Shortcutting Directed and Undirected Networks with a Degree Constraint

Richard B. Tan

Erik Jan van Leeuwen

Jan van Leeuwen

Technical Report UU-CS-2014-020
August 2014

Department of Information and Computing Sciences
Utrecht University, Utrecht, The Netherlands
www.cs.uu.nl

ISSN: 0924-3275

Department of Information and Computing Sciences
Utrecht University
P.O. Box 80.089
3508 TB Utrecht
The Netherlands

Shortcutting Directed and Undirected Networks with a Degree Constraint*

Richard B. Tan[1] Erik Jan van Leeuwen[2] Jan van Leeuwen[3]

[1] Dept. of Computer Science, University of Sciences & Arts of Oklahoma
Chickasha, OK 73018, USA.
B.T.Tan@uu.nl

[2] Dept. of Algorithms and Complexity, Max-Planck Institut für Informatik
D-66123 Saarbrücken, Germany.
erikjan@mpi-inf.mpg.de

[3] Dept. of Information and Computing Sciences, Utrecht University
Princetonplein 5, 3584 CC Utrecht, The Netherlands.
j.vanleeuwen1@uu.nl

Abstract. Shortcutting is the operation of adding edges to a network with the intent to decrease its diameter. We are interested in shortcutting graphs while keeping degree increases 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 the addition of no more than $O(n/\log_{1+\delta} n)$ edges and degree increases bounded by δ. The result also improves on an estimate due to Alon *et al.* Degree increases can be limited to 1 at a small extra cost. For strongly connected, bounded-degree directed graphs Flaxman and Frieze proved that, if ϵn random edges 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 degree increases bounded by $O(1)$ per node. The result is proved in a stronger, parametrized form. For general directed graphs with stability number α, we show that all distances can be reduced to $O(\alpha\lceil\ln \frac{n}{\alpha}\rceil)$ by adding only $\frac{4n}{\ln n/\alpha} + \alpha\phi$ edges and degree increases of at most $O(1)$ per node, with ϕ equal to the so-called feedback-dimension of the graph. Finally, we prove bounds for special classes of graphs, including e.g. graphs with Hamiltonian cycles or paths. Shortcutting with a degree constraint is proved to be NP-complete and $W[2]$-hard, and is shown not to have a polynomial-time $(1-\epsilon)\ln N$-approximation algorithm for any $\epsilon > 0$, unless NP $\subseteq$ DTIME$(N^{\log\log N})$.

Keywords: Networks, Graphs, Rooted Directed Trees, DAGs, Diameter, Compression, Shortcutting, Feedback Dimension, Path Covers, Stability Number, $W[2]$-hardness.

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 and 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.

* Version dated August 8, 2014.

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 diameter and/or inter-node distances are achievable by shortcutting G and how many extra edges are needed for it, allowing a degree increase of at most a small fixed amount at every node.*

Most prior research on the shortcutting problem has *not* taken any degree constraint into account. The general study of the shortcutting problem 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 ([38, 16, 25]). Also the complexity of approximating the number of edges needed to achieve a certain diameter decrease has been studied ([17, 34, 6]). 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 [28] showed that this fact does not hold for digraphs in general. Finally, a shortcut graph is a special instance of a so-called *transitive-closure spanner* of the graph, which have been studied from a different perspective (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 more generally and in more detail and aim at proving sharp bounds on the reductions in diameter or inter-node distance that can be achieved. In the sequel, when we speak of shortcutting we will always mean δ-shortcutting.

Results We consider the shortcutting problem for the undirected and directed cases separately. We first consider the case of *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 (take $\Delta \geq 2+\delta$ in the above bound).

For the case $\delta = 1$, the δ-shortcutting problem was first studied 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 computed 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 bounds are in fact an instance of a more general result. Note that Alon *et al.* [1] (see also [35]) already showed that any connected undirected graph G can be reduced to diameter D by adding at most $\frac{n}{\lfloor D/2 \rfloor}$ edges, *without* taking

Graph class	Diameter	$\sharp$ 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*	$O(\log_{1+\delta} n)$	$O(\frac{\delta n}{\log_{1+\delta} n})$	$O(\delta)$	7
	$O(\delta \log_{1+\delta} n)$	$\frac{n}{3 \log_{1+\delta} n}$	1	8
Rooted directed tree	$O(\varDelta\delta \lceil \log_{1+\delta} h \rceil \log_2 n)$	$\frac{4n}{\delta \log_{1+\delta} \log_{1+\delta} h}$	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	$O(\log_{1+\delta} n)$	$\frac{4n}{\delta \log_{1+\delta} n} + 1$	2	18

Table 1: *Main results on shortcutting and compression. (Compression is marked by* **.) Column 'Delta' gives the bound on the degree increases. For notation and details, see the theorem cited in the last column.*

the degree constraint into account. We extend this result to the case where the degree constraint is taken into account. 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 in our notation, the result asserts that for any integer $\delta \geq 1$, any connected undirected graph can be δ-shortcut to a diameter $O(\log_{1+\delta} n)$ by adding at most $O(\frac{n}{\log_{1+\delta} n})$ extra edges. The extra edges can dtermined in linear time. 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 added edges. As Alon *et al.* [1] proved their result to be worst-case optimal, even for degree-3 trees, so is our degree-constrained extension of it. We show this in Section 3.

After proving several preliminary results on δ-shortcutting rooted directed paths and δ-compressing rooted directed trees in Sections 4 and 5, we turn to the case of (general) *directed* graphs in Section 6. We first prove several 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 [22] proved for the bounded-degree case that, if ϵn random edges 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 $O(\frac{n}{\ln n})$ edges and degree increases bounded by 2. The bounds follow from a more general result that can be tuned a various ways (cf. Table 1).

For general directed graphs we prove several bounds in Sections 6 and 7. In particular, in Section 7 we show that all distances in any 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 edges 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 (see Section 2). We also show that every tournament can be 2-shortcut to diameter $O(\ln n)$, by adding at most a linear number of arcs. An overview of the main results achieved in this paper is shown in Table 1.

In Section 8 we consider the complexity of minimal shortcutting. It is well-known that *without* the degree constraint, the problem of deciding whether the addition of a certain number of edges can reduce diameter is computationally hard. We argue that even in a simple form, the problem *with* the degree constraint remains both 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 and not even fixed-parameter tractable, with the number of extra edges as the parameter. We also prove that shortcutting with a degree constraint is not $(1-\epsilon)\ln N$-approximable by means of a polynomial-time algorithm for any $\epsilon > 0$, unless NP $\subseteq$ DTIME$(N^{\log\log N})$. This is discussed further in Section 8. In Section 9 we give some conclusions and mention a few problems for further work.

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 also be called arcs in the directed case. We do not allow self-loops. We let $n = |V|$, and assume that $n > 1$ throughout. The complement of G is the graph $\overline{G} = \langle V, \overline{E}\rangle$, where $\overline{E}$ is the complement of E in $V \times V$.

The *degree* of a node v is the number of edges incident to v. The degree Δ of a graph G is the maximum of the degrees over all nodes in V. For directed graphs we distinguish between the *in-degree* and the *out-degree* of nodes. 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 $v_1, e_2, v_2, \cdots, e_k, v_k$ of nodes and edges (arcs), 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 arc e_i points from v_{i-1} to v_i.) In walks, it is also assumed that no edge (arc) appears more than once. A *path* in G 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 $V = \{v_1, \cdots, v_n\}$ 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 edges 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 *weight* of a node is the number of nodes in its subtree (itself included). The *depth* of a tree is the longest distance from its root to a leaf. 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.*[1]

[1] Let T be a rooted (directed) tree with n nodes and out-degrees ≤ 2. Let u be the lowest node in T that has $weight(u) > \frac{2}{3}n$. Node u exists and is unique. If $d_{out}(u) = 1$, then take v equal to the one son of u. If $d_{out}(u) = 2$, then both of its subtrees have size $\leq \frac{2}{3}n$. Now take v equal to the root of the largest of the two subtrees. □

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

We denote the complete undirected graph on n nodes by K_n. A *tournament* on n nodes is any directed graph that can be obtained by giving a unique orientation to each edge of K_n.

In graphs G, we write $v \xrightarrow{\star} w$ ($v \xrightarrow{k} w$) 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 $|\pi|$. The *distance* from v to w is the length of the shortest path from v to w. The *diameter* D of a graph is the maximum of the distances in the graph, over all pairs of vertices. Thus, in the directed case, the diameter is defined (finite) only in the case of strongly connected graphs.

For directed acyclic graphs G, the *depth* $d(G)$ is the length of the longest directed path in G.

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.

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 *width* $w(G)$ of a directed acyclic graph G is the size of the largest anti-chain in 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 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 [24].

***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].

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 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

Deterministic time-complexity classes will be denoted by DTIME($f(N)$), where $f(N)$ is a time bounding function. NP is the class of all decision problems that are solvable in nondeterministic polynomial time. We assume that the reader is familiar with the theory of NP-completeness [26].

An optimization problem is said to be efficiently r*-approximable* if there exists a polynomial-time algorithm that computes a solution to the problem that is within a factor r from the optimum solution. We will only use basic facts for NP-optimization problems. For more details, see Ausiello *et al.* [2]. For an overview of the many different types of efficient approximation scheme that have been distinguished and their interrelationships, we refer to [45].

Finally, we will need some concepts from parameterized complexity theory [19]. In this theory one classifies decision problems in the so-called W-hierarchy:

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

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. It is known, however, that all classes are closed under *standard parameterized (FPT-)reductions* and that all classes have complete problems under the implied notion of reducibility (cf. [19]). We will use the following problem, of which the decision version is known to be NP-complete and the parameterized version $W[2]$-complete (cf. [19], p. 444):

HITTING SET
Input: a finite universe $U = \{u_1, \cdots, u_n\}$, a family $\mathcal{S}$ of subsets $S_1, \cdots, S_m \subseteq U$, 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 $i = 1, \cdots, m$.

If a problem is $W[t]$-hard for some $t \geq 1$, then the problem is not fixed-parameter tractable unless $FPT = W[0] = \cdots = W[t]$. In [19] it is extensively argued that equalities like $FPT = W[1]$ or $W[1] = W[2]$ are very unlikely to hold, but they remain open to date.

The following fact follows from a result of Feige [21].

Fact 5 HITTING SET *is not efficiently* $(1-\epsilon)\ln N$*-approximable for any* $\epsilon > 0$ *unless* NP $\subseteq$ DTIME($N^{\log\log N}$)*, where* $N(= n + m)$ *is the size of a problem instance.*[2]

The complexity-theoretic status of the shortcutting problem will be considered in Section 8.

[2] Feige's result was originally proved for SET COVER but this is equivalent to HITTING SET (cf. [2], problem SP7, p. 426).

2.3 Inequalities

In later analyses we use an extension of *Jensen's inequality* (cf. [29]) for convex and concave functions. The extension is given in the following Lemma and seems new.

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 $x_1, \cdots, x_k$ in the joint domain, 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 which seems new as well.

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 $\mu_1, \cdots, \mu_k$ be positive weights. Then for any integer $k \geq 1$ and any $x_1, \cdots, x_k$ in the joint domain, one has*

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

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 $x_{av} = \frac{1}{\Sigma\mu_i}(\mu_1 x_1 + \cdots + \mu_k x_k)$. 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:

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

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. We prove a better bound and give a more efficient algorithm to compute it. In fact, this section will be largely devoted to a more general result.

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 extend this to a result for δ-shortcutting. 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 first outline the basic construction in Subsection 3.1 and then derive the concrete bounds in Subsection 3.2. In Subsection 3.3 we discuss the optimality of the bounds.

3.1 Constructing clusters

We first give the general principle. Let G be an (undirected) connected 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 at least $\lambda+1$ nodes and diameter $\leq \lambda$. Because $n \geq \lambda+1$, G will contain at least one λ-cluster.

Definition 1. *A set of λ-clusters $C_1, \cdots, C_k$ 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 $C_1, \cdots, C_k$ 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 [30]. 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 a sub-tree in maze order and *return* to r' as long as there are unvisited sub-trees left at r.[3]

Maze order traces the entire tree S, respects subtrees, and does so by traversing every edge 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 is performed in linear time. A key property is that consecutively visited nodes are always directly connected. We will use the maze-order traversal to compute a maximal set of disjoint λ-clusters in G as follows.

Assume that at the start, all nodes are colored white. We will reserve the color blue for nodes that are part of an identified λ-cluster, and the color red for nodes that we have to leave aside. Initialize the process by starting at the root r of S, and tracing 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 them and color them blue. We now continue the traversal where we left off, possibly backing up over any blue nodes and 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 when 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 corresponding subtree at r'. We cannot merge this subtree with the λ-cluster of r' as this might violate the diameter constraint of the cluster. Therefore we color the nodes in this entire subtree 'red', leave them aside (as they cannot form a λ-cluster by themselves in S), and continue the traversal at r'. Note that the 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 $B_1, \cdots, B_l$ (with $l \geq 1$ by the initializing part). By design the clusters are all disjoint. We claim that it is a good set. Clearly $l \leq \frac{n}{\lambda+1}$. Next, 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

[3] Maze-order differs from traditional traversal orders like 'pre-order' by explicitly re-visiting a node r every time after one of its subtrees has been traversed [31].

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 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 $C_1, \cdots, C_k$ 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 $C_1, \cdots, C_k$ 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 (or height) of at most $\log_{d-1} k$. We now embed this tree into graph G, by adding the k edges to G and 'connecting' the clusters the way they are linked in T. Note that as 'super nodes' in T, each cluster is incident to precisely d 'extra' edges. When adding the extra edges to G, we divide the 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 k edges on the diameter of G. Let u, v be two arbitrary nodes of G. Both u and v have distance at most λ to any of the clusters, and thus are at a distance of at most $2\lambda - c$ to 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' edge to the 'forward going' edge on the path.

Thus we get a diameter of at most $4\lambda - 2c + 2c\log_{d-1} k$. The bounds in the Lemma follow by substituting $k \leq \frac{n}{\lambda+1}$. By Lemma 2 the clusters $C_1, \cdots, C_k$ can be determined in linear time. The embedding of the edges of T trivially follows in the same linear time-bound. □

3.2 Concrete Bounds

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

We first show how Lemma 3 allows us to extend the result for diameter reduction as given by Alon *et al.* [1] in the unconstrained case. Recall that 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.

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 $\lceil \frac{1+\lceil n^{\frac{\beta}{D}} \rceil}{3} \rceil < n^{\frac{\beta}{D}}$. □

Theorem 2 shows clearly how the result of Alon *et al.* [1] is extended. By a change of parameters one can reformulate the result as follows.

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. For example, take $\lambda = 2$, $c = 2$ and $d = 3$ in Lemma 3. With this setting one obtains that any n-node graph can be 1-shortcut to a diameter D with $D \leq 4 + 4 \log_2 \frac{n}{3}$ by adding at most $n/3$ edges. Moreover, the shortcutting edges can be determined in linear time. This already improves on Bokhari and Raza's construction, both in the number of edges needed and in time complexity. However, one can do better still, using only a *sublinear* number of edges.

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. This gives 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 the requirement is not satisfied, then the diameter of G is bounded by $1 + \delta$ without having to add any edges at all.

Interestingly, Lemma 3 can give diameters smaller than $O(\log_{1+\delta} n)$ while still adding only a sublinear number of edges, provided we allow degrees to increase by more than a constant. Let $\gamma(n)$ be any integer function with $2 \leq \gamma(n) < \log_2 n$.

Theorem 5. *The diameter of any connected graph G of n nodes can be reduced 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 $\lambda \geq 2$. 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 $0 < \rho \leq 1$. 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 rather, 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, 33, 23]). In Section 8 we comment on this for the case in which the degree constraint is taken into account.

4 δ-Shortcutting Rooted Directed Paths

Directed graphs are even more adequate models of networks than undirected ones. However, the δ-shortcutting problem for directed graphs appears to be considerably more difficult. Before we can address the general problem 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 *rooted directed trees.*

4.1 Rooted Directed Paths

Let $G = \langle V, E\rangle$ with $V = \{v_1, \cdots, v_n\}$ and $E = \{(v_i, v_{i+1}) \mid 1 \leq i < n\}$ be a rooted directed path. We view the nodes $v_1, \cdots, v_n$ 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 diameter of a directed path down to $O(\alpha(n))$, where $\alpha(n)$ is the inverse Ackermann function (cf. [41]). The diameter can be reduced to $O(\log_2 n)$ by adding only $O(\frac{n}{\log_2 n})$ edges ([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 diameter $O(\log_2 n)$ by adding only $O(\frac{n}{\log_2 n})$ edges while degree increases remain bounded by at most 1 per node. In fact, we prove the following, stronger result.

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

The proof is divided into three parts. First, we re-organize the layout of G into a special arc structure in Subsection 4.2. The construction is related to the one in 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 Subsection 4.3. The proof that the construction achieves the stated bounds follows in Subsection 4.4. The proof will also show that the entire construction can be done in *linear* time.

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 see that d and λ can be chosen as desired.

We will first re-organize the linear layout of G and place 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 $(\lambda + 1)|n$, 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.

Blocks A block of $\lambda + 1$ nodes will consist of a *list part* and a *switch part*. In the list part the nodes are connected in a directed path ('from left to right'), in the switch part the 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 Figure 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.

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 in T, which is a block 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 'unconnected' 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. Figure 1). When the maze-order traversal of T reaches blocks 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 first son is attached and traversed,
- the block of the first son is traversed and arcs are attached in this block and its subtree recursively,
- returning from this block, the upward arc coming in from the end-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 end 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 its leaves is obtained by a trivial variant of the description above. (NB If n was not a multiple of $\lambda + 1$ to start with, we add the final segment of $n - (\lambda + 1)k$ nodes of the path that we initially left off back in, by a special arc from the end-node of the root block.)

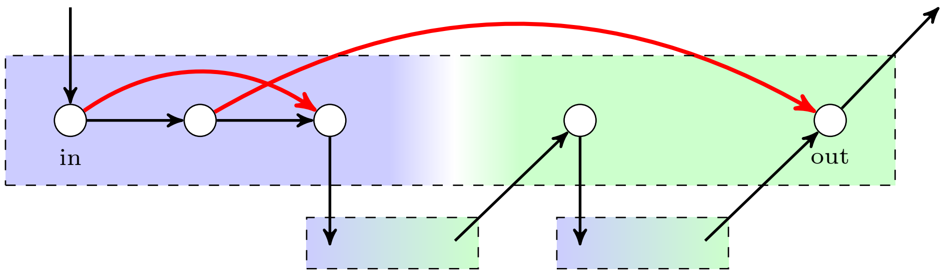

Figure 1: *Design of a block, with its list and switch parts in blue (left) and green (right) respectively. The two red (overarching) arcs are the only shortcut arcs in the block.*

***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. This gives a rooted directed path of n nodes. □

We identify the virtual arc structure that is obtained with G. 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, as follows.

Consider each block B in turn. Add two shortcut arcs to B, if applicable: one arc from B's in-node to the last node of its list part, and one from the node adjacent to B's in-node to its out-node (cf. Figure 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 in turn. At 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 down 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 can be 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 $\lambda = \lfloor 3 + \delta \log_{1+\delta} n \rfloor$. As we 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. We observed that 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 found by a basic linear-time maze-order traversal process.

As in Section 3, 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 edges at all.

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

5 δ-Compressing Rooted Directed Trees

Besides the result for rooted directed paths we need one more auxiliary result, namely on *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 and such that degree increases remain bounded.

δ-Compression relates to various classical issues in the theory of data structures of reducing root-to-node and node-to-root distances in trees (cf. [32]). The problem is different from 'balancing' trees by rotating subtrees, 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]. However, 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 edges 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 $12dn/\lambda$ edges and a degree increase of at most $12d$, 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* as it will be given in detail. A typical application is obtained by taking $d = 2+\delta$ and $\lambda = 6\log_{1+\delta} n$, as shown in Theorem 7 below.

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 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 edge 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 a 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 edges 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 Subsection 5.2 we give the basic idea behind the lemma and the construction of the clusters we need. In Subsection 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 Subsection 5.4 we bound the number of extra edges that are used. In Subsection 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. Finally, we give some concrete applications of the trade-off lemma.

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 $d > 2$.

Set-up We copy G to a rooted directed tree P. We implement all modifications and shortcuts on P rather than on G, level after level, so G itself remains unaltered for the purpose of reference. We use a simple color code. Initially all edges in P are colored 'green'. The edges that are untouched will remain green. The shortcut edges that are gradually added to P are colored 'red'. No other colors are used. In the end we are interested in the depth of P and in the number of red edges that were added in the process.

Labels Whenever a next level of P is constructed, nodes in this level will be labeled A, B, C, or D 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 initialised to label A.

Levels P will be compressed by creating consecutive levels to which clusters (i.e. subtrees) of lower nodes from the tree are 'pulled up'. This will be done by adding shortcut edges to them, thus effectively elevating their root to a higher level and bringing all nodes in the subtree closer to the root. This proceeds recursively. We will define later what clusters are selected.

In general, if the next level i of P is reached for processing (i integer), then up to *two* next levels are created: a *possible* level $i + \frac{1}{2}$ and a level $i + 1$. Level $i + \frac{1}{2}$ consists of the roots of selected clusters that are pulled up and linked to from their ancestor in level i by a shortcut (i.e. red) edge. (This will also pull up all nodes inside the respective clusters, bringing them closer to the root of P.) Nodes in level $i + \frac{1}{2}$ will always be labeled D, to mark them special. Level $i + 1$ consists of the nodes that were pulled up as *sons* of the selected root nodes in level i or level $i + \frac{1}{2}$. This process then continues recursively on the 'pulled up' subtrees in 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.

Clusters We are now ready to describe the recursive path compression process, starting at level 0. Suppose more generally, that we are to process level i (i integer). Note that the part below level i is still 'uncompressed'. We begin with the first part, the construction of clusters in the subtrees that are attached to 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 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 subtrees P_v (in P) by means of a tree covering method due to Geary *et al.* [27, 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. *[27] 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 $n_v = |P_v|$ (the weight of v). Note that P_v is still unlabeled and has only green edges.

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 $d < x_v \leq 1 + 12d$. 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

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

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 $L_v \geq \frac{1}{3d} n_v - 2$, it follows that

$$x_v \leq 1 + \frac{n_v - 1}{L_v - 1} \leq 1 + 3d \cdot \frac{n_v - 1}{n_v - 9d}$$

and hence, using that $n_v > 12d$, one has $x_v \leq 1 + 12d$. 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, which relates to the lower bound of the B-label.

***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 the current set of subtrees of P 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.

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 of size between L_v and $3L_v$, and one cluster of size at most $3L_v$ but possibly less than L_v and containing the root node v (cf. Claim 4 and Lemma 5). 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 edge 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 edge* 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 edges 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 edges). 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 edges in the process (one for every 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 edges). 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 $\delta_{out} \leq x_v - 1 \leq 12d$ (cf. Claim 4). Red edges 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. It is clear that once a label is assigned to a node, its weight does not change after that. □

After level i has been dealt with as described, the compression process continues with level $i+1$. This completes the description of the recursive method. From now on P will denote the *final* tree that is obtained.

Termination Observe that, in the end, P is a rooted directed tree with paths from the root to all nodes of G, consistent with the way the path would be in G if we would include jumps according to the red edges. Hence, these paths are shortcut in a valid way. We show that the construction must be 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 into level i, $weight(u) \leq n/d^i$.*

Proof. By induction. The statement certainly holds for level $i = 0$, i.e. for 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 into 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. □

An immediate consequence of Claim 7 is that the process we described terminates after at most $\log_d n$ integer levels have been created. Also note that in P, by design, any path leading from the root of P downward must eventually enter a node that is labeled B or C, possibly running on into the unmodified subtree attached to this node. A stronger conclusion can be drawn.

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

Proof. Consider the path from the root to u. Let $u_0, u_1, \cdots, u_k$ 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 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 no 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). It follows that the proof is complete. □

5.4 Estimating the Number of Red Edges

Next, we estimate the number of shortcut edges introduced in constructing P. To this end, we will effectively compact P to a tree Q which has as many red edges as P has and internal nodes of out-degree at least 2 only, but which has a smaller number of leaves than P. By Fact 2 the latter will enable us to bound the number of edges in Q, thus including the number of red edges. Q is used *only* for this purpose and need not be actually constructed.

Construction P will be trimmed level after level, by re-examining the entire construction of P from the top down. Recall that the red edges in P are easily identified: these edges are exactly the edges from an A-node to a D-node, where the D-nodes are the nodes that lie in the 'half' levels $i + \frac{1}{2}$ (i integer). We assert:

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

We begin by setting Q equal to G. Then we go through the entire construction of P again and describe how to modify it to reach our goal, level after level. In doing so, we maintain the following *inductive assertion.*

$H(i)$: *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 edges in these levels is the same as in P, and all internal nodes have degree ≥ 2. Moreover, there are no $\frac{1}{2}$-levels j with $j \leq i$ anymore in Q.*

We prove the assertion along with the explanation of how a next level of Q (or of P for that matter) is constructed.

***Claim* 9** *For all integers $i \geq 0$, $H(i)$ holds whenever the compression 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. We proceed by induction. Let $H(i)$ hold for some $i \geq 0$. We show how the elaboration of level i in Q can be arranged such that $H(i+1)$ holds at the point where 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 Q and P. 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, it leads to x_v distinct groups of nodes in level $i+1$. Here a group will be:

- either the set of sons directly connected to from v and arising from a cluster in category **II** and **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 connected to from v via a red edge.

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 will 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 their count by the 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 connected 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 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, but 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, do the following for all D-nodes v' attached to v:

N3: *The D-nodes.* For every D-node v', consider all groups $Z_1, \cdots, Z_k$ 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 edges (including the red edge from v to v'), and connect v to all nodes in $Z_1, \cdots, Z_k$ by a *direct* edge. *Color exactly one of these edges red*, in order to preserve the number of red edges. (We not care about the number of green edges.) 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 remain 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. □

Analysis We conclude that when Q is constructed as given, then $H(i)$ holds for all integers i which occur as levels. But then we are done: promise H follows immediately from the fact that $H(i)$ holds for i equal to the last integer level of P. Thus Q can be constructed as promised in H.

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

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

By promise H all nodes in Q have out-degree ≥ 2. By Fact 2 the number of edges in Q and thus the number of red edges, 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. We first give the basic version and some applications. Next we show that one can bound the degree increases strictly by 1.

Basic version By the overall compression process and its analysis, we can conclude the following proof of Lemma 4.

Proof. Consider any n-node rooted directed tree G. Construct P as specified in Subsections 5.3 and 5.4. By adding the red edges of P to G, the paths from the root to lower nodes in P all become paths in G as well. Thus G is effectively compressed, in a valid manner.

The bounds on the length of the compressed paths in G and the number of red edges that were used, follow from the Claims in Subsections 5.4 and 5.5. In the construction, the degree increases remain bounded by $\delta_{in} \leq 1$ and $\delta_{out} \leq 12d$. □

Observe that the given construction for compressing the tree proceeds top-down and works on every A-, B-, and C-node a constant number of times. Thus, the construction involves only a linear number of cover constructions and takes $O(n^2)$ time. A useful application 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 $O(\frac{\delta n}{\log_{1+\delta} n})$ edges 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 for the lemma are satisfied, including that $n \geq \lambda + 1$. □

However, many other trade-offs between compression and extra edges may be achieved. As an example, we give the following corollary.

Corollary 2. *Any n-node rooted directed tree G can be compressed to a depth of $O(\delta^{\frac{1}{2}} \log_{1+\delta} n)$ by the addition of $O(\delta^{\frac{1}{2}} \frac{n}{\log_{1+\delta} n})$ edges 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 $\lambda = 6\delta^{\frac{1}{2}} \log_{1+\delta} n$ in Lemma 4, for any $\delta \geq 1$. It is easily checked that the constraint on $\log_{1+\delta} n$ again guarantees that the requirements for the lemma are satisfied, including that $n \geq \lambda + 1$. □

Bounding degree increases to 1 A further trade-off can be made, to the effect that the degree increases can all remain strictly bounded by 1, provided that we tolerate a slight increase in depth. The proof is based on a subtle refinement of the construction in 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 edges 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 define $\rho = 12d$. We proceed in a few steps.

We first divide G into clusters (like in Section 3). To this end, create a root cluster consisting of the entire subtree of depth ρ at the root. Next, create clusters in the same way in the subtrees attached to the nodes in level ρ, and so on recursively. If a subtree does not reach to the full depth ρ anymore, we will call it 'incomplete' and *delete* it (except its root) from G for this construction. Let G_ρ be the condensation of G obtained by contracting the clusters to super-nodes. G_ρ is a rooted directed tree again, 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 $\lambda + 2\log_d \frac{n}{12d}$ by the addition of $12d{\cdot}n/12d{\cdot}\lambda = n/\lambda$ edges and a degree increase of at most ρ per node. In particular, at most ρ red (outgoing) edges 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 themselves 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 compressed to a path from the root to the cluster to which node u belongs or, if u belongs to an incomplete subtree, to the root of the cluster to which the latter is attached (thus taking ρ extra steps to reach). 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.

Finally, consider the red edges which emanate from an internal node v, i.e. after the compression. Necessarily v is the root node of a cluster G_v of depth ρ. There are at most ρ red edges leading out of v. These edges all lead to subtrees located 'below', in the original subtrees attached to the leaves of G_v at depth ρ.

Instead of keeping all red edges attached to v, we redistribute them over G_v in such a way that the hierarchical relationships in G are preserved. In particular, if a red edge leads from v to the root of a lower subtree G_w and G_w is part of the full subtree (in G) attached to leaf z of G_v, then we can re-attach the red edge to *any* node on the path from v to z (and still leading 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 edges attached to v can be re-distributed in this way and such that every node in G_v gets at most one red edge. (There is enough room on every path from v to a leaf of G_v even though the paths overlap.)

Redistribute all red edges 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

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

as was to be shown. □

As a typical application of Lemma 6 we mention 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 $\frac{n}{3\log_{1+\delta} n}$ edges 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 $\delta \geq 1$. It is easily checked that the constraint on $\log_{1+\delta} n$ guarantees the requirements of the lemma, including that $n \geq 12d(\lambda+1)$. □

The constraints on n and λ in the given theorems all amount to the requirement that n is not too small in terms of δ. If the requirement is not satisfied, the depth of G is already bounded by $O(\delta)$.

6 δ-Shortcutting Directed Graphs

We now consider δ-shortcutting directed graphs in general. We will actually consider various special cases first, which are of interest in their own right. This includes the

δ-shortcutting problem for DAGs, directed rooted trees with bounded out-degrees, and strongly connected graphs, using the auxiliary results from Sections 4 and 5. 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 Arbitrary DAGs

Let G be a DAG. Recall that the *width* $w = w(G)$ of G is the size of its largest *antichain* (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 distances in an n-node DAG G of width w can be reduced 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 $C_1, \cdots, C_w$ 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 $n_i > 1+\delta$ 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 between them 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 with at most one, possibly shortcut, segment from each chain.

Let there be L chains C_i with $n_i > 1 + \delta$, for some L with $0 \leq L \leq w$. Assume w.l.o.g. that these chains are $C_1, \cdots, C_L$. By Theorem 6, $|\pi|$ 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 by at most 1, at the expense of adding a number of arcs bounded by

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

Segments of a length less than or equal to $1 + \delta$ 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.

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 at most $\Delta \geq 2$. (For $\Delta = 1$ one has a rooted directed path and Theorem 6 applies.)

Theorem 10. *All distances in an n-node rooted directed tree G with (out-)degrees bounded by Δ and height h can be reduced to $O(\Delta\delta\lceil\log_{1+\delta} h\rceil \cdot \log_2 n)$, 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 $\log_{1+\delta} h > 1$.*

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. [32].)

Call any arc from a father node to its leftmost (and 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 in a chain, then following a right-going arc to another head-node, and so on. Suppose π visits K 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 length at most h. Now we 1-shortcut the chains, by the method from Theorem 6. However, we only shortcut chains if they are longer than $\log_{1+\delta} h$ and, also, longer than $1+\delta$. For a chain of length c_i that meets these criteria, this takes $2c_i/(\delta \log_{1+\delta} c_i)$ extra arcs but all distances on the chain are reduced to $O(\delta \log_{1+\delta} h)$, using that $\log_{1+\delta} h > 1$.

***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 lengths $c_1, \cdots, c_L$ 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 the length of π after shortcutting the chains is bounded in the order of

$$K + K \cdot \max(\delta \log_{1+\delta} h, \delta) = O(K\delta\lceil\log_{1+\delta} h\rceil)$$

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. Note that the requirement that $\log_{1+\delta} h > 1$, i.e. $h > 1+\delta$ is not severe. If it is not satisfied, then the height of G is bounded by $1+\delta$ without having to add any arcs at all.

6.3 Strongly Connected Digraphs

When shortcutting strongly connected digraphs, a few results are known when 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. Also, Flaxman and Frieze [22] 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 $O(1)$. In fact we prove a more general result on shortcutting any strongly connected digraph to a diameter $O(\log_{1+\delta} n)$ while keeping the degree increases bounded by $O(\delta)$, provided n is large enough in terms of δ. We also prove a corresponding result in which degree increases are strictly bounded by 2. The proofs make essential use of the results from Section 5.

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 in-tree T_{in} with root r (i.e. with all arcs pointing towards r) that spans G. Likewise there is a rooted directed out-tree T_{out} with root r (with all arcs pointing away from r) that spans G. Consider T_{in} and T_{out} in G.

By Theorem 7, T_{in} and T_{out} can each be compressed to depth $O(\log_{1+\delta} n)$ by adding $O(\frac{\delta n}{\log_{1+\delta} n})$ arcs and with a degree increase of at most $O(\delta)$ at every node. (In the case of T_{in} this follows after reversing the directions of the arcs first, and reversing these arcs and the shortcut arcs again afterwards.) Now 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. □

Directed in- and out-trees as needed in Theorem 11 are typically computed as a side-product of a single-source shortest path algorithm. With the shortcutting of these trees, the shortcut construction for G can thus be done in at most $O(n^2)$ time.

Note that we could have 4-shortcut the strongly connected graph G by first adding two 'oppositely directed' directed paths through all nodes of G at the expense of two extra in- and out-arcs per node, and then applying the result of Theorem 6 to both paths. However, this takes $\Omega(n)$ extra arcs, which is more than we need in the case of Theorem 11. However, if we apply Theorem 8 instead of Theorem 7 in the above proof, we obtain the following result.

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 $\log_{1+\delta} n \geq 4(2+\delta)$.*

As a concrete instance of Theorem 12 we obtain the following fact, which relates to the result of Flaxman and Frieze [22] discussed above.

Corollary 3. *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 Corollary 5.

6.4 General Directed Graphs

We now consider δ-shortcutting general digraphs. We first follow the classical approach based on condensing a graph, using the results for DAGs and for 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 think of w_c as being 'small' with respect to n but this is not needed for the results.

Theorem 13. *The distances in an arbitrary n-node directed graph G can be reduced 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 $H_1, \cdots, H_{n_c}$ 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 $H_1, \cdots, H_{n_c}$ as 'super-nodes', using Theorem 9. Add all the shortcut arcs so obtained to G. (In case of the shortcut arcs of G_c, if such a shortcut arcs connects component H to component H', then the arc is embedded in G by letting it connect an arbitrary node of H to an arbitrary node of H'.)

We first estimate the resulting increase in the size of G. Theorem 11 is applicable and guarantees that the degrees in each strongly connected component can rise by at most $O(\delta)$ if we consider their individual shortcutting only. 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 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 $h_1 + \cdots + h_c = n$ reduces to a bound of

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

which 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 $h_{i_1}, \cdots, h_{i_s}$ for some $s = O(\delta w_c \lceil \log_{1+\delta} \frac{n_c}{w_c} \rceil)$. By the shortcutting the path is kept to a length in the order of:

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

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

Note that $\frac{n}{w_c} \geq \frac{n}{n_c} \geq n_{\min}$, as $w_c \leq n_c$. Crudely estimating the bounds in Theorem 13 we obtain:

Theorem 14. *The distances in an arbitrary n-node directed graph G can be reduced 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 Theorem 13 and Theorem 14 can be modified such that a degree increase of at most 3 is incurred only, at the expense of an extra factor of $O(\delta)$ in the distances.

7 δ-Shortcutting Using Path Covers

In this Section we aim to show that the path cover number $\mu(G)$ of a directed graph G is a useful measure in shortcutting G. The approach generalizes the technique we already used in Subsection 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 path in G. We depict π as an ordered line of nodes 'from left to right'. We will be interested in the number of 'maximal segments' that can be formed on π when it is intersected by another, arbitrary path in G. To this end we need the following concept, which appears to be new.

Definition 3. *The* feedback dimension *of π is the largest $k \geq 0$ such that there exist distinct nodes $u_1, v_1, \cdots u_k, v_k$ '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 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 from a node beyond v_i to a node before v_i.*

If $u_1, v_1, \cdots u_k, v_k$ satisfy the feedback base properties on a path π, then we may assume that every u_i is minimal. 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 $u_1, v_1, \cdots u_k, v_k$ are such that all u_i are minimal, the set is clearly unique for π.

Lemma 7. *Let $u_1, v_1, \cdots u_k, v_k$ 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, we 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. □

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. *The feedback dimension of a digraph 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 $C_1, \cdots, C_k$ 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 its 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, that $u_i \neq v_i$ for each i. Hence, the nodes $u_1, v_1, \cdots, u_k, v_k$ as defined are all distinct. Assume w.l.o.g. that the components $C_1, \cdots, C_k$ were ordered such that the nodes $u_1, v_1, \cdots, u_k, v_k$ are ordered 'from left to right' on π. It is easily verified from their definition that the nodes satisfy the feedback base properties.

(II) Conversely, consider any directed path π in G and nodes $u_1, v_1, \cdots, u_k, v_k$ $(k \geq 1)$ 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 belongs fully to a single strongly connected component, say C_i, of G. By the feedback properties, the C_i $(1 \leq i \leq k)$ must be 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. □

Let $d_c(G)$ denote the depth of G_c. Observe that for an n-node directed path G one has $\phi(G) = 0$ and $d_c(G) = n$. The bound below immediately follows from Lemma 8.

Corollary 4. *For any digraph G, $\phi(G) \leq d_c(G)$.*

7.2 Directed Graphs

The feedback dimension is of interest when it comes to shortcutting directed graphs, as we show now. The result generalizes Theorem 9 for DAGs.

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 reduced 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 $\mu = \mu(G)$. Thus, G can be partitioned into μ node-disjoint directed paths $\pi_1, \cdots, \pi_\mu$. 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 the paths have lengths $n_1, \cdots, n_\mu$, respectively, then the number of added arcs is certainly bounded by

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

as in the proof of Theorem 9 and 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 $O(\delta \log_{1+\delta} n_i)$.

If G has cycles, we may need to do more shortcutting. To see which extra shortcut arcs we need, consider any path π in the path cover of G. We depict 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 $u_i \leq u < v_i$ we have that $f_\pi(u) = v_i$. Also $f_\pi(v_i) = \perp$.*

Proof. Let the path from v_i back to u_i be τ. First, suppose that $f(v_i) = v \neq \perp$, 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 $u_i \leq u < v_i$. 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 that in this case 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 the nodes $u_1, v_1, \cdots, u_k, v_k$ satisfy the feedback base property on π and thus that $k \leq \phi(G)$.

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 $u_1, v_1, \cdots, u_k, v_k$ on π and add a shortcut arc from v_i to u_i for each $1 \leq i \leq k$. 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 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 $u \neq v$, 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 $u_1, v_1, \cdots, u_k, v_k$ be the list of nodes on π as constructed above. It follows that there must be an i $(1 \leq i \leq k)$ 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 considering the subsequent paths of the cover in turn. One easily argues that the path τ' that is ultimately obtained, is still a path from x to y but also satisfies the claim. □

Finally, we shortcut path τ' further by shortcutting its segments on every path of the cover, using the blue arcs. By Claim 14, the length of the resulting path after shortcutting is certainly bounded in the order of:

$$\mu + 2 \cdot \delta(\log_{1+\delta} n_1 + \cdots \log_{1+\delta} n_\mu) + 2\mu(1+\delta) \leq$$

$$\mu + 2\delta\mu \log_{1+\delta} n/\mu + 2\mu(1+\delta) = O(\delta\mu\lceil\log_{1+\delta} n/\mu\ \rceil)$$

Note that in the estimate we account for the possibility that some of the paths in the cover may not be longer than $1+\delta$ and thus are not shortcut. This completes the proof. □

We draw some immediate conclusions from Theorem 15.

Corollary 5. *The distances in an n-node strongly connected, directed graph G with path cover number $\mu = \mu(G)$ can be reduced 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 $\delta \geq 1$.*

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

We can also reformulate Theorem 15 using the stability number of a graph instead of its path cover number.

Theorem 16. *The distances in an n-node directed graph G with stability number $\alpha = \alpha(G)$ and feedback dimension $\phi = \phi(G)$ can be reduced 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 $\delta \geq 1$.*

Proof. By the Gallai-Milgram theorem (cf. Fact 4) one has $\mu(G) \leq \alpha(G)$. The path cover used in the proof of Theorem 15 is easily transformed into one consisting of $\alpha(G)$ paths, by just breaking some of the paths in multiple pieces if necessary. The result now follows by using this cover and α instead of μ in the proof. □

Corollary 6. *In any directed graph G with stability number α and feedback dimension ϕ, the distances between all nodes can be reduced to $O(\alpha\lceil\ln\frac{n}{\alpha}\rceil)$ by adding only $\frac{4n}{\ln n/\alpha} + \alpha\phi$ edges and with degree increases of at most $O(1)$ per node.*

Theorem 16 can easily be modified for the case of strongly connected directed graphs as well, by taking $\phi(G) = 1$.

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, mostly to graphs with known cycles or long paths.

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 distances in G can be reduced 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 $D_1, \cdots, D_\gamma$. Now consider the proof of Theorem 15 and use the path cover with paths $\pi_1, \cdots, \pi_\gamma$, 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 $u_1, v_1, \cdots, u_k, v_k$ 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 distances in any n-node digraph G with a Hamiltonian cycle can be reduced to $O(\log_{1+\delta} n)$ 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. □

By Camion's theorem, every *strongly connected* tournament has a Hamiltonian cycle [12]. Thus, Theorem 17 leads to the following corollary.

Corollary 7. *The distances in any n-node* strongly connected tournament *can be reduced to $O(\log_{1+\delta} n)$ 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 $\log_{1+\delta} n \geq 1$.*

Graphs with Long Paths Theorem 15 can be applied also to e.g. directed graphs that possess one or more very long paths. We consider only the case of 'near-Hamiltonian' directed graphs, i.e. digraphs that have a simple path of length $n-1-t$ for some 'small' t with $t \geq 0$. (NB The case $t = 0$ corresponds to graphs having a Hamiltonian path.)

We first remark that near-Hamiltonian directed graphs are not rare. We characterise a class of near-Hamiltonian graphs by means of a 'forbidden subgraph'. Let $H_{p,q,r}$ be the directed tripartite 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 connected by an arc to all nodes of B, and (iii) all nodes of B connected by an arc to all nodes of C. The following Lemma extends Proposition 4.7 in [3].

Lemma 9. *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 $\geq n-t-1$ ($t \geq 0$).*

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. □

The following observation is immediate from Theorem 15.

Theorem 19. *The distances in any n-node digraph G with a near-Hamiltonian path of length $n-t-1$ ($t \geq 0$) and feedback dimension $\phi(G)$ can be reduced to $O(\delta(t+1)\log_{1+\delta} n)$, by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n} + (t+1)\phi(G)$ 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. □

The following result generalizes Theorem 18.

Theorem 20. *The distances in any n-node digraph G with a Hamiltonian path and feedback dimension $\phi(G)$ can be reduced to $O(\delta \log_{1+\delta} n)$, by the addition of at most $\frac{4n}{\delta \log_{1+\delta} n} + \phi(G)$ 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 $t = 0$. □

Corollary 8. *The distances in any n-node* tournament *can be reduced to $O(\ln n)$, by the addition of at most a linear number of arcs and a degree increase of at most 2 per node.*

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 proof one may want to use a Hamiltonian path that has a small feedback dimension in order to keep the number of shortcut arcs low. A case in which the feedback dimension can be guaranteed to be 1, is already covered by Corollary 7.

8 Complexity of Shortcutting

The general question of reducing the diameter of a network by adding a smallest number of edges is well-known to be computationally hard (cf. [38, 16, 25, 23, 10]). We show that the problem remains computationally hard even when the degree constraint is added. We also comment on the *fixed-parameter (in)tractability* of the problem, where the number of extra edges allowed is the parameter.

We consider 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 reduced by at least 1 by adding at most k edges while increasing the degrees in G by at most 1?

We also consider the optimization version of this problem. In this problem one is asked to minimize the number of edges that have to be added to G in order to reduce its diameter by at least 1, while the degrees in G are still increased by at most 1. Abusing terminology, we call this problem as Shortcutting as well.

The version of the shortcutting 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 *without* the degree constraint [25, 23]. We show that these facts even hold when the degree constraint is imposed. The results follow from the following reduction, from Hitting Set (cf. Subsection 2.2).

Lemma 10. *There is a polynomial-time algorithm which achieves the following. Given a universe $U = \{u_1, \cdots, u_n\}$ and a family $\mathcal{S}$ of sets $S_1, \cdots, S_m \subseteq U$, the algorithm constructs a graph G with $O(n+m)$ vertices such that for any integer k with $1 \leq k \leq n$, a hitting set for $(U, \mathcal{S})$ of size k can be transformed in polynomial time into a set of k edges that reduces the diameter of G by at least 1 when added to G while increasing the degrees in G by at most 1, and vice versa.*

Proof. Construct the 5-layered graph $G = \langle V, E \rangle$ with

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

and with the following edges:

- u_i is connected to S_j, for all $i = 1, \cdots, n$ and $j = 1, \cdots, m$ such that $u_i \in S_j$ (i.e. the edges representing the element-set relation).
- u_i is connected to $u_{i'}$, for all i, i' with $1 \leq i < i' \leq n$ (i.e. the nodes u_i form a clique).
- a is connected to each u_i ($i = 1, \cdots, n$).
- a is connected to each b_i ($i = 1, \cdots, n$).
- b_i is connected to $c_{i'}$, for all $i = 1, \cdots, n$ and $i' = 1, \cdots, 2n+1$ (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 G has diameter 4, where the longest distances are realized (only) between the S_j and $c_{i'}$, for all $j = 1, \cdots, m$ and $i' = 1, \cdots, 2n+1$.

Suppose that $(U, \mathcal{S})$ has a hitting set $H = \{u_{h_1}, \cdots, u_{h_k}\}$ of size k, with $1 \leq k \leq n$. Shortcut G by adding edges from b_i to u_{h_i} for $i = 1, \cdots, k$. Denote the resulting graph by G'. To show that G' has diameter less than 4 it suffices to show that the distances between the S_j and the $c_{i'}$ are reduced (cf. the above observation). Indeed, let $j \in \{1, \cdots, m\}$ and $i' \in \{1, \cdots, 2n+1\}$, and let $i \in \{1, \cdots, k\}$ 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 1 less than the diameter of G. Moreover, we added k edges and increased the degrees by at most 1.

Conversely, suppose that there is a set F of k edges ($1 \leq k \leq n$) that reduces the diameter of G by at least 1 when added to G while increasing the degrees in G by at most 1.[4] Let G' be the graph obtained from G by adding the edges of F. Because $k \leq n$, there is a node $c_{i'}$ for some $i' \in \{1, \cdots, 2n+1\}$ that is not an endpoint of any edge in F. For $j = 1, \cdots, m$, 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). We now construct a hitting set H as follows. For $j = 1, \cdots, m$, 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 $i \in \{1, \cdots, n\}$, 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. We start with the basic observation for each $j \in \{1, \cdots, m\}$, that P_j contains at most three edges (since the diameter of G' is at most three by the definition of F and G), and that the first edge of P_j must be an edge of G by the definition of $c_{i'}$. Hence, if $e_j = e_{j'}$ for distinct $j, j' \in \{1, \cdots, m\}$, 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 $i \in \{1, \cdots, n\}$. 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 $j \in \{1, \cdots, m\}$, 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 has 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 $i \in \{1, \cdots, n\}$, and thus not equal to $e_{j'}$ for any $j' \in \{1, \cdots, m\}$. Therefore, each element in H can be charged to a unique edge from F, and thus $|H| \leq |F| = k$. □

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 G has $m + 4n + 2$ vertices and can be constructed in polynomial time, and that both transformations described above take polynomial time as well. □

Lemma 10 has the following three consequences for the complexity of Shortcutting, refining and improving on the results for diameter reduction in general (cf. [23, 10]).

Theorem 21. Shortcutting *is NP-complete.*

Proof. It is easily seen that Shortcutting is in NP. To see that Shortcutting is NP-hard, recall that Hitting Set is NP-hard [26]. Consider an instance $(U, \mathcal{S}, k)$ of Hitting Set. If $k > |U|$, then the instance is a trivial "yes"-instance, and we return a trivial "yes"-instance of Shortcutting. Otherwise, apply the algorithm of Lemma 10 to $(U, \mathcal{S})$ and let G be the resulting graph. It takes polynomial time to compute G. Moreover, by Lemma 10, $(U, \mathcal{S}, k)$ is a "yes"-instance of Hitting Set if and only if (G, k) is a "yes"-instance of Shortcutting. Hence, Shortcutting is NP-complete. □

Theorem 22. Shortcutting *is W[2]-hard.*

[4] We actually do not use the assumption that the degrees have increased by at most 1 in the proof.

Proof. Recall that Hitting Set is W[2]-complete [19]. Consider an instance $(U, \mathcal{S}, k)$ of Hitting Set. If $k > |U|$, then the instance is a trivial "yes"-instance, and we return a trivial "yes"-instance of Shortcutting. Otherwise, apply the algorithm of Lemma 10 to $(U, \mathcal{S})$ and let G be the resulting graph. It takes polynomial time to compute G. Moreover, by Lemma 10, $(U, \mathcal{S}, k)$ is a "yes"-instance of Hitting Set if and only if (G, k) is a "yes"-instance of Shortcutting. This constitutes a parameterized reduction, and thus Shortcutting is W[2]-hard. □

Observe that the above result implies that Shortcutting is not in FPT, unless FPT = W[1] = W[2] which is considered unlikely (cf. [19, Ch 23]). It follows that the optimization version is unlikely to admit a 'fully polynomial-time approximation algorithm' (FPTAS), as this would imply the existence of an FPT-algorithm for the problem (cf. [19, Theorem 9.3.1]). We can make the following further statement about the approximation hardness.

Theorem 23. Shortcutting *has no* $(1-\epsilon)\ln N$*-factor approximation algorithm for any* $\epsilon > 0$*, unless NP* $\subseteq$ *DTIME*$(N^{\log\log N})$*, where* N *is the number of vertices of the* Shortcutting *instance.*

Proof. Observe that using Lemma 10, it follows that a $(1-\epsilon)\ln N$-factor approximation algorithm for Shortcutting for some $\epsilon > 0$ implies a $(1-\epsilon')\ln(n+m)$-factor approximation algorithm for Hitting Set for some $\epsilon' > 0$. The lemma now follows from Fact 5, based on Feige's classic result for Set Cover [21]. □

The results hold for the analogous problem 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 balancing distances in networks while adding only a bounded number of extra links per node. Our analysis has shown that the problem is intimately related to many classical issues in graph decomposition and covering 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 (Theorem 7 and Theorem 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, 33, 23]. The 'δ-perspective' might be of considerable interest for further study.

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* (to appear).
4. I. Ben-Arroyo Hartman, Variations on the Gallai-Milgram theorem, *Discrete Mathematics* 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, *Theor. Comp. 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 Transactions on Computers* C-35:8 (1986) 757-761.
9. M.L. Bonet, S.R. Buss, The serial transitive closure problem for trees, *SIAM J. Computing* 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 Applic.* 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) 21512152.
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 Mathematics* 44 (1983) 243-250.
15. F.R.K. Chung, Diameters of graphs: old problems and new results, *Congressus Numerantium* 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, 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, pp. 750-759. USA
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, Lecture Notes in Computer Science Vol. 5124, Springer-Verlag, Berlin, 2008, pp. 173-184.
21. U. Feige, A threshold of $\ln n$ for approximating set cover, *J. ACM* 45:4 (1998) 634-652.
22. A.D. Flaxman, A.M. Frieze, The diameter of randomly perturbed digraphs and some applications, *Random Structures & Algorithms* 30:4 (2007) 484 - 504.

23. 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, Lecture Notes in Computer Science Vol. 8283, Springer-Verlag, Berlin, 2013, pp. 383393.
24. T. Gallai, A.N. Milgram, Verallgemeinerung eines graphentheoretischen Satzes von Rédei, *Acta Sc. Math. (Szeged)* 21 (1960) 181-186.
25. Y. Gao, D.R. Hare, J. Nastos, The parametric complexity of graph diameter augmentation, *Discr. Appl. Math.* 161 (2013) 1626-1631.
26. M.R. Garey, D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, W.H. Freeman & Co, San Francisco, 1979.
27. R.F. Geary, R. Raman, V. Raman, Succinct ordinal trees with level-ancestor queries, *ACM Trans. Algorithms* 2:4 (2006) 510-534.
28. 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.
29. J.L.W.V. Jensen, Sur les fonctions convexes et les inégalités entre les valeurs moyennes, *Acta Math.* 30 (1906) 175-193.
30. J. Kleinberg, E. Tardos, *Algorithm design*, Addison-Wesley, Boston, 2006.
31. D.E. Knuth, *The art of computer programming*, Vol 1: *Fundamental algorithms*, 3rd Edition, Addison-Wesley, Reading, MA, 1997.
32. D.E. Knuth, *The art of computer programming*, Vol 3: *Sorting and searching*, 2nd Edition, Addison-Wesley, Reading, MA, 1998.
33. C-L. Li, S.T. McCormick, D. Simchi-Levi, On the minimum-cardinality-bounded-diameter and the bounded-cardinality-minimum-diameter edge addition problems, *Operations Research Letters* 11 (1992) 303-308.
34. A. Meyerson, B. Tagika, Minimizing average shortest path distances via shortcut edge addition, in: I. Dinur *et al.* (eds), *APPROX'09/RANDOM'09*, 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 Structures & Algorithms* 20:2 (2002) 165-183.
36. S. Raskhodnikova, Transitive-closure spanners: a survey, in: O. Goldreich (Ed.), *Property Testing - Current Research and Surveys*, Lecture Notes in Computer Science Vol. 6390, Springer-Verlag, Berlin, 2010, pp 176 - 196.
37. L. Rédei, Ein kombinatorischer Satz, *Acta Litteraria Szeged* 7 (1934) 3943.
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*, Chapters 1-38, Springer, Berlin, 2003.
40. T. Szele, Kombinatorische Untersuchungen über den gerichteten vollständigen Graphen, *Mat. Fiz. Lapok* 50 (1943) 223256 (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, Lecture Notes in Computer Science Vol. 657, Springer-Verlag, 1993, pp. 205-211.
43. M. Thorup, Shortcutting planar digraphs, *Combinatorics, Probability and Computing* 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 of Computing Systems* 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.