CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Computing Diameters in Slim Graphs

Master's thesis in Computer Science-Algorithms, Languages and Logic

BENJAMIN BLOCK
MICHAEL MILAKOVIC

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2018

Master's thesis 2018

Computing Diameters in Slim Graphs

Benjamin Block
Michael Milakovic

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2018

Computing Diameters in Slim Graphs
Benjamin Block and Michael Milakovic

Supervisor: Professor Peter Damaschke, Computer Science and Engineering
Examiner: Associate Professor Alexander Schliep, Computer Science and Engineering

Master's Thesis 2018
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2018

Computing Diameters in Slim Graphs
BENJAMIN BLOCK
MICHAEL MILAKOVIC
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

With the use of large graphs with n vertices and m edges, the current approach for computing the diameter is not efficient. We have investigated a special graph class, namely slim graphs. Slim graphs are graphs whose diameter is at least some fixed fraction of the number of vertices. This constraint allows us to prove structural features in these special graphs. Using these features, we have developed three algorithms which are asymptotically superior to diameter computation in the general case. We present the following three algorithms, for a fixed $0 < k < 1/2$: a $(1-k)$-approximation algorithm of the diameter in $O(n+m)$ time; a deterministic algorithm which computes the diameter in $O(n^2)$ time and a Monte Carlo algorithm which also computes the diameter in $O(n^2)$ time.

Keywords: Computer, approximation, computer science, thesis, algorithms, Monte Carlo algorithm, graphs, slim graphs, diameter computation

Acknowledgements

We want to thank our supervisor Prof. Peter Damaschke, without whom this thesis would not have been possible, for the valuable insights, discussions and the quick responses through email. We also want to thank our examiner Assoc. Prof. Alexander Schliep for the valuable feedback.

Benjamin Block and Michael Milakovic
Gothenburg, June 2018

Contents

List of Figures

1
Introduction

Many algorithms for a wide variety of problems have been developed over the years for general graphs. One such instance is diameter computation. Even though the diameter can be computed in polynomial time, it becomes computationally expensive for graphs that are not even large [1]. In such graphs, approximation algorithms are commonly used [1]. But in special graph classes, such as slim graphs, one might expect to do better (compute faster) than the algorithms running on general graphs. In a subset of slim graphs with n vertices and m edges, one can compute the diameter in $O(m + n)$ complexity compared to $O(mn)$ in the general case [2].

The diameter of a graph is defined as the length of a longest of all shortest paths between any two vertices. In the general case, the fastest known way to compute the diameter is by computing the shortest path for every vertex-pair in the graph (all-pair shortest path; APSP) and returning the largest one.

Slim graphs are characterized by having a large diameter. They are of special interest as they appear in road networks [3], chain molecules, connections between two fixed sites in a network, etc [2]. Slim graphs can inherently behave quite differently depending on the size of the diameter in relation to the number of vertices, which is covered with greater detail in Chapter 2.

This thesis extends the results presented by Damaschke for a broader range of slim graphs by generalizing the properties shown in [2]. A better structural and behavioural understanding of slim graphs can lead to new, faster algorithms than the already known general ones, as shown by Damaschke. There has not been a lot of research concerning slim graphs, but faster algorithms have been developed for other special graph classes. This suggest that it might be worthwhile to investigate slim graphs further.

This chapter introduces diameter computation, the problem definition, scope and contributions of this thesis. We also mention related work to put the thesis into context.

1.1 Problem definition

The main research questions we are answering in this thesis are the following:

- Is it possible to extend the properties and techniques used by Damaschke [2] for a broader range of slim graphs?
- Is it possible to create a Monte Carlo algorithm for a broader range of slim graphs?
- Is it possible to create a Las Vegas algorithm for a broader range of slim graphs faster than APSP?
- For a fixed $0 < k < 1/2$, is it possible to create an (1-k)-approximation algorithm faster than APSP?
- Is it possible to create a deterministic algorithm faster than APSP for a broader range of slim graphs?
- Does a deterministic algorithm, for graphs with diameter greater than $n/3$, in $O(m + n \log n)$ time exist? This question was first formulated by Damaschke in [2].

1.2 Related work

There has not been a lot of work concerning slim graphs, but there exists special graph classes which have been extensively studied. Previous work has demonstrated that structural features in such graph classes can be exploited to create new faster algorithms than their general counterpart [4–9]. Many of the concepts and results presented by Damaschke in [2] have either been generalized or have influenced the results in this thesis.

1.3 Contributions

We present three algorithms: a Monte Carlo algorithm for computing the diameter in $O(n^2)$ time for slim graphs; for a fixed $0 < k < 1/2$, an (1-k)-approximation algorithm in $O(n + m)$ and a deterministic algorithm in $O(n^2)$. We also prove some structural properties of slim graphs which are not used in the algorithms, in the hope that future work might make use of these new insights.

1.4 Limitations

We will not conduct any benchmarking on the algorithms we propose in this thesis, thus the practicality of the algorithms will be difficult to assess. Our primary concern with this thesis is not to optimize the constant factors in the time complexity of the algorithms. Our intention is rather to supply novel insights into slim graphs; and future work might optimize the algorithms further.

2
Prerequisites

This chapter covers various prerequisites including: approximation- and randomization algorithms, basic graph theory and some concepts and definitions needed for the results. It also covers more thoroughly the methods of diameter computation presented by Damaschke [2].

2.1 Algorithms

As the difficulty of a problem an algorithm is intended to solve increase, often also does the complexity of the algorithm. There exists a trade off though, if we allow the output of the algorithm to be in a certain range from the actual value we want to compute, we can usually create an approximation algorithm with lower complexity compared to its exact counterpart. The approximation algorithm solves the problem with a certain guaranteed factor within the optimal solution. One such example is an approximation algorithm for diameter computation. One can trivially compute a 2-approximation of the diameter by picking one arbitrary vertex in a graph, performing a breadth-first search (BFS) from that particular vertex and multiplying the distance of the vertex furthest away by two. This approximation algorithm will in the worst case output a value twice as large as the actual diameter. The time complexity of the algorithm is $O(n+m)$ (one BFS) which is significantly better than the exact algorithm ($O(nm)$, APSP). The algorithm is a so-called 2-approximation because it returns a value at worst twice as large as the actual diameter.

The use of randomization is another approach for reducing the complexity of an algorithm. There are two types of randomization algorithms: Monte Carlo and Las Vegas.

A Las Vegas algorithm always outputs the correct result, but the execution times vary as a consequence of the randomization. An example is quicksort in which a pivot element is typically randomized. If the randomized pivot element turns out to always be the smallest in the collection (or greatest), the time complexity of quicksort will be $O(n^2)$, but in the average case the algorithm has a $O(n \log n)$ time complexity. Notice that even if the randomized pivot element in the collection was a poor choice, quicksort will always return a sorted collection.

A Monte Carlo algorithm is a randomized algorithm whose output is incorrect with some (usually small) probability. As a consequence one can never guarantee to

output the correct value. Comparing Monte Carlo to Las Vegas, the execution time does not vary as a consequence of the randomization, but as stated, the output might be incorrect.

2.2 Basic graph theory

In this thesis we consider only undirected, unweighted and connected graphs with no loops. A path is a sequence of vertices where any two consecutive vertices are connected by an edge. A geodesic path between two vertices p and q, denoted p-q, is the shortest path between vertex p and q. A complete graph is a graph where there exists an edge from every vertex i to every other vertex j. A subgraph S of G is a graph which contains a subset of the vertices and edges of G. A biconnected graph is still connected even if any one vertex is removed. A block (or biconnected component) is a maximal biconnected subgraph.

The diameter of a graph $G = (V, E)$ is defined as the length of the longest of all shortest path between any two nodes. Let $diam(G)$ denote the diameter of a graph G, define $\delta = \frac{diam(G)}{n}$. One can think of δ as the fraction of vertices on the longest geodesic path with respect to the number of vertices in the graph. We define $d_G(u, v)$ to be the length of a geodesic path between vertex u and v in the graph G, although we omit the subscript G if it is clear from context. If a vertex p is unreachable from another vertex q, we denote the distance between p and q as infinity, $d(p, q) = \infty$.

A layer N_k with respect to a reference vertex r is defined as the set: $N_k(r) = \{v \mid d(r, v) = k\}$. The k:th layer from a reference vertex r can be interpreted as the set which contains the vertices with a distance k from r. The depth of a reference vertex r is defined as $\max \{d(r, v) \mid v \in V\}$. The depth of r can be interpreted as the distance to a vertex furthest away from r. The degree of a vertex v is defined as the number of vertices adjacent to v. A hair is defined as a path with one endpoint having degree one, the other endpoint having a degree larger than two and the remaining vertices having degree two. $B(c, u, w)$ holds if $d(c, w) = d(c, u) + d(u, w)$, where u, c and w are vertices of a graph. One can think of $B(c, u, w)$ as being true if the vertex u is "between" vertices c and w, that is, a shortest path from c to w intersects u.

A separator is a set $S \subset V$, such that G is split into two or more connected components when removing all vertices in S from G and all incident edges of S have been removed from G. An articulation point is a separator of size one. Note that every layer from a given reference vertex forms a separator. In Figure 2.1 the highlighted (blue) vertices on the left graph form a separator. The separator (b,e,h) disconnects the vertices a, d and g from the vertices c, f and i. In the graph on the right in Figure 2.1, the highlighted vertices (green) c, e and g form the layer $N_2(a)$. Notice that $N_2(a)$ forms a separator, in this case disconnecting the vertices a, b and d from the vertices f, h and i. Even though every layer is always a separator, it is important to notice that every separator must not be a layer, as can be seen in the left graph

in Figure 2.1, where each of the highlighted vertices (b,e,h) are in different layers from the reference vertex a.

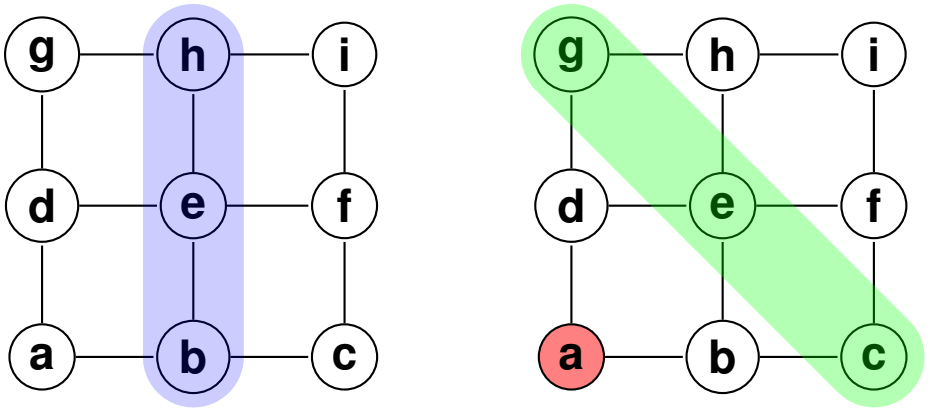

Figure 2.1: In the graph on the left, the highlighted vertices (b,e,h) form a separator in the graph. Removing these vertices will disconnect the the vertices a,d and g from the vertices c,d and i. In the graph on the right, the highlighted (green) vertices (c,e,g) are the layer with distance two from the reference vertex a (marked with red). Notice that the layer forms a separator.

A block cut tree is a graph G = (V,E), where every $v \in V$ is either an articulation point or a biconnected component. Every articulation point in the block-cut tree has edges connecting it to one or several biconnected components. In Figure 2.2, 1 (the graph on the right in Figure 2.2) refers to the biconnected component containing vertices a, b and c. The vertex 2 of the block-cut tree refers to the biconnected component containing vertices e, f and g. The vertex d is the root of the block-cut tree and the only articulation point of this particular example.

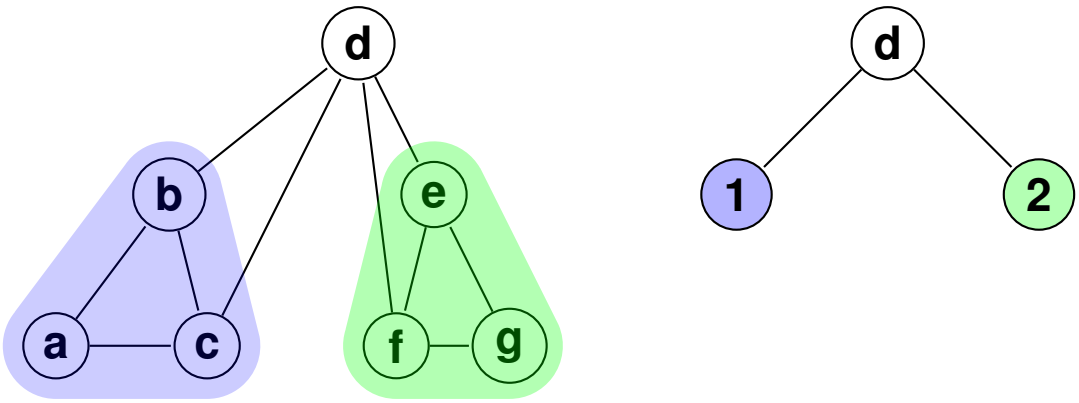

Figure 2.2: The left depicts the original graph and the right is the block-cut tree of that particular graph.

2.3 Definitions

To reason formally about different concepts without ambiguity, such as quantities of layers, this subsection introduces various definitions needed for the results.

For a graph G = (V,E) and vertex $v \in V$ and $i, b \in \mathbb{N}$, we define $\psi(i, v, b)$ as:

$$\psi(i, v, b) = \begin{cases} 1 : |N_i(v)| = b \\ 0 : |N_i(v)| \neq b. \end{cases}$$

One can think of $\psi(i, v, b)$ as indicating if the size of the i:th layer from the reference vertex v ($|N_i(v)|$) is equal to the integer b. One can count the total number of layers from the reference vertex v using ψ in the following way: $\sum_{b=1}^{n} \sum_{i=1}^{\delta n} \psi(i, v, b)$.

For a graph G = (V,E) with vertices $p, q \in V$ and a separator S, we define $\Psi(S, p, q)$ as:

$$\Psi(S, p, q) = \begin{cases} 1 : d_{G-S}(p, q) = \infty \\ 0 : d_{G-S}(p, q) \neq \infty. \end{cases}$$

One can think of $\Psi(S, p, q)$ as indicating if a separator S disconnects vertices p and q (d(p,q) = ∞) in a graph G.

For a graph G = (V,E) with vertices $p, q, v \in V$ and $i, b \in \mathbb{N}$, we define Γ as:

$$\Gamma(p, q, i, v, b) = \begin{cases} 1 : \psi(i, v, b) = 1 \wedge \Psi(N_i(v), p, q) = 1 \\ 0 : \neg\Big[\psi(i, v, b) = 1 \wedge \Psi(N_i(v), p, q) = 1\Big]. \end{cases}$$

One can think of $\Gamma(p, q, i, v, b)$ as indicating if the i:th layer of the reference vertex v is of a particular size b and separates the two vertices p and q in the graph G.

We define min(p) = x_a ($1 \leq a \leq b$), where p is a b-dimensional vector $p = (x_1, x_2, \ldots, x_b)$, such that $\forall i \; x_a \leq x_i$.

In this thesis we are only concerned with the uniform random model. When we mention "at random", "randomness" etc, we do implicitly mean that all the events in the sample space have the same probability.

When we mention that a vertex v has x amount of layers, we do implicitly mean that there exists a vertex w such that $d(v, w) = x$.

2.4 Difficulties with computing the diameter

Prior to describing how one can compute the diameter in various slim graphs faster than the general case, it is worth describing some reason why the diameter can perhaps be difficult to compute.

Consider a graph without cycles, namely a tree. In such a graph one can compute the diameter in the following way [10]: choose an arbitrary vertex v, perform a BFS from v and choose a vertex w furthest away from v. From w one can find the diameter by computing the distance (using BFS) to a vertex furthest away from w.

The mentioned approach does however not work in graphs with cycles, as cycles can create shortcuts. Consider the graph in Figure 2.3 in which there exists cycles and the tree approach no longer works.

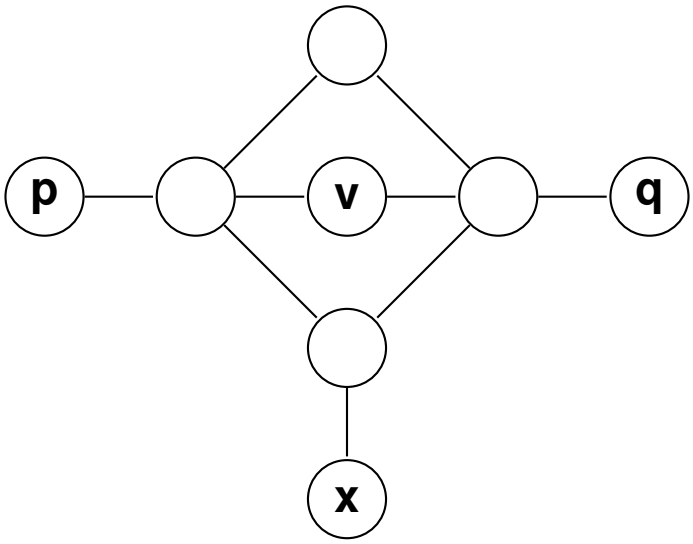

Figure 2.3: An example of a graph where the furthest vertex away from an arbitrary vertex (in this example the vertex v) is not an end point of a longest geodesic path. The path p-v-q is a longest geodesic path. The vertex x is not an endpoint of a longest geodesic path.

In Figure 2.3, assume that we chose the vertex v. The vertex furthest away from v is the vertex x with a distance (d(v,x)) of 3. Vertex x has at most a distance of 4 to every other vertex in the graph. However, a longest geodesic path in this particular graph has a length of 5 (the p-v-q path). Thus the tree approach does not work in the general case as demonstrated with the graph in Figure 2.3, which contains cycles.

Assume that one wants to create a longest geodesic path containing an arbitrary vertex v. Consider performing a BFS from the vertex v to find two vertices p and q which are the furthest away from v. One might consider creating a longest geodesic path including v by choosing the path p-v-q. However, there is a problem of orientation: p and q are furthest away from v but p and q can still (in the worst case) be neighbours ($d(p, q) = 1$). Thus, p-v-q was not even a geodesic path. Figure 2.4 illustrates such a graph:

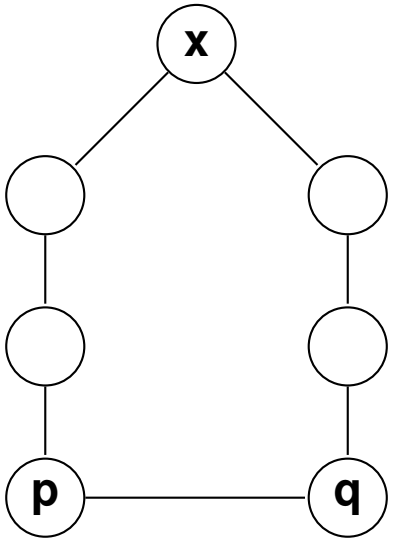

Figure 2.4: A graph illustrating the problem of orientation from a BFS. Vertices p and q are the furthest away from vertex v, but the path p-v-q is not geodesic, and d(p,q) = 1.

However, if a vertex x is an articulation point and a longest geodesic path containing x intersects two connected components, then we can find a longest geodesic path which contains x by measuring the depth of every connected component separated by x, and connect the two deepest with x. Consider Figure 2.5, where a vertex x splits a graph into k connected components.

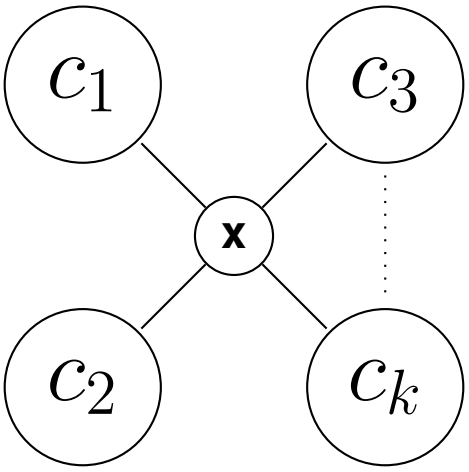

Figure 2.5: A graph illustrating an articulation point x which splits the graph into k connected components.

2.5 Slim graphs

For slim graphs with $\delta > 1/2$, an algorithm for diameter computation in $O(n + m)$ time has been presented by Damaschke [2]. This result is a significant improvement compared to APSP. Graphs with $\delta > 1/2$ always contain at least one articulation

point. As δ gets smaller the structure of the graphs changes. Graphs with $\delta \leq 1/2$ do no longer have to contain an articulation point. Since the algorithm in the case of $\delta > 1/2$ is based on the existence of at least one articulation point, a new approach is needed. In similar fashion to the case $\delta > 1/2$, Damaschke [2] has proven the existence of separators of size at most 2 for graphs with $\delta > 1/3$.

The troublesome part with $\delta > 1/3$ is that a longest geodesic path can intersect a separator of size two. If a longest geodesic path intersects a separator of size two, then we do not know which of the two vertices lies on the path. Damaschke [2] present a Monte Carlo algorithm which computes the diameter with high probability in $O(m + n \log n)$ time.

The following two subsections will describe these two algorithms. We also explain an auxiliary problem, largest mixed sum. The diameter of graphs with $\delta > 1/3$ can be computed by setting up and solving an instance of largest mixed sum.

2.5.1 Computing the diameter in graphs with $\delta > 1/2$

In this subsection we explain the algorithm for computing diameters in graphs with $\delta > 1/2$ presented by Damaschke [2].

We start by creating a block-cut tree, which can be done in $O(n)$ time [5]. One can create a block-cut tree since there exists at least one articulation point. If every connected component from the root vertex in the block-cut tree has fewer than δn vertices, then a longest geodesic path must intersect the articulation point and two connected components. This is true because no component has as many vertices as the diameter, so trivially we have to connect two different components. Thus, in this case, it is sufficient to do a BFS from the articulation point in order to find the longest geodesic path. One can do so by first searching one component, then the second one and so on, and finally combine the two which give the longest distance.

We can remove all vertices in a connected component that are not on a longest geodesic path, the removal will not alter the diameter of the whole graph. This observation is needed in the second case.

If there exists at least one connected component with at least δn vertices then we can not compute the diameter by performing a BFS from an articulation point. Instead, we keep the component with largest depth and from all other components we keep a longest geodesic path. Constructing this new graph results in a connected component with hairs. A longest geodesic path must have one of its endpoints in either the longest or second longest hair. Thus it is sufficient to perform two BFS from the two respective hairs in order to compute the diameter.

2.5.2 Largest Mixed Sum

In the this subsection we describe 2-dimensional largest mixed sum, which is a problem presented and solved by Damaschke [2] in $O(n \log n)$ time. The problem formulations is as following: given n number of pairs (x_i, y_i) and m number of pairs (x'_j, y'_j), the task is to maximize the following expression: min $\{x_i + y'_j, y_i + x'_j\}$.

To understand why this is relevant for diameter computation consider the following example: define G as a graph consisting of two subgraphs C and D which are connected with the paths: u-v and u'-v' (see Figure 2.6). In this particular graph G, a geodesic path between a vertex $p \in C$ to a vertex $q \in D$ can be computed by four executions of BFS from the vertices u, u', v and v'. Either u-v or u'-v' is a subpath of the geodesic path p-q. d(p,q) is thus the shortest of the following two paths: p-u-v-q or p-u'-v'-q (which can be computed by the four previously mentioned executions of BFS). If one wants to compute a longest geodesic path from vertices $p \in C$ to vertices $q \in D$ one can solve the 2-dimensional largest mixed sum. We can set up an instance of largest mixed sum by creating the pairs (x_i, y_i) and (x'_j, y'_j), where $x := d(p, u)$, $y := d(p, u')$, $y' := d(q, v)$ $x' := d(q, v')$. To compute a longest geodesic path between a vertex $p \in C$ to $q \in D$ we maximize the following expression: min $\{x_i + y'_j, y_i + x'_j\}$. This is precisely the 2-dimensional largest mixed sum, which is solvable in $O(n \log n)$ time.

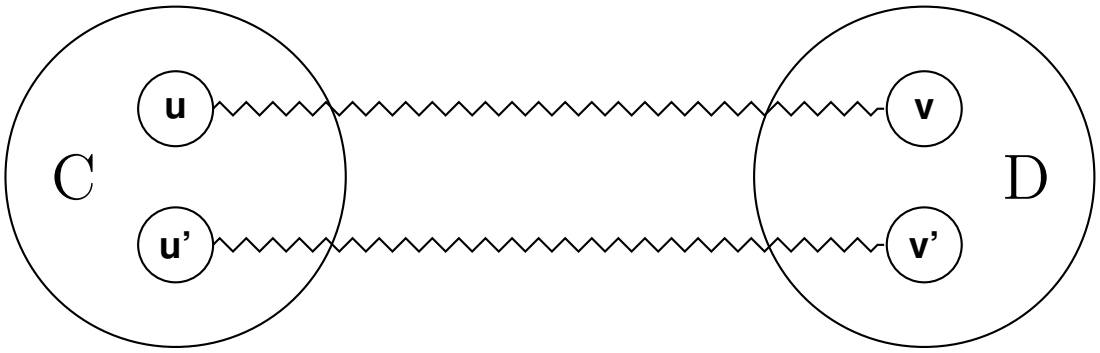

Figure 2.6: The graph G is depicted as its two subgraphs: C and D with its corresponding connections (u-v and u'-v'). A longest geodesic path connecting a vertex $p \in C$ to $q \in D$ can be computed by 2-dimensional largest mixed sum.

2.5.3 Computing the diameter in graphs with $\delta > 1/3$

In this subsection we explain the algorithm for computing diameters in graphs with $\delta > 1/3$ presented by Damaschke [2]. Graphs with $\delta > 1/3$ do not have to contain an articulation point, we can therefore not use the approach mentioned in section 2.5.1.

Let P be a longest geodesic path. Randomly choose the vertices u, v and w. The following happens with constant probability in the lower bound (the probability of choosing vertices in different layers is constant in the lower bound and therefore the probability is at least this constant factor): u, v and w are in a layer (the reference vertex is an endpoint of a longest geodesic path) of size at most 2; $u \in N_i(r)$,

$v \in N_j(r)$, $w \in N_k(r)$, where $i < j < k$. Since a longest geodesic path has to intersect all the layers, P contains u and w with constant probability. Let Q be a subpath of P with endpoints in u and w. We have the following two cases with constant probability:

(1) $|\mathbf{N_j(r)}| = \mathbf{1}$: In this case, v is an articulation point, thus $v \in P$. Remember that P has to intersect every layer and the layer $N_j(r)$ only contains v, P therefore must go through v.
(2) $|\mathbf{N_j(r)}| = \mathbf{2}$: In this case, $v \notin Q$ holds with constant probability. Again, P must intersect $N_j(r)$ which in this case contains two vertices, thus P either goes through v or the other vertex.

We do not know if $|N_j| = 1$ or $|N_j| = 2$. If v is an articulation point then (1) can be true. We perform a BFS from v and if it was the case that $|N_j| = 1$ then we know that the path returned will be a longest geodesic path. However, in the case of $|N_j| = 2$ the BFS may return P, but the important thing is that we only return a geodesic path between two vertices in two different components. Thus a path longer than the diameter is never returned.

If v is not an articulation point, then $|N_j| = 2$ and we suppose (2) is true (Figure 2.7 shows the case when $|N_j| = 2$). All vertices $v_i \in P$ and $v_i \notin Q$ satisfy either $B(v_i, u, w)$ or $B(u, w, v_i)$. Define $C := \{v_i \mid B(v_i, u, w) \wedge v_i \in P\}$ and $D := \{v_i \mid B(u, w, v_i) \wedge v_i \in P\}$. These two sets can be constructed by two BFS with roots u and w. Set up an instance of largest mixed sum where the distances x, x', y and y' are the distances of vertices in C and D to v and w. These distances can be found by performing a BFS from v (the distances to w have already been computed by a previous BFS). Largest mixed sum will return vertices corresponding to endpoints in C and D. Let $p \in C$ and $q \in D$ denote the endpoints of a longest path (reconstructed from largest mixed sum). It is important to never return a value greater than d(p,q), but this never the case because the only way to reach D from C (or vice versa) requires a path to intersect either Q or v. If u and w happened to be on a longest geodesic path then the diameter is always returned.

One can repeat these procedures a constant number of times to get the probability of returning the diameter arbitrarily close to 1.

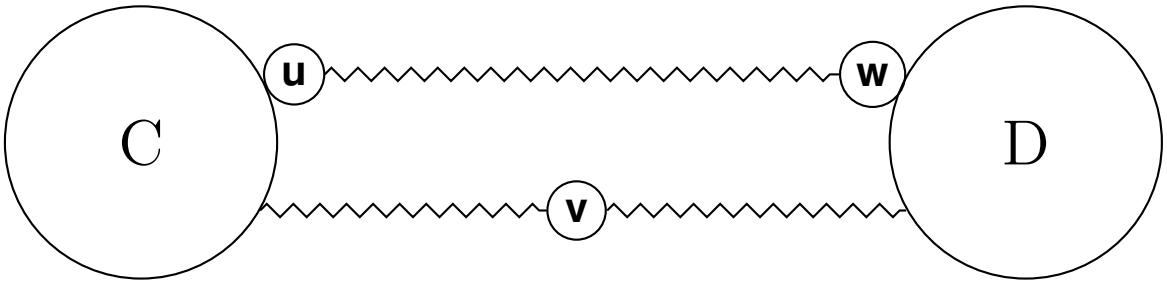

Figure 2.7: Shows the vertices u, v, w and the sets C and D containing the endpoints.

3
Results

Lemma 1. Given a graph $G = (V, E)$ with a longest geodesic p-q path P, the following holds: $\forall v\ \forall z\ v \in P \wedge z \notin P$: $d(z, p) + d(z, q) \geq d(v, p) + d(v, q)$.

Proof. Assume that there exists $w \notin P$ such that $d(w, p)+d(w, q) < d(v, p)+d(v, q)$. In that case we can create a shorter path (from p to q) by choosing the path p-w-q. This contradicts the assumption that P was a longest geodesic path. □

In the following Lemmas we make use of the functions ψ and Ψ. Remember that ψ is used for indicating if a given layer is of a specific size, and Ψ is used for indicating if a given separator disconnects two vertices. Both are defined in section 2.3.

Lemma 2. Given a graph $G = (V, E)$, the following holds: $\forall v\ \exists i\ |N_i(v)| \leq 2/\delta$.

Proof. Let P be a longest geodesic p-q path. The following holds for $v \in P$: $d(v, p) \geq \delta n/2$ or $d(v, q) \geq \delta n/2$. For every vertex $z \notin P$, by Lemma 1, z has a distance of at least $\delta n/2$ to either p or q. Since the following holds for every vertex v: $\sum_{b=1}^{\delta n} \sum_{i=1}^{\infty} \psi(i, v, b) \geq \delta n/2$, we conclude by the pigeonhole principle that there exists at least one layer with at most $2/\delta$ vertices. □

Lemma 3. Given a graph $G = (V, E)$ with n vertices, we have $\forall v \forall i\ |N_i(v)| \leq n - n\delta/2$.

Proof. From Lemma 2, the following holds for every vertex v: $\sum_{b=1}^{\delta n} \sum_{i=1}^{\infty} \psi(i, v, b) \geq \delta n/2$. To create the largest possible layer we let $n\delta/2 - 1$ layers be articulation points. Let the remaining vertices be in the largest layer, thus the largest layer can at most contain $n - \delta n/2$ vertices. □

Lemma 4. Given a graph $G = (V, E)$ with a longest geodesic $p - q$ path P, for every $v \in P$, the following holds: $\forall i\ |N_i(v) \cap P| \leq 2$.

Proof. Assume for contradiction $\exists i\ |N_i(v) \cap P| > 2$, that is, P intersects the i:th layer k times, where $k > 2$. Let the first intersecting vertex on $N_i(v)$ be $v_1 \in P$ and the last intersecting vertex be $v_k \in P$, thus $N_i(v) \cap P = \{v_1, \ldots, v_k\}$. We have only three possible cases when traversing the path P from p to q: (i) v is somewhere between the first and the last intersection, (ii) v is located before the first intersection and (iii) v is located after the last intersection.

We describe what each of the following cases imply, when traversing the longest geodesic p-q path P:

(i) One will reach v_1, then reach v after h intersections, and finally traverse the remaining k-h-1 intersecting vertices, that is:
P := $\{\dots, v_1, \dots, v_{h+1}, \dots, v, \dots, v_j, \dots, v_k, \dots\}$ (v_j only exists if $k - h - 1 \geq 2$).

(ii) One will first reach (or start) in v and from v, one will eventually reach v_1 and the remaining intersecting vertices, that is: P := $\{\dots, v, \dots, v_1, \dots, v_k, \dots\}$.

(iii) One will reach v_1 first, then reach the remaining $k-1$ intersecting vertices, that is: P := $\{\dots, v_1, \dots, v_k, \dots, v, \dots\}$.

Consider the proofs for each case:
(i) If $h > 0$, the distance is $d(v_{h+1}, v) = d(v_1, v) = i$, as v_{h+1} and v_1 are on the same layer. However, if $h = 0$ then $d(v, v_j) = d(v, v_k) =$ i. We choose the subpath $v_1 - v - v_k$, creating a shorter path. Thus P was not a longest geodesic path, contradiction. Figure 3.1 shows an instance of case (i) when $k = 3$.

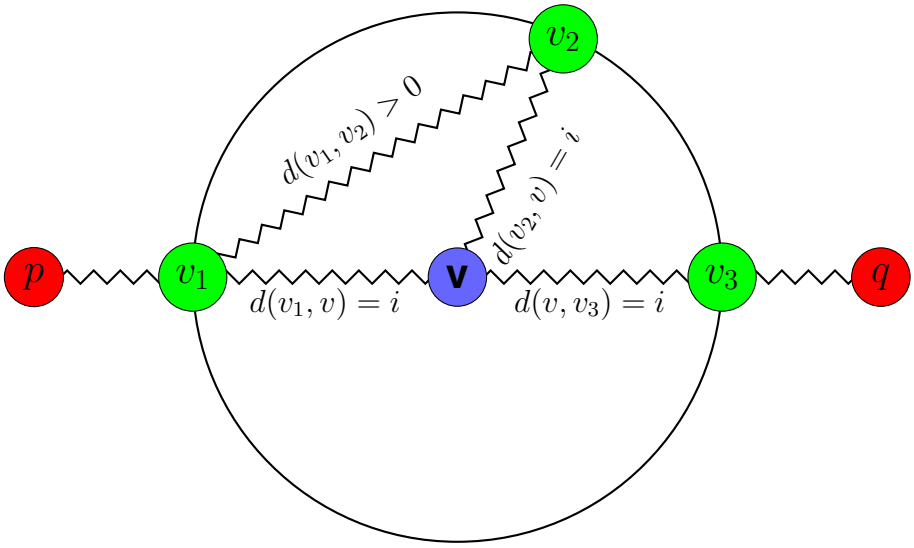

Figure 3.1: An instance of case (i) when $k = 3$. The length of the subpath with three intersections, $v_1 - v_2 - v - v_3$, is greater than 2i. One can create a shorter path by choosing the subpath with two intersections instead, $v_1 - v - v_3$, which length is exactly 2i.

(ii) The distances $d(v, v_1)$ and $d(v, v_k)$ are equal, as the vertices v_1 and v_k are in the same layer. We choose the subpath $v - v_k$ which creates a shorter path. Thus P was not a longest geodesic path, contradiction. Figure 3.2 shows the instance of (ii) when $k = 3$.

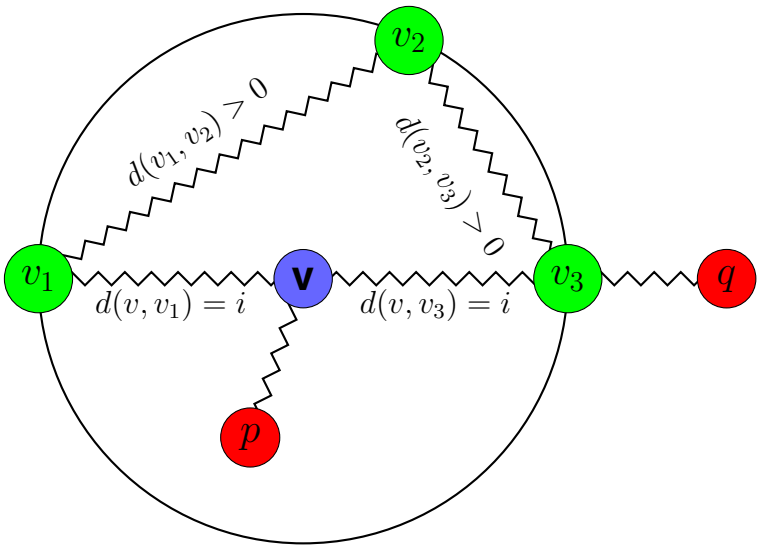

Figure 3.2: An instance of case (ii) when $k = 3$. The length of the subpath with three intersections, $v - v_1 - v_2 - v_3$, is greater than i. One can create a shorter path by choosing the subpath with one intersection $v - v_3$, which length is exactly i.

(iii) The distances $d(v_1, v)$ and $d(v_k, v)$ both equal i, as the vertices v_1 and v_k are in the same layer. We choose the subpath $v_1 - v$, creating a shorter path. Thus P was not a longest geodesic path, contradiction. Figure 3.3 shows an instance of (iii) when $k = 3$. These contradictions conclude the proof. □

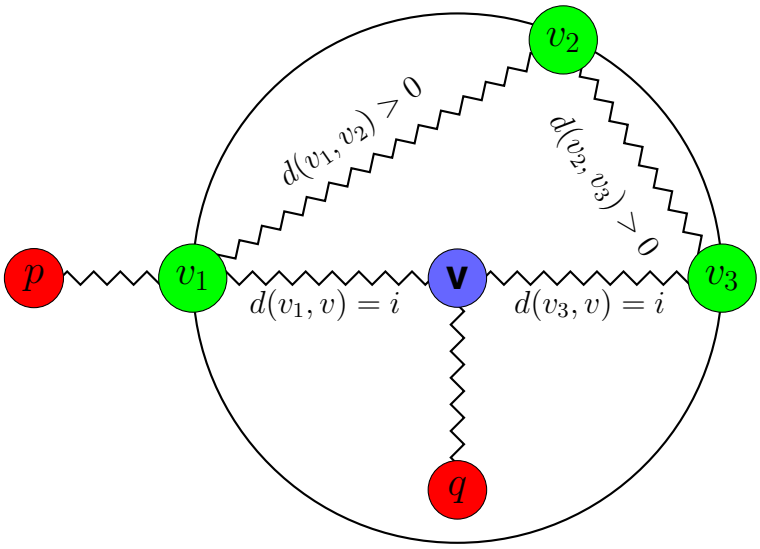

Figure 3.3: An instance of case (iii) when $k = 3$. The length of the subpath with three intersections, $v_1 - v_2 - v_3 - v$, is greater than i. One can create a shorter path by choosing the subpath with one intersection $v_1 - v$, which length is exactly i.

Note that every layer of any vertex on a longest geodesic path can at most be intersected twice by that particular longest geodesic path. Thus the total amount of intersected layers is at least $\delta n/2$.

Lemma 5. Given a graph $G = (V, E)$ with a longest geodesic path P, if $\forall u\ \forall w$ $u, w \in N_i(v) \wedge v \in P$, then $d(u, w) < 2i \implies \{u, w\} \not\subset P$.

Proof. Let p be an endpoint of P. A consequence of Lemma 4 is the following: $|N_i(v)| = 2 \implies p \notin \cup_{k=0}^{i-1} N_k(v)$. Assume for contradiction $u, w \in P$ and $d(u, w) < 2i$. From Lemma 4 we have the case that v is traversed between u and w by P. We can create a shorter P by choosing the subpath $u - w$ instead of the current $u - v - w$. Thus v is not on P, which is a contradiction. □

We define $0 < k < 1/2$ as follows: for any vertex $v \in P$, where P is a longest geodesic $p - q$ path, the distance of v to one endpoint (p or q) is $k\delta n$. We can now bound tighter (than Lemma 2) how many layers a vertex on a longest geodesic path at least has. In particular, the number of layers for any $v \in P$ is at least $(1 - k)\delta n$.

Lemma 6. Let P be a longest geodesic $p - q$ path, given a vertex $v \in P$ and a $0 < k < 1/2$ such that $\Big[d(v, p) = (1 - k)\delta n\Big] \vee \Big[d(v, q) = (1 - k)\delta n\Big]$, the following is true:
$\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \psi(i, v, b) > (1 - k)\delta n - n/(1 + 2/\delta)$. That is, the amount of layers from the reference vertex v that have a size of at most $\lceil 2/\delta \rceil$ is greater than
$(1 - k)\delta n - n/(1 + 2/\delta)$.

Proof. The amount of layers ($N_i(v)$) with size at most $\lceil 2/\delta \rceil$ is the number of layers with size at least $\lceil 2/\delta \rceil + 1$ subtracted from the total amount of layers (see eq. (3.1)).

$$\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \psi(i, v, b) = \sum_{b=1}^{n} \sum_{i=1}^{\infty} \psi(i, v, b) - \sum_{b=\lceil 2/\delta \rceil + 1}^{n} \sum_{i=1}^{\infty} \psi(i, v, b) \tag{3.1}$$

The total amount of layers is at least $(1 - k)\delta n$, as by assumption
$d(v, p) = (1 - k)\delta n \vee d(v, q) = (1 - k)\delta n$.

$$(1 - k)\delta n \leq \sum_{b=1}^{n} \sum_{i=1}^{\infty} \psi(i, v, b) \tag{3.2}$$

If one distributes all n vertices among layers of size $\lceil 2/\delta \rceil + 1$, then the vertex v would have exactly $n/(1 + \lceil 2/\delta \rceil)$ layers. Since there must exist a layer of size at most $2/\delta$ (Lemma 2) we can conclude the following:

$$\sum_{b=\lceil 2/\delta \rceil + 1}^{n} \sum_{i=1}^{\infty} \psi(i, v, b) < \frac{n}{1 + \frac{2}{\delta}}. \tag{3.3}$$

Using the results from eq. (3.2) and eq. (3.3) in eq. (3.1) we get the result in eq. (3.4).

$$\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \psi(i, v, b) > (1 - k)\delta n - \frac{n}{1 + \frac{2}{\delta}} \tag{3.4}$$

□

Lemma 7. Given a graph $G = (V, E)$ with a longest geodesic $p - q$ path P, for any vertex $v \in P$ and a $0 < k < 1/2$ such that $d(v, p) = k\delta n \vee d(v, q) = k\delta n$, the following holds:
$\sum_{i=1}^{\infty} \Psi(N_i(v), p, q) \geq (\delta - 2k\delta)n - 2$. That is, the amount of layers $N_i(v)$ which separate p and q is at least $(\delta - 2k\delta)n - 2$.

Proof. Without loss of generality, assume p is the closest endpoint (of P) to v ($d(v, p) < d(v, q)$). Since $d(v, p) = k\delta n$, we know $p \in N_{k\delta n}(v)$ and $d(v, q) = (1-k)\delta n$, thus $q \in N_{(1-k)\delta n}(v)$. Any layer L "between" $N_{k\delta n}(v)$ and $N_{(1-k)\delta n}(v)$ ($L = N_j(v)$, where $k\delta n < j < (1-k)\delta n$), separates p and q ($d_{G-L}(p, q) = \infty$). The amount of layers L is: $(1-k)\delta n-1-(k\delta n+1)$. We do not include $N_{k\delta n}(v)$ (hence the $+1$) because that particular layer contains p, removing the vertices in $N_{k\delta n}(v)$ from G would remove p from the graph G. We do neither include $N_{(1-k)\delta n}(v)$ for the same reason (exchange p for q), thus: $\sum_{i=1}^{\infty} \Psi(N_i(v), p, q) \geq (1-k)\delta n-1-(k\delta n+1) = (1-2k)\delta n-2$. □

A consequence of Lemma 7 is that a given vertex on a longest geodesic path with a distance of at most $k\delta n$ to one endpoint has at least $(\delta - 2k\delta)n - 2$ layers which disconnect the endpoints of a longest geodesic path. If such a separator (of constant size) can be found, we can find a longest geodesic path by solving an instance of b-dimensional largest mixed sum.

Lemma 8. The inequality $(1 - k)\delta n - \frac{n}{1+\frac{2}{\delta}} - (\delta n - \Big[(1 - 2k)\delta n - 2\Big]) > 0$, where $n > 0$ and $\delta > 0$, has a solution for $k < \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n}$.

Proof. We want to show that the inequality
$(1 - k)\delta n - \frac{n}{1+\frac{2}{\delta}} - (\delta n - \Big[(1 - 2k)\delta n - 2\Big]) > 0$ has a solution for $k < \frac{\delta+1}{3\delta+6}$. We do this by algebraic manipulation:

$$
\begin{aligned}
(1 - k)\delta n - \frac{n}{1 + \frac{2}{\delta}} - (\delta n - \Big[(1 - 2k)\delta n - 2\Big]) &> 0 \\
(1 - k)\delta n - \frac{n}{1 + \frac{2}{\delta}} - \delta n + (1 - 2k)\delta n - 2 &> 0 \\
(1 - k)\delta n - \frac{n}{1 + \frac{2}{\delta}} - \delta n + (1 - 2k)\delta n &> 2 \\
n\Big[(1 - k)\delta - \frac{1}{1 + \frac{2}{\delta}} - \delta + (1 - 2k)\delta\Big] &> 2 \\
n\Big[(1 - k)\delta - \frac{\delta}{2 + \delta} - \delta + (1 - 2k)\delta\Big] &> 2 \\
n\delta\Big[(1 - k) - \frac{1}{2 + \delta} - 1 + (1 - 2k)\Big] &> 2 \\
1 - k - \frac{1}{2 + \delta} - 1 + 1 - 2k &> \frac{2}{\delta n} \\
1 - k - \frac{1}{2 + \delta} - 2k &> \frac{2}{\delta n}
\end{aligned}
$$

$$
\begin{aligned}
1 - 3k - \frac{1}{2+\delta} &> \frac{2}{\delta n} \\
-3k &> \frac{2}{\delta n} + \frac{1}{2+\delta} - 1 \\
k &< \frac{1}{3} - \frac{2}{3\delta n} - \frac{1}{3(2+\delta)} \\
k &< \frac{2+\delta}{3(2+\delta)} - \frac{2}{3\delta n} - \frac{1}{3(2+\delta)} \\
k &< \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n}.
\end{aligned}
$$

□

In Lemma 9, we use the function Γ which is defined in section 2.3, $\Gamma(p, q, i, v, b)$ can be interpreted as indicating if $N_i(v)$ of a particular size b separates the two vertices p and q in a graph G.

Lemma 9. Given a graph $G = (V, E)$ with a longest geodesic $p - q$ path P, for any $v \in P$ with $d(v, p) = k\delta n \vee d(v, q) = k\delta n$, where $0 < k < \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n}$, the following holds for a constant $c > 0$: $\dfrac{\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \Gamma(p, q, i, v, b)}{n} \geq c$.

Proof. When we mention: the amount of layers, total amount of layers etc, we refer to layers with reference vertex v ($N_i(v)$).

The amount of layers, which are of size $\lceil 2/\delta \rceil$ or smaller, is given from Lemma 6:

$$
\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \psi(i, v, b) > \left[(1-k)\delta n - \frac{n}{1 + \frac{2}{\delta}} \right]. \tag{3.5}
$$

We subtract the layers which separate p and q (which are $(1-2k)\delta n - 2$ from Lemma 7) from the total amount of layers (which are at most δn), and get the following expression (the greatest amount of layers which do not separate p and q):

$$
\delta n - \Big[(1-2k)\delta n - 2 \Big]. \tag{3.6}
$$

We subtract eq. (3.6) from eq. (3.5) and get the least amount of layers (which are of size $\lceil 2/\delta \rceil$ or smaller) which separates p and q:

$$
\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \Gamma(p, q, i, v, b) \geq (1-k)\delta n - \frac{n}{1 + \frac{2}{\delta}} - (\delta n - \Big[(1-2k)\delta n - 2 \Big]). \tag{3.7}
$$

We want the amount of layers, which are of size $\lceil 2/\delta \rceil$ and separate p and q, to be strictly greater than 0. We set up the equation 3.8:

$$
\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \Gamma(p, q, i, v, b) \geq (1-k)\delta n - \frac{n}{1 + \frac{2}{\delta}} - (\delta n - \Big[(1-2k)\delta n - 2 \Big]) > 0. \tag{3.8}
$$

Equation (3.8) has a solution for the following values of k (proved in Lemma 8):

$$k < \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n}. \tag{3.9}$$

Since k has to be greater than 0, we conclude the following:

$$\begin{aligned}
\frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n} &> 0 \\
\frac{\delta+1}{3\delta+6} &> \frac{2}{3\delta n} \\
n\frac{\delta+1}{3\delta+6} &> \frac{2}{3\delta} \\
n &> \frac{2(3\delta+6)}{3\delta(\delta+1)} \\
n &> \frac{2(\delta+2)}{\delta(\delta+1)}.
\end{aligned}$$

Since $n > \frac{2(\delta+2)}{\delta(\delta+1)}$, the lowest possible value for n is therefore $\frac{2(\delta+2)}{\delta(\delta+1)}+1$. We substitute n with $\frac{2(\delta+2)}{\delta(\delta+1)}+1$ in eq. (3.9) to get the following inequality:

$$\begin{aligned}
k &< \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta(\frac{2(\delta+2)}{\delta(\delta+1)}+1)} \\
k &< \frac{\delta+1}{3\delta+6} - \frac{2}{\frac{6(\delta+2)}{\delta+1}+1} \\
k &< \frac{\delta+1}{3\delta+6} - \frac{2}{\frac{6(\delta+2)}{\delta+1}+\frac{\delta+1}{\delta+1}} \\
k &< \frac{\delta+1}{3\delta+6} - \frac{2}{\frac{7\delta+13}{\delta+1}} \\
k &< \frac{\delta+1}{3\delta+6} - \frac{2}{\frac{7\delta+13}{\delta+1}} \\
k &< \frac{\delta+1}{3\delta+6} - \frac{2(\delta+1)}{7\delta+13} \\
k &< \frac{(\delta+1)^2}{3(\delta+2)(7\delta+13)}.
\end{aligned}$$

Notice that $k < \frac{(\delta+1)^2}{3(\delta+2)(7\delta+13)}$ is a lower bound on the value of k, since we substituted n for the smallest possible value. From eq. (3.9) we have: $\lim_{n\to\infty}\left[k < \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n}\right] = k < \frac{\delta+1}{3\delta+6}$. Thus as n gets greater, the bound on k becomes greater. Furthermore, eq. (3.8) can be rewritten as:

$$\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \Gamma(p,q,i,v,b) \geq n\Big[(1-k)\delta - \frac{1}{1+\frac{2}{\delta}} - (\delta - (1-2k)\delta)\Big] > 2 \Leftrightarrow$$

$$\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \Gamma(p,q,i,v,b) \geq nc > 2. \quad (3.10)$$

From eq. (3.10), we have: $\frac{\sum_{b=1}^{\lceil 2/\delta \rceil} \sum_{i=1}^{\infty} \Gamma(p,q,i,v,b)}{n} \geq c > \frac{2}{n}$, where $c > 0$ is a constant. □

Lemma 10. Given a graph $G = (V, E)$ with a longest geodesic $p-q$ path P, for any vertex $v \in P$, the following holds: $\Big(d(v,p) < i\Big) \wedge \Big(d(v,q) > i\Big) \implies |P \cap N_i(v)| = 1$.

Proof. P consists of the two subpaths: $p - v$ and $v - q$. We prove the statement of the Lemma by showing $|(p - v) \cap N_i(v)| = 0$ and $|(v - q) \cap N_i(v)| = 1$, thus $|P \cap N_i(v)| = 1$. Below follows a case for each subpath:

(i) $p - v$: Assume one intersecting vertex v_1 (i.e $v_1 \in N_i(v) \wedge v_1 \in (p - v)$). We know $d(p, v_1) > 0$ and $d(v_1, v) = i$, but $d(p, v) < i$ holds by the assumption in the statement of Lemma 10, thus choose the subpath $p - v$ instead of $p - v_1 - v$ which creates a shorter path. Thus, P was not a geodesic path, leading to a contradiction. Trivially, having more than one intersecting vertex in the subpath $p - v$ will make the subpath $p - v$ longer, we conclude: $|(p - v) \cap N_i(v)| = 0$.

(ii) $v - q$: Assume $k > 1$ intersecting vertices in the subpath $v - q$ ($\{v_1, \ldots, v_k\} \subset (v - q) \wedge |\{v_1, \ldots, v_k\} \cap N_i(v)| = k$), the subpath can be written as $v - v_1 - v_k - q$. Since $d(v, v_1) = d(v, v_k)$ and $\forall i \forall j \; i \neq j \implies d(v_i, v_j) > 0$, we choose the subpath $v - v_k - q$ instead, creating a $p - q$ path shorter than P and $|N_i(v) \cap p - q| = 1$. Thus, P was not a geodesic path, leading to a contradiction.

To conclude, $|P \cap N_i(v)| \neq 0$ as by the assumption in the statement of Lemma 10 the distance $d_{G-N_i(v)}(p, q) = \infty$, thus $|P \cap N_i(v)| = 1$. □

Lemma 11. Given a graph $G = (V, E)$ with a longest geodesic path $P = \{v_1, v_2, \ldots, v_k\}$, the following holds: $\forall i \forall j \Big(i > 0 \wedge i \leq k \wedge j > 0 \wedge j \leq k \wedge i \neq j\Big) \implies |v_i - v_j| = d(v_i, v_j)$.

Proof. Let Q be a subpath of P with endpoints $a \in P$ and $b \in P$. Assume for contradiction $|Q| > d(a, b)$, we can create a shorter path between the endpoints of P by replacing the subpath Q with the shorter geodesic path $a - b$. Thus P was not a geodesic path, leading to a contradiction. □

Proposition 1 (b-dimensional largest mixed sum). Given h b-dimensional vectors $p_i = (x_1, x_2, \ldots, x_b)_i$ and h' b-dimensional vectors $p'_j = (x'_1, x'_2, \ldots, x'_b)_j$ ($h \geq h'$). We want to maximize: $\min\{p_i + p'_j\}$. b-dimensional largest mixed sum

can be solved in $O(hh'b)$ time.

Proof. We have h b-dimensional vectors $p_i := (x_1, x_2, x_3 \dots x_b)_i$ and h' b-dimensional vectors $p'_j := (x'_1, x'_2, x'_3 \dots x'_b)_j$. Computing the min $(p_i + p'_j)$ can be done in $O(b)$ time. We compute the min for every pair of h and h' vectors. We keep track of the maximum value (and the corresponding vertices) after every min computation. Computing all these values can be done in $O(hh'b)$. □

Theorem 1. Given a graph $G = (V, E)$ with $\delta > 0$, we can compute the diameter in $O(n^2)$ time with high probability.

Proof. Choose a vertex v at random. Perform a BFS from v to compute all the layers $N_i(v)$. Remove the layers with size strictly greater than $2/\delta$. From the remaining layers, choose one layer $L = \{w_1, w_2, \dots, w_{|L|}\}$ at random. All vertices in L are on a distance l from v. Mark every vertex in G red which has a distance less than l to v. Mark every vertex blue which has a distance greater than l to v. Precisely: $\forall w \in V\Big(d(v,w) > l \implies \text{blue(w)}\Big) \wedge \Big(d(v,w) < l \implies \text{red(w)}\Big)$. Define B as the set of blue vertices, and R as the set of red vertices. Perform a BFS from every $w_i \in L$, where $1 \leq i \leq |L|$. With all of these distances, we set up an instance of $|L|$-dimensional largest mixed sum in the following way: create the two sets, H and H'. H contains an $|L|$-dimensional vector for every red vertex, and is defined as $H = \{(d(r, w_1), d(r, w_2), \dots, d(r, w_{|L|})) \mid r \in R\}$. Similarly, H' is defined as $H' = \{(d(c, w_1), d(c, w_2), \dots, d(c, w_{|L|})) \mid c \in B\}$. Largest mixed sum will return two vertices $o \in \{w \mid d(v, w) < l\}$ and $s \in \{w \mid (d(v, w) > l\}$. We reconstruct the $o - s$ path by performing a BFS from o, and output the length of the $o - s$ path.

Analysis. With constant probability, v will be on a longest geodesic $p - q$ path P with a distance $k\delta n$ (where $k < \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n}$) from either p or q. The layer L will with constant probability separate p and q (consequence of Lemma 9).

The algorithm always outputs the result of largest mixed sum since it always returns a geodesic path. The path returned is always geodesic as a consequence of Lemma 11. It does not matter if a path from a red vertex r to a blue vertex b intersects L multiple times, largest mixed sum still returns a geodesic path. If L separated p and q, we are guaranteed to return a longest geodesic path since $p \in R$ and $q \in B$ (or vice versa). By Proposition 1, $|L|$-dimensional largest mixed sum is solvable in $O(|H||H'||L|)$ time. Since $|H| \leq n$, $|H'| \leq n$ and $|L| \leq 2/\delta$ (δ is a constant), we can solve L-dimensional largest mixed sum in $O(n^2)$ time. Thus the total complexity of the algorithm is $O(n^2)$ since the rest of the computations consist of a constant number of BFS computations. □

Figure 3.4 illustrates a situation in Theorem 1. The vertices w_1 and w_2 are on the randomly chosen layer (depicted as a circle around the vertex v) which separates the two endpoints p and q of a longest geodesic path. The vertices a and b are arbitrary red and blue vertices.

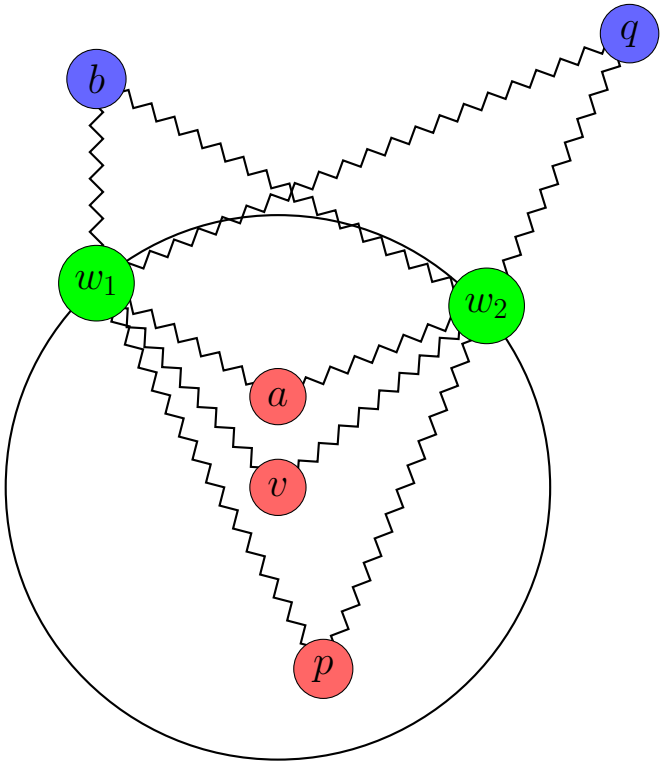

Figure 3.4: Depicts a situation in Theorem 1, with the geodesic paths to w_1 and w_2 which will be used when computing the largest mixed sum.

There are different parts of the algorithm in Theorem 1 which can be improved. The algorithm has two stages of randomization, the first is to choose a random vertex v. With constant probability, the vertex v is on a longest geodesic path with a distance of $k\delta n$ from one endpoint. The second stage is to randomly choose a layer ($N_i(v)$) of constant size. When both these stages are successfully randomized, the algorithm is guaranteed to output the diameter. However, to guarantee that the algorithm will output the diameter, it is sufficient for a randomly chosen vertex to be in the $k\delta n$ neighbourhood of some endpoint of a longest geodesic path (consequence of Lemma 11). Consider the following Lemma:

Lemma 12. Given a graph $G = (V, E)$ with a longest geodesic $p - q$ path P. If a vertex $v \in V$ has the property $d(v, p) = k\delta n \vee d(v, q) = k\delta n$, then $\sum_{b=1}^{n} \sum_{i=1}^{\infty} \psi(i, v, b) \geq (1 - k)\delta n$ holds.

Proof. Without loss of generality, we assume $d(v, p) = k\delta n$. Since v is not necessarily on P, we know $d(v, q) \geq (1 - k)\delta n$ is true. $d(v, q)$ can not be less than $(1 - k)\delta n$, since it would imply that $p - q$ was not a geodesic path, which is a contradiction. Thus $\sum_{b=1}^{n} \sum_{i=1}^{\infty} \psi(i, v, b) \geq (1 - k)\delta n$ holds. □

A consequence of Lemma 12 is that any vertex v in a $k\delta n$-neighbourhood of an endpoint of a longest geodesic path has at least as many layers as a vertex on a longest geodesic path with distance $k\delta n$ to an endpoint. The logic in Lemmas 6-9 will still apply for v. Thus we can choose a layer from v, which separates the two

endpoints of a longest geodesic path, with constant probability. A consequence of this result is that the probability of returning the diameter in Theorem 1 is greater.

If one can deterministically find a vertex within a $k\delta n$-neighbourhood of some endpoint of a geodesic path, then one can also construct a (1-k)-approximation algorithm of the diameter, consider the following proposition:

Proposition 2. Given a graph $G = (V, E)$ with a longest geodesic $p - q$ path P. Finding a vertex v deterministically in $O(n^2)$ time, such that $d(v, p) = k\delta n \vee d(v, q) = k\delta n$, where $0 < k < 1/2$, implies that an (1-k)-approximation algorithm of the diameter in $O(n^2)$ time exists.

Proof. Perform a BFS from the vertex v and return the distance to a vertex furthest away from v. This distance will at least be $(1-k)\delta n$, as v is in a $k\delta n$-neighbourhood of some endpoint of a longest geodesic path. □

Lemma 13. Given a graph $G = (V, E)$, a vertex $v \in V$, a $k > 0$ and a $a \geq 0$, then $\exists x \ |N_x(v)| \leq \frac{1-(\frac{\delta}{2}-k\delta)}{k\delta}$, where $ak\delta n < x \leq (a+1)k\delta n$.

Proof. The least amount of layers for a vertex v ($N_i(v)$) (shown in Lemma 1) is shown in the following expression:

$$\sum_{b=1}^{n}\sum_{i=1}^{\infty}\psi(i, v, b) \geq \frac{\delta n}{2}. \tag{3.11}$$

The layers $N_x(v)$ ($ak\delta n < x < (a+1)k\delta n$) have the greatest size when the remaining layers are of size one. The remaining layers are the layers $N_i(v)$ with the following constraint on i $\Big(1 \leq i \leq ak\delta n\Big) \vee \Big((a+1)k\delta n < i\Big)$. The total amount of vertices in these remaining layers is shown in the following expression:

$$\frac{\delta n}{2} - ((a+1)k\delta n - ak\delta n)) = \frac{\delta n}{2} - k\delta n. \tag{3.12}$$

We calculate the amount of vertices a layer $N_x(v)$ ($ak\delta n \leq x < (a+1)k\delta n$) has, if all the remaining vertices are distributed evenly among these $N_x(v)$ layers:

$$\frac{n - (\frac{\delta n}{2} - k\delta n)}{k\delta n} = \frac{1 - (\frac{\delta}{2} - k\delta)}{k\delta}. \tag{3.13}$$

□

Lemma 14. Given a graph $G = (V, E)$ with a $\delta > 0$ and a longest geodesic $p - q$ path P, we can deterministically find a set of vertices $S \subset V \wedge |S| = c$ (where $c > 0$ is a constant), such that $\exists i \ v_i \in S \wedge d(v_i, p) \leq k\delta n \vee d(v_i, q) \leq k\delta n$, in $O(n+m)$ time.

Proof. Take an arbitrary vertex v and count the number of vertices in the graph by performing a BFS. Compute the greatest k which satisfy $k < \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n}$. We do not know if v has a distance $k\delta n$ to p or q, but it could be the case so we add v to the set S. Choose the layer $|N_x(v)| \leq \lceil\frac{1-(\frac{\delta}{2}-k\delta)}{k\delta}\rceil$, for the largest x in the interval:

$1 \leq x \leq k\delta n$ (the existence of such layer is shown in Lemma 13). Add all the vertices in $N_x(v)$ to the set S. Continue to take another layer $|N_{x'}(v)| \leq \lceil \frac{1-(\frac{\delta}{2}-k\delta)}{k\delta} \rceil$ for the largest x' in the interval: $x+1 \leq x' \leq x+k\delta n$. Add all the vertices in $N_{x'}(v)$ the set S. Continue this approach until there are no interval of size $k\delta n$ left.

Analysis. The number of intervals is at most $\lceil \frac{\delta n}{k\delta n} \rceil = \lceil \frac{1}{k} \rceil$, which is a constant. Furthermore, every layer is of a constant size ($|N_x(v)| \leq \lceil \frac{1-(\frac{\delta}{2}-k\delta)}{k\delta} \rceil$). The intervals together cover the whole graph such that any vertex $v \in V$ has a distance of at most $k\delta n$ to one of the vertices in the set S. □

Theorem 2. Given a graph $G = (V, E)$ with $\delta > 0$, fixed $k > 0$ and a longest geodesic $p - q$ path P. We can compute an (1-k)-approximation in $O(n + m)$ time.

Proof. Compute a set of vertices S as described in Lemma 14. S contains at least one vertex v which has the property $\Big(d(v,p) \leq (1-k)\delta n\Big) \vee \Big(d(v,q) \leq (1-k)\delta n\Big)$. Computing S can be done in $O(n+m)$ time (shown in Lemma 14). For every vertex $v_i \in S$, execute the procedure described in Proposition 2, and return the largest distance. S is of constant size, which implies that only a constant number of BFSs are computed, and therefore the (1-k)-approximation can be computed in $O(n+m)$ time. □

Lemma 15. Given a graph $G = (V, E)$ with a $\delta > 0$, a vertex v and a longest geodesic $p-q$ path P such that $d(v,p) = k\delta n \vee d(v,q) = k\delta n$, where $k < \frac{\delta+1}{3\delta+6} - \frac{2}{3\delta n}$. We can deterministically find in $O(n+m)$ time, a layer $|N_x(v)| \leq \lceil \frac{1-(\frac{\delta}{2}-k\delta)}{k\delta} \rceil$ such that $d_{G-N_x(v)}(p,q) = \infty$.

Proof. Define x such that $k\delta n < x \leq 2k\delta n$, the following is true $d_{G-N_x(v)}(p,q) = \infty$. By Lemma 13, the following holds $\exists x\ k\delta n < x \leq 2k\delta n \implies |N_x(v)| \leq \frac{1-(\frac{\delta}{2}-k\delta)}{k\delta}$. Choose the first layer $N_a(v)$ such that $a > k\delta n$ and $|N_a(v)| \leq \lceil \frac{1-(\frac{\delta}{2}-k\delta)}{k\delta} \rceil$. One can compute k and all the layers from v with one BFS, thus the total complexity is $O(n+m)$. □

Theorem 3. Given a graph $G = (V, E)$ with a $\delta > 0$. We can deterministically compute the diameter in $O(n^2)$ time.

Proof. Execute the procedure in Lemma 14 which creates a set S where at least one vertex $v \in S$ satisfies $\Big(d(v,p) \leq k\delta n\Big) \vee \Big(d(v,q) \leq k\delta n\Big)$, where p and q are endpoints of a longest geodesic path. For every $w \in S$, execute the procedure in Lemma 15 which finds a layer of constant size. We have now arrived at the same setup as Theorem 1, where we colour vertices red and blue depending on their distance to the layer returned by executing the procedure described in Lemma 15. Similarly to Theorem 1, set up an instance of largest mixed sum from which a geodesic path is constructed. Since these procedures are ran for every $w \in S$, and $v \in S \wedge |S| = c$ (where c is a constant), we only perform the procedure in Theorem 1 a constant

number of times. Thus the total time complexity is $O(n^2)$. When the procedure is ran for vertex v, the layer returned by the procedure in Lemma 15 will separate p and q, thus we will in this case reconstruct a longest geodesic path from largest mixed sum. □

4
Discussion

In this chapter we discuss the results, primarily focusing on various bounds, probability and constant factors in the time complexity of the algorithms. We also propose different directions for future work.

4.1 Bounds

In this subsection we will cover various bounds used in the results. We will argue why some of the bounds might be too generous, leading to longer execution times in the algorithms (hidden constants in the time complexity). Moreover, we will also argue why some bounds are tight.

Consider eq. (3.6) where we compute how many layers from a reference vertex v that do not not separate two endpoints of a longest geodesic path:

$$\delta n - \Big[(1-2k)\delta n - 2\Big].$$

Any vertex has at most δn layers (amount of layers can never be larger than the diameter), thus we assume the worst case, i.e. a randomly chosen vertex has precisely δn layers. This bound is trivially tight because one might randomly choose one of the endpoints of a longest geodesic path. Even if one does not choose an endpoint, any vertex on a longest geodesic path can still have δn layers. Consider graphs where all vertices are on the same cycle, in such graphs every vertex is an endpoint of a longest geodesic path, thus every vertex has δn layers.

The amount of layers which separate p and q ($(1-2k)\delta n - 2$) is exact given that the reference vertex is on a longest geodesic $p-q$ path with distance $k\delta n$ from one endpoint. However, if the reference vertex is in the $k\delta n$-neighbourhood of either p or q (the setup in Lemma 12), the amount of layers (from the reference vertex) which separate p or q ($(1-2k)\delta n - 2$) is a lower bound. Assume a vertex v is in a $k\delta n$-neighbourhood of p, then the distance to q is at least $(1-k)\delta n$, as we do not know if $v \in p-q$ or $v \notin p-q$.

The size of the separating layer in Lemma 15 ($\lceil \frac{1-(\frac{\delta}{2}-k\delta)}{k\delta} \rceil$) is not tight since we only consider the existence of a constant sized layer in an interval of size $k\delta n$. One can make a tighter bound by finding the largest b such that $(1-2k)\delta n > bk\delta n$, and use this b to set up an equation similar to Lemma 13.

The algorithms have a bound on the number of vertices $n \geq \frac{2(\delta+2)}{\delta(\delta+1)} + 1$. A bound on n is necessary as the graph has at least $1/\delta$ vertices (a graph with diameter 1). A graph with diameter one is a complete graph, and complete graphs do not have any separators. The Monte Carlo algorithm and the deterministic algorithm rely on separators to compute the diameter.

Lemma 2 proves the existence of a layer, for an arbitrary vertex v ($N_i(v)$), of size at most $2/\delta$ for any graph with a fixed $\delta > 0$. Since the randomly chosen layer in the Monte Carlo algorithm (presented in Theorem 1) is only guaranteed to separate the endpoints of a longest geodesic path with constant probability, if the random vertex v has a distance $k\delta n$ from one of the endpoints of a longest geodesic path, v must have at least $(1-k)\delta n$ layers. Consequently, one can calculate a tighter bound than $2/\delta$, as the assumption in Lemma 2 was that the reference vertex only has $\delta n/2$ layers. Similarly as in Lemma 2, one can calculate the constant sized layer by using the pigeonhole principle:

$$\frac{n}{(1-k)\delta n} = \frac{1}{(1-k)\delta}. \tag{4.1}$$

We know therefore that any vertex v with $(1-k)\delta n$ layers has at least one layer with size at most $1/(1-k)\delta$. We also know (from Lemma 9) that $k < (\delta+1)/(3\delta+6) - 2/(3\delta n)$. The parameter k will always be less than $(\delta+1)/(3\delta+6)$ for any n. We substitute this bound on k in the equation eq. (4.1) and get the following expression for the size at least one of the layers has to have:

$$\begin{aligned}
&\frac{1}{(1-\frac{\delta+1}{3\delta+6})\delta} = \\
&\frac{1}{\delta - \frac{\delta(\delta+1)}{3\delta+6}} = \\
&\frac{1}{\delta - \frac{\delta^2+\delta}{3\delta+6}} = \\
&\frac{1}{\delta\frac{(3\delta+6)}{3\delta+6} - \frac{\delta^2+\delta}{3\delta+6}} = \\
&\frac{1}{\frac{3\delta^2+6\delta}{3\delta+6} - \frac{\delta^2+\delta}{3\delta+6}} = \\
&\frac{1}{\frac{2\delta^2+5\delta}{3\delta+6}} = \\
&\frac{3\delta+6}{2\delta^2+5\delta}.
\end{aligned}$$

Lemma 3 shows that the maximum size of a layer could be $n - n\delta/2$, thus a longest geodesic path could be entirely in one layer or in arbitrary many layers. Since a longest geodesic path can intersect arbitrary many layers we can not easily reject layers which a longest geodesic path will not traverse or localize the layers a longest

geodesic path will traverse.

4.2 Comparing the Monte Carlo- with the deterministic algorithm

The Monte Carlo algorithm and the deterministic algorithm have the same time complexity $O(n^2)$, but the deterministic algorithm has a large hidden constant. It would be of interest to compare the constant in the deterministic algorithm's time complexity with the constant in the Monte Carlo algorithm's time complexity. The Monte Carlo algorithm's constant mainly depends on repeating the procedure a constant number of times to get the probability of returning the diameter arbitrary close to 1. Assume that the constants (of the deterministic and Monte Carlo algorithm) are equal after x repetitions of the Monte Carlo procedure. One could calculate the probability of returning the diameter in the Monte Carlo algorithm after x repetitions of the procedure. If one finds the probability of returning the diameter inadequate, then one never has to consider performing the Monte Carlo algorithm and instead choose the deterministic one since it will perform better. It is therefore of interest to calculate and compare the constant probability in the Monte Carlo algorithm with the deterministic algorithm. Perhaps the Monte Carlo algorithm is more practical if the probability is high even after a relatively small number of repetitions of the procedure.

4.3 Further work

Even if the probability of finding a layer (in the Monte Carlo algorithm) from a randomly chosen vertex v which separates two endpoint of a longest geodesic path does not seem high, one must remember that all the calculations are based on a worst case scenario. In our equations we consider that a vertex v has as many layers as possible (δn) and that all of the layers which do not separate the endpoints are of constant size (at most $2/\delta$). Thus we believe that the worst case is pessimistic. It would be very interesting if one could compute the probability of the average case or at least benchmark the Monte Carlo algorithm on a sample of representative realistic graphs.

We have solved the b-dimensional largest mixed sum in $O(n^2)$ time. This result is worse compared to the 2-dimensional case where Damaschke [2] presents a solution in $O(n \log n)$ time. Our solution could be considered quite naive as we approach the problem in a brute-force way. The 2-dimensional case is solved in a more sophisticated way, which suggests that the b-dimensional case can be improved. If one solves the b-dimensional largest mixed sum in better than $O(n^2)$ time, the complexity of both the Monte Carlo and deterministic algorithms become better as the bottle neck is the largest mixed sum computation. It is also of interest if one can prove that the worst case is $O(n^2)$.

The deterministic algorithm has a large constant factor. One possible approach for making the deterministic algorithm more practical is to reduce the size of the set S in Lemma 14. In graphs which are not huge, the constant factor will bottleneck the algorithm since the set of vertices can get big, and b-dimensional largest mixed sum is computed for every vertex in S. The reduction of vertices in S must be done in $O(n^2)$ time to reduce the constant factor and preserve the time complexity of the algorithm.

We think that it could be of interest to introduce a new ratio or quantifier which further limits the structure of slim graphs. Such a new ratio or identifier could supply even more information about the structure and perhaps allow for even faster algorithms to be developed.

5
Conclusion

We have exploited structural features of slim graphs, mainly involving separators, to create three new algorithms: a Monte Carlo, an approximation and a deterministic algorithm. The approximation algorithm yields a better approximation than the general graph diameter approximation, in $O(n + m)$ time. The deterministic algorithm computes the diameter in $O(n^2)$ time, which makes it asymptotically superior to APSP. We have not calculated the constant factors in the time complexity of these algorithms, and we have neither benchmarked any of the algorithms. Thus it makes reasoning about the practicality difficult, but we have proposed several of natural directions for further work. Even if the algorithms are not practical yet, we have presented multiple results which supply more knowledge about how slim graphs behave, in particular when δ gets smaller than what was considered by Damaschke [2]. We will conclude this thesis by referencing the research questions and describing their respective outcomes.

Is it possible to extend the properties and techniques used by Damaschke [2] for a broader range of slim graphs?
The answer is yes. We have generalized parts of the results presented by Damaschke in various Lemmas, most notably Lemma 2 which generalizes the existence of a constant size separator for smaller δ. We have also presented a solution for largest mixed sum for higher dimensions.

Is it possible to make a Monte Carlo algorithm for a broader range of slim graphs?
Yes, Theorem 1 describes a Monte Carlo algorithm for $\delta > 0$.

Is it possible to create a Las Vegas algorithm for a broader range of slim graphs faster than APSP? This question becomes irrelevant as we present a deterministic algorithm in Theorem 3, although one might develop a Las Vegas algorithm with lower complexity than the algorithm in Theorem 3.

For a fixed $0 < k < 1/2$, is it possible to create an (1-k)-approximation algorithm faster than APSP? Yes, Theorem 2 presents an algorithm for an (1-k)-approximation in $O(n + m)$ time, for a fixed $0 < k < 1/2$.

Is it possible to create a deterministic algorithm faster than APSP? Yes, Theorem 3 presents a deterministic algorithm for a fixed $\delta > 0$, which computes the diameter in $O(n^2)$ time.

Does a deterministic algorithm, for graphs with diameter greater than $n/3$, in $O(m + n \log n)$ time exist? We can not answer this question, although if someone solves b-dimensional largest mixed sum in $O(m + n \log n)$ time, then it is possible to not only solve for $\delta > 1/3$, but also for $\delta > 0$ in $O(m + n \log n)$ time.

Bibliography

[1] Corey Pennycuff and Tim Weninger. Fast, exact graph diameter computation with vertex programming. In *1st High Performance Graph Mining workshop, Sydney, 10 August 2015*. Barcelona Supercomputing Center, 2015.

[2] Peter Damaschke. Computing giant graph diameters. In *27th International Workshop on Combinatorial Algorithms IWOCA, Helsinki, Lecture Notes in Computer Science, vol. 9843*, pages 373–384. Springer, 2016.

[3] Giso H Dal, Walter A Kosters, and Frank W Takes. Fast diameter computation of large sparse graphs using gpus. In *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on*, pages 632–639. IEEE, 2014.

[4] Derek G. Corneil, Feodor F. Dragan, Michel Habib, and Christophe Paul. Diameter determination on restricted graph families. *Discrete Applied Mathematics*, 113(2):143 – 166, 2001.

[5] Jens M. Schmidt. A simple test on 2-vertex- and 2-edge-connectivity. *Information Processing Letters*, 113(7):241 – 244, 2013.

[6] John Hopcroft and Robert Tarjan. Algorithm 447: Efficient algorithms for graph manipulation. *Commun. ACM*, 16(6):372–378, June 1973.

[7] Derek G. Corneil, Stephan Olariu, and Lorna Stewart. Asteroidal triple-free graphs. *SIAM Journal on Discrete Mathematics*, 10(3):399–430, 1997.

[8] Boaz Ben-Moshe, Binay Bhattacharya, Qiaosheng Shi, and Arie Tamir. Efficient algorithms for center problems in cactus networks. *Theoretical Computer Science*, 378(3):237 – 252, 2007. Algorithms and Computation.

[9] Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 515–524, New York, NY, USA, 2013. ACM.

[10] Derek G Corneil, Feodor F Dragan, and Ekkehard Köhler. On the power of bfs to determine a graph's diameter. *Networks*, 42(4):209–222, 2003.