ELSEVIER

Discrete Applied Mathematics 121 (2002) 155–180

DISCRETE APPLIED MATHEMATICS

On claw-free asteroidal triple-free graphs ☆

Harald Hempel *, Dieter Kratsch [1]

Fakultät für Mathematik und Informatik, Friedrich-Schiller-Universität Jena, D-07740 Jena, Germany

Received 17 May 1999; received in revised form 12 March 2001; accepted 2 April 2001

Abstract

We present an $O(n^{2.376})$ algorithm for recognizing claw-free AT-free graphs and a linear-time algorithm for computing the set of *all* central vertices of a claw-free AT-free graph. In addition, we give efficient algorithms that solve the problems INDEPENDENT SET, DOMINATING SET, and COLORING. We argue that all running times achieved are optimal unless better algorithms for a number of famous graph problems such as triangle recognition and bipartite matching have been found. Our algorithms exploit the structure of 2LexBFS schemes of claw-free AT-free graphs.

Keywords: Graphs; Algorithms; NP-complete; Recognition; Claw-free; AT-free

1. Introduction

In this paper we study claw-free AT-free graphs and show that a large number of combinatorial problems can be efficiently solved. Moreover, we give strong arguments that the algorithms we present are optimal. In particular, we consider the following:

1. *RECOGNITION*: We present an $O(n^{2.376})$ recognition algorithm for claw-free AT-free graphs. We show that this is optimal unless one finds an algorithm for triangle recognition that runs faster than $O(n^{2.376})$.

2. *RADIUS*: We prove that there exists a linear-time algorithm for computing the radius of a claw-free AT-free graph.

3. *ALL CENTRAL VERTICES*: Though no linear-time algorithm for finding just *one* central vertex for AT-free graphs is known we give a linear-time algorithm that computes the set of *all* central vertices of a claw-free AT-free graph.

☆ An extended abstract of this paper was presented at the 25th Workshop on Graph-Theoretic Concepts in Computer Science (WG'99) [16].

* Corresponding author.

E-mail addresses: hempel@minet.uni-jena.de (H. Hempel), kratsch@minet.uni-jena.de, kratsch@lita.univ-metz.fr (D. Kratsch).

[1] Current address: Université de Metz, Laboratoire d' Informatique Théorique et Appliqué, Île du Saulcy, 57045 Metz Cedex 01, France.

0166-218X/02/$ - see front matter
PII: S0166-218X(01)00208-6

4. *INDEPENDENT SET*: We give a linear-time algorithm for claw-free AT-free graphs.

5. *DOMINATING SET*: We show that claw-free AT-free graphs admit a linear-time algorithm.

6. *COLORING*: We prove that computing an optimal coloring for claw-free AT-free graphs is closely related to both computing a maximum matching on bipartite graphs and on general graphs. We present an $O(\sqrt{n}m)$ coloring algorithm for claw-free AT-free graphs. Any improvement of this time bound implies the existence of an algorithm for computing a maximum matching for bipartite graphs that runs in time less than $O(n^2 + \sqrt{n}m)$.

Claw-free AT-free graphs, the class of all graphs neither containing a claw as an induced subgraph nor an asteroidal triple, form a subclass of AT-free graphs and contain all complements of bipartite graphs. A *claw* is a graph on four vertices such that one of them is adjacent to the other three vertices which themselves are pairwise non-adjacent. An *asteroidal triple* is a triple of vertices such that for every pair of two of those vertices there exists a path joining them that does not contain any vertex of the closed neighborhood of the third vertex.

With respect to the combinatorial problems listed above the following is known for AT-free graphs. The straightforward and currently the best known upper bound for recognizing (dense) AT-free graphs is $O(n^3)$. We mention that $O(nm + n^{2.82})$ and $O(\overline{m}^{1.5} + n^2)$[2] algorithms[2] can be obtained [20], that are faster than the straightforward algorithm for sparse graphs and complements of sparse graphs, respectively. The best (conditional) lower bound for recognizing AT-free graphs is closely related to the upper bound of triangle recognition which is currently $O(n^\alpha)$ [27], where $O(n^\alpha)$ is the time to multiply two binary $n \times n$ matrices. (At present $\alpha = 2.376$ ... [9].)

It has been shown that the diameter of AT-free graphs can be computed in linear time with an absolute error of 1 by 2LexBFS (see [10]). However, for both computing the diameter and the radius no linear-time algorithm is known for AT-free graphs. To the best of our knowledge there exists no linear-time algorithm computing even *one* central vertex of an AT-free graph. COLORING is one of the most interesting open problems for AT-free graphs, still waiting for a polynomial-time algorithm or a proof of NP-completeness. The best algorithms for INDEPENDENT SET and DOMINATING SET for AT-free graphs have running times of $O(n^4)$ [6] and $O(n^6)$ [21], respectively.

Claw-free AT-free graphs have been considered in the literature before [18,24]. As a consequence of a theorem in [10], the diameter of a claw-free AT-free graph can be computed in linear time. HAMILTON PATH and HAMILTON CIRCUIT can be solved in linear time on (claw,net)-free graphs [5], an interesting superclass of the claw-free AT-free graphs. Kloks et al. have given a characterization of claw-free AT-free graphs. For each connected graph G holds: G is claw-free AT-free if and only if either $\alpha(G) \leqslant 2$

[2] As is standard in graph theory, n and m denote the number of vertices and edges of the input graph, respectively. Similarly, $\overline{m}$ denotes the number of edges in the complement of the input graph.

or G is a claw-free cocomparability graph [18]. Though that characterization does give some insight into what makes graphs claw-free AT-free it does not offer a general tool for solving many of the combinatorial problems we address.

Our approach is mainly based on structural properties of 2LexBFS schemes for claw-free AT-free graphs. 2LexBFS schemes have played a major role in showing that a dominating pair for AT-free graphs can be computed in linear time [12]. We show that almost all levels of a 2LexBFS scheme of a claw-free AT-free graph are cliques, and that the union of any two consecutive levels does not contain three vertices forming an independent set. These and other observations allow us to exploit the 2LexBFS scheme for our purposes.

2. Preliminaries

We consider only finite, undirected, simple, and *connected* graphs. For a graph $G=(V,E)$ and $W \subseteq V$, $G[W]$ denotes the subgraph of G induced by vertices of W. For a vertex x of $G=(V,E)$, $N(x)=\{y \in V: \{x,y\} \in E\}$ is the neighborhood of x and $N[x]=N(x) \cup \{x\}$ is the closed neighborhood of x. For $W \subseteq V$, $N[W]=\bigcup_{x \in W} N[x]$.

Let $G=(V,E)$ be a graph. An *independent set* is a set of pairwise non-adjacent vertices of G. A *clique* is a set of pairwise adjacent vertices. A vertex set $D \subseteq V$ is a *dominating set* of G if every vertex $u \in V \setminus D$ is adjacent to a vertex $v \in D$, i.e., $N[D]=V$. An *independent dominating set* is a vertex set that is dominating and independent. A *matching* is a set of pairwise non-adjacent edges. A (proper) vertex *coloring* assigns a color to each vertex such that different colors are assigned to adjacent vertices. These concepts lead to natural decision and optimization problems. The corresponding graph parameters are $\alpha(G)$, $\omega(G)$, and $m(G)$ to denote the maximum cardinality of an independent set, clique, and matching of G, respectively, and $\gamma(G)$, $\gamma_i(G)$, and $\chi(G)$ to denote the minimum cardinality of a dominating set, independent dominating set, and the minimum number of colors in a coloring of G, respectively.

For standard graph theory notation we refer to [29]. For definitions and properties of special graph classes we refer to [4,15].

Definition. A *claw* is a graph on four vertices such that one of them, called the *center*, is adjacent to the other three vertices which themselves are pairwise non-adjacent. A graph G is called *claw-free* if it has no claw as an induced subgraph.

A triple $\{x,y,z\}$ of vertices of a graph G is an *asteroidal triple* (AT) if for every two of these vertices there is a path between them avoiding the closed neighborhood of the third. A graph G is called *asteroidal triple-free* (AT-free) if it has no asteroidal triple. An AT-free graph is *claw-free AT-free* if it does not contain a claw as an induced subgraph.

It is easy to see that complements of triangle-free graphs (i.e., graphs G with $\alpha(G) \leqslant 2$) form a subclass of claw-free graphs. It is well known that line graphs also form a subclass of claw-free graphs [29].

Corneil et al. initiated and contributed substantially to an extensive research on structural and algorithmic properties of AT-free graphs [11,12]. One of their major discoveries is the relation between an old graph search procedure, called LEXICOGRAPHIC BREADTH-FIRST SEARCH (LexBFS), and so-called dominating pairs [12].

Definition. A pair (x, y) of vertices of a graph G is a *dominating pair* (short DP) if for every path P between x and y, the vertex set of P is a dominating set of G.

Theorem 1 (Corneil et al. [11]). *Every connected AT-free graph has a dominating pair.*

In [12] a linear-time algorithm for computing a dominating pair of a given connected AT-free graph was established. The main idea of this algorithm is the use of LexBFS. LexBFS is a variant of BREADTH-FIRST SEARCH that has been introduced by Rose et al. in 1976 [25]. It has been used as a subroutine in a variety of efficient (often linear-time) graph algorithms. We reproduce the details of the linear-time algorithms LexBFS and 2LexBFS from [25,12].

Procedure LEXBFS(G,x);
Input: a connected graph $G=(V,E)$ and a distinguished vertex x of G
Output: a numbering σ of the vertices of G
begin
  label(x) := $|V|$;
  for each vertex v in $V \setminus \{x\}$ **do**
  label(v) := Λ;
  for $i := |V|$ **downto** 1 **do**
  begin
    pick an unnumbered vertex v with (lexicographically) largest label;
    $\sigma(v) := i$; {assign number i to v}
    for each unnumbered vertex u in $N(v)$ **do**
      append i to label(u)
  end
return σ
end; {LexBFS}
Procedure 2LEXBFS(G);
Input: a connected graph $G=(V,E)$
Output: a numbering σ_2 of the vertices of G
begin
  choose a vertex w of G;
  LexBFS(G,w);
  let σ_1 be the output of LexBFS(G,w) and let x be the vertex with $\sigma_1(x)=1$;
  LexBFS(G,x);
  let σ_2 be the output of LexBFS(G,x);
return σ_2
end; {2LexBFS}

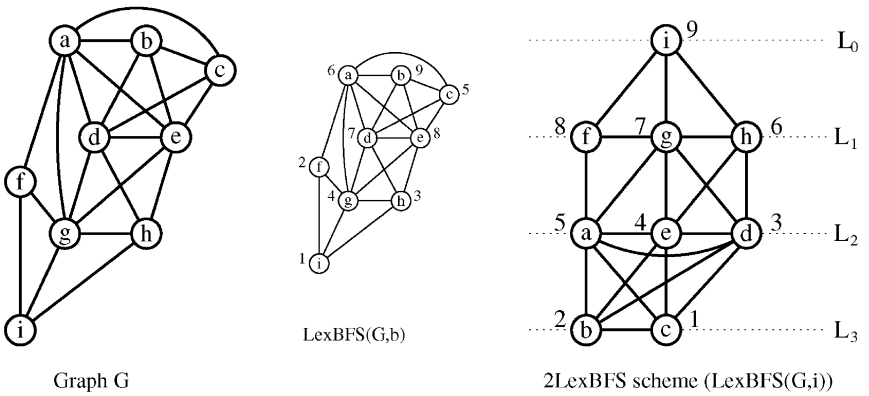

Fig. 1. A claw-free AT-free graph G and one of its 2LexBFS schemes.

A vertex ordering $x_n, x_{n-1}, \ldots, x_1$ is called a *2LexBFS ordering* of G if some 2LexBFS(G) returns σ_2 such that $\sigma_2(x_j) = j$ for all j. A 2LexBFS ordering and the levels $L_0 = \{x_n\}$, $L_1 = N(x_n), \ldots, L_i = \{x_j : d(x_j, x_n) = i\}, \ldots, L_r$ are called a *2LexBFS scheme* of G. Observe that for all $0 \leqslant i \leqslant r$, $L_i = \{x_t, x_{t-1}, \ldots, x_{t'}\}$ for suitable $t \geqslant t'$. In other words, a 2LexBFS ordering induces an ordering of the vertices of each of its levels. Throughout the paper, whenever we refer to an ordering of the vertices of a level of a 2LexBFS scheme we refer to the ordering of the vertices of that level that is induced by the 2LexBFS ordering, that is the vertices of each level are ordered from highest to lowest assigned number (see Fig. 1).

Theorem 2 (Corneil et al. [12]). *Every 2LexBFS ordering $x_n, x_{n-1}, \ldots, x_1$ of a connected AT-free graph has the* dominating pair-property (*DP-property*), *i.e., for all $i \in \{1, 2, \ldots, n\}$, (x_n, x_i) is a dominating pair of $G[\{x_i, x_{i+1}, \ldots, x_n\}]$.*

Consequently, for each 2LexBFS ordering $x_n, x_{n-1}, \ldots, x_1$ of an AT-free graph G, (x_1, x_n) is a DP. Thus, a DP can be computed in linear time by 2LexBFS [12]. A 2LexBFS scheme of an AT-free graph and its DP-property has also been used in [18] to establish a bandwidth approximation algorithm for AT-free graphs. The following characterization of claw-free AT-free graphs is given in the same paper.

Theorem 3 (Kloks et al. [18]). *Let $G = (V, E)$ be a connected graph. Then G is claw-free AT-free if and only if G is a claw-free cocomparability graph or $\alpha(G) \leqslant 2$.*

Note that *cocomparability graphs* (complements of comparability graphs) can be defined as those graphs that admit a *cocomparability ordering*, i.e., a vertex ordering $x_1, x_2, \ldots, x_n$ such that for all $i < j < k$: $\{x_i, x_k\} \in E$ implies $\{x_i, x_j\} \in E$ or $\{x_j, x_k\} \in E$ (see [4]).

3. Recognizing claw-free asteroidal triple-free graphs

The work of Corneil et al. on AT-free graphs led to a variety of interesting questions. A challenging one asks for an efficient recognition algorithm for AT-free graphs and it seemed natural to expect a dramatic improvement over the $O(n^3)$ running time of the straightforward algorithm [11]. However, recently Spinrad revealed a surprising connection to triangle recognition [27]. Triangle recognition is a well-studied fundamental graph algorithms problem and thus is sometimes even used as a lower bound for the complexity of other graph problems.

Theorem 4 (Spinrad [27]). *If there is an* $O(n^c)$ *algorithm for recognizing AT-free graphs then there is an* $O(n^{\max(2,c)})$ *algorithm for recognizing triangle-free graphs.*

The best known algorithm for recognizing a triangle in dense graphs, TRIANGLE(G), uses matrix multiplication and has running time $O(n^\alpha)$ (see [2]). Thus, any $O(n^c)$ algorithm for recognizing AT-free graphs with $c < \alpha$ would not only significantly improve the best known running time of a recognition algorithm for AT-free graphs, it would also improve the best known time bound for triangle recognition.

The reduction of Spinrad can also be used to establish a (conditional) lower bound for the time to recognize claw-free AT-free graphs.

Corollary 5. *If there is an* $O(n^c)$ *algorithm for recognizing claw-free AT-free graphs* (*of diameter* 2) *then there is an* $O(n^{\max(2,c)})$ *algorithm for recognizing triangle-free graphs.*

Proof. We take the opportunity to present the nice construction of Spinrad [27].

Let $G=(V,E)$ be any graph. Assume $V=\{v_1,v_2,\ldots,v_n\}$. A new graph H_G is constructed as follows: Take the graph G, add vertices $v'_1,v'_2,\ldots,v'_n$, and edges $\{v'_i,v'_j\}$ for all $1 \leqslant i,j \leqslant n$ and $\{v'_k,v_\ell\}$ for all $1 \leqslant k,\ell \leqslant n$ such that $k \neq \ell$. Note that $\{v'_1,v'_2,\ldots,v'_n\}$ is a clique in H_G and that the diameter of H_G is 2.

Claim. *G has three pairwise non-adjacent vertices, called an independent triple, if and only if H_G has a claw or an* AT.

> Suppose $\{v_i,v_j,v_k\}$ is an independent triple in G. Then $\{v_i,v_j,v_k\}$ is an AT in H_G which can be seen by inspecting the paths (v_i,v'_k,v_j), (v_i,v'_j,v_k), and (v_j,v'_i,v_k) in H_G. For the inverse direction of the claim to be shown suppose H_G has an AT $\{x,y,z\}$ or a claw $\{c,x,y,z\}$ with center vertex c. In each case $\{x,y,z\}$ is an independent triple in H_G and by the construction of H_G it follows immediately that $\{x,y,z\}$ is also an independent triple in G. This completes the proof of the claim.

Now suppose that we have a $t(n)$ algorithm for recognizing claw-free AT-free graphs. The following algorithm then recognizes triangle-free graphs in time $O(n^2)+t(2n)$: On

input G compute $\overline{G}$ and $H_{\overline{G}}$ in time $O(n^2)$. Then test whether $H_{\overline{G}}$ is claw-free AT-free. By assumption this can be done in time $t(2n)$. The correctness of the algorithm follows immediately from the above proven claim. □

In the following we present an $O(n^\alpha)$ algorithm for recognizing claw-free AT-free graphs. According to Corollary 5 this is the best possible time for recognizing claw-free AT-free graphs without strong consequences, i.e., without an improvement over the best known running time for triangle recognition.

Lemma 6. *Let $G=(V,E)$ be a claw-free AT-free graph. Let $L_0=\{x\}$, $L_1=N(x)$, $L_2,\ldots,L_i=\{w\in V: d(w,x)=i\},\ldots,L_r$ be the levels of a 2LexBFS scheme of G. Then the following statements hold*:

1. *L_i is a clique for all $i=0,2,3,\ldots,r$. (L_1 might not be a clique.)*
2. *$\alpha(G[L_1])\leqslant 2$.*

Proof. Clearly L_0 is a clique. Consider L_i with $i\geqslant 2$. Suppose $u,v\in L_i$ and $\{u,v\}\notin E$. Then u and v have a common neighbor w in L_{i-1} due to the DP-property of any 2LexBFS ordering of an AT-free graph. Clearly, w has a neighbor $z\in L_{i-2}$ and hence $\{w,u,v,z\}$ induces a claw in G, a contradiction.

Suppose $\alpha(G[L_1])\geqslant 3$. Let $\{a,b,c\}$ be an independent set of $G[L_1]$. Then $\{x,a,b,c\}$ induces a claw in G, a contradiction. □

Our recognition algorithm for claw-free AT-free graphs works as follows.

Algorithm RECOGNITION

1. Let $G=(V,E)$ be the input graph and let $V=\{v_1,v_2,\ldots,v_n\}$. Compute the binary adjacency matrix $A=(a_{ij})_{i,j=1,\ldots,n}$ of G (i.e., $a_{ij}=1$ if $\{v_i,v_j\}\in E$ and $a_{ij}=0$ if $\{v_i,v_j\}\notin E$, thus $a_{ii}=0$ for all $i\in\{1,2,\ldots,n\}$). Compute A^2.
2. If all off diagonal entries of A^2 are equal to 1 (i.e., if $\mathrm{diam}(G)\leqslant 2$) call TRIANGLE($\overline{G}$). If the subroutine reports a triangle then reject G. Otherwise accept G. {This completes the test of input graphs G with $\mathrm{diam}(G)\leqslant 2$.}
3. Let $a^2_{ij}=0$ for some i and j, $i\neq j$, so $\mathrm{diam}(G)\geqslant 3$. Call 2LEXBFS($G$) such that first LEXBFS(G,v_i) is performed. Let $\mathscr{S}$ be the resulting 2LexBFS scheme, $x_n,x_{n-1},\ldots,x_1$ be the 2LexBFS ordering, and $L_0,L_1,\ldots,L_r$ be the levels of $\mathscr{S}$.
4. Compute another 2LexBFS scheme $\mathscr{S}'$ by calling LEXBFS(G,x_1). Let $x_1=y_n$, $y_{n-1},\ldots,y_1$ be the resulting 2LexBFS ordering and $L'_0,L'_1,\ldots,L'_{r'}$ be the corresponding levels. {Note $r,r'\geqslant 3$.}
5. If some level L_i or L'_i, $i\neq 1$, is not a clique then reject G.
6. If either TRIANGLE($\overline{G}[L_1]$) or TRIANGLE($\overline{G}[L'_1]$) reports a triangle then reject G.
7. Compute the *gentle squares* $G^{\mathscr{S}}$ and $G^{\mathscr{S}'}$ of G with respect to $\mathscr{S}$ and $\mathscr{S}'$ in the following way. Make L_1 (respectively, L'_1) a clique and add all those edges $\{u,w\}\in E(G^2)\setminus E(G)$ to G for which $u\in L_i$ and $w\in L_{i+2}$ (respectively, $u\in L'_i$ and $w\in L'_{i+2}$) for some i.

8. If $x_n, x_{n-1}, \ldots, x_1$ is not a cocomparability ordering of $G^{\mathscr{S}}$ or $y_n, y_{n-1}, \ldots, y_1$ is not a cocomparability ordering of $G^{\mathscr{S}'}$ then reject G. Otherwise accept G.

Theorem 7. RECOGNITION *recognizes claw-free AT-free graphs in time* $O(n^\alpha)$.

Proof. First consider the correctness. Let $G=(V,E)$ be a graph. Suppose $\mathrm{diam}(G) \leqslant 2$. We will argue that then G has an independent triple if and only if G has an AT or a claw. Since both AT and claw contain independent triples one direction of that claim is trivial. For the other direction let $\{a,b,c\}$ be an independent set of G. If $N(a) \cap N(b) \cap N(c) \neq \emptyset$ then for each vertex $w \in N(a) \cap N(b) \cap N(c)$, $\{w,a,b,c\}$ induces a claw. Otherwise, i.e., if $N(a) \cap N(b) \cap N(c) = \emptyset$, there are pairwise different vertices u, v, w such that $u \in N(a) \cap N(b)$, $v \in N(a) \cap N(c)$, and $w \in N(b) \cap N(c)$ since G has diameter 2. Thus, $\{a,b,c\}$ is an AT of G. This proves the correctness of step 2 of our algorithm.

Assume $\mathrm{diam}(G) \geqslant 3$. Consider the two 2LexBFS schemes $\mathscr{S}$ and $\mathscr{S}'$ computed by the algorithm. Assume both have the properties of Lemma 6. (Note that Lemma 6 establishes the correctness of steps 5 and 6.) We claim that then G is AT-free and any claw in G is a *standard claw* in $\mathscr{S}$ and $\mathscr{S}'$. A standard claw in $\mathscr{S}$ (respectively, $\mathscr{S}'$) is a claw $\{w,a,b,c\}$ with center w such that $w,\ b \in L_i,\ a \in L_{i-1},\ c \in L_{i+1}$ (respectively, $w,\ b \in L'_i,\ a \in L'_{i-1},\ c \in L'_{i+1}$) for some $i \in \{2,3,\ldots,r-1\}$ (respectively, $i \in \{2,3,\ldots,r'-1\}$).

Suppose $\{a,b,c\}$ is an AT in G. Then the construction of $\mathscr{S}$ and $\mathscr{S}'$, in particular, r, $r' \geqslant 3$, guarantees that either $|L_1 \cap \{a,b,c\}| \leqslant 1$ or $|L'_1 \cap \{a,b,c\}| \leqslant 1$. Without loss of generality assume $|L_1 \cap \{a,b,c\}| \leqslant 1$. Thus, no two vertices of $\{a,b,c\}$ are in one level of $\mathscr{S}$. Hence if $a \in L_i$, $b \in L_j$ and $c \in L_k$ with $i < j < k$ then there is no path between vertices a and c avoiding $N(b) \supseteq L_j$. Thus, $\{a,b,c\}$ is no AT. Consequently, G is AT-free. Now assume that $\{w,x,y,z\}$ induces a claw with center w in G. Notice that the construction of $\mathscr{S}$ and $\mathscr{S}'$ guarantees that either $|L_1 \cap \{x,y,z\}| \leqslant 1$ or $|L'_1 \cap \{x,y,z\}| \leqslant 1$. Without loss of generality assume $|L_1 \cap \{x,y,z\}| \leqslant 1$. Then $\{w,x,y,z\}$ is a standard claw in $\mathscr{S}$ since every L_i, $i \neq 1$, is a clique.

The standard claw test is done in steps 7 and 8. A standard claw $\{w,a,b,c\}$ in $\mathscr{S}$ (respectively, $\mathscr{S}'$) with $w,b \in L_i$, $a \in L_{i-1}$ and $c \in L_{i+1}$ having all properties required by the algorithm implies that in the gentle square of G we have the following configuration: $\{a,c\}$ is an edge, $\{a,b\}$ and $\{b,c\}$ are non-edges, and b is between a and c in the 2LexBFS ordering. Consequently, the 2LexBFS ordering of G is not a cocomparability ordering of the gentle square of G. On the other hand, if the 2LexBFS ordering is not a cocomparability ordering of the gentle square then there exists a triple $\{a,b,c\}$ of vertices violating the cocomparability ordering condition; say $\{a,c\}$ is an edge, $\{a,b\}$ and $\{b,c\}$ are non-edges in the gentle square of G, and b is between a and c in the 2LexBFS ordering of G. Clearly the vertices a, b and c must be from pairwise different levels, since all levels of the gentle square are cliques. We may assume without loss of generality that $a \in L_{i-1}$, $b \in L_i$, and $c \in L_{i+1}$ for some $i \in \{2,3,\ldots,r-1\}$. Since $\{a,c\}$ is an edge of the gentle square there is a common neighbor w of a and c in L_i. Since

$i \geqslant 2$ we have that $\{w,b\} \in E$ and hence $\{w,a,b,c\}$ induces a standard claw in G. This completes the proof of correctness of the steps 7 and 8.

Finally, consider the running time. All parts of the algorithm can obviously be done in time $O(n^\alpha)$ using triangle recognition or matrix multiplication, except the test for standard claws. The gentle square of G can be computed in time $O(n^\alpha)$ by using A^2. The test whether a 2LexBFS ordering is a cocomparability ordering for the gentle square can be done in $O(n^\alpha)$ by checking whether the digraph obtained by orienting the edges of the gentle square according to the 2LexBFS ordering (i.e., orient the edges from the smaller to the larger vertex) is transitive. Notice that testing whether a digraph is transitive is equivalent to matrix multiplication [13]. □

Corollary 8. *There is an* $O(n^\alpha)$ *time algorithm that recognizes whether a given AT-free graph is claw-free.*

Our algorithm also improves the running time of the $O(n^3)$ algorithm for computing the asteroidal number of a claw-free graph presented in [19], when used for recognizing asteroidal triples.

Corollary 9. *There is an* $O(n^\alpha)$ *time algorithm that recognizes whether a given claw-free graph is AT-free.*

4. All central vertices

In this section we consider distance properties of claw-free AT-free graphs. Distances in graphs and related graph theoretic parameters such as diameter and radius play an important role in the design and analysis of networks in a variety of networking environments like communication networks, electric power networks, and transportation networks. Until now no fast algorithms for computing the diameter of an arbitrary graph, avoiding the computation of the whole distance matrix, have been designed. There is a collection of work on distance problems for graphs of some special graph classes. We refer to [7,8] for additional references.

We will start by defining the main concepts. Let $G=(V,E)$ be a graph. The *distance* $d(u,v)$ between vertices u and v is the length (i.e., the number of edges) of a shortest path from u to v. The *eccentricity* $e(v)$ of a vertex v is $e(v) := \max_{u \in V} d(u,v)$. The *radius* $r(G)$ and the *diameter* $\mathrm{diam}(G)$ are defined as $r(G) := \min_{v \in V} e(v)$ and $\mathrm{diam}(G) := \max\{d(u,v)\colon u,v \in V\}$, respectively. Thus, $\mathrm{diam}(G) = \max_{v \in V} e(v)$. Finally, a vertex v is a *central vertex* of G if $e(v) = r(G)$. Our goal is to show how to compute all these parameters and the set of all central vertices in linear time. Consequently, our algorithms do not compute the distance matrix of the input graph.

Our first lemma establishes another property of 2LexBFS schemes of claw-free AT-free graphs.

Lemma 10. *Let $L_0 = \{x\}$, $L_1 = N(x), \ldots, L_r$ be the levels of a 2LexBFS scheme of a claw-free AT-free graph G. Then $\alpha(G[L_i \cup L_{i+1}]) \leqslant 2$ for all $i \geqslant 0$. Each vertex $v \in L_1$ has a neighbor in L_2 unless $L_1 \subseteq N[v]$.*

Proof. By Lemma 6, each level L_i, $i \neq 1$, is a clique of G and $\alpha(G[L_1]) \leqslant 2$. Thus, we obtain $\alpha(G[L_i \cup L_{i+1}]) \leqslant 2$ for all $i \neq 1$ and $\alpha(G[L_1 \cup L_2]) \leqslant 3$.

Suppose that $G[L_1 \cup L_2]$ contains an independent set $\{a, b, c\}$. Since L_2 is a clique we get $|L_2 \cap \{a, b, c\}| = 1$. Hence we may assume, without loss of generality, $a, b \in L_1$ and $c \in L_2$. Clearly c has a neighbor $y \in L_1$. Thus, $\alpha(G[L_1]) \leqslant 2$ implies either $\{y, a\} \in E$ or $\{y, b\} \in E$ but not both, since otherwise $\{y, a, b, c\}$ would induce a claw. Suppose both a and b have a neighbor in L_2, called a' and b' (possibly $a' = b'$), respectively. Since L_2 is a clique we obtain that $G[\{a, b, c, x, y, a', b'\}]$ is a graph of diameter 2. But graphs of diameter 2 containing an independent triple, $\{a, b, c\}$ in our case, always contain either a claw or an AT (as shown in the proof of Theorem 7), a contradiction.

Hence the only remaining case to consider is, without loss of generality, that a has no neighbor in L_2. Thus, $a \in L_1$, $N[a] \subset L_1 \cup \{x\}$ and $b \in L_1 \setminus N[a]$. Consequently, $N[a] \subset N[x]$. We claim that there is no vertex w of G for which x will be numbered last by LexBFS(G, w). To see this, assume that for some vertex w of G an execution of LexBFS(G, w) numbers vertex x last. Clearly, $N[a] \subseteq N[x] \setminus \{b\}$ implies that whenever LexBFS(G, w) appends a number to label(a) it also appends this number to label(x). If LexBFS(G, w) assigns a number to a vertex $z \in N[x] \setminus N[a]$ before it assigns numbers to x or a then label(x) becomes lexicographically larger than label(a) and consequently x will be numbered before a, a contradiction. Therefore, LexBFS(G, w) has to assign a number to a before it assigns a number to b. This implies that, during the run of LexBFS(G, w), before LexBFS(G, w) chooses a to be numbered next label$(a) =$ label(x) holds. Consequently, LexBFS(G, w) appends the number assigned to a to label(x) but not to label(b). Thus, from that point on, label(x) is lexicographically larger than label(b) and x cannot be numbered last by LexBFS(G, w), a contradiction. It follows that there is no vertex w of G such that x can be numbered last by an execution of LexBFS(G, w). Thus, no 2LexBFS ordering σ_2 of G yields $\sigma_2(x) = n$, a contradiction. Consequently, $\alpha(G[L_1 \cup L_2]) \leqslant 2$.

In a similar way it can be shown that, if a vertex $v \in L_1$ has no neighbor in L_2 then v is adjacent to all vertices of L_1. □

A few new concepts will turn out to be quite useful. From now on let G be a connected claw-free AT-free graph. Let $L_0 = \{x\}$, $L_1 = N(x), \ldots, L_r$ be the levels of a 2LexBFS scheme of G.

Definition. We call v a *predecessor* of w if there is a path $v = u_i, u_{i+1}, \ldots, u_k = w$ such that $u_j \in L_j$ for $j = i, i + l, \ldots, k$ and $i < k$. For every vertex $w \in L_i$, $1 \leqslant i \leqslant r$, we define $N^{\uparrow}(w) = N(w) \cap L_{i-1}$. Similarly, for every vertex $w \in L_i$, $0 \leqslant i \leqslant r - 1$, we define $N_{\downarrow}(w) = N(w) \cap L_{i+1}$. Consequently, $d^{\uparrow}(w) = |N^{\uparrow}(w)|$ and $d_{\downarrow}(w) = |N_{\downarrow}(w)|$. We

call $v \in L_i$, $0 \leqslant i \leqslant r$, a *plummet* if $d(v,u) = r - i$ for all $u \in L_r$. Note in particular, that x is a plummet.

Define $A = \{u \in L_{r-1}: d_{\downarrow}(u) = |L_r|\}$ and $B = L_{r-1} \setminus A$. Let $B_{\max}$ be the set of vertices v from B that have a maximal down-neighborhood $N_{\downarrow}(v)$ (with respect to set inclusion) among all vertices in B.

It is known that an AT-free graph having a 2LexBFS scheme with levels $L_0, L_1, \ldots, L_r$ has either $\operatorname{diam}(G) = r$ or $\operatorname{diam}(G) = r + 1$ (see [10]). Furthermore, a theorem on AT-free graphs without h-ladder and h^*-ladder given in [10] implies that the diameter of claw-free AT-free graphs can be computed by 2LexBFS in linear time.

Lemma 11. *Let $L_0, L_1, \ldots, L_r$ be the levels of a 2LexBFS scheme of a claw-free AT-free graph G. Let $u \in L_i$ and $v \in L_j$ with $i, j \in \{0, 1, \ldots, r\}$* and *$i \neq j$. Then $d(u,v) \in \{|i-j|, |i-j|+1\}$. Furthermore,* $\operatorname{diam}(G) = r$.

Proof. Let $u \in L_i$, $v \in L_j$, and without loss of generality assume that $i < j$. The properties of LexBFS imply both $d(u,v) \geqslant j - i$ and the existence of a path $x = v_0, v_1, \ldots, v_j = v$ in G with $v_k \in L_k$ for all $k = 0, 1, \ldots, j$.

If $i = 0$ the claim is trivial, since for all $j \geqslant 1$ and all vertices $v \in L_j$ it holds that $d(x,v) = j$. If $i > 1$ then $\{u, v_i\} \in E$ according to Lemma 6. Hence $u, v_i, v_{i+1}, \ldots, v_j = v$ is a path in G implying that $d(u,v) \leqslant j - i + 1$. If $i = 1$ then according to Lemma 10 either $v_1 \in N[u]$ or u has a neighbor $z \in L_2$. Hence either $u, v_1, v_2, \ldots, v_j = v$ or $u, z, v_2, \ldots, v_j = v$ are paths in G. Thus, $d(u,v) \leqslant j - i + 1$ if $i = 1$. This completes the proof for the distance claim.

Note that $d(x,v) = r$ for all $v \in L_r$ by definition of L_r. Hence together with the above proven claim we have $d(u,v) \leqslant r$ for all $u \in L_i$ and $v \in L_j$ where $i \neq j$. In order to prove $\operatorname{diam}(G) = r$ it remains to show that $d(u,v) \leqslant r$ for all $u, v \in L_i$, $i \geqslant 1$. Clearly, if $r \geqslant 2$ this is trivial since all vertices in L_1 are neighbors of the vertex in L_0 and all L_i, $i \geqslant 2$, are cliques. If $r = 1$, then Lemma 10 implies $d(u,v) = 1$ for all $u, v \in L_1 = L_r$. Consequently, G is complete and $\operatorname{diam}(G) = 1$. This shows that $\operatorname{diam}(G) = r$. □

We will now turn to the plummet vertices in a 2LexBFS scheme. The notion of a plummet will be a useful tool to characterize the central vertices in a claw-free AT-free graph. First we state some helpful lemmas. Recall that $A = \{u \in L_{r-1}: d_{\downarrow}(u) = |L_r|\}$, $B = L_{r-1} \setminus A$ and that $B_{\max}$ is the set of all those vertices $v \in B$ that have a maximal down-neighborhood $N_{\downarrow}(v)$ among all vertices in B.

Lemma 12. 1. *Let $u, v \in L_i$, $i \in \{1, \ldots, r-1\}$ such that $N_{\downarrow}(u)$ and $N_{\downarrow}(v)$ are incomparable (with respect to set inclusion). Then $N^{\uparrow}(u) = N^{\uparrow}(v)$.*

2. *If $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$ then $N^{\uparrow}(u) = N^{\uparrow}(u')$ for all vertices $u, u' \in B_{\max}$.*

Proof. Regarding part 1, note that the claim is trivial for $i = 1$. Let $u, v \in L_i$, $i \geqslant 2$, such that $N_{\downarrow}(u)$ and $N_{\downarrow}(v)$ are incomparable and suppose that $N^{\uparrow}(u) \neq N^{\uparrow}(v)$. Without

loss of generality assume that there exists a vertex $a \in L_{i-1}$ such that $a \in N^{\uparrow}(u) \setminus N^{\uparrow}(v)$. Since $N_{\downarrow}(u)$ and $N_{\downarrow}(v)$ are incomparable there also exists a vertex $b \in L_{i+1}$ such that $b \in N_{\downarrow}(u) \setminus N_{\downarrow}(v)$. But clearly $\{u, a, v, b\}$ induces a claw with center u, a contradiction.

Regarding part 2, note that under our assumption, namely, $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$, the already proven claim of part 1 gives $N^{\uparrow}(u) = N^{\uparrow}(u')$ for vertices $u, u' \in B_{\max}$ such that $N_{\downarrow}(u)$ and $N_{\downarrow}(u')$ are incomparable. Due to the definition of $B_{\max}$ it remains to show that $N^{\uparrow}(u) = N^{\uparrow}(u')$ also holds for vertices $u, u' \in B_{\max}$ such that $N_{\downarrow}(u) = N_{\downarrow}(u')$. This can be seen as follows: Let $u, u' \in B_{\max}$ such that $N_{\downarrow}(u) = N_{\downarrow}(u')$. Since $B_{\max} \subseteq L_{r-1} \setminus A$ and $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$ there exists a vertex $v \in B_{\max}$ such that $N_{\downarrow}(u)$ and $N_{\downarrow}(v)$ are incompatible, and thus also $N_{\downarrow}(u')$ and $N_{\downarrow}(v)$, are incomparable. It follows from part 1, that $N^{\uparrow}(u) = N^{\uparrow}(v) = N^{\uparrow}(u')$. □

Lemma 13. *Let $r \geqslant 2$, $u \in B_{\max}$, and $a \in L_{r-2}$ such that $a \notin N^{\uparrow}(u)$ and $a \notin N^{\uparrow}(v)$ for all $v \in A$. Then a is not a plummet.*

Proof. Let $r \geqslant 2$ and $u \in B_{\max}$. Clearly, $L_r \neq N_{\downarrow}(u)$ and hence there exists a vertex $b \in L_r$ such that $b \notin N_{\downarrow}(u)$.

Let $a \in L_{r-2}$ such that $a \notin N^{\uparrow}(u)$ and $a \notin N^{\uparrow}(v)$ for all $v \in A$. Suppose that a is a plummet. Hence $d(a, b) = 2$. This implies that there exists a vertex $w \in L_{r-1}$ such that $\{a, w\} \in E$ and $\{w, b\} \in E$. According to our assumptions $w \notin A$ and $w \neq u$. But this implies that $\{w, a, u, b\}$ induces a claw with center vertex w, a contradiction. □

The next theorem gives a characterization of all plummet vertices of a 2LexBFS scheme of G. We will later show that essentially only the plummets from certain levels of a 2LexBFS scheme of a claw-free AT-free graph are central vertices. Also, it follows from the characterization of the plummets we give below that the radius of a claw-free AT-free graph is indeed roughly half the diameter.

Theorem 14. *A vertex $v \in L_i$, $0 \leqslant i \leqslant r$, is plummet if and only if one of the following holds*:

1. $i = r$ *and* $L_r = \{v\}$.
2. $i = r - 1$ *and* $d_{\downarrow}(v) = |L_r|$.
3. $i = r - 2$, $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$, *and* $v \in N^{\uparrow}(u)$ *for a vertex* $u \in B_{\max}$.
4. $i \leqslant r - 2$ *and v is predecessor of a plummet* $u \in L_{r-2} \cup L_{r-1}$.

Proof. Let $v \in L_i$, $i \geqslant 0$. The claim is immediate for $i \in \{r - 1, r\}$.

Suppose $i = r - 2$. It is obvious that every $v \in N^{\uparrow}(u)$ for a vertex $u \in A$ is a plummet. Let $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$ and let $v \in N^{\uparrow}(u)$ for a vertex $u \in B_{\max}$. It follows from Lemma 12 that for all vertices $u, u' \in B_{\max}$, $N^{\uparrow}(u) = N^{\uparrow}(u')$. Thus, v is not only a predecessor of u but of all $u' \in B_{\max}$. Thus, v is a plummet, since $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$. It remains to show that no other plummets exist in L_{r-2}. So suppose that $a \in L_{r-2}$ is a plummet

and $a \notin N^{\uparrow}(u)$ for all $u \in A \cup B_{\max}$. Hence a satisfies the assumption of Lemma 13 and thus cannot be a plummet, a contradiction.

Suppose $i \leqslant r-3$. Clearly, each predecessor of a plummet from L_{r-2} is a plummet. It remains to show that there are no other plummets in L_i. Suppose that a is a plummet in L_i, $i \leqslant r-3$. We distinguish two cases. The first case is that $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) \neq L_r$ holds, i.e., the only plummets in L_{r-2} are predecessors of vertices in A. In that case there exists a vertex $b \in L_r$ satisfying $N^{\uparrow}(b) = A$. But then a being a plummet implies $d(a,b) = r-i$ which in turn implies a must be a predecessor of a vertex $u \in A$. The second case is that $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$ holds, i.e., there are plummets in L_{r-2} not being predecessors of vertices in A. We have already shown that those plummets in L_{r-2} have to be predecessors of some vertex (and thus also of all vertices) $u \in B_{\max}$. Let $u \in B_{\max}$. Since $B_{\max} \subseteq B = L_{r-1} \setminus A$ there exists a vertex $b \in L_r$ such that $b \notin N_{\downarrow}(u)$. Now suppose that a though being a plummet is not a predecessor of any $v \in A \cup B_{\max}$. Hence a is not a predecessor of u, but since a is a plummet, $d(a,b) = r-i$. It follows that there have to exist vertices $a' \in L_{r-2}$ and $a'' \in L_{r-1}$ such that (i) $d(a,a') = r-i-2$ and (ii) $\{a', a''\}, \{a'', b\} \in E$ and (iii) neither a' nor a'' are plummets (since otherwise a would be a predecessor of a plummet from L_{r-2}). In particular, $\{a', u\} \notin E$. Hence $\{a'', a', u, b\}$ induces a claw with center vertex a'', a contradiction.

This completes the proof of our theorem. □

An immediate consequence of the above theorem is that the radius of a claw-free AT-free graph can be computed in linear time.

Corollary 15. *Let G be a connected claw-free AT-free graph. Then either $r(G) = \lceil \mathrm{diam}(G)/2 \rceil$ or $r(G) = \mathrm{diam}(G) = 2$. Furthermore, the radius of a claw-free AT-free graph can be computed in linear time by 2LexBFS.*

Proof. Let $L_0 = \{x\}$, $L_1, \ldots, L_r$ be the levels of a 2LexBFS scheme of a connected claw-free AT-free graph G. Then Lemma 11 implies $r = \mathrm{diam}(G)$. By Theorem 14 each level L_i contains a plummet vertex if $0 \leqslant i \leqslant r-2$.

Suppose $r = \mathrm{diam}(G) \geqslant 3$. Observe that for all $v \in V$, $e(v) \geqslant \lceil r/2 \rceil$. So we show $r(G) = \lceil \mathrm{diam}(G)/2 \rceil$ by proving that every plummet $z \in L_{\lfloor r/2 \rfloor}$ satisfies $e(z) = \lceil r/2 \rceil$. To see this consider a plummet $z \in L_{\lfloor r/2 \rfloor}$. By Lemma 11, for every $v \in L_j$ with $1 \leqslant j \leqslant r-1$, $d(z,v) \leqslant |\lfloor r/2 \rfloor - j| + 1 \leqslant \lceil r/2 \rceil$. Also, $d(z,x) = \lfloor r/2 \rfloor$ since $z \in L_{\lfloor r/2 \rfloor}$, and $d(z,w) = r - \lfloor r/2 \rfloor = \lceil r/2 \rceil$ for all $w \in L_r$ since z is a plummet. Hence $e(z) = \lceil r/2 \rceil$.

Finally if $\mathrm{diam}(G) = 2$ then either G has a vertex adjacent to all other vertices of G, and thus $r(G) = 1$, or $r(G) = 2$. The case $\mathrm{diam}(G) \leqslant 1$ is trivial.

From the above case analysis it is not hard to see that the radius of a claw-free AT-free graph can be computed in linear time. □

Theorem 16. *There exists a linear-time algorithm that given a 2LexBFS scheme of a claw-free AT-free graph computes all plummet vertices.*

Proof. Let G be a claw-free AT-free graph. Let $L_0 = \{x\}$, $L_1 = N(x), \ldots, L_r$ be the levels of a 2LexBFS scheme. The algorithm given below computes a function $g : V \to \{0, 1\}$ such that a vertex v of G is a plummet if and only if $g(v) = 1$.

Procedure PLUMMET

1. Initialize $g(v) = 0$ for all $v \in V$.
2. Set $g(x) = 1$. If $L_r = \{v\}$ then set $g(v) = 1$.
3. If $r = 2$ then set $g(v) = 1$ for all $v \in L_1$ such that $d_{\downarrow}(v) = |L_r|$.
4. If $r > 2$ then for $i = r - 1$ downto 0 do
   (a) If $i = r - 1$ set $g(v) = 1$ for all $v \in L_i$ such that $d_{\downarrow}(v) = |L_r|$. Let $\hat{n}$ be the number of such vertices.
   (b) If $i = r - 2$
      i. Set $g(v) = 1$ for all $v \in L_i$ such that $v \in N^{\uparrow}(u)$ for some vertex $u \in L_{r-1}$ such that $g(u) = 1$.
      ii. Compute B and determine a vertex with highest down-degree among all vertices in B, call it $u_{\max}$. Note that $u_{\max} \in B_{\max}$. It follows from Theorem 14 that all predecessors of $u_{\max}$ are plummets if and only if $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$. The latter condition can be checked by simply verifying that every vertex in L_r has up-degree greater than $\hat{n}$.
      iii. If, for all $u \in L_r$, $d^{\uparrow}(u) > \hat{n}$ then set $g(v) = 1$ for all vertices $v \in L_{r-2}$ such that $v \in N^{\uparrow}(u_{\max})$.
   (c) If $i \leqslant r-3$ then set $g(v) = 1$ for all $v \in L_i$ such that $v \in N^{\uparrow}(u)$ for some vertex $u \in L_{i+1}$ satisfying $g(u) = 1$.

It is not hard to verify that our algorithm runs in linear time. The correctness of the algorithm follows directly from Theorem 14. Regarding step 4.b note that, if $\bigcup_{u \in B_{\max}} N_{\downarrow}(u) = L_r$, then $N^{\uparrow}(u) = N^{\uparrow}(u')$ for all vertices $u, u' \in B_{\max}$ (see Lemma 12) and hence computing the predecessors of a particular vertex $u_{\max} \in B_{\max}$ is as good as computing the predecessors of all vertices from $B_{\max}$. □

Now we are prepared to state the main theorem of this section, a characterization of all central vertices in a claw-free AT-free graph.

Theorem 17. *Let $L_0 = \{x\}$, $L_1 = N(x), \ldots, L_r$ be the levels of a 2LexBFS scheme of a claw-free AT-free graph G. Then a vertex v of G is a central vertex of G if and only if one of the following holds*:

1. $r = 0$ *or* $r = 1$.
2. $r = 2$ *and* $d(v) = n - 1$.
3. $r = 2$ *and* $d(u) < n - 1$ *for all* $u \in V$.
4. $r = 2k$, *for some integer* $k \geqslant 2$, $v \in L_k$, *and* v *is a plummet*.
5. $r = 2k + 1$, *for some* $k \geqslant 1$, $v \in L_k$ *and* v *is a plummet, or* $v \in L_{k+1}$.

Proof. Cases 1, 2, and 3 do not require further explanation. If $r = 2k$, $k \geqslant 2$, then according to Lemma 11 and Corollary 15 we know that $r(G) = k$. This implies that

a vertex v is a central vertex if and only if $d(u,v) \leqslant k$ for all $u \in V$. Thus, central vertices can only be within L_k. Furthermore recall that for vertices $v \in L_k$ and $u \in L_i,\ i \neq k,\ d(u,v) \in \{|i-k|, |i-k|+1\}$, see Lemma 11. Hence for a vertex $v \in L_k$ to be a central vertex it suffices to satisfy $d(x,v) \leqslant k$ and $d(u,v) \leqslant k$ for all $u \in L_r$. Since $d(x,v) = k$ trivially holds for all $v \in L_k$, we obtain that a vertex in L_k is a central vertex if and only if it is a plummet. If $r = 2k+1,\ k \geqslant 1$, then according to Lemma 11 and Corollary 15 we know that $r(G) = k+1$. This implies that a vertex v is a central vertex if and only if $d(u,v) \leqslant k+1$ for all $u \in V$. Thus, central vertices can only be within L_k or L_{k+1}. In light of Lemma 11, a vertex $v \in L_k \cup L_{k+1}$ is a central vertex if and only if $d(x,v) \leqslant k+1$ and $d(u,v) \leqslant k+1$ for all $u \in L_r$. Observe that all vertices in L_{k+1} satisfy that condition. For the vertices $v \in L_k$ we trivially have $d(x,v) = k$ and thus a vertex in L_k is a central vertex if and only if it is a plummet. This completes the proof. □

Now we are prepared to state the linear-time algorithm for computing all central vertices of a claw-free AT-free graph.

Theorem 18. *There exists a linear-time algorithm for computing all central vertices of a claw-free AT-free graph.*

Proof. Let $G = (V,E)$ be a claw-free AT-free graph. The algorithm computes a function $cen : V \to \{0,1\}$ such that a vertex v of G is central if and only if $cen(v) = 1$.

Algorithm ALL CENTRAL VERTICES

1. Compute a 2LexBFS scheme of G. Let $L_0 = \{x\},\ L_1 = N(x), \ldots, L_r$ be its levels.
2. Initialize $cen(v) = 0$ for all $v \in V$.
3. If $r = 0$ or $r = 1$ set $cen(v) = 1$ for all $v \in V$.
4. If $r = 2$ then set $cen(v) = 1$ for all vertices v such that $d(v) = n-1$. If no such vertex exists then set $cen(v) = 1$ for all $v \in V$.
5. If $r = 2k,\ k \geqslant 2$, then by Theorem 17 the set of all central vertices consists exactly of all plummets in L_k. Call PLUMMET and set $cen(v) = 1$ for all $v \in L_k$ such that $g(v) = 1$.
6. If $r = 2k+1,\ k \geqslant 1$, then by Theorem 17 the set of all central vertices contains exactly all vertices in L_{k+1} and all plummets in L_k. Set $cen(v) = 1$ for all $v \in L_{k+1}$. Call PLUMMET and set $cen(v) = 1$ for all $v \in L_k$ such that $g(v) = 1$.

Since a 2LexBFS scheme can be computed in linear time and also PLUMMET runs in linear time it is not hard to see that our algorithm ALL CENTRAL VERTICES runs in linear time. □

5. Domination, independence, and coloring

We show that for claw-free AT-free graphs there exist linear-time algorithms to compute a minimum dominating set and maximum independent set. We also show that

there exists an $O(\sqrt{n}m)$ algorithm for coloring a given claw-free AT-free graph with minimum number of colors. All algorithms exploit the properties of a 2LexBFS scheme of claw-free AT-free graphs.

The following lemma shows that the total number of potential edges between consecutive levels of a 2LexBFS scheme of a claw-free AT-free graph G is linearly bounded by the number of edges in G. This observation will be useful when proving the running time of our upcoming algorithms for computing a maximum independent set and an optimal coloring for claw-free AT-free graphs.

Lemma 19. *Let $G=(V,E)$ be a connected claw-free AT-free graph and $L_0, L_1, L_2, \ldots, L_r$ be the levels of a 2LexBFS scheme of G.*

Let $E^* = \{\{u,v\}: u \in L_i$ and $v \in L_{i+1}$ for some $i \in \{1,\ldots,r-1\}\}$. Then $|E^*| = O(|E|)$. Furthermore, the number of edges of $G[L_1]$ is $\Theta(|L_1|^2)$.

Proof. Let $n_i = |L_i|$, $0 \leqslant i \leqslant r$. Turan's theorem (see e.g. [29]) applied to the triangle-free graph $\overline{G}[L_1]$ implies that $\overline{G}[L_1]$ has at most ${n_1}^2/4$ edges. Hence the number of edges of $G[L_1]$ is $\Theta({n_1}^2)$.

Combined with Lemma 6 we obtain

$$|E| \geqslant \frac{{n_1}^2}{4} + \sum_{i=2}^{r} \binom{n_i}{2}.$$

On the other hand,

$$|E^*| = \sum_{i=2}^{r} (n_i n_{i-1}).$$

Clearly for all $2 \leqslant i \leqslant r$

$$n_i n_{i-1} \leqslant \frac{{n_i}^2 + {n_{i-1}}^2}{2}$$

and thus leading to

$$\begin{aligned}
\sum_{i=2}^{r} (n_i n_{i-1}) &\leqslant \sum_{i=2}^{r} \frac{{n_i}^2 + {n_{i-1}}^2}{2} \\
&= \frac{{n_1}^2}{2} + \frac{{n_r}^2}{2} + \sum_{i=2}^{r-1} {n_i}^2 \\
&\leqslant 2\left(\frac{{n_1}^2}{4} + \sum_{i=2}^{r} \binom{n_i}{2}\right) + \sum_{i=2}^{r} n_i \\
&\leqslant 2|E| + |V| \\
&= O(|E|)
\end{aligned}$$

Hence $|E^*| = O(|E|)$ which completes the proof. □

5.1. Dominating set and independent dominating set

A minimum dominating set in AT-free graphs can be computed in time $O(n^6)$ [21], while a minimum independent dominating set can be computed in time $O(n^4)$ [6]. In contrast, the decision problems DOMINATING SET and INDEPENDENT DOMINATING SET remain NP-complete when restricted to claw-free graphs. This is due to the fact that both problems are NP-complete on line graphs since EDGE DOMINATING SET and EDGE INDEPENDENT DOMINATING SET are NP-complete [30].

Lemma 20. *Let $G=(V,E)$ be a claw-free AT-free graph and $L_0,L_1,\ldots,L_r$ be the levels of a 2LexBFS scheme of G. Let $u\in L_i$ and $v\in L_{i+2}$, $i\in\{0,1,\ldots,r-2\}$ such that u and v have a common neighbor $w\in L_{i+1}$. Then $L_{i+1}\subseteq(N[u]\cup N[v])$.*

Proof. The statement is trivial for $i=0$. Otherwise suppose it is false for some $i\geqslant 1$. Let $z\in(L_{i+1}\setminus(N[u]\cup N[v]))$. Then $\{w,u,z,v\}$ induces a claw in G, a contradiction. □

Thus, for example, there is a dominating set of G containing exactly one vertex of every level L_i with i even or $i=r$. On the other hand, for every dominating set D of G and for every three consecutive levels L_i,L_{i+1} and L_{i+2} at least one of these levels must contain a vertex of D. Notice that for $u\in L_i$ and $v\in L_{i+3}$, $L_{i+1}\cup L_{i+2}\subseteq N[u]\cup N[v]$ if and only if $N_{\downarrow}(u)=L_{i+1}$ and $N^{\uparrow}(v)=L_{i+2}$.

Lemma 21. *Let $G=(V,E)$ be a claw-free AT-free graph and $L_0,L_1,\ldots,L_r$ be the levels of a 2LexBFS scheme of G. Let $u\in L_i$ and $v\in L_{i+2}$, $i\in\{0,1,\ldots,r-2\}$. Let $x_t,x_{t-1},\ldots,x_{t'}$, $t\geqslant t'$, be the ordering of the vertices from L_{i+1} induced by the 2LexBFS ordering of G. Then it holds that for all j and k, $t'\leqslant j\leqslant t$ and $t'\leqslant k\leqslant t$, if $x_j\in N^{\uparrow}(v)\setminus N_{\downarrow}(u)$ and $x_k\in N_{\downarrow}(u)\setminus N^{\uparrow}(v)$ then $k>j$.*

Proof. The claim for $i=0$ is trivial. So suppose $i\geqslant 1$. Let $u\in L_i$ and $v\in L_{i+2}$. Assume $j>k$ for some $x_j\in N^{\uparrow}(v)\setminus N_{\downarrow}(u)$ and some $x_k\in N_{\downarrow}(u)\setminus N^{\uparrow}(v)$. According to LexBFS and our assumption that $j>k$, $\{x_k,u\}\in E$, and $\{x_j,u\}\notin E$ there exists a vertex u' with higher index than u in the 2LexBFS ordering such that $\{x_k,u'\}\notin E$ and $\{x_j,u'\}\in E$. Hence $u'\in L_i$. Since L_{i+1} is a clique we also have $\{x_j,x_k\}\in E$ and consequently $\{x_j,u',x_k,v\}$ induces a claw in G, a contradiction. □

Lemma 22. *Let $G=(V,E)$ be a claw-free AT-free graph and $L_0,L_1,\ldots,L_r$ be the levels of a 2LexBFS scheme of G. Let $x_t,x_{t-1},\ldots,x_{t'}$, $t\geqslant t'$, be the ordering of the vertices from L_{i+1} induced by the 2LexBFS ordering of G. Let $u\in L_i$ and $v\in L_{i+2}$, $i\in\{0,1,\ldots,r-2\}$, with no common neighbor such that $L_{i+1}\subseteq(N[u]\cup N[v])$. Then there is an index j with $t\geqslant j>t'$ such that $N_{\downarrow}(u)=\{x_t,x_{t-1},\ldots,x_j\}$ and $N^{\uparrow}(v)=\{x_{j-1},x_{j-2},\ldots,x_{t'}\}$.*

Proof. Consider $u \in L_i$ and $v \in L_{i+2}$ with $L_{i+1} \subseteq (N[u] \cup N[v])$ and $N_\downarrow(u) \cap N^\uparrow(v) = \emptyset$. By Lemma 21 there is no vertex of $N^\uparrow(v)$ between two vertices of $N_\downarrow(u)$ and there is no vertex of $N_\downarrow(u)$ between two vertices of $N^\uparrow(v)$ in the 2LexBFS ordering of L_{i+1}. Hence there is an index j with $t \geqslant j > t'$ such that one of the two sets $\{x_t, x_{t-1}, \ldots, x_j\}$ and $\{x_{j-1}, x_{j-2}, \ldots, x_{t'}\}$ equals $N_\downarrow(u)$ and the other one equals $N^\uparrow(v)$.

If $N^\uparrow(v) = \{x_t, t_{t-1}, \ldots, x_j\}$ and $N_\downarrow(u) = \{x_{j-1}, x_{j-2}, \ldots, x_{t'}\}$ then $x_j \in N^\uparrow(v) \setminus N_\downarrow(u)$ and $x_{j-1} \in N_\downarrow(u) \setminus N^\uparrow(v)$, contradicting Lemma 21. Hence $N_\downarrow(u) = \{x_t, x_{t-1}, \ldots, x_j\}$ and $N^\uparrow(v) = \{x_{j-1}, x_{j-2}, \ldots, x_{t'}\}$. □

Our algorithm for computing a minimum dominating set of claw-free AT-free graphs consists of two main parts. The first part is a preprocessing that itself is divided into two phases. In the first phase a 2LexBFS scheme with levels $L_0, L_1, \ldots, L_r$ of the input graph G is computed. Recall that a 2LexBFS scheme induces an ordering of the vertices of each level, i.e., we assume the vertices of each level to be ordered from highest to lowest assigned number. In the second phase of the preprocessing for every level L_i of the previously computed 2LexBFS scheme all vertices $v \in L_i$ satisfying one of the following conditions are determined:

1. $d_\downarrow(v) = |L_{i+1}|$ (*down-dominator*).
2. $d^\uparrow(v) = |L_{i-1}|$ (*up-dominator*).
3. There exists a j such that $N_\downarrow(v)$ equals the set of the first j vertices from L_{i+1} (*left-interval*).
4. There exists a k such that $N^\uparrow(v)$ equals the set of the last k vertices from L_{i-1} (*right-interval*).

The second and main part of our algorithm exploits the information obtained in the preprocessing while proceeding stepwise down the levels of the 2LexBFS scheme and assigning weights to vertices such that the weight of a vertex $v \in L_i$ corresponds to the minimum cardinality of an i-partial dominating set containing v. Informally, an i-partial dominating set is a set of vertices that dominates the vertices of $L_0 \cup L_1 \cup \cdots \cup L_i$ and contains no vertex from $L_{i+1} \cup L_{i+2} \cup \cdots \cup L_r$.

Theorem 23. *There is a linear-time algorithm for computing a minimum dominating set for claw-free AT-free graphs.*

Proof. Let $G = (V, E)$ be a claw-free AT-free graph. Let $L_0, L_1, \ldots, L_r$ be the levels of a 2LexBFS scheme of G.

For a vertex $v \in V \setminus L_r$ we denote the largest and smallest index of vertices in $N_\downarrow(v)$ by $l_\downarrow(v)$ and $r_\downarrow(v)$, respectively. Similarly, for a vertex $v \in V \setminus L_0$ we denote the largest and smallest index of vertices in $N^\uparrow(v)$ by $l^\uparrow(v)$ and $r^\uparrow(v)$, respectively. If $N_\downarrow(v) = \emptyset$ then we call v a *bottom-vertex* and set $l_\downarrow(v) = r_\downarrow(v) = 0$. Note that $N^\uparrow(v) \neq \emptyset$ for all $v \in V \setminus L_0$.

The algorithm we describe below in full detail computes the size of a minimum dominating set and may also be used for computing a minimum dominating set itself.

Algorithm DOMINATING SET

1. Compute a 2LexBFS scheme of the input graph $G=(V,E)$. Let $L_0=\{x\}$, $L_1=N(x),\ldots,L_r$ be its levels and $x=x_n,x_{n-1},\ldots,x_1$ be its 2LexBFS ordering.
2. Initialize for every vertex v of G: $weight(v)=\infty$, $updom(v)=0$, $downdom(v)=0$, $leftint(v)=0$, and $rightint(v)=0$.
3. For every vertex $v\in V\setminus L_r$ compute $d_{\downarrow}(v)$, $l_{\downarrow}(v)$, and $r_{\downarrow}(v)$. For every vertex $v\in V\setminus\{x\}$ compute $d^{\uparrow}(v)$, $l^{\uparrow}(v)$, and $r^{\uparrow}(v)$.
4. For every vertex $v\in L_i$, $0\leqslant i\leqslant r-1$, do
   (a) If $d_{\downarrow}(v)=|L_{i+1}|$ then set $downdom(v)=1$.
   (b) If $downdom(v)=0$, $l_{\downarrow}(v)=\max\{j: x_j\in L_{i+1}\}$, and $d_{\downarrow}(v)=l_{\downarrow}(v)-r_{\downarrow}(v)+1$ then set $leftint(v)=1$.
   (c) If $N_{\downarrow}(v)=\emptyset$ then set $leftint(v)=1$.
5. For every vertex $v\in L_i$, $1\leqslant i\leqslant r$, do
   (a) If $d^{\uparrow}(v)=|L_{i-1}|$ then set $updom(v)=1$.
   (b) If $r^{\uparrow}(v)=\min\{j: x_j\in L_{i-1}\}$ and $d^{\uparrow}(v)=l^{\uparrow}(v)-r^{\uparrow}(v)+1$ then set $rightint(v)=1$.
6. For all levels i, $0\leqslant i\leqslant r$, make an ordered list of all vertices $v\in L_i$ with $leftint(v)=1$ sorted by non-increasing values of $r_{\downarrow}(v)$, and an ordered list of all vertices $w\in L_i$ with $rightint(w)=1$ sorted by non-increasing values of $l^{\uparrow}(w)$.
7. For every vertex $u\in L_1$ such that $L_1\subseteq N[u]$, set $weight(u)=1$ and $i_0=1$.
8. If there is no vertex $u\in L_1$ with $L_1\subseteq N[u]$ then set $weight(x)=1$ and $i_0=0$.
9. For $i:=i_0$ downto $r-1$ do
   (a) Let A_i be the set of all vertices $v\in L_i$ of weight $w_i=\min\{weight(w): w\in L_i\}$.
   (b) For every vertex $u\in L_{i+1}$ update $weight(u):=\min\{weight(u),w_i+1\}$.
   (c) For every vertex $u\in L_{i+2}$ with $updom(u)=1$ update $weight(u):=\min\{weight(u),w_i+1\}$.
   (d) For every vertex $u\in L_{i+2}$ with a predecessor in A_i update $weight(u):=\min\{weight(u),w_i+1\}$.
   (e) Take the ordered list of all vertices $u\in L_i$ with $leftint(u)=1$ and the ordered list of all vertices $v\in L_{i+2}$ with $rightint(v)=1$. Passing both lists (from left to right) find all $v\in L_{i+2}$ in one list for which there is a $u\in A_i$ in the other list such that $r_{\downarrow}(u)+1=l^{\uparrow}(v)$ and update $weight(v):=\min\{weight(v),w_i+1\}$ for each vertex v found.
   (f) If there exists a vertex $v\in A_i$ with $downdom(v)=1$ set $weight(w)=w_i+1$ for every vertex $w\in L_{i+3}$ with $updom(w)=1$.
10. Output the minimum weight taken over all vertices of L_r and all vertices $v\in L_{r-1}$ satisfying $downdom(v)=1$ (and a corresponding dominating set constructed via pointers).

In the following proof of correctness the notion of a partial dominating set will be helpful. An i-partial dominating set, $i\geqslant 0$, is a set $D_i\subseteq V$ such that $N[D_i]\supseteq L_0\cup L_1\cup\cdots\cup L_i$ and D_i contains no vertex of $L_{i+1}\cup L_{i+2}\cup\cdots\cup L_r$. Note, if the algorithm assigns weight $weight(u)=t$ to a vertex $u\in L_i$ then there is an i-partial dominating set D_i containing u such that $|D_i|=t$.

Now let us turn to the correctness of the above algorithm. Steps 1–6 just contain the preprocessing. Observe that steps 7 and 8 assign a weight of 1 to all vertices that dominate $L_0 \cup L_1$, in other words vertices that form a minimum 1-partial dominating set. Clearly, x and also all $u \in L_1$ such that $L_1 \subseteq N[u]$ have that property. These vertices are potential start vertices of the to be constructed minimum dominating set.

Note that some vertices $v, w \in L_1$ might dominate $L_0 \cup L_1 \cup L_2$, and so be part of a minimum dominating set D' of G. However, for every $u' \in L_2$, the vertex sets $\{x, u'\}$ and $\{u, u'\}$ where $u \in L_1$ and $L_1 \subseteq N[u]$ also dominate $L_0 \cup L_1 \cup L_2$ and hence together with $D' \setminus \{v, w\}$ also form a minimum dominating set. Though our algorithm obviously misses the type of minimum dominating sets containing more than one vertex from L_1 it will not miss the type of minimum dominating set described in the previous sentence as can be seen by looking at steps 9b and 9d. In general, the i-partial dominating sets implicitly constructed by the algorithm contain at most one vertex per level of the 2LexBFS scheme.

Claim. *Let weight$(v) <$ weight(v') for some $v, v' \in L_i$, $i \geqslant 0$. If there exists a minimum dominating set containing v' but not v then there exists also a minimum dominating set containing v but not v'.*

To see this let $v, v' \in L_i$, $i \geqslant 0$. Suppose weight$(v) <$ weight(v'). Let D' be a minimum dominating set containing v' but not v and let $D_{v'} \subseteq D'$ be an i-partial dominating set of size weight(v') containing v'. Let D_v be an i-partial dominating set containing v but not v' of size weight(v). Such a set exists according to the way the weights are assigned in our algorithm. Observe that $|D_v \cup (D' \setminus D_{v'})| < |D'|$. Furthermore, for every $y \in L_{i+1}$, $D_v \cup (D' \setminus D_{v'}) \cup \{y\}$ is a dominating set of G of size less than or equal to the size of D'. Hence if there exists a minimum dominating set of G containing v' but not v then there exists also a minimum dominating set of G containing v but not v'.

The above claim justifies that for every level L_i only the vertices in L_i of minimal weight (forming the set A_i) have to be considered (see step 9). In step 9 the algorithm determines for every $v \in A_i$ those vertices in L_{i+1}, L_{i+2}, and L_{i+3} that together with v dominate all of L_{i+1}, $L_{i+1} \cup L_{i+2}$, and $L_{i+1} \cup L_{i+2} \cup L_{i+3}$, i.e., in steps 9b, 9c, 9d and 9e, and 9f, respectively. In other words, the algorithm determines all those vertices in L_{i+1}, L_{i+2}, and L_{i+3} such that each one of those vertices $u \in L_{i'}$, $i' \in \{i+1, i+2, i+3\}$, together with an i-partial dominating set containing v form an i'-partial dominating set.

Clearly, since L_{i+1}, $i \geqslant 1$, is a clique, any vertex $u \in L_{i+1}$ can be chosen. If $i = 0$, in particular if x has been chosen as start vertex (see step 8) then, since x dominates L_1, also any vertex in $L_1 = L_{i+1}$ can be chosen. This shows the correctness of step 9b. Step 9c needs no further explanation. The correctness of steps 9d and 9e rests on the Lemmas 20 and 22. Note in particular that the two lemmas give necessary and sufficient conditions for a vertex in L_{i+2} not being an up-dominator (a case that is handled in step 9c) to form together with an i-partial dominating set containing the corresponding vertex $v \in L_i$ and $i+2$-partial dominating set. The correctness of step 9f

is an immediate consequence of the observation that $v \in L_i$, $w \in L_{i+3}$, $downdom(v) = 1$ and $updom(w) = 1$ imply $L_{i+1} \cup L_{i+2} \subseteq N[\{v, w\}]$.

In light of the above, especially the above claim, our algorithm ensures that at least one vertex $v \in L_r$ or at least one vertex $v' \in L_{r-1}$ with $downdom(v') = 1$ gets assigned a weight that is equal to the size of a minimum dominating set.

Consider the running time. The preprocessing in steps 1–6 can be done in linear time. Computing 2LexBFS(G) is a linear-time algorithm and all other steps essentially require to inspect all neighbors of every vertex. Step 6 can be done in linear time using bucket sort. Steps 7 and 8 can be done in time $O(n)$. Steps 9a, 9b, and 9c can be done in time linear in $|L_i|$, $|L_{i+1}|$, and $|L_{i+2}|$, respectively. Step 9d can be done in time $O(\sum_{v \in L_{i+1}} d(v))$ by simply first marking all nodes in L_{i+1} that are neighbors of some $w \in A_i$ and then assigning weight $w_i + 1$ to all vertices in L_{i+2} that are neighbors of those marked vertices. Step 9e requires time $O(\max\{|L_i|, |L_{i+2}|\})$. Finally, step 9f can be done in time $O(|L_{i+3}|)$.

Hence the overall running time of our algorithm is $O(n + m)$.

It is not hard to see that one can also output a minimum dominating set in linear time by exploiting a suitable pointer structure. □

Though the minimum dominating set constructed by the previous algorithm does not need to be an independent set it has been shown that a minimum independent dominating set in claw-free graphs has the same size as a minimum dominating set [1].

Theorem 24 (Allan and Laskar [1]). *Every claw-free graph G has a minimum dominating set that is independent. Thus, $\gamma(G) = \gamma_i(G)$ for all claw-free graphs.*

In light of the above theorem it is not hard to see that our algorithm DOMINATING SET can be easily modified for computing a minimum independent dominating set in linear time. As already mentioned, our algorithm computes a minimum dominating set with at most one vertex per level of the 2LexBFS scheme. So the only edges in a returned dominating set may occur between vertices in consecutive levels. The sole source for that possibility can be found in step 9b. So one has to modify step 9b in such a way that for every vertex $v \in A_i$ only the vertices $u \in L_{i+1}$ that are *not adjacent* to v are assigned weight $w_i + 1$.

Corollary 25. *There is a linear-time algorithm for computing a minimum independent dominating set for claw-free AT-free graphs.*

5.2. Independent set

As mentioned earlier there exists an $O(n^4)$ algorithm for computing a maximum independent set for AT-free graphs [6]. There is also a polynomial-time algorithm for

computing a maximum independent set for claw-free graphs [23,26]. Notice that the latter problem is a generalization of the well-known MATCHING problem.

Let $G=(V,E)$ be a claw-free AT-free graph and $L_0, L_1, \ldots, L_r$ be the levels of a 2LexBFS scheme of G. Let I be any independent set of G. Then Lemma 6 and Lemma 10 imply $|L_i \cap I| \leqslant 1$ if $i \neq 1$, $|L_1 \cap I| \leqslant 2$ and $|(L_{i-1} \cup L_i) \cap I| \leqslant 2$. Using this observation it can be shown that the search for a maximum independent set can be restricted to independent sets containing at most one vertex per level L_i, for all $i \geqslant 2$. This leads to a longest path problem on an auxiliary directed acyclic graph $\vec{G} = (\{s,t\} \cup \{d_3, d_4, \ldots, d_r\} \bigcup \cup_{i=2}^{r} L_i, \vec{E})$, where

$$\begin{aligned}
\vec{E} = {} & \{(v_i, v_{i+1}) \colon v_i \in L_i, v_{i+1} \in L_{i+1}, \{v_i, v_{i+1}\} \notin E,\ 2 \leqslant i \leqslant r-1\} \\
& \cup \{(v_i, d_{i+1}) \colon v_i \in L_i, N_{\downarrow}(v_i) = L_{i+1},\ 2 \leqslant i \leqslant r-1\} \\
& \cup \{(d_i, v_{i+1}) \colon v_{i+1} \in L_{i+1},\ 3 \leqslant i \leqslant r-1\} \\
& \cup \{(s,v) \colon v \in L_2\} \\
& \cup \{(s,v) \colon v \in L_3 \text{ and } L_1 \text{ is not a clique}\} \\
& \cup \{(v,t) \colon v \in \{d_r\} \cup L_r\}.
\end{aligned}$$

Now we assign a length $\ell(a) \in \{0,1,2,3\}$ to each directed edge a such that $\ell(a)=3$ if $a=(s,v)$ with $v \in L_3$, $\ell(a)=2$ if $a=(s,v)$ with $v \in L_2$, $\ell(a)=0$ if $a=(v_i, d_{i+1})$ with $v_i \in L_i$ and $2 \leqslant i \leqslant r-1$, or if $a=(v,t)$, and $\ell(a)=1$ otherwise.

For a given claw-free AT-free graph G our algorithm computes first a 2LexBFS scheme and then the auxiliary directed acyclic graph $\vec{G}$ and the length function ℓ. Then a longest s,t-path on $\vec{G}$ is computed. Its length is exactly $\alpha(G)$ and the vertex set of a longest path corresponds to a maximum independent set of G.

Theorem 26. *There is a linear-time algorithm for computing a maximum independent set for claw-free AT-free graphs.*

Proof. First, consider the correctness of our above outlined algorithm. Any directed path in $\vec{G}$ corresponds to an independent set in G since all directed edges in $\vec{G}$ join non-adjacent vertices in G and since the LexBFS scheme guarantees that the set of vertices of a path not belonging to $\{s,t\} \cup \{d_3, d_4, \ldots, d_r\}$ is an independent set in G. Furthermore the length of an s,t-path is exactly the cardinality of the corresponding independent set when we carefully treat the edges (s,v). Namely, to an edge $(s,v) \in \vec{E}$ with $v \in L_2$ corresponds an independent set containing x and v (and thus $\ell((s,v))=2$), and to an edge $(s,v) \in \vec{E}$ with $v \in L_3$ corresponds an independent set containing y_1, y_2, and v where y_1 and y_2 are non-adjacent vertices of L_1 (and thus $\ell((s,v))=3$). Hence the length of an edge a equals the number of vertices contributed by a to an independent set corresponding to an s,t-path containing a. Finally, it is not hard to see that Lemma 6 and Lemma 10 imply that every maximum independent set of G is represented by an s,t-path of $\vec{G}$.

Consider the running time. The graph $\vec{G}$ has $O(|V|)$ vertices and $O(|E^*|)$ edges. Thus, by Lemma 19 the size of $\vec{G}$ is $O(|V|+|E|)$. It is not hard to see that $\vec{G}$ can be constructed in time linear in the size of $\vec{G}$. Using a linear-time algorithm for computing a longest path in a directed acyclic graph we obtain overall running time $O(|V|+|E|)$. □

5.3. Coloring

The COLORING problem is NP-complete for claw-free graphs since it remains NP-complete on line graphs which form a subclass of claw-free graphs [17]. In contrast, the alogorithmic complexity of the COLORING problem for AT-free graphs is still an open problem. Let $t_{MATCH}(n,m)$ and $t_{BIPMATCH}(n,m)$ be the times needed to compute a maximum matching on general graphs and on bipartite graphs, respectively. Similarly, let $t_{COLOR}(n,m)$ denote the time to color a claw-free AT-free graph.

Theorem 27. [3] *There is an* $O(n^2+\sqrt{n}m)$ *coloring algorithm for claw-free AT-free graphs. Furthermore,* $t_{COLOR}(n,m) \leqslant \max\{t_{MATCH}(n,m), t_{BIPMATCH}(2n,m)\} + O(n^2)$ *and* $t_{BIPMATCH}(n,m) \leqslant t_{COLOR}(n,m) + O(n^2)$.

Proof. The following algorithm colors a connected claw-free AT-free graph G with a minimum number of colors: Compute a 2LexBFS scheme of G. Let $L_0, L_1, \ldots, L_r$ be its levels. If $r \leqslant 2$ then by Lemma 10, $\alpha(G) \leqslant 2$ and we thus exploit the well-known fact that a minimum coloring of a graph G with $\alpha(G) \leqslant 2$ can be determined by computing a maximum matching M of $\overline{G}$ and by coloring G such that two vertices u and v of G obtain the same color if and only if $\{u,v\}$ is an edge of the maximum matching M. If $r > 2$ we either have that L_1 is a clique, in which case the 2LexBFS ordering of G is a cocomparability ordering of G since now all levels are cliques, or L_1 is not a clique implying that $\alpha(G) \geqslant 3$ and thus G is cocomparability due to Theorem 3. So we use an algorithm solving the COLORING problem for cocomparability graphs by a reduction to MATCHING on a bipartite auxiliary graph [14] (see also [28, p. 274]). It is not hard to verify that the just outlined algorithm runs in time $\max\{t_{MATCH}(n,m), t_{BIPMATCH}(2n,m)\} + O(n^2)$ which proves the first inequality.

Due to the fact that the current best algorithm for maximum matching on graphs has running time $O(\sqrt{n}m)$ [22] it follows that there exists a $O(\sqrt{n}m + n^2)$ coloring algorithm for claw-free AT-free graphs.

For the other inequality to be proven simply observe that the complement of a bipartite graph G satisfies $\alpha(\overline{G}) \leqslant 2$ and thus is a claw-free AT-free graph. So a minimum coloring of $\overline{G}$ induces a maximum matching on G. □

[3] A polynomial-time coloring algorithm for claw-free AT-free graphs was independently found by E. Köhler.

Though the above theorem might give the impression that full complementation of the input graph, and thus an extra $O(n^2)$ in the running time, is necessary in order to apply matching algorithms we emphasize that this is not true.

Theorem 28. *There is an* $O(\sqrt{n}m)$ *algorithm for computing a minimum coloring for claw-free AT-free graphs.*

Proof. Our algorithm first computes a 2LexBFS scheme of the input graph. Then it proceeds stepwise down the levels of the 2LexBFS scheme coloring each newly entered level optimally. The correctness of the algorithm is shown by induction over the levels of the computed 2LexBFS scheme.

We denote the color of a vertex v by $c(v)$.

Algorithm COLORING

1. Compute a 2LexBFS scheme of the input graph G. Let $L_0 = \{x\}$, $L_1 = N(x), \ldots, L_r$ be its levels.
2. Set $c(x) = 1$.
3. Compute a maximum matching of $\overline{G}[L_1 \cup L_2]$. Let $\hat{k}$ be the size of the computed maximum matching.
4. Use colors $\{2, 3, \ldots, |L_1| - \hat{k} + 1\}$ to color the nodes in $L_1 \cup L_2$ in the following way: For every edge from the found maximum matching choose a different color and assign it to the two endpoints of that edge. If there are uncolored vertices of L_2 give one of them color 1. Color the remaining uncolored vertices of $L_1 \cup L_2$ pairwise different with colors $\{|L_1| - \hat{k} + 2, \ldots, |L_1| + |L_2| - \hat{k} - 1\}$.
5. For $i = 2$ to $r - 1$ do
   (a) Let H_i be the graph consisting of all vertices from $L_i \cup L_{i+1}$ and all edges $\{u, v\}$ such that $u \in L_i$, $v \in L_{i+1}$, and $\{u, v\} \notin E$. Compute a maximum matching of H_i. Let k_i be the size of the computed maximum matching. Let l_i be the number of colors used in L_i and let c_i denote the number of colors used in $L_0 \cup L_1 \cup \cdots \cup L_i$.
   (b) Use $|L_{i+1}|$ colors to color the vertices in L_{i+1} in the following way: For every edge $\{u, v\}$, where $u \in L_i$ and $v \in L_{i+1}$, from the found maximum matching assign color $c(u)$ to v. The remaining vertices of L_{i+1} are all colored pairwise different, using at most $c_i - l_i$ colors that have already been used to color vertices in levels $L_0, L_1, \ldots, L_{i-1}$ and that have not been used to color vertices in L_i and if necessary $|L_{i+1}| - k_i - (c_i - l_i)$ new colors.

It is not hard to verify that our algorithm produces a proper coloring. To see this note that on one hand our algorithm ensures that the nodes in each level of the 2LexBFS are colored properly. Since L_i, $i \geqslant 2$, forms a clique we have to use $|L_i|$ different colors for coloring L_i and our algorithm does exactly that. The coloring of two adjacent levels is based on a matching algorithm which again ensures that only non-adjacent vertices get the same color.

It remains to show that the above algorithm produces an optimal coloring. First recall that by Lemma 10, our algorithm constructs a coloring for which each graph $G[L_i \cup L_{i+1}]$ is colored with the minimum number of colors. Now suppose there exists an optimal coloring c_{opt} using less colors than the coloring c found by the algorithm. Hence there exists a smallest i, call it i_0, such that (a) within levels $L_0, L_1, \ldots, L_{i_0}$ c and c_{opt} have used the same number of colors and (b) within levels $L_0, L_1, \ldots, L_{i_0}, L_{i_0+1}$ c uses at least one color more than c_{opt}. We will now show that this cannot happen. Clearly, c is an optimal coloring on $G[L_0 \cup L_1 \cup L_2]$, since $G[L_1 \cup L_2]$ is colored optimally and color 1 is reused on L_2 (if appropriate, i.e., if not all vertices of L_2 are covered by an edge of the maximum matching of $\overline{G}[L_1 \cup L_2]$). Hence $i_0 \geqslant 2$. Since, L_{i_0+1} is a clique, all vertices in L_{i_0+1} are assigned different colors by c_{opt} and c. Suppose that c_{opt} uses more colors of L_{i_0} to also color vertices in L_{i_0+1} than c does. Since c_{opt} is a coloring, there has to exist a matching of non-edges of G between L_{i_0} and L_{i_0+1} that is larger then the maximum matching computed in the algorithm, a contradiction. Hence it remains the case that c_{opt} uses more colors from previous levels that have not been used in L_{i_0} than c does. But also this is impossible, since the algorithm itself uses as many colors from previous levels L_i, $i < i_0$, as possible. This shows that the algorithm indeed computes an optimal coloring.

Regarding the time bound of the algorithm we mention that computing a maximum matching can be done in time $O(\sqrt{n}m)$ [22]. In our algorithm a number of maximum matchings are computed, but every vertex is involved in the computation of a maximum matching at most twice. Any edge appearing in an input to a matching algorithm is an edge of E^* or a non edge of $G[L_1]$ and appears exactly once in an input to a matching algorithm. By Lemma 19, $|E^*| = O(|E|)$ and $G[L_1]$ has $\Theta(|L_1|^2)$ edges. Hence the overall time bound of the algorithm is $O(\sqrt{n}m)$. □

Acknowledgements

The authors are deeply grateful to the anonymous referees for their invaluable suggestions regarding the presentation of the material.

References

[1] N.B. Allan, R. Laskar, On domination and independent domination numbers of a graph, Discrete Math. 23 (1978) 73–76.

[2] N. Alon, R. Yuster, U. Zwick, Finding and counting given length cycles, Algorithmica 17 (1997) 209–223.

[4] A. Brandstädt, V. Bang Le, J. Spinrad, Graph Classes: a Survey, Society for Industrial and Applied Mathematics, Philadelphia, 1999.

[5] A. Brandstädt, F.F. Dragan, E. Köhler, Linear time algorithms for hamiltonian problems on (claw,net)-free graphs, in: Proceedings of the 25th Workshop on Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science, Vol. 1665, eds. P. Widmayer, G. Neyer, S. Eidenbenz, Springer, Berlin 1999, pp. 364–376.

[6] H. Broersma, T. Kloks, D. Kratsch, H. Müller, Independent sets in asteroidal triple-free graphs, SIAM J. Discrete Math. 12 (1999) 267–287.

[7] V. Chepoi, F. Dragan, A linear-time algorithm for finding a central vertex of a chordal graph, in: Proceedings of the 4th Annual European Symposium on Algorithms, Lecture Notes in Computer Science, Vol. 855, ed. J. van Leeuwen, Springer, Berlin, 1994, pp. 159–170.
[8] V. Chepoi, F. Dragan, Finding a central vertex in HHD-free graphs. preprint, University of Rostock, 1998.
[9] D. Coppersmith, S. Winograd, Matrix multiplication via arithmetic progressions, J. Symbolic Comput. 9 (1990) 251–280.
[10] D.G. Corneil, F.F. Dragan, M. Habib, C. Paul, Diameter determination on restricted graph families. in: Proceedings of the 24th Workshop on Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science, Vol. 1517, eds. J. Hromkovic, O. Sykora, Springer, Berlin, 1985, pp. 267–350.
[11] D.G. Corneil, S. Olariu, L. Stewart, Asteroidal triple-free graphs, SIAM J. Discrete Math. 10 (1997) 399–430.
[12] D.G. Corneil, S. Olariu, L. Stewart, A linear time algorithm for dominating pairs in asteroidal triple-free graphs, SIAM J. Comput. 28 (1999) 1284–1297.
[13] M.J. Fischer, A.R. Meyer, Boolean matrix multiplication and transitive closure, in: Proceedings of the 12th Annual Symposium on Switching and Automata Theory, 1971, pp. 129–131.
[14] D.R. Fulkerson, A Note on Dilworth's decomposition theorem for partially ordered sets, Proc. Amer. Math. Soc. 7 (1956) 701–702.
[15] M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs, Academic Press, New York, 1980.
[16] H. Hempel, D. Kratsch, On claw-free asteroidal triple-free graphs. in: Proceedings of the 25th Workshop on Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science, Vol. 1665, eds. P. Widmayer, G. Neyer, S. Eidenbenz, Springer, Berlin, 1999, pp. 377–390.
[17] I. Holyer, The NP-completeness of edge-coloring, SIAM J. Comput. 10 (1981) 718–720.
[18] T. Kloks, D. Kratsch, H. Müller, Approximating the bandwidth for AT-free graphs. in: Proceedings of the 5th Annual European Symposium on Algorithms, Lecture Notes in Computer Science, Vol. 979, ed. P. Spirakis, Springer, Berlin, 1995, pp. 434–447.
[19] T. Kloks, D. Kratsch, H. Müller, Asteroidal sets in graphs. in: Proceedings of the 23rd Workshop on Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science, Vol. 1335, ed. R.H. Möring, Springer, Berlin, 1997, pp. 229–241.
[20] E. Köhler, Graphs without asteroidal triples. Ph.D. thesis, Technische Universität Berlin, 1999.
[21] D. Kratsch, Domination and total domination on asteroidal triple-free graphs, Discrete Appl. Math. 99 (2000) 111–123.
[22] S. Micali, V.V. Vazirani, An $O(V^{1/2}E)$ algorithm for finding maximum matching in general graphs. in: Proceedings of the 20th IEEE Symposium on Foundations of Computer Science, New York, 1980, pp. 17–25.
[23] G.J. Minty, On maximal independent sets of vertices in claw-free graphs, J. Combin. Theory, Ser. B 28 (1980) 284–304.
[24] A. Parra, P. Scheffler, Characterizations and algorithmic applications of chordal graph embeddings, Discrete Appl. Math. 79 (1997) 171–188.
[25] D.J. Rose, R.E. Tarjan, G.S. Lueker, Algorithmic aspects of vertex elimination on graphs, SIAM J. Comput. 5 (1976) 266–283.
[26] N. Sbihi, Algorithme de recherche d'un stable de cardinalite maximum dans un graphe sans etoile, Discrete Math. 29 (1980) 53–76.
[27] J. Spinrad, private communication.
[28] D. West, Parameters of partial orders and graphs: packing, covering, and representation. In Graphs and order. The role of graphs in the theory of ordered sets and its applications, Proc. NATO Adv. Study Inst., NATO ASI Ser., Ser. C 147, ed. I. Rival, 1985, pp. 267–350.
[29] D. West, Introduction to Graph Theory, Prentice-Hall, Upper Saddle River, NJ, 1996.
[30] M. Yannakakis, F. Gavril, Edge dominating sets in graphs, SIAM J. Appl. Math. 38 (1980) 364–372.

For further reading

H. Alt, N. Blum, K. Mehlhorn, M. Paul, Computing a maximum matching in a bipartite graph in time $O(n^{1.5}\sqrt{m/\log n})$, Inform. Process. Lett. 37 (1991) 237–240.