

# On the Directed Full Degree Spanning Tree Problem

Daniel Lokshtanov<sup>2</sup>, Venkatesh Raman<sup>1</sup>, Saket Saurabh<sup>2</sup>, and Somnath Sikdar<sup>1</sup>

<sup>1</sup> The Institute of Mathematical Sciences, India.<sup>1</sup>  
vraman@imsc.res.in, sikdar@cs.rwth-aachen.de

<sup>2</sup> The University of Bergen, Norway.<sup>2</sup>  
daniello@ii.uib.no, saket@imsc.res.in

**Abstract.** We study the parameterized complexity of a directed analog of the FULL DEGREE SPANNING TREE problem where, given a digraph  $D$  and a nonnegative integer  $k$ , the goal is to construct a spanning out-tree  $T$  of  $D$  such that at least  $k$  vertices in  $T$  have the same out-degree as in  $D$ . We show that this problem is  $W[1]$ -hard even on the class of directed acyclic graphs. In the dual version, called REDUCED DEGREE SPANNING TREE, one is required to construct a spanning out-tree  $T$  such that at most  $k$  vertices in  $T$  have out-degrees that are different from that in  $D$ . We show that this problem is fixed-parameter tractable and that it admits a problem kernel with at most  $8k$  vertices on strongly connected digraphs and  $O(k^2)$  vertices on general digraphs. We also give an algorithm for this problem on general digraphs with running time  $O(5.942^k \cdot n^{O(1)})$ , where  $n$  is the number of vertices in the input digraph.

## 1 Introduction

The FULL DEGREE SPANNING TREE problem asks, given a connected undirected graph  $G$  and a nonnegative integer  $k$  as inputs, whether  $G$  has a spanning tree  $T$  in which at least  $k$  vertices have the same degree in  $T$  as in  $G$ . This problem was first studied by Pothof and Schut [22] in the context of water distribution networks where the goal is to determine the flow in a network by installing a small number of flow-meters. It so happens that to measure the flow in each pipe of the network, it is sufficient to find a spanning tree of the network and install flow-meters at those vertices whose degree in the spanning tree is smaller than that in the network. To find the optimal number of flow-meters (an expensive equipment), one needs to find a spanning tree with the largest number of vertices of full degree.

This problem has attracted a lot of attention [4, 6, 19, 17, 16]. Bhatia et al. [4] studied this problem from the point-of-view of approximation algorithms and gave a factor- $\Theta(\sqrt{n})$  algorithm for it, where  $n$  is the number of vertices in the input graph. They also showed that this problem admits no factor  $O(n^{1/2-\epsilon})$  approximation algorithm unless  $\text{NP} = \text{co-R}$ . For planar graphs, a polynomial-time approximation scheme (PTAS) was presented. Independently, Broersma et al. [6] developed a PTAS for planar graphs and showed that this problem can be solved in polynomial time in special classes of graphs such as bounded treewidth graphs and co-comparability graphs. Guo et al. [17] studied the parameterized complexity of this problem and showed it to be  $W[1]$ -hard. Gaspers et al. [16] give an  $O(1.9465^n \cdot n^{O(1)})$  algorithm for the optimization version of this problem.

One can parameterize the  $d$ -FDST problem from the “other end” and ask whether a graph  $G$  has spanning tree  $T$  in which at most  $k$  vertices have degrees different from that in  $G$ . This problem has been studied under the name VERTEX FEEDBACK EDGE SET and is defined as follows. Given a connected undirected graph  $G = (V, E)$  and a nonnegative integer  $k$ , find an edge subset  $E'$  incident on at most  $k$  vertices such that  $G[E \setminus E']$  is acyclic. Note that if there exists such an edge set  $E'$ , then there exists  $E'' \subseteq E'$  such that  $G[E \setminus E'']$

is a spanning tree in which at most  $k$  vertices have degrees different from that in  $G$ . Khuller et al. [19] show that this problem is MAX SNP-hard and describe a  $(2 + \epsilon)$ -approximation algorithm for it for any fixed  $\epsilon > 0$ . Guo et al. [17] show that this problem is fixed-parameter tractable by demonstrating a problem kernel with at most  $4k$  vertices.

In this paper, we consider a natural generalization of these problems to directed graphs. An *oriented tree* is a tree in the undirected sense each of whose edges has been assigned a direction. We say that a subdigraph  $T$  of a directed graph  $D = (V, A)$  is an *out-tree* if it is an oriented tree with exactly one vertex  $s$  of in-degree zero (called the *root*). An out-tree that contains all vertices of  $D$  is an *out-branching* of  $D$ . Given a digraph  $D = (V, A)$  and an out-tree  $T$  of  $D$ , we say that a vertex  $v \in V$  is of *full degree* if its out-degree in  $T$  is the same as that in  $D$ ; otherwise,  $v$  is said to be of *reduced degree*. We define the DIRECTED FULL DEGREE SPANNING TREE ( $d$ -FDST) problem as follows.

*Input:* Given a directed graph  $D = (V, A)$  and a nonnegative integer  $k$ .  
*Parameter:* The integer  $k$ .  
*Question:* Does there exist an out-branching of  $D$  in which at least  $k$  vertices are of full degree?

We call the dual of this problem the DIRECTED REDUCED DEGREE SPANNING TREE ( $d$ -RDST) problem.

*Input:* Given a directed graph  $D = (V, A)$  and a nonnegative integer  $k$ .  
*Parameter:* The integer  $k$ .  
*Question:* Does there exist an out-branching of  $D$  in which at most  $k$  vertices are of reduced degree?

*Our Contribution.* We study the parameterized complexity of the problems  $d$ -FDST and  $d$ -RDST. We show that  $d$ -FDST is  $W[1]$ -hard even in the class of directed acyclic graphs (DAGs) by a reduction from INDEPENDENT SET. We show that  $d$ -RDST is fixed-parameter tractable by exhibiting a problem kernel with at most  $O(k^2)$  vertices. For strongly connected digraphs,  $d$ -RDST admits a kernel with at most  $8k$  vertices. We also design a branching algorithm for the  $d$ -RDST problem with running time  $O(5.942^k \cdot n^{O(1)})$ , where  $n$  is the number of vertices in the input digraph.

*Related Results.* The FULL DEGREE SPANNING TREE problem is one of the many variants of the generic CONSTRAINED SPANNING TREE problem, where one is required to find a spanning tree of a given (di)graph subject to certain constraints. This class of problems has been studied intensely [1, 8, 10, 12, 15, 13, 18, 23].

In [12], the authors consider the problem MAX LEAF SPANNING TREE where one is required to find a spanning tree of an undirected graph with the maximum number of leaves. When parameterized by the solution size, this problem admits a kernel with  $3.75k$  vertices. In the directed variant of this problem, one has to decide whether an input digraph  $D$  has an out-branching with at least  $k$  leaves. This problem admits a kernel with  $O(k^3)$  vertices, provided the root of the out-branching is *given as part of the input* [13], and has an algorithm with run-time  $O(3.72^k \cdot n^{O(1)})$  [10]. Another such problem is MAX INTERNAL SPANNING TREE, where the objective is to find a spanning tree (or an out-branching, in case of digraphs) with

at least  $k$  internal vertices. For undirected graphs, a  $3k$ -vertex kernel and an algorithm with running time  $O(8^k \cdot n^{O(1)})$  is known for this problem [15]. For directed graphs, an  $O(k^2)$ -vertex kernel due to [18] and an algorithm with running time  $O(40^k \cdot n^{O(1)})$  due to [8] is known.

*Organization of the Paper.* In Section 2 we define the relevant notions related to digraphs and parameterized complexity. In Section 3 we show that  $d$ -FDST is W[1]-hard even when the input digraph is restricted to be a DAG. In Section 4 we show that the  $d$ -RDST problem is fixed-parameter tractable by demonstrating a kernel with at most  $O(k^2)$  vertices. We first demonstrate a kernel with  $8k$  vertices for strongly connected digraphs and use the ideas therein to develop the  $O(k^2)$  kernel for general digraphs. In Section 5 we develop an algorithm for the  $d$ -RDST problem with running time  $O(5.942^k \cdot n^{O(1)})$ . Finally in Section 6, we end with some concluding remarks and open questions.

## 2 Preliminaries

In this section we fix our notation and terminology. We first discuss terminology related to digraphs and then provide the basic definitions from parameterized complexity needed in this paper.

### 2.1 Digraphs: Basic Terminology

The notation and terminology that we follow are from [3]. Given a digraph  $D$  we let  $V(D)$  and  $A(D)$  denote the vertex set and arc set, respectively, of  $D$ . If  $u, v \in V(D)$ , we say that  $u$  is an *in-neighbour* (*out-neighbour*) of  $v$  if  $(u, v) \in A(D)$  ( $(v, u) \in A(D)$ ). The in-degree  $d^-(u)$  (out-degree  $d^+(u)$ ) of  $u$  is the number of in-neighbours (out-neighbours) of  $u$ . Given a subset  $V' \subseteq V(D)$ , we let  $D[V']$  denote the digraph induced on  $V'$ . The *underlying undirected graph*  $U(D)$  is the undirected graph obtained from  $D$  by disregarding the orientation of arcs and deleting an edge for each pair of parallel edges in the resulting graph. The *connectivity components* of  $D$  are the subdigraphs induced by the vertices of components of  $U(D)$ .

A digraph is *oriented* if every pair of vertices has at most one arc between them. A  $(v_1, v_s)$ -*walk* in  $D = (V, A)$  is a sequence  $v_1, \dots, v_s$  of vertices such that  $(v_i, v_{i+1}) \in A$  for all  $1 \leq i \leq s - 1$ . A *dicycle* is a walk  $v_1, v_2, \dots, v_s$  such that  $s \geq 3$ , the vertices  $v_1, \dots, v_{s-1}$  are distinct and  $v_1 = v_s$ . A digraph with no dicycles is called a *directed acyclic graph* (DAG). A digraph  $D$  is *strongly connected* if for every pair of distinct vertices  $u, v \in V(D)$ , there exists a  $(u, v)$ -walk and a  $(v, u)$ -walk. A *strong component* of a digraph is a maximal induced subdigraph that is strongly connected. The *strong component digraph*  $SC(D)$  is the directed acyclic graph obtained by contracting each strong component to a single vertex and deleting any parallel arcs obtained in this process. A strong component  $S$  of a digraph  $D$  is a *source strong component* if no vertex in  $S$  has an in-neighbour in  $V(D) \setminus V(S)$ . The following is a necessary and sufficient condition for a digraph to have an out-branching.

**Proposition 1** ([3]) *A digraph  $D$  has an out-branching if and only if  $D$  has a unique source strong component.*

One can obtain the strongly connected components of a digraph  $D$  in time  $O(n + m)$  [7], where  $n = |V(D)|$  and  $m = |A(D)|$ . Each strong component can be stored as an  $n$ -bit vector where the  $i$ th bit is a one if and only if vertex  $i$  is in the strong component. It is now easy to see that one can verify in  $(n + m)$  time whether there exists a unique source strong component.

## 2.2 Parameterized Complexity

A parameterized problem is a subset of  $\Sigma^* \times \mathbb{Z}^{\geq 0}$ , where  $\Sigma$  is a finite alphabet and  $\mathbb{Z}^{\geq 0}$  is the set of nonnegative numbers. An instance of a parameterized problem is therefore a pair  $(I, k)$ , where  $k$  is the parameter. In the framework of parameterized complexity, the running time of an algorithm is viewed as a function of two quantities: the size of the problem instance *and* the parameter. A parameterized problem is said to be *fixed-parameter tractable (fpt)* if there exists an algorithm that takes as input  $(I, k)$  and decides whether it is a YES or NO-instance in time  $O(f(k) \cdot |I|^{O(1)})$ , where  $f$  is a function depending only on  $k$ . The class FPT consists of all fixed parameter tractable problems.

Closely related to the notion of an FPT-algorithm is the concept of a kernel. A *kernelization of a parameterized problem*  $Q$  is a polynomial-time many-one reduction from  $Q$  to  $Q$  that maps a given instance  $(I, k)$  to an equivalent instance  $(I', k')$  such that  $|I'| \leq g(k)$  and  $k' \leq h(k)$ , where  $g$  and  $h$  are two computable functions. The two instances are equivalent in the sense that  $(I, k)$  is a YES-instance if and only if  $(I', k')$  is a YES-instance. The function  $g$  is called the *size of the kernel*. It is well-known that a parameterized problem has a kernelization if and only if it is in FPT [11]. However even if an FPT-algorithm is known for an NP-complete problem, it is the case that the size of the kernel obtained from it is exponential in the parameter. Moreover an FPT-algorithm gives no clue as to the best possible kernel that can be obtained for the problem. It is therefore of independent interest to consider kernelization algorithms for parameterized problems.

A parameterized problem  $\pi_1$  is *fixed-parameter reducible* to a parameterized problem  $\pi_2$  if there exist functions  $f, g : \mathbb{Z}^{\geq 0} \rightarrow \mathbb{Z}^{\geq 0}$ ,  $\Phi : \Sigma^* \times \mathbb{Z}^{\geq 0} \rightarrow \Sigma^*$  and a polynomial  $p(\cdot)$  such that for any instance  $(I, k)$  of  $\pi_1$ ,  $(\Phi(I, k), g(k))$  is an instance of  $\pi_2$  computable in time  $f(k) \cdot p(|I|)$  and  $(I, k) \in \pi_1$  if and only if  $(\Phi(I, k), g(k)) \in \pi_2$ . Two parameterized problems are *fixed-parameter equivalent* if they are fixed-parameter reducible to each other. The basic complexity class for fixed-parameter intractability is W[1] as there is strong evidence that W-hard problems are not fixed-parameter tractable. To show that a problem is W-hard, one needs to exhibit a fixed-parameter reduction from a known W-hard problem to the problem at hand. For more on parameterized complexity see [11, 14, 21].

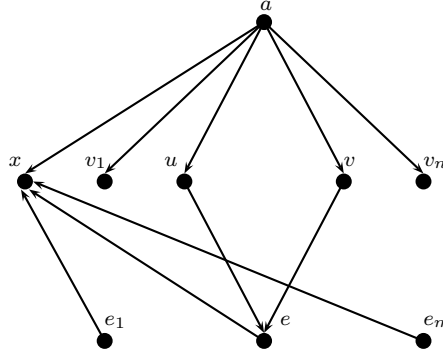
## 3 The $d$ -FDST Problem

We now show that  $d$ -FDST is W[1]-hard two important digraph classes: DAGs and strongly connected digraphs. This is a modification of the reduction presented in [4] (Lemma 3.2).

**Theorem 1.** *The  $d$ -FDST problem is W[1]-hard on directed acyclic graphs (DAGs) and strongly connected digraphs. Also the  $d$ -RDST problem is NP-hard on strongly connected digraphs.*

*Proof.* We show that  $k$ -INDEPENDENT SET, which is known to be W[1]-complete [11], fixed-parameter reduces to the  $d$ -FDST problem. Let  $(G, k)$  be an instance of the  $k$ -INDEPENDENT SET problem where we assume  $G$  to be a *connected* undirected graph on  $n$  vertices and  $m$  edges. Construct a directed graph  $D$  as follows. The vertex set  $V(D)$  consists of  $n + m + 2$  vertices:  $v_1, \dots, v_n, e_1, \dots, e_m, a, x$ , where the vertices  $v_i$ , for  $1 \leq i \leq n$ , and  $e_j$ , for  $1 \leq j \leq m$ , “correspond”, respectively, to the vertices and edges of  $G$  and  $a, x$  are two special vertices. The digraph  $D$  can be viewed as a three-layer graph. Layer one consists of vertex  $a$ . Layer

two consists of the vertices  $x, v_1, \dots, v_n$  and vertex  $a$  has an out-arc to each vertex in layer two. Layer three consists of the vertices  $e_1, \dots, e_m$  and each  $e_j$ , for  $1 \leq j \leq m$ , has an out-arc to vertex  $x$ . If  $e = \{u, v\} \in E(G)$  then the vertices  $u$  and  $v$  in layer two have an out-arc each to vertex  $e$  in layer three. This completes the description of  $D$ . It is easy to verify that  $D$  is a DAG.



**Fig. 1.** The digraph  $D$ .

Observe that in any out-branching  $T$  of  $D$  every vertex (except the root) has exactly one in-neighbor and that:

1. Vertices  $a, e_1, \dots, e_m$  are the only vertices of  $D$  of in-degree zero and therefore the root of  $T$  must be one of these. Moreover, if  $a$  preserves its out-degree in  $T$  then vertices  $e_1, \dots, e_m$  must be of reduced degree.
2. At most one vertex from among  $e_1, \dots, e_m$  can preserve its out-degree in  $T$  because each of them has an out-arc to  $x$ .
3. Vertex  $x$  preserves its out-degree in  $T$  because  $x$  is of out-degree zero.

We claim that  $G$  has an independent set of size  $k$  if and only if the digraph  $D$  has an out-branching with  $k + 2$  vertices of full degree. Suppose that  $G$  has a  $k$ -independent set on the vertices  $v_{i_1}, \dots, v_{i_k}$ . Consider the subdigraph  $T'$  induced by the vertices  $a, v_{i_1}, \dots, v_{i_k}$  and their out-neighbors. It is easy to verify that  $T'$  is actually an out-tree rooted at  $a$  in which vertices  $a, x, v_{i_1}, \dots, v_{i_k}$  have full degree. For each edge  $e \in E(G)$  that is not incident to any vertex in the  $k$ -independent set, arbitrarily choose one of its endpoints, say  $v$ , and add the arc  $(v, e)$  to the out-tree  $T'$ . This converts the out-tree  $T'$  into an out-branching  $T$  with at least  $k + 2$  vertices of full degree. Conversely suppose that  $D$  admits an out-branching  $T$  in which at least  $k + 2$  vertices preserve their out-degree. We consider two cases.

*Case 1.* Vertex  $a$  preserves its out-degree. Then no vertex from layer three preserves its out-degree, as each of these vertices has an out-arc to  $x$ . Since  $x$  is the other vertex of full degree, it must be that  $k$  vertices from among  $v_1, \dots, v_n$  preserve their out-degree. No pair from among these  $k$  vertices form an edge in  $G$ , for otherwise, they would have an out-arc to the same vertex  $e$  in layer three in  $T$  and this would contradict the assumption that  $T$  is an out-branching. Hence these  $k$  vertices must be independent in  $G$ .

*Case 2.* Vertex  $a$  does not preserve its out-degree. By Observation 2, at most one vertex from layer three can preserve its out-degree and, by Observation 3,  $x$  preserves its out-degree in every out-branching. Hence at least  $k$  vertices from among the  $v_1, \dots, v_n$  preserve their out-degree. These vertices form a  $k$ -independent set in  $G$ . This shows that  $d$ -FDST is W[1]-hard on DAGs.

By modifying the above reduction from  $k$ -INDEPENDENT SET, we show that  $d$ -FDST is W[1]-hard on the class of strongly connected digraphs (and hence that the  $d$ -RDST problem is NP-hard on this class of digraphs). Given an instance  $(G, k)$  of  $k$ -INDEPENDENT SET, it is no loss of generality to assume that  $G$  is a *connected non-bipartite* graph. Construct the digraph  $D$  as above with just one modification: add the arc  $(x, a)$ . It is easy to verify that the resulting digraph is strongly connected. Then  $G$  has an independent set of size  $k$  if and only if  $D$  admits an out-branching with  $k+2$  vertices of full degree. Suppose  $G$  has a  $k$ -independent set on the vertex set  $\{v_{i_1}, \dots, v_{i_k}\}$ . Since  $G$  is non-bipartite, there exists an edge  $e_j$  both of whose endpoints are in  $V(G) \setminus \{v_{i_1}, \dots, v_{i_k}\}$ . It is easy to see that there is an out-branching with  $e_j$  as root in which the vertices  $e_j, x, v_{i_1}, \dots, v_{i_k}$  are of full degree. Conversely suppose that  $D$  has an out-branching with  $k+2$  vertices of full degree. Between  $a$  and  $x$ , at most one can preserve its out-degree and among  $a, e_1, \dots, e_m$  at most one can preserve its out-degree. Therefore at least  $k$  vertices from among  $v_1, \dots, v_n$  preserve their out-degree. These vertices form an independent set in  $G$ . This shows that  $d$ -RDST is NP-hard on strongly connected digraphs.

This completes the proof of the theorem. □

## 4 $d$ -RDST: A Problem Kernel

In this section we show that  $d$ -RDST admits a problem-kernel with  $O(k^2)$  vertices and is therefore fixed-parameter tractable. We first consider the special case when the input digraph is strongly connected and establish a kernel with  $8k$  vertices for this case. This will give some insight as to how to tackle the general case.

Observe that if  $(D, k)$  is a YES-instance of the  $d$ -RDST problem and  $T$  is a solution out-branching (one in which at most  $k$  vertices are of reduced out-degree), then the subdigraph of  $D$  induced by the vertices of full degree is a forest in the undirected sense and hence has treewidth one (for more on treewidth, see [5, 20]). Therefore the underlying undirected graph  $U(D)$  has treewidth at most  $k+1$ . Moreover one can show that the property of having an out-branching with at most  $k$  vertices of reduced out-degree is expressible in monadic second-order logic [9]. One can now use the results of Arnborg et al. [2] to conclude that for every fixed  $k$  the  $d$ -RDST problem can be decided in linear time. This shows that the  $d$ -RDST problem is fixed-parameter tractable. However the running time dependence of this algorithm on  $k$  is huge making it impractical. In what follows, we give an alternative algorithm with a more well-behaved dependence on the parameter  $k$ .

### 4.1 A Linear Kernel for Strongly Connected Digraphs

We actually establish the  $8k$ -vertex kernel for a more general class of digraphs, those in which every vertex has out-degree at least one. Call this class of digraphs *out-degree at least one digraphs* and denote it by  $\mathcal{D}_1^+$ . It is easy to see that strongly connected digraphs (SCDs) is

a subclass of  $\mathcal{D}_1^+$ . Since a digraph in  $\mathcal{D}_1^+$  can have vertices of in-degree zero, it follows that SCDs form a proper subclass of  $\mathcal{D}_1^+$ .

A common technique to establish a kernel is to devise a set of *reduction rules* which when applied to the input instance (in some specified sequence) produces the kernel. Recall that a *reduction rule* for a parameterized problem  $Q$  is a polynomial-time algorithm that takes an input  $(I, k)$  of  $Q$  and

1. either correctly decides the instance, or
2. outputs an “equivalent” instance  $(I', k')$  of  $Q$  such that  $k' \leq k$ .

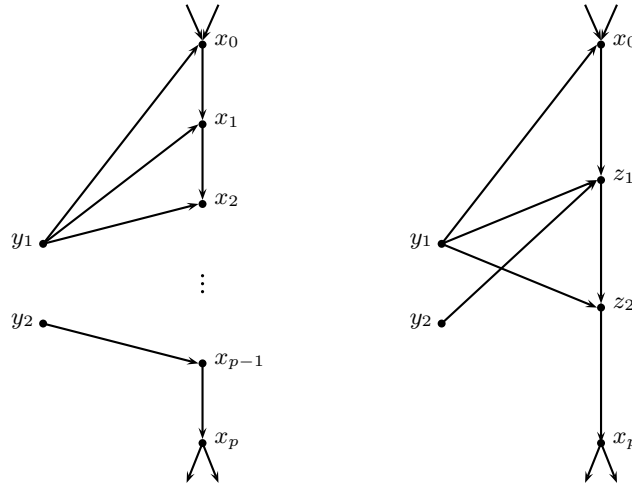
Two instances are *equivalent* if they are both YES-instances or both NO-instances. An instance  $(I, k)$  of a parameterized problem  $Q$  is *reduced with respect to a set  $\mathcal{R}$  of reduction rules* if the instance  $(D', k')$  output by any reduction rule in  $\mathcal{R}$  is the original instance  $(D, k)$  itself.

There are three simple reduction rules for the case where the input is an  $\mathcal{D}_1^+$ -digraph. We assume that the input is  $(D, k)$ .

**Rule 1.** If there exists  $u \in V(D)$  such that  $d^-(u) \geq k + 2$  then return NO; else return  $(D, k)$ .

**Rule 2.** If there are  $k+1$  vertices of out-degree at least  $k+1$  then return NO; else return  $(D, k)$ .

**Rule 3 (The Path Rule).** Let  $x_0, x_1, \dots, x_{p-1}, x_p$  be a sequence of vertices in  $D$  such that  $p \geq 4$  and for  $0 \leq i \leq p-1$  we have  $d^+(x_i) = 1$  and  $(x_i, x_{i+1}) \in A(D)$ . Let  $Y_0$  be the set of in-neighbours of  $x_1, \dots, x_{p-1}$  and let  $Y := Y_0 \setminus \{x_0, x_1, \dots, x_{p-2}\}$ . Delete the vertices  $x_1, \dots, x_{p-1}$  and add two new vertices  $z_1, z_2$  and the arcs  $(x_0, z_1), (z_1, z_2), (z_2, x_p)$ . If  $y \in Y$  has at least two out-neighbors in  $\{x_1, \dots, x_{p-1}\}$  then add arcs  $(y, z_1), (y, z_2)$ . If  $y \in Y$  has exactly one out-neighbor in  $\{x_1, \dots, x_{p-1}\}$  then add the arc  $(y, z_1)$ . Return  $(D, k)$ . See Figure 2.



**Fig. 2.** Illustrating the *Path Rule*: the left and right-hand sides show, respectively, the situation before and after the transformation. Vertex  $y_1$  has two neighbors and vertex  $y_2$  just one neighbor in the set  $\{x_1, \dots, x_{p-1}\}$ .

It is easy to see that Rules 1 and 2 are indeed reduction rules for the  $d$ -RDST problem on out-degree at least one digraphs. If a vertex  $v$  has in-degree at least  $k + 2$  then at least  $k + 1$

in-neighbors of  $u$  must be of reduced degree in any out-branching. This shows that Rule 1 is a reduction rule. If a vertex  $u$  has out-degree  $k + 1$  and is of full degree in some out-branching  $T$  then  $T$  has at least  $k + 1$  leaves. Since the input digraph is such that every vertex has out-degree at least one, this means that in  $T$  there are at least  $k + 1$  vertices of reduced degree. This shows that any vertex of out-degree  $k + 1$  must necessarily be of reduced degree in any solution out-branching. Therefore if there are  $k + 1$  such vertices the given instance is a NO-instance. This proves that Rule 2 is a reduction rule.

**Lemma 1.** *Rule 3 is a reduction rule for the  $d$ -RDST problem.*

*Proof.* It is sufficient to show that if  $(D', k)$  is the instance obtained by one application of Rule 3 to an instance  $(D, k)$ , then  $D$  has an out-branching with at most  $k$  vertices of reduced out-degree if and only if  $D'$  has an out-branching with at most  $k$  vertices of reduced degree.

Suppose  $D'$  has an out-branching  $T'$  with at most  $k$  vertices of reduced degree. There are two cases to consider. In the first case, there are no arcs from  $Y$  to  $z_1$  or  $z_2$  in  $T'$ . In this case we may assume without loss of generality that the path  $x_0 \rightarrow z_1 \rightarrow z_2$  occurs as a sub-path of  $T'$ . For if  $x_0 \rightarrow z_1 \rightarrow z_2$  is not a subpath of  $T'$ , then one of  $z_1$  or  $z_2$  has in-degree zero in  $T'$ . Hence it must be that either  $z_1$  or  $z_2$  is the root of  $T'$ . If  $z_1$  is the root of  $T'$  then  $x_0$  is a leaf in  $T'$ ; if  $z_2$  is the root then  $z_1$  is a leaf. In either case, we can make  $x_0$  the root and maintain the path  $x_0 \rightarrow z_1 \rightarrow z_2$  without increasing the number of vertices of reduced degree. In order to construct an out-branching  $T$  for  $D$ , replace  $x_0 \rightarrow z_1 \rightarrow z_2$  by the path  $x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_{p-1}$ . Moreover if  $T'$  contains the arc  $(z_2, x_p)$  then, in constructing  $T$ , add the arc  $(x_{p-1}, x_p)$ . Note that  $T$  has at most  $k$  vertices of reduced degree.

In the second case, there exists at least one vertex  $y \in Y$  with arcs to  $\{z_1, z_2\}$ . Suppose that  $T'$  contains the arcs  $(y_1, z_1), (y_2, z_2)$ , where  $y_1$  and  $y_2$  are (not necessarily distinct) vertices in  $Y$ . Note that both  $x_0$  and  $z_1$  are of out-degree zero in  $T'$  and hence of reduced degree. Observe that  $T' \setminus \{z_1\}$  is an out-branching for  $D' \setminus \{z_1\}$  as  $z_1$  is a leaf in  $T'$ . We transform  $T'$  into another out-branching for  $D'$  by deleting the arc  $(y_1, z_1)$  and adding the arc  $(x_0, z_1)$ . In this new out-branching,  $x_0$  is of full degree and  $y_1$  is of reduced degree but the number of vertices of reduced degree does not increase.

We can therefore assume without loss of generality that in  $T'$  there is exactly one vertex  $y \in Y$  with an out-arc to  $\{z_1, z_2\}$ . Suppose  $(y, z_2) \in A(T')$ . Then  $y$  must be of reduced degree as whenever we have an arc  $(y, z_2)$ , we also have an arc  $(y, z_1)$ . In this case we transform  $T'$  by deleting the arcs  $(y, z_2), (x_0, z_1)$  and introducing the arcs  $(y, z_1), (z_1, z_2)$ . The resulting digraph is an out-branching with at most  $k$  vertices of reduced degree as  $x_0$  now is of reduced degree but  $z_1$  is of full degree. Therefore we are left to consider the case when  $y$  has an arc to  $z_1$  only. Let  $x_s$  be the first out-neighbor of  $y$  in  $\{x_1, \dots, x_{p-1}\}$ . Delete  $z_1, z_2$  and connect  $x_0$  to the dipath  $x_1 \rightarrow \dots \rightarrow x_{s-1}$  and  $y$  to the dipath  $x_s \rightarrow \dots \rightarrow x_{p-1}$ . Add the arc  $(x_{p-1}, x_p)$  if  $(z_2, x_p) \in A(T')$ . The resulting digraph is an out-branching for  $D$  with at most  $k$  vertices of reduced degree.

To prove the converse, suppose that  $D$  has an out-branching  $T$  with at most  $k$  vertices of reduced degree. Again there are two cases to consider.

*Case 1.* There are no arcs from  $Y$  to any  $x_i$ , for  $1 \leq i \leq p - 1$ , in  $T$ . There are two sub-cases here. Either  $T$  contains the dipath  $x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_{p-1} \rightarrow x_p$ , in which case we can compress it to the path  $(x_0, z_1, z_2, x_p)$  to obtain an out-branching  $T'$  for  $D'$  with at most  $k$  vertices of reduced degree. Otherwise one of the vertices  $x_1, \dots, x_p$  must be the root of  $T$ . If  $x_p$  is



the root, then  $T$  contains the dipath  $x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_{p-1}$  and we replace it by  $(x_0, z_1, z_2)$  to obtain an out-branching  $T'$  of  $D'$ . If one of  $x_1, \dots, x_{p-1}$  is the root, then delete  $x_1, \dots, x_{p-1}$ , make  $z_1$  the root and add the arcs  $(z_1, z_2), (z_2, x_p)$ . This transforms  $T$  into an out-branching of  $D'$  with at most  $k$  vertices of reduced degree.

*Case 2.* Now suppose that in  $T$  the vertices  $y_{i_1}, \dots, y_{i_s} \in Y$  have out-neighbors in  $x_1, \dots, x_{p-1}$ . Since  $T$  is an out-branching, the set of out-neighbors of  $y_{i_j}$  and  $y_{i_l}$  are disjoint for all  $j \neq l$ . In  $T$ , the out-neighbors of  $y_{i_j}$  in  $\{x_1, \dots, x_{p-1}\}$  can be ordered in the natural way according to their position in the path  $x_1 \rightarrow \dots \rightarrow x_{p-1}$ . Let  $x_{q_j}$  be the first out-neighbor of  $y_{i_j}$  among  $\{x_1, \dots, x_{p-1}\}$  in  $T$ . Transform  $T$  into a digraph  $T_1$  by deleting out-arcs such that for  $1 \leq j \leq s$ , the only out-neighbor of  $y_{i_j}$  among  $\{x_1, \dots, x_{p-1}\}$  is  $x_{q_j}$ . Sort the vertices  $y_{i_1}, \dots, y_{i_s}$  in increasing order based on the order of the vertices  $x_{q_j}$  in the path  $x_1 \rightarrow \dots \rightarrow x_{p-1}$ . Without loss of generality we assume that the sorted order is also  $y_{i_1}, \dots, y_{i_s}$ .

Transform  $T_1$  yet again by connecting the vertices  $x_i$  so that the resulting digraph (which we continue to call  $T_1$ ) contains the paths:

- $x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_{q_1-1}$ ;
- $y_{i_j} \rightarrow x_{q_j} \rightarrow \dots \rightarrow x_{q_{j+1}-1}$ , for  $1 \leq j \leq s-1$ ;
- $y_{i_s} \rightarrow x_{q_s} \rightarrow \dots \rightarrow x_{p-1} \dots$ .

Note that in  $T$ , the vertices  $x_{q_j-1}$ , for  $1 \leq j \leq s$ , are of out-degree zero. Moreover the last path  $y_{i_s} \rightarrow x_{q_s} \rightarrow \dots$  in the sequence contains all vertices  $x_{q_s}, \dots, x_{p-1}$  but may or may not contain the vertex  $x_p$ . Observe that  $T_1$  is an out-tree and that the only vertices  $y \in \{y_{i_1}, \dots, y_{i_s}\}$  whose out-degree is reduced in this transformation had at least two out-neighbors among the vertices  $\{x_1, \dots, x_{p-1}\}$  in  $T$ . Hence for every  $y_{i_j}$  whose out-degree is reduced in transforming  $T$  to  $T_1$ , there exists a distinct vertex in  $x_1, \dots, x_{p-1}$  of out-degree zero in  $T$  which is of full degree in  $T_1$ . Thus the number of vertices of reduced degree does not change in this transformation.

Now delete the arcs  $(y_{i_1}, x_{q_1}), \dots, (y_{i_{s-1}}, x_{q_{s-1}})$ , and add  $(x_{q_1-1}, x_{q_1}), \dots, (x_{q_{s-1}-1}, x_{q_{s-1}})$  so that the resulting digraph  $T_2$  contains the path  $x_0 \rightarrow \dots \rightarrow x_{q_{s-1}}$ . In this transformation, the vertices which possibly have their out-degree reduced are  $y_{i_j}$ , for  $1 \leq j \leq s-1$ , but an equal number of vertices  $x_{q_j-1}$ , for  $1 \leq j \leq s-1$ , attain full degree. Therefore the number of vertices of reduced out-degree in  $T_2$  is at most that in  $T_1$ . To obtain an out-branching of  $D'$  from  $T_2$ , delete  $x_1, \dots, x_{q_{s-1}}$ , add the arcs  $(y_{i_s}, z_1), (z_1, z_2)$  and connect  $z_2$  to the out-neighbor of  $x_{p-1}$ , if any. Note that this transforms  $T_2$  into an out-branching of  $D'$  with at most  $k$  vertices of reduced degree.

This completes the proof of the lemma. □

It is easy to see that Rules 1 and 2 can be applied in  $O(n)$  time and that Rule 3 can be applied in  $O(n+m)$  time. Note that Rule 3 is *parameter independent*, that is, an application of the rule does not affect the parameter. Consequently, it makes sense to talk about a digraph being reduced with respect to Rule 3 as distinct from an instance of  $d$ -RDST being reduced with respect to Rule 3. Our kernelization algorithm consists in applying Rules 1 to 3 repeatedly until the given instance is reduced.

We next describe a lemma that we repeatedly make use of in the sequel. Given a directed graph  $D$ , we let  $V_i(D) \subseteq V(D)$  denote the set of vertices of out-degree  $i$ ;  $V_{\geq i}(D) \subseteq V(D)$  denotes the set of vertices of out-degree at least  $i$ .

**Lemma 2.** *Let  $D$  be a directed graph reduced with respect to the Path Rule (Rule 3) and let  $T$  be an out-branching of  $D$  with root  $r$  such that  $X$  is the set of vertices of reduced out-degree. Then*

$$|V(T)| \leq 4|V_0(T) \cup V_{\geq 2}(T) \cup X| \leq 4(|V_0(T)| + |X \cup V_0(T)|).$$

*Proof.* If we view the out-branching  $T$  as an undirected graph,  $V_0(T)$  is the set of leaves and  $V_{\geq 2}(T)$  is the set of vertices of degree at least three along with the root  $r$ , if  $d_T^+(r) \geq 2$ . Thus  $V_{\geq 2}(T)$  has at most one vertex of total degree two and all other vertices are of total degree at least three. It is a well-known fact that a tree with  $l$  leaves has at most  $l - 1$  internal vertices of degree at least three. Since  $V_{\geq 2}(T)$  has at most one vertex of total degree two, we have  $|V_{\geq 2}(T)| \leq |V_0(T)|$ .

Now consider the vertices of the out-branching  $T$  which have out-degree exactly one. Define  $W := X \cup V_0(T) \cup V_{\geq 2}(T)$  and let  $\mathcal{P}$  be the set of maximal dipaths in  $T$  such that for any dipath  $P = x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_p$  in  $\mathcal{P}$  we have that (1)  $d_D^+(x_i) = 1$  for  $0 \leq i \leq p - 1$ , and (2)  $x_p \in W$ . Observe that every vertex with out-degree exactly one in  $T$  is contained in exactly one path in  $\mathcal{P}$ . Also observe that the set of vertices of out-degree exactly one in  $T$  not contained in  $W$  is precisely the set  $V_1(T) \setminus X$ . Therefore  $|V_1(T) \setminus X| \leq \sum_{P \in \mathcal{P}} (|P| - 1)$ , where  $|P|$  denotes the number of vertices in the path  $P$ . By Rule 3, any dipath  $P \in \mathcal{P}$  has at most four vertices and since the number of dipaths in  $\mathcal{P}$  is at most  $|W|$ , we have

$$|V_1(T) \setminus X| \leq 3 \cdot |\mathcal{P}| \leq 3 \cdot |W| \leq 3|X \cup V_0(T) \cup V_{\geq 2}(T)|.$$

Since  $|V(T)| \leq |V_1(T) \setminus X| + |X \cup V_0(T) \cup V_{\geq 2}(T)|$ , we have

$$|V(T)| \leq 4|V_0(T) \cup V_{\geq 2}(T) \cup X| \leq 4(|V_0(T)| + |V_0(T) \cup X|).$$

This completes the proof of the lemma.  $\square$

We can now bound the size of a YES-instance of the  $d$ -RDST problem on  $\mathcal{D}_1^+$ -digraphs that have been reduced with respect to Rules 1 to 3.

**Theorem 2.** *Let  $(D, k)$  be a YES-instance of the  $d$ -RDST problem on out-degree at least one digraphs reduced with respect to Rules 1 to 3. Then  $|V(D)| \leq 8k$ .*

*Proof.* Since  $(D, k)$  is a YES-instance of the problem, let  $T$  be an out-branching of  $D$  and let  $X$  be the set of vertices of reduced degree in  $T$ , where  $|X| \leq k$ . Every vertex of  $D$  is of out-degree at least one and hence  $V_0 \subseteq X$ , where  $V_0$  is the set of leaves in  $T$ . Consequently  $|X \cup V_0| \leq k$  and  $|V_0| \leq k$  and by Lemma 2, we have  $|V(T)| \leq 8k$ , as claimed.  $\square$

Observe that the crucial step in the proof above was to bound the number of leaves in the solution out-branching. For  $\mathcal{D}_1^+$ -digraphs this is easy since every leaf is a vertex of reduced degree. This is not the case with general digraphs which may have an arbitrary number of vertices of out-degree zero, all of which are of full degree in any out-branching. In the next subsection we present a set of reduction rules for the  $d$ -RDST problem in general digraphs which help us bound the number of vertices of out-degree zero in terms of the parameter  $k$ .

## 4.2 An $O(k^2)$ -Vertex Kernel in General Digraphs

For general digraphs, we first consider an *annotated version* of the problem as this seems to help in developing reduction rules. Eventually we will revert to the original unannotated version. An instance of the annotated version consists of a triplet  $(D, X, k)$  where  $D$  and  $k$  are, respectively, the input digraph and the parameter, and  $X$  is a subset of  $V(D)$  such that in *any* out-branching with at most  $|X| + k$  vertices of reduced degree, the vertices of  $X$  must be of reduced degree. The question in this case is to decide whether  $D$  admits an out-branching where the set of vertices of reduced degree is  $X \cup S$ , where  $S \subseteq V(D) \setminus X$  and  $|S| \leq k$ . Call such an out-branching a *solution out-branching*. To obtain a kernel for  $d$ -RDST, we apply the reduction rules to an instance  $(D, k)$  after setting  $X = \emptyset$ .

Given an instance  $(D, X, k)$ , we define the *conflict set* of a vertex  $u \in V(D) \setminus X$  as

$$C(u) := \{v \in V(D) \setminus X : N^+(u) \cap N^+(v) \neq \emptyset\}.$$

Clearly vertices of out-degree zero have an empty conflict set. If a vertex  $v$  has a non-empty conflict set then in any out-branching either  $v$  has its degree reduced or *every* vertex in  $C(v)$  has its degree reduced. Moreover if  $u \in C(v)$  then  $v \in C(u)$  and in this case we say that  $u$  and  $v$  are in conflict. The *conflict number* of  $D$  is defined as  $c(D) := \sum_{v \in V(D) \setminus X} |C(v)|$ .

We assume that the input instance is  $(D, X, k)$  and the kernelization algorithm consists in applying each reduction rule repeatedly, in the order given below, until no longer possible. Therefore when we say that Rule  $i$  is indeed a reduction rule we assume that the input instance is reduced with respect to the rules preceding it.

**Rule 0.** If  $u \in X$  and  $d^+(u) = 1$ , delete the out-arc from  $u$  and return  $(D, X, k)$ .

The vertices in  $X$  are of reduced degree in any solution out-branching. Thus if a vertex in  $X$  has out-degree exactly one, this out-arc will never be part of a solution out-branching, and deleting it will not change the solution structure.

**Rule 1.** If there exists  $u \in V(D)$  such that the number of in-neighbors of  $u$  in  $V(D) \setminus X$  is at least  $k + 2$  then return NO; else return  $(D, X, k)$ .

In the last subsection, we already showed that this rule is indeed a reduction rule.

**Rule 2.** If  $u \in V(D) \setminus X$  and  $|C(u)| > k$ , set  $X \leftarrow X \cup \{u\}$  and  $k \leftarrow k - 1$ . Furthermore if  $d^+(u) = 1$  then delete the out-arc from  $u$  and return  $(D, X, k)$ .

If the conflict set  $C(u)$  of  $u \in V(D) \setminus X$  is of size at least  $k + 1$  and if  $u$  is of full degree in some out-branching  $T$ , then every vertex in  $C(u)$  must be of reduced degree in  $T$ . Therefore if  $(D, X, k)$  is a YES-instance then  $u$  must have its degree reduced in *every* solution out-branching. In addition, if  $u$  has out-degree exactly one, then this out-arc cannot be part of any solution out-branching and can be deleted. This shows that Rule 2 is a reduction rule.

**Rule 3.** If  $c(D) > 2k^2$  then return NO, else return  $(D, X, k)$ .

**Lemma 3.** *Rule 3 is a reduction rule for the  $d$ -RDST problem.*

*Proof.* To see why Rule 3 qualifies to be a reduction rule, construct the *conflict graph*  $\mathcal{C}_{D,X}$  of the instance  $(D, X, k)$  which is defined as follows. The vertex set  $V(\mathcal{C}_{D,X}) := V(D) \setminus X$  and two vertices in  $V(\mathcal{C}_{D,X})$  have an edge between them if and only if they are in conflict. Since the size of the conflict set of any vertex is at most  $k$  (as  $D$  is reduced with respect to Rule 2), the degree of any vertex in  $\mathcal{C}_{D,X}$  is at most  $k$ . The key observation is that if  $T$  is any solution out-branching of  $(D, X, k)$  in which the set of vertices of reduced degree is  $X \cup S$  with  $S \subseteq V(D) \setminus X$ , then  $S$  forms a vertex cover of  $\mathcal{C}_{D,X}$ . Since we require that  $|S| \leq k$ , the number of edges in  $\mathcal{C}_{D,X}$  is at most  $k^2$ . For a vertex  $v \in V(D) \setminus X$ , let  $d'(v)$  be the number of neighbors of vertex  $v$  in the conflict graph  $\mathcal{C}_{D,X}$ . Observe that  $c(D) := \sum_{v \in V(D) \setminus X} |C(v)| = \sum_{v \in V(D) \setminus X} d'(v) \leq 2k^2$ . The last inequality follows from the fact that sum of degrees of vertices in a graph is equal to twice the number of edges.  $\square$

**Rule 4.** If  $u \in V(D)$  such that  $d^+(u) = 0$  and  $d^-(u) = 1$  then delete  $u$  from  $D$  and return  $(D, X, k)$ .

It is easy to see that Rule 4 is a reduction rule: vertex  $u$  is of full degree in any solution and it does not determine whether its parent is of full or reduced degree in a solution out-branching and therefore can be deleted. To obtain a solution out-branching for  $D$  from a solution  $T'$  for  $D \setminus v$ , simply add the arc between  $u$  and its parent in  $T'$ .

**Rule 5.** Let  $u \in V(D)$  be of out-degree zero and let  $v_1, \dots, v_r$  be its in-neighbors, where  $r > 2$ . Delete  $u$  and add  $\binom{r}{2}$  new vertices  $u_{ij}$ , where  $1 \leq i < j \leq r$ ; for a newly added vertex  $u_{ij}$  add the arcs  $(v_i, u_{ij})$  and  $(v_j, u_{ij})$ . Return  $(D, X, k)$ .

Note that vertex  $u$  forces at least  $r - 1$  vertices from  $\{v_1, \dots, v_r\}$  to be of reduced degree in any out-branching of  $D$ . This situation is captured by deleting  $u$  and introducing  $\binom{r}{2}$  vertices as described in the rule. These  $\binom{r}{2}$  vertices then force at least  $r - 1$  vertices from  $\{v_1, \dots, v_r\}$  to be of reduced degree in any out-branching of the transformed graph. The upshot is that each vertex of out-degree zero has in-degree exactly two.

**Lemma 4.** *Rule 5 is a reduction rule for the  $d$ -RDST problem.*

*Proof.* Let  $(D, X, k)$  and  $(D', X, k)$  be the instances of  $d$ -RDST before and after one application of Rule 5, respectively. We claim that  $(D, X, k)$  is a YES-instance if and only if  $(D', X, k)$  is a YES-instance.

Let  $T$  be an out-branching of  $D$  that certifies that  $(D, X, k)$  is a YES-instance. Then at least  $r - 1$  vertices from  $\{v_1, \dots, v_r\}$  are of reduced degree in  $T$ . Transform  $T$  into an out-branching  $T'$  for  $(D', X, k)$  as follows. Delete  $u$  from  $T$  and introduce the vertices  $u_{ij}$  for  $1 \leq i < j \leq r$ . If  $v_i \in \{v_1, \dots, v_r\}$  was of full degree in  $T$  then in  $T'$  add the arcs  $(v_i, u_{pq})$  for all  $1 \leq p < q \leq r$ ; otherwise add the arcs  $(v_1, u_{pq})$  for all  $1 \leq p < q \leq r$ . The out-branching  $T'$  certifies that  $(D', X, k)$  is a YES-instance.

Conversely suppose that the out-branching  $T'$  certifies that  $(D', X, k)$  is a YES-instance. Again at least  $r - 1$  vertices from  $\{v_1, \dots, v_r\}$  are of reduced degree in  $T'$ . Transform  $T'$  into an out-branching  $T$  for  $(D, X, k)$  as follows. Delete the vertices  $u_{ij}$  for  $1 \leq i < j \leq r$  and introduce vertex  $u$ . If  $v_i \in \{v_1, \dots, v_r\}$  was of full degree in  $T'$ , add the arc  $(v_i, u)$  in  $T$ ; otherwise add the arc  $(v_1, u)$ . Clearly  $T$  certifies that  $(D, X, k)$  is a YES-instance.  $\square$

**Rule 6.** If  $u, v \in V(D) \setminus X$  have  $p > 1$  common out-neighbors of out-degree zero, delete all but one of them. Return  $(D, X, k)$ .

**Rule 7.** If  $u \in V(D)$  is of out-degree zero such that at least one in-neighbor of  $u$  is in  $X$ , delete  $u$ . Return  $(D, X, k)$ .

By Rule 5, it is clear that if  $u, v \in V(D) \setminus X$  have at least two common out-neighbors of out-degree zero then these out-neighbors have in-degree exactly two. It is intuitively clear that these out-neighbors are equivalent in some sense and it suffices to preserve just one of them. It is easy to show that the original instance has a solution out-branching if and only if the instance obtained by one application of Rule 6 has a solution out-branching. As for Rule 7, if  $u$  has two in-neighbors  $v$  and  $w$  and if  $v \in X$ , we can delete the arc  $(v, u)$  without altering the solution structure. But then  $v$  is a private neighbor of  $w$  of out-degree zero and hence can be deleted by Rule 4.

**Rule 8 (The Path Rule).** Let  $x_0, x_1, \dots, x_{p-1}, x_p$  be a sequence of vertices in  $D$  such that  $p \geq 4$  and for  $0 \leq i \leq p-1$  we have  $d^+(x_i) = 1$  and  $(x_i, x_{i+1}) \in A(D)$ . Let  $Y_0$  be the set of in-neighbours of  $x_1, \dots, x_{p-1}$  and let  $Y := Y_0 \setminus \{x_0, x_1, \dots, x_{p-2}\}$ . Delete the vertices  $x_1, \dots, x_{p-1}$  and add two new vertices  $z_1, z_2$  and the arcs  $(x_0, z_1), (z_1, z_2), (z_2, x_p)$ . If  $y \in Y$  has at least two out-neighbors in  $\{x_1, \dots, x_{p-1}\}$  then add arcs  $(y, z_1), (y, z_2)$ . If  $y \in Y$  has exactly one out-neighbor in  $\{x_1, \dots, x_{p-1}\}$  then add the arc  $(y, z_1)$ . Return  $(D, X, k)$ .

This is Rule 3 from the previous subsection where it was shown to be a reduction rule for the  $d$ -RDST problem (note that the proof of Lemma 1 did not use the fact that the input was an out-degree at least one digraph). By Rule 0, no vertex on the path  $x_0, x_1, \dots, x_{p-1}$  is in  $X$  and therefore the proof of Lemma 1 continues to hold for the annotated case as well.

It is easy to see that a single application of Rules 5 or 6 takes time  $O(n^2)$ ; all other rules take time  $O(n+m)$ . We are now ready to bound the number of vertices of out-degree zero in a reduced instance of the annotated problem.

**Lemma 5.** *Let  $(D, X, k)$  be a YES-instance of the annotated  $d$ -RDST problem that is reduced with respect to Rules 0 through 8 mentioned above. Then the number of vertices of out-degree zero in  $D$  is at most  $k^2$ .*

*Proof.* Let  $u$  be a vertex of out-degree zero. By Rules 4 and 5, it must have exactly two in-neighbors, say,  $x$  and  $y$ . By Rule 7, neither  $x$  nor  $y$  is in  $X$  and are therefore still in conflict in the reduced graph. Hence, either  $x$  or  $y$  must be of reduced degree in any solution out-branching. Furthermore any vertex not in  $X$  can have at most  $k$  out-neighbors of out-degree zero since, by Rule 2, any vertex not in  $X$  is in conflict with at most  $k$  other vertices and, by Rule 6, two vertices in conflict can have at most one common out-neighbor of out-degree zero. Since  $(D, X, k)$  is assumed to be a YES-instance, at most  $k$  vertices can lose their out-degree in any solution out-branching. Moreover, by Rule 4, any vertex of out-degree zero is an out-neighbor of at least one vertex of reduced degree. Therefore the total number of vertices of out-degree zero is at most  $k^2$ .  $\square$

**Lemma 6.** *Let  $(D, k)$  be a YES-instance of the  $d$ -RDST problem and suppose that  $(D_1, X, k_1)$  is an instance of the annotated  $d$ -RDST problem reduced with respect to Rules 0 through 8 by repeatedly applying them on  $(D, k)$ , by initially setting  $X = \emptyset$ . Then  $|V(D_1)| \leq 8(k^2 + k)$ .*

*Proof.* Since reduction rules map YES-instances to YES-instances and does not allow the parameter to increase, it is clear that  $(D_1, X, k_1)$  is a YES-instance of the annotated  $d$ -RDST

problem and that  $k_1 + |X| \leq k$ . Therefore let  $T_1$  be a solution out-branching of  $(D_1, X, k_1)$ . A leaf of  $T_1$  is either a vertex of out-degree zero in  $D_1$  or a vertex of reduced degree. By Lemma 5, the total number of vertices of out-degree zero in  $D_1$  is at most  $k_1^2 \leq k^2$  and since  $T_1$  is a solution out-branching, the total number of vertices of reduced degree is at most  $k_1 + |X| \leq k$ . Thus the number of leaves of  $T_1$  is at most  $k^2 + k$  and by Lemma 2 we have  $|V(T_1)| \leq 4(k^2 + k + k^2 + k) = 8(k^2 + k)$ .  $\square$

We now show how to obtain a kernel for the original (unannotated) version of the problem. Let  $(D, k)$  be an instance of the  $d$ -RDST problem and let  $(D', X, k')$  be the instance obtained by applying reduction rules 0 through 8 on  $(D, k)$  until no longer possible, by initially setting  $X = \emptyset$ . By Lemma 6, we know that if  $(D, k)$  is a YES-instance then  $|V(D')| \leq 8(k^2 + k)$  and that  $k' + |X| = k$ . To get back an instance of the unannotated version, apply the following transformation on  $(D', X, k')$ . If  $X \neq \emptyset$ , add a directed path  $Y = y_1, \dots, y_{k+2}$  to  $D'$  and for  $x \in X$  add the out-arc  $(x, y_i)$  for  $1 \leq i \leq k + 2$ . Call the resulting digraph  $D''$ .

We claim that  $(D', X, k')$  has a solution out-branching  $T'$  with at most  $|X| + k'$  vertices of reduced degree and where all vertices in  $X$  have their degree reduced if and only if  $D''$  admits an out-branching with at most  $k$  vertices of reduced degree. Suppose  $T'$  is a solution out-branching for  $(D', X, k')$ . To obtain a solution out-branching  $T''$  of  $D''$  simply add the path  $Y$  to  $T'$  and the out-arc  $(x_1, y_1)$ . Clearly  $T''$  has at most  $k' + |X|$  vertices of reduced degree. Conversely let  $T''$  be a solution out-branching for  $D''$ . First note that every vertex in  $X$  must be of reduced degree in  $T''$ . For if  $x \in X$  is of full degree then  $k + 1$  vertices  $y_1, \dots, y_{k+1}$  are of reduced degree, contradicting the fact that  $T''$  has at most  $k$  vertices of reduced degree. Therefore we may assume that, in  $T''$ , vertex  $x_1$  has an out-arc to the start vertex  $y_1$  of the path  $y_1, \dots, y_{k+2}$  which appears as is in the out-branching. That is, we may assume that the vertices in the path  $Y$  are always of full degree in any out-branching and that there exists at most  $k$  vertices in  $V(D'') \setminus (V(Y) \cup X)$  of reduced degree in  $T''$ . To obtain a solution out-branching  $T'$  for  $(D', X, k')$  simply delete the path  $Y$  from  $T''$ .

Since we add at most  $k + 2$  vertices in this transformation, we have

**Theorem 3.** *The  $d$ -RDST problem admits a problem kernel with at most  $8k^2 + 9k + 2$  vertices.*

## 5 An Algorithm for the $d$ -RDST Problem

In this section we describe a branching algorithm for  $d$ -RDST with running time  $O(5.942^k \cdot n^{O(1)})$ . We first observe that in order to construct a solution out-branching of a given digraph, it is sufficient to know which vertices will be of reduced degree.

**Lemma 7.** *Let  $D = (V, A)$  be a digraph and let  $X$  be the set of vertices of reduced degree in some out-branching of  $D$ . Given  $D$  and  $X$ , one can in polynomial time construct an out-branching of  $D$  in which the vertices of reduced degree is a subset of  $X$ .*

*Proof.* We describe an algorithm that constructs such an out-branching of  $D$ . Given  $D$  and  $X$ , our algorithm first constructs a digraph  $D'$  with vertex set  $V(D)$  in which

1. all vertices in  $V(D) \setminus X$  are connected to their out-neighbors in  $D$  by *solid* arcs;
2. a vertex  $x \in X$  has a *dotted* out-arc to a vertex  $y$  if  $(x, y) \in A(D)$  and  $y$  has no solid in-arc in  $D'$ .

We are guaranteed that there exists an out-branching of  $D'$  in which all solid arcs are present but in which one or more dotted arcs may be missing. Note that in  $D'$ , a vertex with a solid in-arc has no other (solid or dotted) in-arcs.

Our algorithm now runs through all possible choices of the root of the proposed out-branching. For each choice of root, it does a modified breadth-first search (BFS) starting at the root. In the modified BFS-routine, when the algorithm visits a vertex  $v$ , it solidifies all dotted out-arcs from  $v$ , if any. For each dotted arc  $(v, w)$  that it solidifies, it deletes all dotted in-arcs to  $w$ . The algorithm then inserts the out-neighbors of  $v$  in the BFS-queue. If the BFS-tree thus constructed includes all vertices of  $D'$ , the algorithm outputs this out-branching, or else, moves on to the next choice of root.

*Claim.* Suppose that  $r$  is the root of an out-branching of  $D'$  in which  $X$  is the set of vertices of reduced degree. Then the above algorithm, on selecting  $r$  as root, succeeds in constructing an out-branching in which the vertices of reduced degree is a subset of  $X$ .

In order to prove this claim, it is sufficient to show that in the BFS-tree  $T$  constructed by the algorithm with  $r$  as root, every vertex of  $D'$  is reachable from  $r$ . This suffices because the algorithm ensures that every vertex in  $V(D') \setminus X$  is of full degree in  $T$ .

Therefore let  $v$  be a vertex not reachable from  $r$  such that the distance, in  $D'$ , from  $v$  to  $r$  is the shortest among all vertices not reachable from  $r$  in  $T$ . Let  $r, v_1, \dots, v_l, v$  be a shortest dipath from  $r$  to  $v$  in  $D'$ . By our choice of  $v$ , all vertices  $v_1, \dots, v_l$  are reachable from  $r$  in  $T$ . Note that the arc  $(v_l, v)$  must have been dotted and in fact all in-arcs to  $v$  were dotted in  $D'$ . When the algorithm visited  $v_l$ , the only reason it could not solidify the arc  $(v_l, v)$  must have been because  $v$  *already* had a solid in-arc into it and hence the arc  $(v_l, v)$  had already been deleted. Suppose that  $v$  has a solid in-arc from  $u$ . Then  $u$  must have already been visited *before*  $v_l$  at which time the dotted arc  $(u, v)$  was solidified. But this means that  $u$ , and hence  $v$ , is reachable from  $r$  in the BFS-tree  $T$ , a contradiction.  $\square$

We now have an  $O(k^{O(k)} \cdot n^{O(1)})$  algorithm for the  $d$ -RDST problem: Given  $(D, k)$ , we first obtain a kernel of size  $O(k^2)$  using Theorem 3 and then run over all possible vertex-subsets  $X$  of the kernel of size at most  $k$  to determine the set of vertices of reduced degree. Then using Lemma 7, we verify whether one can indeed construct an out-branching in which the set of vertices of reduced degree is  $X$ .

In the rest of this section, we give an improved algorithm with running time  $O(c^k \cdot n^{O(1)})$ , for a constant  $c$ . Our algorithm (see Figure 3) is based on the simple observation that if two vertices  $u$  and  $v$  of the input digraph  $D$  have a common out-neighbor then one of them must be of reduced degree in *any* out-branching of  $D$ . The algorithm recurses on vertex-pairs that have a common out-neighbor and, along each branch of the recursion tree, builds a set  $X$  of vertices which would be the candidate vertices of reduced degree in the out-branching that it attempts to construct. When there are no vertices to branch on, it reduces the instance  $(D, X, k)$  with respect to the following rules.

**Rule 1'**. If  $u \in X$  and  $d^+(u) = 1$ , delete the out-arc from  $u$  and return  $(D, X, k)$ . (This is Rule 0 from Section 4.2.)

**Rule 2'**. Let  $u \in V(D)$  be of out-degree zero and let  $v_1, \dots, v_r$  be its in-neighbors. If  $v_i \in X$  for all  $1 \leq i \leq r$ , assign  $v_1$  as the parent of  $u$  and delete  $u$ . If there exists  $1 \leq i \leq r$  such that  $v_i \notin X$  then assign  $v_i$  as the parent of  $u$  and delete  $u$ . Return  $(D, X, k)$ .

**Rule 3'**. This is Rule 8 from Section 4.2.

**RDST** ( $D, X, k$ )

*Input:* A digraph  $D = (V, A)$ ;  $X \subseteq V$ , such that the vertices in  $X$  will be of reduced degree in the out-branching that is being constructed; an integer parameter  $k$ . The algorithm is initially called after setting  $X = \emptyset$ .

*Output:* An out-branching of  $D$  in which every vertex of  $X$  is of reduced degree and with at most  $k$  vertices of reduced degree in total, if one exists, or NO, signifying that no such out-branching exists.

1. If  $k < 0$  or  $|X| > k$  return NO.
2. If no two vertices in  $V(D) \setminus X$  have a common out-neighbor then
  - (a) Reduce  $(D, X, k)$  with respect to Rules 1' through 5'.
  - (b) For each  $(k - |X|)$ -sized subset  $Y$  of  $V(D) \setminus X$ , check if there exists an out-branching of  $D$  in which the vertex set of reduced degree is  $X \cup Y$ . If yes, then “expand” this out-branching to an out-branching for the original instance and return the solution; else return NO.
3. Let  $u, v \in V(D) \setminus X$  be two vertices with a common out-neighbor then
  - (a)  $X \leftarrow X \cup \{u\}$ ;  $Z = \text{Call } \mathbf{RDST}(D, X, k - 1)$ .
  - (b) If  $Z \neq \text{NO}$  then return  $Z$ .
  - (c)  $X \leftarrow X \cup \{v\}$ ; Return  $\mathbf{RDST}(D, X, k - 1)$ .

**Fig. 3.** Algorithm **RDST**.

Rule 1' is a reduction rule because a vertex of out-degree exactly one that is of reduced degree must necessarily lose its only out-arc. As for Rule 2', we know that in the instance  $(D, X, k)$  obtained after the algorithm finishes branching, no two vertices of  $V(D) \setminus X$  have a common out-neighbor and therefore at least  $r - 1$  in-neighbors of  $u$  must be in  $X$  (and of reduced degree). If all in-neighbors of  $u$  are of reduced degree, we arbitrarily fix one of them as parent of  $u$  (so that we can construct an out-branching of the original instance later on) and delete  $u$ . If exactly  $r - 1$  in-neighbors of  $u$  are already of reduced degree, we choose that in-neighbor not in  $X$  as the parent of  $u$  and delete  $u$ . Also note that when applying Rule 3' to a path  $x_0, x_1, \dots, x_{p-1}, x_p$ , the vertices  $x_0, x_1, \dots, x_{p-1}$  are not in  $X$ , by Rule 1'. Therefore if  $Y$  is the set of in-neighbors of  $x_1, \dots, x_{p-1}$ , excluding  $\{x_0, x_1, \dots, x_{p-2}\}$ , then  $Y \subseteq X$ .

Observe the following:

1. By Rule 2', no vertex in the reduced instance  $(D, X, k)$  has out-degree zero.
2. Every vertex in the subdigraph induced by  $V(D) \setminus X$  has in-degree at most one and hence each connectivity component (a connected component in the undirected sense) is either a dicycle, or an out-tree or a dicycle which has out-trees rooted at its vertices. Thus each connectivity component has at most one dicycle and if a component does have a dicycle then it can be transformed into an out-branching by deleting an arc from the cycle. Such a digraph is called a *pseudo out-forest* [24].

We now reduce the instance  $(D, X, k)$  with respect to the following two rules:

**Rule 4'.** If at least  $k + 1 - |X|$  connectivity components of  $D[V \setminus X]$  contain dicycles, then return NO; else return  $(D, X, k)$ .

**Rule 5'.** If a connectivity component of  $D[V \setminus X]$  is a dicycle  $C$  such that no vertex in  $V(C)$  has an out-neighbor in  $X$ , pick a vertex  $u \in X$  with an arc to  $C$  and fix it as the “entry point” to  $C$ ; delete  $C$  and set  $k \leftarrow k - 1$ ; return  $(D, X, k)$ .



Rule 4' is a reduction rule as every connectivity component that has a dicycle contains at least one vertex that will be of reduced degree. If the number of such components is at least  $k + 1 - |X|$ , one cannot construct an out-branching with at most  $k$  vertices of reduced degree where all vertices in  $X$  have their degree reduced. To see that Rule 5' is a reduction rule, first note that since  $C$  has no out-arcs, it cannot contain the root of the proposed out-branching. Any path from the root to  $C$  must necessarily include a vertex from  $X$  and it does not matter which arc out of  $X$  we use to get to  $C$ , since every vertex in  $X$  has its degree reduced anyway. Moreover, in any out-branching, exactly one vertex of  $C$  must be of reduced degree. Therefore if  $(D', X', k')$  is the instance obtained by one application of Rule 5' to the instance  $(D, X, k)$ , then it is easy to see that these instances must be equivalent. Also note that each application of Rule 1' through 5' takes time  $O(n + m)$ .

**Lemma 8.** *Let  $(D, X, k)$  be an instance of the  $d$ -RDST problem in which no two vertices of  $V(D) \setminus X$  have a common out-neighbor, and reduced with respect to Rules 1' through 5'. Then  $|V(D) \setminus X| \leq 7|X|$ .*

*Proof.* Let  $D'$  be a digraph obtained from  $D$  by deleting all out-arcs from the vertices in  $X$ . Therefore in  $D'$ , every vertex of  $X$  has out-degree zero and in-degree at most one. We show that a connectivity component of  $D'$  that has  $p$  vertices of  $X$  has at most  $7p$  vertices of  $V(D') \setminus X$ . This will prove the lemma.

If a connectivity component of  $D'$  is an out-tree  $T'$ , then every leaf of this out-tree is a vertex of  $X$ . If  $T'$  has  $p$  leaves, then applying Lemma 2 to  $T'$ , we have that  $|V(T')| \leq 8p$ . Since exactly  $p$  of these vertices are from  $X$ , the number of vertices of  $V(D') \setminus X$  in the out-tree is at most  $7p$ . Therefore let  $R$  be a connectivity component of  $D'$  containing a dicycle such that  $|V(R) \cap X| = p$ . Then  $R$  has exactly one dicycle, say  $C$ . By Rule 5',  $C$  has a vertex  $x$  with an out-neighbor in  $V(R) \setminus V(C)$ , and therefore  $d_R^+(x) \geq 2$ . Let  $y$  be the out-neighbor of  $x$  in  $C$ . Delete the arc  $(x, y)$  to obtain an out-branching  $T$  with root  $y$ . Note that the number of leaves in  $T$  is the same as that in  $R$ . Moreover in transforming  $R$  to  $T$ , only one vertex (namely  $x$ ) loses its out-degree. By Lemma 2,

$$|V(T)| \leq 4|x \cup V_0(T) \cup V_{\geq 2}(T)| \leq 4|V_0(T)| + 4|x \cup V_{\geq 2}(T)|,$$

and since  $|x \cup V_{\geq 2}(T)| \leq 1 + (|V_0(T)| - 1) = p$ , we have  $|V(T)| \leq 8p$ . Consequently  $|V(R) \setminus X| \leq 7p$ .

This completes the proof of the lemma.  $\square$

To construct an out-branching, it is sufficient to choose the remaining  $k - |X|$  vertices of reduced degree from the vertices in  $V(D) \setminus X$ . Setting  $|X| = c$ , the exponential term in the running time of the algorithm is bounded above by the function

$$\sum_{c=0}^k 2^c \cdot \binom{7c}{k-c} \leq k \cdot \max_{0 \leq c \leq k} 2^c \cdot \binom{7c}{k-c}.$$

We will show that the function  $\max_{0 \leq c \leq k} 2^c \cdot \binom{7c}{k-c}$  is bounded above by  $(k + 1) \cdot 5.942^k$ .

**Theorem 4.** *Given a digraph  $D$  and a nonnegative integer  $k$ , one can decide whether  $D$  has an out-branching with at most  $k$  vertices of reduced degree, and if so, construct such an out-branching in time  $O(5.942^k \cdot n^{O(1)})$ .*

Finally we prove the claimed upper-bound for the function  $\max_{0 \leq c \leq k} 2^c \cdot \binom{7c}{k-c}$ . We first need a lemma.

**Lemma 9.** *For nonnegative integers  $k \leq n$  and any real  $x > 0$ ,*

$$\binom{n}{k} \leq \frac{(1+x)^n}{x^k}.$$

*Proof.* Since

$$x^{-k}(1+x)^k = x^{-k} \cdot \sum_{i=0}^n \binom{n}{i} x^i = \binom{n}{k} + \Delta,$$

where  $\Delta = \sum_{0 \leq i \leq n, i \neq k} \binom{n}{i} x^{i-k} \geq 0$ , the result follows immediately.  $\square$

**Lemma 10.** *For a nonnegative integer  $k$ , the function*

$$h(k) = \max_{0 \leq c \leq k} 2^c \cdot \binom{7c}{k-c}$$

*is bounded above by  $(k+1) \cdot 5.942^k$ .*

*Proof.* Using the bound in Lemma 9, we have that for any  $0 \leq c \leq k$  and any  $x > 0$ ,

$$2^c \cdot \binom{7c}{k-c} \leq 2^c \cdot \frac{(1+x)^{7c}}{x^{k-c}} = \frac{(2x(1+x)^7)^c}{x^k}.$$

Since the above inequality holds for *any*  $x > 0$ , it holds, in particular, for the positive roots of the equation  $2x(1+x)^7 - 1 = 0$ . This equation has exactly one positive root and its value is approximately 0.16830. Substituting this value, we obtain

$$2^c \cdot \binom{7c}{k-c} \leq \frac{1}{0.16830^k} \leq 5.942^k,$$

and since there are  $k+1$  such terms in  $h(k)$ , the value of  $h(k)$  is at most  $(k+1) \cdot 5.942^k$ .  $\square$

## 6 Concluding Remarks

We studied a natural generalization of the FULL DEGREE SPANNING TREE problem to directed graphs. We showed that the  $d$ -FDST problem is W[1]-hard even on the class of DAGs and that the  $d$ -RDST problem is fixed-parameter tractable. For the  $d$ -RDST problem, we obtained a kernel with at most  $8k^2 + 9k + 2$  vertices and an algorithm with running time  $O(5.942^k \cdot n^{O(1)})$ . Natural open questions are to investigate whether  $d$ -RDST admits a linear-vertex kernel and design algorithms with better running times.

## References

1. N. ALON, F.V. FOMIN, G. GUTIN, M. KRIVELEVICH AND S. SAURABH. *Spanning Directed Trees With Many Leaves*. To appear in SIAM Journal of Discrete Math.
2. S. ARNBORG, J. LAGERGREN AND D. SEESE. *Easy Problems for Tree-Decomposable Graphs*. Journal of Algorithms, Vol. 12, pp. 308-340, 1991.

3. J. BANG-JENSEN AND G. GUTIN. *Digraphs: Theory, Algorithms and Applications*. Springer-Verlag, 2000.
4. R. BHATIA, S. KHULLER, R. PLESS AND Y. SUSSMANN. *The Full-Degree Spanning Tree Problem*. Networks, Vol. 36, pp. 203-209, 2000.
5. H.L. BODLAENDER AND A.M.C.A. KOSTER. *Combinatorial Optimization on Graphs of Bounded Treewidth*. The Computer Journal, Vol. 51, Issue 3, 2008.
6. H.J. BROERSMA, A. HUCK, T. KLOKS, O. KOPPIOS, D. KRATSCHE, H. MÜLLER AND H. TUINSTRAS. *Degree-Preserving Forests*. Networks, Vol. 35, pp. 26-39, 2000.
7. T.H. CORMEN, C.E. LIESERSON, R.L. RIVEST AND C. STEIN. *Introduction to Algorithms*, 2nd Edition. M.I.T. Press, 2001.
8. N. COHEN, F.V. FOMIN, G. GUTIN, E.J. KIM, S. SAURABH AND A. YEO. *Algorithms for Finding  $k$ -Vertex Out-Trees and its Applications to the  $k$ -Internal Out-Branching Problem*. Manuscript.
9. B. COURCELLE. *The Expression of Graph Properties and Graph Transformations in Monadic Second-Order Logic*. Handbook of Graph Grammars and Computing by Graph Transformation. Vol. 1: Foundations, pp. 313-400. World Scientific Publishing Co. Inc., 1997.
10. J. DALIGAULT, G. GUTIN, E.J. KIM AND A. YEO. *FPT-Algorithms and Kernels for the Directed  $k$ -Leaf Problem*. Journal of Computer and System Sciences, Vol. 76, Issue 2, pp. 144-152, 2010.
11. R.G. DOWNEY AND M.R. FELLOWS. *Parameterized Complexity*. Springer-Verlag, 1999.
12. V. ESTIVILL-CASTRO, M.R. FELLOWS, M.A. LANGSTON AND F.A. ROSAMOND. *FPT is P-Time Extremal Structure I*. In Proc. of Algorithms and Complexity in Durham (ACiD), 2005.
13. H. FERNAU, F.V. FOMIN, D. LOKSHTANOV, D. RAIBLE, S. SAURABH AND Y. VILLANGER. *Kernel(s) for Problems With No Kernels: On Out-Trees with Many Leaves*. In Proc. of the 26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009, pp. 421-432, Schloss Dagstuhl-Leibnitz-Zentrum für Informatik, Germany, 2009.
14. J. FLUM AND M. GROHE. *Parameterized Complexity Theory*. Springer-Verlag, 2006.
15. F.V. FOMIN, S. GASPERIS, S. SAURABH AND S. THOMASSE. *A Linear Vertex Kernel for Maximum Internal Spanning Tree*. In Proc. of the 20th International Symposium on Algorithms and Computation (ISAAC 2009), Springer LNCS, Vol. 5878, pp. 275-282, 2009.
16. S. GASPERIS, S. SAURABH AND A.A. STEPANOV. *A Moderately Exponential-Time Algorithm for Full-Degree Spanning Tree*. In Proc. of the 5th Annual Conference on Theory and Applications of Models of Computation (TAMC08), Springer LNCS, Vol. 4978, pp. 478-489, 2008.
17. J. GUO, R. NIEDERMEIER, AND S. WERNICKE. *Fixed-Parameter Tractability Results for Full-Degree Spanning Tree and Its Dual*. In Proc. the 2nd International Workshop on Parameterized and Exact Computation (IWPEC), Springer LNCS, Vol. 4196, pp. 203-214, 2006.
18. G. GUTIN, E.J. KIM AND I. RAZGON. *Minimum Leaf Out-Branching and Related Problems*. In Proc. of the 4th International Conference on Algorithmic Aspects in Information and Management (AAIM'08), Springer LNCS Vol. 5034, pp. 235-246, 2008.
19. S. KHULLER, R. BHATIA AND R. PLESS. *On Local Search and Placement of Meters in Networks*. SIAM Journal on Computing Vol. 32, Issue 2, pp. 470-487, 2003.
20. T. KLOKS. *Treewidth: Computations and Approximations*. Springer, 1994.
21. R. NIEDERMEIER. *An Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.
22. I.W.M. POTHOF AND J. SCHUT. *Graph-Theoretic Approach to Identifiability in a Water Distribution Network*. Memorandum 1283, Universiteit Twente, Twente, The Netherlands, 1995.
23. E. PRIETO AND C. SLOPER. *Reducing to Independent Set Structure: The Case of  $k$ -Internal Spanning Tree*. Nordic Journal of Computing, Vol. 12, Issue 3, pp. 308-318, 2005.
24. S. SAURABH. *Exact Algorithms for Optimization and Parameterized Versions of Some Graph-Theoretic Problems*. PhD Thesis, Homi Bhabha National Institute, Mumbai, India, August 2007.