The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Sublinear Search Spaces for Shortest Path Planning in Grid and Road Networks

Johannes Blum
Julius-Maximilians-Universität Würzburg
97072 Würzburg, Germany
blum@informatik.uni-wuerzburg.de

Stefan Funke
Universität Stuttgart
70569 Stuttgart, Germany
funke@fmi.uni-stuttgart.de

Sabine Storandt
Julius-Maximilians-Universität Würzburg
97072 Würzburg, Germany
storandt@informatik.uni-wuerzburg.de

Abstract

Shortest path planning is a fundamental building block in many applications. Hence developing efficient methods for computing shortest paths in e.g. road or grid networks is an important challenge. The most successful techniques for fast query answering rely on preprocessing. But for many of these techniques it is not fully understood why they perform so remarkably well and theoretical justification for the empirical results is missing. An attempt to explain the excellent practical performance of preprocessing based techniques on road networks (as transit nodes, hub labels, or contraction hierarchies) in a sound theoretical way are parametrized analyses, e.g., considering the highway dimension or skeleton dimension of a graph. But these parameters tend to be large (order of $\Theta(\sqrt{n})$) when the network contains grid-like substructures – which inarguably is the case for real-world road networks around the globe. In this paper, we use the very intuitive notion of bounded growth graphs to describe road networks and also grid graphs. We show that this model suffices to prove sublinear search spaces for the three above mentioned state-of-the-art shortest path planning techniques. For graphs with a large highway or skeleton dimension, our results turn out to be superior. Furthermore, our preprocessing methods are close to the ones used in practice and only require randomized polynomial time.

Introduction

Shortest paths in general graphs can be computed by Dijkstra's algorithm in near-linear time. Nevertheless, this is still too slow for many practical purposes, as e.g. providing driving directions in real-time for road networks of continental size, or finding shortest paths in large grid domains as game maps. This spurred the development of preprocessing based shortest path speed-up techniques. Here, auxiliary data is created in a preprocessing phase which can then be used to prune the search space for subsequent queries yet preserving optimality of the result. Incarnations of this scheme, as contraction hierarchies [CH] (Geisberger et al. 2012), transit nodes [TN] (Bast et al. 2007), and hub labels [HL] (Abraham et al. 2011b), allow the answering of shortest path queries on large road networks and grids in milliseconds or less, while exhibiting small preprocessing times and space consumption. But there is still a lack of theoretical explanation why these approaches perform so remarkably well in practice. The main question is which characteristics of a network are necessary or sufficient to guarantee efficiency.

The notion of highway dimension (Abraham et al. 2010) was introduced for this purpose. Intuitively, the highway dimension h of a graph is small if there exist sparse local hitting sets for shortest paths of a certain length. For CH and HL, query times in $\mathcal{O}(h \log D)$ were proven (with D denoting the network diameter), and $\mathcal{O}(h^2)$ for TN. The space consumption was shown to be in $\mathcal{O}(nh \log D)$ or $\mathcal{O}(hn)$, respectively. The hope would be that real-world networks exhibit a highway dimension which is logarithmic in the size n of the network. But for grid graphs with uniform costs, $h \in \Theta(\sqrt{n})$ was proven. Grids comply with all standard characterizations of road networks, as constant maximum degree Δ, a linear number of edges $m = |E|$, and (near) planarity. And all three – CH, TN and HL – have been successfully applied to (pure) grid graphs (Storandt 2013; Antsfeld et al. 2012; Delling et al. 2014). The notion of highway dimension is not sufficient to explain these results as, e.g., the query times of TN are superlinear for $h \in \Theta(\sqrt{n})$. And as many real-world road networks contain large grid-like substructures, a highway dimension in the same order is to be expected. For the U.S. road network and for the German road network with about 20 million nodes each, a highway dimension of more than 1,000 is known, which renders a logarithmic highway dimension unlikely.

We will use a different model, assuming that the underlying graph metric has bounded growth. This model was empirically proven to reflect real-world road networks well (Funke and Storandt 2015), and also subsumes grid graphs. We show that the bounded growth assumption suffices to prove sublinear query times, a clearly subquadratic space consumption and randomized polynomial time preprocessing for CH, TN and HL in unweighted graphs.

Related Work

There have been some attempts besides the highway dimension trying to explain the good performance of shortest path planning techniques. We will now briefly review these methods and ideas. For a concise overview, we refer to Table 1.

For HL, the recently described skeleton dimension k (Kosowski and Viennot 2017) is also suitable to prove a the-

Copyright © 2018, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

	original result	$\mathbf{h}, \mathbf{k}, \mathbf{t} \in \mathcal{O}(\sqrt{\mathbf{n}})$
CH: graphs with highway dimension h		
query	$\mathcal{O}(h \log D)$	$\mathcal{O}(\sqrt{n} \log D)$
space	$\mathcal{O}(nh \log D)$	$\mathcal{O}(n\sqrt{n} \log D)$
CH: minor-closed graphs with balanced separators		
query	$\mathcal{O}(\sqrt{n})$	
space	$\mathcal{O}(n \log n)$	
CH: graphs with treewidth t		
query	$\mathcal{O}(t \log n)$	$\mathcal{O}(\sqrt{n} \log n)$
space	$\mathcal{O}(nt \log n)$	$\mathcal{O}(n\sqrt{n} \log n)$
CH: bounded growth model, correct queries w.h.p.*		
query	$\mathcal{O}(\sqrt{n} \log n)$	
space	$\mathcal{O}(n \log^2 n)$	
CH: bounded growth model (NEW!)*		
query	$\mathcal{O}(\sqrt{n} \log n)$	
space	$\mathcal{O}(n \log D)$	
special TN variant: graphs with highway dimension h		
query	$(\Delta + h \log D)$	$\mathcal{O}(\Delta + \sqrt{n} \log D)$
space	$\mathcal{O}(nh \log D)$	$\mathcal{O}(n\sqrt{n} \log D)$
TN: graphs with highway dimension h		
query	$\mathcal{O}(h^2)$	$\mathcal{O}(n)$
space	$\mathcal{O}(hn + m)$	$\mathcal{O}(n\sqrt{n} + m)$
TN: bounded growth model (NEW!)*		
query	$\mathcal{O}(n^{2/3} \log^{8/3} n)$	
space	$\mathcal{O}(n^{4/3} \log^{4/3} n)$	
HL: graphs with highway dimension h		
query	$\mathcal{O}(h \log D)$	$\mathcal{O}(\sqrt{n} \log D)$
space	$\mathcal{O}(nh \log D)$	$\mathcal{O}(n\sqrt{n} \log D)$
HL: graphs with skeleton dimension k*		
query	$\mathcal{O}(k \log D)$	$\mathcal{O}(\sqrt{n} \log D)$
space	$\mathcal{O}(nk \log D)$	$\mathcal{O}(n\sqrt{n} \log D)$
HL: bounded growth model (NEW!)*		
query	$\mathcal{O}(\sqrt{n})$	
space	$\mathcal{O}(n\sqrt{n})$	

Table 1: Theoretical results for shortest path speed-up techniques in dependency of n (number of nodes), m (number of edges), Δ (maximum degree), D (diameter), h (highway dimension), k (skeleton dimension) and t (tree width). Space consumption is measured in machine words. Results marked with a * exhibit (randomized) polynomial preprocessing times. For results in dependency of h there exist also polytime preprocessing variants with slightly increased query/space bounds.

oretical query time of $\mathcal{O}(k \log D)$ and a space consumption of $\mathcal{O}(nk \log D)$. For graphs with bounded maximum degree, $k \in \mathcal{O}(h)$ is known. For some graphs there is an exponential gap between highway and skeleton dimension, as shown in (Kosowski and Viennot 2017) via a carefully weighted square grid where $h \in \Omega(\sqrt{n})$ and $k \in \mathcal{O}(\log n)$. However, for other grids one obtains again a skeleton dimension of $k \in \Theta(\sqrt{n})$, leading to the same time and space bounds as in the highway dimension dependent analysis.

In (Bauer et al. 2013), CH were analyzed by considering only the topology of the network. For graphs with a tree width t, the query time was shown to be in $\mathcal{O}(t \log n)$, the space consumption in $\mathcal{O}(nt \log n)$. But again, for grids, we have $t \in \Theta(\sqrt{n})$.

In (Funke and Storandt 2015), the bounded growth model was suggested to capture the characteristics of road networks. Then CH was analyzed by drawing a connection to randomized skip lists. Queries were proven to be answered correctly with high probability (w.h.p). We will also use the bounded growth model in our analysis, improving the previous result for CH, and showing that this model is also suitable to explain the good performance of TN and HL.

Contribution

We show that the bounded growth model in combination with randomized preprocessing is suitable to prove sublinear search space sizes for the three state-of-the-art shortest path planning techniques, contraction hierarchies (CH), transit nodes (TN) and hub labels (HL) in unweighted networks, while using subquadratic space. More precisely, we provide the following results:

- We analyze the relationship between bounded growth and the skeleton dimension of the graph. In particular, we show that the integrated skeleton dimension of a bounded growth graph is upper bounded by $\mathcal{O}(\sqrt{n})$. This allows us to achieve new improved query times and space bounds for HL on graphs with $h, k \in \Theta(\sqrt{n})$.
- For randomized CH, we improve the space consumption reported in (Funke and Storandt 2015) from $\mathcal{O}(n \log^2 n)$ to $\mathcal{O}(n \log D)$ by using a new random contraction order. Furthermore, we show that randomized CH can be constructed in polynomial time such that all queries are answered correctly (and not only w.h.p. as in (Funke and Storandt 2015)).
- We show how to instrument ϵ-net theory and random sampling for TN preprocessing. We provide a parametrized analysis which allows to trade space consumption against query time. On that basis, we achieve the smallest known space bound for TN and sublinear query times on graphs with $h \in \Theta(\sqrt{n})$.

The new results are summarized in Table 1 as well.

Bounded Growth and Skeleton Dimension

We first introduce the basic notation and formally define the bounded growth model. For the remainder of this paper we consider a directed graph $G(V, E)$ with $n = |V|$ nodes, $m = |E|$ edges and uniform edge costs. In real-world road networks the ratio between the longest and the shortest edge is typically bounded by a small constant, so subsampling long edges to achieve uniform edge lengths does not increase the graph size considerably (similar arguments were used in (Kosowski and Viennot 2017)). We assume all shortest paths to be unique in G, which is a standard assumption but can also be enforced, e.g., by lexicographic sorting of edges. We denote with $d_v(w)$ the shortest path distance from v to w. All nodes within a distance r of v are said to be contained in the ball $B_r(v)$.

Bounded Growth Model

Throughout our analysis, we assume the graph metric to have *bounded growth* (not to be confused with the notion of growth bounded graphs as used in (Kuhn et al. 2005)). Formally, we demand that for some constant $c \in \mathbb{R}^+$ (w.l.o.g. $c \geq 1$), the number of nodes at distance r from a node v is bounded by cr, implying $|B_r(v)| \leq cr(r+1)/2 \in \mathcal{O}(r^2)$. Intuitively, this reflects the area growth of a disk in the Euclidean plane in dependency of its radius.

In (Funke and Storandt 2015), it was shown empirically that the bounded growth model represents real-world road networks well. In fact, c can be computed for a network in polynomial time (the same is true for k but not for h or t). The results reported in (Funke and Storandt 2015) imply that for Euclidean edge costs, $c = 1$ is valid. We furthermore observe that grids with uniform costs fit the model well as the number of nodes at distance r is bounded by $4r$. Hence our model subsumes grid networks with a highway dimension, a skeleton dimension as well as a tree width of $\Theta(\sqrt{n})$.

Relation to Skeleton Dimension

For a formal definition of the skeleton dimension (Kosowski and Viennot 2017), let $\tilde{T}_u$ be the geometric realization of the shortest path tree T_u of some vertex u. Intuitively, $\tilde{T}_u$ consists of infinitely many infinitely short edges such that for every edge vw of T_u and any $\alpha \in [0,1]$ there is a vertex in $\tilde{T}_u$ at distance α from v and distance $1-\alpha$ from w. The *skeleton* T_u^* of T_u is now defined as the subtree of $\tilde{T}_u$ induced by all vertices v that have a descendant w satisfying $d_v(w) \geq \frac{1}{2} d_u(v)$. The *skeleton dimension* k of a graph G is defined as $k = \max_{u \in V, r>0} x_{u,r}^*$ where $x_{u,r}^*$ denotes the number of vertices in T_u^* that are at distance r from u.

The doubling dimension of a graph with skeleton dimension k was shown to be $2k+1$ (Kosowski and Viennot 2017). We now investigate the relationship between the skeleton dimension and the bounded growth model.

Lemma 1 *The skeleton dimension of a bounded growth graph with uniform edge costs is upper bounded by $\mathcal{O}(\sqrt{n})$.*

Proof. Consider a graph $G(V,E,l)$ with the mentioned properties. Then there is a vertex $u \in V$ such that the geometric realization $\tilde{T}_u$ of the shortest path tree T_u contains a set $\tilde{S}$ of k vertices at some depth r that have a descendant at distance at least $r/2$. Every vertex from $\tilde{S}$ has $\lfloor r/2 \rfloor > r/4$ descendants contained in G. For every $v \in \tilde{S}$ let w_v be the first descendant of v satisfying $w_v \in V$ if $v \notin V$, otherwise let $w_v = v$. Then for $S = \{w_v : v \in \tilde{S}\}$ we have $|S| = |\tilde{S}| = k$ and as G has bounded growth we have $k \leq c \cdot \lceil r \rceil < c \cdot (r+1)$. This means that $n \geq kr/4 > k(k/c - 1)/4$, so $k \in \mathcal{O}(\sqrt{n})$. ∎

The authors of (Kosowski and Viennot 2017) also introduced the *integrated skeleton dimension* which weights the vertices in a shortest path tree according to their distance. For a vertex $u \in V$ the integrated skeleton dimension is $\hat{k}(u) = \sum_{r \in \mathbb{N}} x_{u,r}^*/r$ with $x_{u,r}^*$ being defined as above. On general graphs $\hat{k}(u)$ is bounded by $\mathcal{O}(k \log D)$. If the graph has bounded growth, we can however show a bound of $\mathcal{O}(\sqrt{n})$ as a consequence of the following Lemma (observe that $x_{u,r}^* \leq x_{u,r}$).

Lemma 2 *Let $x_{u,r}$ denote the number of nodes at distance r from a node u. Then we have $\sum_{r=1}^{D} x_{u,r}/r \in \mathcal{O}(\sqrt{n})$ in bounded growth graphs.*

Proof. As $\sum_{r=1}^{D} x_{u,r} = n$ and $x_{u,r} \leq cr$ for $r \in \mathbb{N}$, the sum is bounded for $x_{u,r} = cr$ if $r = 1, \ldots, \sqrt{2n}$ and $x_{u,r} = 0$ otherwise by $\sum_{r=1}^{\sqrt{2n}} cr/r = c \cdot \sqrt{2n} \in \mathcal{O}(\sqrt{n})$. ∎

Corollary 3 *The integrated skeleton dimension of any vertex u in a bounded growth graph is at most $\mathcal{O}(\sqrt{n})$.*

While these results might be of independent interest, we will explicitly use them when bounding the query time and the space consumption of HL in bounded growth graphs.

Hub Labels

In the hub labels (HL) approach, every node v gets assigned a set of labels $L(v)$. Here a label is a node w, together with the distance $d_v(w)$. The goal is to find concise label sets which fulfill the so-called cover property, that is, for every $s,t \in V$ the label set intersection $L(s) \cap L(t)$ contains a node w on the shortest path from s to t. If this is the case, queries can be answered by simply summing up $d_s(w) + d_t(w)$ for all $w \in L(s) \cap L(t)$ and keeping track of the minimum. Computing $L(s) \cap L(t)$ can be done by a merging-like step assuming the label sets are presorted by node IDs. Therefore the query time is in $\mathcal{O}(|L(s)| + |L(t)|)$, while the space consumption is in $\mathcal{O}(\sum_{v \in V} |L(v)|)$.

Previous Analyses

In (Abraham et al. 2013; 2011a) hub labels were constructed by computing multiple hitting sets H_r for sets of shortest paths with length $r = 1, 2, 4, \cdots, D$. The label set of a single node v is then determined by $L(v) = \bigcup_r (H_r \cap B_{2r}(v))$. As there are $\log D$ many radii to consider and $H_r \cap B_{2r}(v) \in \mathcal{O}(h)$ according to the definition of the highway dimension, the label size and therefore the query time is in $\mathcal{O}(h \log D)$ for exponential time preprocessing (computing optimal hitting sets), and the space consumption in $\mathcal{O}(nh \log D)$. If hitting sets are constructed via a greedy algorithm in polynomial time, the label size and the space consumption increase by a factor of $\mathcal{O}(\log n)$. In (Kosowski and Viennot 2017), a more practical algorithm for HL was introduced and analyzed. There, a shortest path tree is computed for each node, and then hub labels are selected on certain subpaths via a randomized process. A thorough analysis shows that this leads on average to the selection of $\mathcal{O}(k \log D)$ labels per node, so the total space consumption is in $\mathcal{O}(nk \log D)$.

Analysis in the Bounded Growth Model

We keep our analysis close to the one in dependency of the skeleton dimension(Kosowski and Viennot 2017). Here, random values $\rho(e)$ are assigned to every edge e and for all $u, v \in V$ a hub edge $\eta(u,v)$ is selected as the edge with the minimum value of ρ on the so-called central subpath of u and v. Intuitively, the central subpath of u and v contains all the edges from the shortest u-v-path whose endpoints are not too close to u and v. For more details we refer to

(Kosowski and Viennot 2017). The label set $L(u)$ of a node u is then constructed as the set of the endpoints of all hub edges $\eta(u, v)$. In total, this takes time $\mathcal{O}(n \log n)$ for every label set. The average size of a single label set can be bounded by $\mathcal{O}(\frac{1}{n}\sum_{u\in V}\hat{k}(u))$. From Corollary 3 the following Theorem follows.

Theorem 4 *HL in bounded growth graphs can be computed in randomized polynomial time with expected query times of $\mathcal{O}(\sqrt{n})$ and a space consumption of $\mathcal{O}(n\sqrt{n})$.*

This improves on the respective results reported in (Kosowski and Viennot 2017) assuming $k \in \Theta(\sqrt{n})$ by a factor of $\log n$ and is clearly superior to the highway dimension dependent analysis for $h \in \Theta(\sqrt{n})$.

Contraction Hierarchies

CH construction relies on the so-called node contraction operation. Here, a node v is deleted from the graph, and shortcut edges are inserted between the neighbors of v if they are necessary to preserve the pairwise shortest path distances. The preprocessing phase of CH consists of contracting all nodes one-by-one until the graph is gone. In the end, a new graph $G^+(V, E \cup E^+)$ is constructed, with E^+ being the set of shortcut edges that were inserted during the contraction process. So the space consumption of CH is determined by $\Theta(|E^+|)$. The rank of a node in the order of contraction is called $rank(v)$. In an s-t-query, bidirectional Dijkstra runs are used from s and t in G^+. But edges (v, w) are only relaxed if $rank(v) < rank(w)$, i.e., w was contracted after v. It was proven that both runs will settle the node that was contracted last on the original shortest path from s to t in G. Hence identifying p such that $d_s(p) + d_t(p)$ is minimized leads to correct query answering. The search space of a query is defined by the number of nodes settled in these Dijkstra runs.

Note that any contraction order leads to correct queries, but the space consumption and the search space sizes heavily depend on the contraction order.

Previous Analyses

The topology based analysis of CH (Bauer et al. 2013) inspired a different CH construction scheme based on nested dissections. It was proven that this scheme also leads to excellent performance in practice (Dibbelt, Strasser, and Wagner 2014). Nevertheless, the theoretical results do not directly apply for practical instances as the computation of a balanced separator of minimum size as required in the preprocessing is NP-hard. Hence these computations are replaced by heuristics when applied to real networks.

The NP-hard preprocessing as well as the polynomial time preprocessing when assuming a highway dimension of h both involve the enumeration of all shortest paths in the network and computing (approximate) hitting sets, taking at least superquadratic time and space in n. Therefore, these schemes cannot be applied to large real-world networks either (Abraham et al. 2013).

The skip list inspired randomized CH construction was shown to be implementable (Funke and Storandt 2015). Nevertheless, it also differs from the heuristic construction described in the original CH paper. We will revisit the original node contraction scheme and show that it leads to even better space bounds for randomized CH.

Analysis in the Bounded Growth Model

In contrast to previous provably efficient CH construction schemes (Abraham et al. 2011a; 2013; Bauer et al. 2013; Dibbelt, Strasser, and Wagner 2014), which involve rather heavy algorithmic machinery such as graph separators or hitting set algorithms, our randomized scheme maintains the simplicity of the original CH idea. While in (Geisberger et al. 2012) (and most state-of-the-art implementations of CH) the order is determined on-the-fly using quantities like edge difference, we simply contract the nodes in (uniform) random order, otherwise we employ exactly the same contraction process. Also our query algorithm closely resembles the original query algorithm with a slight modification similar to the stall-on-demand technique in (Geisberger et al. 2012).

Space Consumption. The space consumption of a CH is defined by the number of shortcuts created in the preprocessing. Hence we now aim for an upper bound on this number.

Lemma 5 *The expected number of shortcuts created in a CH with random contraction order is upper bounded by $\mathcal{O}(n \log D)$ in bounded growth graphs.*

Proof. Consider a pair v, w of nodes with shortest path $\pi = vv_1v_2 \ldots v_{r-1}w$ of length r. A shortcut from v to w is created if and only if $rank(w) > rank(v)$ and the ranks of all intermediate nodes on π are smaller than $rank(v)$. For a random permutation, the probability of the event S 'shortcut (v, w) created' to happen is

$$P(S) = \frac{(r-1)!}{(r+1)!} = \frac{1}{r(r+1)}$$

since w must have the largest rank amongst the $r + 1$ nodes, v the second largest, and the remaining ranks can be arbitrarily distributed. Now we compute the expected number of shortcuts created by summing over all pairs of nodes v, w, always considering the probability of a shortcut being created: $\sum_{v,w} P(S) = \sum_{r=1}^{D}\sum_{v,w:d(v,w)=r} P(S)$. Having rearranged the sum according to the distance of the involved nodes, we continue by plugging in our derived probability and making use of the bounded growth property which implies that there are no more than $n \cdot cr$ pairs of nodes with distance r:

$$\sum_{r=1}^{D}\sum_{v,w:d(v,w)=r}\frac{1}{r(r+1)} \leq \sum_{r=1}^{D}(n\cdot cr)\frac{1}{r(r+1)}$$
$$= c\cdot n\sum_{r=1}^{D}\frac{1}{r+1} = \mathcal{O}(n\log D)$$

■

Search Space Size. As in (Funke and Storandt 2015), we define the search space $SS(v)$ for a node $v \in V$ as the number of nodes that are pushed into the priority queue (PQ) during a CH-Dijkstra run from v. We will first analyze the *direct search space (DSS)* of v. A node w is in $DSS(v)$ if on the shortest path from v to w all nodes have rank at most $rank(w)$. Hence, w will be settled with the correct distance $d(v,w)$ in the CH-Dijkstra run. Unfortunately, $SS(v)$ is typically a superset of $DSS(v)$ as also nodes on monotonously increasing (with respect to rank) but non-shortest paths are considered. We will modify the query algorithm to bound the number of such nodes.

Lemma 6 *The probability of a node w at distance r to be contained in $DSS(v)$ is $1/(r+1)$.*

Proof. Consider the nodes $v = v_0 v_1 v_2 \ldots v_r = w$ of the shortest path from v to w. The node w is in $DSS(v)$ iff $rank(v_i) < rank(w)$ for all $i = 0, \ldots r-1$. Clearly each of the v_i, $i = 0, \ldots r$ has the same probability $1/(r+1)$ of having the largest rank amongst $v_0, v_1, \ldots, v_r$, the Lemma follows. ■

Lemma 7 *The expected size of $DSS(v)$ in a bounded growth graph is $\mathcal{O}(\sqrt{n})$.*

Proof. Recall that $x_{v,r}$ denotes the number of nodes at distance r, furthermore let p_r be the probability of a node at distance r to be in $DSS(v)$. Then we can sum up over all nodes at all possible distances as follows:

$$\sum_{r=0}^{D} x_{v,r} p_r = \sum_{r=1}^{D} x_{v,r}/(r+1) < \sum_{r=1}^{D} x_{v,r}/r$$

The claim now follows from Lemma 2. ■

Unfortunately, the actual search space $SS(v)$ during a query execution might be considerably larger than the direct search space $DSS(v)$. As in actual CH implementations (via the technique of stall-on-demand, (Geisberger et al. 2012)) we will modify the search procedure, pruning nodes from $SS(v)$ (by not putting them into the priority queue) which for sure cannot be part of the shortest path we are after.

To that end, we first need to investigate the correlation of a node w being in $DSS(v)$, its distance from v, and its rank. Intuitively, when considering a shortest path π consisting of $r+1$ nodes we would expect the highest rank appearing amongst the nodes of π to be $n - n/(r+2) = n(1 - 1/(r+2))$. The following Lemma formalizes this intuition showing that the probability for deviating greatly from the expected rank is slim.

Lemma 8 *The probability of a node w at distance r to v for being in $DSS(v)$ and having $rank(w) \leq n(1 - c'/r \ln r)$ is less than $r^{-(c'+1)}$.*

Proof. Let $v = v_0 v_1 \ldots v_r = w$ be the shortest path from v to w. The probability p for all nodes v_i, $0 \leq i \leq r$ having rank at most $R := n(1 - c'/r \ln r)$ (a clear prerequisite for w being contained in $DSS(v)$ and having rank at most R) can be upper bounded by counting all permutations where this happens dividing by the total number of permutations

$$p = \frac{\binom{R}{r+1}(r+1)!(n-r-1)!}{n!} = \frac{R!(n-r-1)!}{n!(R-r-1)!}$$
$$< \left(\frac{R}{n}\right)^{r+1} = \left(1 - \frac{c' \ln r}{r}\right)^{r+1} \leq \left(e^{\frac{-c' \ln r}{r}}\right)^{r+1}$$
$$< r^{-c'}$$

Amongst all these permutations we are only interested in those permutations which result in w being in $DSS(v)$. But those are exactly those where the rank of w is maximal amongst the ranks of $v_0, \ldots, v_r$. So given that all nodes $v_0, \ldots, v_r$ have rank $\leq R$, the probability of w being in $DSS(v)$ is $1/(r+1) < r^{-1}$. Hence the probability for w being in $DSS(v)$ and having rank at most R is upper bounded by $r^{-(c'+1)}$. ■

For appropriate choice of c', we can now actually prove that with high probability there is no node in $DSS(v)$ with too small a rank:

Lemma 9 *For $c' > 2$, the probability that there exist vertices $v \in V$ and $w \in DSS(v)$ with the shortest path from v to w having length $r > n^{1/4}$ and the rank of w being less than $n(1 - c'/r \ln r)$ is upper bounded by $n^{\frac{-c'+7}{4}}$.*

Proof. There are less than n^2 pairs (v, w) to consider. According to Lemma 8, the probability of $w \in DSS(v)$ and the rank of w being small is bounded by $r^{-(c'+1)}$ for each of them,. For $r > n^{1/4}$ this probability is upper bounded by $n^{-(c'+1)/4}$. The statement follows by the union bound over the at most n^2 bad events. ■

This suggests a modification of the standard search procedure and pushing nodes w with tentative distance $d(w)$ into the priority only if (1) $d(w) \leq n^{1/4}$ or (2) $d(w) > n^{1/4}$ and $rank(w) \geq n\left(1 - \frac{c' \ln d(w)}{d(w)}\right)$. So when constructing a CH based on a random permutation of the nodes and employing the above search procedure, with high probability, all queries are answered correctly. There is a slim chance less than $n^{\frac{-c'+7}{4}}$ that we incorrectly prune out one or more nodes in one of the direct search spaces.

Lemma 10 *The expected size of the search space for an s-t query is bounded by $\mathcal{O}(\sqrt{n} \log n)$*

Proof. Let $x_{s,r}$ denote the number of nodes at distance r from s. The number of nodes with $r \leq n^{1/4}$ can be easily bounded by $\sum_{r=0}^{n^{1/4}} c \cdot r \leq c\sqrt{n} \in \mathcal{O}(\sqrt{n})$. For the nodes further away, we only sum up the ones with a rank $> n(1 - \frac{c' \ln r}{r})$. The probability of a node to have such a rank is $\frac{c' \ln r}{r}$. So we are interested in $\sum_{r=n^{1/4}}^{D} x_{s,r} \frac{c' \ln r}{r}$. As $\frac{c' \ln r}{r}$ decreases with growing r, and the sum is maximized for $D = \sqrt{2n}$, we can upper bound the sum as follows:

$$\sum_{r=n^{1/4}}^{D} x_{s,r} \frac{c' \ln r}{r} \leq \sum_{r=1}^{\sqrt{2n}} c \cdot r \frac{c' \ln r}{r} = c \cdot c' \sum_{r=1}^{\sqrt{2n}} \ln r$$
$$\leq c \cdot c' \sqrt{2n} \ln \sqrt{2n} \in \mathcal{O}(\sqrt{n} \log n)$$

So in total, the search graph of s contains at most $\mathcal{O}(\sqrt{n}\log n)$ nodes. The same argumentation holds for t respectively, the statement of the Lemma follows. ■

From Monte Carlo to Las Vegas Preprocessing. The randomized CH construction with the modified query routine guarantees an expected data structure size of $\mathcal{O}(n\log D)$ and expected search space size of $\mathcal{O}(\sqrt{n}\log n)$, yet the outcome is only correct with probability at least $1-n^{(-c'+7)/4}$. It is not difficult, though, to guarantee that *no* far away nodes exhibit too small a rank. This can be done by performing a Dijkstra computation from each node and checking that all nodes further away than $n^{1/4}$ exhibit large enough rank. If a node with too small a rank is found, the whole preprocessing is repeated. For a choice of $c' \geq 8$, in expectation less than one repetition is necessary. The construction cost can be bounded by $\mathcal{O}(n^2\log^2 n + mn\log n)$, the verification has cost $\mathcal{O}(n^2\log n + nm)$, hence we end up with the following theorem:

Theorem 11 *For bounded growth graphs, a randomized CH with $\mathcal{O}(n\log D)$ shortcuts and search space sizes of $\mathcal{O}(\sqrt{n}\log n)$ can be computed in expected $\mathcal{O}(n^2\log^2 n + nm\log n)$ time.*

Remember that for $h,t \in \mathcal{O}(\sqrt{n})$, for the number of shortcuts only a bound of $\mathcal{O}(n\sqrt{n}\log n)$ could be obtained in previous work. Our result for bounded growth graphs is better by a factor of $\sqrt{n}$ (with matching or improved search space sizes), and matches results previously only achieved for planar graphs and minor-closed graphs with balanced separators (Bauer et al. 2013; Milosavljević 2012). Note also that the space consumption of a CH in dependency of h was lower bounded by $\Theta(nh\log D)$ in (White 2015). Hence our space consumption of $\mathcal{O}(n\log D)$ cannot be beaten by any analysis based on (even constant) highway dimension.

Transit Nodes

The TN algorithm relies on the observation that all optimal paths from some small region to faraway destinations (for some notion of far) pass through a small set of so-called access nodes. If 'far' is defined as having a distance at least r, this means, there needs to be a concise hitting set for all shortest paths that leave/enter the ball $B_r(v)$ for every node $v \in V$. This hitting set is called the access node set $AN(v)$ of v. The union of all access node sets forms the transit node set T. For a suitable radius r, access node sets of close-by nodes can have large intersections, hence it is possible to construct small transit node sets in practice. This is important, because for every pair of transit nodes, the shortest path distance is precomputed and stored in a look-up table. Therefore the space consumption is quadratic in the number of transit nodes. In addition, every node stores the distances to all its access nodes. So the total space consumption can be expressed as $|T|^2 + \sum_{v\in V}|AN(v)|$.

There are two types of queries in the end, 'long' queries with $d_s(t) > r$, and 'short' queries. A 'long' s-t-query reduces to check all access node distances of s and t and the respective distances between them, all of which is precomputed. Hence the query time is in $|AN(s)|\cdot|AN(t)|$. For 'short' queries, the TN approach does not guarantee correctness. Therefore, a local Dijkstra computation (up to radius r) is used as fall-back. As $d_s(t)$ is not known beforehand, the query is first treated as long query. If the distance value returned is $\geq 5r$, the result is correct for sure, see (Eisner and Funke 2012). Otherwise, the query is treated subsequently as 'short' query and the best outcome of both query procedures is returned.

Previous Analysis

In (Abraham et al. 2013), the TN approach was analyzed by first fixing $|T| \leq \sqrt{m}$ and then computing hitting sets T_r for $r = D, D/2, \cdots$ to choose the smallest r such that $|T_r| \leq \sqrt{m}$ holds. Access nodes are then computed per node v by collecting the set of transit nodes that first hit a shortest path starting at v. Note that the radius r to distinguish short and long queries can not be fixed a priori with this approach, but is only known after the preprocessing.

In the end, the space consumption is dominated by storing the access node sets. This implies that the total space consumption might be improved by allowing a slightly larger transit node set. In the following, we provide a parametrized analysis which allows to balance the space consumption better. This also enables to fix the radius r for short queries a priori. Our preprocessing time will turn out to be subquadratic for reasonable choices of r. This is a significant improvement compared to (Abraham et al. 2013), where both preprocessing time and space are superquadratic at best (and cubic in a naive implementation). Furthermore, our query times will be sublinear even if the networks contains large grid-like structures.

VC-Dimension and ϵ-Net Construction

In our TN construction, we do not fix the transit node set size a priori. Instead we consider the radius r as a parameter. The goal is then to hit all shortest paths of length at least r with a concise set of nodes.

The general hitting set problem is NP-hard with an inapproximability bound of $\ln|\mathcal{S}|(1-o(1))$ with $|\mathcal{S}|$ being the number of sets in the system. Better bounds are achievable, though, if the set system has a low VC-dimension (Vapnik and Chervonenkis 2015). In case S is a set of unique shortest paths, it was proven that the VC-dimension d is at most 2 for undirected networks (Abraham et al. 2011a; Tao, Sheng, and Pei 2011) and 3 for directed networks (Funke, Nusser, and Storandt 2014). For a set system with VC-dimension d, a hitting set of size $\mathcal{O}(d\log(dOPT)\cdot OPT)$ can be computed efficiently (Brönnimann and Goodrich 1995). The fact that shortest path sets have a constant VC-dimension was exploited in (Abraham et al. 2013) to find hitting sets of size $\mathcal{O}(h\log h)$ instead of $\mathcal{O}(h\log n)$ in polytime.

In our analysis, we use the fact that for systems with constant VC-dimension d, there exist small ϵ-nets. An ϵ-net for $\epsilon \in [0,1]$ is a hitting set for all sets $S \in \mathcal{S}$ with $|S| \geq \epsilon|U|$. So setting $\epsilon n = r$, the respective ϵ-net is a valid set of transit nodes for parameter r. It was shown that a random sample of U of size $d/\epsilon \log 1/\epsilon$ is an ϵ-net with constant probability (Haussler and Welzl 1986). For a random sample of size $|T| = \mathcal{O}(n/r \log n/r)$, we can check whether it is indeed an r/n-net for G in polynomial time. Hence, we can find in expected polynomial time a true r/n-net which then serves as transit node set T.

It remains to compute the expected number of access nodes for $v \in V$. As all shortest paths of length $\geq r$ are hit, it suffices to count the ϵ-net nodes in an r-radius around v, that is $AN(v) = B_r(v) \cap T$. According to our bounded growth model $|B_r(v)| \in \mathcal{O}(r^2)$. The probability for some node to be in T is $d/r \log n/r$ (with $d = 2$ or $d = 3$). Hence the expected number of access nodes is $E(|AN(v)|) = \mathcal{O}(r \log n/r)$.

Theorem 12 *In bounded growth graphs, TN can be computed in expected $\mathcal{O}(n^2/r \log^2 n + nr^2 \log r)$ time. The expected space consumption is in $\mathcal{O}(n^2/r^2 \log^2 n + nr \log n)$ and the expected query time in $\mathcal{O}(r^2 \log^2 n)$.*

Proof. The first phase of the preprocessing consists of choosing a random sample of size $\mathcal{O}(n/r \log n/r)$, and checking its validity. The check requires Dijkstra runs from each $v \in V$ up to distance r, making sure that every shortest path of length r is hit. This takes a total time of $\mathcal{O}(nr^2 \log r)$ when invoking the bounded growth model. As the success probability for a random sample to be a valid net is constant, we only expect a constant number of repetitions to find a valid net. It remains to compute the pairwise distances between the transit nodes, which can be accomplished in $\mathcal{O}(|T|n \log n) = \mathcal{O}(n^2/r \log^2 n)$. The computation of the access node distances for every $v \in V$ is equivalent to the validity check. So the total preprocessing time is in $\mathcal{O}(n^2/r \log^2 n + nr^2 \log r)$. The expected space consumption can be computed as $|T|^2 + \sum_{v \in V} E(|AN(v)|) = \mathcal{O}(n^2/r^2 \log^2 n + nr \log n)$ with $|T| = \mathcal{O}(n/r \log n)$ and $E(|AN(v)|) = \mathcal{O}(r \log n)$, upper bounding $\log n/r$ terms by $\log n$. The expected query time is $E(|AN(v)|)^2 = \mathcal{O}(r^2 \log^2 n)$ for long queries and $\mathcal{O}(r^2 \log r)$ for short queries, hence $\mathcal{O}(r^2 \log^2 n)$ in total. ■

The above Theorem allows to choose r such that the total space consumption is minimized.

Lemma 13 *In bounded growth graphs, TN can be computed in randomized polynomial time with an expected space consumption of $\mathcal{O}(n^{4/3} \log^{4/3} n)$ and expected query times of $\mathcal{O}(n^{2/3} \log^{8/3} n)$.*

Proof. Let $f(r) = n^2/r^2 \log^2 n + nr \log n$ describe the space consumption up to a constant factor. To find the global minimum, we compute $f'(r) = -2n^2/r^3 \log^2 n + n \log n = 0$. Rearranging this term results in $r^3 = 2n \log n$. Hence the radius minimizing the space consumption is in $\Theta(n^{1/3} \log^{1/3} n)$. Inserting this term in the formulas for space consumption and query time given in Theorem 12 concludes the proof. ■

We observe that under the assumption $h \in \Theta(\sqrt{n})$, our new bound $\mathcal{O}(n^{4/3} \log^{4/3} n)$ is the smallest known space consumption bound for TN; and we achieve sublinear query times using this space. But we can also allow the same space consumption as required in (Abraham et al. 2013) with polynomial time preprocessing and analyze the query times under this condition.

Lemma 14 *In bounded growth graphs, ϵ-net based TN can be computed in randomized polynomial time with a space consumption of $\mathcal{O}(n\sqrt{n} \log n)$ and query times of $\mathcal{O}(\sqrt{n} \log^4 n)$.*

Proof. The goal is to find the smallest r such that $\mathcal{O}(n^2/r^2 \log^2 n + nr \log n) = \mathcal{O}(n\sqrt{n} \log n)$. The first summand requires $r \geq cn^{1/4} \log n$, the second $r \in \mathcal{O}(\sqrt{n})$. As $\log n \in \mathcal{O}(n^{1/4})$, both conditions are fulfilled for $r \in \Theta(n^{1/4} \log n)$. The respective query time is then $\mathcal{O}(n^{1/2} \log^2 n \log^2 n) = \mathcal{O}(\sqrt{n} \log^4 n)$. ■

Compared to the query time of $\mathcal{O}((h \log h)^2) = \mathcal{O}(n \log^2 n)$ for $h \in \Theta(\sqrt{n})$, our query times are better by a factor of $\sqrt{n}/\log^2$ and clearly sublinear. If we allow a space consumption of $\mathcal{O}(n\sqrt{n} \log^2 n)$ as required for the special TN-variant in (Abraham et al. 2010) with polynomial preprocessing, we get $r = n^{1/4}$ as smallest feasible radius and hence a query time of $\mathcal{O}(\sqrt{n} \log^2 n)$. This matches the result in (Abraham et al. 2010) assuming $h \in \Theta(\sqrt{n})$.

Corollary 15 *For $r = \Theta(n^\epsilon \log^{\mathcal{O}(1)} n)$ the preprocessing time is subquadratic for all $\epsilon \in]0, 0.5[$ according to Theorem 12.*

We observe that this Corollary applies for all our analyzed radius values $r \in \Theta(n^{1/3} \log^{1/3} n)$, $r \in \Theta(n^{1/4})$ and $r \in \Theta(n^{1/4} \log n)$.

In summary, our analysis in the bounded growth model yields better bounds on the space consumption, the query times and the preprocessing times for TN compared to the h-dependent analysis for $h \in \Theta(\sqrt{n})$.

Conclusions and Open Problems

We showed that the bounded growth model is sufficient to prove sublinear search spaces for contraction hierarchies, transit nodes and hub labels. Results for all three approaches were previously only available in dependency of the highway dimension of the network. For graphs where the highway dimension (as well as the treewidth and the skeleton dimension) is in the order of $\mathcal{O}(\sqrt{n})$ – which is the case for grid graphs and also most likely for many road network instances – our derived bounds on the search space sizes, the space consumption, and the preprocessing times match previous results or are superior.

In terms of analyzing contraction hierarchies, it is still open how to get a sublinear bound on the number of *edge relaxations* during a query when allowing grid structures. Furthermore, lower bounds for query times and space consumption when assuming bounded growth would be of interest to see if our analyses are tight.

References

Abraham, I.; Fiat, A.; Goldberg, A. V.; and Werneck, R. F. F. 2010. Highway dimension, shortest paths, and provably efficient algorithms. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, 782–793.

Abraham, I.; Delling, D.; Fiat, A.; Goldberg, A. V.; and Werneck, R. F. F. 2011a. VC-dimension and shortest path algorithms. In *Proc. 38th International Colloquium on Automata, Languages and Programming, (ICALP)*, 690–699.

Abraham, I.; Delling, D.; Goldberg, A. V.; and Werneck, R. F. 2011b. A hub-based labeling algorithm for shortest paths in road networks. In *International Symposium on Experimental Algorithms*, 230–241. Springer.

Abraham, I.; Delling, D.; Fiat, A.; Goldberg, A. V.; and Werneck, R. F. 2013. Highway dimension and provably efficient shortest path algorithms. Technical Report MSR-TR-2013-91, Microsoft Research.

Antsfeld, L.; Harabor, D. D.; Kilby, P.; Walsh, T.; et al. 2012. Transit routing on video game maps. In *AIIDE*.

Bast, H.; Funke, S.; Sanders, P.; and Schultes, D. 2007. Fast routing in road networks with transit nodes. *Science* 316(5824):566–566.

Bauer, R.; Columbus, T.; Rutter, I.; and Wagner, D. 2013. Search-space size in contraction hierarchies. In *Proc. 40th International Colloquium on Automata, Languages, and Programming (ICALP)*, 93–104.

Brönnimann, H., and Goodrich, M. T. 1995. Almost optimal set covers in finite vc-dimension. *Discrete & Computational Geometry* 14(4):463–479.

Delling, D.; Goldberg, A. V.; Pajor, T.; and Werneck, R. F. 2014. Robust exact distance queries on massive networks. *Microsoft Research, USA, Tech. Rep* 2.

Dibbelt, J.; Strasser, B.; and Wagner, D. 2014. Customizable contraction hierarchies. In *Proc. 13th International Symposium on Experimental Algorithms (SEA)*, 271–282.

Eisner, J., and Funke, S. 2012. Transit nodes lower bounds and refined construction. In *Proc. 14th Workshop on Algorithm Engineering and Experiments (ALENEX)*, 141–149.

Funke, S., and Storandt, S. 2015. Provable efficiency of contraction hierarchies with randomized preprocessing. In *Proc. 26th International Symposium on Algorithms and Computation ISAAC*, 479–490.

Funke, S.; Nusser, A.; and Storandt, S. 2014. On k-path covers and their applications. *Proceedings of the VLDB Endowment* 7(10):893–902.

Geisberger, R.; Sanders, P.; Schultes, D.; and Vetter, C. 2012. Exact routing in large road networks using contraction hierarchies. *Transportation Science* 46(3):388–404.

Haussler, D., and Welzl, E. 1986. Epsilon-nets and simplex range queries. In *Proceedings of the Second Annual Symposium on Computational Geometry*, SCG '86, 61–71.

Kosowski, A., and Viennot, L. 2017. Beyond highway dimension: Small distance labels using tree skeletons. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, 1462–1478. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Kuhn, F.; Moscibroda, T.; Nieberg, T.; and Wattenhofer, R. 2005. Fast deterministic distributed maximal independent set computation on growth-bounded graphs. *Distributed Computing* 273–287.

Milosavljević, N. 2012. On optimal preprocessing for contraction hierarchies. In *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, 33–38. ACM.

Storandt, S. 2013. Contraction hierarchies on grid graphs. In *KI 2013: Advances in Artificial Intelligence*. Springer. 236–247.

Tao, Y.; Sheng, C.; and Pei, J. 2011. On k-skip shortest paths. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, 421–432. ACM.

Vapnik, V. N., and Chervonenkis, A. Y. 2015. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of Complexity*. Springer. 11–30.

White, C. 2015. Lower bounds in the preprocessing and query phases of routing algorithms. In *Proc. 23rd Annual European Symposium on Algorithms (ESA)*, 1013–1024.