Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 273 (2025) 397–404

Procedia Computer Science

www.elsevier.com/locate/procedia

XIII Latin American Algorithms, Graphs, and Optimization Symposium (LAGOS 2025)

Finding subdigraphs in digraphs of bounded directed treewidth[1]

Raul Lopes[a,b], Ignasi Sau[a,*]

[a] *LIRMM, Université de Montpellier, CNRS, Montpellier, France*

[b] *Hamburg University of Technology, Institute for Algorithms and Complexity, Hamburg, Germany*

Abstract

It is well known that directed treewidth does not enjoy the nice algorithmic properties of its undirected counterpart. There exist, however, some positive results that, essentially, present XP algorithms for the problem of finding, in a given digraph D, a subdigraph isomorphic to a digraph H that can be formed by the union of k directed paths (with some extra properties), parameterized by k and the directed treewidth of D. Our motivation is to tackle the following question: Are there subdigraphs, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth? In a nutshell, the main message of this article is that, other than the directed paths, the only digraphs that seem to behave well with respect to directed treewidth are the stars. For this, we present a number of positive and negative results, generalizing several results in the literature, as well as some directions for further research.

Keywords: Directed graphs; Directed treewidth; Parameterized complexity; Subdigraph isomorphism; Dynamic programming; Hardness result

1. Introduction

Treewidth of undirected graphs [26] is probably the most successful graph parameter from an algorithmic viewpoint, as capitalized by the celebrated Courcelle's theorem [6], stating that all problems that are expressible in (the very powerful) monadic second-order logic can be solved in linear time when restricted to graphs of bounded treewidth; using terminology from parameterized complexity, they are *fixed-parameter tractable* (FPT). Unfortunately, the directed counterpart of treewidth, called *directed treewidth* [25, 12], does not enjoy the nice algorithmic properties of undirected treewidth. Indeed, on the one hand, directed treewidth seems harder to compute (or to approximate) than its undirected counterpart [3, 15, 5, 12]. On the other hand, and closer to the topic of this article, some "basic" problems that fit the setting of Courcelle's theorem [6] cannot be solved efficiently in digraphs when parameterized by directed treewidth, as observed by Lampis et al. [16]. This is the case, for instance, of Directed Hamiltonian Path that is W[2]-

[1] Ignasi Sau was funded by French project ELIT (ANR-20-CE48-0008-01). Raul Lopes was funded by French project ELIT (ANR-20-CE48-0008-01) and HIDSS-0002 DASHH (Data Science in Hamburg - Helmholtz Graduate School for the Structure of Matter).

* Corresponding author.
E-mail address: raul.lopes@tuhh.de, ignasi.sau@lirmm.fr

1877-0509
10.1016/j.procs.2025.10.324

hard parameterized by directed treewidth (hence, unlikely to be FPT), or Max Directed Cut that is even NP-hard restricted to directed acyclic graphs (DAGs), which have directed treewidth zero [16]. Ganian et al. [9] provide concrete reasons for which digraph width measures (including directed treewidth) cannot have the same nice algorithmic properties as undirected treewidth. Since our focus is algorithmic, in this discussion we deliberately omit the recent impressive progress on *structural* properties of directed treewidth [11, 10, 14].

However, there are a few positive algorithmic results concerning directed treewidth, two of which are particularly relevant to us. Namely, in the article where directed treewidth was introduced, Johnson et al. [12] proved that Directed Hamiltonian Path and Directed Disjoint Paths can be solved in XP time (that is, in polynomial time for each fixed value of the parameter; see [8, 7] for basic background on parameterized complexity) parameterized by directed treewidth (plus the number of paths in the latter problem). Later, de Oliveira Oliveira [23] provided an algorithmic meta-theorem for directed treewidth generalizing the previous results, by showing, informally and in a simplified statement, that the problem of deciding whether a digraph D contains a subdigraph H consisting of the union of k directed paths and satisfying a given monadic second-order formula φ can be solved in XP time parameterized by k, the size of φ, and the directed treewidth of D.

It is worth noting that both algorithms in [12, 23] have an XP dependence on the directed treewidth, and, more crucially, that these XP algorithms are for problems consisting in finding *directed paths* satisfying certain properties, namely being pairwise disjoint and rooted at prescribed terminals in [12], or satisfying a monadic second-order formula in [23].

The question that arises naturally from the previous discussion, and which is our main motivation, is the following: are there subdigraphs, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

Our contribution. In a nutshell, the main message of this article is that, other than the (directed) paths, the only digraphs that seem to behave well with respect to directed treewidth are the *stars*. In what follows, we make the former vague statement more precise.

Driven by the above question, we are interested in the following problem, called Subdigraph Isomorphism: given a host digraph D of directed treewidth at most w and a digraph H, decide whether D contains a subdigraph isomorphic to H.[2] To the best of our knowledge Subdigraph Isomorphism in its full generality, and considering as a parameter the directed treewidth of the input digraph, has been unexplored in the literature, in contrast with its undirected counterpart (see for instance [21]).

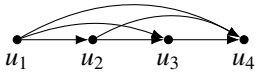
Unsurprisingly, Subdigraph Isomorphism is NP-complete in general (for instance, if H is a transitive tournament, there is a trivial reduction from Clique in undirected graphs), so our focus is on parameterized algorithms when taking as the parameters (combinations of) w and some parameter κ depending on H. Observe that if we take $\kappa(H) = |V(H)|$, then the problem can be trivially solved in time $|V(D)|^{O(\kappa(H))}$ by just brute-forcing over all vertex sets of D of size $|V(H)|$ and checking whether the subdigraph of D induced by them contains a digraph isomorphic to H. Thus, in order to face non-trivial questions, we consider parameters $\kappa(H)$ smaller than the size of H. In the spirit of the meta-algorithm of de Oliveira Oliveira [23] (albeit, without the logical ingredient), when H can be defined as the union of k digraphs belonging to some allowed collection $\mathcal{A}$ of digraphs, the natural choice that we consider is $\kappa(H) = k$. Note that in [23], the collection $\mathcal{A}$ consists of all directed paths.

Our main positive result is an XP algorithm when $\mathcal{A}$ contains directed paths and stars (of arbitrary size). More precisely, we provide an XP algorithm to solve Subdigraph Isomorphism parameterized by w (the directed treewidth of the input graph) and the number of directed paths and oriented stars whose union yields the desired subdigraph H (cf. Theorem 1 for the precise statement).

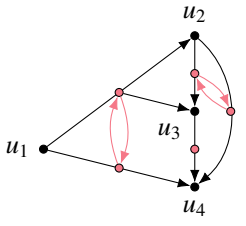
On the negative side, we provide a number of hardness results showing that, as far as H deviates slightly from being definable by the union of few paths or stars, then Subdigraph Isomorphism is NP-complete even restricted to digraphs of bounded directed treewidth (at most two, and zero in almost all cases); see Theorem 2 for the precise statement and Figure 1 for an illustration of some of the graphs H for which we prove NP-completeness. Note that the digraphs depicted from Figure 1**(c)** to Figure 1**(f)** are "close" to being definable by the union of few paths or

[2] A clarification is in place here. Note that the way we have defined Subdigraph Isomorphism, it does *not* encompass, for instance, the Directed Disjoint Paths problem, where the input contains pairs of vertices, called the *terminals*, to be joined by the corresponding paths. Nevertheless, as it will become clear, the XP algorithm that we present solves the *rooted* version (that is, with terminals) as well.

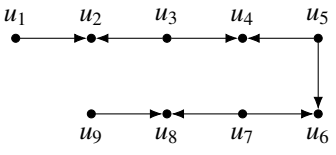
stars, in the sense that they can be defined by the union of few digraphs that are "close" to directed paths or stars, namely an antidirected path in Figure 1**(c)**, a digraph obtained from a (large) star by attaching small stars to each leaf in Figure 1**(d)** (the hardness result also holds if one removes the root s), small subdivisions of exactly two stars in Figure 1**(e)**, or a "caterpillar-like" digraph in Figure 1**(f)**. The cases depicted in Figure 1**(a)** and Figure 1**(b)** will be discussed in the next section.



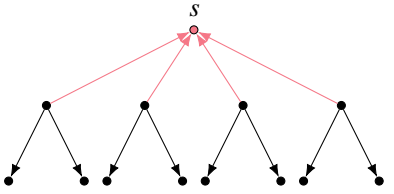
(a) A transitive tournament T.



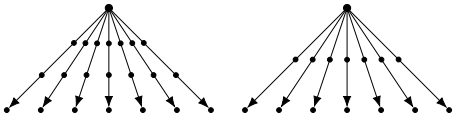
(b) The expansion of T.



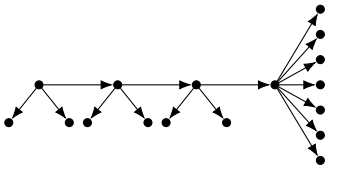
(c) An antidirected path of length eight.



(d) Disjoint union of 2-out-stars, plus a 4-in-star with center s having all other centers as leaves.



(e) The disjoint union of H_1 and H_2, where H_i is the i-subdivision of a 7-out-star.



(f) A caterpillar ending on a 7-out-star.

Fig. 1: Examples of digraphs from the classes mentioned in Theorem 2. Items **3.** and **4.** of Theorem 2 are represented by **(d)**: the former when s is deleted from the digraph, and the latter with the digraph as it is.

Organization. In Section 2 we provide an overview of the techniques that we use to prove our positive and negative results. Due to space limitations, all proofs and full formal definitions are omitted in this extended abstract, and can be found in the full version of this paper[3]. We conclude the article in Section 3 with some directions for further research.

2. Overview of our techniques

In this section we describe in more detail our results and present the main ideas involved in the proofs. As mentioned above, full details can be found in the corresponding appendices.

The XP algorithm. On the positive side, as mentioned in Section 1, we give an XP algorithm for the case when H is formed by the disjoint union of stars and internally disjoint paths between the centers of those stars (note that unions of paths are also captured in this way, by considering stars with just one vertex as the endpoints), which we name Rooted Stars-Paths Subdigraph Isomorphism. Our XP algorithm is parameterized by the number of stars and the directed treewidth of the given digraph. The algorithm consists of two parts. First, we notice that if the goal is to find only the stars, one can reduce the problem to an integer system of inequations, where the inequations are given by the size of the neighborhoods of the vertices candidates for the centers of the stars, and the size of the stars that

[3] The full version of this paper is available at https://arxiv.org/abs/2508.13830.

we want to build. This part runs in FPT time parameterized by the number of stars after the candidates for the centers are given, using the fact that an integer system of linear inequations can be solved in FPT time parameterized by the number of variables [13]. Additionally, we can add the following query to the algorithm: if, for every candidate vertex $v \in D$, we are given a number $\mathsf{slack}(v)$ representing how many vertices in the usable neighborhood of v we want to *avoid* using for the stars, can we find the stars while respecting the slacks? This is needed for the next part of the algorithm, where the goal is to route the paths while avoiding sufficiently many vertices of the usable neighborhoods such that the stars can be built.

More precisely, in the second part, we introduce a new version of the Directed Disjoint Paths problem where, in addition to the terminal set $\{s_1, t_1, \ldots, s_r, t_r\}$ that we want to connect with pairwise vertex-disjoint paths, we are given subsets $X_1, \ldots, X_k \subseteq V(D)$ and integers $x_1, \ldots, x_k$, and the goal is to route the paths while using at most x_i vertices of X_i. We call this the Subset Avoiding Directed Disjoint Paths (SADDP) problem. Applying the framework of Johnson et al. [12] (see also Lopes and Sau [17] for another application of this framework), we show that SADDP is XP with parameters r and k. As in the case of the XP algorithm for DDP in [12], we exploit the bounded breakability of paths to obtain a dynamic programming algorithm for our problem. We remark that directed treewidth appears only in this part of the algorithm.

Given a set of candidates for the centers of the stars, for each choice of the slacks for those vertices, we build an instance of SADDP where the sets X_i are built from the neighborhoods of the candidates, and solve it in XP time with parameters r, k, and the directed treewidth of the given digraph. Since the number of possible choices of for the slacks is $O(n^k)$, this strategy yields an XP algorithm. Since the end goal is an XP algorithm, we can pay the cost of guessing where to embed the centers of the stars in the digraph and thus use the rooted version to solve the unrooted one. Note that, with this approach, an XP algorithm is the best we can hope for, since Directed Disjoint Paths, which we solve as a particular case of our algorithm, is known to be W[1]-hard (hence, unlikely to be FPT) even restricted to DAGs [27].

Our XP algorithm is formally stated in the following theorem.

Theorem 1. *The* Rooted Stars-Paths Subdigraph Isomorphism *problem is solvable in time* $n^{O(k+r+w)}$*, where* w *is the directed treewidth of the input digraph* D*,* k *is the number of stars forming* H*, and* r *is the number of paths in the collection* $\mathcal{P}$*.*

Hardness results. Our hardness results are itemized in Theorem 2, stated at the end of this section, and follow from several distinct reductions, which we proceed to explain here, as well as defining the digraphs that we consider. Namely, we consider the following cases for the choice of the target digraph H in Subdigraph Isomorphism. We say that H is an *antidirected path* if H is an orientation of an undirected path where the arcs alternate between forward and backwards arc along the path. In this case, we improve on a result from Bang-Jensen et al. [1, Theorem 2.2] which proves that the rooted version of the problem is NP-complete. Their reduction does not generate a DAG, however, and we show how to modify it in order to do so. In short, by increasing the size of the antidirected paths added to the constructed digraph D in the reduction, we ensure that no two vertices which are not sinks nor sources in D are within a directed path of D.

An *out-star* (*in-star*) is an orientation of an undirected star where all arcs are oriented away from (towards) the center of the star. If H is formed by the disjoint union of out-stars (in-stars) with two leaves, we provide a novel reduction from a matching problem in bipartite graphs, introduced and shown to be NP-complete by Plaisted and Zaks [24]. In this matching problem, we are given a bipartite graph G with parts V_1 and V_2 together with partitions $\mathcal{P}_1$ and $\mathcal{P}_2$ of V_1 and V_2, respectively. The goal is to decide if G has a perfect matching M such that no two distinct edges $\{a_1, b_1\}, \{a_2, b_2\} \in M$ have the property that a_1, a_2 are in the same part of $\mathcal{P}_1$ or b_1, b_2 are in the same part of $\mathcal{P}_2$. A perfect matching with such properties is said to be *consistent*. If the goal is to prove the result for out-stars, we can begin as follows. For each $e \in E(G)$ with endpoints u, v, one can add the arc $\{u, v\}$ to the digraph D plus a new vertex v_e with out-neighbors u and v, then search for $|V_1| = |V_2|$ out-stars in D. This does not guarantee that the leaves of the out-stars are selecting edges of G forming a consistent matching. Thus, we exploit the extra properties of the results of [24].

The crucial property is that in [24] the authors show that the problem remains hard even if every part of $\mathcal{P}_1$ and every part of $\mathcal{P}_2$ has size two. This allows our reduction to distinguish four possible configurations for each pair A, B with $A \in \mathcal{P}_1$ and $B \in \mathcal{P}_2$, depending on how many edges there are between A and B. For each configuration with two

or more edges between the sets, we add some new stars to the digraph and increase the number of stars we are going to search for in the instance of Subdigraph Isomorphism. Then we show that the stars that we added in the configurations are sufficient to select a consistent matching in G if the appropriate number of stars is found.

For $i \in [2]$, let S_i be the digraph formed by subdividing i times every arc of an out-star. For the case when H is formed by the disjoint union of a copy of S_1 and a copy of S_2, we provide a reduction from a particular case of 3-SAT in which every literal appears *exactly* twice in the clauses. This version of 3-SAT was shown to be NP-complete by Berman et al. [2]. The specification on the number of times each literal can appear is key for the reduction to work. We start with a vertex s and add to the digraph $2n$ (not necessarily disjoint) paths of length three leaving s, where n is the number of variables. Each pair of paths represents a choice of a truth value to a variable. The two last vertices of each path are associated with the two positive and two negated occurrences of a variable in the clauses, and each path leaving s uses the two vertices associated with the two positive or the two negative occurrences of a variable in the clauses. Thus the second clause, built from a structure associated with the clauses, can only have leaves in the two vertices *not* selected by the path leaving s, for each variable. The star with center s acts as a *selector* for the variables, and a star with a center c associated with the clauses acts as a *validator* for the clauses.

The role of breakability. In all the aforementioned cases in our hardness reductions, the constructed digraphs are DAGs. The key "separation" notion of directed treewidth is that of *w-guarded* sets. In short, given an integer w, a set of vertices X of a digraph D is w-guarded if there is a set $Z \subseteq V(D) \setminus S$ of size at most w such that no path of D starts in X, intersects $V(D) \setminus (X \cup Z)$, and returns to X without using a vertex of Z.

In many cases, parameterized algorithms exploiting the structure of digraphs with bounded directed treewidth begin by showing that, when the goal is to find some subdigraph $H \subseteq D$, that H cannot be split into too many pieces by a w-guarded set. In other words, if X is w-guarded, the maximum number of weak components in the subdigraph induced by $V(H) \cap X$ is bounded by some function of w. This approach is used, for example, in [12, 17]. We formalize this notion by saying that H has *bounded breakability*. A priori, one might expect that bounded breakability is a strong enough condition to ensure the existence of XP algorithms for NP-hard problems restricted to digraphs of bounded directed treewidth. Somehow surprisingly, we show that this is not the case, even if the target digraph also has bounded maximum degree together with bounded breakability.

Defining a class $\mathcal{H}$ of digraphs with bounded breakability, bounded maximum degree, and such that Subdigraph Isomorphism is NP-complete with respect to target digraphs in $\mathcal{H}$ turned out to be no easy task. We do so by augmenting a classical construction that is used to bound the maximum degree of digraphs. Given a digraph D, we define the *expansion* of D as the digraph D' built following a series of steps (cf. Figure 1**(a)** and Figure 1**(b)**). Namely, and informally, we apply the classical procedure of replacing all the arcs leaving a vertex $u \in V(D)$ by an out-arborescence with root u (that is, the orientation of a tree with root r where all arcs are oriented away from u). Then, we do the same for all vertices of $V(D)$ which have in-degree greater than two, but now for incoming arcs. This bounds the maximum degree of D', but is not enough to bound its breakability.

For the breakability, we add arcs to D' in order to ensure that the local out- and in-arborescences added to D' when a vertex u is processed are augmented to "almost strong" digraphs. We need to be careful here: although an increase in the connectivity is needed to ensure that for every $u, v \in V(D')$ there is a path from u to v or from v to u (or both) when D' is the expansion of a transitive tournament, we need to avoid creating paths between the "original" vertices x, y of D' (that is, $x, y \in V(D)$) when there are no paths paths between x and y in D. Informally, the connectivity needs to increase, but not by too much that we lose control on the directed treewidth of expansions of DAGs. This is important to maintain since the goal is to prove hardness for expansions of tournaments when the base digraph has directed treewidth at most three.

Let T' be the expansion of a transitive tournament T. Proving that Subdigraph Isomorphism is hard when H is the expansion of a transitive tournament is easy. We simply follow the same idea as in the proof of hardness for transitive tournaments. Proving that the breakability of T' is bounded, however, is much harder. We first show that, given a vertex $v \in V(T')$, one can define a partition $V(T') \setminus \{v\}$ into four sets $\{X_i^v \mid i \in [4]\}$ with the following property: for any $x, y \in X_i^v$, there is a path from x to y or from y to x *avoiding* v. From here, the proof follows by induction. In short, given a vertex v in a guard Z of size w, we split $V(T')$ as mentioned into sets $\{X_i^v \mid i \in [4]\}$. Since v is part of the guard, each $V(T') \cap X_i^v$ is $w-1$-guarded. By induction, $V(T') \cap X_i^v \cap X$ has at most 4^{w-1} weak components when X is a w-guarded set, which implies that the breakability of T' is at most 4^w.

Our hardness results are summarized in the following theorem. For an integer $k \geq 1$, a *k-homogeneous star* is a a digraph isomorphic to either a k-out-star or to a k-in-star.

Theorem 2. *Let $k \geq 1$ be an integer.* Subdigraph Isomorphism *is* NP*-complete if*

1. *the host digraph D is a DAG and H is an antidirected path;*
2. *the host digraph D has directed treewidth three and H is the expansion of a transitive tournament (and thus of maximum degree at most seven);*
3. *the host digraph D is a DAG and H is formed by the disjoint union of 2-out-stars or the disjoint union of 2-in-stars;*
4. *the host digraph D is a DAG and H is formed by the disjoint union of k copies of 2-in-stars (2-out-stars) plus a k-homogeneous star whose leaves are all the k centers of the other stars;*
5. *the host digraph D is a DAG and H is formed by the disjoint union of digraphs H_1 and H_2 where each H_i is built from a k-homogeneous star where each arc is i-subdivided;*
6. *the host digraph D has directed treewidth one and H is caterpillar ending in a k-homogeneous star and every other branching vertex has exactly two leaves as in-neighbors or out-neighbors.*

3. Conclusions and further research

Our article initiates a systematic study of the following fascinating question. Let $\mathcal{A}$ be a collection of *allowed* digraphs, let H be a digraph that can be built as the union of at most k digraphs in $\mathcal{A}$, and let D be a digraph with directed treewidth at most w. For which collections $\mathcal{A}$ the problem of deciding whether D contains a subdigraph isomorphic to H can be solved in XP time parameterized by k and w? Let us call a collection $\mathcal{A}$ *easy* (resp. *hard*) if the answer to the previous question is positive (resp. negative). Note that a negative answer will be typically conditioned to some complexity hypothesis, such as $\mathsf{P} \neq \mathsf{NP}$ or another ones.

Our XP algorithm (cf. Theorem 1) shows that if $\mathcal{A}$ contains the directed paths and the stars (oriented away from or towards the center), then it is easy. On the other hand, our hardness results (cf. Theorem 2) provide a number of examples of hard collections $\mathcal{A}$ containing digraphs that are "close" to paths and stars (cf. Figure 1), namely by showing that the problem is NP-complete for fixed small values of k and w.

The main message of our article is that a good collection $\mathcal{A}$ is unlikely to contain many digraphs that differ substantially from (combinations of) paths and stars. Nevertheless, we fall short of providing a full characterization of easy and hard collections. This seems to be a very challenging question, in particular because of the reasons that we proceed to discuss.

For a positive integer p, we say that a collection of digraphs $\mathcal{A}$ is *p-easy* (resp. *p-hard*) if the answer to the above question is positive (resp. negative) for every $k \leq p$ (resp. for some $k \leq p$). Hence, a collection $\mathcal{A}$ is easy if and only if it is p-easy for every $p \geq 1$. Let H_ℓ be the digraph made of ℓ pairwise disjoint arcs. Then deciding whether a host digraph D contains H_ℓ as a subdigraph can be done in polynomial time for any ℓ, by just running a classical maximum matching algorithm [4] in the underlying undirected graph of D. Thus, the collection $\mathcal{A} = \{H_\ell \mid \ell \in \mathbb{N}\}$ is 1-easy, but its elements *cannot* be formed as the union of few paths or stars, and therefore it escapes the setting of Theorem 1. Note, however, that this collection $\mathcal{A}$ is 2-hard, because an antidirected path (cf. Figure 1**(c)**) can be obtained as the union of two elements in $\mathcal{A}$, and we have proved that the corresponding problem is NP-complete in DAGs (item **1.** of Theorem 2).

Exploiting the potential of a maximum matching algorithm, we can construct other 1-easy collections $\mathcal{A}'$ that escape the setting of Theorem 1. Indeed, let H'_ℓ be the digraph obtained from an out-star with ℓ leaves by subdividing every arc once (and keeping the same orientation in both new arcs), and let $\mathcal{A}' = \{H'_\ell \mid \ell \in \mathbb{N}\}$. Note that the elements in $\mathcal{A}'$ cannot be obtained as the union of few paths or stars, and therefore it also escapes the setting of Theorem 1. We claim that $\mathcal{A}'$ is 1-easy. Indeed, we proceed to present a polynomial-time algorithm to decide whether a digraph D contains a subdigraph isomorphic to H'_ℓ, for any $\ell \geq 0$ (without even needing to use a bound on the directed treewidth of D). For a vertex $v \in V(D)$, let N_v be the out-neighborhood of v, let A_v be the set of arcs of D having a vertex in N_v

as the tail (note that the head may also be in N_v), and let D_v be the subdigraph of D defined by the union of the arcs in A_v. It is easy to see that D contains a subdigraph isomorphic to H'_ℓ if and only if there is a vertex $v \in V(D)$ such that the underlying graph of D_v contains a matching of size at least ℓ, which can be clearly decided in polynomial time. A symmetric argument applies to the collection containing subdivisions of in-stars.

How far can we go with this matching-based approach? A next natural step could be, for instance, try to generalize the above algorithm to find a digraph H defined by the disjoint union of *two* subdivided stars H'_{ℓ_1} and H'_{ℓ_2}, as defined in the previous paragraph, for two positive integers ℓ_1 and ℓ_2. Note that this digraph H is really close to the one depicted in Figure 1**(e)** (hence, a hard collection), also consisting of two subdivided out-stars, but one subdivided once and the other one subdivided *twice*, while in H both stars are subdivided *once*. It turns out that finding H has a strong connection with a notoriously open problem called Exact Matching. In this problem, we are given an edge colored undirected graph G, with colors red and blue, and an integer ℓ, the goal is to decide whether or not the graph contains a perfect matching with exactly ℓ red edges. Exact Matching is known to be solvable in randomized polynomial time [22] (hence, in the class RP and unlikely to be NP-hard), but it is still unknown whether it is in P (that is, solvable in *deterministic* polynomial time). In particular, Exact Matching is known to be in P only for very restricted graph classes, such as cliques, complete bipartite graphs, graphs of bounded independence number, or planar graphs; see [19, 20, 18] for the recent work of El Maalouly on this problem.

Back to our problem, consider a digraph D with only two vertices v_1 and v_2 of "large" out-degree, and suppose that $|V(D)|$ is even. Let ℓ_1 and ℓ_2 be positive integers such that $\ell_1 + \ell_2 = (|V(D)| - 2)/2$, and suppose that both ℓ_1 and ℓ_2 are "large", in the sense that there is no vertex in $V(D) \setminus \{v_1, v_2\}$ with out-degree at least $\min\{\ell_1, \ell_2\}$. We construct from D an undirected graph G, with edges colored red and blue, as follows. We define G to be the underlying graph of $D \setminus \{v_1, v_2\}$, and we color an edge of G red (resp. blue) if at least one of its endpoints is an out-neighbor of v_1 (resp. v_2) in D; we can add parallel edges if we want to avoid the same edge to be colored red and blue. It can be easily verified that D contains the disjoint union of H'_{ℓ_1} and H'_{ℓ_2} as a subdigraph if and only if G admits a perfect matching using exactly ℓ_1 red edges. We leave this elusive case as an open problem.

In order to unveil other easy collections, it would be desirable to integrate the matching "technology" into the techniques that we used to obtain our XP algorithm, namely the approach based on solving an integer system of linear inequations to deal with the stars, and the approach based on dynamic programming on an arboreal decomposition to deal with the paths. Unfortunately, this integration seems quite challenging, since matching algorithms have a *global* nature, but both approaches in our XP algorithm have a *local* nature (in the case of stars it is clear, and in the case of paths, the local nature is given by the few crossings that the paths can have in the dynamic programming algorithm). Thus, we think that for finding other easy collections $\mathcal{A}$, substantially new algorithmic ideas are needed.

Finally, it would be very interesting to explore the existence of a generalization of the logical meta-theorem of de Oliveira Oliveira [23] to subgraphs H that can be formed by the union of k directed paths *or stars*.

References

[1] Bang-Jensen, J., Bessy, S., Jackson, B., Kriesell, M., 2017. Antistrong digraphs. Journal of Combinatorial Theory, Series B 122, 68–90. doi:10.1016/j.jctb.2016.05.004.

[2] Berman, P., Karpinski, M., Scott, A.D., 2003. Approximation hardness and satisfiability of bounded occurrence instances of SAT. Electronic Colloquium on Computational Complexity (ECCC) TR03-022. URL: https://eccc.weizmann.ac.il/eccc-reports/2003/TR03-022/index.html.

[3] Bodlaender, H.L., 1996. A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM Journal on Computing 25, 1305–1317. doi:10.1137/S0097539793251219.

[4] Bondy, A., Murty, M.R., 2008. Graph Theory. Springer-Verlag London.

[5] Campos, V., Lopes, R., Maia, A.K., Sau, I., 2022. Adapting the directed grid theorem into an fpt algorithm. SIAM Journal on Discrete Mathematics 36, 1887–1917. doi:10.1137/21M1452664.

[6] Courcelle, B., 1990. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. Information and Computation 85, 12–75. doi:10.1016/0890-5401(90)90043-H.

[7] Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S., 2015. Parameterized Algorithms. Springer.

[8] Downey, R.G., Fellows, M.R., 2013. Fundamentals of Parameterized Complexity. Texts in Computer Science, Springer.

[9] Ganian, R., Hlinený, P., Kneis, J., Meister, D., Obdrzálek, J., Rossmanith, P., Sikdar, S., 2016. Are there any good digraph width measures? Journal of Combinatorial Theory, Series B 116, 250–286. doi:10.1016/J.JCTB.2015.09.001.

[10] Giannopoulou, A.C., Kawarabayashi, K., Kreutzer, S., Kwon, O., 2020. The Directed Flat Wall Theorem, in: Proc. of the 13st ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 239–258. doi:10.1137/1.9781611975994.15.

[11] Hatzel, M., Kreutzer, S., Milani, M.G., Muzi, I., 2024. Cycles of well-linked sets and an elementary bound for the directed grid theorem, in: Proc. of the 65th IEEE Annual Symposium on Foundations of Computer Science (FOCS), pp. 1–20. doi:10.1109/FOCS61266.2024.00011.

[12] Johnson, T., Robertson, N., Seymour, P.D., Thomas, R., 2001. Directed tree-width. Journal of Combinatorial Theory, Series B 82, 138–154. doi:10.1006/jctb.2000.2031.

[13] Jr., H.W.L., 1983. Integer programming with a fixed number of variables. Mathematics of Operations Research 8, 538–548. doi:10.1287/MOOR.8.4.538.

[14] Kawarabayashi, K., Kreutzer, S., 2015. Towards the Graph Minor Theorems for Directed Graphs, in: Proc. of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP), pp. 3–10. doi:10.1007/978-3-662-47666-6_1.

[15] Korhonen, T., Lokshtanov, D., 2023. An improved parameterized algorithm for treewidth, in: Proc. of the 55th Annual ACM Symposium on Theory of Computing (STOC), pp. 528–541. doi:10.1145/3564246.3585245.

[16] Lampis, M., Kaouri, G., Mitsou, V., 2011. On the algorithmic effectiveness of digraph decompositions and complexity measures. Discrete Optimization 8, 129–138. doi:10.1016/J.DISOPT.2010.03.010.

[17] Lopes, R., Sau, I., 2022. A relaxation of the directed disjoint paths problem: A global congestion metric helps. Theoretical Computer Science 898, 75–91. doi:10.1016/J.TCS.2021.10.023.

[18] Maalouly, N.E., 2023. Exact Matching: Algorithms and Related Problems, in: Proc. of the 40th International Symposium on Theoretical Aspects of Computer Science (STACS), pp. 29:1–29:17. doi:10.4230/LIPICS.STACS.2023.29.

[19] Maalouly, N.E., 2024. Towards a Deterministic Polynomial Time Algorithm for the Exact Matching Problem. Ph.D. thesis. ETH Zurich, Zürich, Switzerland. doi:10.3929/ETHZ-B-000675444.

[20] Maalouly, N.E., Haslebacher, S., Wulf, L., 2024. On the exact matching problem in dense graphs, in: Proc. of the 41st International Symposium on Theoretical Aspects of Computer Science (STACS), pp. 33:1–33:17. doi:10.4230/LIPICS.STACS.2024.33.

[21] Marx, D., Pilipczuk, M., 2014. Everything you always wanted to know about the parameterized complexity of Subgraph Isomorphism (but were afraid to ask), in: Proc. of the 31st International Symposium on Theoretical Aspects of Computer Science (STACS), pp. 542–553. doi:10.4230/LIPICS.STACS.2014.542.

[22] Mulmuley, K., Vazirani, U.V., Vazirani, V.V., 1987. Matching is as easy as matrix inversion. Combinatorica 7, 105–113. doi:10.1007/BF02579206.

[23] de Oliveira Oliveira, M., 2016. An algorithmic metatheorem for directed treewidth. Discrete Applied Mathematics 204, 49–76. doi:10.1016/j.dam.2015.10.020.

[24] Plaisted, D.A., Zaks, S., 1980. An NP-complete matching problem. Discrete Applied Mathematics 2, 65–72. doi:10.1016/0166-218X(80)90055-4.

[25] Reed, B.A., 1999. Introducing directed tree width. Electronic Notes in Discrete Mathematics 3, 222–229. doi:10.1016/S1571-0653(05)80061-7.

[26] Robertson, N., Seymour, P.D., 1990. Graph minors. IV. tree-width and well-quasi-ordering. Journal of Combinatorial Theory, Series B 48, 227–254. doi:10.1016/0095-8956(90)90120-O.

[27] Slivkins, A., 2010. Parameterized tractability of edge-disjoint paths on directed acyclic graphs. SIAM Journal on Discrete Mathematics 24, 146–157. doi:10.1137/070697781.