ELSEVIER

Discrete Mathematics 230 (2001) 215–252

DISCRETE MATHEMATICS

www.elsevier.com/locate/disc

Forbidden minors to graphs with small feedback sets

Michael J. Dinneen[a,*], Kevin Cattell[b], Michael R. Fellows[b]

[a]*Department of Computer Science, University of Auckland, Private Bag 92019, Auckland, New Zealand*
[b]*Department of Computer Science, University of Victoria, PO Box 3055, Victoria, BC, Canada V8W 3P6*

Received 27 September 1996; revised 10 December 1998; accepted 14 October 1999

Abstract

Finite obstruction set characterizations for lower ideals in the minor order are guaranteed to exist by the *graph minor theorem*. In this paper we characterize several families of graphs with small feedback sets, namely k_1-Feedback Vertex Set, k_2-Feedback Edge Set and (k_1,k_2)-Feedback Vertex/Edge Set, for small integer parameters k_1 and k_2. Our constructive methods can compute obstruction sets for any minor-closed family of graphs, provided the pathwidth (or treewidth) of the largest obstruction is known. © 2001 Elsevier Science B.V. All rights reserved.

1. Introduction

One of the most famous results in graph theory is the characterization of planar graphs due to Kuratowski: a graph is planar if and only if it does not contain either of $K_{3,3}$ or K_5 as a minor. The *obstruction set* (set of forbidden minors) for planarity thus consists of these two graphs.

The celebrated Graph Minor Theorem (GMT) of Robertson and Seymour [33] has the consequence of establishing non-constructively that many natural graph properties have Kuratowski-type characterizations; that is, they can be characterized by finite obstruction sets in an appropriate partial order. By ‘non-constructively’ we mean that the GMT may allow one to conclude that a finite obstruction set exists for a particular graph property, without providing any information on how it might be computed.

Fellows and Langston [16] showed that, in principle, there is a systematic method of computing the obstructions sets for most natural properties (although there are a few examples in topological graph theory for which their methods are not known to apply). The main elements of their approach include: (1) the use of bounded pathwidth or treewidth search spaces, (2) the use of finite-index congruences on boundaried graphs based on test sets, and (3) a partial order that allows for the termination of the search to

[*] Corresponding author.
E-mail address: mjd@cs.auckland.ac.nz (M.J. Dinneen).

PII: S0012-365X(00)00083-2

be established. In this paper we build upon and develop these earlier results. The central question addressed by our paper is: how can this general approach be implemented in a practical manner?

On the basis of an extensive implementation of a general obstruction set theorem-prover (consisting of over several thousand lines of C^{++} code), based on the general theory described in this paper, we report on the successful computation of the obstruction sets for some simple but non-trivial properties having obstruction sets of feasible size. The particular obstruction sets that we identify are of less interest in themselves than as a case study in the details of this kind of computation.

There are several general points of reference that form the context of our work:

(1) *The practicality of bounded treewidth and pathwidth algorithmics is an important unresolved question.*

Although there have been numerous papers concerning linear-time algorithms for various graph properties (including those that we address in this paper) based on bounded treewidth and pathwidth techniques, there has been a notable lack of practical implementation of these approaches (see, for example, [1,4,6,12]). The reason for this is that the linear-time complexity results hide large constants from several different sources in the asymptotic analysis. In order to successfully compute obstruction sets, we must deal with *some* of the same kinds of practical problems that arise in implementing general bounded treewidth- and pathwidth-based algorithms. Indeed, significant parts of our implementation could be used in the implementation of 'direct' linear-time recognition algorithms for the properties considered. Our results represent one of the few successful implementations of any kind of bounded treewidth or pathwidth algorithmics, as opposed to purely theoretical asymptotic linear-time 'results'.

(2) *Obstruction sets are frequently enormously large.*

While the GMT insures that many interesting graph properties have finite obstructions sets, these seem to be disappointingly huge in many cases.

Example I. The number of obstructions for planarity (genus 0) is two (by Kuratowski's Theorem [25]) in the minor order. The number of obstructions for embedding in the projective plane is 35 [19]. The number of obstructions for torus embedding is at least 900 [32], and is expertly estimated to be around 2500 [21]. A not unreasonable guess for the number of obstructions for genus 2 embedding would be ninety billion.

Example II. The number of obstructions for pathwidth 1 is two [2]. The number of obstructions for pathwidth 2 is 110 [24]. The number of obstructions for pathwidth 3 is at least 60 000 000 [23].

These numbers indicate that while we may one day succeed in identifying the obstruction set for the torus (on which a considerable amount of effort has already been expended [14,32,34]), the genus 2 obstruction set may well remain beyond reach and beyond any possible interest (although see point (4) below). Our work in implementing a general obstruction set theorem-prover has been guided by the following philosophical conjecture concerning this approach.

The Obstruction Set Mechanization Conjecture. For natural graph properties that are characterized by obstruction sets of feasible size, we should eventually be able to compute these obstruction sets mechanically.

Note that if this conjecture is correct it would be quite interesting, as it would lead to the automation (or at least partial automation) of an important area of classical graph theory. The particular case of the obstructions to torus embedding is an especially appealing target, where the results of this paper are intended to contribute techniques and experience in handling some of the sources of large constants.

(3) *Obstruction sets are occasionally useful in the theory of graphs.*

We do not know presently of any reason why the particular obstruction sets that we report in this paper probably are of interest in themselves, although clearly the property of having a bounded-size Feedback Vertex Set or Feedback Edge Set arises in some contexts and applications (see [17,28,29]). However, a general case can be made that natural obstruction sets are worth identifying whenever they have feasible size (i.e., whenever they *can* be known) because they sometimes turn out to be useful in proving various theorems. The following is one example of such a use of obstruction set information.

Example III. Negami formulated the beautiful and unresolved conjecture that a graph has a finite planar cover if and only if it embeds in the projective plane [30,31]. Arguments by Archdeacon and Negami [3] established that if there is a counterexample, then there must be a counterexample among the 35 obstructions to embedding in the projective plane [19]. A case analysis of the obstruction set, including one case settled only recently in [20], shows that the only possible counterexample to the (still open) conjecture is $K_{1,2,2,2}$.

The proof of the Four Color Theorem is another example of an argument based on considering the elements of a different kind of obstruction set.

(4) *Obstruction sets may be useful in the design of heuristic algorithms.*

For some properties that have large obstruction sets, and for some input distributions, almost all of the useful information from an algorithmic point of view may be concentrated in just a few of the obstructions. Thus, knowing obstruction sets can be useful in the design of heuristic and approximation algorithms of certain kinds.

Example IV. The Pathwidth problem has been considered in VLSI design under the rubric of Gate Matrix Layout. Langston and Ramachandramurthi showed in [27] that even though there are 110 obstructions for Gate Matrix Layout width 3 (which is equivalent to Pathwidth 2), almost all of the useful information for the graphs arising from actual electronic circuits was concentrated in the single obstruction K_4. This allows for a particularly fast and simple heuristic algorithm for the property based on this 'approximate' obstruction set.

Revisiting (2), this kind of approach may mean that partial knowledge of huge obstruction sets may still be useful in algorithm design for some graph properties. A random sampling of the leaves of the search trees described in Section 3 may be one means of computing such an effective sample.

This paper is organized as follows. The next section formally introduces our graph families based on feedback sets. Section 3 presents our general computation theory for computing obstruction sets within the minor order. The last three sections contain our main family-specific results: Section 4 addresses the feedback vertex set families, Section 5 covers the feedback edge set families, and Section 6 investigates the feedback vertex/edge set families.

2. Preliminaries

We begin with some standard definitions and notations. For two graphs G and H, H is a *minor* of G (denoted by $H \leqslant_m G$) if and only if a graph isomorphic to H can be obtained from G by a sequence of operations chosen from: (1) taking a subgraph, and (2) contracting an edge. This operation defines the *minor order* on graphs. A family of graphs $\mathscr{F}$ is a *lower ideal* with respect to $\leqslant_m$ if for all graphs G and H, the conditions (1) $H \leqslant_m G$ and (2) $G \in \mathscr{F}$ imply $H \in \mathscr{F}$. The *obstruction set* $\mathcal{O}(\mathscr{F})$ for $\mathscr{F}$ with respect to $\leqslant_m$ is the set of minimal elements of the complement of $\mathscr{F}$. This characterizes $\mathscr{F}$ in the sense that $G \in \mathscr{F}$ if and only if it is not the case that for some $H \in \mathcal{O}(\mathscr{F})$, $H \leqslant_m G$. The motivation for our research is the consequence of the following Graph Minor Theorem (GMT), formerly known as Wagner's Conjecture, by Robertson and Seymour.

Theorem 1 (GMT). *The minor order*, $\leqslant_m$, *is a well-partial order.*

This theorem guarantees that $\mathcal{O}(\mathscr{F})$ is finite for any minor-order lower ideal $\mathscr{F}$.

2.1. Graphs with small feedback sets

In this paper we characterize by obstructions two main types (and a third hybrid type) of simple graph families. The first family consists of those graphs for which all cycles can be covered with a small set of vertices. The second family consists of those graphs for which all cycles can be covered with a small set of edges. We also study a generalized variety of these two graph families. In this later case cycles are covered by a small number of both vertices and edges.

Graphs with small feedback sets are desirable for many reasons. One specific application deals with the task of minimizing costs in the construction of broadcast-display networks. For this particular model we have two types of nodes and two types of connecting lines. Each node can display and broadcast messages. The less expensive nodes have simple hardware that simply receives, displays and sends messages to neighboring nodes. The more expensive nodes can also detect whether an incoming message is currently being displayed and when not to rebroadcast it. Likewise, the expensive communication lines can sense and censor what is transmitted (by using some type of message buffer). In this model any node can originate a message. Once the message has been flooded and displayed, further broadcasts of the message should cease. By designating a small subset (i.e., a feedback set) of the nodes or edges as smart (expensive) hardware the broadcasting will terminate automatically. (We assume that all nodes do not resend a message back along its incoming message line.) Thus minimizing the size of these feedback sets within the network is important.

These graph families are based on the following two well-known problems (see [18,9]).

Problem 2 (*Feedback Vertex Set*(FVS)).

Input: A graph $G=(V,E)$ and a non-negative integer $k \leqslant |V|$.

Question: Is there a subset $V' \subseteq V$ with $|V'| \leqslant k$ such that V' contains at least one vertex from every cycle in G?

A set V' in the above problem is called a *feedback vertex set* for the graph G. The family of graphs that have a feedback vertex set of size at most k is denoted by k-Feedback Vertex Set. It is easy to verify that for each fixed k the set of graphs in k-Feedback Vertex Set is a lower ideal in the minor order. For a given graph G, let FVS(G) denote the least k such that G has a feedback vertex set of cardinality k. Our second problem of interest is now stated.

Problem 3 (*Feedback Edge Set* (FES)).

Input: A graph $G=(V,E)$ and a non-negative integer $k \leqslant |E|$.

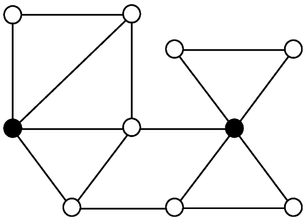
Question: Is there a subset $E' \subseteq E$ with $|E'| \leqslant k$ such that $G \backslash E'$ is acyclic?

The edge set E' is a *feedback edge set*. Also, for a given graph G, let FES(G) denote the least k such that G has a feedback edge set of cardinality k, and the family k-Feedback Edge Set$=\{G \mid \text{FES}(G) \leqslant k\}$.

$$\begin{tikzcd}
 & (k_1+1,k_2+1) & \\
(k_1,k_2+1) \arrow[ur] \arrow[rr] & & (k_1+1,k_2) \arrow[ul] \\
 & (k_1,k_2) \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Fig. 1. The (k_1,k_2)-Feedback Vertex/Edge Set family containment diagram.

Example 4. Displayed below is a graph in the 2-Feedback Vertex Set family. Notice that when the two black vertices are removed from the example, the graph becomes acyclic (a forest).



The reader should note that the graph in the previous example requires 6 edges in any feedback edge set and thus it is a member of 6-Feedback Edge Set.

We now define a third problem based on the above two feedback set problems, where we keep both vertex and edge integer parameters.

Problem 5 (*Feedback Vertex/Edge Set* (FVES)).

Input: A graph $G=(V,E)$ and two non-negative integers $k_1 \leqslant |V|$ and $k_2 \leqslant |E|$.

Question: Is there a subset $V' \subseteq V$ with $|V'| \leqslant k_1$ and a subset $E' \subseteq E$ with $|E'| \leqslant k_2$ such that $(G\backslash E')\backslash V'$ is acyclic?

For fixed integer parameters k_1 and k_2, the graphs that satisfy Problem 5 are members of the feedback set family (k_1,k_2)-Feedback Vertex/Edge Set. For instance, the graph in Example 4 is a member of $(1,3)$-Feedback Vertex/Edge Set. Fig. 1 shows the set-inclusion relationships between various (k_1,k_2)-Feedback Vertex/Edge Set graph families. This diagram illustrates, via the horizontal arrow, that it is easier to cover a graph's cycles with vertices as opposed to edges. Note that we have the following family equivalences:

$$k_1\text{-}\textit{Feedback Vertex Set} = (k_1, 0)\text{-}\textit{Feedback Vertex/Edge Set},$$
$$k_2\text{-}\textit{Feedback Edge Set} = (0, k_2)\text{-}\textit{Feedback Vertex/Edge Set}.$$

2.2. Membership algorithms for the feedback set problems

We now discuss what is known about the computational complexity of solving the various feedback set problems.

It is well-known that the general feedback vertex set problem (Problem 2, where k is part of the input) is $\mathcal{NP}$-complete [18]. However, for many families of graphs the optimization problem of finding the minimum size k for a feedback vertex set can be done in polynomial time. For example, an $O(n^4)$ algorithm is given in [11] for the feedback vertex set problem on co-comparability graphs, which is a superclass of permutation graphs. Among many other known $\mathcal{NP}$-complete problems, Problem 2 can be solved in linear-time for graphs of bounded treewidth (or pathwidth) [1,5]. Later in Section 4.1 we present a linear-time algorithm for the case of graphs of bounded pathwidth. We use a finite-state version of this algorithm to compute the obstruction set for k-Feedback Vertex Set.

It is also known that the general feedback edge set problem (Problem 3) is in the polynomial-time solvable class $\mathcal{P}$. In fact, by a simple formula, given in Section 5.1, we can compute in linear time the minimum size k for a feedback edge set of any graph.

If k is fixed then Problem 2 can be solved in polynomial time by a standard brute-force algorithm. Membership testing is done by checking whether any subset of vertices (or edges) of size k is a feedback set. That is, for fixed k, this brute-force algorithm runs in $\binom{n}{k} \cdot n = O(n^{k+1})$ time, where n equals the number of vertices of the input graph. There exists a more practical membership algorithm that runs in $O((2k)^k n^2)$ time. This algorithm by Fellows and Downey (see [15]) is based on (1) a quick algorithm by Itai and Rodeh in [22] for finding short cycles and (2) the fact that a graph G of minimum degree three with girth at least $2k$ is not in k-Feedback Vertex Set. Also because these graphs have bounded treewidth there exists a theoretically known (but possibly not practical) linear-time membership algorithm since the family is closed under taking minors [5] (or, alternatively, since the family is expressible in monadic second-order logic [12]).

There are two families of optimization problems related to the generalized feedback set problem (Problem 5). If either k_1 or k_2 is fixed, the problem is to minimize the other parameter for any input graph. The first class of problems is tractable (in the complexity class $\mathcal{P}$) and the second class is intractable (i.e., $\mathcal{NP}$-complete). For checking graph membership in (k_1, k_2)-Feedback Vertex/Edge Set we have one polynomial-time membership algorithm. This algorithm runs in $O(n^{k_1+k_2+1})$ time by checking all subsets of vertices and edges of size at most k_1 and k_2, respectively; note the input graph may have fewer than k_1 vertices or k_2 edges. (Since this family is of bounded treewidth for fixed k_1 and k_2, there also exists a theoretically optimal linear-time membership algorithm.)

2.3. Graphs with bounded width

Before presenting our theory for computing minor-order obstructions, we formally define the concept of graphs of bounded (combinatorial) width. Our search theory for finding obstructions is based upon two types of widths. The first classifies those graphs with a narrow path-like structure.

Definition 6. A *path decomposition* of a graph $G=(V,E)$ is a sequence $P=X_1,X_2,\ldots,X_r$ of subsets of V that satisfy the following conditions:

(1) $\bigcup_{1\leqslant i\leqslant r} X_i = V$.
(2) For every edge $(u,v)\in E$, there exists an X_i, $1\leqslant i\leqslant r$, such that $u\in X_i$ and $v\in X_i$.
(3) For $1\leqslant i<j<k\leqslant r$, $X_i\cap X_k\subseteq X_j$.

The *width* of a *path decomposition* $X_1,X_2,\ldots,X_r$ is $\max_{1\leqslant i\leqslant r}|X_i|-1$. The *pathwidth* of a *graph* G is the minimum pathwidth over all path decompositions of G. The family of graphs that have pathwidth at most k is denoted by k-Pathwidth.

The second width metric, which is more popular in the literature, classifies those graphs with a narrow tree-like structure.

Definition 7. A *tree decomposition* of a graph $G=(V,E)$ is a tree T together with a collection of subsets T_x of V indexed by the vertices x of T that satisfies:

(1) $\bigcup_{x\in T} T_x = V$.
(2) For every edge (u,v) of G there is some x such that $u\in T_x$ and $v\in T_x$.
(3) If y is a vertex on the unique path in T from x to z then $T_x\cap T_z\subseteq T_y$.

The *width* of a tree decomposition is the maximum value of $|T_x|-1$ over all vertices x of the tree T. A graph G has *treewidth* at most k if there is a tree decomposition of G of width at most k. The family of graphs that have treewidth at most k is denoted by k-Treewidth.

3. Efficiently computing obstructions

In this section we present a general theory for computing minor-order obstructions when we have the following two ingredients: (1) a pathwidth or treewidth bound on the obstructions, and (2) a family congruence for the family.

Our current theory has evolved from the seminal work presented in [16], where the underlying theory uses the GMT to prove termination of a finite-state search procedure. The results in [26] can be used to prove termination without the GMT. The application of these results for the computation of any particular obstruction set requires additional problem-specific results. These results are non-trivial, but seem to be generally available (in one form or another) for virtually every natural minor-closed family of graphs. We contribute to the feasible aspects of computing obstruction sets.

The basic theory of finite-state obstruction-set computations is applied to a particular lower ideal $\mathscr{F}$ as follows. First, the search space is framed by some type of lemma (specific to $\mathscr{F}$ and of variable difficulty to prove in a sufficiently tight form) that establishes a bound on the maximum pathwidth or treewidth of the graphs in $\mathcal{O}(\mathscr{F})$. Once the search space has been limited to graphs of a specific pathwidth or treewidth bound, we organize the search space algebraically. This is accomplished by describing a finite set of graph-building operators Σ such that every graph in the search space is represented by a string in Σ^* Associated to $\mathscr{F}$ we define a partial order $\leqslant$ on Σ^* such that: (1) $\leqslant$ is compatible with concatenation within Σ^*, (2) $\leqslant$ has a finite number of minimal elements, and (3) from the minimal elements of Σ^* with respect to $\leqslant$ we can recover the obstruction set for $\mathscr{F}$. In order to implement the search we employ problem-specific algorithms that determine $\leqslant$ minimality, and decide membership in $\mathscr{F}$.

The text that follows is a brief but complete description of our search theory. The interested reader should read [13] for further details regarding the actual computer implementation, which includes many efficiency improvements omitted from this paper.

We search for obstructions within the set of graphs of bounded pathwidth (or bounded treewidth). We now describe an algebraic representation for these graphs of bounded width.

Definition 8. A *t-boundaried graph* $G=(V,E,\partial,f)$ is an ordinary graph $G=(V,E)$ together with (1) a distinguished subset of the vertex set $\partial \subseteq V$ of cardinality t, the *boundary* of G, and (2) a bijection $f:\partial \to \{0,1,2,\ldots,t-1\}$. A *boundaried graph* $G=(V,E,\partial,f)$ is an ordinary graph $G=(V,E)$ together with a boundary $\partial \subseteq V$ and labeling injection $f:\partial \to \{0,1,2,\ldots\}$.

The graphs of pathwidth at most t are generated exactly by strings of (unary) operators from the following *operator set* $\Sigma_t = V_t \cup E_t$:

$$V_t=\{\textcircled{0},\ldots,\textcircled{t}\} \quad \text{and} \quad E_t=\{\boxed{ij} \mid 0 \leqslant i<j \leqslant t\}.$$

To generate the graphs of treewidth at most t, an additional (binary) operator $\oplus$, called *circle plus*, is added to Σ_t. The semantics of these operators on boundaried graphs G and H of boundary size at most $t+1$ are as follows:

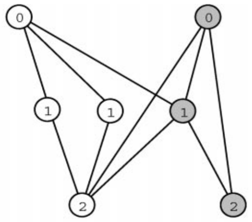
$G\textcircled{i}$ Add an isolated vertex to the graph G, and label it as the new boundary vertex i.

$G\ \boxed{ij}$ Add an edge between boundary vertices i and j of G (ignore if operation causes a multi-edge).

$G \oplus H$ Take the disjoint union of G and H except that equal- labeled boundary vertices of G and H are identified.

It is syntactically incorrect to use the operator $\boxed{ij}$ without being preceded by both ⓘ and ⓙ, and the operator $\oplus$ must be applied to graphs with the same boundary ∂. A graph described by a string (tree, if $\oplus$ is used) of these operators is called a *t-parse*, and has an implicit labeled boundary ∂ of at most $t+1$ vertices. By convention, a t-parse always begins with the operator string [⓪, ①, …, ⓣ] which represents the edgeless graph of order $t+1$. Throughout this paper, we refer to a t-parse and the graph it represents interchangeably.

Example 9. A 2-parse and the graph it represents; the shaded vertices denote the final boundary ∂.



[⓪, ①, ②, $\boxed{0\ 1}$, $\boxed{1\ 2}$, ①, $\boxed{0\ 1}$, $\boxed{1\ 2}$, ①, $\boxed{0\ 1}$, $\boxed{1\ 2}$, ⓪, $\boxed{0\ 1}$, $\boxed{0\ 2}$, ②, $\boxed{0\ 2}$, $\boxed{1\ 2}$]

For ease of discussion throughout the remaining part of this paper, we limit ourselves to bounded pathwidth in the obstruction set search theory and only point out places where any difficulty may occur with a bounded treewidth search.

Definition 10. Let $G = (g_1, g_2, \ldots, g_n)$ be a t-parse and $Z = (z_1, z_2, \ldots, z_m)$ be any sequence of operators over Σ_t. The *concatenation* $(\cdot)$ of G and Z is defined as

$$G \cdot Z = (g_1, g_2, \ldots, g_n, z_1, z_2, \ldots, z_m).$$

The t-parse $G \cdot Z$ is called an *extended t-parse*, and $Z \in \Sigma_t^*$ is called an *extension*. (For the treewidth case, G and Z are viewed as two connected subtree factors of a parse tree $G \cdot Z$ instead of two parts of a sequence of operators.)

The following sequence of definitions and results forms our theoretical basis for computing minor-order obstruction sets.

Definition 11. Let G be a t-parse. A t-parse H is a ∂-*minor* of G, denoted $H \leqslant_{\partial m} G$, if H is a combinatorial minor of G such that no boundary vertices of G are deleted by the minor operations, and the boundary vertices of H are the same as the boundary vertices of G.

Definition 12. Let G be a t-parse. A t-parse H is a *one-step ∂-minor* of G if H is obtained from G by a single ∂-minor operation (one non-boundary isolated vertex deletion, one edge deletion, or one non-boundary-boundary edge contraction).

Both the family of graphs of pathwidth at most k, k-Pathwidth and the family of graphs of treewidth at most k, k-Treewidth are lower ideals in the minor order. Thus any ∂-minor H of a t-parse G can be represented as a t-parse. Our minor-order algorithms actually operate on the t-parses directly, bypassing any unnecessary conversion to and from the standard graph representations.

Definition 13. Let $\mathscr{F}$ be a fixed graph family and let G and H be t-parses. We say G and H are *$\mathscr{F}$-congruent* (written $G \sim_{\mathscr{F}} H$) if for every extension $Z \in \Sigma_t^*$,

$$G \cdot Z \in \mathscr{F} \Leftrightarrow H \cdot Z \in \mathscr{F}.$$

If G is not congruent to H, denoted by $G \not\sim_{\mathscr{F}} H$, then we say G is *distinguished* from H (by Z), and Z is a *distinguisher* for G and H. Otherwise, G and H *agree* on Z. The congruence $\sim_{\mathscr{F}}$ is called the *canonical congruence* for $\mathscr{F}$ (of width t).

Definition 14. A set $T \subseteq \Sigma_t^*$ is a *testset* if $G \not\sim_{\mathscr{F}} H$ implies there exists $Z \in T$ that distinguishes G and H.

In the more familiar and general setting of t-boundaried graphs (using an analogue of the Myhill–Nerode Theorem [16]), a testset T may be considered to be a subset of t-boundaried graphs where concatenation $(\cdot)$ is replaced solely by circle plus $\oplus$. It is all right for the pathwidth of $G \oplus T$ to be greater than the pathwidth of the t-boundaried graph G that is being tested. As we will see later, a testset is only useful for finding obstruction sets if it has finite cardinality.

Definition 15. A t-parse G is *non-minimal* if G has a proper ∂-minor H such that $G \sim_{\mathscr{F}} H$. Otherwise, we say G is *minimal.* A t-parse G is a *boundary obstruction* if G is minimal and $G \notin \mathscr{F}$. The set of all *boundary obstructions* for $\mathscr{F}$ is denoted $\mathcal{O}_\partial(\mathscr{F})$.

In general, if a family $\mathscr{F}$ is a minor-order lower ideal and G is minimal with respect to $\mathscr{F}$, then for each proper ∂-minor H of G, there exists an extension Z such that

$$G \cdot Z \notin \mathscr{F} \quad \text{and} \quad H \cdot Z \in \mathscr{F}.$$

That is, there exists a distinguisher for each proper ∂-minor H of G.

The obstruction set $\mathcal{O}(\mathscr{F})$ for a family $\mathscr{F}$ is obtainable from the *boundary obstruction set $\mathcal{O}_\partial(\mathscr{F})$*, that is the set of boundaried graphs (t-parse s) that are boundary obstructions. This is achieved by contracting (possibly zero) edges on the boundaries of $\mathcal{O}_\partial(\mathscr{F})$, whenever the search space of width $\partial - 1$ is large enough. In our search for $\mathcal{O}_\partial(\mathscr{F})$, we must prove that each t-parse generated is minimal or non-minimal. The following two results substantially reduce the computation time required to determine these proofs.

Lemma 16. *A t-parse G is minimal if and only if G is distinguished from each one-step ∂-minor of G. Or equivalently, G is non-minimal if and only if G is $\mathscr{F}$-congruent to a one-step ∂-minor.*

Proof. We prove the second statement. Let G be non-minimal and suppose there exists two distinct proper ∂-minors K and H of G such that $K \leqslant_{\partial m} H$ and $K \sim_{\mathscr{F}} G$. It is sufficient to show $H \sim_{\mathscr{F}} G$.

For any extension $Z \in \Sigma_t^*$, if $G \cdot Z \in \mathscr{F}$ then $H \cdot Z \in \mathscr{F}$ since $H \cdot Z \leqslant_{\partial m} G \cdot Z$ and $\mathscr{F}$ is a ∂-minor lower ideal. Now let Z be any extension such that $G \cdot Z \notin \mathscr{F}$. Since $K \sim_{\mathscr{F}} G$, we have $K \cdot Z \notin \mathscr{F}$. And since $K \cdot Z \leqslant_{\partial m} H \cdot Z$, we also have $H \cdot Z \notin \mathscr{F}$. Therefore, G is $\mathscr{F}$-congruent to H. □

Lemma 17 (Prefix lemma). *If $G_n = [g_1, g_2, \ldots, g_n]$ is a minimal t-parse then any prefix t-parse $G_m = [g_1, g_2, \ldots, g_m]$, $m < n$, is also minimal.*

Proof. Assume G_n is non-minimal. It suffices to show that any extension of G_n is non-minimal. Let H be a one-step ∂-minor of G_n such that for every $Z \in \Sigma_t^*$,

$$G_n \cdot Z \in \mathscr{F} \Leftrightarrow H \cdot Z \in \mathscr{F}.$$

Let $g_{n+1} \in \Sigma_t$ and $G_{n+1} = G_n \cdot g_{n+1}$. Now $H' = H \cdot g_{n+1}$ is a one-step ∂-minor of G_{n+1} such that for all $Z \in \Sigma_t^*$,

$$G_{n+1} \cdot Z = G_n \cdot (g_{n+1} \cdot Z) \in \mathscr{F} \Leftrightarrow H' = H \cdot (g_{n+1} \cdot Z) \in \mathscr{F}.$$

Thus, any extension of G_n is non-minimal. □

The above two lemmata also hold when the circle plus operator $\oplus$ is included in Σ_t. For illustration consider the Prefix lemma: If G is a non-minimal t-parse with a $\mathscr{F}$-congruent proper minor G', and Z is any t-parse, then $(G \oplus Z)'$ is a $\mathscr{F}$-congruent minor of a non-minimal $G \oplus Z$, where we use the prime symbol to denote the corresponding minor operation done to the G part of $G \oplus Z$. (The awkward notation is needed since $G' \oplus Z$ may equal $G \oplus Z$ when common boundary edges exist in both G and Z.)

The Prefix lemma implies that every minimal t-parse is obtainable by extending some minimal t-parse, providing a finite tree structure for the search space. In other words, the search tree may be pruned whenever a non-minimal t-parse is found. See Fig. 2 for an illustration of this search process. Since most $(t+1)$-boundaried graphs have many t-parse representations, we can further reduce the size of the search tree by enforcing a canonical structure on the t-parses considered. That is, we want to generate just one isomorphic copy of each underlying boundaried graph. To do this we have to ensure that every prefix of every canonic boundary obstruction (a minimal leaf of the

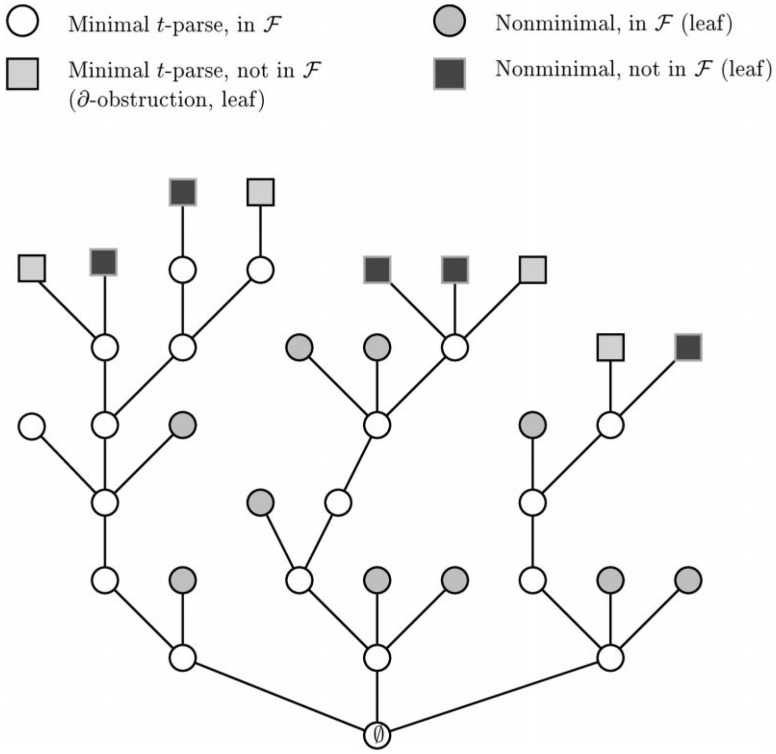

Minimal t-parse, in $\mathcal{F}$
Nonminimal, in $\mathcal{F}$ (leaf)
Minimal t-parse, not in $\mathcal{F}$ (∂-obstruction, leaf)
Nonminimal, not in $\mathcal{F}$ (leaf)
$\emptyset$

Fig. 2. A typical t-parse search tree (each edge denotes one operator).

search tree) is also canonic (see [13]). This reduces the out-degree of every node in the search tree to sometimes less than $|\Sigma_t|$ (and sometimes 0).

We currently use the four techniques given in Fig. 3 to prove that a t-parse in the search tree is minimal or non-minimal. They are listed in the order that they are attempted; if one succeeds, the remainder do not need to be performed. The first three of these may not succeed, though the fourth method always will. However, if we are fortunate to have a *minimal* finite-state congruence (i.e., not a refinement of the minimum automaton for $\sim_{\mathscr{F}}$) in step 2 of Fig. 3 then we can stop at that step since distinct final states (equivalence classes) imply the existence of an extension to distinguish the two states (and their t-parse representatives). An example of such a minimal finite-state congruence was used in our k-Vertex Cover characterizations [7].

4. Graphs with small feedback vertex sets

We now focus on two problem-specific details for finding the k-Feedback Vertex Set obstruction sets: a finite-index congruence and a complete testset (i.e., steps 2 and 4 of Fig. 3). We first present a practical, linear time algorithm for the feedback

(1) *Direct nonminimal test.* These are easily observable properties of t-parses that imply t-parses nonminimal. For any k–Feedback Vertex Set family, the existence of a degree one non-boundary vertex is an example of such a property.
(2) *Finite-state congruence algorithm.* Such an algorithm is a refinement of the minimal finite-state (linear/tree) automaton for $\sim_{\mathcal{F}}$. This means that if a t-parse G and a one-step ∂-minor G' of G have the same final state, then $G \sim_{\mathcal{F}} G'$, and G is nonminimal. If G and G' have distinct final states, no conclusion can be reached (unless the algorithm is actually the minimal finite-state congruence).
(3) *Random minor-distinguisher search.* The proof that a t-parse G is minimal can consist of a distinguisher for each one-step ∂-minor G' of G. Such distinguishers can often be easily obtained by randomly generating a sequence of operators Z such that $G \cdot Z \notin F$, and then checking if $G' \cdot Z \in F$.
(4) *Full testset proof.* We use a complete testset (see Definition 14) to determine if a t-parse G is distinguished from each of its one-step ∂-minors. A t-parse G is nonminimal if and only if it has a one-step ∂-minor G' such that G and G' agree on every test.

Fig. 3. Determining if a t-parse is minimal or non-minimal.

vertex set problem on graphs of bounded pathwidth/treewidth in t-parse form. This general-purpose algorithm is altered to act as a finite-index congruence, that is a refinement of the canonical congruence. We then show how to produce testsets for the graph families k-Feedback Vertex Set, $k \geqslant 0$, with respect to any boundary size t.

4.1. A finite state algorithm (for FVS)

Throughout the following discussion the boundary size (and width) of a t-parse is fixed. Recall that the current set of boundary vertices of a t-parse is denoted by the ∂ symbol. For any subset S of the boundary ∂, we define the following for all prefixes G_m of G_n, $m \leqslant n$.

$$F_m(S) = \begin{cases} \text{The least } k \text{ such that there is a feedback vertex set } V' \text{ of} \\ G_m \text{ with } V' \cap \partial = S \text{ and } |V'| = k, \text{ otherwise } \infty \text{ whenever} \\ (G_m \cap \partial) \backslash S \text{ contains a cycle.} \end{cases}$$

Any feedback vertex set V', consisting of $F_m(S)$ vertices of G_m, is called a *witness set* with respect to S if $V' \cap \partial = S$. Associated with V' is a *witness forest* consisting of the trees (or topological equivalent trees as described below) containing at least one boundary vertex in $G_m \backslash V'$. A witness forest tells us how tight the boundary vertices are held together. Some of these forests are more concise than others for representing how vertex deletions can break up the boundary.

For two witness forests A and B, with respect to $F_m(S)$, we say $A \leqslant_w B$ if the following two conditions hold:

(1) For any two boundary vertices i and j, i and j are connected in A if and only if i and j are connected in B.
(2) If for any t-parse extension Z where there exists some non-boundary vertex b of B such that $(B\backslash\{b\}) \cdot Z$ is acyclic then there exists a non-boundary vertex a of A such that $(A\backslash\{a\}) \cdot Z$ is acyclic.

Also two witness forests A and B are (*topological*) *equivalent*, $A \equiv_w B$, if $A \leqslant_w B$ and $B \leqslant_w A$. Any witness forest with a minimal (not necessarily minimum) number of vertices (within its $\equiv_w$ equivalence class) is called a *park*. The next lemma provides a way of cleaning up a forest to yield a park.

Lemma 18. *A witness forest W of G_m may be reduced to a park as follows*:
(a) *all leafs* (*end vertices*) *not on the boundary may be pruned*, *and*
(b) *any non-boundary vertex v of degree two may have an incident edge contracted if the neighborhood $N(v) \nsubseteq \partial$.*

Proof. See [10]. □

There may exist alternative witness forests that preserve minimum-sized feedback vertex sets for all extensions of G_m. A witness forest W is considered to be a park if the above lemma cannot be applied to W.

Lemma 19. *There are at most* $3t$–3 *vertices in any park for boundary size t.*

Proof. First we consider the degree two non-boundary vertices. For such a vertex v, each of its neighbors must be a boundary vertex. After viewing v and its two incident edges as a single edge between two boundary vertices, we see that at most $t-1$ such vertices can occur. Otherwise, a cycle would exist on the boundary.

Now we consider the remaining non-boundary vertices. Let p be the number of such vertices and e be the edge size of the subpark. Using the fact that the size of a forest must be strictly less than the order, we have $e < t + p$. Since the sum of the vertex degrees is twice the size, we also have $t + 3p \leqslant 2 \cdot e$. Combining these inequalities while solving for p we get

$$\frac{t+3p}{2} \leqslant e \leqslant t + p - 1 \quad \text{or} \quad p \leqslant t - 2.$$

Summing up the boundary (t), the degree two vertices ($t-1$), and the degree three or more vertices ($t-2$), shows that the order of any park can be at most $3t-3$. □

Corollary 20. *There is a finite number of parks with boundary size t.*

Proof. Since we have a bound on the number of vertices for a park, we can apply Cayley's Tree Formula (i.e., by counting the number of labeled trees/forests) to get a bound on the total number of distinct parks. There are n^{n-2} labeled trees of order n. □

The results of the previous lemma and its corollary may be strengthened. See, for example, the closely related Lemma 24. However, these bounds are sufficient for our purposes — to show that there is a manageable (constant) number of parks (i.e., our algorithm can be used as a finite-index congruence).

For each subset S (with complement $\bar{S} = \partial \backslash S$) of the set of boundary vertices our algorithm keeps track of the related parks in the following sets.

$$P_m(S) = \{P \mid P \text{ is a park of } G_m \text{ with leaves and branches over } \bar{S}\}.$$

Now we finally present a linear time dynamic-programming algorithm for the feedback vertex set problem which is used as our finite congruence for t-parses. This general-purpose algorithm has the same structure as our vertex cover algorithm given in [7], indicating a standard approach for developing such algorithms. The one-pass algorithm simply makes a transition from one state to another for each operator of a t-parse $G_n = [\textcircled{0}, \ldots, \textcircled{t}, g_{t+1}, \ldots, g_n]$. Thus, after all the parks $\{P_m(S) \mid S \subseteq \partial\}$ are determined (for G_m), all the parks $\{P_i(S) \mid S \subseteq \partial\}$ for $i < m \leqslant n$ are never referenced and may be discarded.

Our algorithm, given in Fig. 4, starts by setting the sizes for the minimal feedback vertex sets on $G_m = G_t$, the edgeless graph with $t + 1$ boundary vertices. This is done for all $S \subseteq \partial$. There is only one park associated with $F_t(S)$ at this stage, namely the isolated forest with $t + 1 - |S|$ vertices. We break up the dynamic step into cases depending on what type of operator is at position $m + 1$ and the condition (selected in S or not) of any affected boundary vertices of G_m or G_{m+1}. These transitions are described in Cases 1–4 of Fig. 4. When the algorithm reaches the end of the t-parse, it computes the minimum number of vertices needed in any feedback vertex set for G_n by taking the least $F_n(S)$.

For space reasons we leave out the rules required to update the sets of parks $P_m(S)$ throughout each iteration of step II of the feedback vertex set algorithm. This procedure essentially entails extending the parks with the current operator and reducing them by the rules given in Lemma 18. For example in Case 4 where question (b) is true, the set of witness forests $P_{m+1}(S)$ for $F_{m+1}(S)$ becomes the set of witness forests $P_m(S)$ with the edge (i, j) added to each forest. If equality holds for the two $F_m(\ )$'s in Cases 1 and 2 we combine park sets by simply taking the union of the two sets of witness forests.

Example 21. Table 1 shows values of $F_m(S)$ for the application of the feedback vertex set algorithm to the 2-parse given in Example 9. As can been seen by examining the graph in Example 9, a minimum feedback vertex set has cardinality 2, which corresponds to the minimum value in the last column.

I For $m = t$ and every $S \subseteq \partial$ set

$$F_t(S) = |S| \quad .$$

II For $t \leq m < n$ do the following cases:
Case 1: vertex operator $\textcircled{i}$ and $i \notin S$

$$F_{m+1}(S) = \min\{F_m(S), F_m(S \cup \{i\})\}$$

Case 2: vertex operator $\textcircled{i}$ and $i \in S$

$$F_{m+1}(S) = \min\{F_m(S), F_m(S \setminus \{i\})\} + 1$$

Case 3: edge operator $\boxed{i\ j}$ where $i \in S$ or $j \in S$

$$F_{m+1}(S) = F_m(S)$$

Case 4: edge operator $\boxed{i\ j}$ where $i \notin S$ and $j \notin S$
a) If the edge operator creates a cycle on $\bar{S}$ in G_{m+1} or $F_m(S) = \infty$ then

$$F_{m+1}(S) = \infty \quad .$$

b) If there exists a park in $P_m(S)$ such that i and j are in different trees then

$$F_{m+1}(S) = F_m(S)$$

else

$$F_{m+1}(S) = F_m(S) + 1 \quad .$$

III Compute answer:

$$FVS(G) = \min\{F_n(S) \mid S \subseteq \partial\} \quad .$$

Fig. 4. A general feedback vertex set algorithm for t-parses.

Theorem 22. *For any t-parse* $G_n = [\textcircled{0}, \ldots, \textcircled{t}, g_{t+1}, \ldots, g_n]$, *the algorithm in Fig.* 4 *correctly computes* $\mathrm{FVS}(G_n)$.

Proof. See [10]. □

The dynamic program, given in Fig. 4, for determining the feedback vertex set of a pathwidth t-parse is easily modified to handle treewidth t-parses. All that is needed is to add a Case 5 in part II which takes care of the circle plus operator $G_i \oplus G_j$. This new case is a little messy since the states for the two subtree parses G_i and G_j need to be interleaved. Briefly stated, this is done by checking all combinations (unions) of boundary subsets S_i and S_j of G_i and G_j (resulting in a subset S of $G_i \oplus G_j$) along with checking which best parks from G_i can be glued together with the compatible parks from G_j to form a set of parks for $G_i \oplus G_j$. If the glued parks create any cycles then the value of $F_{\text{‘tree index’}}(S)$ needs to be increased to account for additional feedback vertices.

Table 1
Feedback vertex set state tables computed for Example 21

m	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
S g_m	–	$\boxed{01}$	$\boxed{12}$	①	$\boxed{01}$	$\boxed{12}$	①	$\boxed{01}$	$\boxed{12}$	⓪	$\boxed{01}$	$\boxed{02}$	②	$\boxed{02}$	$\boxed{12}$
$\emptyset$	0	0	0	0	0	1	1	1	2	1	1	∞	1	1	∞
$\{0\}$	1	1	1	1	1	1	1	1	1	2	2	2	2	2	**2**
$\{1\}$	1	1	1	1	1	1	2	2	2	2	2	2	2	2	**2**
$\{2\}$	1	1	1	1	1	1	1	1	1	1	1	1	2	2	**2**
$\{0,1\}$	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
$\{0,2\}$	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3
$\{1,2\}$	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3
$\{0,1,2\}$	3	3	3	3	3	3	3	3	3	3	3	3	4	4	4

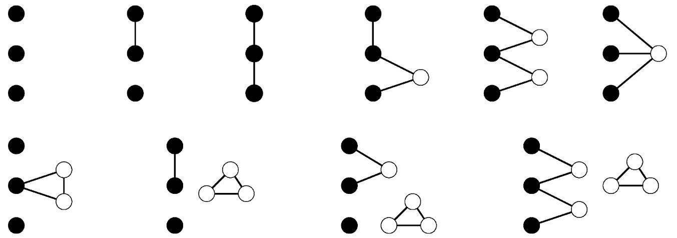
We can convert the above feedback vertex set algorithm to a finite-index congruence for k-Feedback Vertex Set. This is accomplished by restricting the values of $F_m(S)$ to be in $\{0,1,\ldots,k,k+1\}$; we are only interested in knowing whether or not there exists a feedback vertex set of size at most k containing S. (The value of $k+1$ acts as the value ∞ in the congruence.) In our application for finding the k-Feedback Vertex Set obstruction sets, we actually use a congruence with slightly fewer states than the one just described. The key idea to this improvement is noticing that if a park P is a minor of a park P' then only the representative P is needed as a witness. We estimate that this allows us to prove approximately 5% more t-parses non-minimal via the dynamic-programming congruence check. That is, for certain instances we avoid our CPU-intensive testset proof method, which is described next.

4.2. *A complete testset (for* FVS*)*

A finite testset for the feedback vertex set canonical congruence $\sim_{\mathscr{F}}$ is easy to produce. The individual tests closely resemble the parks described above. The testset that we use consists of forests augmented with isolated triangles (and/or triangles solely attached to a single boundary vertex). Our k-Feedback Vertex Set testset T_t^k consists of all t-boundaried graphs that have the following properties:

(1) Each graph is a member of k-Feedback Vertex Set.
(2) Each graph is a forest with zero or more isolated triangles, K_3's.
(3) Every isolated triangle has at most one boundary vertex.
(4) Every degree one vertex is a boundary vertex (i.e., every tree component has at least two boundary vertices).
(5) Every non-boundary degree two vertex is adjacent to two boundary vertices.

Example 23. Some 3-boundaried tests (of T_3^1) for 1-Feedback Vertex Set are shown below.



The above restrictions on members of T_t^k gives an upper bound on the number of vertices, as stated in the following lemma. Hence T_t^k is a finite testset.

Lemma 24. *The number of vertices for any test* $T \in T_t^k$ *is at most* $3k + 2t - 1$.

Proof. See [10]. □

The above bound is tight since the test T consisting of k isolated triangles and $t-1$ interior degree two vertices, each adjacent to boundary vertex i and $i+1$, has $3k+2t-1$ vertices (see, for example, the last test given in Example 23).

Since these k-Feedback Vertex Set testsets are based solely on t-boundaried graphs, they are useful for both pathwidth and treewidth t-parse obstruction set computations.

Theorem 25. *The set of t-boundaried graphs* T_t^k *is a complete testset for the graph family* k-Feedback Vertex Set.

Proof. Assume G and H are two t-boundaried graphs that are not $\mathscr{F}$-congruent within the family $\mathscr{F} = k$-Feedback Vertex Set. Let Z be any t-boundaried graph that distinguishes G and H with $G \oplus Z \in \mathscr{F}$ and $H \oplus Z \notin \mathscr{F}$. We show how to build a t-boundaried graph $T \in T_t^k$ from Z that also distinguishes G and H. Let W be a set of k witness vertices such that $(G \oplus Z)\backslash W$ is acyclic. From W, let $W_G = W \cap G$, $W_\partial = W \cap \partial$ and $W_Z = W \cap Z$. Take T' to be $Z \backslash W$ plus $|W_Z|$ isolated triangles, plus $|W_\partial|$ triangles with each containing a single boundary vertex from W_∂. If T' contains any component $C \not\simeq K_3$ without boundary vertices, replace it with FVS(C) isolated triangles. Clearly, $G \oplus T' \in \mathscr{F}$ since W_G plus one vertex from each of the non-boundary isolated triangles of T' is a witness set of k vertices. If $H \oplus T' \in \mathscr{F}$ then this contradicts the fact that $H \oplus Z \notin \mathscr{F}$ by using a witness set containing W_Z, W_∂ and the interior witness vertices of H (with respect to $H \oplus T'$). Finally, we construct a distinguisher $T \in T_t^k$ by minimizing T' (using the reducing operations of Lemma 18) to satisfy the 5 properties listed above. (Note that the extension T is created by not eliminating any cycles in the extension T'.) □

For the graph family 1-Feedback Vertex Set on boundary size 4, the above testset consists of only 546 tests. However, for 2-Feedback Vertex Set on boundary size 5, the above testset contains 14 686 tests. As can be seen by the increase in the number of tests, a more compact feedback vertex set testset would be needed (if possible) before we attempt to work with boundary sizes larger than 5. The large number of tests (especially T_5^2) for the feedback vertex set families indicates why using the testset step to prove t-parses minimal or non-minimal is the most CPU-intensive part of our obstruction set search (and is why this is attempted last).

*4.3. The k-*Feedback Vertex Set *obstructions*

Our search for the 1-Feedback Vertex Set and 2-Feedback Vertex Set obstructions is now presented. As mentioned in Section 3, we need some type of lemma that bounds the search space. The following well-known treewidth bound can be found in [35] along with other introductory information concerning the minor order and obstruction sets. We provide a proof in order to suggest how generous the bound is for the k-Feedback Vertex Set obstructions, which is a very small subset of the $(k+1)$-Feedback Vertex Set family.

Lemma 26. *A graph in k-*Feedback Vertex Set *has treewidth at most $k+1$.*

Proof. Let $G=(V,E)$ be a member of k-Feedback Vertex Set and $V' \subseteq V$ be a set of k witness vertices such that $G' = G \setminus V'$ is acyclic. The remaining forest G' has a tree decomposition T of width 1. The tree decomposition T' consisting of the vertex sets of $T=\{T_x\}$ augmented as $T'_x = T_x \cup V'$ is a tree decomposition for G of width $k+1$. □

Corollary 27. *An obstruction for k-*Feedback Vertex Set *has treewidth at most $k+2$.*

Proof. Let G be an obstruction and v any vertex of G. By definition of being a proper minor of an obstruction, $G' = G \setminus \{v\}$ is a member of k-Feedback Vertex Set. Since G' has a tree decomposition T of width at most $k+1$, we can add the vertex v to each vertex set of T, yielding a tree decomposition of width at most $k+2$ for G. □

For the graph family 2-Feedback Vertex Set we can derive a stronger statement.

Theorem 28. *If G is an obstruction to* 2-Feedback Vertex Set *then the pathwidth of G is at most* 4.

Poof. For any obstruction G we use the following two properties:

(1) For any edge (u,v), $G \setminus \{(u,v)\}$ is in 2-Feedback Vertex Set with some feedback set $\{x,y\}$ such that $\{u,v\} \cap \{x,y\} = \emptyset$.

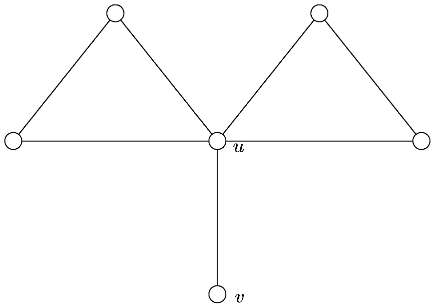


Fig. 5. Forbidden substructure within the 2-Pathwidth obstructions.

(2) The obstruction G does not contain any vertices of degree 1, and for any vertex u of degree 2 there is an edge between the neighbors of u.

If any obstruction did not satisfy the first property then $\{x, y\}$ would also be a feedback vertex set for G, which we know has to have cardinality three. If any obstruction did not satisfy the second property then we could find a single edge-contracted minor G' such that $\mathrm{FVS}(G') = \mathrm{FVS}(G)$.

Property 1 implies that there exist two vertices x and y such that $G' = G \setminus \{x, y\}$ has exactly one cycle. If G' has pathwidth at most 2 then G has pathwidth at most 4. If G' has pathwidth more than 2 then it must contain at least one of the pathwidth 2 obstructions as a minor. In particular, any such obstruction for 2-Pathwidth must also be a member of 1-Feedback Edge Set. All of the 23 possible forbidden minors with one cycle, given in [24], have at least three pendant paths of length 2, i.e., three legs of the spider graph $S(K_{1,3})$, attached to the single cycle ($S(K_{1,3})$ is $K_{1,3}$ with each edge subdivided).

Property 2 is applied as follows. By considering incident edges from vertices x and y to G', we know that G must have: (a) three disjoint cycles or (b) one cycle and a disjoint mini-clover (see Fig. 5) as proper minors. But this means for (a) that G is properly above the 2-Feedback Vertex Set obstruction $3K_3$ and for (b) Property 1 cannot hold for the stem edge (u, v) of the mini-clover.

Thus for any obstruction G there exists two vertices x and y such that $G' = G \setminus \{x, y\}$ has pathwidth at most 2. This fact implies that G has pathwidth at most 4. □

Besides the single obstruction K_3 for the trivial family 0-Feedback Vertex Set, the connected obstructions for 1-Feedback Vertex Set and the connected obstructions for 2-Feedback Vertex Set are shown in Figs. 8 and 9. The two connected obstructions for 1-Feedback Vertex Set were found in about 3 h of accumulated CPU time when combining 4 worker processes, a database manager process, and a dispatcher process running concurrently. Our pathwidth 4 search for 2-Feedback Vertex Set consumed over 40 thousand hours of CPU time running for about three months in duration while averaging 20 workers (initially with a collection of 15–30 SUN Sparcs, and later including a few IBM 6000s and two Cray Y-MPs).

Table 2
Summary of our 2-Feedback Vertex Set obstruction set computation

Pathwidth Four Prefixes for Feedback Vertex Set 2	Boundaried obstructions	Minimal t-parses	Total proofs
[0,1,2,3,4,01,02,0,03,12]	0		
[0,1,2,3,4,01,02,0,01,13]	0		
[0,1,2,3,4,01,02,03,0,12]	0		
[0,1,2,3,4,01,02,03,12,14]	0		
[0,1,2,3,4,01,02,03,14,24]	0		
[0,1,2,3,4,01,02,03,12,13]	0		
[0,1,2,3,4,01,02,03,04,12]	1	15	211
[0,1,2,3,4,01,02,03,0,04]	0		
[0,1,2,3,4,01,02,0,01,12]	0	150	2251
[0,1,2,3,4,01,02,0,03,04]	0	233	3271
[0,1,2,3,4,01,02,03,04,0]	10	5177	74611
[0,1,2,3,4,01,02,03,0,01]	16	68634	1013641
[0,1,2,3,4,01,02,0,01,03]	13	153772	2286001
Prefix = [0, 1, 2, 3, 4, 01, 02, 0, 01, 02]			
Prefix + [03, 04, 0]	9	105482	1565416
Prefix + [03, 04, 12]	10	91976	1359376
Prefix + [03, 04, 13]	0	5241	78436
Prefix + [03, 04, 34]	0	509	7636
Prefix + [03, 12]	10	35976	532651
Prefix + [03, 13]	0	260	3886
Prefix + [03, 14]	0	45	676
Prefix + [03, 34]	0	41	616
Prefix + [12]	2	10517	157231
Prefix + [13]	0	13	151

Table 2 contains a brief summary of how many proofs our distributive computer system had to find for 2-Feedback Vertex Set (pathwidth 4). The first column states various starting (or restarting) points in the search tree. Lack of memory and disk space is the main reason for the separate runs. (Recall our search process of Fig. 2; we can independently search throughout the minimal t-parse space, beginning at various internal nodes.) The second column gives the number of canonic non-boundaried obstructions that have the given prefix. The 'minimal nodes' column gives the number of minimal t-parses that we encountered; these are the internal nodes of our search tree plus any boundaried obstructions. The last column gives the total number of graphs the system had to check. This total includes those t-parses that were proved minimal or non-minimal. The missing entries in the table represent places that were fast dead-end runs (i.e., small subtrees of the search tree leading only to non-minimal t-parses) and we did not bother keeping the proofs.

We believe that 2-Feedback Vertex Set may be the only feasible feedback vertex set family to characterize since there are at least 744 obstructions to 3-Feedback Vertex Set. In fact this count is a very small percentage since we know of an obstruction with order 15 and we have only searched through a subset of the graphs with maximum order 10.

For the two obstruction sets for the 'within one/two vertices of acyclic' families, we present only the connected obstructions since any disconnected obstruction O of the lower ideal k-FEEDBACK VERTEX SET is a union of graphs from $\bigcup_{i=0}^{k-1} \mathcal{O}(i\text{-FEEDBACK VERTEX SET})$ such that $\mathrm{FVS}(O) = k+1$.

Example 29. Since K_3 is an obstruction for 0-FEEDBACK VERTEX SET, and K_4 is an obstruction for 1-FEEDBACK VERTEX SET, the graph $K_3 \cup K_4$ is an obstruction for 2-FEEDBACK VERTEX SET.

Some patterns become apparent in these two sets of obstructions such as the following easily-proven observation.

Observation 30. For the family k-FEEDBACK VERTEX SET, the complete graph K_{k+3}, the augmented complete graph $A(K_{k+2})$ which has vertices $\{1,2,\ldots,k+2\} \cup \{v_{i,j} \mid 1 \leqslant i < j \leqslant k+2\}$ and edges

$$\{(i,j) \mid 1 \leqslant i < j \leqslant k+2\} \cup \{(i, v_{i,j}) \text{ and } (v_{i,j}, j) \mid 1 \leqslant i < j \leqslant k+2\},$$

and the augmented cycle $A(C_{2k+1})$ are obstructions.

A useful property unique to k-FEEDBACK VERTEX SET obstructions that does not hold for the other feedback set families studied in this paper is the following result, which implies that t-parses with cut-vertices are non-minimal.

Lemma 31. *If G is an obstruction to k-*FEEDBACK VERTEX SET *then G has no cut-vertices.*

Proof. See [10]. □

5. Graphs with small feedback edge sets

This section first focuses on two problem-specific areas for computing the k-FEEDBACK EDGE SET obstruction sets using our general method of computing forbidden minors: a direct minimal test and a complete testset (i.e., Steps 1 and 4 of Fig. 3). With these developed ingredients we computed the obstructions for k-FEEDBACK EDGE SET for $k \leqslant 5$ (pathwidth at most 4). This section then describes a family-specific algorithm that does not require a pathwidth or treewidth bound for generating all of the forbidden minors for k-FEEDBACK EDGE SET. With this algorithm we verified that the obstructions for 5-FEEDBACK EDGE SET do indeed have pathwidth at most 4, and also computed the connected obstructions for 6-FEEDBACK EDGE SET.

5.1. *A minimal t-parse algorithm (for* FES*)*

We first describe a simple graph-theoretical characterization for the graphs that are within k edges of acyclic, where k is any non-negative integer. This trivial result also shows that Problem 3 (i.e., determining the minimum feedback edge set of a graph) has a linear-time decision algorithm.

Theorem 32. *A graph $G=(V,E)$ with c components has* $\mathrm{FES}(G)=k$ *if and only if* $|E|=|V|-c+k$.

Proof. For $k=0$ the result follows from the standard result for characterizing forests. If $\mathrm{FES}(G)=k$ then deleting the k witness edges produces an acyclic graph and thus $|E|=|V|-c+k$. Now consider a graph G with $|V|-c+k$ edges for some $k>0$. Since G has more edges than a forest can have, there exists an edge e on a cycle. Let $G'=(V,E\backslash\{e\})$. By induction $\mathrm{FES}(G')=k-1$. Adding the edge e to a witness edge set E' for G' shows that $\mathrm{FES}(G)=k$. □

Unlike the k-Feedback Vertex Set lower ideals, it is not obvious that the family k-Feedback Edge Set is a lower ideal in the minor order. However, with the above theorem one can easily prove this.

Corollary 33. *For each $k\geqslant 0$, the family of graphs k-*Feedback Edge Set *is a lower ideal in the minor order.*

Proof. We show that the three basic minor operations do not increase the number of edges required to remove all cycles of a graph. An isolated vertex deletion removes both a vertex and a component at the same time, so k is preserved in the formula $|E|=|V|-c+k$. For an edge deletion the number of components can increase by at most one, so with $|E|$ decreasing by one, the value of k does not increase. For an edge contraction, the number of vertices decreases by one, the number of edges decrease by at least one, and the number of components stays the same, so k does not increase. □

The above corollary allows us to characterize each k-Feedback Edge Set lower ideal in terms of obstruction sets. We abstractly characterize these below.

Theorem 34. *A connected graph G is an obstruction for k-*Feedback Edge Set *if and only if* $\mathrm{FES}(G)=k+1$ *and every edge contraction of G removes at least two edges* (*i.e., the open neighborhoods of adjacent vertices overlap*).

Proof. This follows from the fact that an edge contraction that does not remove at least two edges is the only basic minor operation that does not decrease the number of edges required to eliminate all cycles, for a connected graph with every edge on some cycle. □

The above theorem gives us a precise means of testing for minimal and non-minimal t-parses (see Step 1 of Section 3). Furthermore, in Section 5.3 below, we present a constructive method based on this theorem for generating all of the connected obstructions for k-Feedback Edge Set.

5.2. A complete testset (for FES*)*

Somewhat surprisingly, a usable testset for each feedback edge set family has already been presented in Section 4.2. We now prove that the previously given feedback vertex set tests can also be used here.

Lemma 35. *The testset T_t^k for the family k*-Feedback Vertex Set *is also a testset for k*-Feedback Edge Set.

Proof. See [10]. □

It is interesting to note from the above proof that, in addition to the out-of-family tests, the isolated triangles in the tests for k-Feedback Edge Set do not contain any boundary vertices. Thus, the number of graphs in a testset for k-Feedback Edge Set is substantially smaller than the order of the testset for k-Feedback Vertex Set.

5.3. Directly generating the k-Feedback Edge Set *obstructions*

A consequence of our direct characterization of graphs in k-Feedback Edge Set (recall Theorem 34) is the following constructive characterization of the minimal forbidden minors.

Lemma 36. *A graph G is a minor-order obstruction for some feedback edge family k*-Feedback Edge Set *if and only if the edges of G are defined by a set of non-identical K_3 cliques*

$$\{(a_1,b_1,c_1),(a_2,b_2,c_2),\ldots,(a_m,b_m,c_m)\}.$$

Proof. Let $G=\langle(a_1,b_1,c_1),(a_2,b_2,c_2),\ldots,(a_m,b_m,c_m)\rangle$ denote the well-defined graph $V(G)=\bigcup_{i=1}^{m}\{a_i,b_i,c_i\}$ and $E(G)=\{(u,v)\mid\{u,v\}\subseteq\{a_i,b_i,c_i\}$ for some $1\leqslant i\leqslant m\}$. Consider $k'=\mathrm{FES}(G)$. We show that G is an obstruction for k-Feedback Edge Set where $k=k'-1$. To do this we show that every one-step minor of G is a member of k-Feedback Edge Set. Since $\delta(G)\geqslant 2$ there are no isolated vertices to delete. Let $e=(u,v)$ be an edge of G and $G'=G\backslash\{e\}$ (edge deletion case). Since G does not contain any cut-edges, G' has the same number of components as G. So using Theorem 32 $\mathrm{FES}(G')=k$. If $G'=G/e$ (edge contraction case) then $|E(G')|\leqslant|E(G)|-2$, $|V(G')|=|V(G)|-1$, and thus $\mathrm{FES}(G')\leqslant k$. Therefore, G is an obstruction to k-Feedback Edge Set.

We now show that every obstruction G of k-Feedback Edge Set can be represented by a set of at most $k+1$ non-identical K_3 cliques. By Theorem 34 we know that every edge must lie on some K_3 clique. So it is clear that $m<|E(G)|$. Let $G_m=$

$\langle(a_1,b_1,c_1),(a_2,b_2,c_2),\ldots,(a_m,b_m,c_m)\rangle$ be a representation for G such that for each clique (a_i,b_i,c_i), $2\leqslant i\leqslant m$, at least one of the three edges (a_i,b_i), (b_i,c_i) and (a_i,c_i) is not present in the graph represented by

$$G_{i-1}=\langle(a_1,b_1,c_1),(a_2,b_2,c_2),\ldots,(a_{i-1},b_{i-1},c_{i-1})\rangle.$$

To complete the proof we show that $\text{FES}(G_i) > \text{FES}(G_{i-1})$ for all i. Using our characterization theorem of graphs in k-Feedback Edge Set, we can compute the change in $\text{FES}(G_i)$ from $\text{FES}(G_{i-1})$ by considering the following cases contributed by adding (a_i,b_i,c_i) to G_{i-1}:

# New vertices	# New edges	# New components	New FES
3	3	+1	+1
2	3	0	+1
1	3	$\{-1,0\}$	$\{+1,+2\}$
1	2	0	+1
0	3	$\{-2,-1,0\}$	$\{+1,+2,+3\}$
0	2	$\{-1,0\}$	$\{+1,+2\}$
0	1	0	+1

For example, if the clique (a_i,b_i,c_i) adds 1 vertex and 2 edges (which is incident to the new vertex) to the previous G_{i-1}, the number of components stays the same. Thus in this case by applying Theorem 32 we see that the minimum size of a feedback edge set must go up by 1. For all possible cases, the change in the size of the minimum feedback edges set is positive. Thus, since $\text{FES}(G_m)=k+1$ we have $m\leqslant k+1$. □

An explicit simple algorithm for computing all of the connected k-Feedback Edge Set obstructions is given in Fig. 6. For this procedure, we do not need to know the pathwidth or treewidth of the largest obstruction.

Corollary 37. *The algorithm given in Fig.* 6 *computes all the connected obstructions for the family k-*Feedback Edge Set.

Proof. We show that for any connected obstruction O we can construct it without adding an isolated triangles to a previous connected obstruction for k'-Feedback Edge Set, $k'<k$. Let $\{T_1,T_2,\ldots,T_m\}$ be a minimum set of covering triangles for O. Consider the graph G with vertices $\{T_1,T_2,\ldots,T_m\}$ and edges $\{(T_i,T_j)\mid T_i\cap T_j\neq\emptyset\}$. Since O is connected G is connected. Thus we can construct O without adding isolated triangles by using any breadth-first or depth-first spanning tree sequence of G. The proof of Lemma 36 gives us the upper bound of $k+1$ times through the outer most i loop. □

*5.4. The k-*Feedback Edge Set *obstructions*

Since the family k-Feedback Edge Set is contained in the family k-Feedback Vertex Set, the maximum treewidth of any obstruction for k-Feedback Edge Set is at most

```
function FindObstructions(integer $k$)
GraphSet obsts[1].add($K_3$);
for $i$ from 2 to $k+1$
    for each $G$ in obst[$i-1$] do
        Comment: add 0 vertices (1, 2 or 3 edges)
        for every triple of vertices $(u, v, w)$ of $G$ do
            Graph $H$ = new Graph($G$);
            $H$.addTriangle( $(u, v, w)$ );
            GraphSet obsts[$i$].add($H$);
        Comment: add 1 vertex (2 or 3 edges)
        for every pair of vertices $(u, v)$ of $G$ do
            Graph $H$ = new Graph($G$) $\cup \{x\}$;
            $H$.addTriangle( $(u, v, x)$ );
            GraphSet obsts[$i$].add($H$);
        Comment: add 2 vertices (3 edges)
        for every vertex $u$ of $G$ do
            Graph $H$ = new Graph($G$) $\cup \{x, y\}$;
            $H$.addTriangle( $(u, x, y)$ );
            GraphSet obsts[$i$].add($H$);
    next $G$;
next $i$;
$\mathcal{O}(\mathcal{F}) = \{G \in \text{obst}[i] \mid FES(G) = k+1 \text{ and } 1 \leq i \leq k+1\}$;
end
```

Fig. 6. An algorithm to generate the k-Feedback Edge Set obstructions.

$k_1 \backslash k_2$	0	1	2	3	4	5	6	
0	▨	▨	▨	▨	▨	▨	▨	
1	▨	▨	?					
2	▨							

Fig. 7. The characterized (k_1, k_2)-Feedback Vertex/Edge Set families.

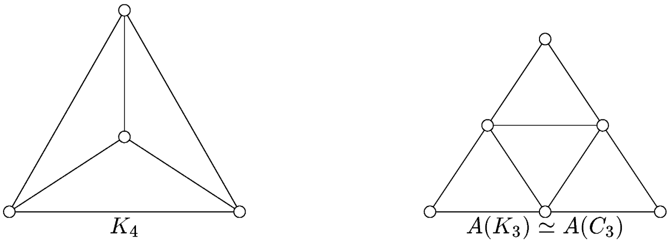


Fig. 8. Connected obstructions for 1-Feedback Vertex Set.

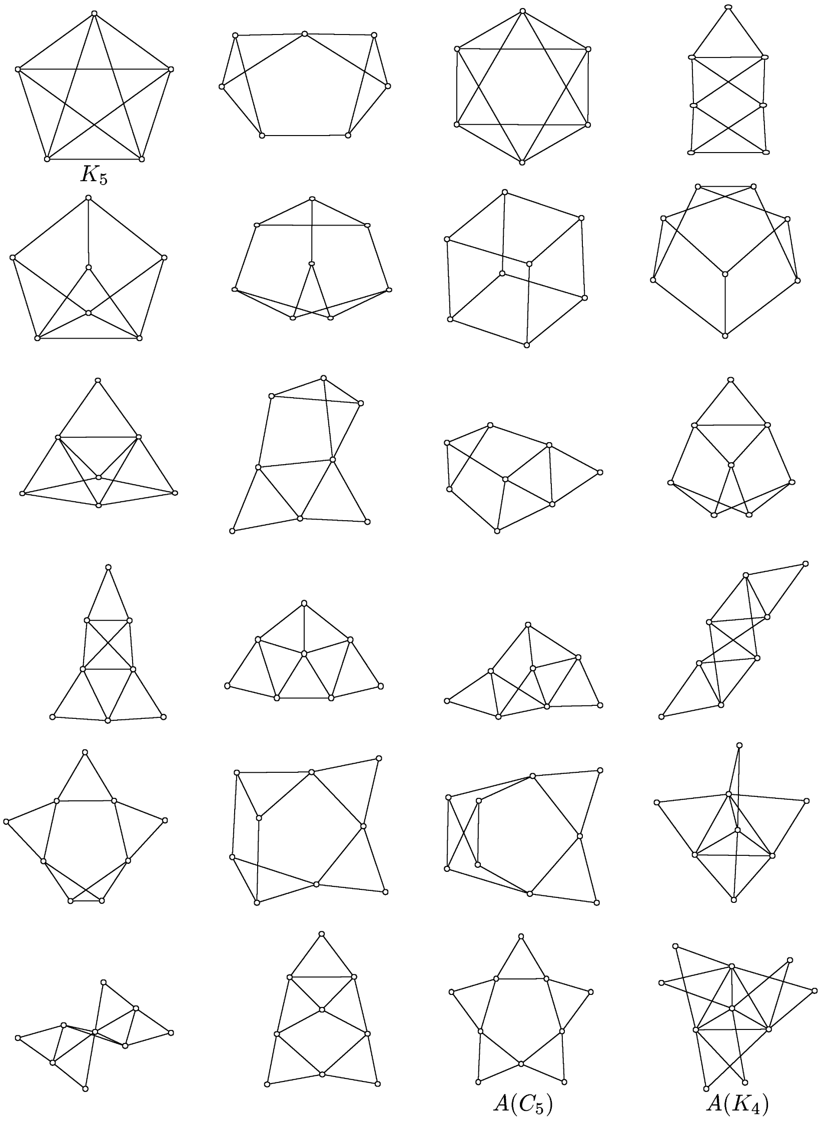


Fig. 9. Connected obstructions for 2-Feedback Vertex Set.

$k+2$. Also, the same arguments given in Section 4.3 regarding pathwidth apply to k-Feedback Edge Set as well.

For the family 0-Feedback Edge Set, it is trivial to show that K_3 is the only obstruction. The connected obstructions for the graph families 1-Feedback Edge Set through 3-Feedback Edge Set are shown in Figs. 10–12. There are well over 100 connected obstructions for the 4-Feedback Edge Set family. Any disconnected obstruction for k-Feedback Edge Set is easily determined by combining connected obstructions for j-Feedback Edge Set, $j<k$, since $\mathrm{FES}(G_1)+\mathrm{FES}(G_2)=\mathrm{FES}(G_1\cup G_2)$.

Table 3
The number of k-Feedback Edge Set obstructions for $k \leqslant 6$

k-FES	# connected obstructions	with $\delta(G) > 2$	and also biconnected
1	2	0	0
2	7	1	1
3	27	1	1
4	120	3	3
5	642	8	7
6	3767	24	21

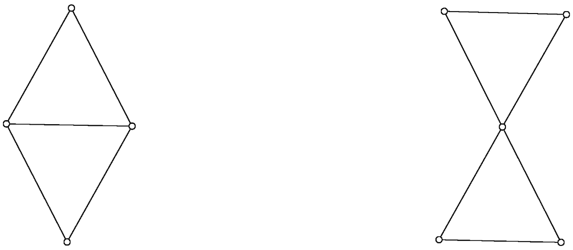


Fig. 10. Connected obstructions for 1-Feedback Edge Set.

Our constructive method shows that we can obtain all of the obstructions for k-Feedback Edge Set directly from the sets $\mathcal{O}(j$-Feedback Edge Set$)$, $j < k$. In fact most of the obstructions can be obtained from the immediately preceding obstruction set (i.e., with $j = k - 1$), by using the following observations.

Observation 38. If G is a connected obstruction for k-Feedback Edge Set then the following are all connected obstructions for $(k+1)$-Feedback Edge Set.

1. G with an added subdivided edge attached to an edge of G.
2. G with an attached K_3 on one of the vertices of G.
3. G with an added edge (u, v), whenever there exists a path of length at least two between u and v in $G \backslash E$ for each feedback edge set E of $k + 1$ edges.

It is easy to see that if an obstruction has a vertex of degree two then it is predictable by observations 1 and 2. The fourth (central) 2-Feedback Edge Set obstruction in Fig. 11 (wheel W_3) and the last 3-Feedback Edge Set obstruction in Fig. 12 (W_4) are two examples of graphs where observation 3 predicts the graph. Those 4-Feedback Edge Set, 5-Feedback Edge Set, and 6-Feedback Edge Set obstructions without vertices of degree two and cut-vertices are shown in Figs. 13–15. The third 4-Feedback Edge Set obstruction in Fig. 13 is not predictable from the 3-Feedback Edge Set obstructions by using any of the above observations. Here deleting any edge from this obstruction leaves a contractable edge that does not remove any cycles, that is, all single edge deleted

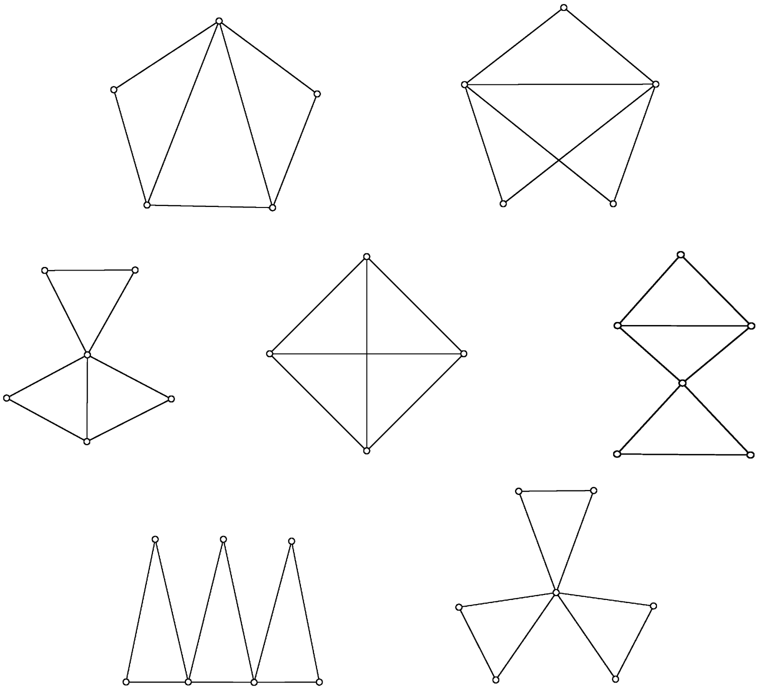


Fig. 11. Connected obstructions for 2-Feedback Edge Set.

minors are 'non-minimal' (see Theorem 34). This obstruction is easily constructed by 4 vertex triples, as promised by our direct enumeration algorithm.

Table 3 shows a summary of how many k-Feedback Edge Set obstructions there are for $k \leqslant 6$. The third column of the table gives the counts for the number of connected obstructions without vertices of degree 2. The fourth column is obtained from the third by subtracting the number of remaining obstructions with a cut-vertex. About 20 days of CPU time (using a single Sparc-20) was used to compute the 6-Feedback Edge Set obstructions.

6. Graphs with small hybrid feedback sets

In this penultimate section we generalize the two earlier feedback set families where we are allowed to cover cycles with both vertices and edges. First we need to prove that these hybrid feedback set families (i,j)-Feedback Vertex/Edge Set can be characterized by minors.

Lemma 39. *For any two non-negative integers i and j, the graph family* (i,j)-Feedback Vertex/Edge Set *is a lower ideal in the minor order.*

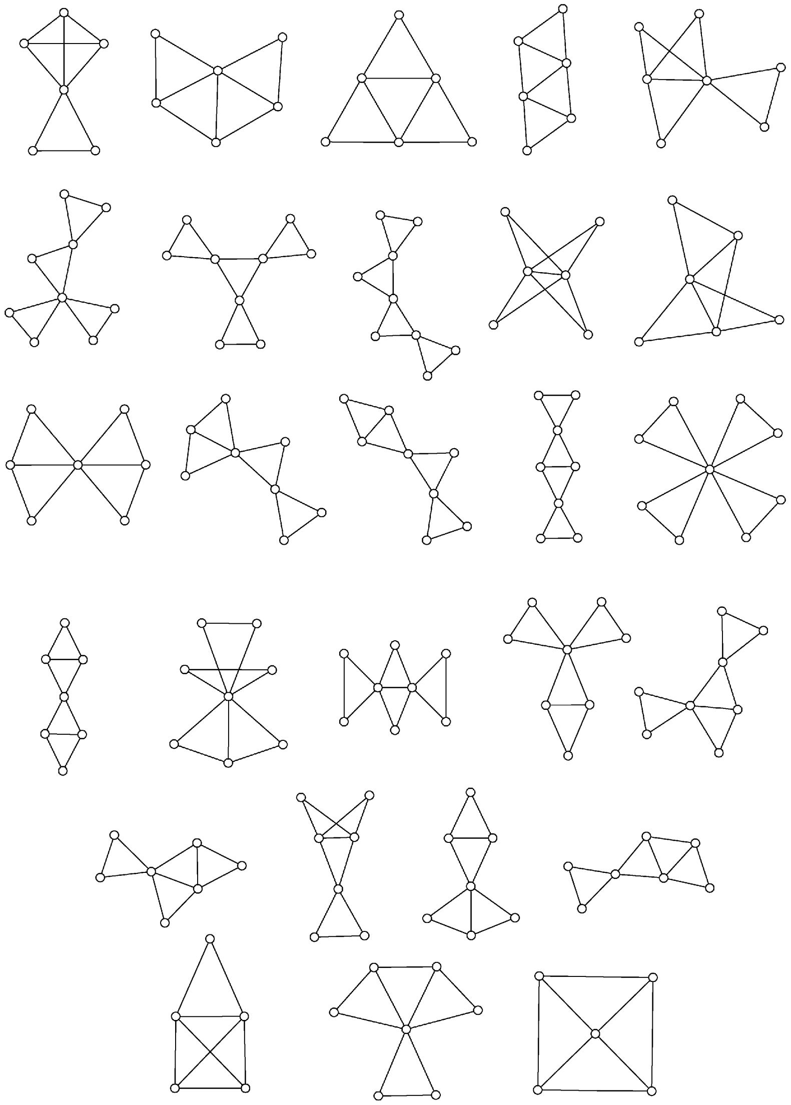


Fig. 12. Connected obstructions for 3-Feedback Edge Set.

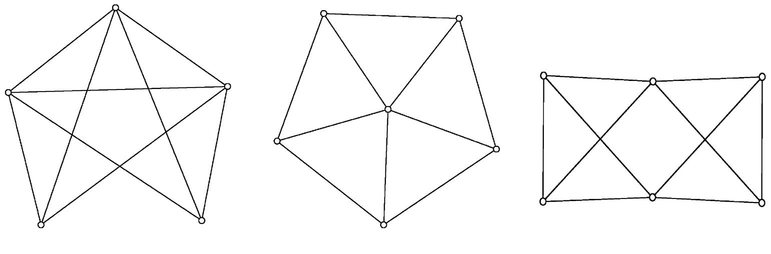


Fig. 13. Biconnected 4-Feedback Edge Set obstructions without degree 2 vertices.

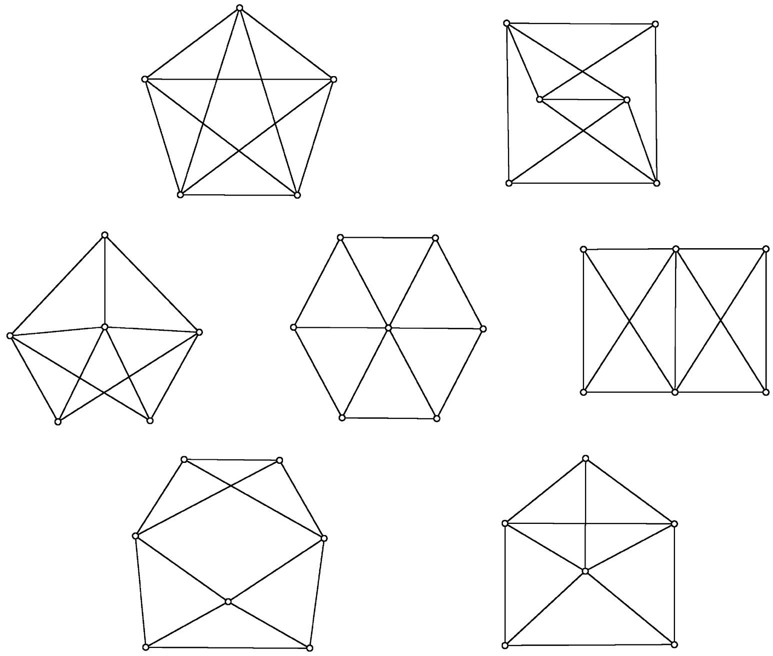


Fig. 14. Biconnected 5-Feedback Edge Set obstructions without degree 2 vertices.

Proof. Let G be a member of (i,j)-Feedback Vertex/Edge Set with a witness pair W_v (set of vertices) and W_e (set of edges) such that any edge in W_e is not incident to any vertex in W_v, $|W_v| \leqslant i$ and $|W_e| \leqslant j$. Consider a one-step minor $G' \leqslant_m G$. We have $G' \backslash W_v \leqslant_m G \backslash W_v$. So $G' \backslash W_v$ is a member of j-Feedback Edge Set. This implies that G' is a member of (i,j)-Feedback Vertex/Edge Set. □

A family congruence for (i,j)-Feedback Vertex/Edge Set over boundaried graphs can be defined by testsets that are very similar but not exactly the same to the $(i+j)$-Feedback Vertex Set and $(i+j)$-Feedback Edge Set testsets. Consider a test T of $T_t^{(i+j)}$ for $(i+j)$-Feedback Vertex Set, we need to distinguish between the vertex and edge witnesses. This can be accomplished by replacing a certain number of isolated K_3's in T with the graph $K_4^- =$ 'K_4 minus one edge' $=$ 'C_4 with a chord'.

Lemma 40. *We can modify the testset for* $(i+j)$-Feedback Vertex Set *to be a finite testset for* (i,j)-Feedback Vertex/Edge Set.

Proof. For $\mathscr{F} = (i,j)$-Feedback Vertex/Edge Set, let Z be an extension such that for two t-parse s G and H, $G \cdot Z \in \mathscr{F}$ and $H \cdot Z \notin \mathscr{F}$. We show how to construct a test T from Z with the structural properties of the $(i+j)$-Feedback Vertex Set tests, except that at most i isolated K_4^- may be present. Let $W_v \subseteq V(G \cdot Z)$ and $W_e \subseteq E(G \cdot Z)$ be

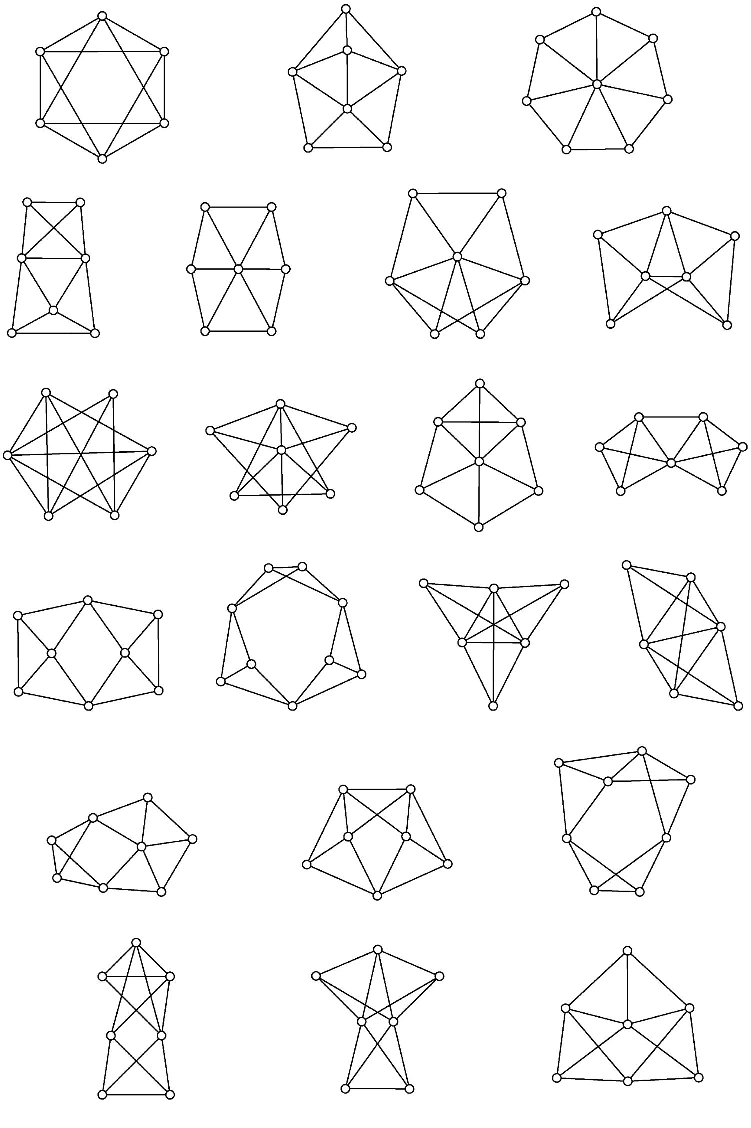


Fig. 15. Biconnected 6-Feedback Edge Set obstructions without degree 2 vertices.

a pair of witness sets such that

$$(G\cdot Z\backslash W_e)\backslash W_v \in \mathscr{F}$$

and $|W_v|\leqslant i$ and $|W_e|\leqslant j$. We create T from Z as follows: (1) replace every edge in $W_e \cap Z$ with an isolated K_3, (2) delete W_v from Z and add an isolated K_4^- while preserving boundary labels, and (3) reduce the resulting trees by the methods of creating a k-Feedback Vertex Set test (see Section 4.2). The graph $G\cdot T$ is a member of $\mathscr{F}$

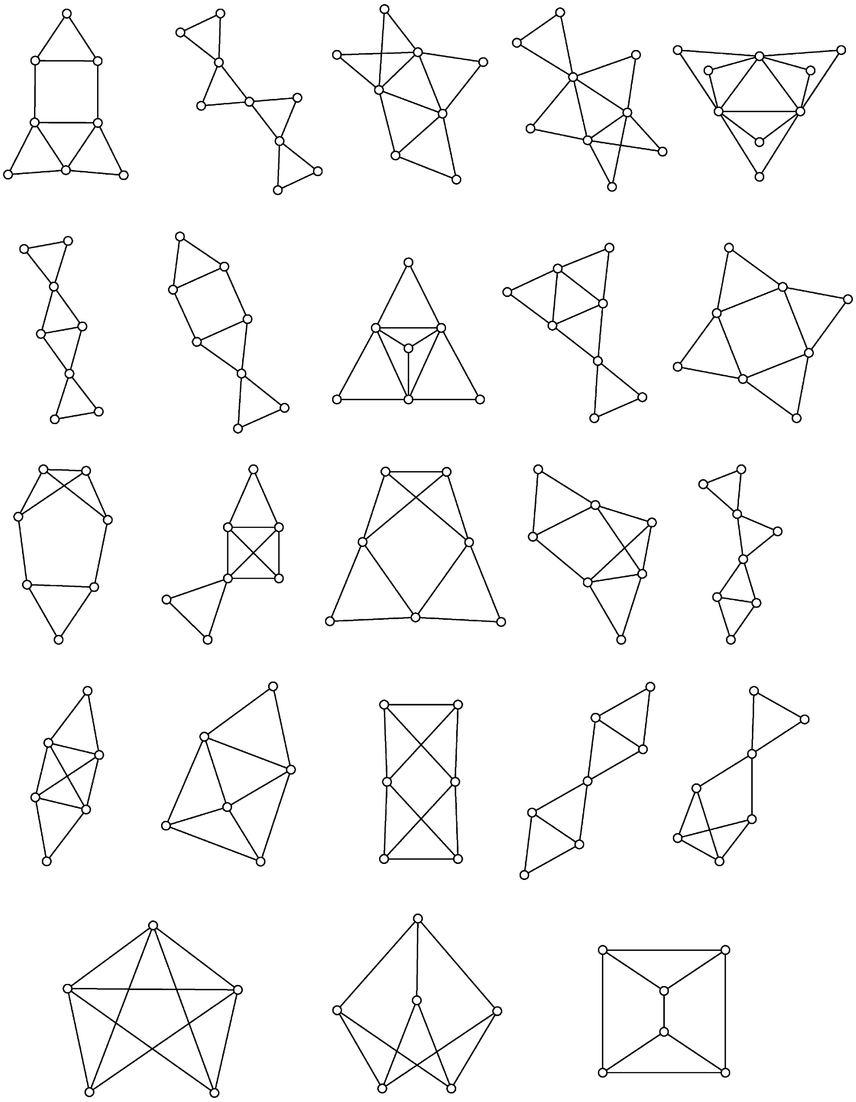


Fig. 16. Connected obstructions for (1, 1)-Feedback Vertex/Edge Set.

by the same witness pair (W_v, W_e). And the graph $H \cdot T$ is not a member of $\mathscr{F}$ or otherwise we could have found a witness pair for $H \cdot Z$.

6.1. *The* (1, 1)-Feedback Vertex/Edge Set *obstructions*

In Fig. 7 we show (by shaded boxes) the 10 families of graphs based on small feedback sets that we have characterized in this paper. We characterized the hybrid graph family (1, 1)-Feedback Vertex/Edge Set by the methods of our search theory

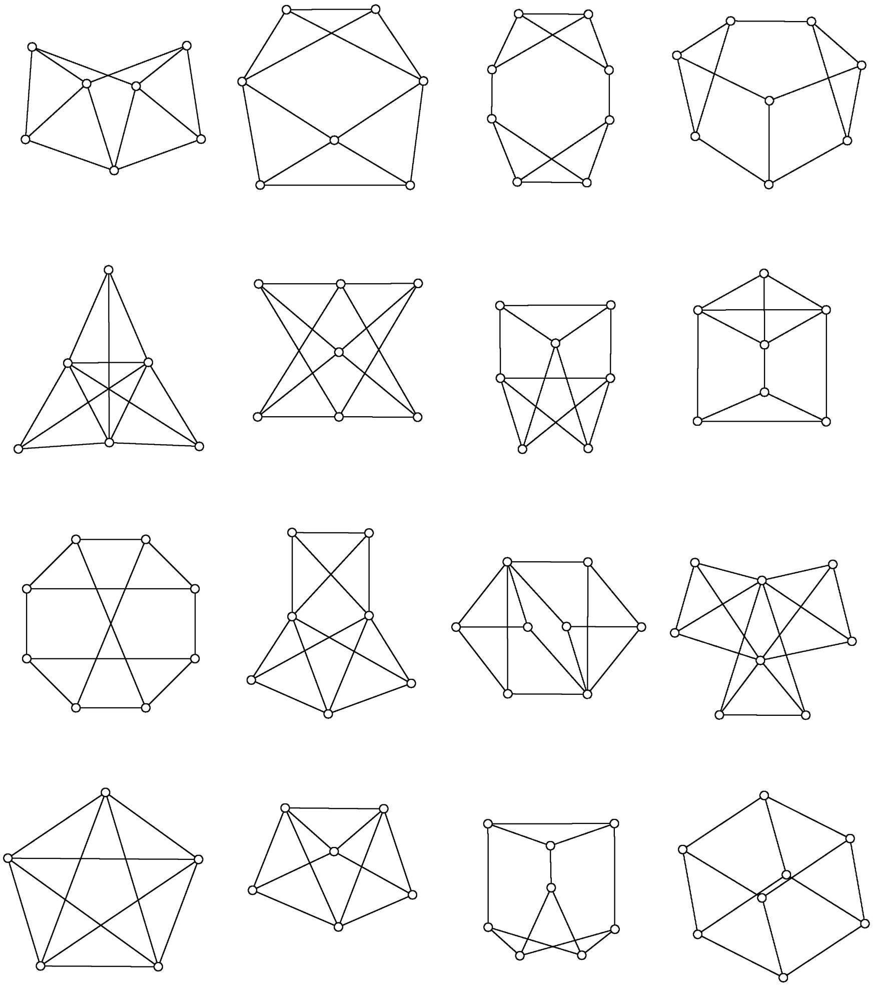


Fig. 17. Known biconnected $(1,2)$-Feedback Vertex/Edge Set obstructions without degree 2 vertices.

of Section 3. To do so, we needed the following pathwidth bound on the obstructions of $(1,1)$-Feedback Vertex/Edge Set.

Lemma 41. *If G is an obstruction to* $(1,1)$-Feedback Vertex/Edge Set *then the pathwidth of G is at most* 3.

Proof. The proof is similar to our pathwidth 4 bound for the obstructions of 2-Feedback Vertex/Edge Set. See [10]. □

In Fig. 16 we show the 23 connected obstructions to the $(1,1)$-Feedback Vertex/Edge Set family of graphs. There are also three disconnected obstructions: $3\cdot K_3$, $K_3\cup K_4$, and $2\cdot K_4^-$.

We have not completely classified the next larger family $(1,2)$-Feedback Vertex/Edge Set. We display a partial list of the connected obstructions with no degree two vertices in Fig. 17. We currently have found 246 connected obstructions. We conjecture that pathwidth 4 bounds the width of the largest obstruction in $\mathcal{O}((1,2)$-Feedback Vertex/Edge Set).

7. Conclusion and open problems

The main results of this paper present a practical version of the general obstruction set computation methods first described by Fellows and Langston [16]. The theory that we describe here and have implemented has led to the successful computation of a number of non-trivial and previously unknown obstruction sets, including k-vertex cover for $k \leqslant 5$ [7] and, as we describe here, for small values of k-Feedback Vertex Set and k-Feedback Edge Set and related properties.

An important part of our results on these particular problems consists of efficient congruences for the properties, based on test sets, that could be also be used in a direct algorithmic approach to recognizing these properties for graphs of bounded pathwidth (treewidth) that are given with a path- (tree-) decomposition.

The main open problem is whether the general machinery developed here can be successfully applied to compute more interesting unknown obstruction sets, such as the obstruction set for torus embedding that has resisted substantial amounts of traditional human effort of graph theorists for over two decades. One of the main obstacles in attacking the torus obstruction set with these methods has been the lack of a good bound on the maximum pathwidth of the torus obstructions. (A congruence based on test sets is known for this problem.)

In recent work we have found a way to adapt the machinery described in this paper so that obstruction sets can be computed without knowing in advance a bound on maximum obstruction pathwidth. There is a way of automatically detecting that a width is sufficient to capture all of the obstructions. These results are described elsewhere [8] and have not yet been implemented.

Acknowledgements

We thank the two anonymous referees for their detailed and helpful comments on an earlier version of this paper.

References

[1] S. Arnborg, J. Lagergren, D. Seese, Easy problems for tree-decomposable graphs, J. Algorithms 12 (1991) 308–340.

[2] S. Arnborg, A. Proskurowski, D.G. Corneil, Minimal forbidden minor characterization of a class of graphs, Colloq. Math. Soc. János Bolyai 52 (1987) 49–62.

[3] D. Archdeacon, S. Negami, unpublished manuscript.

[4] H.L. Bodlaender, On linear time minor tests and depth first search, J. Algorithms 14 (1993) 1–23.

[5] H.L. Bodlaender, A linear time algorithm for finding tree-decompositions of small treewidth, SIAM J. Comput. 25 (1996) 1305–1317.

[6] R.B. Borie, Recursively constructed graph families: membership and linear algorithms, Ph.D. Thesis, Georgia Institute of Technology, School of Information and Computer Science, 1988.

[7] K. Cattell, M.J. Dinneen, A characterization of graphs with vertex cover up to five, in: V. Bouchitte, M. Morvan (Eds.), Orders, Algorithms and Applications, ORDAL'94, Lecture Notes on Computer Science, Vol. 831, Springer, Berlin, 1994, pp. 86–99.

[8] K. Cattell, M.J. Dinneen, R.G. Downey, M.R. Fellows, M.A. Langston, Computing graph minor obstructions sets, Theoret. Comput. Sci. 233 (2000) 107–127.

[9] K. Cattell, M.J. Dinneen, M.R. Fellows, Obstructions to within a few vertices or edges of acyclic, in: Proceedings of the Fourth Workshop on Algorithms and Data Structures, WADS '95, Lecture Notes on Computer Science, Vol. 955, Springer, Berlin, 1995, pp. 415–427.

[10] M.J. Dinneen, K. Cattell, M.R. Fellows, Forbidden minors to graphs with small feedback sets, Report CDMTCS-019, Centre for Discrete Mathematics and Theoretical Computer Science, University of Auckland, New Zealand, October 1996, see `http://www.cs.auckland.ac.nz/CDMTCS/researchreports/019mjd.pdf`.

[11] S.R. Coorg, C.P. Rangan, Feedback vertex set on cocomparability graphs, Networks 26 (1995) 101–111.

[12] B. Courcelle, The monadic second-order logic of graphs III: treewidth, forbidden minors and complexity issues, Inform. Théor. 26 (1993) 257–286.

[13] M.J. Dinneen, Bounded combinatorial width and forbidden substructures, Ph.D. Dissertation, Department of Computer Science, University of Victoria, P.O. Box 3055, Victoria, BC, Canada V8W 3P6, December 1995.

[14] H. Didjev, J. Reif, An efficient algorithm for the genus problem with explicit construction of forbidden subgraphs, in: IEEE Symposium on Foundations of Computer Science Proceedings, Vol. 31, 1991, pp. 337–347.

[15] R. Downey, M.R. Fellows, Parameterized computational feasibility, in: P. Clote, J. Remmel (Eds.), Feasible Mathematics II, Proceedings of the Second Cornell Workshop on Feasible Mathematics, Birkhauser, Basel, 1995, pp. 219–244.

[16] M.R. Fellows, M.A. Langston, An analogue of the Myhill-Nerode Theorem and its use in computing finite-basis characterizations, in: Proceedings of the IEEE Symposium on Foundations of Computer Science, Vol. 30, 1989, pp. 520–525.

[17] M. Funke, G. Reinelt, A polyhedral approach to the feedback vertex set problem, in: Integer Programming and Combinatorial Optimization, Vancouver, BC, Lecture Notes in Computer Science, Vol. 1084, Springer, Berlin, 1996, pp. 445–459.

[18] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$-Completeness, Freeman, New York, 1979.

[19] H.H. Glover, J.P. Huneke, There are finitely many Kuratowski graphs for the projective plane, in: Graph Theory and Related Topics, Academic Press, New York, 1979, pp. 201–206.

[20] P. Hliněný, $K_{4,4}-K_2$ has no finite planar cover. KAM Series — Discrete Mathematics — Combinatorics — Operations Research — Optimization, No. 96–312, 1996 (also a Charles University technical report).

[21] J. Philip Huneke, Personal communication.

[22] A. Itai, M. Rodeh, Finding a minimum circuit in a graph, in: Nineth Proceedings of the ACM Symposium on the Theory of Computing, 1977, pp. 1–10.

[23] N.G. Kinnersley, Obstruction set isolation for layout permutation problems, Ph.D. Thesis, Department of Computer Science, Washington State University, Pullman, WA 99164, 1989.

[24] N.G. Kinnersley, M.A. Langston, Obstruction set isolation for the Gate Matrix Layout problem, Discrete Appl. Math. 54 (1994) 169–213.

[25] K. Kuratowski, Sur le problè des courbes gauches en topologie, Fund. Math. 15 (1930) 271–283.

[26] J. Lagergren, S. Arnborg, Finding minimal forbidden minors using a finite congruence, in: Proceedings of the 19th International Colloquium on Automata, Languages and Programming, Lecture Notes on Computer Science, Vol. 510, Springer, Berlin, 1991, pp. 533–543.

[27] M.A. Langston, S. Ramachandramurthi, Dense layouts for series-parallel circuits, Technical Report, Department of Computer Science, University of Tennessee, Knoxville, TN 37996–1301, 1992.

[28] E.L. Lloyd, M.L. Soffa, C.-C. Wang, On locating minimum feedback vertex sets, J. Comput. System Sci. 37 (3) (1988) 292–311.

[29] H. Nakano, Y. Nakanishi, Structure of solution space in discrete optimization problem and its application to minimum feedback edge set problem, Technical Report, Osaka University, Vol. 25, No. 1254–1275, 1975, pp. 287–299.

[30] S. Negami, The spherical genus and virtually planer graphs, Discrete Math. 70 (1988) 159–168.

[31] S. Negami, Graphs which have no finite planar covering, Bull. Inst. Math., Academia Sinica 16 (4) (1988) 377–384.

[32] E.T. Neufeld, Practical toroidality testing, Master's Thesis, Department of Computer Science, University of Victoria, P.O. Box 3055, Victoria, BC, Canada V8W 3P6, 1993.

[33] N. Robertson, P.D. Seymour, Graph minors — a survey, in: Surveys in Combinatorics, Vol. 103, Cambridge University Press, Cambridge, 1985, pp. 153–171.

[34] P.D. Seymour, A bound on the excluded minors for a surface, Bell Communications Research, Morristown, New Jersey, February 1993.

[35] J.V. Leeuwen, Handbook of Theoretical Computer Science: A: Algorithms and Complexity Theory, MIT Press/North Holland Publishing Company, Amsterdam, 1990.