Feedback vertex set and longest induced path on AT-free graphs

Dieter Kratsch[1], Haiko Müller[2]*, and Ioan Todinca[3]

[1] LITA, Université de Metz, 57045 Metz Cedex 01, France
[2] School of Computing, University of Leeds, Leeds, LS2 9JT, United Kingdom
[3] LIFO, Université d'Orléans, BP 6759, 45067 Orléans Cedex 2, France

Abstract. We present a polynomial time algorithm to compute a minimum (weight) feedback vertex set for AT-free graphs, and extending this approach we obtain a polynomial time algorithm for graphs of bounded asteroidal number.
We also present an $O(nm^2)$ algorithm to compute a longest induced path in AT-free graphs.

1 Introduction

A *feedback vertex set* (fvs) of an undirected graph $G = (V, E)$ is a set W containing at least one vertex of each cycle of G. Hence $W \subseteq V$ is a feedback vertex set of $G = (V, E)$ if and only if $G - W$ is a forest. The FEEDBACK VERTEX SET (FVS) problem is, given a graph G and an integer k, whether G contains a feedback vertex set of size at most k.

An *induced path* $P = (W, F)$ of a graph G is a tree $G[W] = P$ of maximum degree two. The LONGEST INDUCED PATH (LIP) problem asks whether the input graph contains an induced path of given length.

Both FVS and LIP are NP-complete [11]. Both problems are definable in a logical language called LinEMSOL($\tau_{1,L}$). Thus their weighted versions are linear time solvable on graphs of bounded clique-width [9]. Hence both problems are linear-time solvable for graphs of bounded treewidth, cographs, distance-hereditary graphs, etc.

There are very few papers on algorithms for LIP. A very interesting one by Gavril studies the weighted version of LIP and obtains polynomial time algorithms for various graph classes among them k-bounded hole families of graphs, interval-filament graphs, polygon-circle graphs, circle graphs and circular-arc graphs [12].

On the contrary, there is a lot of research on algorithms for FVS. The problem is fixed-parameter tractable [10]. There has been a series of papers on approximation algorithms on minimum (weight) FVS resulting in a 2-approximation algorithm for minimum weight FVS [1]. Clearly FVS had been and still is a benchmark covering problem. It is also important since FVS approximation algorithms are used as subroutines in approximation algorithms for other typically network analysis problems. See e.g. [14] where a PTAS for FVS on planar graphs is constructed as a by-product.

Concerning the complexity of FVS on special graph classes the following is known. FVS is NP-complete for bipartite and for planar graphs. A minimum (weight) fvs can

* hm@comp.leeds.ac.uk

be computed by a polynomial time algorithm for interval graphs ($O(n + m)$) [18], permutation graphs ($O(nm)$) [16], cocomparability graphs ($O(n^2m)$) [17] and convex bipartite graphs ($O(n^2m)$) [17]. Furthermore FVS is mentioned as polynomial time solvable for chordal graphs and circular-arc graphs in [19]. J. Spinrad points out in [19] that the complexity of FVS on AT-free graphs in unknown.

Three independent vertices form an *asteroidal triple* (AT for short) if any two of them are connected by a path avoiding the neighbourhood of the third vertex. Graphs without asteroidal triples are called *AT-free*. AT-free graphs contain cocomparability graphs, permutation graphs and interval graphs. In an important paper Corneil, Olariu and Stewart were the first to study structural properties of AT-free graphs [8]. In the last ten years various papers on the complexity of NP-complete problems for AT-free graphs have been published (see e.g. [5–7, 15]). Inspecting their results one finds out that there are LinEMSOL($\tau_{1,L}$)-definable problems that remain NP-complete on AT-free graphs. CLIQUE is NP-complete on AT-free graphs [6] and MINIMUM DOMINATING CLIQUE remains NP-complete even on cocomparability graphs [3]. This provides another motivation to study FVS on AT-free graphs.

Our paper is organised as follows. In Section 2 we summarise the main ingredients of an approach developed in [6] to design polynomial time algorithms for problems on AT-free graphs (see also [5, 7]). In Section 3 we demonstrate the approach for LIP on AT-free graphs (which is much easier than for FVS) and obtain an $\mathcal{O}(nm^2)$ time algorithm. In Section 4 we present a polynomial time algorithm for FVS on AT-free graphs. In Section 5 we show how to extend the approach to graphs of bounded asteroidal number and we obtain a polynomial time algorithm for FVS.

2 Preliminaries

We use standard graph theory notation throughout. For nonadjacent vertices u and v we denote by $C_G(u, v)$ the connected component of $G - N[v]$ containing u. As usual we omit the index if this does not cause confusion. By a connected component we mean a maximal connected induced subgraph of a graph. Thus we denote a connected component of a graph G with vertex set C by $G[C]$. Note that $w \in C(u, v)$ if and only if $u \in C(w, v)$. Using this notation, the vertices u, v and w form an AT if and only if $u \in C(v, w)$, $v \in C(w, u)$ and $w \in C(u, v)$.

2.1 Blocks and Intervals

Let $G = (V, E)$ be a connected graph and $v \in V$. The pair (v, C) is called a *block* of G if $G[C]$ is a connected component of $G - N[v]$. Thus removing a closed neighbourhood $N[v]$ for some vertex v from a graph G this graph decomposes into the connected components of $G - N[v]$ and it also decomposes into blocks.

Let u and v be two nonadjacent vertices of G and let $B = C(v, w) \cap C(w, v)$. The triple (v, B, u) is called an *interval* of G. Obviously (v, B, u) is an interval if and only if (u, B, v) is an interval. Notice that B may be empty.

Removing a closed neighbourhood from a block decomposes it into blocks and exactly one interval. Formally, consider a block (v, C) and let $u \in C$. Removing $N[u]$

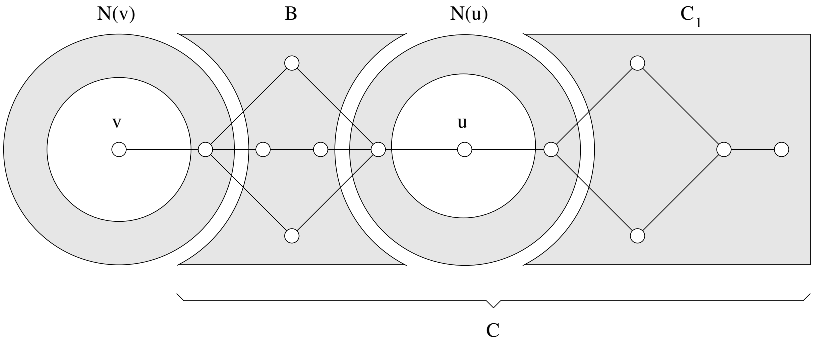

Fig. 1. Blocks and intervals

from $G[C]$ partitions C into $B, C_1, \ldots, C_p$, where (v, B, u) is an interval and (u, C_i) are blocks of G (see Figure 1).

The LIP and FVS problems are solved by dynamic programming on blocks.

2.2 Representing Neighbourhoods

Definition 1. *Consider a vertex v of the graph G and let S be an independent set contained in $N(v)$. We say that a set $R \subseteq S$ is a* representation of S w.r.t. v *if $N(S) \setminus N(v) = N(R) \setminus N(v)$ and R is of minimum size for this property.*

The following lemma shows that in AT-free graphs, each representation of an independent set $S \subseteq N(v)$ contains at most four vertices.

Lemma 1. *Let $G[C]$ be a connected component of $G - N[v]$ where $G = (V, E)$ is an AT-free graph, $v \in V$ and $C \subset V$. The neighbourhood of an independent set $S \subseteq N(v)$ in C can be represented by a subset of size at most two,* i.e. *there is a set $R_C \subseteq S$ with $|R_C| \leqslant 2$ and $N(R_C) \cap C = N(S) \cap C$. The outside neighbourhood of S can be represented by a subset of size at most four,* i.e. *there is a set $R \subseteq S$ with $|R| \leqslant 4$ and $N(R) \setminus N(v) = N(S) \setminus N(v)$.*

Proof. We consider three pairwise nonadjacent vertices $r, s, t \in N(v)$ with private neighbours $a, b, c \in C$. That is, $a \in (C \cap N(r)) \setminus (N(s) \cup N(t))$, $b \in (C \cap N(s)) \setminus (N(r) \cup N(t))$ and $c \in (C \cap N(t)) \setminus (N(r) \cup N(s))$. If a, b and c induce a triangle in G then r, s and t form an AT of G—a contradiction. Otherwise let $ab \notin E$. Now a, b and v form an AT of G because $G[C]$ is connected—a contradiction again. Hence an independent triple of vertices in $N(v)$ cannot have private neighbours in the same component of $G - N[v]$.

Next let a, b and c be vertices in three different connected components of $G - N[v]$ having private neighbours $r, s, t \in S$. That is $r \in (S \cap N(a)) \setminus (N(b) \cup N(c))$, $s \in (S \cap N(b)) \setminus (N(a) \cup N(c))$ and $t \in (S \cap N(c)) \setminus (N(a) \cup N(b))$. Clearly $\{a, b, c\}$ is an independent set of G and hence these vertices form an AT of G. Since G is AT-free at most two components of $G - N[v]$ contain vertices having private neighbours in $N(v)$. $\square$

3 Longest Induced Path

3.1 Analysis

Lemma 2. *Let $P = (W, F)$ be an induced path of an AT-free graph G. Then for every vertex $v \in W$ and every component $G[C]$ of $G - N[v]$ the graph $P[C]$ is connected.*

Proof. Conversely, let (t, u, v, w, x) be a subpath of P with $t, x \in C$. Since P is induced the vertices t and x are non-adjacent. Since $G[C]$ is connected there is a path in $G[C]$ joining t and x. This is, t, v and x form an AT of G—a contradiction. □

Thus any longest induced path passing through v will enter at most two blocks (v, C) and (v, C'), each of them at most once. Let $\ell(u, v, w, x)$ denote the length of a longest induced path $(u, v, w, x, \dots)$ in $G[N[v] \cup C(x, v)]$. These values represent partial solutions for the block $(v, C(x, v))$ of G. Note that we store several partial solutions for each block (v, C), namely the longest extensions of all possible paths (u, v, w).

By $\mathcal{P}_l(G)$ we denote the set of induced paths of length l in G and define

$$\mathrm{lip}(G) = \max\{l : \mathcal{P}_l(G) \neq \varnothing\} .$$

If $G = (V, E)$ contains at least one path of length three the values $\ell(u, v, w, x)$ fulfil the following recursion for all $(u, v, w, x) \in \mathcal{P}_3(G)$.

$$\ell(u,v,w,x) = \begin{cases} 3 & \text{if there is no } y \in C(x,v) \text{ such that } (u,v,w,x,y) \in \mathcal{P}_4(G) \\ 1 + \max\{\ell(v,w,x,y) : & \\ \quad y \in C(x,v), u \notin C(y,w), (u,v,w,x,y) \in \mathcal{P}_4(G)\} & \text{otherwise} \end{cases} \tag{1}$$

$$\mathrm{lip}(G) = \max\{\ell(u,v,w,x) : (u,v,w,x) \in \mathcal{P}_3(G)\} \tag{2}$$

3.2 Algorithm

The above recurrence translates to the following recursive algorithm on $G = (V, E)$.

step 1 Compute $\mathcal{P}_3(G)$. If $\mathcal{P}_3(G) = \varnothing$ then if $E = \varnothing$ then $\mathrm{lip}(G) = 0$ else if each component of G is complete then $\mathrm{lip}(G) = 1$ else $\mathrm{lip}(G) = 2$.
step 2 Create a list of all blocks (v, C) of G and sort by $|C|$.
step 3 For each block (v, C) and each $(u, v, w, x) \in \mathcal{P}_3(G)$ with $x \in C$ compute $\ell(u, v, w, x)$ according to Equation (1) using the values $\ell(v, w, x, y)$ computed earlier in this step.
step 4 Determine $\mathrm{lip}(G)$ by Equation (2) using the values $\ell(u, v, w, x)$ computed in step 3.

Theorem 1. *There is an $\mathcal{O}(nm^2)$ algorithm to compute the maximum length of an induced path in AT-free graphs.*

Proof. First we show that our algorithm indeed computes $\text{lip}(G)$for every graph $G = (V, E)$. This is obviously true if $\mathcal{P}_3(G) = \varnothing$. We can modify our algorithm such that it stores a path $(u, v, w, x, \dots)$ of length $\ell(u, v, w, x)$ for each such value computed. If one of these paths is not an induced one it contains a subpath corresponding to an induced cycle. The shortest such cycle has length at most 5 because longer induced cycles contain ATs. But in Equation (1) we check that every sequence of five consecutive vertices induces a path and not a cycle. Hence G contains an induced path of length l if our algorithm computes $\text{lip}(G) = l$.

Now let (t, u, v, w, x) be a subpath of any induced path in G. By Lemma 2 the vertices t and x belong to different connected components of $G - N[v]$. In other words, our algorithm detects a longest induced path in G.

It remains to bound the running time. As usual let $n = |V|$ and $m = |E|$.

step 1 For each pair of edges $vw, xy \in E$ we check whether $(v, w, x, y) \in \mathcal{P}_3(G)$. This can be done in $O(m^2)$ time.

step 2 The number of blocks is $\mathcal{O}(n^2)$ and for fixed v we find all blocks (v, C) in time $\mathcal{O}(n + m)$. Hence this step needs time $\mathcal{O}(nm + n^2)$.

step 3 Here we compute $\mathcal{O}(m^2)$ values $\ell(u, v, w, x)$. For each one we minimise over $\mathcal{O}(n)$ vertices y. Consequently this step takes $\mathcal{O}(nm^2)$ time.

step 4 In the last step we minimise over all $(u, v, w, x) \in \mathcal{P}_3(G)$ in $\mathcal{O}(m^2)$ time.

Hence the overall running time can be bounded by $\mathcal{O}(nm^2)$. □

Gavril [12] considers maximum weighted induced paths in graphs in so-called k-bounded hole families (k-chordal graphs in [2]). For $k = 5$ (AT-free graphs are a 5-bounded hole family) the running time of his algorithm can be bounded by $\mathcal{O}(nm^3)$. Our algorithm can be modified to run on vertex weighted graphs without slowdown. Thus a maximum weight induced path on AT-free graphs can be computed by an $\mathcal{O}(nm^2)$ time algorithm.

4 Feedback Vertex Set

4.1 Analysis

Let

$$\text{fvs}(G) = \min\{|W| : G - W \text{ is a forest}\} \tag{3}$$

and let K be a set of vertices inducing a forest in the graph G. For a fixed vertex $v \in K$, denote by L the set of leaves of $G[K]$ contained in $N(v)$ and let $R' = K \cap N(v) \setminus L$. Note that in the forest $G[K]$ every vertex in L is adjacent to v only and every vertex in R' is adjacent to v and at least one vertex outside $N[v]$. Let R be a representation of L w.r.t. v. The triple (v, R, R') we call a *local map* of K. In the example of Figure 2, the forest K is represented by a bold line. The triple $(v, \{b, d\}, \{g\})$ is a local map of K.

Lemma 3. *Let (v, R, R') be a local map of $K \ni v$ inducing a forest $G[K]$ in a graph G that might as well contain ATs. The following conditions hold:*

1. $R \cup R' \subseteq N(v)$*;*

2. *$R \cup R'$ is an independent set of G;*
3. *$N(x) \setminus N(v) \subset N(x') \setminus N(v)$ implies $x \in R$ and $x' \in R'$ for all $x, x' \in R \cup R'$;*
4. *If G is AT-free then $|R \cup R'| \leqslant 4$.*

Proof. Condition 1 follows directly from the definition.

Conversely, if $G[R \cup R']$ contains an edge xy, then x, y and v would form a cycle in $G[K]$. This implies Condition 2.

For Condition 3, firstly notice that if $x, x' \in R$ and $N(x) \setminus N(v) \subset N(x') \setminus N(v)$ then $N(R \setminus \{x\}) \setminus N(v) = N(R) \setminus N(v)$, contradicting the fact that R is a representation of $(K \cap N(v)) \setminus R'$ w.r.t. v. Suppose now that $x \in R'$ and $x \in R \cup R'$. Let z be a neighbour of x in $K \setminus N[v]$. Then z is adjacent to x' too, so v, x, z and x' induce a cycle in $G[K]$. Condition 3 follows.

Finally we consider Condition 4. Lemma 1 implies $|R| \leqslant 4$ and $|R'| \leqslant 2$, hence $|R \cup R'| \leqslant 6$. The proof of $|R \cup R'| \leqslant 4$ is omitted here due to space restrictions. □

A triple (v, R, R') is called *local map* of G if it satisfies the conditions of Lemma 3. The set of all local maps of G is denoted by $\mathcal{R}(G)$ or simply $\mathcal{R}$.

Definition 2. *Let (v, R, R') be a local map of $G = (V, E)$. We say that $K \subseteq V$ is* compatible *with (v, R, R') if*

- *$G[K]$ is a forest;*
- *$\{v\} \cup R \cup R' \subseteq K$;*
- *Every internal vertex $x \in N(v)$ of $G[K]$ belongs to R';*
- *R is a representation of $(K \cap N(v)) \setminus R'$, w.r.t. v.*

Let $\mathcal{K}(v, R, R')$ denote the set of all sets K compatible with (v, R, R').

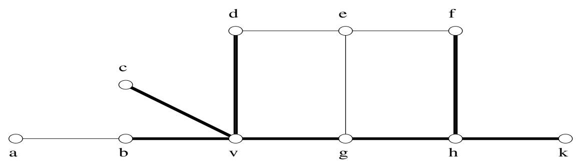

Fig. 2. Local maps

In particular, K must contain the star induced by v, R and R', and the only vertices of $K \cap N(v)$ which may have a neighbour in $K \setminus N[v]$ are the elements of R'. We point out that the vertices of R' are not required to have neighbours in $K \setminus N[v]$, they are just the only ones that *may* have neighbours in $K \setminus N[v]$. In Figure 2, the forest K is compatible with both $(v, \{b, d\}, \{g\})$ and $(v, \{d\}, \{b, g\})$. The former is a local map of the forest $K \cup \{a\}$. In other words, every induced forest containing the vertex v is compatible with its local map (v, R, R'), but some vertices in R' might be leaves of another forest compatible with (v, R, R').

Our algorithm will compute, for each local map (v, R, R') and for each block (v, C), the largest forest contained in $N[v] \cup C$, compatible with (v, R, R'). The following lemma shows in particular that it suffices to compute forests $K_1, K_2 \in \mathcal{K}(v, R, R')$ such that $K_1 \cap N(v)$ and $K_2 \cap C$ be of maximum size.

Lemma 4. *Let $K_1, K_2 \subseteq V$ be compatible with $(v, R, R') \in \mathcal{R}(G)$. For any block (v, C) of G, $K_3 = (K_1 \setminus C) \cup (K_2 \cap C)$ is compatible with (v, R, R').*

Proof. We only show that K_3 induces a forest in G because the other condition is obviously fulfilled. Conversely let $Z \subseteq K_3$ induce a shortest cycle in $G[K_3]$. Clearly $Z \cap C \neq \varnothing$ and $Z \cap N[v] \neq \varnothing$. Let $(a, x_1, \ldots, x_p, b)$ be a subpath of Z such that $x_i \in C, \forall i, 1 \leqslant i \leqslant p$ and $a, b \in N(v)$ (possibly $a = b$). Since $a, b \in K_1 \cap C$, there is a vertex a' (resp. b') of $R \cup R'$ such that $x_1 \in N(a')$ (resp. $x_p \in N(b')$). If $a' = b'$, then $(a', x_1, \ldots, x_p)$ is a cycle of G, else $(v, a', x_1, \ldots, x_p, b')$ is a cycle of G, contradicting the fact that $v, a', b', x_1, \ldots, x_p \in K_2$ and $G[K_2]$ is acyclic. □

Lemma 5. *Let (v, B, w) be an interval of $G = (V, E)$ and let $K_v, K_w \subseteq V$ be compatible with $(v, R, R'), (w, Q, Q') \in \mathcal{R}(G)$, respectively. Then $K = (K_v \setminus C(w, v)) \cup (K_w \setminus C(v, w))$ is compatible with both (v, R, R') and (w, Q, Q') if $K_0 = \{v, w\} \cup R \cup R' \cup Q \cup Q'$ is compatible with both local maps.*

Proof. We assume $K_0 \in \mathcal{K} = \mathcal{K}(v, R, R') \cap \mathcal{K}(w, Q, Q')$. Observe that $K_w \cap N(v) \cap N(w) = K_v \cap N(v) \cap N(w) = R' \cap Q'$.

Firstly, we show that $K'_v = (K_v \setminus C(w, v)) \cup K_0$ is compatible with both (v, R, R') and (w, Q, Q'). By applying Lemma 4 to the block $(v, C(w, v))$, with $K_1 = K_v$ and $K_2 = K_0$, we obtain that $K'_v \in \mathcal{K}(v, R, R')$. Since $K'_v \subseteq K_v \in \mathcal{K}(w, Q, Q')$ and K'_v contains $\{w\} \cup Q \cup Q'$, we have $K'_v \in \mathcal{K}(w, Q, Q')$. For similar reasons, $K'_w = (K_w \setminus C(v, w)) \cup K_0 \in \mathcal{K}$.

We apply Lemma 4 to the block $(v, C(w, v))$, with $K_1 = K'_v$ and $K_2 = K'_w$ and obtain that $K \in \mathcal{K}(v, R, R')$. By Lemma 4 again, applied to $(w, C(v, w))$ with $K_1 = K'_w$ and $K_2 = K'_v$, we conclude that $K \in \mathcal{K}(w, Q, Q')$. □

4.2 Synthesis

Let (v, C) be a block of an AT-free graph $G = (V, E)$, $(v, R, R') \in \mathcal{R}(G)$ and $F \subseteq V$. We define

$$\mathsf{start}(v, R, R') = \max_{K \in \mathcal{K}(v,R,R')} |K \cap N[v]| \,, \tag{4}$$

$$\mathsf{branch}(v, R, R', C) = \max_{K \in \mathcal{K}(v,R,R')} |K \cap C| \,, \tag{5}$$

$$\mathsf{accum}(F, v, R, R') = \sum_{\substack{(v,C) \text{ is block} \\ C \cap F = \varnothing}} \mathsf{branch}(v, R, R', C) \,. \tag{6}$$

Then

$$\mathrm{fvs}(G) = |V| - \max_{(v,R,R') \in \mathcal{R}(G)} \big(\mathsf{start}(v, R, R') + \mathsf{accum}(\varnothing, v, R, R')\big) \tag{7}$$

follows directly from Equations (3–6) and Lemma 4.

4.3 Computing start

As usual $\alpha(G)$ denotes the maximum size of an independent set $S \subseteq V$ of $G = (V, E)$. For a local map (v, R, R') of G we define

$$\mathsf{Inf}(v, R, R') = \{x \in N(v) : N(x) \setminus N(v) \subseteq N(R)\} \setminus (N(R) \cup N(R'))\,. \quad (8)$$

Lemma 6.

$$\mathsf{start}(v, R, R') = 1 + |R \cup R'| + \alpha(G[\mathsf{Inf}(v, R, R')]) \quad (9)$$

holds for all local maps $(v, R, R') \in \mathcal{R}(G)$.

Proof. Let S be a maximum independent set of $G[\mathsf{Inf}(v, R, R')]$. Then $R \cup R'$ is an independent set by definition of a local map, and $R \cup R' \cup S$ is an independent set by Equation (8). Hence $\{v\} \cup R \cup R' \cup S \in \mathcal{K}(v, R, R')$ and we conclude $\mathsf{start}(v, R, R') \geqslant 1 + |R \cup R'| + \alpha(G[\mathsf{Inf}(v, R, R')])$.

On the other hand we have $\{v\} \cup R \cup R' \subseteq K$ by Definition 2 and $|K \setminus (\{v\} \cup R \cup R')| \leqslant |S|$ for any $K \in \mathcal{K}(v, R, R')$ by Definition 2 again and our choice of the set S. This implies $\mathsf{start}(v, R, R') \leqslant 1 + |R \cup R'| + \alpha(G[\mathsf{Inf}(v, R, R')])$. □

4.4 Computing branch

Lemma 7. *Let K induce a forest in an AT-free graph G. Consider a vertex $v \in K$ and a block (v, C) such that $K \cap C \neq \varnothing$. Then there is a vertex $w \in K \cap C$ such that $K \cap B = \varnothing$ for the interval (v, B, w).*

Proof. Let w be any vertex in $K \cap C$ and assume $u \in K \cap B$. By an inductive argument it suffices to show $A \cup N[u] \subseteq B \cup N[w]$ and $w \notin A \cup N[v]$ where (v, A, u) is an interval. The inclusion follows from $u \in B$ and $w \in A \cup N[u]$ would imply that u, v and w form an AT of G. □

Lemma 8. *For every block (v, C) of an AT-free graph G and all $(v, R, R') \in \mathcal{R}(G)$ we have*

$$\begin{aligned}&\mathsf{branch}(v, R, R', C)\\ &= \max_{\substack{w \in C \text{ and}\\ \{v,w\} \cup R \cup R' \cup Q \cup Q' \in\\ \mathcal{K}(v,R,R') \cap \mathcal{K}(w,Q,Q')}} \big(\mathsf{start}(w, Q, Q') - |R' \cap S'| + \mathsf{accum}(\{v\}, w, Q, Q')\big) \end{aligned} \quad (10)$$

where $(w, Q, Q') \in \mathcal{R}(G)$ and $\mathsf{branch}(v, R, R', C) = 0$ if there is no suitable local map (w, Q, Q') exists for any vertex $w \in C$.

Proof. Let K induce a forest in G that is compatible with (v, R, R'). By Lemma 7 there is a vertex $w \in C$ such that the interval between v and w does not contain vertices in K. Let (w, Q, Q') be the corresponding local map of K. For each component $G[D]$ of $G - N[v]$ we have $|K \cap D| \leqslant \mathsf{branch}(w, Q, Q')$ by Equation (5). Summing these values up for all $D \not\ni v$ we obtain $|K \cap \bigcup_{D \not\ni v} D| \leqslant \mathsf{accum}(\{v\}, w, Q, Q')$. Moreover Equation (4) implies $|K \cap N[w] \setminus N[v]| \leqslant \mathsf{start}(w, Q, Q') - |R' \cap S'|$. Since the interval between

v and w does not contain vertices in K it follows $|K \cap C| \leqslant \max\big(\mathsf{start}(w,Q,Q') - |R' \cap S'| + \mathsf{accum}(\{v\}, w, Q, Q')\big)$.

The other way around, let the $\max$ in Equation (10) be attained for $w \in C$ and the local map (w,Q,Q'). For each block (w,D) with $v \notin D$ let $K_D \in \mathcal{K}(w,Q,Q')$ be a subset of D such that $|K_D| = \mathsf{branch}(w,Q,Q',D)$, $K_w \in \mathcal{K}(w,Q,Q')$ be a subset of $N[w]$ with $|K_w| = \mathsf{start}(w,Q,Q')$ and $K_v \in \mathcal{K}(v,R,R')$ be a subset of $N[v]$ with $|K_v| = \mathsf{start}(v,R,R')$. Then by Lemmas 4 and 5 the set $K_v \cup K_w$ is compatible with (v,R,R') and (w,Q,Q') and hence $K = K_v \cup K_w \cup \bigcup_{D \not\ni v} K_D$ is compatible with (v,R,R') and fulfils $|K \cap C| = \max\big(\mathsf{start}(w,Q,Q') - |R' \cap S'| + \mathsf{accum}(\{v\}, w, Q, Q')\big)$. □

4.5 Algorithm

Our algorithm computes $\mathrm{fvs}(G)$ for AT-free graphs G as follows:

step 1 For all $(v,R,R') \in \mathcal{R}(G)$ compute $\mathsf{start}(v,R,R')$ using Equation (9)
step 2 Create a list of all blocks (v,C) of G and sort by $|C|$
step 3 For each block (v,C) and each local map (v,R,R') compute $\mathsf{branch}(v,R,R',C)$ according to Equations (10) and (6) using the values $\mathsf{start}(w,Q,Q')$ computed in step 1 and values $\mathsf{branch}(w,Q,Q',D)$ computed earlier in this step.
step 4 Determine $\mathrm{fvs}(G)$ by Equations (7) and (6) using the values $\mathsf{branch}(v,R,R',C)$ computed in step 3

Theorem 2. *There is an $\mathcal{O}(n^8m^2)$ algorithm to compute the minimum size of a feedback vertex set in AT-free graphs.*

Proof. The correctness of this algorithm follows from Lemmas 6 and 8. We analyse the running time for an input $G = (V,E)$ where $n = |V|$, $m = |E|$ and n_v is the degree $d_G(v)$ of $v \in V$.

step 1 For each $(v,R,R') \in \mathcal{R}$ we have $|R \cup R'| \leqslant 4$, see Condition 4 in Lemma 3. Hence we have $\mathcal{O}(n_v^4)$ local maps for each $v \in V$. This adds up to $|\mathcal{R}| = \mathcal{O}(n^3m)$ because $\sum_{v \in V} n_v = 2m$.
We compute $\alpha(G[\mathsf{Inf}(v,R,R')])$ in time $\mathcal{O}(n_v^3)$ time because G is AT-free [15]. So the total time for this step is $\mathcal{O}(n^6m)$.
step 2 The number of blocks is $\mathcal{O}(n^2)$ and for fixed v we find all blocks (v,C) in time $\mathcal{O}(n+m)$. Hence this step needs time $\mathcal{O}(nm+n^2)$.
step 3 Here we compute $\mathcal{O}(n^4m)$ values $\mathsf{branch}(v,R,R',C)$. For each one we minimise over $\mathcal{O}(n^3m)$ local maps (w,Q,Q'). Since $|Q \cup Q'| \leqslant 4$ we look up a single value $\mathsf{start}(w,Q,Q')$ or $\mathsf{branch}(w,Q,Q',D)$ in constant time. Consequently this step takes $\mathcal{O}(n^8m^2)$ time.
step 4 The last step is similar to step 3 and takes time $\mathcal{O}(n^7m^2)$.

Hence the overall running time can be bounded by $\mathcal{O}(n^8m^2)$. □

5 Feedback Vertex Set in graphs with bounded asteroidal number

The following definition from [13] generalises asteroidal triples.

Definition 3. *An independent set A of G is called* asteroidal set *if for every vertex $a \in A$ there is a connected component $G[C]$ of $G - N[a]$ such that $A \setminus \{a\} \subseteq C$. The* asteroidal number *of G, denoted by $\mathrm{an}(G)$, is the maximum cardinality of an asteroidal set of G.*

It is easy to see that an independent set is asteroidal if and only if each three-element subset forms an AT. The following Ramsey-type lemma will be used in the proof of Lemma 10.

Lemma 9. *Each graph on more than $2k^3$ vertices has $k + 1$ pairwise nonadjacent vertices or a 2-connected subgraph on $k + 1$ vertices.*

Proof. Let $G = (V, E)$ be any graph on $|V| > 2k^3$ vertices that has no 2-connected subgraph on $k + 1$ vertices. By $\mathcal{B}$ we denote the set of all subsets $B \subseteq V$ such that $G[B]$ is a maximal subgraph of G without cut vertex. Two different sets $A, B \in \mathcal{B}$ are either disjoint or their intersection contains just one cut vertex of G. Clearly $|\mathcal{B}| \geqslant \lceil (2k^3 + 1)/k \rceil = 2k^2 + 1$. The set of cut vetices of G is denoted by C. We construct the forest $T = (\mathcal{B}, C, F)$ with $\{B, c\} \in F$ if $c \in B$ [20]. If T contains $k + 1$ leaves we can choose a vertex from each leaf. These vertices form an independent set of G.

Otherwise we fix $B_0 \in \mathcal{B}$ and partition $\mathcal{B}$ into *layers* $L_i = \{B : d_T(B_0, B) = 2i\}$, $i \geqslant 0$. Since T has at most k leaves we have $|L_i| \leqslant k$ for all i. Hence there are more than $2k$ nonempty layers and T has a path containing at least $2k+1$ nodes in $\mathcal{B}$. Finally we choose a vertex from every other node in $\mathcal{B}$ on this path. This can easily be done such that these vertices form an independent set of G. $\square$

Lemma 10. *For all vertices v of graph G with $\mathrm{an}(G) \leqslant k$ every independent set $S \subseteq N(v)$ has a representation of size at most $2k^3$.*

Proof. Conversely assume that S is an independent set containing more than $2k^3$ vertices such that S is its own representation. Then every vertex $s \in S$ has a private neighbour $t \in N(s) \setminus (N(v) \cup \bigcup_{s' \in S \setminus \{s\}} N(s'))$. Let T be the set of these private neighbours. Since $|T| = |S| > 2k^3$ the Lemma 9 ensures that $G[T]$ contains an independent set T' with $|T'| = k + 1$ or a 2-connected subgraph $G[C]$ on $k + 1$ vertices. In the latter case let $S' \subseteq S$ be the set of private neighbours of C. In both cases T' and S', respectively, form an asteroidal set of G contradicting $\mathrm{an}(G) \leqslant k$. $\square$

Here we generalise our approach in Section 4 to graphs of bounded asteroidal number. First we reconsider Lemma 3. Let $G[K]$ be a forest in G containing the vertex v. We consider the local map (v, R, R') of K, i.e. R' is the set of internal vertices of $G[K]$ belonging to $N(v)$ and R is a representation of $(K \cap N(v)) \setminus R'$.

Lemma 11. *Let (v, R, R') be a local map of $K \ni v$ inducing a forest $G[K]$ in G. The following conditions hold:*

1. $R \cup R' \subseteq N(v)$;

2. $R \cup R'$ *is an independent set of* G*;*
3. $N(x) \setminus N(v) \subset N(x') \setminus N(v)$ *implies* $x \in R$ *and* $x' \in R'$ *for all* $x, x' \in R \cup R'$*;*
4. $|R| \leqslant 2k^3$ *and* $|R'| \leqslant k$ *where* $k = \mathrm{an}(G)$.

Proof. Conditions 1–3 are the same as in Lemma 3. $|R| \leqslant 2k^3$ follows from Lemma 10. It remains to prove $|R'| \leqslant k$. For each $s \in R'$ we choose a neighbour t in $K \setminus N[v]$. Let T be the set of these neighbours. Then $G[R' \cup T]$ contains exactly $|R'|$ edges, *i.e.* T is an independent set of private neighbours. In other words, T is an asteroidal set of G. Condition 4 follows. $\square$

Now a *local map* of G is a triple (v, R, R') satisfying the conditions of Lemma 11. The set of all local maps of G is denoted by $\mathcal{R}(G)$ again. We retain the concept of *compatibility* as introduced in Definition 2.

Lemma 4 holds for all graphs. We extend Lemma 5 as follows. For an asteroidal set A of G consider the set $B(A) = \bigcap_{a,b \in A, a \neq b} C(b, a)$. We say that $(A, B(A))$ is an *interval* of G. Especially every block (v, C) is an interval $(\{v\}, C)$.

Lemma 12. *Let* (A, B) *be an interval of* $G = (V, E)$*. Let* $K_a \subseteq V$ *be compatible with* $(a, R_a, R'_a) \in \mathcal{R}(G)$ *for all* $a \in A$*. Then* $K = \bigcup_{a \in A}(K_a \setminus \bigcup_{b \in A \setminus \{a\}} C(b, a))$ *is compatible with* (a, R_a, R'_a) *for all* $a \in A$ *if* $K_0 = A \cup \bigcup_{a \in A}(R_a \cup R'_a)$ *is compatible with* (a, R_a, R'_a) *for all* $a \in A$*.*

Proof. omitted

We keep the definition of start, branch and accum (Equations (4), (5) and (6)). Since Lemma 6 holds for all graphs we can compute $\mathsf{start}(v, R, R')$ as before, see Equation (9). To compute $\mathsf{branch}(v, R, R', C)$ we need the following generalisation of Lemma 7.

Lemma 13. *Let* (v, C) *be a block of* G *and let* $G[K]$ *be a forest containing* v*. There is an asteroidal set* $A \subseteq K$ *of* G *with* $v \in A$ *such that the interval* $(A, B(A))$ *satisfies* $B(A) \cap K = \varnothing$*.*

Proof. We start with $A = \{v\}$ and $B(A) = C$. While $B(A) \cap K \neq \varnothing$,

- choose a vertex $w \in B(A) \cap K$ and let $S(v, w) = N(w) \cap N(C(v, w))$ be the unique minimal v, w-separator contained in $N(w)$;
- remove from A all the vertices a such that $S(v, w)$ separates v and a;
- add w to A.

We show that, after each step of the loop, A is an asteroidal set. At the beginning of the iteration, for any distinct vertices $a, a' \in A$, there is a a, w-path avoiding $N[a']$, by the fact that $w \in B(A) \subseteq C(a, a')$. At the end of the iteration, for any $a \in A \setminus \{v\}$, there is a a, v-path avoiding $N[w]$, otherwise a would be removed from A. Hence A is an asteroidal set.

At each step of the loop $B(A_{\mathrm{new}}) = B(A_{\mathrm{old}}) \cap C(v, w)$ holds. So we iterate at most n times. $\square$

Lemma 14. *For every block* (v, C) *of a graph* G *and all* $(v, R, R') \in \mathcal{R}(G)$ *we have*

$$\mathsf{branch}(v, R, R', C) = \max_{\substack{A \subseteq C \\ A \cup \{v\} \text{ is asteroidal} \\ A \cup \{v\} \cup \bigcup_{a \in A \cup \{v\}} (R_a \cup R'_a) \in \bigcap_{a \in A \cup \{v\}} \mathcal{K}(a, R_a, R'_a)}} \Big(|\bigcup_{a \in A} R'_a \setminus R'| + \sum_{a \in A} \big(\mathsf{start}(a, R_a, R'_a) - |R'_a| + \mathsf{accum}(\{v\}, a, R_a, R'_a)\big)\Big) \quad (11)$$

where $R_v = R$*,* $R'_v = R'$ *and* $(a, R_a, R'_a) \in \mathcal{R}(G)$ *for all* $a \in A$*.*

Proof. omitted

From Lemmas 6 and 14 we derive an algorithm similar to the one given in Subsection 4.5. We claim that it runs in polynomial time whenever the asteroidal number of the input is bounded by a constant. The tricky point is to bound the size of $\mathcal{R}(G)$. Here Lemma 11 can help. Due to its Condition 4 we know $|R| \leqslant 2k^3$ and $|R'| \leqslant k$ for every local map $(v, R, R') \in \mathcal{R}(G)$ if $\mathrm{an}(G) \leqslant k$. Hence $|\mathcal{R}(G)| < n^{2k^3+k+1}$ where $k = \mathrm{an}(G)$.

References

1. V. Bafna, P. Berman, T. Fujito: A 2-approximation algorithm for the undirected feedback vertex set problem, *SIAM Journal on Discrete Mathematics* **12** (1999), pp. 289–297.
2. H. Bodlaender, D. Thilikos: Treewidth for graphs with small chordality. *Discrete Applied Mathematics* **79** (1997), pp. 45–61.
3. A. Brandstädt, D. Kratsch: On domination problems for permutation and other graphs, *Theoretical Computer Science* **54** (1987) pp. 181–198.
4. A. Brandstädt, V.V. Lozin: On the linear structure and clique width of bipartite permutation graphs, *RRR-29-2001*, Rutgers University.
5. H.J. Broersma, A. Huck, T. Kloks, O. Koppius, D. Kratsch, H. Müller and H. Tuinstra: Degree-preserving trees, *Networks* **35** (2000), pp. 26–39.
6. H.J. Broersma, T. Kloks, D. Kratsch and H. Müller: Independent sets in asteroidal triple-free graphs, *SIAM Journal on Discrete Mathematics* **12** (1999), pp. 276–287.
7. H.J. Broersma, T. Kloks, D. Kratsch and H. Müller: A generalization of AT-free graphs and a generic algorithm for solving triangulation problems, *Algorithmica* **32** (2002) pp. 594–610.
8. D.G. Corneil, S. Olariu and L. Stewart: Asteroidal triple-free graphs, *SIAM Journal on Discrete Mathematics* **10** (1997), pp. 399–430.
9. B. Courcelle, J.A. Makowsky, U. Rotics: Linear time solvable optimization problems on graphs of bounded clique-width, *Theory of Computing Systems* **33** (2000), pp. 125–150.
10. R.G. Downey, M.R. Fellows: *Parameterized complexity*, Springer, 1997.
11. M.R. Garey and D.S. Johnson: *Computers and Intractability: A guide to the Theory of NP-completeness*, Freeman, New York, 1979.
12. F. Gavril: Algorithms for maximum weight induced paths, *Information Processing Letters* **81** (2002), pp. 203–208
13. K. Kloks, D. Kratsch and H. Müller: Asteroidal sets in graphs, *Proceedings WG'97*, Lecture Notes in Computer Science N^o 1335, 229–241, Springer-Verlag, 1997.

14. J. Kleinberg, A. Kumar: Wavelength conversion in optical networks, *Journal of Algorithms* **38** (2001) pp. 25–50.
15. E. Köhler: Graphs without asteroidal triples, PhD thesis, TU Berlin 1999. `ftp://ftp.math.tu-berlin.de/pub/combi/ekoehler/diss`
16. D.Y. Liang: On the feedback vertex set problem in permutation graphs, *Information Processing Letters* **52** (1994) pp. 123–129.
17. D.Y. Liang, M.S. Chang: Minimum feedback vertex sets in cocomparability graphs and convex bipartite graphs, *Acta Informatica* **34** (1997) pp. 337–346.
18. C.L. Lu, C.Y. Tang: A linear-time algorithm for the weighted feedback vertex problem on interval graphs, *Information Processing Letters* **61** (1997) pp. 107–111.
19. J. Spinrad: *Efficient graph representations*, American Mathematical Society, Fields Institute Monograph Series 19, 2003.
20. R. Tarjan: Depth-first search and linear graph algorithms. *SIAM Journal on Computing* **1** (1972) pp. 146–160.