Noname manuscript No.
(will be inserted by the editor)

Directed Steiner Trees With Diffusion Costs*

Dimitri Watel · Marc-Antoine Weisser · Cédric Bentz · Dominique Barth

the date of receipt and acceptance should be inserted later

Abstract Given a directed arc-weighted graph G with n nodes, a root r and k terminals, the *Directed Steiner Tree problem* (DST) consists in finding a minimum-weight tree rooted at r and spanning all the terminals. If this problem has several applications in multicast routing in packet switching networks, the modeling is not adapted anymore in networks based upon the circuit switching principle in which some nodes, called *non diffusing nodes*, are unable to duplicate packets. We define a more general problem, namely the *Directed Steiner Tree with a Limited number of Diffusing nodes* (DSTLD), that enables us to model multicast in a network containing at most d diffusing nodes. We show that DSTLD is XP with respect to d, and use this result to build a $\left\lceil \frac{k-1}{d} \right\rceil$-approximation algorithm for DST that is XP in d. We deduce from that result a strong inapproximability property. In particular, we prove that, under the assumption that $NP \not\subseteq ZTIME[n^{\log^{O(1)} n}]$, there is no polynomial-time approximation algorithm for DSTLD with ratio $\Omega\left(\frac{k}{d}\right)$. We finally give an evaluation of performances of an exact algorithm dedicated to the case $d \leq 3$.

Keywords Approximation · Parameterized Complexity · Directed Steiner Tree · Diffusing node

* This paper is an extended version of [Watel et al., 2014].

Dimitri Watel · Marc-Antoine Weisser
SUPELEC System Sciences, Computer Science DPT., 91192 Gif Sur Yvette, France
E-mail: dimitri.watel@supelec.fr, marc-antoine.weisser@supelec.fr

Cédric Bentz
CEDRIC-CNAM 292 rue Saint-Martin F-75141 Paris CEDEX 03
E-mail: cedric.bentz@cnam.fr

Dimitri Watel · Dominique Barth
University of Versailles, 45 avenue des Etats-Unis, 78035, France
E-mail: dominique.barth@prism.uvsq.fr

1 Introduction

Given a directed arc-weighted graph with n nodes, the *Directed Steiner Tree problem* (DST) asks for a minimum-weight tree rooted at a specified node r (called the *root*) and spanning a specified set X of k nodes, called *terminals*. This problem, as well as its undirected counterpart, are known to have applications essentially in multicast routing where one wants to minimize the bandwidth consumption [Cheng and Du, 2001, Voß, 2006, Rugeli and Novak, 1995, Novak, 2001]. DST is used, instead of the undirected version [Karp, 1972], when assuming that the network is symmetric is not relevant anymore.

Although the undirected version can be approximated in polynomial time within a constant ratio [Kou et al., 1981, Robins and Zelikovsky, 2000], it was proved in [Karp, 1972] that DST is a generalization of the Set Cover problem and therefore is inapproximable within a ratio strictly better than $\ln(n)$ unless $NP \subseteq TIME[n^{O(\log\log n)}]$ [Feige, 1998]. It was later proved that, unless $NP \subseteq ZTIME[n^{\log^{O(1)} n}]$, there is no $\log^{2-\varepsilon}(k)-$approximation algorithm for any $\varepsilon > 0$ [Halperin and Krauthgamer, 2003].

The best known approximation ratio for DST is $O(k^{\varepsilon})$ for any $\varepsilon > 0$ [Charikar et al., 1998]. It uses the following result: computing a tree with fixed height l with minimum weight yields an $O(k^{\frac{1}{l}})$-approximation algorithm [Zelikovsky, 1997, Helvig et al., 2001]. Note that this latter approximation is neither polynomial nor XP in the parameter l. Both DST and its undirected counterpart are FPT with respect to the parameter k, as they can be solved by an exact algorithm that runs in time $O(3^k n+2^k(k+\log(n))n+n^2)$ and in space $O(2^k n)$ [Dreyfus and Wagner, 1971, Ding et al., 2007], but are W[2]-Hard with respect to the parameter "Optimal solution value" [Downey and Fellows, 1999, Jones et al., 2013].

Nevertheless, using DST to model multicast amounts to assuming that when a *branching node* (a node with at least 2 successors) of the tree receives a data, it is able to transmit it to its multiple successors. This is indeed the case in classical packet switching networks. However, previous works emphasize the fact that, in optical networks, this assumption does not hold anymore as most of the nodes, called *non diffusing nodes*, cannot copy any data, but can only send it to one of its successors. As a consequence, a non diffusing branching node has to receive p copies of the data to transmit it to p successors, and the weight of the arc entering such a node in a tree solution for the DST model must be paid p times. Fortunately, some routers, called *diffusing nodes*, can duplicate data and thus need to receive it only once. Examples of multicast trees with diffusing and non diffusing nodes are given in Figure 1. In the DST problem, it is implicitly assumed that every node is diffusing. The fact that this may not be the case, and therefore that this may induce additional constraints on the Steiner trees to be computed, was first considered in undirected wavelength division multiplexing networks [Malli et al., 1998]. The optimal placement of diffusing nodes for given multicast trees was then studied in [Lin and Wang, 2005]. Polynomial-time

solvability and approximability results have been established in some cases, for instance in undirected graphs where the diffusing nodes are already located [Du et al., 2005, Guo et al., 2005, Reinhard et al., 2009], or in case where a multicast tree is given [Reinhard et al., 2012].

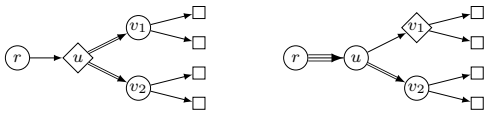

Fig. 1 Two examples of multicast trees with one diffusing node and seven non diffusing nodes. On the left side, u is the diffusing node, and r sends the data once to u, which then copies it twice to v_1 and v_2, so that those nodes can transmit it to the terminals. On the right side, v_1 is the diffusing node, and r has to send the data three times to u, as v_1 needs to receive the data once and v_2 needs to receive it twice.

The authors of the present paper previously studied a more constrained model, both in the directed and undirected cases, where no arc or edge could be used twice or more to send the same data: in other words, every branching node of the solution must be a diffusing node [Watel et al., 2013]. This problem is equivalent to finding a minimum-weight Steiner tree which uses a limited number of branching nodes [Watel et al., 2013, Gargano et al., 2002]. Finding a minimum-weight directed Steiner tree satisfying this constraint becomes a problem much harder to solve (exactly or approximately) than DST, even if both the number of authorized branching nodes and the number of terminals are fixed. This is due to families of instances where it is NP-Complete to decide whether there exists a feasible solution, regardless of its weight.

If no node is diffusing, including the root, then the solution cannot contain any branching node. This case (called the *Steiner Path problem*) is polynomially equivalent to the *Steiner Cycle problem* [Salazar-González, 2003, Steinová, 2010], where one wants to find a minimum-weight cycle containing the root and the terminals. As in the case where one looks for a directed Steiner tree using a limited number of branching nodes, it may be hard to decide whether a feasible solution exists for given instances of this problem. On the contrary, in DST, every node is assumed to be diffusing, and thus no arc needs to be used more than once. As optimal solutions for DST are trees with at most k leaves, there is no need to select more than $k-1$ diffusing nodes: the (at most) $k-1$ branching nodes of the tree.

In order to reflect the fact that not all nodes are diffusing, one first has to consider a new way to define the value of a feasible solution, previously introduced in [Reinhard et al., 2009, Reinhard et al., 2012]. The *load* of an arc in a solution is defined as the amount of data transiting through it, in order to transfer information from the root to terminals, and the *weight* of that arc is equal to its weight multiplied by its load. In the undirected model introduced in [Reinhard et al., 2009, Reinhard et al., 2012], only trees were considered as

feasible solutions, since it was assumed that, in these networks, optimal solutions were trees. In such a model, one can recursively define the load as 1 for an arc entering a diffusing node or a terminal (assuming that terminals have outdegree 0), and as the sum of the loads of its outgoing arcs for any other node. It turns out that this definition can be generalized to directed acyclic graphs, but cannot be applied to general digraphs (see Section 2).

In this paper, we also limit the number of diffusing nodes that can exist in the network under consideration by associating a fixed cost to the act of installing a device enabling diffusion at any given node, turning it into a diffusing node. The goal could then be either to minimize the total design cost of the network (which is equal to the cost of installing devices plus the weight of the computed Steiner tree), or to install devices while not exceeding a given budget (limiting the number of diffusing nodes), in order to minimize the design cost of the network (i.e., the weight of the computed Steiner tree). Both problems are actually polynomial-time equivalent in theory, but not necessarily with respect to approximation or fixed-parameter tractability. We propose to study the latter one, which appears to be closer to real-life considerations and to problems previously studied in the literature. We give an example in Figure 2.

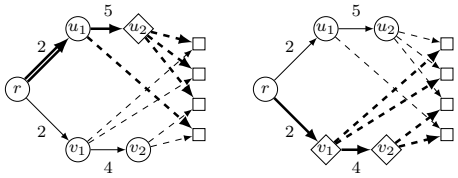

Fig. 2 Examples of optimal placement and diffusion. The weight of the dashed arcs is 0. On the left side, we can install at most one diffusing node; u_2 is chosen, and the weight of the computed solution is $2 * 2 + 5 = 9$. On the right side, we can install at most two diffusing nodes: v_1 and v_2 are chosen, and the weight of the computed solution is $2 + 4 = 6$.

Our results. We introduce and study the so-called *Directed Steiner Tree problem with a Limited number of Diffusing nodes* (DSTLD), refining several previous models that were defined in [Du et al., 2005, Guo et al., 2005, Reinhard et al., 2009, Reinhard et al., 2012], and somewhat overcoming difficulties raised by the models given in [Watel et al., 2013, Gargano et al., 2002, Salazar-González, 2003, Steinová, 2010]. For instance, in DSTLD, we do not impose the optimal solution to be a tree, as this seems to be a more accurate model for asymmetric network applications with diffusing routers (since, as illustrated in Figure 3, not imposing this constraint may lead to cheaper solutions). We prove this problem to be XP but W[2]-Hard with respect to the number of authorized diffusing routers d, and NP-Complete even if $k - d$ is fixed.

The core of our paper lies in two results related to approximation. The first one is a positive one. As already mentioned, computing the minimum-weight Steiner tree of fixed height l yields a $O(k^{\frac{1}{l}})$-approximation algorithm for DST [Zelikovsky, 1997, Helvig et al., 2001], but this algorithm is neither polynomial nor XP in the parameter l. Here, we are not interested in the height of the tree, but in the number of diffusing nodes it contains. We claim that one can transform any instance of DST to get an instance of DSTLD with $d < k$ diffusing nodes, solve that instance to get a solution T_d, and build from T_d a $\lceil \frac{k-1}{d} \rceil$-approximation for the DST instance. As DSTLD is XP in d, this approximation for DST is also XP in d. The second one is a negative one, as it is a strong inapproximability result for DSTLD deduced from the previous result. However, it should be noticed that both results are obtained by exhibiting links between the best approximation ratio for DST and the one for DSTLD, that could lead to further improvements in the future.

The paper is organized as follows. The next section provides an example where returning a graph containing a cycle yields a better solution than returning a tree. Then, in Section 3, we detail some notations that will be useful in this paper, and in Section 4 we properly define the new problem that we consider. Sections 5 and 6 respectively study the parameterized complexity and the approximability of this problem. Finally, Section 7 gives an evaluation of the efficiency of an exact algorithm dedicated to the case $d \leq 3$.

2 The optimal solution is not always a tree

The model developed by [Reinhard et al., 2009, Reinhard et al., 2012] assumes that, in the undirected case, the optimal routings are always trees. This property cannot be claimed for directed cases. Figure 3 illustrates a case where returning a graph containing a cycle is a better answer.

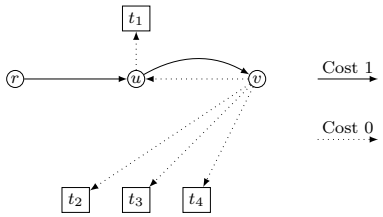

Fig. 3 In this example, the maximal number of allowed diffusing nodes is 1.

We assume that only one node can be selected as a diffusing node, including r. If r is selected, then it sends the data once per terminal. The arc (r, u) is used 4 times, and (u, v) 3 times, for a total weight of $4 * 1 + 3 * 1 = 7$. If u is diffusing, then the root sends the data to u, and u dispatches it to each terminal, for a total weight of $1 + 3 * 1 = 4$. Finally, if v is selected, the data is transmitted only once over the path $\{(r, u), (u, v)\}$, the node v is able to

send one data to t_1 using the $\{(v,u),(u,t_1)\}$ path. The number of data per arc is one for every arc, and the total weight is $1+1=2$. This optimal routing is not a tree but it is always possible to describe the routing as a tree. For example, this solution can be described as $\{(r,v),(v,t_1),(v,t_2),(v,t_3),(v,t_4)\}$, a tree where each arc describes a shortest path in the original graph.

3 Notations

Given a directed graph H, we define as $V(H)$ its vertex set, and as $A(H)$ its arc set. Moreover, given an instance of DST, we denote by n the number of vertices in this instance, by m the number of arcs, by X the set of terminals and by k the number of terminals. If we further want to impose a bound on the maximum number of diffusing nodes in the returned solution, we shall denote this bound by d.

Let u and v be two nodes in a directed graph G weighted over its arcs with function ω. We define as $P(u,v)$ the shortest path linking u and v in G, and as $\omega^{\triangleright}(u,v)$ the weight of this path. If $P(u,v)$ does not exist, then $\omega^{\triangleright}(u,v)=+\infty$. If multiple shortest paths exist, one is arbitrary chosen as $P(u,v)$. If G' is a subgraph of G, then $\omega(G')=\sum_{a\in A(G')}\omega(a)$, and similarly $\omega^{\triangleright}(G')=\sum_{(u,v)\in A(G')}\omega^{\triangleright}(u,v)$; notice that $\sum_{\substack{a\in\bigcup P(u,v),\\(u,v)\in A(G')}}\omega(a)\leq\omega^{\triangleright}(G')\leq\omega(G')$.

We assume without loss of generality that the root does not have any predecessor and has only one successor. We also assume the terminals to be leaves. If not, we first preprocess the graph to ensure those properties. For instance, if the root r has a predecessor or two successors, we simply add a new node r', define r' as the new root of the instance and add an arc (r',r) with weight 0.

Definition 1 Let $\mathcal{I}=(G=(V,A),r,X,\omega)$ be an instance of DST. Then the *shortest paths instance* $\mathcal{I}^{\triangleright}=(G^{\triangleright}=(V,V\times V),r,X,\omega^{\triangleright})$ defines the instance where $G^{\triangleright}$ is a complete graph weighted by the lengths $\omega^{\triangleright}$ of the shortest paths in G.

$\omega^{\triangleright}$ satisfies the triangle inequality. Since we assume that the root has no predecessor and that the terminals are all leaves, each arc entering r and each arc leaving a terminal has infinite weight in the shortest paths instance.

Definition 2 A *branching node* is a node with at least two successors.

4 The Directed Steiner Tree with Limited number of Diffusing nodes

In this section, we define our model and explain its relation with the optical network problem with diffusing nodes. The main remark is that a diffusing

node in a solution for $\mathcal{I}$ is in fact a branching node in a solution for $\mathcal{I}^{\triangleright}$. From Figure 3, an optimal solution to DSTLD is not necessarily a tree in the initial graph, but it is always a tree in the associated shortest paths instance.

Problem 1 Given a DST instance $\mathcal{I} = (G, r, X, \omega)$ and an integer $d \in [1; k-1]$, the Directed Steiner Tree problem with Limited number of Diffusing nodes (DSTLD) consists, in the shortest paths instance $\mathcal{I}^{\triangleright}$ associated to $\mathcal{I}$, in the search for a tree $T^{\triangleright}$ rooted at r, spanning X, containing at most d branching nodes and minimizing the weight $\omega^{\triangleright}(T^{\triangleright}) = \sum_{(u,v)\in A(T^{\triangleright})} \omega^{\triangleright}(u,v)$.

We do not consider the case $d = 0$ as each terminal is a leaf.

As, from Section 2, an optimal solution in G may not be a tree, we no longer refer to feasible solutions to DSTLD as Steiner trees in G but either as *diffusions in* G or as *Steiner trees in* $G^{\triangleright}$.

For instance, we give in Figure 4 the shortest paths instance $\mathcal{I}^{\triangleright}$ for the DSTLD instance $\mathcal{I}$ given as an example in Figure 3. In $\mathcal{I}$, we are searching for a diffusion with at most one diffusing node. As a consequence, we are searching for a tree $T^{\triangleright}$ in $\mathcal{I}^{\triangleright}$ with at most one branching node. The optimal solution is $T^{\triangleright} = \{(r,v),(v,t_1),(v,t_2),(v,t_3),(v,t_4)\}$, where v is the unique branching node.

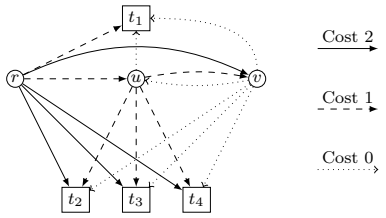

Fig. 4 Shortest paths instance $\mathcal{I}^{\triangleright}$ for the DSTLD instance $\mathcal{I}$ given in Figure 3. The arcs of weight $+\infty$ do not appear.

4.1 Link with DST

When $d \geq k-1$, the instance of DSTLD and the instance of DST described in Problem 1 are equivalent:

- From a feasible solution T for DST, one can build a feasible solution $T^{\triangleright}$ for DSTLD by replacing each arc (u,v) of T by the associated arc (u,v) in $G^{\triangleright}$. Since $G^{\triangleright}$ is weighted by lengths of shortest paths, we have $\omega^{\triangleright}(T^{\triangleright}) \leq \omega(T)$.
- Conversely, from a feasible solution $T^{\triangleright}$ for DSTLD, one can build a feasible solution T for DST by returning $\bigcup_{(u,v)\in A(T^{\triangleright})} P(u,v)$. The two solutions satisfy $\omega(T) \leq \omega^{\triangleright}(T^{\triangleright})$.

As a consequence, any $\alpha(k)-$approximation algorithm for DSTLD implies an $\alpha(k)-$approximation algorithm for DST. As, in this particular case, we have $d \geq k-1$, this result also holds for an $\alpha(d)-$approximation for DSTLD.

Theorem 1 *DSTLD is NP-Complete and any approximation algorithm for DSTLD with ratio $\alpha(d)$ or $\alpha(k)$ implies an approximation algorithm with ratio $\alpha(k)$ for DST.*

4.2 Application of DSTLD to multicast in an optical network

From a feasible solution of DSTLD, we can determine the diffusing nodes of the network and the load inside each arc needed to transmit the data from the root to all terminals. Let $T^{\triangleright}$ be a feasible solution of an instance of DSTLD.

- Each branching node in $T^{\triangleright}$ is a *diffusing node* in G.
- The *load* $l(a)$ of an arc a in G is the number of times it appears in path $P(u,v)$ for every $(u,v) \in A(T^{\triangleright})$.

We now suppose that the data is transmitted from the root to each terminal using $T^{\triangleright}$. It is copied at each diffusing node in the original graph. As a consequence, the data goes through each path $P(u,v)$ for $(u,v) \in A(T^{\triangleright})$ exactly once, thus through each arc a number of times equal to its load. Note that some diffusing nodes in G can be non branching nodes, and some branching nodes in G can be non diffusing. The weight of $T^{\triangleright}$ is, as defined in the previous model of [Reinhard et al., 2009, Reinhard et al., 2012], the sum, over all the arcs, of the weight of each arc multiplied by its load.

For example, in Figures 3 and 4, the optimal solution for DSTLD is the directed tree $T^{\triangleright} = \{(r,v),(v,t_1),(v,t_2),(v,t_3),(v,t_4)\}$ of the shortest paths instance $\mathcal{I}^{\triangleright}$. The weight of that tree is 2 and v is its unique branching node. In the original instance $\mathcal{I}$, we can notice that the corresponding shortest paths are $\{(r,u,v),(v,u,t_1),(v,t_2),(v,t_3),(v,t_4)\}$. As a consequence, v is chosen as a diffusing node, r sends a data to v through u, and v copies it to the four terminals: this is the solution described in Section 2.

Lemma 1 $\omega^{\triangleright}(T^{\triangleright}) = \sum_{a \in A(G)} l(a)\omega(a)$

Remark 1 The DSTLD model does not allow a non diffusing root to send the data more than once although the root has no such limitation in a real network. This permits us to simplify the proofs of this paper by reducing the number of different types of node: diffusing or not.

Remark 2 Contrary to the models in [Watel et al., 2013, Gargano et al., 2002, Salazar-González, 2003, Steinová, 2010], we can decide in polynomial time if an instance contains a feasible solution. Indeed, if the original graph contains a path from the root to every terminal, then we can return the star (with one branching node) in $\mathcal{I}^{\triangleright}$ that is centered at the root and contains all the terminals. On the contrary, if some terminal is unreachable from the root, then the instance has no solution of finite weight.

4.3 Application of DSTLD to multicast in an optical network where the diffusing nodes are already chosen

DSTLD assumes that every node in the network is able to diffuse if needed, but at most d branching nodes will actually diffuse in order to limit, for example, energy consumption or signal loss quality. On the contrary, we can assume, as in [Du et al., 2005, Guo et al., 2005, Reinhard et al., 2009], that the possible diffusing nodes D are already placed in the graph, for instance because the technology for diffusing routers is specific. We set d to $|D|$.

Note that any data sent from the root or a diffusing node must reach, along a shortest path, either another diffusing node, or a terminal. Consequently, we can, as in [Du et al., 2005, Guo et al., 2005], look for a solution in the shortest paths instance $\mathcal{I}_r^{\triangleright}$ restricted to $\{r\} \cup X \cup D$, instead of the complete instance $\mathcal{I}^{\triangleright}$.

Because $d = |D|$, no tree in $\mathcal{I}_r^{\triangleright}$ rooted at r and spanning X can have more than d branching nodes. As a consequence, an optimal multicast routing is then described by an optimal directed Steiner tree T^* in $\mathcal{I}_r^{\triangleright}$. Moreover, any α-approximation algorithm for DST gives an α-approximation algorithm for this problem.

5 Parameterized Complexity

In this section, we establish three parameterized complexity results over the DSTLD model. The two first ones claim that DSTLD parameterized by d belongs to the class XP, although it is W[2]-Hard. The last one studies DSTLD parameterized by $k - d$ and shows it is an NP-Complete problem.

In order to show the first result, we firstly prove that any tree with internal non branching nodes can be reduced to a smaller tree.

Lemma 2 *Let $T^{\triangleright}$ be a feasible solution of an instance $\mathcal{I}^{\triangleright}$ of DSTLD. A directed tree with weight at most the one of $T^{\triangleright}$, rooted at r, and containing only all the terminals X and the branching nodes B of $T^{\triangleright}$ can be obtained in polynomial time from $T^{\triangleright}$.*

Proof We delete all the cycles and leaves not in $X \cup B$. We get a directed tree rooted at r with lower weight, with less branching nodes, and where all the leaves are in $X \cup B$. Let E be the set $\{r\} \cup X \cup B$. We now replace each path with endpoints in E and internal nodes not in E by a single arc of $G^{\triangleright}$. As the weights satisfy the triangular inequality, the weight of the tree does not increase. The obtained tree is a feasible solution and contains only r, X and B.

□

By Lemma 2, one of the optimal solutions is such a tree in $G^{\triangleright}$. As any terminal in the original graph is a leaf, it is also a leaf in any optimal solution. Thus, we can reduce the search space to the trees with k leaves and at most d internal nodes.

Theorem 2 *DSTLD is XP when parameterized by d.*

Proof We prove there is an exact polynomial-time algorithm for DSTLD when d is fixed. Let $\mathcal{I}^\triangleright$ be an instance of DSTLD. Let $\kappa = (u_1, u_2, \ldots, u_j)$ be j distinct nodes of V, with $j \leq d$. We now search for a minimum-weight feasible solution $T^\triangleright_\kappa$ containing every node of κ and where each branching node is in κ. Obviously, if κ is exactly the set of branching nodes of an optimal solution T^* of $\mathcal{I}^\triangleright$, then $T^\triangleright_\kappa$ is also an optimal solution: by iterating over all possible sets κ, we thus return an optimal solution.

By Lemma 2, if $T^\triangleright_\kappa$ has a finite weight, then it can be searched for among all the trees rooted at r, containing only κ and X, and in which r has only one child if it is not in κ. We now point out that, if $r \in \kappa$, then the tree $T^\triangleright_\kappa$ is a spanning tree (rooted at r) of the shortest paths graph $G^\triangleright$ restricted to r, κ and X. Thus, we search for a minimum-weight spanning tree $T^\triangleright$ in that graph. Every branching node of $T^\triangleright$ is in κ: indeed, every terminal is a leaf in G, unless the weight of $T^\triangleright_\kappa$ is infinite, so $T^\triangleright$ does not use any terminal as a branching node.

If $r \notin \kappa$, then we search for $T^\triangleright_\kappa$ among all the spanning trees (rooted at r) of the shortest paths graph $G^\triangleright$ restricted to r, κ and X, where r has only one successor: there are d such trees. Every branching node of such a tree is in κ in this case too.

A minimum-weight directed spanning tree can be found in time $O((1+k+d)^2)$ [Tarjan, 1977], so our algorithm runs in time $O(dn^d(1+k+d)^2)$.

□

Theorem 3 *DSTLD is W[2]-Hard when parameterized by d.*

Proof We use a variant of the classic Directed Steiner Tree reduction from the Set Cover problem. Given a set of elements U, a set S of subsets of U and an integer N, the Set Cover problem is to find at most N sets of S covering all the elements in U. We consider neither the trivial instances where either one set covers all the elements or all the sets are needed in order to cover all the elements, nor the trivial instances where some elements from U cannot be covered. This problem is $W[2]$-complete with respect to N [Downey and Fellows, 1999].

We define a fixed-parameter reduction from this parameterized problem to the decision problem associated to DSTLD. Given an instance of the Set Cover problem, we construct an instance $\mathcal{I} = (G = (V, A), r, X, \omega, d, N')$ of the decision problem associated to DSTLD, and the corresponding shortest paths instance $\mathcal{I}^\triangleright$. We ask whether $\mathcal{I}^\triangleright$ contains a tree $T^\triangleright$ rooted at r, covering X, containing at most d branching nodes, and satisfying $\omega^\triangleright(T^\triangleright) \leq N'$. For each set in S we add a *set node* s in V. For each element in U, we add an *terminal element* in X. Finally we add a root r to V. We link r to each set node with an arc of weight 1, and link a set node to an terminal element with an arc of weight 0 if the associated element is in the associated set in the Set Cover instance. We set the parameter d to $N+1$ and the parameter N' to N. The shortest paths instance, in which we want to compute $T^\triangleright$, can be built by adding, for each terminal t, an arc from r to t of weight 1. For each pair of

vertices u, v such that there exists no path from u to v so far, we also add an arc of weight $+\infty$ from u to v. An example is shown in Figure 5 (where arcs of weight $+\infty$ do not appear).

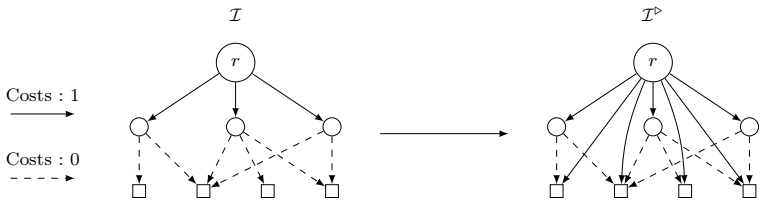

Fig. 5 Example of reduction from a Set Cover instance with $U = \{x_1, x_2, x_3, x_4\}$ and $S = \{\{x_1, x_2\}, \{x_2, x_3, x_4\}, \{x_2, x_4\}\}$. The instance $\mathcal{I}$ of DSTLD is on the left side and the associated shortest paths instance $\mathcal{I}^{\triangleright}$ is on the right side.

If there exists a cover $c \subset S$ with $|c| \leq N$, let $T^{\triangleright}$ be the tree of $\mathcal{I}^{\triangleright}$ rooted at r and using each set node associated with a set of c to cover X. As we do not consider the Set Cover instances where an optimal solution contains only one set, we have $|c| > 1$. Consequently, the root r is a branching node of $T^{\triangleright}$, which has $|c| + 1 \leq d$ branching nodes. The weight of $T^{\triangleright}$ is exactly $|c| \leq N = N'$. Thus, the instance $\mathcal{I}^{\triangleright}$ contains a feasible solution.

We now assume that there exists a feasible solution $T^{\triangleright}$ of $\mathcal{I}^{\triangleright}$. As we do not consider the Set Cover instances where an optimal solution contains only one set, the root r has at least two children in $T^{\triangleright}$. Let D_1 be the set nodes of $T^{\triangleright}$ that are branching nodes, let X_1 be the terminals covered by an arc going out of a set node of D_1, and let X_2 be the other terminals, covered either directly by an arc linking the root to it or by an arc going out of a non-branching set node. The weight of $T^{\triangleright}$ is $|D_1| + |X_2| \leq N' = N$. We build a cover c of X by selecting each set node of D_1 and an arbitrary set covering it for each element of X_2. Then $|c| \leq N$, and c is a feasible solution of the Set Cover instance.

This FPT reduction proves DSTLD to be W[2]-hard with respect to d. □

Theorem 4 *Even if $p = k - d$ is a fixed parameter, DSTLD is NP-Complete, and any $\alpha(d)$-approximation algorithm for DSTLD in this case yields an $\alpha(k')$-approximation algorithm for DST (where k' is the number of terminals in an arbitrary instance of DST).*

Sketch of proof. This theorem extends Theorem 1. Let p be a fixed integer. From an instance $\mathcal{I}$ for DST with k' terminals, one can build an instance $\mathcal{I}_d$ for DSTLD with k terminals satisfying $k - d = p$ by adding to $\mathcal{I}$ a star of weight 0 centered at r and having p leaves, all of them being terminals. Thus, $k' = k - p = d$. Obviously, from any feasible solution for $\mathcal{I}_d$, one can build in polynomial time a solution for $\mathcal{I}$ at least as good as the solution for $\mathcal{I}_d$, and vice versa (since a solution for $\mathcal{I}$ that has more than $k' - 1$ diffusing nodes can easily be transformed into a cheaper one that has at most $k' - 1$ diffusing nodes). Consequently, any $\alpha(d)$-approximation for $\mathcal{I}_d$ is an $\alpha(k')$-approximation for $\mathcal{I}$. □

6 Approximability

In this section we are interested in approximation results over DSTLD. The first subsection gives a k-approximation for DSTLD. The second subsection establishes an approximation ratio between an optimum solution for a DSTLD instance and an optimum solution for an associated DST instance. The last subsection shows an inapproximability result for DSTLD which is a consequence of the first result.

6.1 A k-approximation for DSTLD

Lemma 3 *The algorithm which returns the tree $T^\triangleright = \{(r,x), x \in X\}$ is a k-approximation.*

Proof This tree is a feasible solution as it contains one branching node (or 0 is there is only one terminal) and d is always greater than 1. Let $\hat{T^\triangleright}$ be an optimal solution of weight $\hat{\omega^\triangleright}$. For every terminal x, $\hat{T^\triangleright}$ contains a path from r to x, thus $\omega^\triangleright(r,x) \leq \hat{\omega^\triangleright}$. Consequently, $\omega^\triangleright(T^\triangleright) = \sum_{x \in X} \omega^\triangleright(r,x) \leq k \cdot \hat{\omega^\triangleright}$.

□

This k-approximation is similar to the shortest paths algorithm for DST that returns the union of all the shortest paths from r to the terminals. However, this one differs from the shortest paths algorithm, as the weight of the returned tree $T^\triangleright$ is the sum of the weights of the shortest paths, which is greater than the weight of the union of the shortest paths.

One can build better approximations by using the algorithm described in the proof of Theorem 2. Let $\mathcal{I}_d = (G, r, X, \omega, d)$ be an instance of DSTLD: any tree with less than d branching nodes is a feasible solution for $\mathcal{I}_d$. As a consequence, we could return an optimal solution of $\mathcal{I}_{d'} = (G, r, X, \omega, d')$, with $d' \leq d$. As DSTLD is XP with respect to d, this approximation is XP with respect to d'. Note that the algorithm described in Lemma 3 returns a feasible solution of $\mathcal{I}_1$. This solution may not be optimal, as an optimal solution can use a node different from the root as branching.

Remark 3 Note that the algorithm returning an optimal solution of $\mathcal{I}_{d'}$, for any fixed integer $d' \geq 1$, is also a k-approximation, and the ratio is tight (meaning there are instances for which this ratio is reached).

6.2 How DST can be approximated by DSTLD

Restricting the search space to trees with fixed height l gives a $O(k^{\frac{1}{l}})$-approximation for DST [Zelikovsky, 1997, Helvig et al., 2001]. However, this approximation is neither polynomial nor XP in l. We are interested here in reducing the number of authorized diffusing nodes instead of the height of the tree.

Any DST instance $\mathcal{I} = (G, r, X, \omega)$ can be transformed into a DSTLD instance $\mathcal{I}_d$ by adding a parameter $d \leq k-1$. We now prove that computing an optimal solution for $\mathcal{I}_d$ gives an approximated solution for $\mathcal{I}$. As DSTLD is XP with respect to the parameter d, it is possible to compute this solution efficiently for small values of d. We now assume that $k > 1$.

Let T^* be an optimal solution for $\mathcal{I}$, and let T_d^* be an optimal solution for $\mathcal{I}_d$.

Lemma 4 $\frac{\omega^{\triangleright}(T_d^*)}{\omega(T^*)} \leq \left\lceil \frac{k-1}{d} \right\rceil$.

Proof We will transform T^* into a feasible solution $T_d^{\triangleright}$ of $\mathcal{I}_d$, by replacing d subtrees of T^* by stars. We first build the equivalent tree T^* in $G^{\triangleright}$: each arc (u, v) of T^* is replaced by the associated arc (u, v) in $G^{\triangleright}$. As (u, v) is weighted in $G^{\triangleright}$ by the length $\omega^{\triangleright}(u, v)$ of a shortest path between u and v, its weight does not increase.

We define for a node u of T^* the subtree rooted at u as $T^*(u)$ and the set of terminals it reaches as $X(u)$. Let v be a node such that:

$$\begin{cases} |X(v)| \ \geq 1 + \left\lceil \frac{k-1}{d} \right\rceil \\ |X(w)| < 1 + \left\lceil \frac{k-1}{d} \right\rceil \end{cases} \text{for each successor } w \text{ of } v \text{ in } T^*.$$

If no node can satisfy those properties, then $|X(u)| < 1 + \left\lceil \frac{k-1}{d} \right\rceil$ for every node u. In that case, we choose v as the first branching node reached by r in T^* (or r itself if it is a branching node).

In $G^{\triangleright}$, $S(v)$ is the star containing the arc (v, t) for all $t \in X(v)$. The weight $\omega^{\triangleright}(S(v))$ of $S(v)$ is at most $\sum_{t \in X(v)} \sum_{a \in P_{T^*}(v,t)} \omega^{\triangleright}(a)$, where $P_{T^*}(v, t)$ is the path from v to t in T^*. Moreover, for an arc a in $T^*(v)$, the number of distinct paths $P_{T^*}(v, t)$ containing a cannot be more than $\left\lceil \frac{k-1}{d} \right\rceil$. Indeed, each successor of v reaches at most $\left\lceil \frac{k-1}{d} \right\rceil$ terminals. So, we have:

$$\omega^{\triangleright}(S(v)) \leq \left\lceil \frac{k-1}{d} \right\rceil \cdot \omega^{\triangleright}(T^*(v)) \tag{1}$$

We temporarily replace in T^* the subtree $T^*(v)$ by a terminal $t^{(1)}$. The obtained tree $T^{(1)}$ contains at most $k - |X(v)| + 1 \leq k - \left\lceil \frac{k-1}{d} \right\rceil \leq k - \frac{k-1}{d}$ terminals. We repeat this operation and build the trees $T^{*(2)}$, $T^{*(3)}$, ... until it remains only one terminal. As each operation removes at least $\frac{k-1}{d}$ terminals, this operation is repeated at most d times.

We now expand all the stars in reverse order. The resulting graph is a tree $T_d^{\triangleright}$ in $G^{\triangleright}$ containing all the terminals in X and at most d branching nodes (the roots of the stars). By equation (1), $\omega^{\triangleright}(T_d^{\triangleright}) \leq \left\lceil \frac{k-1}{d} \right\rceil \omega^{\triangleright}(T^*) \leq \left\lceil \frac{k-1}{d} \right\rceil \omega(T^*)$. □

Remark 4 There is no $\left\lceil \frac{d_2}{d_1} \right\rceil$-approximation ratio between two optimal solutions of $\mathcal{I}_{d_1}$ and $\mathcal{I}_{d_2}$, because the Steiner tree can have less than $k-1$ branching nodes. The only known result is $\omega(T^*) = \omega^{\triangleright}(T_{k-1}^*) \leq \cdots \leq \omega^{\triangleright}(T_2^*) \leq \omega^{\triangleright}(T_1^*) \leq k \cdot \omega(T^*)$.

Theorem 5 *From any $\alpha(d,k)$-approximation algorithm for DSTLD, one can obtain, for any $d \in [\![1; k-1]\!]$, an approximation algorithm of ratio $\alpha(d,k) \cdot \left\lceil \frac{k-1}{d} \right\rceil$ for DST.*

Proof If we compute our α-approximation algorithm over $\mathcal{I}_d$, we get in $G^{\triangleright}$ a tree $T_d^{\triangleright}$ satisfying $\omega^{\triangleright}(T_d^{\triangleright}) \leq \alpha(d,k) \cdot \omega^{\triangleright}(T_d^*)$. The tree $T = \bigcup_{(u,v) \in A(T_d^{\triangleright})} P(u,v)$ is a feasible solution for $\mathcal{I}$ with weight at most $\omega^{\triangleright}(T_d^{\triangleright})$.

By Lemma 4, $\omega(T) \leq \alpha(d,k) \left\lceil \frac{k-1}{d} \right\rceil \cdot \omega(T^*)$.

□

With this technique, one can either choose a fixed parameter d to get a polynomial approximation with ratio $\left\lceil \frac{k-1}{d} \right\rceil$ for DST (to be more specific, this approximation is XP with respect to d, from Theorem 2), or choose a variable parameter $d = f(k)$ for some function f (for instance $d = \lceil \log(k) \rceil$), and compute a polynomial approximation with ratio $\alpha(d,k)$ for DSTLD to get a polynomial approximation with ratio $\alpha(f(k),k) \left\lceil \frac{k-1}{f(k)} \right\rceil$ for DST.

6.3 Inapproximability result for DSTLD

Using the previous result and the current best known lower bound for the best approximability ratio for DST, we can prove the following corollary, extending the inapproximability result given in [Watel et al., 2014]. The key idea of this corollary is to use a variable parameter d such that $\alpha(d,k) \left\lceil \frac{k-1}{d} \right\rceil$ is always smaller than a lower bound on the best possible approximation ratio for DST.

Corollary 1 *Unless $NP \subseteq ZTIME(t)$, for any $\varepsilon > 0$ and any $c > 0$, there is no $\alpha(d,k)$-approximation algorithm for DSTLD if there is some function $f : \mathbb{N} \to \mathbb{N}$ satisfying $f(k) \in [\![1; k-1]\!]$ and $\alpha(f(k),k) \cdot \left\lceil \frac{k-1}{f(k)} \right\rceil \leq c \cdot \log^{2-\varepsilon}(k)$, except for a finite number of values of k.*

Proof Let K be the set of integers k for which $\alpha(f(k),k) \cdot \left\lceil \frac{k-1}{f(k)} \right\rceil > c \cdot \log^{2-\varepsilon}(k)$.

If such an approximation exists, then, by Theorem 5, an $\alpha(f(k),k) \cdot \left\lceil \frac{k-1}{f(k)} \right\rceil$-approximation for DST would exist. As a consequence, we would be able to design a $c \cdot \log^{2-\varepsilon}(k)$-approximation for DST, except for $k \in K$. Furthermore, $c \cdot \log^{2-\varepsilon}(k) \leq \log^{2-(\varepsilon/2)}(k)$, except for a finite number of values of k. Let K' be the finite set of values of k for which this inequality does not hold. For all $\varepsilon > 0$, there is no $\log^{2-(\varepsilon/2)}(k)$-approximation algorithm for DST, unless $NP \subseteq ZTIME(t)$, even if we do not consider the DST instances such that $k \in K \cup K'$ [Halperin and Krauthgamer, 2003].

Consequently, there is no $\alpha(d,k)$-approximation algorithm for DSTLD.

□

Using this corollary, we can now exhibit the following inapproximability result:

Corollary 2 *Unless* $NP \subseteq ZTIME(t)$*, for any* $\varepsilon > 0$ *and any* $c > 0$*, there is no* $\alpha(d,k) = c \cdot \lceil \frac{k-1}{d} \rceil \cdot \log^{2-\varepsilon}(k)$*-approximation algorithm for DSTLD.*

Proof Let $f(k) = \frac{k-1}{2}$: we have $\alpha(f(k),k) \cdot \left\lceil \frac{k-1}{f(k)} \right\rceil = c \cdot \left\lceil \frac{k-1}{f(k)} \right\rceil^2 \cdot \log^{2-\varepsilon}(k) \leq 4c \cdot \log^{2-\varepsilon}(k)$. By Corollary 1, this result is proven.

□

We can similarly prove the following stronger result:

Corollary 3 *Unless* $NP \subseteq ZTIME(t)$*, for any* $\varepsilon > 0$*, any* $c_1 > 0$ *and any* $c_2 > 1$*, there is no* $\alpha(d,k)$*-approximation algorithm for DSTLD, where* $\alpha(d,k) = c_1 \lceil \frac{k-1}{d} \rceil \cdot \log^{2-\varepsilon}(k)$ *if* $d = \lceil \frac{k-1}{c_2} \rceil$ *and* $\alpha(d,k) = +\infty$ *otherwise.*

Note that Corollary 1 does not prove, for example, that there is no $\left\lceil \frac{k-1}{d^{\varepsilon'}} \right\rceil$-approximation algorithm for DSTLD if $\varepsilon' < 1$. Indeed, any function $f(k)$ such that $\left\lceil \frac{k-1}{f(k)^{\varepsilon'}} \right\rceil \cdot \left\lceil \frac{k-1}{f(k)} \right\rceil \leq c \cdot \log^{2-\varepsilon}(k)$ satisfies $f(k) \leq k$ for any sufficiently big k. However, in the corollary, $f(k)$ is lower than k.

It is not clear how should the approximation ratio vary with d. Is it an increasing function or a deacreasing function? The fact that the problem is XP with d and NP-Complete with $k - d$ could be an argument showing that the problem is harder to solve when d increases.

7 Evaluation of performances

We give in this section an evaluation of how the weight of a minimum-weight directed Steiner tree (with $k-1$ allowed diffusing nodes) is affected when we use at most one, two or three diffusing nodes. We study the algorithm XP with respect to d described in Theorem 2. For the DST problem, the trivial shortest paths algorithm (a k-approximation returning the union of the shortest paths from the root to the terminals) is a very good algorithm in practice (fast and efficient regardless of its approximation ratio), often used to solve DST. We want to determine whether a similar trivial algorithm is as practical for DSTLD, as it can be used to approximate DSTLD and DST (as explained in the previous section).

The algorithm was implemented in Java 1.7.0_025 and the experiments were run on Ubuntu 12.10 with Intel Core 3.10 GHz processors.

7.1 Description of the test set

A test set named *SteinLib* aggregates most of the generated Undirected Steiner Problem instances that can be found in the literature [Koch et al., 2001]. Each instance is given with the weight of an optimal solution. To run the experiments on directed graphs, we transformed the SteinLib instances into bidirected instances where each edge was replaced by two opposite arcs with the same

weight and, without loss of generality, chose any terminal as the root. Indeed, as the instances are bidirected, the weight of an optimal solution for DST or DSTLD is the same, regardless of the terminal we choose to be the root. We then processed each instance to ensure that each terminal was a leaf and that the root had only one successor. We finally built the shortest paths instance using the Roy-Warshall-Floyd algorithm to compute the shortest paths between all pairs of vertices in time $O(n^3)$, where n is the number of vertices [Floyd, 1962].

The test sets were generated from parts of the following SteinLib groups: WRP3, I080 and I160, ES10FST to ES100FST, B and C. Some instances of these groups (c11, and wrp3-55 to wrp3-99) and of other groups were not considered, because of the time or the memory needed to compute the associated shortest paths instance. The size of the test set is 422.

The number of nodes in each instance is, for the most part (350 instances), less than 200. There are only four instances with more than 1000 nodes. The maximum number of nodes is 1414 for the instance wrp3-45.

7.2 Quality of the returned solutions

Considering an instance with an optimal weight equal to ω^* when $k-1$ diffusing nodes are allowed, we computed the optimal weights ω_1, ω_2 and ω_3 of the instance when respectively one, two and three diffusing nodes were allowed. The first weight is the weight of an optimal directed Steiner tree and was given with the SteinLib instance. The three other weights were obtained by adapting the algorithm of Theorem 2 to those simple cases: this gives a linear-time algorithm for $d = 1$, a quadratic-time algorithm for $d = 2$ and a cubic-time algorithm for $d = 3$.

We studied the relative errors $\delta_1 = \frac{\omega_1 - \omega^*}{\omega^*}$, $\delta_2 = \frac{\omega_2 - \omega^*}{\omega^*}$ and $\delta_3 = \frac{\omega_3 - \omega^*}{\omega^*}$. To be more specific, we computed the mean, the standard deviation and the number of instances for which the error is null. The results are given in Table 1.

	δ_1			δ_2			δ_3		
SteinLib groups	M	SD	$= 0$	M	SD	$= 0$	M	SD	$= 0$
B and C	108	71	0	78	61	0	62	55	3
I080 and I160	23	20	35	13	14	54	8	11	74
ES10FST to ES100FST	432	196	0	274	129	0	198	99	0

Table 1 This table contains, among all the instances built from the given groups and for each relative error δ_i, the mean M in percent, the standard deviation SD of δ_i in percent and the number of instances for which δ_i is null.

The results for the WRP3 group do not appear in this table, because the relative error was always less than 10^{-3} but never null. This may be due to the fact that each arc of each instance is weighted with a very large number: the

weight of an optimal solution is high, and the weight of every feasible solution is close to the optimal weight.

We recall that we preprocessed every instance such that the terminals are leaves. Let $\mathcal{I} = (G, r, X, \omega)$ be a DST instance with k terminals. Let Δ be the maximum outdegree of the nodes of G. Finally, let T be a feasible solution of $\mathcal{I}$ with b branching nodes. The number of leaves of T is at most $\Delta \cdot b - (b-1)$ (this can easily be proved by induction on b). As all the terminals are leaves, $k \leq \Delta \cdot b - (b-1)$, thus $b \geq \frac{k-1}{\Delta-1}$. Let $\mathcal{I}_d = (G, r, X, \omega, d)$ be a DSTLD instance such that $d < b$. The b branching nodes of T (seen as a feasible solution of $\mathcal{I}_d$) cannot all be diffusing nodes in $\mathcal{I}_d$: some arcs are necessarily used more than once, and the weight of an optimal solution of $\mathcal{I}_d$ is greater than the weight of an optimal solution of $\mathcal{I}$. Intuitively, the greater is the gap between d and $\frac{k-1}{\Delta-1}$, the greater is the gap between the two optimal weights.

It appears that for the groups I080 and I160, $\frac{k-1}{\Delta-1} \leq 4$. For B, $\frac{k-1}{\Delta-1} \leq 3$ except for four instances (b09, b12, b15 and b18 for which the values are respectively 7 , 5, 8 and 6). For C, $\frac{k-1}{\Delta-1} \leq 9$, except for five instances (c04, c05, c09, c10, c15 for which the values are respectively 12, 27, 12, 22, 13). Finally, for WRP3, $\frac{k-1}{\Delta-1} \leq 5$, except for five instances (wrp3-29, wrp3-36, wrp3-41, wrp3-43 and wrp3-52 for which the values are respectively 6, 8, 9, 10, 7). On the contrary, for the groups ES30FST to ES100FST, the degree is small and the number of terminals is high. $\frac{k-1}{\Delta-1}$ is often constant for all the instances in the same group. The values vary from 9 (for ES30FST) to 32 (for ES100FST). Consequently, it appears there is a correlation between $\frac{k-1}{\Delta-1}$ and the results in Table 1. This strengthens our intuition.

Note that, for each group, the mean of δ_3 is half the mean of δ_1. It appears we can benefit from looking for solutions with more than one diffusing nodes.

Nevertheless, δ_1, δ_2 and δ_3 are high. This means that the deletion of a large number of diffusing nodes is costly to a network and that it seems better to allow more than 3 diffusing nodes, even for small networks.

7.3 Evaluation of the computation time

We also registered, for each instance, the time in milliseconds required to compute each solution. This time does not include the time needed to compute the shortest paths instance. We grouped the instances in categories depending on their number of nodes. The results, for each category, are given in Table 2.

The high values of the standard deviations are due to the small number of instances in each category and the fact that each category depends only on the number of nodes, whereas the computation time also depends on the number of arcs and terminals.

According to Table 2, and considering a real-time application where a network has to satisfy an input multicast request, one can only search for a solution with one diffusing node as only this algorithm takes less than one second to return an optimal solution. On the contrary, the time needed to compute an optimal solution with at most 3 diffusing nodes remains acceptable for any

	Size	$d = 1$		$d = 2$		$d = 3$	
		M	SD	M	SD	M	SD
$n \leq 100$	178	< 1	1.36	18	20	483	477
$100 < n \leq 200$	172	4	5	264	274	$13{\cdot}10^3$	$14{\cdot}10^3$
$200 < n \leq 300$	32	13	10	1341	727	$98{\cdot}10^3$	$59{\cdot}10^3$
$300 < n \leq 400$	6	13	12	1965	1735	$194{\cdot}10^3$	$176{\cdot}10^3$
$400 < n \leq 500$	23	94	196	$8{\cdot}10^3$	$8{\cdot}10^3$	$1516 \cdot 10^3$	$1618 \cdot 10^3$
$n > 500$	11	61	32	$24 \cdot 10^3$	$18 \cdot 10^3$	$9719{\cdot}10^3$	$10916 \cdot 10^3$

Table 2 This table contains, in milliseconds, the mean M and the standard deviation SD of the time required to compute ω_1, ω_2 and ω_3 for each category of instance. The second column gives the number of instances in each category.

network provisioning application, where the purpose is to install 3 or more diffusing nodes in a network, in such a way that the weight of each multicast request which would be satisfied using at most 3 diffusing nodes is minimized.

Considering how the computation time grows with the number of allowed diffusing nodes, this algorithm seems unlikely to be used when $d > 3$. Consequently, it seems necessary to find an approximation algorithm or a heuristic algorithm to compute feasible solutions allowed to use more than 3 diffusing nodes.

8 Conclusion and perspectives

We have proposed a generalization of the Directed Steiner Tree problem in order to model multicast routing in networks containing at most d diffusing nodes. We have proved that this problem is XP in d by providing an algorithm to solve it. We have used this algorithm to design an approximation algorithm for the Directed Steiner Tree problem which runs in time XP in d. However, we have also shown a strong inapproximability result for DSTLD.

Even so, the only polynomial-time approximation algorithm that was considered for DSTLD has a ratio depending only on k. According to the evaluation of performance, considering the initial application (multicast requests in optical networks), the exact algorithm XP in d we provide for DSTLD cannot be used in practice with more than $d > 3$ diffusing nodes, due to its high computational time, whereas a better algorithm (providing exact or approximate solutions) would be interesting, as allowing at most 3 diffusing nodes in a network is penalizing. Thus, it seems necessary to build new approximation algorithms for DSTLD with better ratios, depending on k and d.

References

[Charikar et al., 1998] Charikar, M., Chekuri, C., Cheung, T., and Dai, Z. (1998). Approximation algorithms for directed Steiner problems. *Proc. SODA*, pages 192–200.

[Cheng and Du, 2001] Cheng, X. and Du, D.-z. (2001). *Steiner trees in industry*, volume 11. Kluwer.

[Ding et al., 2007] Ding, B., Yu, J. X., Wang, S., and Qin, L. (2007). Finding top-k min-cost connected trees in databases. *ICDE*.

[Downey and Fellows, 1999] Downey, R. and Fellows, M. (1999). *Parameterized complexity*. Springer, series: monographs in computer science edition.

[Dreyfus and Wagner, 1971] Dreyfus, S. E. and Wagner, R. A. (1971). The Steiner problem in graphs. *Networks*, 1(3):195–207.

[Du et al., 2005] Du, H., Jia, X., Wang, F., Thai, M. Y., and Li, Y. (2005). A Note on Optical Network with Nonsplitting Nodes. *JCO*, 10:199–202.

[Feige, 1998] Feige, U. (1998). A threshold of ln n for approximating Set Cover. *JACM*, 45(4):634–652.

[Floyd, 1962] Floyd, R. (1962). Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345.

[Gargano et al., 2002] Gargano, L., Hell, P., Stacho, L., and Vaccaro, U. (2002). Spanning trees with bounded number of branch vertices. *Proc. ICALP*, pages 355–365.

[Guo et al., 2005] Guo, L., Wu, W., Wang, F., and Thai, M. (2005). An approximation for minimum multicast route in optical networks with nonsplitting nodes. *Journal of Combinatorial Optimization*, 10(4):391–394.

[Halperin and Krauthgamer, 2003] Halperin, E. and Krauthgamer, R. (2003). Polylogarithmic inapproximability. In *Proc. STOC, ACM*, pages 585–594. ACM.

[Helvig et al., 2001] Helvig, C. S., Robins, G., and Zelikovsky, A. (2001). An improved approximation scheme for the Group Steiner Problem. *Networks*, 37(1):8–20.

[Jones et al., 2013] Jones, M., Lokshtanov, D., Ramanujan, M., Saurabh, S., and Suchy, O. (2013). Parameterized complexity of directed Steiner tree on sparse graphs. *Proc. ESA 2013*, pages 671–682.

[Karp, 1972] Karp, R. (1972). Reducibility among combinatorial problems. In *Complexity of Computer Computations, The IBM Research Symposia Series*, pages 85–103. Springer.

[Koch et al., 2001] Koch, T., Martin, A., and Voß, S. (2001). SteinLib: An updated library on Steiner tree problems in graphs. In *Combinatorial Optimization*, volume 11, pages 285–325. Springer.

[Kou et al., 1981] Kou, L., Markowsky, G., and Berman, L. (1981). A fast algorithm for Steiner trees. *Acta informatica*, 15:141–145.

[Lin and Wang, 2005] Lin, H.-c. and Wang, S.-w. (2005). Splitter Placement in All-Optical WDM Networks. *Global Telecommunications Conference*.

[Malli et al., 1998] Malli, R., Zhang, X., and Qiao, C. (1998). Benefit of Multicasting in All-Optical Networks. *SPIE Proc. Conf. All-Optical Networking*.

[Novak, 2001] Novak, R. (2001). A note on distributed multicast routing in point-to-point networks. *Computers & Operations Research*, 28(12):1149–1164.

[Reinhard et al., 2012] Reinhard, V., Cohen, J., Tomasik, J., Barth, D., and Weisser, M.-A. (2012). Optimal configuration of an optical network providing predefined multicast transmissions. *Comput. Netw.*, 56(8):2097–2106.

[Reinhard et al., 2009] Reinhard, V., Tomasik, J., Barth, D., and Weisser, M.-A. (2009). Bandwidth Optimization for Multicast Transmissions in Virtual Circuit Networks. *IFIP Networking*, pages 859–870.

[Robins and Zelikovsky, 2000] Robins, G. and Zelikovsky, A. (2000). Improved Steiner tree approximation in graphs. In *Proc. SODA*, pages 770–779.

[Rugeli and Novak, 1995] Rugeli, J. and Novak, R. (1995). Steiner tree algorithms for multicast protocols. *Manuscript*.

[Salazar-González, 2003] Salazar-González, J.-J. (2003). The Steiner cycle polytope. *European Journal of Operational Research*, 147(3):671–679.

[Steinová, 2010] Steinová, M. (2010). Approximability of the Minimum Steiner Cycle Problem. *Computing and Informatics*, 29:1349–1357.

[Tarjan, 1977] Tarjan, R. (1977). Finding optimum branchings. *Networks*, 7(1):25–35.

[Voß, 2006] Voß, S. (2006). Steiner tree problems in telecommunications. In *Handbook of optimization in telecommunications*, pages 459–492. Springer.

[Watel et al., 2013] Watel, D., Weisser, M., Bentz, C., and Barth, D. (2013). Steiner Problems with Limited Number of Branching Nodes. *Proc. SIROCCO*, pages 310–321.

[Watel et al., 2014] Watel, D., Weisser, M., Bentz, C., and Barth, D. (2014). Directed Steiner Tree with Branching Constraint. *Proc. COCOON*, pages 263–275.

[Zelikovsky, 1997] Zelikovsky, A. (1997). A series of approximation algorithms for the acyclic directed Steiner tree problem. *Algorithmica*, 18(1):99–110.