Discrete Applied Mathematics 220 (2017) 91–117

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Shortcutting directed and undirected networks with a degree constraint

Richard B. Tan [a], Erik Jan van Leeuwen [b], Jan van Leeuwen [c,*]

[a] Department of Computer Science, University of Sciences & Arts of Oklahoma, Chickasha, OK 73018, USA

[b] Department of Algorithms and Complexity, Max-Planck Institut für Informatik, Saarland Informatics Campus, D-66123 Saarbrücken, Germany

[c] Department of Information and Computing Sciences, Utrecht University, Princetonplein 5, 3584 CC Utrecht, The Netherlands

ARTICLE INFO

Article history:
Received 8 August 2014
Received in revised form 16 December 2016
Accepted 19 December 2016
Available online 13 January 2017

Keywords:
Networks
Graphs
Rooted directed trees
DAGs
Compression
Shortcutting
Diameter
Feedback dimension
Path covers
Stability number
NP-hardness
$W[2]$-hardness

ABSTRACT

Shortcutting is the operation of adding edges to a network with the intent to decrease its diameter. We are interested in shortcutting networks while keeping degree increases in the network bounded, a problem first posed by Chung and Garey. Improving on a result of Bokhari and Raza we show that, for any $\delta \geq 1$, every undirected graph G can be shortcut in linear time to a diameter of at most $O(\log_{1+\delta} n)$ by adding no more than $O(n/\log_{1+\delta} n)$ edges such that degree increases remain bounded by δ. The result extends an estimate due to Alon et al. for the unconstrained case. Degree increases can be limited to 1 at only a small extra cost. For strongly connected, bounded-degree directed graphs Flaxman and Frieze proved that, if ϵn random arcs are added, then the resulting graph has diameter $O(\ln n)$ with high probability. We prove that $O(n/\ln n)$ edges suffice to shortcut any strongly connected directed graph to a graph with diameter less than $O(\ln n)$ while keeping the degree increases bounded by $O(1)$ per node. The result is proved in a stronger, parameterized form. For general directed graphs with stability number α, we show that all distances can be shortcut to $O(\alpha\lceil \ln \frac{n}{\alpha}\rceil)$ by adding only $\frac{4n}{\ln n/\alpha} + \alpha\phi$ edges while keeping degree increases bounded by at most $O(1)$ per node, where ϕ is equal to the so-called feedback-dimension of the graph. Finally, we prove bounds for various special classes of graphs, including graphs with Hamiltonian cycles or paths. Shortcutting with a degree constraint is proved to be strongly NP-complete and $W[2]$-hard, implying that the problem is neither likely to be fixed-parameter tractable nor efficiently approximable unless $FPT = W[2]$.

1. Introduction

Shortcutting is the operation of adding links (lines, edges) to a network with the intent to decrease its diameter. Shortcutting networks increases their transmission capacity and decreases network delay. Adding links to nodes in order to reduce a network's diameter is not free of charge, however. In many instances, the number of links that can be added to a node is limited due to physical or even economical constraints. Hence, in reality one may be able to add only a limited number of extra links per network node. It is this type of constraint that we are interested in.

* Corresponding author.
E-mail addresses: rbtan@usao.edu (R.B. Tan), erikjan@mpi-inf.mpg.de (E.J. van Leeuwen), J.vanLeeuwen1@uu.nl (J. van Leeuwen).

http://dx.doi.org/10.1016/j.dam.2016.12.016
0166-218X/

Specifically, we are interested in δ-*shortcutting* arbitrary networks G. In this problem we wish to shortcut a network subject to the constraint that the number of edges added per node is bounded by a fixed integer value $\delta \geq 1$. We require also that *added edges only link nodes that are connected in the transitive closure of* G, to remain faithful to the structure of the network. We model networks as finite graphs and phrase the shortcutting problem accordingly. We consider the following general question, for both undirected and directed n-node graphs G:

> *what reductions in existing node-to-node distances or diameter are achievable by shortcutting* G, *and how many extra edges are needed for it*, *when degree increases must remain limited by at most a small amount at every node*.

The study of shortcutting graphs *without* constraints seems to have been initiated by Chung and Garey [16]. In the undirected case, many studies show how to achieve small, even constant diameters with only a linear number of additional edges, both for special graph classes and in general (see e.g. [1,7,9,44]). The problem of determining whether some number of edges suffices to achieve a non-trivial reduction in diameter has been studied in many papers, leading e.g. to tight bounds, NP-hardness and even W[2]-hardness results [16,24,38]. Also the complexity of approximating the number of edges needed to achieve a certain diameter decrease has been studied [6,17,34]. Integer LP-techniques have been applied to it for achieving shortest paths between nodes in the expected case on certain classes of networks [10]. For the directed case, Thorup [42,43] showed that all m-edge planar digraphs can be shortcut to a poly-logarithmic diameter by the addition of at most m extra edges, but Hesse [27] showed that this fact does not hold for digraphs in general. Finally, a shortcutting of a graph may be viewed as a special case of a so-called *transitive-closure spanner* of the graph, although the latter has a different connotation (see e.g. [36]).

The shortcutting problem as we study it here, i.e. *with* degree constraint, seems to have received very little attention before. Chung and Garey [16] suggested to constrain the maximum degree of nodes in the problem, but few results seem to have been obtained for it. The only earlier study of δ-shortcutting seems to be due to Bokhari and Raza [8], who considered the problem for undirected graphs for the interesting case $\delta = 1$. In this paper we study the δ-shortcutting problem in general and aim at sharp bounds on the reductions in diameter or inter-node distances that can be achieved. In the sequel, when we speak of shortcutting we will always mean δ-shortcutting for some $\delta \geq 1$.

1.1. Results

We consider the shortcutting problem for the undirected and directed cases separately. We first review our results for *undirected* graphs. One fact can be noted straight off: a graph with maximum degree $\Delta \geq 3$ can have a diameter of $\log_{\Delta-1} n$ at best, based on the Moore bound (see Section 2). Hence, when a graph is δ-shortcut, one can hope for a diameter of about $\log_{1+\delta} n$ at best (taking $\Delta \geq 2 + \delta$ in the above bound).

For $\delta = 1$, the first results for the δ-shortcutting problem were obtained by Bokhari and Raza [8]. They showed that any connected undirected graph can be 1-shortcut to a diameter D with $D = O(\log_2 n)$, by adding at most n edges. They also showed that the edges needed for the shortcutting can be determined by an $O(n^2)$ algorithm. In Section 3 we improve on this, by giving an algorithm that 1-shortcuts a graph to a diameter D with $D = O(\log_2 n)$ by adding only $O(\frac{n}{\log_2 n})$ edges, by means of an $O(n)$ algorithm.

The improved bound is an instance of a more general result. Note that Alon et al. [1] (see also [35]) already showed that *without* degree constraints, any connected undirected graph G can be reduced to diameter D by adding at most $\frac{n}{\lfloor D/2 \rfloor}$ edges. We prove that any connected graph G can be shortcut in linear time to a diameter $O(D)$ by adding at most $\frac{n}{\lfloor D/2 \rfloor}$ edges while keeping degree increases smaller than $n^{\frac{2}{D}}$ (provided $D \geq 4$). Reformulating this for the shortcutting problem, the result states that for any integer $\delta \geq 1$, any connected undirected graph can be δ-shortcut in linear time to a diameter $O(\log_{1+\delta} n)$ by adding at most $O(\frac{n}{\log_{1+\delta} n})$ extra edges. The degree increases can be limited to 1 at the expense of an extra factor δ in diameter but saving a factor δ on the number of extra edges. As Alon et al. [1] proved their result to be worst-case optimal, so is our degree-constrained extension of it. We show this in Section 3.

Next we consider the shortcutting problem for *directed* graphs. Before turning to the general case, we prove several key results for δ-shortcutting rooted directed paths and δ-compressing rooted directed trees in Sections 4 and 5. These results are used in Section 6, e.g. to obtain useful bounds for 1-shortcutting DAGs and rooted directed trees depending on parameters like the width of the DAG or the height of the tree, respectively. As another step towards the general case, we consider the problem of shortcutting strongly connected digraphs.

For strongly connected digraphs, we note that Flaxman and Frieze [21] proved for the bounded-degree case that, if ϵn random arcs are added to the graph, then the resulting graph has diameter $O(\ln n)$ with high probability. We show that *any* strongly connected n-node digraph can be shortcut to a diameter of $O(\ln n)$, by adding $O(\frac{n}{\ln n})$ arcs and keeping degree increases bounded by 2. The bounds follow from a more general result that can be tuned in various further ways.

For general directed graphs we prove several bounds in Sections 6 and 7. In particular, in Section 7 we show that all distances in an n-node directed graph can be shortcut to $O(\alpha(G) \cdot \lceil \ln \frac{n}{\alpha(G)} \rceil)$, again by the addition of at most a sublinear number of arcs and keeping degree increases bounded by 2. Here $\alpha(G)$ is the *stability number* of G. The result involves an interesting application of the Gallai–Milgram theorem. As a corollary we show that every tournament can be 2-shortcut to a diameter $O(\ln n)$, by adding at most a linear number of arcs. To estimate the number of shortcut arcs needed in the results

Table 1
Main results on shortcutting and compression. Column 'Delta' lists the bounds on the degree increases. For notation and details, see the theorem cited in the last column.

Graph class	Diameter	♯ Edges	Delta	Thm
Undirected	$O(\frac{D}{\beta})$	$\frac{n}{\lfloor D/\beta \rfloor}$	$n^{\frac{\beta}{D}}$	2
	$O(\log_{1+\delta} n)$	$\frac{n}{\log_{1+\delta} n}$	δ	3
	$O(\delta \log_{1+\delta} n)$	$\frac{n}{\delta \log_{1+\delta} n}$	1	4
Rooted directed path	$O(\delta \log_{1+\delta} n)$	$\frac{2n}{\delta \log_{1+\delta} n}$	1	6
Rooted directed tree[a]	$O(\log_{1+\delta} n)$	$O(\frac{\delta n}{\log_{1+\delta} n})$	$O(\delta)$	7
	$O(\delta \log_{1+\delta} n)$	$\frac{n}{3 \log_{1+\delta} n}$	1	8
Rooted directed tree	$O(\Delta\delta \lceil \log_{1+\delta} h \rceil \log_2 n)$	$\frac{4n}{\delta \log_{1+\delta} \log_{1+\delta} h}$	1	10
Directed acyclic	$O(\delta w \lceil \log_{1+\delta} n/w \rceil)$	$\frac{4n}{\delta \log_{1+\delta} n/w}$	1	9
Strongly connected	$O(\log_{1+\delta} n)$	$O(\frac{\delta n}{\log_{1+\delta} n})$	$O(\delta)$	11
	$O(\delta \log_{1+\delta} n/\delta)$	$O(\frac{n}{\log_{1+\delta} n})$	2	12
General directed	$O(\delta w_c \log^2_{1+\delta} n)$	$O(\frac{\delta n}{\log_{1+\delta} n_{\min}})$	$O(\delta)$	14
	$O(\delta\alpha \lceil \log_{1+\delta} \frac{n}{\alpha} \rceil)$	$\frac{4n}{\delta \log_{1+\delta} n/\alpha} + \alpha\phi$	2	16
Hamiltonian directed	$O(\delta \log_{1+\delta} n)$	$\frac{4n}{\delta \log_{1+\delta} n} + 1$	2	18

[a] Compression.

in detail, we exploit a new graph parameter, the *feedback dimension* $\phi(G)$. An overview of the main results achieved in this paper is shown in Table 1.

Finally, in Section 8 we consider the *complexity* of shortcutting. It is well-known that *without* degree constraint, the problem of deciding whether adding a certain number of edges can reduce the diameter even by 1 is computationally hard. We argue that even when the degree constraint *is* taken into account, the problem remains both (strongly) NP-complete and $W[2]$-hard. This implies that the exact number of edges to be added to a graph in order to decrease its diameter while allowing degrees to increase only by a given constant is likely to be hard to compute in general and not even fixed-parameter tractable, with the number of extra edges as the parameter (unless some powerful hypothesis fails). We also prove that shortcutting is not likely to be well-approximable by means of a polynomial-time algorithm. In Section 9 we give some conclusions and mention some problems for further research.

2. Preliminaries

In this section we list a number of basic concepts and results that will be used in the sequel.

2.1. Graph theory

$G = \langle V, E\rangle$ will denote a connected undirected or directed graph with vertex set V and edge set E. Edges will be called arcs in the directed case. We do not allow self-loops. We let $n = |V|$, $m = |E|$ and assume that $n, m > 1$ throughout. The complement of G is the graph $\overline{G} = \langle V, \overline{E}\rangle$, where $\overline{E}$ consists of the edges (arcs) e between distinct nodes of V such that $e \notin E$.

The complete undirected graph on n nodes will be denoted by K_n. A *tournament* on n nodes is any directed graph that can be obtained from K_n, by assigning a unique orientation to each of its edges.

The *degree* of a node v is the number of edges incident to v. The degree Δ of a graph G is the maximum degree of any node in V. In directed graphs we distinguish between the *in-degree* and the *out-degree* of a node. When δ-shortcutting a directed graph we use δ_{in} and δ_{out} to denote the increases in in- and out-degree, respectively. We let $\delta = \max\{\delta_{in}, \delta_{out}\}$ in this case.

A *walk* in a graph G is any alternating sequence $v_1, e_2, v_2, \ldots, e_k, v_k$ of nodes and edges (arcs) of G, for some $k \geq 1$, such that for each i $(2 \leq i \leq k)$, e_i is incident to both v_{i-1} and v_i. (In the directed case this means that for each i, arc e_i points from v_{i-1} to v_i.) In walks, it is assumed that no edge (arc) appears more than once. A *path* is a walk in which no vertex appears more than once.

A *rooted directed path* is a directed graph $G = \langle V, E\rangle$ with $V = \{v_1, \ldots, v_n\}$ and $E = \{(v_i, v_{i+1}) \mid 1 \leq i < n\}$. The sub-path from v_i to v_j $(i < j)$, denoted by $[v_i, v_j]$, is called a *segment* of G.

A *rooted tree* is a tree in which one node, denoted by r, is designated as the *root*. We think of a rooted tree as having its root at the 'top' and all its edges drawn downward. In a *rooted directed tree* we assume that all arcs are directed 'away from the root'. An *ordered* tree is a tree in which the subtrees at every internal node are ordered from left to right.

The *depth* of a rooted tree is the length of the longest path from its root to a leaf of the tree. The *weight* of a node in a rooted tree is the number of nodes in its subtree (itself included). The following fact is known as the *centroid theorem* for rooted trees.

Fact 1. *A rooted directed tree with n nodes and all its out-degrees* ≤ 2 *must contain an internal node* v *such that the subtree rooted at* v *contains between* $\frac{1}{3}n$ *and* $\frac{2}{3}n$ *nodes.*

Fact 2. *A rooted directed tree with l leaves and all internal out-degrees* ≥ 2 *has at most* $2 \cdot l - 2$ *arcs.*

In the case of directed acyclic graphs (DAGs), directed paths are normally called *chains*. An *anti-chain* is any set of nodes of which no two lie on a same directed path in G. The *depth* $d(G)$ of a directed acyclic graph G is the length of the longest chain in G, the *width* $w(G)$ is the size of the largest anti-chain in G.

In graphs G, we write $v \overset{\star}{\to} w$ ($v \overset{k}{\to} w$) to denote that nodes v and w are connected by a (directed) path (of length k). The length (number of edges) of a path π will be denoted by $|\pi|$. If a path from v to w exists in G, then the *distance* from v to w is defined as the length of the shortest path from v to w. Otherwise their distance is said to be infinite or undefined. The *diameter* D of G is the maximum of all node-to-node distances in the graph. Thus, the diameter of a (directed) graph is finite or defined only if it is (strongly) connected.

2.1.1. Path covers

Let G be a directed graph. The *path cover number* $\mu(G)$ of G is the smallest number of node-disjoint directed paths that cover (partition) the entire graph.

For directed acyclic graphs G, a decomposition of G into a number of disjoint chains is called a *chain decomposition* of G. The well-known theorem of Dilworth (cf. [39], Corollary 14.2a) implies the following:

Fact 3. *For any directed acyclic graph G, one has* $\mu(G) = w(G)$.

For directed acyclic graphs G, both $\mu(G)$ and a smallest path cover can be computed in polynomial time by means of standard techniques from combinatorial optimization [39]. Any decomposition of G into $w(G)$ chains is called a *Dilworth decomposition*.

Let G be a general directed graph. A stable (or independent) set in G is any set of nodes in G that are pairwise non-adjacent. The *stability number* $\alpha(G)$ of G is defined as the size of the largest stable set in G. The following fact follows from the Gallai–Milgram theorem [23].

Fact 4. *For any directed graph G, one has* $\mu(G) \leq \alpha(G)$.

It is known that every strongly connected directed graph G with $\alpha(G) \leq 2$ has a Hamiltonian path [14]. It is therefore conjectured that Fact 4 can be strengthened to $\mu(G) \leq \alpha(G) - 1$ in the case of strongly connected digraphs [4].

Both $\mu(G)$ and $\alpha(G)$ are NP-hard to determine, in general. However, it is known that a path cover of at most k disjoint paths together with an independent set of k nodes, for some $k \leq \alpha(G)$, can be computed in polynomial time [11].

2.1.2. Moore trees

A d-tree is defined as an (undirected) rooted tree in which every node has degree at most d. A *full* (or Moore) d-tree of n nodes is a d-tree with n nodes in which all levels are filled to maximum size except possibly the lowest level. In this case, if the leaves all appear in level $k > 0$, all internal nodes except possibly those in level $k - 1$ have full degree d.

A full d-tree is the tightest way of packing n nodes in a degree-d graph while minimizing diameter. This follows from the *Moore bound* which states that the number of nodes n in a graph of degree d and diameter D must satisfy (cf. [15]):

$$n \leq 1 + d + d(d-1) + \cdots + d(d-1)^{D-1} = 1 + \frac{d}{d-2}((d-1)^D - 1).$$

Note that a full d-tree of n nodes has a depth at most $\log_{d-1} n$ ($d \geq 3$).

A *complete* d-tree is a finite d-tree in which all levels from the root down are filled to the maximum possible size. In a complete d-tree the root has d sons, and all other interior nodes have $d - 1$ sons. Clearly, given a full d-tree T on n nodes, the smallest extension of T to a complete d-tree will have at most dn nodes (obtained by completing the degrees in the lowest interior level of T).

2.2. Complexity theory

We use several basic facts from the analysis of algorithms and computational complexity theory. P and NP are the classes of decision problems that are solvable in deterministic and nondeterministic polynomial time, respectively. By definition P $\subseteq$ NP, but it is unknown whether P = NP. We assume that the reader is familiar with the theory of NP-completeness [25]. A problem is said to be *strongly NP-complete* if it is NP-complete even when its numeric parameters are bounded by a polynomial in the length of the instance.

The classes of P- and NP-optimization problems are denoted by PO and NPO, respectively (cf. Ausiello et al. [2]). An optimization problem is said to be *efficiently r-approximable* if there exists a polynomial-time algorithm to compute a solution to the problem that is within a factor r from the optimum. The class of all NPO-problems which can be efficiently c-approximated for some constant c, is denoted by APX.

The quest for efficient algorithms to approximate optimal solutions within a factor $1+\epsilon$ for any $\epsilon > 0$ has given rise to *polynomial-time, efficient polynomial-time* and *fully polynomial time approximation schemes*. The corresponding problem classes satisfy:

$$PO \subseteq FPTAS \subseteq EPTAS \subseteq PTAS \subseteq APX \subseteq NPO$$

where most inclusions are proper provided P $\neq$ NP (EPTAS $\subset$ PTAS unless FPT $=$ W[1], see below). For an overview of the different approximation classes and their interrelationships, see [2,19,45].

Finally, a problem is called *fixed-parameter tractable* if its instances $\langle I, k\rangle$ can be decided or solved by an algorithm with a running time of the form $O(f(k)\cdot n^{O(1)})$, where f is some computable function depending only on the parameter k and n the problem size. In parameterized complexity theory one classifies many well-known decision problems in the so-called W-hierarchy [19]:

$$FPT = W[0] \subseteq W[1] \subseteq W[2] \subseteq \cdots \subseteq W[poly]$$

with the class of fixed-parameter tractable problems (FPT) at the base level. None of the inclusions in the hierarchy is known to be strict. However, all classes are closed under *standard parameterized reductions* and have complete problems under the implied notion of reducibility. We will later use the following problem, of which the decision version is known to be strongly NP-complete and the parameterized version is $W[2]$-complete (cf. [19], p. 444):

Hitting Set
Input: a universe $U = \{u_1, \ldots, u_n\}$, a family $\mathcal{S}$ of subsets $S_1, \ldots, S_m$ of U, and an integer $k \geq 1$.
Question: does $(U, \mathcal{S})$ have a 'hitting set' of size at most k, i.e. does there exist a subset $H \subseteq U$ with $|H| \leq k$ such that $S_i \cap H \neq \emptyset$ for $i = 1, \ldots, m$.

Hitting Set is not in FPT and thus also not in EPTAS (cf. [33], Prop. 6.1) unless $FPT = W[2]$. Further facts for the inapproximability of Hitting Set follow from its duality to the Set Cover problem, for which strong results are known (cf. [2], problem SP7, p. 426).

2.3. Inequalities

In later analyses we will use the following extension of *Jensen's inequality* (cf. [28]) for convex and concave functions. Let f and g be functions defined over some common interval.

Lemma 1. *Let f, g be positive real functions such that f is concave, and $\frac{f}{g}$ and g are both monotone nondecreasing. Then for any integer $k \geq 1$ and any $x_1, \ldots, x_k$ in the joint domain of f and g, one has*

$$\frac{f(x_1)}{g(x_1)} + \cdots + \frac{f(x_k)}{g(x_k)} \leq 2k \cdot \frac{f(x_{av})}{g(x_{av})}$$

where $x_{av} = \frac{1}{k}(x_1 + \cdots + x_k)$.

The lemma follows by taking $\mu_1 = \cdots = \mu_k = 1$ in the following, more general inequality. The inequality is sufficient for our purposes.

Theorem 1. *Let f, g be positive real functions such that f is concave, and $\frac{f}{g}$ and g are both monotone nondecreasing. Let $\mu_1, \ldots, \mu_k$ be positive weights. Then for any integer $k \geq 1$ and any $x_1, \ldots, x_k$ in the joint domain of f and g, one has*

$$\mu_1\frac{f(x_1)}{g(x_1)} + \cdots + \mu_k\frac{f(x_k)}{g(x_k)} \leq 2\Sigma\mu_i \cdot \frac{f\left(\frac{1}{\Sigma\mu_i}(\mu_1x_1 + \cdots + \mu_kx_k)\right)}{g\left(\frac{1}{\Sigma\mu_i}(\mu_1x_1 + \cdots + \mu_kx_k)\right)}.$$

Proof. Assume w.l.o.g. that $k \geq 2$, that the arguments are ordered: $x_1 \leq \cdots \leq x_k$, and that the x_i's are not all equal. Let $x_{av} = \frac{1}{\Sigma\mu_i}(\mu_1x_1 + \cdots + \mu_kx_k)$. Then there must be an index t such that $x_i \leq x_{av}$ for all $1 \leq i \leq t$ and $x_j \geq x_{av}$ for $t+1 \leq j \leq k$. Now write:

$$\mu_1\frac{f(x_1)}{g(x_1)} + \cdots + \mu_k\frac{f(x_k)}{g(x_k)} = \left(\mu_1\frac{f(x_1)}{g(x_1)} + \cdots + \mu_t\frac{f(x_t)}{g(x_t)}\right) + \left(\mu_{t+1}\frac{f(x_{t+1})}{g(x_{t+1})} + \cdots + \mu_k\frac{f(x_k)}{g(x_k)}\right).$$

By monotonicity, the first summand is bounded by

$$(\mu_1 + \cdots + \mu_t)\frac{f(x_{av})}{g(x_{av})} \leq \Sigma\mu_i \cdot \frac{f(x_{av})}{g(x_{av})}.$$

By Jensen's inequality one has $\mu_1 f(x_1) + \cdots + \mu_k f(x_k) \leq \Sigma\mu_i f(x_{av})$. Thus the second summand can be bounded by

$$\mu_{t+1}\frac{f(x_{t+1})}{g(x_{av})} + \cdots + \mu_k\frac{f(x_k)}{g(x_{av})} \leq \frac{1}{g(x_{av})}(\mu_1 f(x_1) + \cdots + \mu_k f(x_k)) \leq \Sigma\mu_i \cdot \frac{f(x_{av})}{g(x_{av})}.$$

By combining these estimates, the theorem follows. □

3. δ-shortcutting undirected graphs

Bokhari and Raza [8] proved that one can 1-shortcut a graph and get a diameter D with $D \leq 4\log_2\frac{n+2}{3}$ by adding at most n edges using an $O(n^2)$-time algorithm. In this section we prove a better bound and give a linear-time algorithm. In fact, we will do this for δ-shortcutting in general.

We note that Alon et al. [1] showed that the diameter of any connected undirected graph can be reduced to (at most) D by adding at most $\frac{n}{\lfloor D/2 \rfloor}$ edges, without taking a degree constraint into account. We show that for any $\delta \geq 1$ one can shortcut a graph to a diameter of $O(\log_{1+\delta} n)$, by adding at most $O(\frac{n}{\log_{1+\delta} n})$ edges in linear time, while keeping the degree increases bounded by δ. We also show a corresponding result in which the degree increases remain bounded by 1.

We outline the general principle of the construction in Section 3.1 and then derive the concrete bounds in Section 3.2. In Section 3.3 we discuss the optimality of the bounds.

3.1. Constructing clusters

Let G be a connected undirected graph with $n > 1$ nodes and m edges, and let λ be an integer with $1 \leq \lambda \leq n-1$. Define a λ-*cluster* in G to be any connected subset of $\lambda + 1$ nodes in G. Because $n \geq \lambda + 1$, G contains at least one λ-cluster. The diameter of a λ-cluster is at most λ.

Definition 1. A set of λ-clusters $C_1, \ldots, C_k$ in G is called *good* if the clusters are (node-)disjoint and any node u of G that does not belong to any of the clusters has a distance of at most λ to at least one of them.

If $C_1, \ldots, C_k$ are (node-)disjoint λ-clusters in G, then necessarily $k \leq \frac{n}{\lambda+1}$. Good sets of λ-clusters exist, e.g. any maximal set of λ-clusters is good.

Lemma 2. *A good set of disjoint λ-clusters in G can be computed in $O(n+m)$ time.*

Proof. We first compute a rooted spanning tree S of G, which can be done in linear time by any of several standard techniques [29]. Note that S has $O(n)$ nodes and edges. Now traverse S in *maze-order* recursively as follows: visit the root r of S, and recursively 'visit an unvisited sub-tree in maze order and *return* to r' as long as there are unvisited sub-trees left at r. (For a discussion of tree traversal orders we refer to [30].)

Maze order traces the entire tree S as a 'maze'. Consecutively visited nodes are always adjacent, and every edge is traversed twice: once in a downward direction ('away from the root') and once upward ('back to the root'). It is immediate that a maze-order traversal takes only linear time. We will use a maze-order traversal of S to compute a good set of disjoint λ-clusters for G as follows.

Assume that at the start, all nodes are colored white. We use the color blue for nodes that are part of an identified λ-cluster, and the color red for nodes that must be left aside when forming clusters. Initialize the process by starting at the root r of S, and trace the maze-order until exactly $\lambda + 1$ different nodes have been visited. The visited nodes must form a λ-cluster. We assign a unique cluster name to the nodes and color them blue. We continue the traversal where we left off, possibly backing up over any blue nodes and necessarily continuing at a white-colored son of some blue node (unless the process is at an end).

This process is continued in principle, identifying yet another λ-cluster and coloring its nodes blue every time precisely $\lambda + 1$ different white nodes have been visited in the continued maze-order traversal. However, if the traversal backs up to a blue node r' before completing a new cluster, an 'exception' occurs. Note that the most recent segment of the traversal must have started at a white son of r' and thus we have effectively completed the traversal of the full subtree of this son of r'. As the nodes in this subtree cannot form a λ-cluster by themselves in S (i.e. given the λ-cluster that were already fixed), we color the nodes in this entire subtree red, leave them aside, and continue the traversal at r'. Note that maze-order traversal will never back up to a red node. The process ends when the traversal returns to the root r of S for the last time. There will be no more white nodes left in S at that time.

Let the clustering process end with λ-clusters $B_1, \ldots, B_l$ (with $l \geq 1$ by the initializing part). By design the clusters are all disjoint and have $\lambda + 1$ nodes each. We claim that it is a good set. To this end, consider any node u that does not belong to any of the clusters B_i $(1 \leq i \leq l)$. Clearly u must have been colored red during the procedure. Thus u must be contained in a fully red subtree, necessarily of size $\leq \lambda$ and attached to a blue node, i.e. to a node of one of the clusters B_i. Thus u has a distance at most λ to at least one of the clusters. It follows that the computed set of λ-clusters is good. The whole procedure to compute the clusters takes only linear time. □

Lemma 3. *Let integers c, d be such that $3 \leq c+1 \leq \min(d, \lambda+1)$. Then the diameter of G can be reduced to at most*

$$4\lambda - 2c + 2c \log_{d-1} \frac{n}{\lambda+1}$$

by the addition of at most $\frac{n}{\lambda+1}$ edges and a degree increase of at most $\delta = \lceil d/(c+1) \rceil$. Moreover, the necessary edges for it can be determined in linear time.

Proof. Let G, λ, c and d be as given. Let $C_1, \ldots, C_k$ be any good set of λ-clusters in G. For each cluster C, fix some connected subset of $c+1$ nodes of C as its *nucleus* $N[C]$.

Consider the clusters $C_1, \ldots, C_k$ and make them into the nodes of a full d-tree T on k nodes. T will have at most k edges and a depth of at most $\log_{d-1} k$. We now embed this tree into G, by adding the extra edges and 'connecting' the clusters the way they are linked in T in the following way. Note that each cluster is incident to at most d 'extra' edges in T. When adding the extra edges to G, we divide the, up to d edges incident to each cluster C evenly over the $c+1$ nodes of its nucleus $N[C]$. Doing this increases the degrees of the nucleus nodes (and only those) by at most $\lceil d/(c+1) \rceil$.

We now estimate the effect of adding the extra edges on the diameter of G. Let u, v be two arbitrary nodes of G. Both u and v have a distance at most λ to any of the clusters, and thus are at a distance of at most $2\lambda - c$ to the nearest node in the nuclei of the respective clusters. The clusters are at a distance of at most $2 \log_{d-1} k$ in T and thus of at most $2c \log_{d-1} k$ in G, using that it takes at most c steps inside every cluster (in fact, over its nucleus) to switch from the 'incoming' node to the node with the 'forward going' edge on the path.

This leads to a diameter of at most $4\lambda - 2c + 2c \log_{d-1} k$ for the augmented graph. The lemma now follows, by substituting $k \leq \frac{n}{\lambda+1}$. By Lemma 2 the clusters $C_1, \ldots, C_k$ can be determined in linear time. The embedding of the edges of T follows in the same linear time-bound. □

3.2. Concrete bounds

Several conclusions can be drawn from the above construction, depending on the value of δ and on the number of extra edges we allow.

We first show how Lemma 3 allows us to extend a result due to Alon et al. [1] for diameter reduction in general. Alon et al. [1] (see also [35]) proved that the diameter of an n-node connected undirected graph can be reduced to (at most) D by adding at most $\frac{n}{\lfloor D/2 \rfloor}$ edges, without taking a degree constraint into account. This can be strengthened as follows.

Theorem 2. *Any connected graph G of n nodes can be shortcut in linear time to a diameter of $O(\frac{D}{\beta})$ by the addition of at most $\frac{n}{\lfloor D/\beta \rfloor}$ edges while keeping degree increases smaller than $n^{\frac{\beta}{D}}$, for any (real) $\beta, D > 0$ such that $D \geq 2\beta$.*

Proof. Take $c = 2$, $d = 1 + \lceil n^{\frac{\beta}{D}} \rceil$ and $\lambda = \lfloor \frac{D}{\beta} \rfloor$ in Lemma 3, and recall that $n > 1$. The conditions on β and D guarantee that the lemma applies, noting that always $n^{\frac{\beta}{D}} > 1$. With this choice of parameters one obtains a diameter bounded by

$$4\lambda - 4 + 4 \cdot \log_{\lceil n^{\frac{\beta}{D}} \rceil} \frac{n}{\lambda+1} < 4\lambda + 4\frac{D}{\beta} \frac{\log_2 \frac{n}{\lambda+1}}{\log_2 n} \leq 8\frac{D}{\beta}$$

and the number of extra edges remains bounded by $\frac{n}{\lfloor D/\beta \rfloor}$. As $\lceil n^{\frac{\beta}{D}} \rceil > 1$, the degree increases remain bounded by $\lceil \frac{1+\lceil n^{\frac{\beta}{D}} \rceil}{3} \rceil < n^{\frac{\beta}{D}}$. □

By a change of parameters one can reformulate the result as follows, in a form that is more readily applied to δ-shortcutting.

Theorem 3. *Any connected graph G of n nodes can be δ-shortcut in linear time to a diameter of $O(\log_{1+\delta} n)$ by adding at most $\frac{n}{\log_{1+\delta} n}$ edges, for any integer $\delta \geq 1$ and provided $\log_{1+\delta} n > 1$.*

Proof. Take $\beta = \frac{1}{2}$ and $D = \log_{1+\delta} n$ in Theorem 2. For a direct proof, take $c = 2$, $d = 2 + \delta$ and $\lambda = \lceil \log_{1+\delta} n \rceil$ in Lemma 3. Note that $(2+\delta)/3 \leq \delta$. This gives the stated bounds. □

By a suitable choice of parameter values one can limit the degree increases in G even to 1, at the expense of a slightly larger bound on the diameter but using fewer extra edges. This improves on Bokhari and Raza's result, as it requires only a *sublinear* number of shortcut edges and takes only linear time to construct them.

Theorem 4. *Any connected graph G of n nodes can be* 1-shortcut *in linear time to a diameter of $O(\delta \log_{1+\delta} n)$ by adding at most $\frac{n}{\delta \log_{1+\delta} n}$ edges, for any integer $\delta \geq 1$ and provided $\log_{1+\delta} n > 1$.*

Proof. Take $c = 1 + \delta$, $d = 2 + \delta$ and $\lambda = \lceil \delta \log_{1+\delta} n \rceil$ in Lemma 3. Note that $\delta < \lambda$, hence $c + 1 \leq \min(d, \lambda + 1)$ and the lemma indeed applies. As $d = c + 1$ we obtain a 1-shortcut of G, with the stated bounds. □

Note that the requirement that $\log_{1+\delta} n > 1$, i.e. $n > 1 + \delta$, in Theorems 3 and 4 is not severe. If it is not satisfied, the diameter of G is already bounded by $1 + \delta$, without having to add any edges at all.

By tuning the parameters properly, Lemma 3 can even give sublogarithmic diameters while still using only a sublinear number of shortcut edges, provided we allow degrees to increase by more than a constant. Let $\gamma(n)$ be any integer function with $2 \le \gamma(n) < \log_2 n$.

Theorem 5. *The diameter of any connected graph G of n nodes can be shortcut to $O(\log_2 n/\log_2 \gamma(n))$ in linear time, by adding at most $n \cdot \frac{\log_2 \gamma(n)}{\log_2 n}$ edges and a degree increase of at most $\gamma(n)$ per node.*

Proof. Let $c = 2$, $d = 1 + \gamma(n)$, and $\lambda = \lceil \frac{\log_2 n}{\log_2 \gamma(n)} \rceil$. Note that $\lambda \ge 2$. By Lemma 3 we obtain a diameter bound of

$$4\lambda - 4 + 4\log_2 n/\log_2 \gamma(n) < 8\frac{\log_2 n}{\log_2 \gamma(n)}$$

while adding a number of extra edges and keeping degree increases bounded as stated. □

For example, take $\gamma(n) = (\log_2 n)^\rho$ for any ρ with $0 < \rho \le 1$. It follows from Theorem 5 that any connected graph G may be shortcut to a diameter of $O(\frac{\log_2 n}{\rho \log_2 \log_2 n})$ by adding only a sublinear number of edges, while keeping degree increases bounded by $(\log_2 n)^\rho$.

3.3. Optimality

The bounds in Section 3.2 can be tuned in various ways, especially when it comes to the constant factors. We list some cases where the bounds are best possible, in order of magnitude.

First of all, we consider the case of a *path*, i.e. a connected graph with n nodes and maximum degree 2. Chung and Garey [16] (see also [46]) proved that in order to reduce the diameter of a path to (at most) D, the number of edges that must be added to achieve this is approximately $\frac{n}{D}$, with no constraint on the degree increase per node. It follows that Theorem 2 is essentially optimal here and so is Lemma 3 in this case, with the parameters used in proving the theorem.

For the general case, consider Theorem 2 or the formulation of the bounds as in Theorem 3. Note that Alon et al. [1] proved their result to be worst-case optimal even for degree-3 trees, without the degree constraint. But then it follows that the degree-constrained extension of the result is worst-case optimal in this case as well, up to constant factors.

Several studies have focused on the minimum diameter achievable by adding some specified number of edges. This problem is computationally hard, even as an approximation problem [18,22,32]. In Section 8 we discuss this problem in detail, including the case in which the degree constraint is taken into account.

4. δ-shortcutting rooted directed paths

The δ-shortcutting problem for directed graphs appears to be difficult. Before we can address the general case in Section 6, we need several auxiliary results that are of interest in their own right. In the present section we consider the shortcutting problem for *rooted directed paths*. In Section 5 we consider the problem for *rooted directed trees*.

4.1. Rooted directed paths

Let $G = \langle V, E\rangle$ with $V = \{v_1, \ldots, v_n\}$ and $E = \{(v_i, v_{i+1}) \mid 1 \le i < n\}$ be a rooted directed path. We view the nodes $v_1, \ldots, v_n$ as being laid out on a line from left to right, with arcs between consecutive nodes directed from left to right.

Yao [46] (see also [7]) showed that by adding $O(n)$ arcs one can reduce the defined distances in a directed path down to $O(\alpha(n))$, where $\alpha(n)$ is the inverse Ackermann function (cf. [41]). The distances can be reduced to $O(\log_2 n)$ by adding only $O(\frac{n}{\log_2 n})$ arcs [7]. However, these results do not keep degree increases constantly bounded.

We show that one can shortcut any n-node rooted directed path to a graph with longest defined distances bounded by $O(\log_2 n)$ by adding only $O(\frac{n}{\log_2 n})$ arcs while degree increases remain bounded by at most 1 per node. In fact, we prove the following, stronger result.

Theorem 6. *All (defined) distances in a rooted directed path G of n nodes can be shortcut to $O(\delta \log_{1+\delta} n)$, by adding at most $\frac{2n}{\delta \log_{1+\delta} n}$ arcs and keeping degree increase limited to at most 1 per node, for any integer $\delta \ge 1$ and provided $\log_{1+\delta} n \ge 1$.*

The proof is divided into three parts. First, we re-organize the layout of G into a special arc structure in Section 4.2. The construction is reminiscent of the proof of Lemma 3 using clusters, now called 'blocks', but is more subtle. The new layout is better suited for shortcutting, as explained in Section 4.3. The proof that the construction achieves the stated bounds follows in Section 4.4. The proof will also show that the entire construction can be done in *linear* time.

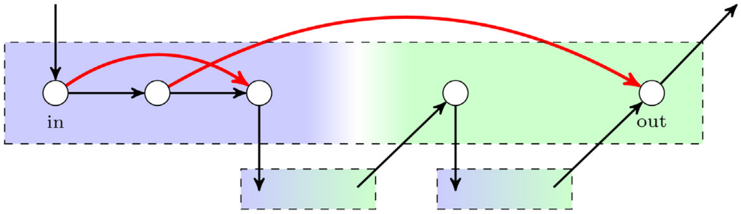

Fig. 1. Design of a block, with its list and switch parts in blue (left) and green (right) respectively. The two red (overarching) arcs are the only shortcut arcs in the block. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.2. Constructing blocks

Let G be a rooted directed path of n nodes. Let d, λ be integers such that $3 \leq d \leq \lambda - 1$. We will later choose specific values for d and λ as needed. Assume that $n \geq \lambda + 1$.

We first re-organize the linear layout of G and group its nodes into $k = \lfloor \frac{n}{\lambda+1} \rfloor$ *blocks* of size $\lambda + 1$ in a special way, as described below. We assume w.l.o.g. that $\lambda + 1$ divides n, otherwise we restrict the construction to the highest multiple of $\lambda + 1$ nodes counted from the beginning of the path and add the remaining, up to λ nodes separately at the end of the construction. This will not affect the bounds.

4.2.1. Blocks

A block of $\lambda + 1$ nodes will consist of a *list part* followed by a *switch part*. In the list part the nodes are connected in a directed path (from left to right), in the switch part nodes are not directly connected to each other but are still viewed as being laid out 'from left to right'. We will determine later how big the list and switch parts of a block have to be. The leftmost node of a block is called its *in-node*, the rightmost one is called its *out-node* (see Fig. 1).

Before describing the blocks further, we outline our overall plan of adding in the arcs that connect the blocks. We symbolically make the k blocks to the 'nodes' of a full d-tree T on k nodes. T itself is undirected and has depth $\log_{d-1} k$. In order to trace out all the blocks and thus all the nodes of G eventually, we follow the consecutive steps of a *maze-order traversal* of T (cf. the proof of Lemma 2).

In this traversal, each block is entered by a 'downward arc' that leads from its father in T to its in-node and, after traversing all its sons (at most d sons in case of the root of T, at most $d - 1$ for an internal node, and 0 for a leaf), the block is exited again over an 'upward arc' from the out-node back to its father.

4.2.2. Arc structure

In order to add the arcs into the structure we are building, we must define 'where' the arcs we just described begin and return, respectively, in the father of a block (which is a block in T itself). This is where the switch part of the blocks comes in. We design the switch part of a block such that it has just the right number of 'free' nodes to accommodate the arcs to and from the sons of the block.

For a block B with s sons in T ($0 \leq s \leq d$) this design looks as follows. The list part of B has $\lambda + 1 - s$ nodes beginning at its in-node, and the switch part has s nodes of which the last one is its out-node (cf. Fig. 1). When the maze-order traversal of T reaches B, the traversal proceeds in the following way:

- B is entered at its in-node,
- the traversal then traces the list part to its end,
- here the downward arc to the in-node of the first son is attached and traversed,
- the block of the first son is traversed and proceeding recursively, arcs are attached in this block and its subtrees,
- returning from this block, the upward arc coming in from the out-node of the block is attached to the first node of B's switch part and followed,
- at this and any subsequent node that is reached in the switch part of B, the downward arc to the next son of B (if any) is attached and traversed, and the return arc is attached to the next node in its switch part and traversed, and so on,
- until the traversal returns from the out-node of the last son of B, in which case the traversal continues over the upward arc from this node to the father of B.

The arc structure at the root of T and at the leaves is obtained by a trivial variant of the description above. Also, if n did not divide $\lambda + 1$ in the beginning, we now add the final segment of $n - (\lambda + 1)k \leq \lambda$ nodes of G that we left off in the construction, back in. We do this by adding an arc from the out-node of the root block to the first node of segment.

Claim 1. *Starting at the in-node of the root block, the traversal of the resulting arc structure precisely traces a rooted directed path of n nodes.*

Proof. All nodes in the created arc structure, except the in- and out-nodes of the root block, have gotten precisely one incoming and one outgoing arc. Starting at the in-node of the root block, the traversal process connects all blocks and traces their nodes completely as in the original path. □

We identify G with the virtual arc structure that is obtained. Observe that, as $\lambda + 1 - d \geq 2$, the list part of each block contains at least 2 nodes.

4.3. Shortcuts

So far we have only re-organized the layout of G. We now add shortcut arcs to G to reduce (defined) node-to-node distances, as follows.

Consider each block B in turn. Add two shortcut arcs to B as follows, if applicable: one arc from B's in-node to the last node of its list part, and one arc from the node adjacent to B's in-node to its out-node (cf. Fig. 1). Thus, one arc shortcuts over the list part, and the other shortcuts over the entire block. If the list part consists of only 2 nodes, we do not add the first arc. If the list part spans the whole block, which happens in leaf nodes, then we do not add the second arc.

Observe that the added arcs are all valid shortcut arcs, as they connect nodes 'in the direction of the path'. Also, the added arcs increase degrees by at most 1 at every node. (In fact, one verifies that at every node $\delta_{in} + \delta_{out} \leq 1$.) We make two claims about the effect of the shortcutting.

Claim 2. *Consider any block B. Starting at its in-node, at the last node of its list part, or at any node of its switch part, all nodes in the switch part of B further to the right in the block are reached by traversing at most $O(d)$ arcs.*

Proof. Consider the different nodes in B. From the in-node, one reaches the last node of the list part in one step by means of the shortcut arc. Suppose we are at this last node of the list part, or at any node further to the right in the switch part that is not the out-node. Call the node u. The downward arc at u brings us to the in-node of the son of B that is attached. By going to the node adjacent to the in-node and traversing the shortcut arc from there, one reaches the out-node of this block and can return to B by traversing the upward arc there, reaching the node to the immediate right of u in B. (If the son at u is a leaf, we follow the shortcut arc from the in-node straight to the out-node in this block.) This takes a total of 4 steps. Hence, all nodes further to the right in the switch part of B are reached in at most $4d$ steps. □

Claim 3. *Let $u, v \in G$ with u occurring before v on the path. Then the distance from u to v in the original graph is shortcut to $O(\lambda + d \log_{d-1} k)$ in the shortcut graph.*

Proof. Let B_u and B_v be the blocks in which u and v occur, respectively. If $B_u = B_v$ we are done, using Claim 2 and the fact that there are at most $\lambda + 1$ nodes in each block. Assuming $B_u \neq B_v$, let B_{lca} be the (block of the) lowest common ancestor of B_u and B_v in T.

Assume w.l.o.g. that $B_u \neq B_{lca}$. Then first go 'up' from B_u to B_{lca} as follows. By traversing at most $\lambda + 4d$ arcs one reaches the out-node of B_u (cf. Claim 2). Following the upward arc from this out-node one reaches the switch part of the father node of B_u. Traversing $O(d)$ more arcs one reaches its out-node and, continuing this way, one eventually reaches the switch part of B_{lca}, say in node u'. All this takes $O(d \log_{d-1} k)$ steps so far. By the linear order of the nodes along the rooted directed path from u to v, two cases can arise:

(a) If B_v is equal to B_{lca}, then either $u' = v$ or v is a node further to the right of u' in the switch part of B_{lca}. By Claim 2, $O(d)$ more arc traversals suffice to reach v and we are done.

(b) If B_v is not equal to B_{lca}, then B_v is located in the sub-tree of a son of B_{lca}. This son of B_{lca} in question must be the son of a node v' in the switch part to the right of u'. Then we move from u' to v' in $O(d)$ steps (cf. Claim 2) and move from v' down the path in T towards B_v, spending $O(d)$ steps in each intermediate block to switch over to the desired subtree in which the downward path continues. Eventually B_v is reached, after at most $O(d \log_{d-1} k)$ steps again.

When B_v is reached we enter it at its in-node. It takes another $\lambda + O(d)$ steps to reach v in B. Adding up the bounds proves the claim. □

4.4. Completing the proof

We now have all the ingredients for the proof of Theorem 6.

Proof. Set $d = 2 + \delta$ and $\lambda = \lfloor 3 + \delta \log_{1+\delta} n \rfloor$. If $n \leq \lambda$, the theorem holds without having to add any arcs at all. Thus, assume $n \geq \lambda + 1$. As we also assumed that $\log_{1+\delta} n \geq 1$, we have $3 \leq d \leq \lambda - 1$ and the given construction applies. With at most two shortcut arcs per block, at most $2\frac{n}{\lambda+1} \leq 2\frac{n}{\delta \log_{1+\delta} n}$ arcs are added in total. In the construction, all degree increases remain bounded by 1. By Claim 3 the inter-node distances in the shortcut graph are all bounded by $O(\delta \log_{1+\delta} n)$. □

Observe that the underlying block structure and thus the shortcut arcs can be determined in the course of a basic linear-time maze-order traversal process. Similar to the remark for Theorem 4, the requirement that $\log_{1+\delta} n \geq 1$, i.e. $n \geq 1 + \delta$, in Theorem 6 is not severe. If the requirement is not satisfied, then the length of G is bounded by δ without having to add any arcs at all.

Corollary 1. *Any rooted directed path G of n nodes can be* 1-shortcut *in linear time so all finite distances are shortcut to $O(\delta \lceil \log_{1+\delta} n \rceil)$ by the addition of at most $\frac{2n}{\delta \log_{1+\delta} n}$ arcs, for any $\delta \geq 1$.*

5. δ-compressing rooted directed trees

Besides the result for rooted directed paths we need another auxiliary result, namely for *δ-compressing* rooted directed trees. This is the problem to shortcut the paths from the root to all other nodes of the tree only, under the usual requirements of shortcutting and such that degree increases remain bounded.

δ-Compression relates to the classical issue in the theory of data structures to reduce the root-to-node or node-to-root distances in trees (cf. [31]). Common approaches like balancing by 'rotating' subtrees do not apply, as we are not allowed to create new nodes or change hierarchical relationships. However, the problem is related to *path compression*, for which highly sublogarithmic bounds on path length can be achieved in almost linear time [41]. Yet, the path collapsing rules used in these methods typically increase node degrees severely, which we do not allow here.

We show the following key result. By varying the parameters d and λ, which may depend on n, one can obtain all sorts of trade-offs between the compression and the number of extra arcs used for it.

Lemma 4. *Let reals d, λ be such that $\lambda \geq 12d$ and $d > 2$. Then any n-node rooted directed tree can be shortcut such that all root-to-node distances are reduced to at most $\lambda + 2\log_d n$ by the addition of at most $12dn/\lambda$ arcs and a degree increase of at most $6d$ at every node, provided $n \geq \lambda + 1$.*

Note that Lemma 4 applies to 'general' rooted directed trees, i.e. without any restriction on out-degrees. The proof is long and detailed. A typical application is obtained by taking $d = 2 + \delta$ and $\lambda = 6\log_{1+\delta} n$, as shown in Theorem 7. In Lemma 6 we show that the degree increases can remain strictly bounded by 1, at the expense of an extra factor of $O(d)$ in depth. The construction is no longer linear time but still quadratic, in general.

5.1. Preliminary remarks

If out-degrees were, for example, bounded by 2, then a version of Lemma 4 could be proved as follows. Let $G = G_r$ be an n-node rooted directed tree with root r and out-degrees ≤ 2. We may assume w.l.o.g. that G is ordered. Thus, let G_{r_1} be the left subtree of r and, if it exists, G_{r_2} the right subtree of r.

In order to compress G, we observe that by the centroid theorem for binary trees (cf. Fact 1) there is an internal node v such the subtree G_v rooted at v has a size between $\frac{1}{3}n$ and $\frac{2}{3}n$. Assume w.l.o.g. that $v \in G_{r_1}$. Now add a shortcut arc from r to v, and subsequently recurse on the *three* binary trees $G_{r_1} \backslash G_v$, G_{r_2}, and G_v. It is easily seen that the process ends after $O(\log_3 n)$ iterations and that G gets compressed to depth $O(\log_3 n)$, with degree increases bounded by 1.

The depth reductions we prove in this section follow the same pattern but are more complex for the following reasons: we do not make any assumption on out-degrees, we want to limit the number of shortcut arcs to a sublinear bound, and we want the result to be tunable with the parameters given in Lemma 4.

The section is organized as follows. In Section 5.2 we give the basic idea behind the lemma and the construction of the clusters we need. In Section 5.3 we give the recursive construction for compressing a rooted directed tree, observing that degrees do not increase by more than a tunable parameter. We also prove that the construction terminates after at most logarithmically many levels have been created. In Section 5.4 we bound the number of extra arcs that are used. In Section 5.5 we combine all ingredients and prove the trade-off lemma. We also show how the construction can be modified to keep all degree increases strictly bounded by 1. Applications follow in later sections.

5.2. Basic steps

Let G be an arbitrary rooted directed tree with n nodes. Let d, λ be (real) values possibly depending on n such that $n \geq \lambda + 1$, $\lambda \geq 12d$ and $d > 2$.

5.2.1. Set-up

Let P be a copy of G. We implement all modifications and shortcuts on P rather than on G, which remains unaltered for the purpose of reference. P will be modified level after level, from the top down. Whenever a next level $i + 1$ is constructed in P, lower subtrees will be 'pulled up' to this level to bring them closer to the root. As a result the composition of the lower levels in P changes as well, as will be explained below.

To facilitate the analysis, we use a simple *color code*. Initially all arcs in P are colored 'green'. The arcs that are not changed remain green. The shortcut arcs that are gradually added to P are colored 'red'. No other colors are used. In the end we are interested in the ultimate depth of P and in the total number of red arcs that were added in the process.

5.2.2. Labels

Whenever a next level of P is constructed, nodes in this level will be labeled A, B, C, or D as given by the following *legend*:

- A: the weight w of the node satisfies $w > \lambda$,
- B: the weight w of the node satisfies $\frac{1}{6d}\lambda \leq w \leq \lambda$,

- C: the weight w of the node satisfies $w < \frac{1}{6d}\lambda$,
- D: an intermediate label (to be explained below).

The weights refer to the weight a node has 'left' in P at any stage. (NB In preceding stages subtrees may have been removed from the node's own subtree in P.) Once a node is labeled, its label will not change. The root node is initialized to label A.

5.2.3. Levels

P is compressed level after level. If level i has been completed and the next level must be constructed, the first step is to divide the nodes below level i into clusters (subtrees) of a suitable size. We will see later how the clusters are selected. Next, the clusters are 'pulled up' by adding a (red) shortcut arc from their ancestor in level i to their root. This effectively elevates the roots to a higher level and brings all nodes in the clusters closer to the root. In fact, it will be useful to pull the roots up to an intermediate level $i + \frac{1}{2}$, so level $i + 1$ consists entirely of the *son* of the nodes that are root of a cluster. The process then continues with level $i + 1$.

The process starts at level 0, containing the root. If level i of P is reached to be processed (i integer), the part of P up to and including this level will have been compressed, and all nodes in it are labeled. Directed paths from the root to nodes in this part of P all obey the hierarchical relationships given by G but benefit from the shortcuts already created. The nodes of level i still have their 'old' subtrees attached as they had them in G except that possibly lower parts (i.e. subtrees) were already cut off from it, pulled up as clusters and attached to a higher level in P during the process. Nodes are assigned to next levels only explicitly in the course of the process.

5.2.4. Clusters

We are now ready to describe the path compression process, starting at level 0. Suppose more generally, that we are to process level i (i integer). In order to prepare for compressing the part below level i, we begin by constructing suitable clusters in the subtrees attached to the nodes in level i.

The compressions at level i depend on a node's label. The subtrees of B- and C-nodes in level i do not have to be compressed further: if a path from the root reaches any of these nodes, then any node in their subtree is reached by an additional λ steps at most (cf. the legend for labels B and C). We therefore leave the subtrees of B- and C-nodes untouched from here on, in particular their nodes will *not* be assigned any (lower) level and thus do not participate in the further process. D-nodes will not occur in integer levels (cf. the description of the levels), and thus we are left with the task of specifying what should happen at the 'heavy' A-nodes. If there are no A-nodes in level i, then the construction *stops* with level i.

Assuming there are A-nodes v in level i, we will decompose their subtree P_v (in P) by means of a tree covering method due to Geary et al. [26,20]. This method covers the vertices of a rooted (ordered) tree by means of *clusters*, i.e. connected subtrees, in such a way that any two clusters are either disjoint or only have their root in common. Any maximal set of clusters joined at a common root in this context will be called a *pinned set of clusters*.

The property we use is the following:

Lemma 5 (*[26]*). *Given any integer $L \geq 2$, a rooted (ordered) tree can be covered with clusters that all have size between L and $3L$, except possibly for one cluster that contains the root which may have size less than L.*

Consider any A-node v in level i. Let P_v be its subtree in P, and let n_v be the size of P_v (i.e. the weight of v). Note that P_v is still unlabeled and has only green arcs.

Definition 2. Let $L_v = \lceil \frac{1}{3d} n_v - 1 \rceil$.

Because v is an A-node, we have $n_v > \lambda \geq 12d$ (by assumption) and thus that $L_v \geq 4$, which is good enough for applying Lemma 5. Cover P_v by a set of clusters, using L_v as the value of L in Lemma 5.

Claim 4. *For any A-node v, the number of clusters x_v in the cover satisfies $d < x_v \leq 1 + 6d$. All clusters have size smaller than $\frac{1}{d} n_v$.*

Proof. The number x_v must be large enough so $x_v \cdot 3L_v \geq n_v$. It follows that $x_v \cdot 3 \cdot \lceil \frac{1}{3d} n_v - 1 \rceil \geq n_v$. If $x_v \leq d$, then

$$x_v \cdot 3 \cdot \left\lceil \frac{1}{3d} n_v - 1 \right\rceil < d \cdot 3 \cdot \frac{1}{3d} n_v = n_v$$

using that $\lceil \beta - 1 \rceil < \beta$ for all β. This is a contradiction. Hence, $x_v > d$.

Next observe that necessarily $1 + (x_v - 1)(L_v - 1) \leq n_v$, accounting for one cluster that can possibly have size less than L_v and excluding the roots from the other clusters. By substituting that $L_v \geq \frac{1}{3d} n_v - 1$, it follows that

$$x_v \leq 1 + \frac{n_v - 1}{L_v - 1} \leq 1 + 3d \cdot \frac{n_v - 1}{n_v - 6d} \leq 1 + 6d$$

using that $n_v > 12d$. The bound on the size of the clusters trivially follows from the value of L_v. □

For later reference we also observe the following bound.

Claim 5. *For all A-nodes v one has $L_v - 1 \geq \frac{1}{6d}\lambda$.*

Proof. For A-nodes v one has $n_v > \lambda$. It follows that

$$L_v - 1 \geq \frac{1}{3d} n_v - 2 \geq \frac{1}{3d}\lambda - 2 = \frac{1 - \frac{6d}{\lambda}}{3d} \cdot \lambda.$$

Because $\lambda \geq 12d$ by assumption, one gets $L_v - 1 \geq \frac{1}{6d}\lambda$. □

In the next subsection we explain how the clusters are used for shortcutting the part of P consisting of the current subtrees below the A-nodes.

5.3. Compression by shortcutting

Assume that there are A-nodes in level i. We now describe the second part of the path compression process, the actual compression of the subtrees attached to the A-nodes in level i.

5.3.1. Construction

By treating all A-nodes in level i as specified below, the levels $i + \frac{1}{2}$ and $i + 1$ are formed. (NB. Recall that we will not touch the B- and C-nodes and their subtrees in level i anymore. In particular, the nodes in their subtrees will not be assigned to any further levels anymore and have basically become invisible from now on.)

Consider any arbitrary A-node v in level i. Assume that P_v is covered by $x_v - 1 > d - 1 > 1$, thus at least 2 clusters which all have size between L_v and $3L_v$ except for possibly one cluster that may have smaller size and that contains the root node v (cf. Lemma 5 and Claim 4). We divide these clusters into three categories and do the following, in the given order.

I: *The clusters not containing v as a root.* These clusters cover the low subtrees of P_v, in the form of (disjoint) pinned sets that are arranged following the structure of P. Detach each of these pinned sets from P_v (by dropping the green arc between their common root node and the father of this node). For each of the pinned sets, do the following:

- Let the clusters of the pinned set we consider have common root v'. Put v' into level $i + \frac{1}{2}$ and add a *red arc* from v to v'. Label v' by D.
- Next we re-position the clusters pinned at v'. Consider the clusters (i.e. subtrees) attached to v' one at a time, and consider the *sons* of v' in any such cluster. (Note that all arcs in the clusters are still green.) Put all these sons of v', with their respective subtrees attached, into level $i + 1$ (keeping them attached to v' by the existing green arcs). Label the sons by A, B, or C according to the legend. Proceed until all clusters attached to v' have been dealt with.

Repeat the above for all pinned sets of clusters in this category. Note that in this way all paths from v to nodes in these clusters get shortcut, and that we add at most $x_v - 1$ red arcs in the process (one for each cluster).

II: *The clusters of size between L and $3L$ that have v as a root.* In this case we proceed as above, except that we do not need to put any node in level $i + \frac{1}{2}$ this time. (Root v is already in level i.) For completeness, here is what we do. Consider the pinned set of clusters of the given size bound, with common root v (if such a set exists). Consider the clusters attached to v one at a time, and consider the sons of v in any such cluster. Put all these sons, with their respective subtrees attached, at level $i+1$ (keeping them attached to v by the existing green arcs). Label the sons by A, B, or C, according to the legend. Proceed until all clusters attached to v have been dealt with.

III: *One cluster of size less than L and which has v as a root.* This case is handled exactly as case II above. (We list this case separately only for the later analysis.)

The construction guarantees that $\delta_{in} \leq 1$ and $\delta_{out} \leq x_v - 1 \leq 6d$ (cf. Claim 4). Red arcs are only added in **I**, leading to nodes labeled D in level $i + \frac{1}{2}$.

Claim 6. *Assigned labels remain consistent with the legend.*

Proof. Once a label is assigned to a node, further stages of the compression can only pull up clusters inside the subtree that remains below the node. In particular, once a label is assigned to a node, its weight no longer changes. □

After level i, the compression process proceeds with level $i + 1$ provided it has A-nodes again. This continues until all levels have been dealt with. From now on, let P denote the *final* tree that is obtained.

5.3.2. Termination

Observe that, in the end, P is a rooted directed tree with paths from the root to all lower nodes, consisting of the paths in G but now including 'jumps' using the red arcs. Hence, the paths are shortcut in a valid way. We show that the number of levels created in the process remains logarithmically bounded.

Claim 7. *Let $i \geq 0$ (i integer) be any level that is created in the compression process. For any node u assigned to level i, $weight(u) \leq n/d^i$.*

Proof. By induction. The statement certainly holds for level $i = 0$, consisting of the root of P. Suppose the statement holds up to and including level i for some integer $i \geq 0$, and let u be added to level $i + 1$. Then u is added because it is the son of a node that is the root of a cluster in P_v. Then by Claim 4:

$$weight(u) \leq \frac{1}{d} n_v \leq n/d^{i+1}.$$

This completes the induction. □

Claim 7 immediately implies that the process we described, must terminate after at most $\log_d n$ integer levels have been created. Also notice that, by design, every path in P from the root to a lowest possible node must eventually reach a node labeled B or C and then run on into the (unmodified) subtree under this node. This leads to the following observation.

Claim 8. *Any path in P from the root to a lower node u in the tree has length at most $\lambda + 2\log_d n$.*

Proof. Consider the path from the root to u. Let $u_0, u_1, \ldots, u_k$ be the nodes of the path on the consecutive integer levels that the path visits, with u_0 the root of P. By Claim 7, and accounting for any intermediate 'half' levels, u_k is at most $2\log_d n$ levels deep. Now the following two cases can occur:

- *u_k has label A*. Then at least one more integer level follows after the level containing u_k, by construction. But as u_k is the last node on an integer level on the path to u, we must either have that $u = u_k$ or that u is a D-labeled son of u_k. Thus the path to u is not longer than $1 + 2\log_d n$.
- *u_k has label B or C*. Then the subtree at u_k is not compressed further. We now have that $u = u_k$ or that u is a node in the (uncompressed) subtree attached to u_k which, by the label of u_k, has depth no more than λ. Thus the path to u is not longer than $\lambda + 2\log_d n$.

Note that u_k cannot have label D (D-labels occur only in 'half' levels). This completes the proof. □

The compression process works on every A-, B-, and C-node a constant number of times. In particular, the construction involves only a linear number of cover constructions and takes $O(n^2)$ time.

5.4. Estimating the number of red arcs

We will now estimate the number of shortcut arcs introduced in constructing P. To this end, we will compact and modify P into a tree Q which has as many red arcs as P has but internal nodes of out-degree at least 2 only and a (potentially) smaller number of leaves as well. By Fact 2, this will enable us to bound the number of arcs in Q, which will thus be a bound on the number of red arcs in particular.

We will first describe how Q is obtained from P, by examining the entire construction of P from the beginning. Q need not be actually constructed but is used only for the purpose of analysis.

5.4.1. Construction

We will trim P by combining and deleting as many subtrees as possible that do not contain a red edge, keeping only the roots of these subtrees and their labels as a measure of the number of nodes they represent. For the resulting tree Q we want the following property to hold:

H: Q is constructed such that it has only integer levels, contains as many red arcs as P, and has internal nodes with label A and leaves with label B only. Moreover, its internal nodes have out-degree ≥ 2.

To achieve it, we set Q to a copy of G, like we did for P. Next, we go through the entire construction of P and modify it, level after level. We maintain the following *inductive assertion* for Q up to level i:

$H(i)$: The nodes at levels j with $j \leq i$ in Q are all labeled A or B, the A-nodes in these levels are the same as in P, the number of red arcs in these levels is the same as in P, and all internal nodes in these levels have degree ≥ 2. Moreover, there are no $\frac{1}{2}$-levels j with $j \leq i$ anymore in Q.

We prove the assertion by simultaneously explaining how a next level of Q is obtained from the corresponding level of P.

Claim 9. *Q can be constructed level after level such that for all integers $i \geq 0$, $H(i)$ holds whenever the construction process has come to the point where level i is going to be processed.*

Proof. We proceed inductively. Obviously $H(0)$ holds, because the root of P and thus of Q, is labeled A and the other conditions are vacuously satisfied. Let $H(i)$ hold for some $i \geq 0$. We show how the elaboration of level i can be arranged such that $H(i+1)$ holds again when level $i+1$ is going to be processed, if at all.

To determine how level i should be processed in Q, consider how levels $i+\frac{1}{2}$ and $i+1$ are constructed for P. We only need to look at how A-nodes are expanded. By the induction hypothesis, the A-nodes in level i are the same in P and Q. If there are no A-nodes in this level, the process stops. Assuming there are, let v be an arbitrary A-node in level i.

The construction starts by determining a cover of P_v, the subtree attached to v in P and thus in Q. By Claim 4 this leads to $x_v \geq d$ clusters, which are subsequently divided into categories **I**, **II**, and **III** and used for the compression at this level. We now show how level $i+\frac{1}{2}$ can be dropped in Q, possibly at the expense of a slight modification of level $i+1$ which can only affect the B- and C-labeled nodes in this level.

Consider the result of the compression process at node v. Define a *group* to be the collection of all sons of the root of any cluster in P_v. Observing the process at v, the compression leads to x_v distinct groups of nodes in level $i+1$. Here a group will be:

- either a set of direct sons of v, arising from a cluster in category **II** or **III**, or
- the set of sons attached to a D-node v' in level $i+\frac{1}{2}$ arising from a cluster in category **I**, where v' is directly linked from v via a red arc.

Recall that all nodes in the groups are labeled A, B, or C according to their weight, as they are assigned to level $i+1$.

We now trim and contract the groups, to obtain a tighter tree than P. To this end we do the following with the nodes in the groups as delineated in level $i+1$, in the given order:

N1: *The A- and B-nodes*. Simply keep these nodes in the level, with their labels. However, delete all subtrees attached to the B-nodes (as we know enough about the size of their subtree by their label alone).

N2: *The C-nodes*. Our aim is to *delete* all of them from this level. However, we have to be careful because we do not want to get rid of too many groups in doing so (if we have many groups of just C-nodes, for example). Consider the x_v groups Z in level $i+1$ in turn, and do the following:

- If Z contains an A- or a B-node, then delete all C-labeled nodes from Z.
- If Z originates from the *one* cluster in category **III** and consists entirely of C-nodes, then delete all of these C-nodes and thus the entire group Z. (Note that this group is linked to directly from A-node v and thus not from any D-node.)
- If Z originates from any cluster in category **I** or **II** and consists entirely of C-nodes then observe that, by the definition of the categories, the group and the subtrees attached to the nodes in it together consist of at least $L_v - 1$ nodes. (NB The root of the cluster is not part of the group.) We proved in Claim 5 that $L_v - 1 \geq \frac{1}{6d}\lambda$. Thus, we can safely do the following: *delete* all nodes of Z and their attached subtrees and *replace the entire group by one new node* that is labeled B and that is attached to the same node (v or a son v' of v in level $i+\frac{1}{2}$ to which the group itself was attached). This sufficiently accounts for the replaced nodes and their subtrees (i.e. as a lower bound, which is all we will need for the counting).

Note that we have deleted at most one group. However, there will be at least $x_v - 1 > d - 1 > 1$ thus at least 2 non-empty groups left, all attached either directly to v or to a D-node v' attached to v. All nodes in the groups are now labeled A or B.

It is a consequence of the above construction that the groups attached to the D-nodes in level $i+\frac{1}{2}$ have all remained present, if only in modified form because we got rid of all C-nodes in them. As a final step we eliminate the D-nodes in level $i+\frac{1}{2}$, and thus the entire level as a level of Q. To this end, we do the following for all D-nodes v' attached to v:

N3: *The D-nodes*. Consider all groups $Z_1, \ldots, Z_k$ attached to v', where necessarily $k \geq 1$ and all groups have their current composition with only A- and B-nodes. Delete node v' and its incident arcs (including the red arc from v to v'), and connect v to all nodes in $Z_1, \ldots, Z_k$ by a *direct* arc. *Color exactly one of these arcs red*, in order to preserve the number of red arcs. (We do not care about the number of green arcs.)

After doing this for all D-nodes, level $i+\frac{1}{2}$ is empty and is deleted. Because at least $x_v - 1 > d - 1 > 1$, thus at least 2 non-empty groups under v remained after **N1** and **N2**, the result of **N3** is that v has degree greater than $d-1$, thus ≥ 2.

Repeating the above for all A-nodes v in level i leads to a level $i+1$ of Q and proves $H(i+1)$. This completes the induction. □

5.4.2. Termination

The given construction of Q terminates like the one for P, when the next level has no A-nodes anymore. By construction, it follows that at this point Q has become a rooted directed tree that exactly satisfies the requirements of property H.

Claim 10. *P contains at most $12d \cdot \frac{n}{\lambda}$ red arcs.*

Proof. By property H, it suffices to count the number of red arcs in Q. Let Q have l leaves. As all leaves are labeled B and their labels represent counts of *disjoint* sets of nodes (of subtrees of the corresponding nodes in P) we must have $l \cdot \frac{1}{6d}\lambda \leq n$, using the lower bound for B-nodes from the legend. Hence, $l \leq 6d \cdot n/\lambda$.

By property H all nodes in Q have out-degree ≥ 2. By Fact 2 it follows that the number of arcs in Q and thus the number of red arcs, is bounded by $2 \cdot l$. With the bound on l, the Claim follows. □

5.5. Concrete bounds

We now have all the ingredients for the proof of Lemma 4 and some concrete bounds for δ-compressing rooted directed trees. We first give the basic version, and then show how one can keep degree increases bounded strictly by 1.

5.5.1. Bounds for δ-compression

The analysis of the given compression process, lead to the following proof of Lemma 4.

Proof. Consider any n-node rooted directed tree G. Construct P as specified in Section 5.3. By adding the red arcs of P to G, the paths from the root to lower nodes in P all become paths in G as well. This effectively compresses G in a valid way.

The bounds on the length of the compressed paths in G and the number of shortcut arcs that were used, follow from the Claims in Sections 5.3 and 5.4. In the construction, the degree increases remained bounded by $\delta_{in} \leq 1$ and $\delta_{out} \leq 6d$. □

A useful application of Lemma 4 is the following result.

Theorem 7. *Any n-node rooted directed tree G can be compressed to a depth of $O(\log_{1+\delta} n)$ by the addition of $O(\frac{\delta n}{\log_{1+\delta} n})$ arcs and a degree increase of at most $O(\delta)$ per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n \geq 2(2+\delta)$.*

Proof. Take $d = 2 + \delta$ and $\lambda = 6 \log_{1+\delta} n$ in Lemma 4, for any $\delta \geq 1$. The constraint on $\log_{1+\delta} n$ guarantees that the requirements in the lemma are satisfied, including that $n \geq \lambda + 1$. □

Various trade-offs between the depth after compression and the number of shortcut arcs used for it may be achieved, by varying the choice of d and λ in Lemma 4.

5.5.2. Bounding degree increases to 1

We now show that a rooted directed tree can be compressed such that all degree increases remain strictly bounded by 1, provided we tolerate a slight increase in depth. The proof is based on a subtle refinement of the construction for Lemma 4.

Lemma 6. *Let reals d, λ be such that $\lambda \geq 12d$ and $d > 2$. Then any n-node rooted directed tree can be shortcut such that all root-to-node distances are reduced to at most $12d \cdot (\lambda + 2 \log_d \frac{n}{12})$ by the addition of at most n/λ arcs and a degree increase of at most 1, provided $n \geq 12d(\lambda + 1)$.*

Proof. Let G be an n-node rooted directed tree, and let $\rho = 12d$. Define a ρ-block to be any rooted directed subtree of G consisting of all nodes that are reachable from the root of the block by a path of length at most ρ. We now proceed in a few steps.

We first divide G top-down into ρ-blocks as follows. Start with the ρ-block at the root of G (assuming it exists). Next, create ρ-blocks in the same way in the subtrees attached to the nodes in level ρ, and continue recursively. If any subtree in this process does not have nodes at depth ρ anymore, we call it *incomplete* and do not proceed with it (as this subtree admits no ρ-block anymore). Consider the tree of ρ-blocks so obtained, and let G_ρ be the condensed graph obtained by contracting the ρ-blocks to 'super-nodes'. G_ρ is a rooted directed tree, with at most $n/\rho = n/12d$ nodes.

Now apply the construction from Lemma 4 to G_ρ, where we note that $\frac{n}{12d} \geq \lambda + 1$ by assumption. This compresses G_ρ to a depth of at most $\lambda + 2 \log_d \frac{n}{12d}$, by the addition of at most $12d \cdot n/12d \cdot \lambda = n/\lambda$ arcs and a degree increase of at most ρ per node. In particular, at most $6d = \rho/2$ red (outgoing) arcs are added to every node of G_ρ, and the in-degrees are increased by at most 1.

Considering that the super-nodes of G_ρ are depth-ρ subtrees of G, we have effectively obtained a compression of G. A path from the root to a node u in G has been shortcut to a path from the root to the ρ-block to which u belongs or, if u belongs to an incomplete subtree, to the root of the ρ-block to which the latter is attached (thus taking ρ extra steps to reach its subtree). It follows that u is reached in at most $\lambda + 2 \log_d \frac{n}{12d} + 2\rho$ steps, using the shortcuts as they stand.

To reduce the current degree increases, we examine the red arcs again which emanate from any internal node v in G, i.e. after the compression. Necessarily v is the root of a ρ-block G_v, which has depth ρ. There are at most $\rho/2$ red arcs leading out of v. These arcs all lead to subtrees located 'below', in the original subtrees of G as they are attached to the leaves of G_v at depth ρ.

Instead of keeping all red arcs attached to v we redistribute them over G_v, making sure that the hierarchical relationships in G are preserved. In particular, if a red arc leads from v to the root of a lower subtree G_w and G_w is part of the subtree (in G) attached to leaf z of G_v, then we can re-attach this red arc to *any* node on the path from v to z (and still linking on to the root of G_w). As all paths from v to a leaf of G_v consist of precisely ρ nodes, it is easily seen that all red arcs attached to v can be re-distributed in this way and such that every node in G_v gets at most one red arc as outgoing edge. (There is enough room on every path from v to a leaf of G_v.)

Redistribute all red arcs as described. The modification leads to a compression of G with $\delta_{in} \leq 1$ and $\delta_{out} \leq 1$. However, for every shortcut path from the root of G to a node u we now have to account for up to ρ extra steps in every cluster on the way. This gives a total depth bounded by

$$\rho \cdot \left(\lambda + 2 \log_d \frac{n}{12d}\right) + 2\rho = 12d \cdot \left(\lambda + 2 \log_d \frac{n}{12d} + 2\right) = 12d \cdot \left(\lambda + 2 \log_d \frac{n}{12}\right)$$

as was to be shown. □

A concrete application of Lemma 6 is the following result.

Theorem 8. *Any n-node rooted directed tree G can be compressed to a depth of* $O(\delta \log_{1+\delta} n)$ *by the addition of at most* $\frac{n}{3\log_{1+\delta} n}$ *arcs and a degree increase of at most* 1 *per node, for any* $\delta \geq 1$ *and provided* $\log_{1+\delta} n \geq 4(2+\delta)$.

Proof. Take $d = 2+\delta$ and $\lambda = 3\log_{1+\delta} n$ in Lemma 6, for any $\delta \geq 1$. The constraint on $\log_{1+\delta} n$ guarantees that the requirements in the lemma are satisfied, including that $n \geq 12d(\lambda+1)$. □

6. δ-shortcutting directed graphs

We now consider δ-shortcutting directed graphs in general. We investigate a number of special cases first. This includes the δ-shortcutting problem for DAGs, rooted directed trees with bounded out-degrees, and strongly connected graphs. The results are combined in the problem for general directed graphs. In Section 7 we approach the general case in a different way.

6.1. Directed acyclic graphs

Let G be a DAG. Recall that the *width* $w = w(G)$ of G is the size of its largest *anti-chain* (cf. Section 2). The width of G is an important measure when shortcutting arbitrary DAGs. Assume w.l.o.g. that $w(G) < n$ in the results below.

Theorem 9. *All finite distances in an n-node DAG G of width w can be shortcut to* $O(\delta w\lceil \log_{1+\delta} n/w\rceil)$*, by adding at most* $\frac{4n}{\delta \log_{1+\delta} n/w}$ *arcs and a degree increase of at most* 1 *per node, for any* $\delta \geq 1$.

Proof. By Dilworth's theorem for DAGs (cf. Fact 3) one can decompose G into w disjoint chains $C_1, \ldots, C_w$ which partition the nodes. Let $n_i = |C_i|$ and assume w.l.o.g. that $n_i \geq 2$ $(1 \leq i \leq w)$. We 1-shortcut each of the chains C_i with $n_i \geq 1+\delta$ by the method underlying Theorem 6.

We now estimate the effect of this shortcut of G. Consider any two nodes u and v of G, and let π be the shortest directed path from u to v in G. Let $C = C_i$ be any of the chains in the decomposition. If π ever intersects C, let x_C be the first node on C that it hits and y_C the last (i.e. after possibly traversing some sections of other chains in between). Necessarily $y_C = x_C$ or y_C lies 'above' x_C on the chain, as G is acyclic. Thus we can replace the entire segment of π from x_C to y_C by the segment from x_C to y_C on this single chain and thus by the shortcut path over C if C was shortcut. It follows that π can be modified to a path from u to v that consists of at most one, possibly shortcut, segment from each chain.

Let there be L chains C_i with $n_i \geq 1+\delta$, for some L with $0 \leq L \leq w$. Assume w.l.o.g. that these chains are $C_1, \ldots, C_L$. By Theorem 6, the length of π is bounded in order of magnitude by

$$\delta(\log_{1+\delta} n_1 + \cdots + \log_{1+\delta} n_L) + (w-L)(1+\delta).$$

This can be estimated by

$$\delta(\log_{1+\delta} n_1 + \cdots + \log_{1+\delta} n_w) + (w-L)(1+\delta) \leq \delta w \log_{1+\delta} n/w + w(1+\delta)$$

using Jensen's inequality for concave functions (cf. Lemma 1) and the fact that $n_1 + \cdots + n_w = n$. The final expression is bounded by $O(\delta w\lceil \log_{1+\delta} n/w\rceil)$. The construction increases degrees in G by at most 1, at the expense of adding a number of arcs bounded by

$$\frac{2}{\delta}\left(\frac{n_1}{\log_{1+\delta} n_1} + \cdots + \frac{n_L}{\log_{1+\delta} n_L}\right).$$

Chains of size less than $1+\delta$ are not shortcut and thus do not contribute to the count. Nevertheless, we can estimate the expression by

$$\frac{2}{\delta}\left(\frac{n_1}{\log_{1+\delta} n_1} + \cdots + \frac{n_w}{\log_{1+\delta} n_w}\right) \leq \frac{4}{\delta} \cdot \frac{n}{\log_{1+\delta} n/w}$$

where the latter bound follows from Lemma 1. □

The construction in the proof clearly takes linear time, except for the initial part of constructing a Dilworth decomposition, which takes polynomial time. In case the width w of G is large, the bound on the number of shortcut arcs in Theorem 9 becomes large. In the next subsection we show how to work around it for rooted directed trees.

6.2. Rooted directed trees

The shortcutting problem for rooted directed trees generalizes that of rooted directed paths studied in Section 4. It also extends the compression problem for rooted directed trees studied in Section 5.

Without any degree constraints, Chazelle [13] (also [44]) proved that n-node (undirected) trees can be shortcut to a diameter of $O(\alpha(m,n))$ by adding m edges. Here $\alpha(m,n)$ is the inverse Ackermann function [41]. To show what can be achieved by δ-shortcutting, we consider rooted directed trees with maximum out-degree $\Delta \geq 2$. (For $\Delta = 1$ one has a rooted directed path and Theorem 6 applies.)

Theorem 10. *All root-to-node distances in an n-node rooted directed tree G with out-degrees bounded by Δ and height $h-1$ can be shortcut to $O(\Delta\delta \log_{1+\delta} h \cdot \ln n)$, by adding at most $4n/(\delta \log_{1+\delta} \log_{1+\delta} h)$ arcs and with a degree increase of at most 1 per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} h > 1 + \delta$.*

Proof. Order G such that at any internal node, the sons are ordered from left to right by *decreasing weight*. (This ordering is similar to the one used in so-called *leftist trees*, cf. [31].)

Call any arc from a father node to its leftmost (thus, heaviest) son a *left-going* arc and all the other arcs *right-going*. Call any node that is either the root of G or reached from its father by a right-going arc, a *head-node*. For any head-node u, let C_u be the chain obtained by starting at u and tracing all the left-going arcs until a leaf node is reached.

Observe that the chains C_u with u ranging over all head-nodes, are all disjoint and together cover all nodes of G (and thus form a decomposition). Consider the chains. Let x and y be two nodes in G and let π be the shortest directed path between them, say leading from x down to y. The path zigzags down G, starting with some segment on a chain, then following a right-going arc to another head-node, and so on. Suppose π visits K distinct chains.

Claim 11. *$K \leq 1 + \Delta \ln n$.*

Proof. Note that, if an internal node of weight m has s sons ($1 \leq s \leq \Delta$), then its left son will have weight $\geq \frac{m}{s}$. Thus the sons reached by a right-going arc will have weight at most $m - \frac{m}{s} \leq m(1 - \frac{1}{\Delta})$.

Assume w.l.o.g. that π begins at a head node. The node has weight at most n. Path π will eventually visit precisely $K-1$ further head nodes. The final one will have weight at most

$$n \cdot \left(1 - \frac{1}{\Delta}\right)^{K-1}.$$

As $n \cdot (1 - \frac{1}{\Delta})^{K-1} \geq 1$, it follows that $K - 1 \leq \Delta \ln n$. □

By assumption all chains have at most h nodes. Now 1-shortcut the chains, by the method from Theorem 6. However, we only shortcut chains if they are longer than $\log_{1+\delta} h$. For a chain with c_i nodes that meets the criteria, this takes $2c_i/(\delta \log_{1+\delta} c_i)$ extra arcs but all defined distances on the chain are reduced to $O(\delta \log_{1+\delta} c_i)$, using that $\log_{1+\delta} c_i \geq \log_{1+\delta} h > 1$.

Claim 12. *The number of shortcut arcs needed is bounded by $4n/(\delta \log_{1+\delta} \log_{1+\delta} h)$.*

Proof. Suppose we 1-shortcut a total of L chains of $c_1, \ldots, c_L$ nodes respectively. Because the chains are disjoint we have $c_1 + \cdots + c_L \leq n$, and by the threshold criterion we have $\frac{1}{L}(c_1 + \cdots + c_L) \geq \log_{1+\delta} h$. Thus, the total number of shortcut arcs is bounded by

$$\frac{2}{\delta}\left(\frac{c_1}{\log_{1+\delta} c_1} + \cdots + \frac{c_L}{\log_{1+\delta} c_L}\right) \leq \frac{4n}{\delta \log_{1+\delta} \log_{1+\delta} h}$$

by applying Lemma 1. □

Finally, we conclude that all chains C_i are either no longer than $\log_{1+\delta} h$ or have been shortcut so inter-node distances over the chain remain bounded by $O(\delta \log_{1+\delta} c_i) = O(\delta \log_{1+\delta} h)$. Hence, after shortcutting over the chains the length of π will remain bounded in the order of

$$K + K \cdot \delta \log_{1+\delta} h = O(K\delta \log_{1+\delta} h)$$

where K is as above. (The first K-term accounts for the right-going arcs on π.) Substituting the bound from Claim 11 gives the result. □

By Corollary 1 the shortcuts in the construction underlying Theorem 10 can be computed in linear time.

6.3. Strongly connected digraphs

When shortcutting strongly connected digraphs, some results are known in case the degree constraint is *not* imposed. For example, Thorup [42] observed that all strongly connected digraphs can be shortcut to a diameter ≤ 4, by at most doubling the number of arcs. Flaxman and Frieze [21] proved that if ϵn random arcs are added to a strongly connected bounded-degree digraph, then the resulting graph has diameter $O(\ln n)$ with high probability.

We will show that $O(\frac{n}{\ln n})$ arcs *always* suffice to shortcut *any* n-node strongly connected digraph to a diameter $O(\ln n)$, while keeping the degree increases bounded by $O(1)$. Before we do this we first prove a more general result, on shortcutting any strongly connected digraph to a diameter $O(\log_{1+\delta} n)$ while keeping degree increases bounded by $O(\delta)$, provided n is large enough in terms of δ. Next we prove a corresponding result in which all degree increases are kept strictly bounded by 2.

Theorem 11. *The diameter of any strongly connected directed graph G can be reduced to $O(\log_{1+\delta} n)$, by the addition of $O(\frac{\delta n}{\log_{1+\delta} n})$ arcs and a degree increase of at most $O(\delta)$ per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n \geq 2(2+\delta)$.*

Proof. Let r be an arbitrary node in G. Because G is strongly connected, there is a rooted directed tree T_{in} with root r and all arcs pointing towards r that spans G. By the same argument, there is a rooted directed tree T_{out} with root r and all arcs pointing away from r that spans G as well.

By Theorem 7, both T_{in} and T_{out} can be compressed to a depth of $O(\log_{1+\delta} n)$ by adding $O(\frac{\delta n}{\log_{1+\delta} n})$ arcs, with a degree increase of at most $O(\delta)$ at every node. (Theorem 7 was only proved for trees with arcs pointing away from r, but the same result holds after reversing the directions of the arcs.) Add the shortcut arcs of T_{in} and T_{out} to G, thus combining the two compressions. This gives a valid shortcutting of G.

Consider any two nodes u and v in G. Then v can be reached from u by following a path over (the compression of) T_{in} from u to r, and then following a path over (the compression of) T_{out} from r to v. The total length of the path is bounded by $O(\log_{1+\delta} n)$. Thus G is shortcut as desired. □

Rooted directed trees as used in Theorem 11 are typically obtained as the side-product of a single-source shortest path algorithm. Together with the shortcutting of the trees, the shortcutting of G can thus be computed in at most $O(n^2)$ time.

Clearly, one can easily 4-shortcut a strongly connected graph G. This can be done by adding the necessary arcs so all nodes get linked in a directed path π as well as in the path π^R with all arcs of π reversed, and applying the result of Theorem 6 to both paths. However, this may take $\Omega(n)$ extra arcs, which is more than we need in the case of Theorem 11. However, we can achieve a better result with fewer shortcut arcs, as follows.

Theorem 12. *The diameter of any strongly connected directed graph G can be reduced to $O(\delta \log_{1+\delta} n)$, by the addition of $O(\frac{n}{\log_{1+\delta} n})$ arcs and a degree increase of at most 2 per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n \geq 4(2+\delta)$.*

Proof. Consider the trees T_{in} and T_{out} as in the proof of Theorem 11 but now 1-shortcut each of them using Theorem 8. This gives the result. □

As a concrete instance of Theorem 12 we obtain the following fact, which relates to the result of Flaxman and Frieze [21] discussed above.

Corollary 2. *The diameter of any strongly connected directed graph G can be reduced to $O(\ln n)$, by the addition of $O(\frac{n}{\ln n})$ arcs and a degree increase of at most 2 per node.*

We return to strongly connected digraphs in Section 7 (cf. Corollary 4).

6.4. General directed graphs

We now consider δ-shortcutting general digraphs. We first follow the classical approach based on condensing the graph, using the results for DAGs and strongly connected graphs obtained above. In Section 7 we follow a different approach, using path covers.

Let G be a directed graph, and let $n_{\min}$ ($n_{\max}$) be the number of nodes in the smallest (resp. largest) strongly connected component of G. Let G_c be the *condensation* of G, i.e. the DAG obtained by 'shrinking' each strongly connected component of G to a single node. Let n_c be the number of nodes in G_c, Δ_c the maximum degree of any node in G_c, d_c its depth, and w_c its width. We assume that $n > n_c > w_c$.

Theorem 13. *The (defined) distances in an arbitrary n-node directed graph G can be shortcut to $O(\delta w_c \cdot \lceil \log_{1+\delta} \frac{n_c}{w_c} \rceil \cdot \log_{1+\delta} \frac{n}{w_c})$ by the addition of $O(\frac{\delta n}{\log_{1+\delta} n/n_c})$ arcs and a degree increase of at most $O(\delta)$ per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n_{\min} \geq 2(2+\delta)$.*

Proof. Let $H_1, \ldots, H_{n_c}$ be the strongly connected components of G. Let the number of nodes of H_i be h_i ($1 \leq i \leq n_c$). Shortcut the components H_i using Theorem 11. Subsequently shortcut G_c, the condensation of G with $H_1, \ldots, H_{n_c}$ as 'super-nodes', using Theorem 9. Add all the shortcut arcs so obtained to G. (If a shortcut arc of G_c connects component H to component H', then the arc is embedded in G by letting it link an arbitrary node of H to an arbitrary node of H'.)

Theorem 11 guarantees that the degree increases in each strongly connected component remain bounded by $O(\delta)$, if we only consider the shortcutting inside the component. By Theorem 9 the shortcutting of G_c increases the degree of at most one node in each component by 1. This bounds the total degree increases in G to $O(\delta)$ per node overall. The total number of arcs added is in the order of:

$$\left(\frac{\delta h_1}{\log_{1+\delta} h_1} + \cdots + \frac{\delta h_{n_c}}{\log_{1+\delta} h_{n_c}}\right) + \frac{n_c}{\delta \log_{1+\delta} n_c/w_c}$$

which by application of Lemma 1 and using that $h_1 + \cdots + h_c = n$ reduce to a bound of

$$\frac{\delta n}{\log_{1+\delta} n/n_c} + \frac{n_c}{\delta \log_{1+\delta} n_c/w_c} = O\left(\frac{\delta n}{\log_{1+\delta} n/n_c}\right).$$

The latter follows because $\frac{\delta n}{\log_{1+\delta} n/n_c}$ is monotone in n and already subsumes the second term when n is close to n_c.

It remains to show that G is adequately shortcut. Consider any two nodes u and v of G and let there be a directed path from u to v. Let strongly connected components H and H' be such that $u \in H$ and $v \in H'$, and assume w.l.o.g. that $H \neq H'$. The path from u to v can be viewed as a path over G_c, beginning in super-node H and ending in super-node H'. The path 'traverses' the intermediate super-nodes by going from an incoming node to an outgoing node inside each component that is visited.

By the shortcutting, Theorem 11 implies that any component of size h can be traversed in only $O(\log_{1+\delta} h)$ steps. Also, by Theorem 9 the path need not visit more than $O(\delta w_c \lceil \log_{1+\delta} \frac{n_c}{w_c} \rceil)$ super-nodes in total. Let the super-nodes on the path from u to v have sizes $h_{i_1}, \ldots, h_{i_s}$ for some $s = O(\delta w_c \lceil \log_{1+\delta} \frac{n_c}{w_c} \rceil)$. It follows that the path can be shortcut to a length the order of:

$$\log_{1+\delta} h_{i_1} + \cdots + \log_{1+\delta} h_{i_s} = O(s \log_{1+\delta} n/s) \leq O\left(\delta w_c \left\lceil \log_{1+\delta} \frac{n_c}{w_c} \right\rceil \cdot \log_{1+\delta} n/w_c\right).$$

The last step follows because $\log_{1+\delta} n/s$ is monotone decreasing in s. This proves that G is shortcut as claimed. □

To estimate the result of Theorem 13 we note that $\frac{n}{w_c} \geq \frac{n}{n_c} \geq n_{\min}$, as $w_c \leq n_c$. Substituting it leads to the following bounds.

Theorem 14. *The (defined) distances in an arbitrary n-node directed graph G can be shortcut to* $O(\delta w_c \cdot \log^2_{1+\delta} n)$ *by the addition of* $O(\frac{\delta n}{\log_{1+\delta} n_{\min}})$ *arcs and a degree increase of at most* $O(\delta)$ *per node, for any* $\delta \geq 1$ *and provided* $\log_{1+\delta} n_{\min} \geq 2(2+\delta)$.

Finally, using Theorem 12 instead of Theorem 11 in the given proof, both Theorems 13 and 14 can be modified such that the degree increases all remain bounded by at most 3, at the expense of an extra factor of $O(\delta)$ in the distance estimates.

7. δ-Shortcutting digraphs using path covers

In this section we show that the path cover number $\mu(G)$ of a directed graph G can be a useful measure in shortcutting G. The approach generalizes the one we already used in Section 6.1 for the special case of DAGs.

We begin by outlining the notion of *feedback dimension* in a digraph. Next we show how digraphs can be effectively 2-shortcut and apply it, for example, to graphs with bounded stability number and feedback dimension. Finally we show how the result can be specialized to various classes of graphs, including graphs that have Hamiltonian cycles, disjoint cycle covers or long paths.

7.1. Feedback dimension

Let G be a directed graph and π a directed path in G. We view π as laid out 'from left to right'. We are interested in the number of segments that can be formed on π when it is intersected by any other path in G. To be more precise, we introduce the following concept.

Definition 3. The *feedback dimension* of π is the largest $k \geq 0$ such that there exist distinct nodes $u_1, v_1, \ldots, u_k, v_k$ 'from left to right' on π that satisfy the following properties (the *feedback base properties*):

- for each i $(1 \leq i \leq k)$ there is a path in G from v_i back to u_i,
- for each i $(1 \leq i \leq k)$ v_i is maximal, that is, there is no path in G from a node on π beyond v_i to a node on π before v_i.

If $u_1, v_1, \ldots, u_k, v_k$ satisfy the feedback base properties on a path π, then we may assume that every u_i is minimal on π as well. For, suppose that some u_i $(1 \leq i \leq k)$ was not minimal. It means there would be a node w on π beyond u_i that has an arc back to a node z before u_i $(1 \leq i \leq k)$. Then we have $u_i < w \leq v_i$ and $v_{i-1} < z$, by the second property. Moving u_i back to z is easily seen to preserve the feedback base properties of the list. By repeating this as long as needed, u_i must become minimal. If $u_1, v_1, \ldots, u_k, v_k$ are such that all u_i are minimal, the sequence of nodes is clearly unique for π.

Lemma 7. *Let $u_1, v_1, \ldots, u_k, v_k$ satisfy the feedback base properties on π. Then for each $i \neq j$ ($1 \leq i, j \leq k$), any walk from v_i back to u_i is node-disjoint from any walk from v_j back to u_j.*

Proof. If not, one could combine the intersecting walks and create a path from v_j to u_i and from v_i to u_j respectively, which is impossible by the feedback base properties. □

Definition 4. The *feedback dimension* $\phi(G)$ of G is the largest k for which G has a path of feedback dimension k.

The feedback dimension of a digraph G is 0 if and only if G is acyclic, and 1 if G is strongly connected. In general we have the following.

Lemma 8. *$\phi(G)$ is the largest k for which there exist k distinct strongly connected components and a path π in G such that each of the k strongly connected components contains at least one arc of π.*

Proof. The lemma clearly holds for $\phi(G) = 0$, i.e. when G is acyclic. We next observe the following, for any directed graph G with $\phi(G) \geq 1$.

(I) Suppose there are k distinct strongly components $C_1, \ldots, C_k$ and a path π in G with the stated property. For each i ($1 \leq i \leq k$), let u_i be the node at which π enters C_i for the first time and v_i the node where π exits C_i for the last time. By strong connectedness, each segment $[u_i, v_i]$ is fully contained in C_i. It follows that the segments $[u_i, v_i]$ ($1 \leq i \leq k$) are disjoint and also, as π and C_i have at least one arc in common and π visits no node more than once, that $u_i \neq v_i$ for each i. Hence, the nodes $u_1, v_1, \ldots, u_k, v_k$ as defined are all distinct. Assume w.l.o.g. that the components $C_1, \ldots, C_k$ were ordered such that nodes $u_1, v_1, \ldots, u_k, v_k$ are ordered 'from left to right' on π. It is easily verified that the nodes satisfy the feedback base properties.

(II) Conversely, consider any directed path π in G and nodes $u_1, v_1, \ldots, u_k, v_k$ ($k \geq 1$) on π that satisfy the feedback base properties. Each segment $[u_i, v_i]$ of π ($1 \leq i \leq k$) consists of at least one arc and must belong to a strongly connected component, say C_i, of G. By the feedback properties, the C_i ($1 \leq i \leq k$) are necessarily all distinct. Thus π has at least one arc in common with k distinct strongly connected components.

We conclude that the feedback dimension is precisely the largest k with the stated property. □

Using Lemma 8 one can show that $\phi(G)$ is polynomial-time computable, by reduction to a longest path problem for DAGs. Let G_c be the condensation of G (cf. Section 6.4), and let $d_c(G)$ be the depth of G_c. The following bound is immediate from Lemma 8.

Corollary 3. *For any digraph G, $\phi(G) \leq d_c(G)$.*

7.2. Directed graphs

We now show how the feedback dimension plays a role when shortcutting a directed graph. The result below generalizes Theorem 9 for DAGs. Because all digraphs we consider must have at least one arc, we have $\mu(G) < n$.

Theorem 15. *The distances in an n-node directed graph G with path cover number $\mu = \mu(G)$ and feedback dimension $\phi = \phi(G)$ can be shortcut to $O(\delta\mu \cdot \lceil \log_{1+\delta} n/\mu \rceil)$ by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n/\mu} + \mu\phi$ arcs and a degree increase of at most 2 per node, for any $\delta \geq 1$.*

Proof. Let G have path cover number $\mu = \mu(G)$. Partition G accordingly into μ node-disjoint directed paths $\pi_1, \ldots, \pi_\mu$. As a first step in shortcutting G, we apply Theorem 6 to 1-shortcut each of these paths that is longer than $1 + \delta$. If there are t of these paths and they consist of $n'_1, \ldots, n'_t$ nodes respectively, and $n'_1 + \cdots + n'_t = n'$, then the number of added shortcut arcs is certainly bounded by

$$\frac{2}{\delta}\left(\frac{n'_1}{\log_{1+\delta} n'_1} + \cdots + \frac{n'_t}{\log_{1+\delta} n'_t}\right) \leq \frac{4}{\delta} \cdot \frac{n'}{\log_{1+\delta} n'/t} \leq \frac{4}{\delta} \cdot \frac{n}{\log_{1+\delta} n/\mu}$$

as in the proof of Theorem 9, using Lemma 1. Let the shortcut arcs that are added in this stage be colored *blue*. Each path π_i is thus shortcut to a length $O(\delta \log_{1+\delta} n_i)$.

To see which extra shortcut arcs we need, consider each path π in the path cover of G. View π as an ordered line of nodes 'from left to right'. Define the function f_π on the nodes of π as follows:

$f_\pi(x)$ = "*the rightmost node $y > x$ on π such that there is a path in G from y back to x, and $\perp$ if no such node y exists*".

Assume that f_π is non-trivial on π, i.e. that $f_\pi(x) \neq \perp$ for at least one $x \in \pi$. Let u_1 be the first node on π for which $f_\pi(x) \neq \perp$ and $v_1 = f_\pi(u_1)$, let u_2 be the first node to the right of v_1 for which $f_\pi(x) \neq \perp$ and $v_2 = f_\pi(u_2)$, and so on. Let u_k, v_k be the last pair on π so constructed.

Claim 13. *For each i ($1 \leq i \leq k$) and for each u with $u_i \leq u < v_i$ on the path we have that $f_\pi(u) = v_i$. Also $f_\pi(v_i) = \perp$.*

Proof. For any i with $1 \le i \le k$, let the path from v_i back to u_i be τ. First, suppose by way of contradiction that $f(v_i) = v \ne \perp$, for some node v to the right of v_i. Then there must be a path τ' from v back to v_i. Let z be the first node where τ' intersects τ. Because τ' certainly intersects τ in v_i, node z is well-defined. Concatenating the segment of τ' from v to z and the segment of τ from z to u, gives us a path from v to u. This contradicts that $f_\pi(u_i) = v_i$. Hence, $f_\pi(v_i) = \perp$.

Next, let u be any node with $u_i \le u < v_i$. We first note that there is path from v_i back to u. (To see this, let z be the first node of τ where τ intersects the segment $[u_i, u]$ on π. By concatenating the segment of τ from v_i to z and the segment from z to u of π, we obtain a path from v_i back to u.) Suppose there was a node v to the right of v_i for which there existed a path from v back to u. Like before one easily argues again that there must be a path from v back to v_i, contradicting that $f_\pi(v_i) = \perp$. Hence, $f_\pi(u) = v_i$. □

It immediately follows from the definition of f_π and Claim 13 that for every non-trivial f_π, the nodes $u_1, v_1, \dots, u_k, v_k$ satisfy the feedback base property on π and thus, that $k \le \phi(G)$.

We can now take the second step in shortcutting G. In this step we consider each path π of the path cover of G. If f_π is non-trivial on π, then construct the nodes $u_1, v_1, \dots, u_k, v_k$ on π and add a shortcut arc from v_i to u_i for each $1 \le i \le k$. This adds at most $\mu\phi$ more shortcut arcs and further increases the in- and out-degrees in G by at most another 1. Let the shortcut arcs added in this stage be colored *yellow*.

Consider the overall effect of the shortcutting we have now achieved. Let x, y be any two nodes in G and, assuming there is a path from x to y, let τ be a shortest possible directed path from x to y in G (i.e. before shortcutting it).

Claim 14. *τ can be replaced by a path τ' from x to y such that for every path π in the path cover, at most one yellow shortcut arc is used and τ' consists of at most two (ordered) segments on π.*

Proof. We show how to construct τ'. Consider any directed path π of the cover. Suppose τ and π intersect. Let u be the first node on τ incident with π and v the last, respectively. If $u = v$, we leave τ unchanged and continue with a next path in the cover. If $u \ne v$, then two possibilities can arise:

- *u precedes v on π.* Then without even using any shortcut arcs, the entire segment from u to v on τ can be replaced by the single segment from u to v over π. Carry out this replacement in τ.
- *v precedes u on π.* Because there now is a path from u back to v, we necessarily have that f_π is non-trivial. Let $u_1, v_1, \dots, u_k, v_k$ be the list of nodes on π as constructed above. It follows that there must be an i ($1 \le i \le k$) such that $v, u \in [u_i, v_i]$. Hence, the segment from u to v on the path τ can be replaced by: the segment from u to v_i over π, followed by the yellow arc from v_i back to u_i and next, the segment from u_i to v over π. Carry out the corresponding replacement in τ.

Continue the construction by subsequently considering all the paths of the cover. It is easily seen that the path τ' that is ultimately obtained, is still a path from x to y but also satisfies the claim. □

Finally, we shortcut τ' further by shortcutting every segment on a path of the cover, using the blue arcs. Let the paths in the cover consist of $n_1, \dots, n_\mu$ nodes, respectively. By Claim 14, the length of τ after all the shortcuttings is certainly bounded in the order of:

$$\mu + 2 \cdot \delta(\log_{1+\delta} n_1 + \cdots \log_{1+\delta} n_\mu) + \mu(1+\delta) \le \mu + 2\delta\mu \log_{1+\delta} n/\mu + \mu(1+\delta) = O(\delta\mu \lceil \log_{1+\delta} n/\mu \rceil).$$

In the estimate we have accounted for the possibility that some of the paths in the cover may not be longer $1+\delta$ and thus, that τ is not shortcut over them by any blue arcs. The initial term μ accounts for the arcs used to switch between consecutively visited paths of the cover. This completes the proof. □

Corollary 4. *The distances in an n-node strongly connected, directed graph G with path cover number $\mu = \mu(G)$ can be shortcut to $O(\delta\mu \cdot \lceil \log_{1+\delta} \frac{n}{\mu} \rceil)$ by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n/\mu} + \mu$ arcs and a degree increase of at most 2 per node, for any $\delta \ge 1$.*

Proof. For strongly connected graphs G we have $\phi(G) = 1$. The result now follows directly by applying Theorem 15. □

Using the Gallai–Milgram theorem (cf. Fact 4) one can reformulate Theorem 15 using the stability number of a graph instead of its path cover number.

Theorem 16. *The (defined) distances in an n-node directed graph G with stability number $\alpha = \alpha(G)$ and feedback dimension $\phi = \phi(G)$ can be shortcut to $O(\delta\alpha \cdot \lceil \log_{1+\delta} \frac{n}{\alpha} \rceil)$ by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n/\alpha} + \alpha\phi$ arcs and a degree increase of at most 2 per node, for any $\delta \ge 1$.*

Proof. By the Gallai–Milgram theorem one has $\mu(G) \le \alpha(G)$. The path cover used in the proof of Theorem 15 is easily transformed into one consisting of $\alpha(G)$ paths, by breaking some of the paths into multiple pieces if necessary. The result now follows by using this cover and α instead of μ in the proof. □

Corollary 5. *The (defined) distances in an n-node directed graph G with stability number $\alpha = \alpha(G)$ and feedback dimension $\phi = \phi(G)$ can be shortcut to $O(\alpha \lceil \log_2 \frac{n}{\alpha} \rceil)$ by adding only $\frac{4n}{\log_2 n/\alpha} + \alpha\phi$ edges and with degree increases of at most 2 per node.*

Theorem 16 and Corollary 5 can both be reformulated again for the case of strongly connected directed graphs, by substituting $\phi(G) = 1$.

Finally, we note that more powerful applications of Theorem 15 may result if one can augment a directed graph by a small number of arcs at every node that preserve transitive relationships but lower the μ- and/or ϕ-value of the graph. We do not digress on this type of 'completion problem' here.

7.3. Using cycles and paths

We now consider a number of useful applications of Theorem 15, to digraphs with known cycle covers or long paths. This leads to several observations for tournaments.

7.3.1. Graphs covered by cycles

The first class of graphs we consider consists of the directed graphs G which can be covered by γ disjoint cycles, for some $\gamma > 1$. There is a sizeable literature on the problem of finding conditions that guarantee that an (un-)directed graph G can be partitioned into a small number of disjoint cycles. Note that strongly connected directed graphs can be covered by at most $\alpha(G)$ directed cycles [5], but in general these cycles need not be disjoint.

Theorem 17. *Let G be any n-node directed graph that can be covered by γ disjoint cycles, for some $\gamma \geq 1$. Then the (defined) distances in G can be shortcut to $O(\delta\gamma \cdot \lceil \log_{1+\delta} \frac{n}{\gamma} \rceil)$ by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n/\gamma} + \gamma$ arcs and a degree increase of at most 2 per node, for any $\delta \geq 1$.*

Proof. Suppose that G can be covered by γ disjoint cycles $D_1, \ldots, D_\gamma$. Now consider the proof of Theorem 15 and use the path cover with paths $\pi_1, \ldots, \pi_\gamma$, where π_i is obtained from D_i by deleting one arc ($1 \leq i \leq \gamma$). Note that for each π_i and nodes $u_1, v_1, \ldots, u_k, v_k$ on π_i constructed in the proof, we must have $k \leq 1$. This follows from the feedback base properties and the fact that π_i can be closed to a cycle, namely D_i. The result now follows from the estimates in the proof of Theorem 15, by substituting γ for μ and 1 for ϕ. □

Theorem 18. *The (defined) distances in any n-node digraph G with a Hamiltonian cycle can be shortcut to $O(\delta \log_{1+\delta} n)$ by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n} + 1$ arcs and a degree increase of at most 2 per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n \geq 1$.*

Proof. Apply Theorem 17 with $\gamma = 1$. The result also follows by noting that graphs G with a Hamiltonian cycle have $\mu(G) = \phi(G) = 1$. In this case, the result also follows immediately from Theorem 15. □

7.3.2. Graphs with long paths

The next class of graphs we consider consists of directed graphs that just possess one or more very long paths. We consider only the case of 'near-Hamiltonian' directed graphs, i.e. digraphs that contain a directed path of length $n - 1 - t$ for some 'small' t with $t \geq 0$. (The case $t = 0$ corresponds to graphs with a Hamiltonian path.)

We first remark that near-Hamiltonian directed graphs are easy to obtain, e.g. by means of a 'forbidden subgraph' construction, as follows. Let $H_{p,q,r}$ be the directed tri-partite graph with node partition $A \cup B \cup C$ such that (i) $|A| = p$, $|B| = q$, and $|C| = r$, (ii) all nodes of A are linked by an arc to all nodes of B, and (iii) all nodes of B linked by an arc to all nodes of C. The following lemma can be seen as extending Proposition 4.7 in [3].

Lemma 9. *Let t be an integer, with $t \geq 0$. Let G be a digraph such that $\overline{G}$ does not contain a subgraph $H_{1,t+1,1}$. Then G has a path of length $\geq n - t - 1$.*

Proof. Assume by way of contradiction that the longest path in G has length $\leq n - t - 2$. Let π be such a path, beginning at node x and ending at node y. Let $A = \{y\}$, B a set of $t + 1$ nodes not contained in π, and $C = \{x\}$. Consider the $H_{p,q,r}$ on $A \cup B \cup C$. If the latter is not a subgraph of $\overline{G}$, then G must have at least one arc from y to some node in B or from some node of B to x. This arc could be used to extend π, contradicting that π was longest. □

Theorem 19. *Let t be an integer, with $t \geq 0$. The (defined) distances in any n-node digraph G with a directed path of length $n-t-1$ and feedback dimension ϕ can be shortcut to $O(\delta \cdot (t+1) \log_{1+\delta} n)$, by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n/(t+1)} + (t+1)\phi$ arcs and a degree increase of at most 2 per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n \geq 1$.*

Proof. Graphs G with a near-Hamiltonian path of length $n - t - 1$ have $\mu(G) \leq t + 1$. The result now follows from Theorem 15. □

Theorem 20. *The (defined) distances in any n-node digraph G with a Hamiltonian path and feedback dimension ϕ can be shortcut to $O(\delta \log_{1+\delta} n)$, by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n} + \phi$ arcs and a degree increase of at most 2 per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n \geq 1$.*

Proof. Apply Theorem 19 with $t = 0$. The result also follows from Theorem 15 by taking $\mu = 1$. □

7.3.3. Tournaments

Shortcutting tournaments is of special interest. We make some general observations, applying the results above.

Corollary 6. *The (defined) distances in any n-node* tournament *with feedback dimension ϕ can be shortcut to $O(\delta \log_{1+\delta} n)$, by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n} + \phi$ arcs and a degree increase of at most 2 per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n \geq 1$.*

Proof. By Rédei's theorem, every tournament has a Hamiltonian path [37]. The result now follows by applying Theorem 20. □

Tournaments contain many Hamiltonian paths [40], thus in the given proof one may want a Hamiltonian path with a small feedback dimension in order to keep the number of shortcut arcs low. For *strongly connected* tournaments, the feedback dimension can be bounded by 1. This follows from Camion's theorem, which asserts that every strongly connected tournament has a Hamiltonian cycle [12]. In this case, both Theorem 18 and Corollary 6 lead to the following result.

Corollary 7. *The (defined) distances in any n-node* stronglyconnected tournament *can be shortcut to $O(\delta \log_{1+\delta} n)$ by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n} + 1$ arcs and a degree increase of at most 2 per node, for any $\delta \geq 1$ and provided $\log_{1+\delta} n \geq 1$.*

8. Complexity of shortcutting

We now consider the difficulty of shortcutting, in the general framework of computational complexity theory (cf. Section 2.2). We aim at fundamental hardness results.

The general problem of reducing the diameter of a network by adding a smallest possible number of edges is well-known to be computationally hard (cf. [10,16,22,24,38]). For shortcutting *without* the degree constraint, the problem in which one is just asked to reduce the diameter by at least 1 by adding at most k edges, is known to be NP-complete and W[2]-hard [22, 24]. We aim to show that these facts hold even *with* the degree constraint imposed, i.e. for the shortcutting problem as we defined it.

We will study the following basic version of the shortcutting problem, which underlies many of the variants we have considered.

> Shortcutting
> *Input*: connected (undirected) graph G, integer $k \geq 1$.
> *Question*: can the diameter of G be shortcut by at least 1 by adding at most k edges while increasing the degrees in G by at most 1?

We will also comment on the *fixed-parameter (in)tractability* of the problem, where the number of shortcut edges allowed is the parameter.

8.1. Reduction

We begin by defining a suitable, polynomial-time computable reduction $\mathcal{R}$ of a problem of known complexity to Shortcutting. The problem we use for this purpose is Hitting Set (cf. Section 2.2), and $\mathcal{R}$ will be designed such that the following property holds.

Lemma 10. *Let $\langle U, \mathcal{S}, k\rangle$ be any instance of* Hitting Set, *with a universe $U = \{u_1, \ldots, u_n\}$, a family of subsets $\mathcal{S} = \{S_1, \ldots, S_m\}$ of U, and an integer k ($1 \leq k \leq n$). Then $\mathcal{R}$ transforms this instance into an instance $\langle G, k\rangle$ of* Shortcutting, *with G consisting of $O(n + m)$ vertices, such that a hitting set of size k for $\mathcal{S}$ can be transformed in polynomial time into a set of k shortcut edges that reduce the diameter of G by at least 1 while increasing the degrees in G by at most 1, and vice versa.*

Proof. Design $\mathcal{R}$ as follows. Let $\langle U, \mathcal{S}, k\rangle$ be any instance of Hitting Set as given. Let $\mathcal{R}$ transform this instance into an instance $\langle G, k\rangle$, where $G = \langle V, E\rangle$ is the 5-layered graph with

$$V = \{S_1, \ldots, S_m\} \cup \{u_1, \ldots, u_n\} \cup \{a\} \cup \{b_1, \ldots, b_n\} \cup \{c_1, \ldots, c_{2n+1}\}$$

and with the following edges:

- an edge between u_i and S_j for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$ such that $u_i \in S_j$ (i.e. the edges representing the element-set relation),
- an edge between u_i and $u_{i'}$ for all i, i' with $1 \leq i < i' \leq n$ (i.e. the nodes u_i form a clique),
- an edge between a and each u_i ($i = 1, \ldots, n$),
- an edge between a and each b_i ($i = 1, \ldots, n$), and
- an edge between b_i and $c_{i'}$ for all $i = 1, \ldots, n$ and $i' = 1, \ldots, 2n + 1$ (i.e. the nodes b_i and $c_{i'}$ induce a complete bipartite graph).

No other edges except those specified above are present in E. Observe that graph G is connected and that it has $m + 4n + 2$ nodes and diameter 4, where the longest distances in G are realized (only) between the nodes S_j and $c_{i'}$, for all $j = 1, \dots, m$ and $i' = 1, \dots, 2n + 1$.

Clearly $\mathcal{R}$ is polynomial-time computable. We now show that it also provides an effective translation between a hitting set of size k in $\langle U, \mathcal{S}\rangle$ and a set of at most k shortcut edges that reduce the diameter of G by at least 1 without increasing the degrees in G by more than 1.

(I) Constructing a shortcutting of G from a hitting set for $\mathcal{S}$. Suppose that $(U, \mathcal{S})$ has a hitting set $H = \{u_{h_1}, \dots, u_{h_k}\}$ of size k, with $1 \le k \le n$. Shortcut G by adding an edge from node b_i to node u_{h_i}, for each $i = 1, \dots, k$. Clearly we add no more than k edges to G and increase the degrees in G by at most 1. We are left with proving that the added edges indeed reduce the diameter of G.

Denote the resulting graph by G'. To show that G' has diameter less than 4, it is sufficient to show that the distances between the S_j and the $c_{i'}$ are reduced (cf. the above observation). Indeed, let $j \in \{1, \dots, m\}$ and $i' \in \{1, \dots, 2n + 1\}$, and let $i \in \{1, \dots, k\}$ be such that $u_{h_i} \in S_j$. (Note that i exists because H is a hitting set.) Then the path from $c_{i'}$ to b_i to u_{h_i} to S_j exists in G' and has length 3. Therefore, the diameter of G' is indeed 1 less than the diameter of G and G has been validly shortcut.

(II) Constructing a hitting set for $\mathcal{S}$ *from a shortcutting of G*. Conversely, suppose that there is a set F of at most k edges $(1 \le k \le n)$ that reduce the diameter of G by at least 1 when added to G while increasing the degrees in G by at most 1. (We actually do not need the latter assumption in the rest of the proof.) Let G' be the graph obtained from G by adding the edges of F. Because $k \le n$, there is a node $c_{i'}$ for some $i' \in \{1, \dots, 2n+1\}$ that is not an endpoint of any edge in F. For $j = 1, \dots, m$, let P_j be any shortest path in G' between $c_{i'}$ and S_j, and let e_j denote the last edge of P_j (i.e. the one that is incident on S_j).

Now construct a set H as follows. For $j = 1, \dots, m$, if $e_j \in F$, then we add any element of S_j to H; otherwise, e_j is an edge (of G) between S_j and u_i for some $i \in \{1, \dots, n\}$, and we add u_i to H. By construction, H is a hitting set for $(U, \mathcal{S})$.

Claim 15. *H has size at most k.*

Proof. As a basic observation we note that for each $j \in \{1, \dots, m\}$, path P_j contains at most three edges (as the diameter of G' is at most three by the definition of F) and that its first edge must be an edge of G, by the definition of $c_{i'}$.

We first argue that all e_j $(1 \le j \le m)$ are distinct. After all, if $e_j = e_{j'}$ for distinct $j, j' \in \{1, \dots, m\}$, then this edge must be an edge between S_j and $S_{j'}$, and thus P_j and $P_{j'}$ contain exactly three edges. Furthermore, the middle edge of $P_{j'}$ must be an edge $f \in F$ from b_i to S_j for some $i \in \{1, \dots, n\}$. But then $c_{i'}$ to b_i to S_j is a path from $c_{i'}$ to S_j of two edges, contradicting that P_j is a shortest such path. Therefore, the e_j are distinct.

From the basic observation it also follows that, if $e_j \notin F$ for some $j \in \{1, \dots, m\}$, then the edge f_j of P_j that precedes e_j must be in F (as no path in G between $c_{i'}$ and S_j had at most three edges). In particular, f_j is incident to the element that we added to H for j and to a b_i node for some $i \in \{1, \dots, n\}$, and thus not equal to $e_{j'}$ for any $j' \in \{1, \dots, m\}$. Therefore, each element in H can be matched to a unique edge from F. Hence $|H| \le |F| = k$. □

Finally, we can add some arbitrary elements of the universe to H to make it have size exactly k.

To complete the proof, we only need to observe that both transformations described above are easily seen to be polynomial-time computable. □

8.2. Consequences

Using Lemma 10 we can obtain several conclusions for the complexity of Shortcutting, refining and improving on the results for diameter reduction in general (cf. [10,22]).

Theorem 21. Shortcutting *is strongly NP-complete and W[2]-hard.*

Proof. It is easily argued that Shortcutting belongs to NP. By Lemma 10 the transformation $\mathcal{R}$ designed in Section 8.1 provides a polynomial-time computable reduction from Hitting Set to Shortcutting. Moreover, because $\mathcal{R}$ does not involve any numeric parameters that are not polynomially bounded and preserves the parameter k, it satisfies both the requirements of a pseudo-polynomial transformation [25] and that of a (standard) parameterized polynomial-time reduction [19]. Hence, because Hitting Set is strongly NP-complete and W[2]-hard, Shortcutting is NP-complete and W[2]-hard as well. □

Theorem 21 implies that Shortcutting is not fixed-parameter tractable, unless FPT = W[2] (cf. Section 2.2 or [19]). In case one looks for ways to circumvent the computational hardness of Shortcutting by approximation, the following conclusion is important as well.

Corollary 8. Shortcutting *does not have an efficient polynomial time approximation scheme (EPTAS), unless FPT* $= W[2]$.

Proof. This follows from Bazgan's theorem ([33], Prop. 6.1, p. 70) which asserts that, if an optimization problem admits an EPTAS, then the standard parameterized version is fixed-parameter tractable. The corollary now follows from the preceding remarks. □

Further facts for the (in-)approximability of Shortcutting may be derived from the connection to Hitting Set and by duality, to Set Cover for which strong results are known (cf. Section 2.2).

The results in this section hold for directed graphs as well. This follows by modifying Lemma 10 to the case of DAGs. The construction uses a similar 5-layered graph G but now the edges between the u-nodes are omitted and all remaining edges are directed 'upwards', from the c-nodes to the b nodes, from the b-nodes to the a-node and so on. The correctness proof simplifies because shortcut arcs must obey the transitive relationships.

9. Conclusions

The δ-shortcutting problem for directed and undirected graphs is motivated by the practical concern of reducing distances in networks while adding only a bounded number of extra links per node. Our analysis has shown that the problem leads to many classical issues in graph decompositions and coverings with core structures like paths and cycles.

We proved that all undirected graphs as well as all strongly connected directed graphs can be shortcut to a logarithmic diameter, by adding at most a sublinear number of edges while keeping degree increases bounded by a constant. For general directed graphs we have proved similar results, depending on parameters like the width of their condensed graph or their stability number. Although the diameter bounds seem tight in several cases, they can most likely be tuned or improved further.

For example, if the precise bounds for δ-compressing rooted directed trees (Theorems 7 and 8) can be improved, then the results for strongly connected digraphs (Theorem 11) and for general digraphs (Theorem 13) can automatically be improved as well. This may well be the case when attention is restricted to special, e.g. bounded-degree graphs. Of course the strength of the present results is that they do not rely on any such restrictions.

Many interesting problems remain for further study. For example, Thorup [43] showed that by adding at most n edges, the distances in a rooted directed planar graph can be reduced to $O(\log n)$. Does this result remain valid if degree increases must remain bounded by a constant? Theorem 15 indicates that degree increases of even 1 or 2 per node enable substantial shortcuts in general digraphs. Can better bounds be achieved when higher degree increases are allowed?

Finally, we have proved only the most essential facts for the parameterized complexity of δ-shortcutting. There are many further questions here, notably about the minimum diameters achievable by adding some specified number of edges (cf. [18,22,32]). The 'δ-perspective' seems of considerable interest for further study.

Acknowledgments

We thank the referees for several useful comments and suggestions.

References

[1] N. Alon, A. Gyárfás, M. Ruszinkó, Decreasing the diameter of bounded degree graphs, J. Graph Theory 35 (3) (2000) 161–172.
[2] G. Ausiello, et al., Complexity and Approximation - Combinatorial Optimization Problems and their Approximability Properties, Springer-Verlag, Berlin, 1999.
[3] J. Balogh, J. Barát, D. Gerbner, A. Gyárfás, G. Särközy, Partitioning 2-edge-colored graphs by monochromatic paths and cycles, Combinatorica 34 (5) (2014) 507–526.
[4] I. Ben-Arroyo Hartman, Variations on the Gallai-Milgram theorem, Discrete Math. 71 (1988) 95–105.
[5] S. Bessy, S. Thomassé, Spanning a strong digraph by α circuits: A proof of Gallai's conjecture, Combinatorica 27 (6) (2007) 659–667.
[6] D. Bilò, L. Gualà, G. Proietti, Improved approximability and non-approximability results for graph diameter decreasing problems, Theoret. Comput. Sci. 417 (2012) 12–22.
[7] H.L. Bodlaender, G. Tel, N. Santoro, Trade-offs in non-reversing diameter, Nordic J. Comput. 1 (1) (1994) 111–134.
[8] S.H. Bokhari, A.D. Raza, Reducing the diameters of computer networks, IEEE Trans. Comput. C-35 (8) (1986) 757–761.
[9] M.L. Bonet, S.R. Buss, The serial transitive closure problem for trees, SIAM J. Comput. 24 (1) (1995) 109–122.
[10] R. Brauer, G. D'Angelo, D. Delling, A. Schlumm, D. Wagner, The shortcut problem - complexity and algorithms, J. Graph Algorithms Appl. 16 (2) (2012) 447–481.
[11] K. Cameron, An algorithmic note on the Gallai-Milgram theorem, Networks 20 (1) (1990) 43–48.
[12] P. Camion, Chemins et circuits hamiltoniens de graphes complets, C. R. Acad. Sci. Paris 249 (1959) 2151–2152.
[13] B. Chazelle, Computing on a free tree via complexity preserving mappings, Algorithmica 2 (3) (1987) 337–361.
[14] C.C. Chen, P. Manalastras Jr., Every finite strongly connected digraph with stability 2 has a Hamilton path, Discrete Math. 44 (1983) 243–250.
[15] F.R.K. Chung, Diameters of graphs: old problems and new results, Congr. Numer. 60 (1987) 295–317.
[16] F.R.K. Chung, M.R. Garey, Diameter bounds for altered graphs, J. Graph Theory 8 (1984) 511–534.
[17] E.D. Demaine, M. Zadimoghaddam, Minimizing the diameter of a network using shortcut edges, in: H. Kaplan (Ed.), Algorithm Theory - SWAT 2010, Proc. 12th Scandinavian Symposium and Workshops, in: Lecture Notes in Computer Science, vol. 6139, Springer-Verlag, Berlin, 2010, pp. 420–431.
[18] Y. Dodis, S. Khanna, Designing networks with bounded pairwise distance, in: 31st Ann. Symposium on Theory of Computing (STOC'99), Proceedings, ACM Press, USA, 1999, pp. 750–759.
[19] R.G. Downey, M.R. Fellows, Fundamentals of Parameterized Complexity, Springer, London, 2013.
[20] A. Farzan, J.I. Munro, A uniform approach towards succinct representation of trees, in: J. Gudmundsson (Ed.), Algorithm Theory - SWAT 2008, Proc. 11th Scandinavian Workshop, in: Lecture Notes in Computer Science, vol. 5124, Springer-Verlag, Berlin, 2008, pp. 173–184.
[21] A.D. Flaxman, A.M. Frieze, The diameter of randomly perturbed digraphs and some applications, Random Struct. Algorithms 30 (4) (2007) 484–504.
[22] F. Frati, S. Gaspers, J. Gudmundsson, L. Mathieson, Augmenting graphs to minimize the diameter, in: L. Cai, et al. (Eds.), Algorithms and Computation (ISAAC 2013), Proc. 24th International Symposium, in: Lecture Notes in Computer Science, vol. 8283, Springer-Verlag, Berlin, 2013, pp. 383–393.
[23] T. Gallai, A.N. Milgram, Verallgemeinerung eines graphentheoretischen Satzes von Rédei, Acta Sci. Math. (Szeged) 21 (1960) 181–186.
[24] Y. Gao, D.R. Hare, J. Nastos, The parametric complexity of graph diameter augmentation, Discrete Appl. Math. 161 (2013) 1626–1631.
[25] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman & Co., San Francisco, 1979.
[26] R.F. Geary, R. Raman, V. Raman, Succinct ordinal trees with level-ancestor queries, ACM Trans. Algorithms 2 (4) (2006) 510–534.

[27] W. Hesse, Directed graphs requiring large numbers of shortcuts, in: SIAM-ACM Symposium on Discrete Algorithms (SODA'03), Proceedings, SIAM, Philadelphia, 2003, pp. 665–669.
[28] J.L.W.V. Jensen, Sur les fonctions convexes et les inégalités entre les valeurs moyennes, Acta Math. 30 (1906) 175–193.
[29] J. Kleinberg, E. Tardos, Algorithm Design, Addison-Wesley, Boston, 2006.
[30] D.E. Knuth, The Art of Computer Programming, Vol. 1: Fundamental algorithms, third ed., Addison-Wesley, Reading, MA, 1997.
[31] D.E. Knuth, The Art of Computer Programming, Vol. 3: Sorting and Searching, second ed., Addison-Wesley, Reading, MA, 1998.
[32] C.-L. Li, S.T. McCormick, D. Simchi-Levi, On the minimum-cardinality-bounded-diameter and the bounded-cardinality-minimum-diameter edge addition problems, Oper. Res. Lett. 11 (1992) 303–308.
[33] D. Marx, Parameterized complexity and approximation algorithms, Comput. J. 51 (1) (2008) 60–78.
[34] A. Meyerson, B. Tagika, Minimizing average shortest path distances via shortcut edge addition, in: I. Dinur, et al. (Eds.), APPROX'09/RANDOM'09, in: Lecture Notes in Computer Science, vol. 5687, Springer-Verlag, Berlin, 2009, pp. 272–285.
[35] M. Parnas, D. Ron, Testing the diameter of graphs, Random Struct. Algorithms 20 (2) (2002) 165–183.
[36] S. Raskhodnikova, Transitive-closure spanners: a survey, in: O. Goldreich (Ed.), Property Testing - Current Research and Surveys, in: Lecture Notes in Computer Science, vol. 6390, Springer-Verlag, Berlin, 2010, pp. 176–196.
[37] L. Rédei, Ein kombinatorischer Satz, Acta Litt. Szeged 7 (1934) 39–43.
[38] A.A. Schoone, H.L. Bodlaender, J. van Leeuwen, Diameter increase caused by edge deletion, J. Graph Theory 11 (3) (1987) 409–427.
[39] A. Schrijver, Combinatorial Optimization: Polyhedra and Efficiency, Vol. A: Paths, Flows, Matchings, Springer, Berlin, 2003, (Chapters 1-38).
[40] T. Szele, Kombinatorische Untersuchungen über den gerichteten vollständigen Graphen, Mat. Fiz. Lapok 50 (1943) 223–256. (German translation: Publ. Math. Debrecen 13 (1966) 145–168).
[41] R.E. Tarjan, Efficiency of a good but not linear set union algorithm, J. ACM 22 (2) (1975) 215–225.
[42] M. Thorup, On shortcutting digraphs, in: E.W. Mayr (Ed.), Graph-Theoretic Concepts in Computer Science (WG'92), Proceedings, in: Lecture Notes in Computer Science, vol. 657, Springer-Verlag, 1993, pp. 205–211.
[43] M. Thorup, Shortcutting planar digraphs, Combin. Probab. Comput. 4 (1995) 287–315.
[44] M. Thorup, Parallel shortcutting of rooted trees, J. Algorithms 23 (1) (1997) 139–159.
[45] E.J. van Leeuwen, J. van Leeuwen, Structure of polynomial-time approximation, Theory Comput. Syst. 50 (4) (2012) 641–674.
[46] A.C. Yao, Space–time tradeoff for answering range queries (Extended abstract), in: ACM Symposium on Theory of Computing, (STOC'82), ACM Press, New York, 1982, pp. 128–136.