ELSEVIER

Discrete Applied Mathematics 113 (2001) 73–85

DISCRETE APPLIED MATHEMATICS

Reload cost problems: minimum diameter spanning tree ☆

Hans-Christoph Wirth[a,*], Jan Steffan[b]

[a] *Department of Computer Science, University of Würzburg, Am Hubland, 97074 Würzburg, Germany*
[b] *Darmstadt University of Technology, IT Transfer Office, Wilhelminenstr. 7, 64283 Darmstadt, Germany*

Abstract

We examine a network design problem under the *reload cost* model. Given an undirected edge colored graph, reload costs on a path arise at a node where the path uses consecutive edges of different colors. We consider the problem of finding a spanning tree of minimum diameter with respect to the reload costs. We present lower bounds for the approximability even on graphs with maximum degree 5. On the other hand we provide an exact algorithm for graphs of maximum degree 3. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Transportation problems; Network design; Diameter; Spanning tree; Node weighted graphs

1. Introduction and related work

Network design problems are graph theoretic optimization problems which have a wide area of applications. We consider a scenario where a bunch of different providers runs a subnetwork each. The transportation costs inside of each subnetwork are negligible. Costs arise only at points where the underlying provider changes.

This approach can be used to model a cargo transportation network which uses different means of transportation. Here usually changing the carrier involves cost and time expensive unloading and reloading of the goods. Another application is modeling data transmission costs arising in large communication networks. The cost and time needed for data conversion at interchange points between incompatible subnetworks usually dominate the costs arising by routing the information packets within each of the subnetworks.

☆ Supported by Deutsche Forschungsgemeinschaft (DFG), Grant NO 88/15-3.

* Corresponding author.

E-mail addresses: wirth@informatik.uni-wuerzburg.de (H.-C. Wirth), steffan@ito.tu-darmstadt.de (J. Steffan).

0166-218X/01/$ - see front matter
PII: S0166-218X(00)00392-9

We model the scenario by an edge colored graph. Edges of the same color belong to the same subnetwork. In our reload cost model, costs arise at each node, depending on the pair of colors of the edges used by the walk through that node. For practical applications it is useful to assume that the reload costs satisfy the triangle inequality, since otherwise one could save costs by performing more than one reload job at the same node.

The problem investigated in this paper, DIAMETER-TREE, is to search for a spanning tree of minimum diameter with respect to reload costs. This is motivated by the fact that the diameter is an upper bound on the reload costs between an arbitrary pair of nodes.

For general reload cost functions, we can show that DIAMETER-TREE is not approximable at all even if restricted to graphs of maximum node degree 5. If the reload cost function satisfies the triangle inequality, we give a logarithmic lower bound on the approximation factor on general graphs and a lower bound of 3 on graphs with maximum degree 5. On the other hand, DIAMETER-TREE can be solved exactly when restricted to graphs of maximum degree 3. A preliminary version of this paper appeared in [8].

This problem is related to the minimum label spanning tree problem, which was introduced by Chang and Leu [1]. Here the goal is to find a spanning tree which uses as least as possible many different colors. In [6], the authors give both a logarithmic approximation algorithm and a logarithmic lower bound for the minimum label spanning tree problem.

The minimum diameter spanning tree problem on graphs with nonnegative edge lengths is equivalent to the *absolute* 1-*center problem* [5] and can be solved in time $O(|E||V| + |V|^2 \log |V|)$ [4]. A generalization on graphs with edge lengths and costs is the NP-hard *minimum diameter problem* [7], where the goal is to find a spanning subgraph of minimum diameter and total cost constrained by a given budget.

The paper is organized as follows: Section 2 contains the definition of the problem. Section 3 presents hardness results, in particular for graphs of bounded degree 5. Section 4 presents an exact algorithm for graphs of degree 3.

2. Preliminaries and problem formulation

Definition 1 (*Graph with reload costs*)**.** Let $G = (V,E)$ be an undirected graph with parallel edges allowed. Let $\chi : E \to X$ be a mapping from the set of edges to a set X of colors. A function $c\colon X^2 \to \mathbb{N}_0$ is called a *reload cost function*, if for all pairs $x_1, x_2 \in X$

(1) $c(x_1,x_2) = c(x_2,x_1)$,
(2) $x_1 = x_2 \Rightarrow c(x_1,x_2) = 0$,
(3) $x_1 \neq x_2 \Rightarrow c(x_1,x_2) > 0$.

If additionally for all $e_1, e_2, e_3 \in E$ which are incident in one single node
(4) $c(\chi(e_1),\chi(e_3)) \leqslant c(\chi(e_1),\chi(e_2)) + c(\chi(e_2),\chi(e_3))$, then the reload cost function is said to *satisfy the triangle inequality*.

We use the shorter notation $c(e_1,e_2) := c(\chi(e_1),\chi(e_2))$ throughout the paper.

Reload costs on a path arise at a node where two consecutive edges are of different colors. Consequently, a path of one edge only has reload costs zero. For a path $p = (e_1, e_2, \ldots, e_k)$ consisting of $k > 1$ edges, we have reload costs

$$c(p) := \sum_{i=1}^{k-1} c(e_i, e_{i+1})$$

These costs induce a distance function on the graph in a natural way:

Definition 2 (*Reload cost distance*). Let $G = (V, E, \chi, c)$ be a graph with reload costs. Then, the *induced reload cost distance function* is given by

$$dist_G^c(v, w) := \min\{c(p) \mid p \text{ is a path from } v \text{ to } w \text{ in } G\}.$$

In contrast, for an edge length function $l : E \to \mathbb{N}_0$, the *length* of a path p is given by $l(p) := \sum_{i=1}^{k} l(e_i)$, and the *induced length distance function* is consequently $dist_G^l(v, w) := \min\{l(p) \mid p \text{ is a path from } v \text{ to } w \text{ in } G\}$.

We are now ready to define the problem under study.

Definition 3 (*Problem diameter-tree*). An instance of Diameter-Tree is given by a graph $G = (V, E, \chi, c)$ with reload costs. The goal is to find a spanning tree $T \subseteq E$ of the graph, such that the diameter with respect to the reload costs, i.e.,

$$diam^c(T) := \max_{v,w \in V} dist_T^c(v, w),$$

is minimized among all spanning trees.

By Δ-Diatree-Tree we denote the problem where the reload cost function satisfies the triangle inequality.

A straightforward idea to deal with reload cost problems would be to transform the graph to its *line graph* and map the reload costs to edge lengths in the line graph. (The line graph of a graph $G = (V, E)$ has node set E and an edge (e_1, e_2) if and only if e_1 and e_2 are adjacent edges in G.) Unfortunately, there is no easy relationship between a spanning tree of a graph and a spanning tree of its line graph which can be exploited to make this approach work out of the box.

Let Π be a minimization problem. A polynomial running time algorithm A is called an *α-approximation algorithm* for Π, if for each instance π of Π with optimal solution $\mathrm{OPT}(\pi)$, the solution $A(\pi)$ produced by the algorithm satisfies $A(\pi) \leqslant \alpha\,\mathrm{OPT}(\pi)$. A problem Π is called *not approximable within* α, if there is no α-approximation algorithm unless $\mathrm{P} = \mathrm{NP}$ or $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\mathrm{O}(\log\log n)})$.

3. Hardness results

In this section we show that Diameter-Tree and even Δ-Diameter-Tree is NP-hard and we provide inapproximability results. We will show that the hardness results extend

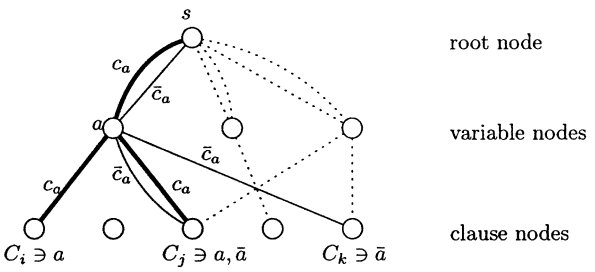

Fig. 1. Reduction from 3-SAT to Δ-DIAMETER-TREE used in the proof of Theorem 4.

to graphs where the node degree is bounded by 5. In contrast, in Section 4 we will give an exact algorithm for graphs with maximum degree 3.

Theorem 4. *Unless* P = NP, DIAMETER-TREE *is not approximable within any factor* $f(n)$ *on a graph with* n *nodes, even when restricted to graphs of maximum degree* 5. *Here,* f *is any polynomial time computable function.*

Proof. We perform a reduction from 3-SAT [3, Problem LO2]. An instance π of 3-SAT is given by a set $A = \{a_1, a_2, \ldots, a_n\}$ of variables and a set $C = \{C_1, \ldots, C_k\}$ of clauses over A. Each clause is of the form $C_i = \{l_{i1}, l_{i2}, l_{i3}\}$, where l_{ij} is a literal, i.e., a variable a or its negation $\bar{a}$. The goal is to find a truth assignment satisfying all clauses.

We now construct an instance π' of DIAMETER-TREE. The construction of the graph $G = (V, E)$ is shown in Fig. 1. Set $V := \{s\} \cup A \cup C$. For each variable $a \in A$, we introduce two colors $x_a, x_{\bar{a}}$ representing the positive and negative literal.

For each variable $a \in A$, insert two parallel edges between the nodes a and s of color x_a and $x_{\bar{a}}$, respectively. For each literal l of a clause $C_i \in C$, where l is the positive or negative variable a, add an edge between nodes C_i and a of color x_l to the graph.

The construction can be slightly modified to guarantee that the constructed graph is of bounded degree 5 (confer Fig. 2). Consider variable a. Instead of connecting the clause nodes directly to variable node a, we introduce auxiliary nodes which can be connected to node a by a subgraph of maximum degree 5. To avoid a root node of high degree, we choose a new color and replace the root by a suitable (with respect to the degree bound) tree of edges of the new color.

Let the reload cost function c be given as

$$c(x_{l_1}, x_{l_2}) := \begin{cases} K, & \text{if } l_1 = \bar{l}_2, \\ 0, & \text{if } l_1 = l_2, \\ 1, & \text{otherwise,} \end{cases}$$

where $K > 1$ is some large constant. Informally, the reload costs are expensive if two incident edges represent a variable and its negation, while they are low if the edges represent different variables.

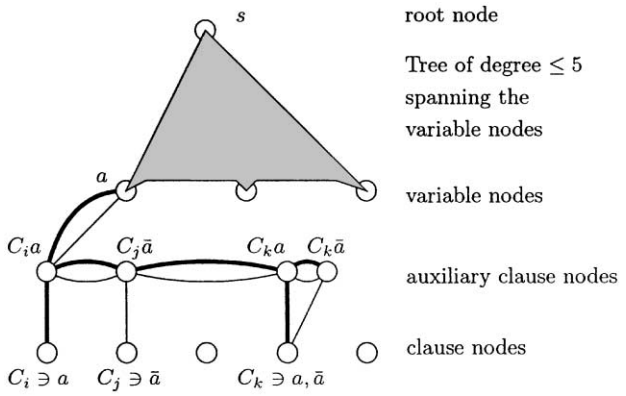

Fig. 2. Reduction for degree 5 bounded graphs.

Assume that there is a spanning tree T of G such that each clause node C_i is connected by a path of cost $< K+1$ to the root s. Without increasing path lengths to the root, one can force that each of the clause nodes is a leaf. Therefore, the path from a clause node to the root uses exactly one variable node and is of cost 1. Consequently, none of the auxiliary clause nodes is incident to edges of more than one color. Hence the colors of the edges adjacent to the variable nodes induce a valid assignment for π. Conversely, it is easy to see that a valid solution for π can be used to construct a spanning tree with the property that each clause node is connected by a path of cost one to the root.

As a consequence, if π has a valid solution, then an optimum spanning tree has diameter 2. On the other hand, the diameter is at least $K+2$ if π admits no valid assignment.

Assume that there is an approximation algorithm for DIAMETER-TREE with performance $f(n)$. Choose $K > f(n)$. Then the algorithm must solve the instance π' exactly which is equivalent to solving 3-SAT by our observations. □

For the remaining part of the paper we turn over to Δ-DIAMETER-TREE which is more interesting for practical applications as pointed out in the introduction. From the proof of Theorem 4 and the fact that the triangle inequality holds for $K \leqslant 2$, we get 2 as a lower bound on the approximability. The following modification of the construction guarantees a slightly better lower bound.

Setup two identical copies G_1, G_2 of the graph given above which are connected by identifying the root nodes. Now, if there is a spanning tree of diameter $< 2K+2$, then there must be one partial graph G_i with the property that each clause node is connected to the root by a path of cost $< K+1$ which means that the underlying instance of 3-SAT must have a solution. Conversely, if the underlying instance of 3-SAT has a valid assignment, then we can construct a spanning tree which joins each clause node to the root by a path of reload cost 1 and therefore has diameter 2. With $K = 2$, for any

$\alpha < (2K+2)/2 = 3$, any α-approximation algorithm for Δ-Diameter-Tree must in fact solve the underlying instance π exactly. This is summarized in the following corollary.

Corollary 5. *Unless* P = NP, Δ-Diameter-Tree *is not approximable with any factor* $\alpha < 3$, *even when restricted to graphs of maximum degree* 5.

In the remaining part of this section we will show a lower bound on the approximability of Δ-Diameter-Tree which holds on general graphs.

Theorem 6. *Unless* $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{O(\log\log n)})$, Δ-Diameter-Tree *is not approximable within any factor* $\alpha < 1/6 \ln n$ *on a graph with* n *nodes.*

Proof. We use a reduction from Minimum Dominating Set (Mds) [3, Problem GT 2]. Given a graph $G = (V,E)$, a node set $D \subseteq V$ is called *dominating*, if each node $v \in V \setminus D$ is adjacent to a node from D. A minimum dominating set is a dominating set of minimum cardinality. From [2] it follows that it is impossible to approximate Mds within a factor $\alpha < \ln|V|$ unless $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{O(\log\log n)})$.

Let $G' = (V',E')$ be an instance of Mds, $V' = \{v'_1, \ldots, v'_n\}$. In the following we will construct an instance of Δ-Diameter-Tree from G' where the spanning tree is related to a dominating set and the reload cost diameter to the cardinality of this set.

We first outline the main idea behind the construction. Start with a graph consisting of two layers, V and V^*, each being a copy of node set V', i.e., $V := \{v_i \mid v'_i \in V'\}$ and $V^* := \{v^*_i \mid v'_i \in V'\}$. Make layer V to be a complete graph. Then insert edges (v_i, v^*_j) between the layers exactly if v'_j is dominated by v'_i in G'. These edges form stars rooted at nodes in later V.

Assume that each of the stars is assigned a unique color, and that the reload costs between stars of different colors are large. Then a spanning tree of minimum reload cost diameter would prefer nodes of V^* to be leaves, and it appears to be a collection of stars joined by edges in layer V. The set of centers of those stars form a dominating set in the original graph. We now describe how to force that these joining edges form a path. After that we can exploit the fact that the diameter of the tree is related to the number of joining edges which is again related to the cardinality of the dominating set.

For each edge (v_i, v_j) in layer V place a node v_{ij} in the middle of that edge. For each node $v_k \in V$, replace the star around v_k by multiple copies, one copy for each pair v_i, v_j. Assure that reload costs between stars with identical centers are large, while costs equal 1 at the nodes placed in the middle of edges. This construction forces that star centers are incident to edges of one color only, and hence the joining edges described above form a path in layer V. We will now present a formal description of the details.

Given $G' = (V',E')$, construct a graph G with node set $V \cup V^\times \cup V^*$, where

$$
\begin{aligned}
V &:= \{v_i \mid v'_i \in V'\},\\
V^\times &:= \{v_{ij} \mid v_i, v_j \in V, i \neq j\} \quad \text{with } v_{ij} \text{ and } v_{ji} \text{ identified}\\
V^* &:= \{v^*_i \mid v'_i \in V'\}.
\end{aligned}
$$

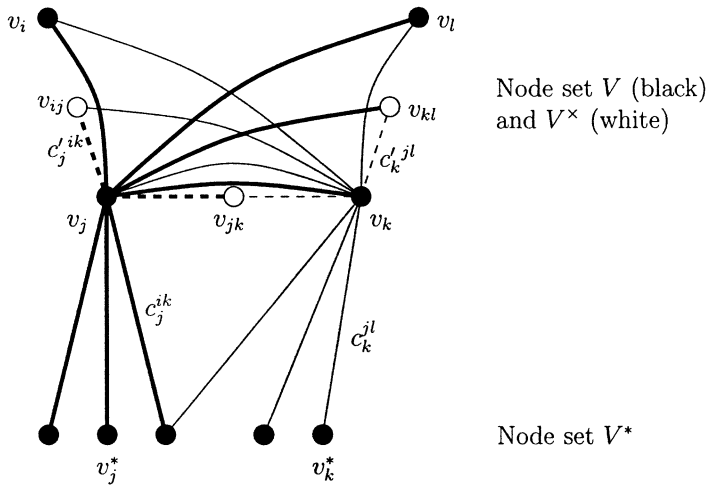

Fig. 3. Graph used in the reduction, restricted to the stars S^{ik}_j and S^{jl}_k.

Confer Fig. 3 for an illustration of the construction. For any pairwise disjoint $1 \leqslant i, j, k \leqslant n$, add a star S^{ik}_j with center v_j and fingers

$$\{v^* \mid v' \in N(v'_j)\} \cup (V \setminus \{v_j\}) \cup (V^\times \setminus \{v_{ij}, v_{jk}\})$$

of color c^{ik}_j. (By $N(v')$ we denote the neighbors of v' in graph G' including v' itself.) Again, identify colors c^{ik}_j and c^{ki}_j, and remove parallel edges of the same color. Now complete star S^{ik}_j by fingers v_{ij} and v_{jk} of color c'^{ik}_j. Call such a color a *path-color*.

Finally, for $1 \leqslant i, j, k, l \leqslant n$, pairwise disjoint, $i = l$ allowed, set the reload costs

$$c(c^{ik}_j, c'^{ik}_j) := 1,$$
$$c(c'^{ik}_j, c'^{jl}_k) := 1,$$

otherwise set the reload costs to some large constant $\Omega > n + 1$.

Notice that the resulting graph is of size polynomial in the size of G'. In fact, if $n' = |V'|$ is the number of nodes of G', then the constructed graph consists of $n = 1/2(n'^2 + 3n')$ nodes.

We claim that the triangle inequality is satisfied. Consider a node from V^*. Since there are no path-colored edges incident, for all pairs of incident edges the reload costs are equal to Ω and the triangle inequality is satisfied. Consider a node $v_{jk} \in V^\times$. The set of incident colors contains a set of path colors with pairwise reload cost 1. All remaining costs equal Ω. Therefore the triangle inequality holds even in this case. Finally, consider a node $v_j \in V$. There are no three (pairwise different) incident colors such that two of the pairs have reload cost 1 each. Hence the triangle inequality is satisfied even in this remaining case.

We remark that the graph induced by node set $V \cup V^{\times}$ is independent on the structure of G'. Observe that there is always a spanning tree with diameter at most $n+1$: Choose a path-colored path in the graph induced by $V \cup V^{\times}$ such that the two edges adjacent to a node $v_j \in V$ are of the same color, say $c_j'^{ik}$. Connect the remaining nodes from V^* to the tree by edges of the star S_j^{ik}. Consequently, in any optimum solution to Δ-DIAMETER-TREE there is no node at which reload costs Ω arise.

Consider a spanning tree with no node where reload cost Ω appear. This tree must consist of a path-colored path in the graph induced by $V \cup V^{\times}$, supplemented by some stars with fingers in $V \cup V^{\times} \cup V^*$. If K, $K \subseteq V$, is the set of nodes from V on the path, then the diameter of the tree is at most $|K|+1$. Moreover, the corresponding set K' is a dominating set in G'. Conversely, it is easy to construct a spanning tree of diameter at most $|K'|+1$ in graph G out of a dominating set K' in graph G'.

Let OPT be the diameter of an optimum solution in G, then there is a dominating set in G' of size OPT -1. (Recall $n = 1/2(n'^2 + 3n')$.) Assume there was an approximation algorithm for Δ-DIAMETER-TREE with performance $\alpha < 1/6 \ln n$. Then out of the approximate solution on G we can construct a dominating set in G' of size at most

$$\begin{aligned}\alpha\,\mathrm{OPT} - 1 < \tfrac{1}{6}\ln n\,\mathrm{OPT} - 1 &\leqslant \tfrac{1}{6}\ln(n'^2 + 3n')2(\mathrm{OPT}-1)\\ &\leqslant \tfrac{1}{3}\ln n'^3(\mathrm{OPT}-1) = \ln n'(\mathrm{OPT}-1).\end{aligned}$$

This would imply a polynomial time approximation algorithm for MDS with performance better than $\ln|V'|$ which is a contradiction. □

4. Exact solution for graphs with maximum degree 3

In this section we provide an exact algorithm for graphs with maximum degree 3. We assume that the reload costs satisfy the triangle inequality which is not a serious restriction in practical applications as stated in the introduction. The main idea is to map the graph G with reload costs to an equivalent graph H with edge lengths and then use known algorithms for finding a minimum diameter spanning tree on an edge weighted graph for solving the problem.

Given a graph $G=(V,E)$, we setup a graph H with node set $V(H) := V \cup V^{\varphi} \cup E^{\varphi}$. Here φ is a bijection; in other words, the node set of H consists of two copies of the node set of G and additionally one node for each edge of G. For each edge $e=(v,w) \in E$, graph H contains the four edges (v,e^{φ}), (e^{φ},w), $(v^{\varphi},e^{\varphi})$, and $(e^{\varphi},w^{\varphi})$. The construction is illustrated in Figs. 4 and 5.

The length of edges incident to a node from set V is adjusted such that the sum of the edge lengths on a walk through the node equals the reload costs of the related

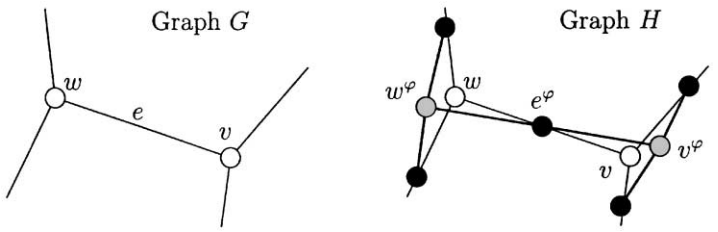

Fig. 4. Construction of the auxiliary graph.

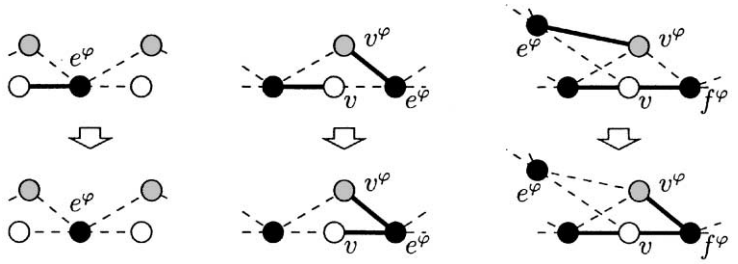

Fig. 5. Edge swaps performed by Algorithm REMOVE Left: line 2; center: line 4; right: line 6

walk in the original graph, i.e.,

$$l(e_1^\varphi, v) + l(v, e_2^\varphi) = c(e_1, e_2) \quad \text{for all } v \in V,\ e_i \text{ incident to } v. \tag{1}$$

For a node v of degree d, the set of equations (1) consists of $\binom{d}{2}$ equations with d variables which in general does not have a solution if $d > 3$. Therefore, we restrict ourselves to graphs of maximum degree 3. The length of the remaining edges, i.e., edges incident to a node from set V^φ, is set to a large constant $\Omega > \max\{c(x,x') \mid x,x' \in X\}$.

By this construction we are enabled to express the cost of paths in G in terms of the length of corresponding paths in H as stated in the following lemma, which is proven by a straightforward application of Eq. (1).

Lemma 7. *Let $G=(V,E)$ be a graph with reload cost function c. Let H be the graph constructed as described above, and l be the computed edge lengths. Then for each pair $v,w \in V$,*

$$dist_G^c(v,w) = dist_H^l(v^\varphi, w^\varphi) - 2\Omega.$$

Moreover, if c satisfies the triangle inequality, then $l \geqslant 0$.

At this point we call Algorithm MINDIAMETERSPANNINGTREE as described by Hassin and Tamir [4] on the graph H to compute a minimum diameter spanning tree T_1 of graph H with respect to edge lengths l.

We can assume without loss of generality that nodes from V^φ do not appear as interior nodes of T_1. This is due to the fact that the path through a node v^φ would consist of two Ω-edges and could be replaced by a corresponding path through node v without increasing the diameter of T_1. As a consequence, by removing all edges of weight Ω from T_1, the resulting subgraph would still be a tree spanning the node set $V \cup E^\varphi$.

An Ω-length edge (v^φ, e^φ) is called a *dangling edge*, if its *projection*, i.e., the edge (v, e^φ), is not part of the tree. Notice that if T_1 has no dangling edges, then its projection is still a tree which spans V. Moreover, removing all Ω-weight edges yields a tree of minimum diameter as will be shown in the following. It is the task of Algorithm REMOVE to remove all dangling edges from T_1 without increasing the diameter of the tree.

Lemma 8. *Algorithm* REMOVE *performs at most* $|E| + 2|V|$ *iterations.*

Lemma 9. *The subgraph* T_2 *produced by Algorithm* REMOVE *is a tree spanning the node set* $V \cup V^\varphi$. *No node from* E^φ *appears as a leaf in* T_2. *There is no dangling edge in* T_2.

Proof. The first claim clearly holds for the initial graph T_1. Each edge swap operation is performed at leaves and therefore does not affect the connectivity. A node is removed if and only if it is a leaf from set E^φ. Dangling edges are removed by the algorithm. □

Lemma 10. *If* $T_2 := \text{REMOVE}(T_1)$ *is the tree computed by Algorithm* REMOVE, *then*

$$diam^l(T_2) \leqslant diam^l(T_1).$$

Proof. We show that in each iteration the diameter does not increase.

Line 2: Removal of a leaf cannot increase the diameter.

Line 4: Assume that the diameter of the tree would strictly increase by the operation. Then the new diameter is attained by a path p starting with the inserted edge (v, e^φ) and ending in an edge of weight Ω. Since $|V^\varphi| = |V| \geqslant 2$, the ending edge is not (v^φ, e^φ). By replacing the starting edge (v, e^φ) by edge (v^φ, e^φ), we construct a longer path which was part of the tree before the operation. This contradicts our assumption.

Line 6: Consider f^φ as the root of the current tree before the operation. Since the condition in line 2 was not satisfied, f^φ is not a leaf, hence there are at least two subtrees hanging from f^φ. Each of the subtrees must contain at least one Ω-length edge and therefore have height at least Ω, since otherwise one of the conditions in line 2 or line 4 would have been satisfied. Therefore adding the edge (v^φ, f^φ) of length Ω to the root does not increase the diameter. □

Algorithm 2 assembles the presented algorithms and constructs the final solution for the original problem on graph G.

Algorithm 1 Algorithm REMOVE.

Input: A tree T_1 spanning the node set $V \cup V^\varphi \cup E^\varphi$, such that no node from V^φ is an interior node

```
1  repeat
2    if there is a leaf e^φ ∈ E^φ in the tree then
3      remove e^φ from the tree
4    else if there is a dangling edge (v^φ, e^φ) such that v is a leaf then
5      replace the edge connecting v to the tree by edge (v, e^φ)
6    else if there is a dangling edge (v^φ, e^φ) such that v is an interior node then
7      let f^φ be one of the neighbors of v in the tree
8      replace (v^φ, e^φ) by (v^φ, f^φ)
9    end if
10 until no more changes have been made
```

Output: A tree T_2 spanning $V \cup V^\varphi$ with no dangling edges

Theorem 11. *Let T be the tree returned by Algorithm 2. Then for all spanning trees T' of G,*

$$diam^c(T) \leqslant diam^c(T'),$$

i.e., the tree T is optimal with respect to $diam^c$.

Proof. We show the claim by stating the following chain of inequalities,

$$diam^c(T) + 2\Omega \overset{\text{(i)}}{\leqslant} diam^l(T_2) \overset{\text{(iii)}}{\leqslant} diam^l(T_1) \overset{\text{(iii)}}{\leqslant} diam^l(T'_H) \overset{\text{(ii)}}{\leqslant} diam^c(T') + 2\Omega,$$

where T, T_1, and T_2 are trees as denoted by the algorithms, T' is an arbitrary spanning tree of G, and T'_H is a tree constructed out of T' as described below.

(i) Let v, w be arbitrary nodes in T. Let $p = (e_1, \ldots, e_k)$ be the path between v and w in T, and $c(p) = dist^c_T(v, w)$ its cost. By construction of T, for each edge $e = (v', w') \in T$, tree T_2 contains the two edges (v', e^φ) and (e^φ, w'). Therefore, the path q in T_2 from e^φ_1 to e^φ_k uses the nodes $e^\varphi_2, \ldots, e^\varphi_{k-1}$. From (1) it follows that q is of length $dist^l_{T_2}(e^\varphi_1, e^\varphi_k) = c(p)$.

We claim that there is in fact a path of length at least $c(p) + 2\Omega$ in T_2. To see this, we show that q can be augmented by a sub-path of length at least Ω at both endpoints. Consider endpoint e^φ_1, the other case is similar. By Lemma 9, v^φ is spanned by T_2; let (v^φ, f^φ) be the connecting edge, which is of length Ω. If $f^\varphi = e^\varphi_1$, we are done. Otherwise, since the edge is not dangling, we have $(f^\varphi, v) \in T_2$, and by construction of T, also $(v, e^\varphi_1) \in T_2$.

If we choose p as a maximal path, i.e., $c(p) = diam^c(T)$, it follows

$$diam^l(T_2) \geqslant diam^c(T) + 2\Omega. \tag{2}$$

(ii) Let $T' = (V, E')$ be an arbitrary spanning tree of G. Then we construct a tree T'_H in graph H by choosing the edge set $\bigcup_{e=(v,w)\in E'} \{(v, e^\varphi), (e^\varphi, w)\}$. Note that T'_H spans the node set $V \cup E'^\varphi$.

Algorithm 2 Algorithm for problem Δ-DIAMETER-TREE on graphs with maximum degree 3.

Input: Graph $G=(V,E)$ with reload costs c, maximum degree 3
1 Construct auxiliary graph H with edge lengths l
2 $T_1 \leftarrow$ MINDIAMETERSPANNINGTREE(H,l)
3 $T_2 \leftarrow$ REMOVE(T_1)
4 $E' \leftarrow \{e=(v,w)\in E \mid (v,e^{\varphi})\in T_2 \wedge (e^{\varphi},w)\in T_2\}$
Output: $T=(V,E')$

For any two nodes $e_1^{\varphi}, e_2^{\varphi} \in E'^{\varphi}$, by construction of the tree and using (1), we have $dist^{l}_{T'_H}(e_1^{\varphi}, e_2^{\varphi}) \leqslant diam^{c}(T')$. We claim that the remaining nodes can be connected to the tree T'_H such that for each node the distance to the nearest node in E'^{φ} is bounded by Ω. If this claim holds, we have

$$diam^{l}(T'_H) \leqslant diam^{c}(T') + 2\Omega. \tag{3}$$

The claim is true for each node in V. Complete the construction of T'_H in the following way: Connect nodes from V^{φ} to T'_H by one Ω-length edge each. Then, the claim also holds for the nodes in V^{φ}. Connect the remaining nodes from $E^{\varphi}\backslash E'^{\varphi}$ to T'_H by an arbitrary edge with endpoint in V each. Since the sum of the lengths of two edges adjacent to a node from V is bounded by Ω, even for the remaining nodes the claim holds.

(iii) Note that the resulting tree T'_H is a spanning tree in H. By Lemma 10 and optimality of T_1, we have

$$diam^{l}(T_2) \leqslant diam^{l}(T_1) \leqslant diam^{l}(T'_H). \tag{4}$$

Putting (2), (3), and (4) together, the claim follows. □

We now summarize our results.

Corollary 12. *Algorithm* 2 *solves problem* Δ-DIAMETER-TREE *on graphs with degree bound* 3. *The running time is in* $O(|E|^2 \log |E|)$.

Proof. It remains to show the claim on the running time. The auxiliary graph H has $2|V|+|E|$ nodes and $4|E|$ edges. From [4] it follows that MINDIAMETERSPANNINGTREE can be implemented to run in time $O(4|E|(2|V|+|E|)+(2|V|+|E|)^2 \log(2|V|+|E|)) \in O(|E|^2 \log |E|)$. By Lemma 8, REMOVE performs at most $O(|E|)$ iterations, each of which needs time $O(|E|)$. Hence, MINDIAMETERSPANNINGTREE dominates the running time and the claim follows. □

5. Conclusion

Table 1 shows a summary of results presented in this paper. The notion "approximability" refers to the existence of a polynomial time approximation algorithm with specified performance, assuming that $NP \subseteq DTIME(n^{O(\log\log n)})$ is not true. V

Table 1
Summary of results presented in this paper

Problem	Lower bound on approximability	Reference
DIAMETER-TREE	Any $f(\|V\|)$	Theorem 4
Δ-DIAMETER-TREE	$1/6 \cdot \ln \|V\|$	Theorem 6
DIAMETER-TREE, degree 5	Any $f(\|V\|)$	Theorem 4
Δ-DIAMETER-TREE, degree 5	3	Corollary 5
Δ-DIAMETER-TREE, degree 3	Polynomial time solvable	Corollary 12

denotes the set of nodes of the graph, and f is any polynomial time computable function.

A natural open question is the complexity of the problem for graphs with maximum degree 4. On the other hand, a major goal is to search for a non-trivial approximation algorithm on general graphs. Better results may also be obtained by other restrictions of the class of graphs or the reload cost function.

To our knowledge, reload costs have not been considered in the literature so far. This cost structure is related to node weighted graphs, but the new aspect is that the cost at a node depend on the edges used by the walk through that node.

It is possible to formulate other well known network design problems under the reload cost model, e.g. the problem of finding a spanning tree of minimum total cost. Here, the notion "total cost" is to be defined more precisely, since it makes sense either to count the number of reloading nodes (i.e., nodes with more than one color incident) or to sum up the costs of all possible paths in the resulting tree.

References

[1] R.-S. Chang, S.-J. Leu, The minimum labeling spanning trees, Inform. Process. Lett. 63 (1997) 277–282.
[2] U. Feige, A threshold of $\ln n$ for approximating set cover, in: *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC'96)*, 1996, pp. 314–318.
[3] M.R. Garey, D.S. Johnson, *Computers and Intractability (A guide to the theory of NP-completeness)*, W.H. Freeman, New York, 1979.
[4] R. Hassin, A. Tamir, On the minimum diameter spanning tree problem, Inform. Process. Lett. 53 (1995) 109–111.
[5] O. Kariv, S.L. Hakimi, An algorithmic approach to network location problems, part I: The p-center, SIAM J. Appl. Math. 37 (1979) 513–537.
[6] S.O. Krumke, H.-C. Wirth, On the minimum label spanning tree problem, Inform. Process. Lett. 66 (2) (1998) 81–85.
[7] J. Plesník, The complexity of designing a network with minimum diameter, Networks 11 (1981) 77–85.
[8] H.-C. Wirth, J. Steffan, On minimum diameter spanning trees under reload costs, in: *Proceedings of the 25th International Workshop on Graph-Theoretic Concepts in Computer Science* (WG'99), Lecture Notes in Computer Science, Vol. 1665, eds. P. Widmayer, G. Neyer, S. Eidenbenz, Springer, Berlin, 1999, pp. 78–88.