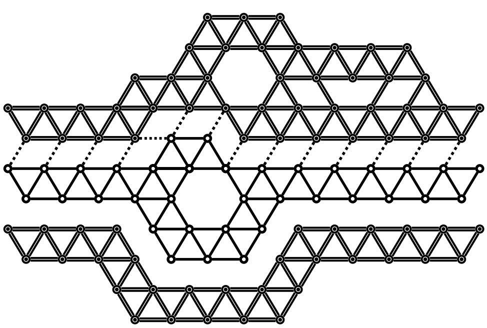

Fig. 13. Negator

The upper house tests $x \vee y$, the lower one $y \vee z$. Both houses together test $(x \vee y) \vee (y \vee z)$.

Each inlet of the satisfaction test component is directly connected to an outlet of a truth assignment component if the corresponding variable x is a positive literal in the clause, i.e. x appears unnegated as x. Otherwise ($\overline{x}$ appears as negative literal in the clause) we include the negator from Fig. 13 into the channel.

7.2 Equivalence

Let $a : X \to \{0, 1\}$ be a truth assignment of the variables in φ such that $a(\varphi) = 1$. We describe a stable cutset in the reduction graph.

The truth assignment component with caps at all four outlets allows exactly two stable cutsets, which are disjoint. These correspond with the truth values 0 (false) and 1 (true). For each variable $x \in X$ we choose the stable set in the truth assignment component that is given by $\varphi(x)$. These stable sets are extended along the channels into the satisfaction test components.

Because $a(\varphi) = 1$, for each clause there is at least one true literal. If literal x is true (upper right inlet), we choose two nonadjacent vertices in the four-cycle of the upper black house, and the bank vertex in the lower house. Whatever the truth value of the literals y (left inlet) and z (lower right inlet) is, this set of vertices extends to a stable cutset in the satisfaction test component. If z is true we swap the roles of upper and lower house. Finally, if y is true we can choose nonadjacent vertices in the four-cycles of both houses because the stable cutset enforced by the left inlet contains vertices both in the lower and upper branch of the component. Since this works in every satisfaction test component, we constructed a stable cutset of the reduction graph.

Now assume a stable cutset S of the reduction graph R is given. Then there is a bank component of $R - S$ containing all double vertices not in S, and a water component of $R - S$ containing all remaining bold vertices. We claim that S, restricted to the truth assignment components, defines a satisfying truth assignment $a : X \to \{0, 1\}$ for φ. Because the channels propagate the truth values between the truth assignment components and satisfaction test component, it remains to be shown that for each clause there is a true literal.

Each satisfaction test component contains two adjacent bank vertices incident with black edges. Clearly at most one of them belongs to S. This vertex separates its black house from the bank component. The other black house belongs to the bank component, and is separated from the water component by two nonadjacent vertices in its four-cycle. One of these vertices is bold. It marks a true literal in clause corresponding with this satisfaction test component.

8 Conclusion

While it has been shown that deciding whether or not a claw-free graph with maximum degree five [15], or a graph without 4-clique [2] contains a stable cutset is an **NP**-complete problem, we have proved in this paper that it can be decided in polynomial time whether or not

- a claw-free graph with maximum degree at most four,
- a claw-free graph without 4-clique, or
- a claw-free planar graph

contains a stable cutset.

In contrast, it is **NP**-complete to decide whether or not a planar graph with maximum degree five contains a stable cutset. The computational complexity of the stable cutset problem still remains open for graphs with maximum degree four, and even for planar graphs with maximum degree at most four.

References

1. P. Bonsma, *The complexity of the matching-cut problem for planar graphs and other graph classes*, Proc. WG 2003, LNCS 2880 (2003) 93–105.
2. A. Brandstädt, F. Dragan, V.B. Le, T. Szymczak, *On stable cutsets in graphs*, Discr. Appl. Math. 105 (2000) 39–50.
3. G. Chen, R.J. Faudree, M.S. Jacobson, *Fragile graphs with small independent cuts*, J. Graph Theory 41 (2002) 327–341.
4. G. Chen, X. Yu, *A note on fragile graphs*, Discrete Math. 249 (2002) 41–43.
5. V. Chvátal, *Recognizing decomposable Graphs*, J. Graph Theory 8, (1984) 51–53.
6. D.G. Corneil, J. Fonlupt, *Stable set bonding in perfect graphs and parity graphs*, J. Combin. Theory (B) 59 (1993) 1–14.
7. G. di Battista, M. Patrignani, F. Vargiu, *A Split&Push approach to 3D orthogonal drawing*, J. Graph Algorithms Appl. 1 (2000) 105–133.
8. A.M. Farley, A. Proskurowski, *Networks immune to isolated line failures*, Networks 12 (1982) 393–403.
9. A.M. Farley, A. Proskurowski, *Extremal graphs with no disconnecting matching*, Congressus Nummerantium 41 (1984) 153–165.
10. T. Feder, P. Hell, S. Klein, R. Motwani, *List partitions*, SIAM J. Discrete Math. 16 (2003) 449–478.
11. R.L. Graham, *On primitive graphs and optimal vertex assigments*, Ann. N.Y. Acad. Sci. 175 (1970) 170–186.
12. R.L. Hemminger, L.W. Beineke, *Line graphs and line digraphs*, In: *Selected Topics in Graph Theory I*, L.W. Beineke, R.T. Wilson, eds., Academic Press, London, (1978) 271–305.
13. S. Klein, C.M.H. de Figueiredo, *The NP-completeness of multi-partite cutset testing*, Congressus Numerantium 119 (1996) 217–222.
14. D. Lichtenstein, *Planar formulae and their uses*, SIAM Journal on Computing 11 (1982) 320–343.
15. V.B. Le, B. Randerath, *On stable cutsets in line graphs*, Theor. Comput. Sci. 301 (2003) 463–475.
16. A.M. Moshi, *Matching cutsets in graphs*, J. Graph Theory 13, (1989) 527–536.
17. M. Patrignani, M. Pizzonia, *The complexity of the matching-cut problem*, Proc. WG 2001, LNCS 2204 (2001) 284–295.
18. R.E. Tarjan, *Decomposition by clique separators*, Discr. Math. 55 (1985) 221–232.
19. I. Tollis, G. di Battista, P. Eades, R. Tamassia, *Graph drawing. Algorithms for the visualization of graphs*, Prentice Hall, Upper Saddle River, NJ, 1999.
20. A. Tucker, *Coloring graphs with stable cutsets*, J. Combin. Theory (B) 34 (1983) 258–267.
21. S.H. Whitesides, *An algorithm for finding clique cut-sets*, Inf. Process. Lett. 12 (1981) 31–32.
22. S.H. Whitesides, *An method for solving certain graph recognition and optimization problems, with applications to perfect graphs*, Ann. Discr. Math. 21 (1984) 281–297.