



Sublinear search spaces for shortest path planning in grid and road networks

Johannes Blum¹ · Stefan Funke² · Sabine Storandt¹

Accepted: 8 July 2021 / Published online: 29 July 2021
© The Author(s) 2021

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. However, 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. Still, these parameters may be large in case 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. Furthermore, our preprocessing methods are close to the ones used in practice and only require expected polynomial time.

Keywords Shortest path planning · Bounded growth model · Road network dimensions

✉ Johannes Blum
blum@inf.uni-konstanz.de
Stefan Funke
funke@fmi.uni-stuttgart.de
Sabine Storandt
storandt@inf.uni-konstanz.de

¹ Universität Konstanz, Universitätsstr. 10, 78457 Konstanz, Germany

² Universität Stuttgart, Universitätsstr. 38, 70569 Stuttgart, Germany

1 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 like 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, search space sizes in $\mathcal{O}(h \log D)$ were proven (with D denoting the network diameter). For TN, query times of $\mathcal{O}(h^2)$ are possible. The space consumption was shown to be in $\mathcal{O}(nh \log D)$ or $\mathcal{O}(hn)$, respectively. Note that there is a distinction between search space size and query time. The search space size is defined as the number of nodes that are considered during query answering, while the query time also accounts for additional operations required for query answering. For HL these two are asymptotically the same as the usual query answering routine takes time linear in the number of considered nodes. For TN, the query time is asymptotically quadratic in the number of considered nodes. For CH, query answering is more involved and hence the search space size and the query time cannot be related in a straightforward manner. While the query time is at most quadratic in the search space size, scalability experiments conducted on large networks suggest that the query time is clearly subquadratic in the search space size (Blum and Storandt 2018b).

Given the existing bounds for CH, HL, and TN based on the highway dimension, one could conjecture be that real-world networks exhibit a highway dimension which is (poly)logarithmic in the size n of the network. However, in grid graphs with uniform costs we have $h \in \Theta(\sqrt{n})$ (Abraham et al. 2010). 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; Dellling et al. 2014). Accordingly, the notion of highway dimension alone is not sufficient to explain all these results as, e.g., the query times of TN are at least linear for $h \in \Theta(\sqrt{n})$. Moreover, as many real-world road networks contain large grid like substructures, a highway dimension in the same order is to be expected. For the road networks of the U.S. and Germany with about 20 million nodes each, a lower bound on the highway dimension of more than 500 is known (Blum and Storandt 2018a). In Kosowski and Viennot (2017) it was shown that the grid-like street net-

work of Brooklyn already has a highway dimension of at least 172, which was proven through a packing lower bound. Due to the high complexity of highway dimension computation, exact values are currently not known even for small networks.

We will use a different model, assuming that the underlying graph metric has bounded growth. This model subsumes grid graphs, and its validity can be tested on general networks in polynomial time. We show that the bounded growth assumption suffices to prove sublinear search spaces, a clearly subquadratic space consumption, and in expectation polynomial time preprocessing for CH, TN, and HL in unweighted graphs.

1.1 Related work

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

For HL, the skeleton dimension k (Kosowski and Viennot 2017) is suitable to prove a theoretical search space size of $\mathcal{O}(k \log D)$ and a space consumption of $\mathcal{O}(nk \log D)$. It is known that graphs with constant maximum degree have a skeleton dimension of $k \in \mathcal{O}(h)$ (Kosowski and Viennot 2017), but that in general, highway dimension and skeleton dimension are incomparable (Blum 2019). In Gupta et al. (2019), the skeleton dimension was used to analyze the performance of 3-hopsets, another route planning technique similar to TN. For TN and CH, however, no bounds in dependency of the skeleton dimension are known so far. In Bauer et al. (2013), CH were analyzed by considering only the topology of the network. For graphs with a treewidth t , the search space size was shown to be in $\mathcal{O}(t \log n)$ and the space consumption in $\mathcal{O}(nt \log n)$.

However, the models based on highway dimension, skeleton dimension, and treewidth cannot fully explain the performance of CH, HL, and TN on grid graphs: Abraham et al. (2010) observed that a square grid with unit edge lengths has highway dimension $h \in \Theta(\sqrt{n})$. There are grid-like graphs of constant skeleton dimension and highway dimension $h \in \Theta(\sqrt{n})$, but on a unit length square grid we also obtain a skeleton dimension of $k \in \Theta(\sqrt{n})$. Moreover, it is a well-known fact that the treewidth of a square grid is $t \in \Theta(\sqrt{n})$, see e.g., Robertson and Seymour (1986). This means for instance that for unit grid graphs, the analyses based on highway dimension, skeleton dimension, and treewidth only give us upper bounds of $\mathcal{O}(\sqrt{n} \log D)$ or $\mathcal{O}(\sqrt{n} \log n)$ on the search space sizes of CH and HL, which appears far too pessimistic, given the actual performance of these algorithms on grid graphs.

The bounded growth model was introduced in Funke and Storandt (2015). Assuming this model, CH were analyzed by drawing a connection to randomized skip lists (Pugh 1990). Queries were proven to be answered correctly with high probability (w.h.p.) with an expected search space size of $\mathcal{O}(\sqrt{n} \log n)$. We will also use the bounded growth model in our analysis, improving the previous result for CH, and showing that this model can also be used to explain the good performance of TN and HL.

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 (treewidth)

Preprocessing		Original result	Result for $h, k, t \in \mathcal{O}(\sqrt{n})$
NP	P	P*	
CH: graphs with highway dimension h (Abraham et al. 2013)			
•	◦	◦	$\mathcal{O}(h \log D)$
•	◦	◦	$\mathcal{O}(nh \log D)$
CH: graphs with highway dimension h (Abraham et al. 2011a, 2013)			
◦	•	◦	$\mathcal{O}(h \log h \log D)$
◦	•	◦	$\mathcal{O}(nh \log h \log D)$
CH: minor-closed graphs with balanced separators (Bauer et al. 2013; Dibbelt et al. 2014)			
•	◦	◦	$\mathcal{O}(\sqrt{n})$
•	◦	◦	$\mathcal{O}(n \log n)$
CH: graphs with treewidth t (Bauer et al. 2013; Dibbelt et al. 2014)			
•	◦	◦	$\mathcal{O}(t \log n)$
•	◦	◦	$\mathcal{O}(nt \log n)$
CH: bounded growth model, correct queries w.h.p. (Funke and Storandt 2015)			
◦	•	•	$\mathcal{O}(\sqrt{n} \log n)$
◦	•	•	$\mathcal{O}(n \log^2 n)$
CH: bounded growth model (NEW, Theorem 2)			
◦	•	•	$\mathcal{O}(\sqrt{n} \log n)$
◦	•	•	$\mathcal{O}(n \log D)$
TN: graphs with highway dimension h , special TN variant (Abraham et al. 2010)			
•	◦	◦	$\Delta + h \log D$
			$\Delta + \sqrt{n} \log D$

Table 1 continued

Preprocessing			Original result	Result for $h, k, t \in \mathcal{O}(\sqrt{n})$
NP	P	P*		
•	◦	◦	$\mathcal{O}(nh \log D)$	$\mathcal{O}(n\sqrt{n} \log D)$
TN: graphs with highway dimension h , special TN variant (Abraham et al. 2010)				
◦	•	◦	$\Delta + h \log n \log D$	$\Delta + \sqrt{n} \log n \log D$
◦	•	◦	$\mathcal{O}(nh \log n \log D)$	$\mathcal{O}(n\sqrt{n} \log n \log D)$
TN: graphs with highway dimension h (Abraham et al. 2013)				
•	◦	◦	$\mathcal{O}(h^2)$	$\mathcal{O}(n)$
•	◦	◦	$\mathcal{O}(nh + m)$	$\mathcal{O}(n\sqrt{n} + m)$
TN: graphs with highway dimension h (Abraham et al. 2013)				
◦	•	◦	$\mathcal{O}((h \log h)^2)$	$\mathcal{O}(n \log^2 n)$
◦	•	◦	$\mathcal{O}(nh \log n + m)$	$\mathcal{O}(n\sqrt{n} \log n + m)$
TN: graphs with skeleton dimension k (NEW, Theorem 4)				
◦	•	•	$\mathcal{O}(k^2 \log^2 n)$	$\mathcal{O}(n \log^2 n)$
◦	•	•	$\mathcal{O}(n \log n(k + \log n))$	$\mathcal{O}(n\sqrt{n} \log n)$
TN: bounded growth model (NEW, Lemma 12)				
◦	•	•	$\mathcal{O}(n^{2/3} \log^{8/3} n)$	
◦	•	•	$\mathcal{O}(n^{4/3} \log^{4/3} n)$	
TN: bounded growth model (NEW, Lemma 13)				
◦	•	•	$\mathcal{O}(\sqrt{n} \log^4 n)$	
◦	•	•	$\mathcal{O}(n\sqrt{n} \log n)$	
HL: graphs with highway dimension h (Abraham et al. 2013)				
•	◦	◦	$\mathcal{O}(h \log D)$	$\mathcal{O}(\sqrt{n} \log D)$
•	◦	◦	$\mathcal{O}(nh \log D)$	$\mathcal{O}(n\sqrt{n} \log D)$

Table 1 continued

Preprocessing		Original result	Result for $h, k, t \in \mathcal{O}(\sqrt{n})$
NP	P P*		
HL: graphs with highway dimension h (Abraham et al. 2011a, 2013)			
○	● ○	$\mathcal{O}(h \log h \log D)$	$\mathcal{O}(\sqrt{n} \log n \log D)$
○	● ○	$\mathcal{O}(nh \log h \log D)$	$\mathcal{O}(n\sqrt{n} \log n \log D)$
HL: graphs with skeleton dimension k (Kosowski and Viennot 2017)			
○	● ●	$\mathcal{O}(k \log D)$	$\mathcal{O}(\sqrt{n} \log D)$
○	● ●	$\mathcal{O}(nk \log D)$	$\mathcal{O}(n\sqrt{n} \log D)$
HL: graphs with treewidth t (Gavoille et al. 2004, Corollary 2)			
●	○ ○	$\mathcal{O}(t \log n)$	$\mathcal{O}(\sqrt{n} \log n)$
●	○ ○	$\mathcal{O}(nt \log n)$	$\mathcal{O}(n\sqrt{n} \log n)$
HL: bounded growth model (NEW, Theorem 1)			
○	● ●	$\mathcal{O}(\sqrt{n})$	
○	● ●	$\mathcal{O}(n\sqrt{n})$	

Preprocessing markers indicate whether the underlying method for preprocessing poses an NP-hard problem (NP), or can be computed in polynomial time (P) or expected polynomial time (P*). The rows with an ss flag provide result about the search space size of the respective technique, the ones flagged qt show query times, and the rows flagged sc give space consumption bounds. Space consumption is measured in machine words

1.2 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. Here, we build on and significantly extend the results from the conference version (Blum et al. 2018). More precisely, we provide the following results:

- We analyze the relationship between bounded growth and the skeleton dimension of the graph. We show that the skeleton dimension of an unweighted bounded growth graph is in $\mathcal{O}(\sqrt{n})$. We use this relation to prove bounds on HL search space sizes. On the other hand, we show that even a constant skeleton dimension does not imply the bounded growth property.
- We formalize an observation by Abraham et al. (2011b) which states that on any graph, the HL search spaces can be bounded by the CH search spaces. This yields search spaces of size $\mathcal{O}(t \log n)$ and a space consumption of $\mathcal{O}(nt \log n)$ for HL.
- 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 $h \in \Theta(\sqrt{n})$.
- We furthermore discuss how ϵ -net theory can be applied to TN preprocessing on graphs with low skeleton dimension. This is the first analysis of TN based on the skeleton dimension k .
- We extend the results from Blum et al. (2018) to weighted graphs, and describe how our preprocessing algorithms can be modified to handle weighted graphs without explicitly subdividing the whole graph into edges of unit length, which would introduce a large number of unnecessary nodes. Instead, we introduce the maximum edge length ℓ_{\max} as an additional parameter and show that this adds a factor of ℓ_{\max}^2 to the space consumption and a factor of ℓ_{\max} to the search space sizes of CH and TN, whereas the bounds from the unweighted settings still hold in the case of HL.

Our new results are also included in Table 1.

2 Bounded growth and skeleton dimension

We first introduce 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. In Sect. 7, however, we describe how our results can be applied to weighted graphs. 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 by $d_v(w)$ the shortest path distance from v to w . For a node v and a radius r , we define the set of all nodes within a distance r of v as the ball $B_r(v) = \{w \mid d_v(w) \leq r\}$.

2.1 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 there exists some constant $c \in \mathbb{R}^+$ (w.l.o.g. $c \geq 1$) such that for every $r > 0$, the number of nodes at distance r from a node v is at most cr . If this property is fulfilled, it immediately follows that

$$|B_r(v)| \leq cr(r + 1)/2 \in \mathcal{O}(r^2).$$

Intuitively, this reflects the area growth of a disk in the Euclidean plane in dependency of its radius. As Fig. 1 shows, this comes very close to the growth of a ball in real-world road networks.

In Funke and Storandt (2015), it was shown empirically that the bounded growth model represents real-world road networks well. In fact, the parameter c can be computed for a network in polynomial time (the same is true for the skeleton dimension but not for the highway dimension or the treewidth). The results reported in Funke and Storandt (2015) imply that for Euclidean edge costs, $c = 1$ is a good model. 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 treewidth of $\Theta(\sqrt{n})$.

2.2 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 is a subdivision of T_u into 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 *There are graphs with uniform edge costs and constant skeleton dimension k that do not have bounded growth.*

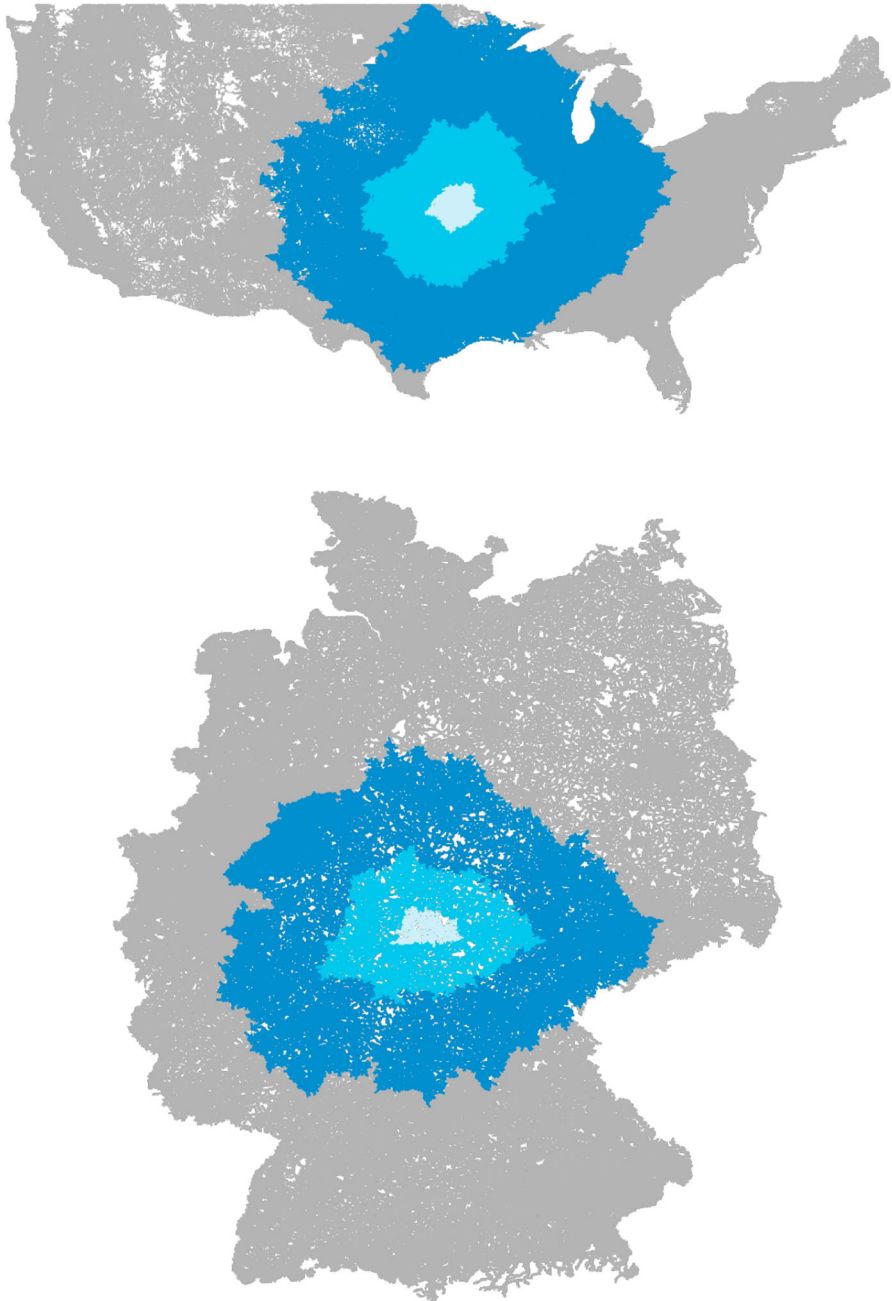
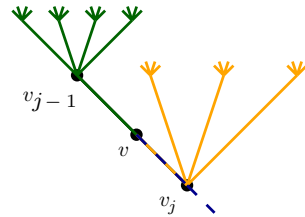


Fig. 1 Balls of different radii in the road networks of the US and Germany. Every ball resembles a disk in the Euclidean plane

Fig. 2 The different branch types of \tilde{T}_v : type (i) in green, type (ii) in blue and type (iii) in yellow (Color figure online)



Proof To construct a graph with the mentioned properties we start with a complete quadtree (a tree where every internal node has four children) of height d rooted at some node s . Then we subdivide every edge connecting two nodes of height i and $(i - 1)$, respectively, into 3^i edges, which gives us a tree T . We show that T has skeleton dimension $k \leq 8$. Consider therefore some node v of T . The node v lies on some simple path between two nodes v_j and v_{j-1} , that were adjacent in the original quadtree and had height j and $(j - 1)$, respectively. If v has degree 5 (which means that it was already contained in the quadtree) we choose $v_j = v$. We have $d_s(v) = \sum_{i=j+1}^d 3^i + \ell$ for some $\ell \in \{0, \dots, 3^j - 1\}$, i.e., v is the ℓ -th node on the path from v_j to v_{j-1} which has length 3^j .

Consider the skeleton T_v^* of v 's shortest path tree T_v . We show that T_v^* contains at most 8 branches which implies $k \leq 8$. We distinguish three different types of branches of the geometric realization \tilde{T}_v of T_v : (i) branches that include v_{j-1} , (ii) branches that include s and (iii) the remaining branches (cf. Fig. 2).

Consider some node u of \tilde{T}_v contained in some type (i) branch. The distance from u to some furthest descendant w in \tilde{T}_v is $d_u(w) = d_v(w) - d_v(u) = \sum_{i=1}^j 3^i - \ell - d_v(u) = \frac{3}{2}(3^j - 1) - \ell - d_v(u)$. For u to be contained in the skeleton we require $d_u(w) \geq \frac{1}{2}d_v(u)$ which gives us $d_v(u) \leq (3^j - 1) - \frac{2}{3}\ell = (3^j - \ell) + \frac{1}{3}\ell - 1 \leq d_v(v_{j-1}) + 3^{j-1} - 1$. As the node v_{j-1} has 4 children and all nodes at distance at most $3^{j-1} - 1$ from v_{j-1} have degree 2, the number of type (i) branches in T_v^* is bounded by 4. Similarly, one can show that the number of type (ii) and (iii) branches are bounded by 1 and 3, respectively, from which the bound on the skeleton dimension follows.

To prove that there is no constant c which bounds the number of nodes at distance r from every node by cr , observe that there are 4^d leaves at distance $\sum_{i=1}^d 3^i = \frac{3}{2}(3^d - 1)$ from the root s . □

Lemma 2 *The skeleton dimension of a bounded growth graph with uniform edge costs is upper bounded by $\mathcal{O}(\sqrt{n})$.*

Proof Consider a bounded growth graph $G(V, E)$ with uniform edge costs and skeleton dimension k . 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 $k \in \mathcal{O}(\sqrt{n})$. □

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 metric 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 3 *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 We have $\sum_{r=1}^D x_{u,r} = n - 1$ and $x_{u,r} \leq cr$. Hence, the sum $\sum_{r=1}^D x_{u,r}/r$ is maximized for

$$x_{u,v} = \begin{cases} cr & \text{if } r \leq \sqrt{2n} \\ 0 & \text{otherwise.} \end{cases}$$

It follows that

$$\sum_{r=1}^D x_{u,r}/r \leq \sum_{r=1}^{\sqrt{2n}} x_{u,r}/r \leq \sqrt{2n} \cdot c \in \mathcal{O}(\sqrt{n}). \square$$

□

Corollary 1 *The integrated skeleton dimension $\hat{k}(u)$ of any vertex u in a bounded growth graph is in $\mathcal{O}(\sqrt{n})$.*

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.

3 Preprocessing-based shortest path algorithms

In this section we briefly introduce the algorithms contraction hierarchies (CH), transit nodes (TN), and hub labels (HL), which we will study in the remainder of this paper. Given two vertices s and t of a (weighted) graph, they compute the shortest path distance $d_s(t)$ from s to t . Note that in many applications, it is more desirable to compute the actual shortest path from s to t , not only its length. However, with minor modifications, all three algorithms can also be used to compute shortest paths. As we already mentioned before, CH, TN, and HL consist of two phases: In the preprocessing phase, auxiliary Data is computed, which is used in the query phase to quickly compute the shortest path distance $d_s(t)$ for a given query pair (s, t) .

3.1 Hub labels

In the hub labels (HL) approach (Abraham et al. 2011b), every node v gets assigned a label set $L(v)$. 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. Provided that the label sets are sorted, we can compute the intersection $L(s) \cap L(t)$ by a merging-like step in linear time. Therefore the query time is in $\mathcal{O}(|L(s)| + |L(t)|)$, while the space consumption is in $\mathcal{O}(\sum_{v \in V} |L(v)|)$.

3.2 Contraction hierarchies

In the preprocessing phase of contraction hierarchies (CH), a so called overlay graph is computed, based on the node contraction operation. Contracting a node v means to delete it from the graph, and to insert shortcut edges between its neighbors 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 empty. In the end, the overlay graph is given by $G^+(V, E \cup E^+)$, where E^+ is the set of shortcut edges inserted during the contraction process. This means that the space consumption of CH is determined by $\mathcal{O}(|E^+|)$.

The rank of a node in the order of contraction is called $rank(v)$. To answer an s - t -query, bidirectional Dijkstra runs are used from s and t in G^+ . However, 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 yields the correct result. 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 query answering, but the space consumption and the search space sizes heavily depend on the contraction order.

Relation to Hub Labels Abraham et al. (2011b) observed that hub labels can be computed based on CH. More precisely, we can choose the label set $L(v)$ of every node $v \in V$ as the CH search space of v , i.e. the set of all nodes that are visited in a CH-Dijkstra run from v . However, in order to fulfill the cover property, it suffices to consider only the vertices w for which the shortest v - w -path contains only nodes of rank at most $rank(w)$, the so called direct search space $DSS(v)$. An illustration can be found in Fig. 3. We obtain the following lemma.

Lemma 4 *Consider a CH overlay graph $G^+(V, E \cup E^+)$ of a graph $G(V, E)$ and let $DSS(v)$ be the direct CH search space of a node v . There are HL label sets such that for any node v the label size is bounded by $|DSS(v)|$ and the total space consumption is bounded by $\sum_{v \in V} |DSS(v)|$.*

It follows straight that there is no graph on which the CH approach is superior to all feasible HL data structures in terms of the search space size. In Lemma 7 (Sect. 5) we will show that in a bounded growth graph the expected size of the direct search space is bounded by $\mathcal{O}(\sqrt{n})$. Lemma 4 hence implies an alternative proof to Theorem 1 (Sect. 4), where we use the relation of bounded growth and skeleton dimension to show

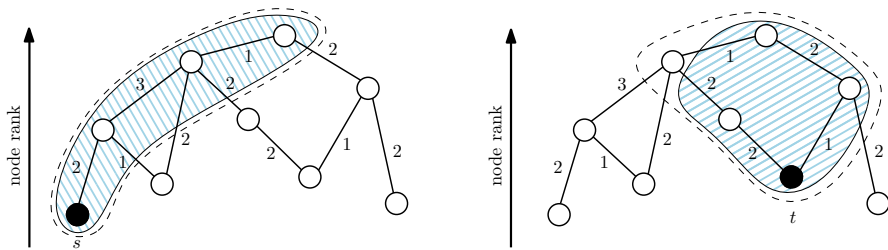


Fig. 3 The search space $SS(v)$ (dashed region) and the direct search space $DSS(v)$ (shaded region) of two nodes in an CH overlay graph. The direct search space constitutes a valid HL label set $L(v)$ for every node v

bounds of $\mathcal{O}(\sqrt{n})$ and $\mathcal{O}(n\sqrt{n})$ on the search space sizes and space consumption of HL, respectively.

In Bauer et al. (2013) it was shown, that for graphs with treewidth t there is a contraction order such that the number of nodes in the resulting CH search space is bounded by $\mathcal{O}(t \log n)$. Exploiting Lemma 4 we can conclude the following corollary which relates the HL data structure and the treewidth.

Corollary 2 *For graphs with treewidth t , there exists a HL data structure where for every node the label size is bounded by $\mathcal{O}(t \log n)$ and the space consumption is $\mathcal{O}(nt \log n)$.*

This matches the label size previously shown by Gavaille et al. (2004).

3.3 Transit nodes

The TN algorithm relies on the observation that all shortest 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 checking 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 fall-back algorithm is used, e.g., a local Dijkstra run up to radius r . As $d_s(t)$ is not known beforehand, the query is first treated as long query. If the distance value returned is at least $5r$, 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.

4 Hub labels

In Abraham et al. (2011a, 2013) hub labels were constructed by computing multiple hitting sets H_r for sets of shortest paths with length $r = 1, 2, 4, \dots, D$. 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 NP-hard preprocessing (computing hitting sets of minimum size), 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 $\mathcal{O}(\log n)$. (Kosowski and Viennot 2017) presented a more practical algorithm for HL, where labels are selected via a randomized process. A thorough analysis shows that it chooses on average $\mathcal{O}(k \log D)$ labels per node, so the total space consumption is in $\mathcal{O}(nk \log D)$.

4.1 Analysis in the bounded growth model

We now briefly sketch the algorithm by Kosowski and Viennot and study its performance on graphs with bounded growth. The algorithm uses edge-based labels, i.e., every label set $L(v)$ is a set of edges such that for every pair of nodes $s, t \in V$, the shortest s - t -path contains some edge $\eta \in L(s) \cap L(t)$. This notion is slightly stronger than the aforementioned node-based approach, but given an edge hub labeling one can easily obtain valid node label sets by choosing from every edge $\eta \in L(v)$ the incident node with minimum id.

The algorithm starts with assigning a random real value $\rho(e) \in [0, 1]$ to every edge e of the graph. Then it chooses for every pair of nodes $u, v \in V$ a *hub edge* $\eta(u, v)$ as the edge with minimum value of ρ amongst all edges (u', v') on the shortest u - v -path that satisfy $d_u(u') \geq \frac{5}{12}d_u(v)$ and $d_u(v') \leq \frac{7}{12}d_u(v)$. Finally, the label set of every node u is chosen as $L(u) = \{\eta(u, v) \mid v \in V\}$. This preprocessing algorithm can be implemented on top of n runs of Dijkstra's algorithm such that it takes time $\mathcal{O}(n^2 \log n)$. It was shown that the average size of a single label set can be bounded by $\mathcal{O}(\frac{1}{n} \sum_{u \in V} \hat{k}(u))$. In combination with Corollary 1 we obtain the following Theorem.

Theorem 1 *HL in bounded growth graphs can be computed in 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 $h \in \Theta(\sqrt{n})$.

5 Contraction hierarchies

Inspired by the topology based analysis of CH (Bauer et al. 2013), another CH construction scheme was developed, based on so-called nested dissections. It was proven that this scheme also leads to excellent performance in practice (Dibbelt et al. 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 than proven in Funke and Storandt (2015).

5.1 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 et al. 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 $\pi = vv_1v_2 \dots v_{r-1}w$ of length r . A shortcut from v to w is created if and only if $\text{rank}(w) > \text{rank}(v)$ and the ranks of all intermediate nodes on π are smaller than $\text{rank}(v)$. 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 :

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

□

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 later 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 $v = v_0v_1v_2 \dots v_r = w$ of the shortest path from v to w . The node w is in $DSS(v)$ iff $rank(v_i) < rank(w)$ for all $i = 0, \dots, r - 1$. Clearly each of the $v_i, i = 0, \dots, r$ has the same probability $1/(r + 1)$ of having the largest rank amongst v_0, v_1, \dots, 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 3. □

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 are very unlikely to 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 $v = v_0v_1 \dots v_r = w$ 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

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

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, \dots, v_r . So given that all nodes v_0, \dots, 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 $r^{-(c'+1)}$. \square

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. \square

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:

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

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. \square

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 2 *For bounded growth graphs, a randomized CH with an expected number of $\mathcal{O}(n \log D)$ 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.

6 Transit nodes

In Abraham et al. (2013), the TN approach was analyzed by first fixing an upper bound $\tau \leq \sqrt{m}$ on the size of the transit node set T and then computing hitting sets T_r for $r = D, D/2, \dots$ to choose $T = T_r$ for the smallest r such that $|T_r| \leq \sqrt{\tau}$ 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.

6.1 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 \mathcal{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 et al. 2011) and 3 for directed networks (Funke et al. 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 polynomial time.

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 .

6.2 Analysis in the bounded growth model

It remains to compute the expected number of access nodes for $v \in V$. As all shortest paths of length at least 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 we have $|B_r(v)| \in \mathcal{O}(r^2)$. 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 3 *TN can be computed in expected $\mathcal{O}(n^2/r \log^2 n + nr^2 \log r)$ time in bounded growth graphs. 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 then $E(|AN(v)|)^2 = \mathcal{O}(r^2 \log^2 n)$ for long queries. For short queries we use the algorithm of Dijkstra up to a distance of r , which takes $\mathcal{O}(r^2 \log r)$ time. Hence, the total runtime is $\mathcal{O}(r^2 \log^2 n)$. \square

For not too large r , the bounds on space consumption and query time hold with high probability.

Lemma 11 *For $r \in \mathcal{O}(\sqrt{n})$, with high probability the space consumption is in $\mathcal{O}(n^2/r \log^2 n + nr \log n)$ and the query time in $\mathcal{O}(r^2 \log^2 n)$.*

Proof We first show that with high probability we have $|AN(v)| \leq 2 \cdot E(|AN(v)|)$ if $r \in \mathcal{O}(n^{1-\epsilon})$ for some $\epsilon > 0$. For any node v , the number of access nodes $AN(v)$ follows a binomial distribution of $|B_r(v)| \in \mathcal{O}(r^2)$ trials with success probability $p \in \mathcal{O}(1/r \log n/r)$. Hoeffding's inequality yields that the probability of $|AN(v)| \geq 2 \cdot E(|AN(v)|) = 2p \cdot \mathcal{O}(r^2)$ is at most

$$e^{-2p^2 \mathcal{O}(r^2)} = e^{-\mathcal{O}(\log^2 n/r)} = e^{-\mathcal{O}(\log^2 n)} = n^{-\mathcal{O}(\log n)} = n^{-\mathcal{O}(1)},$$

where we use the fact that $\log n/r \in \mathcal{O}(\log n)$ if $r \in \mathcal{O}(n^{1-\epsilon})$ for some $\epsilon > 0$. This shows that the query time is in $\mathcal{O}(r^2 \log^2 n)$ with high probability.

Similarly, the number of transit nodes $|T|$ follows a binomial distribution of n trials with probability $p \in \mathcal{O}(1/r \log n/r)$. We obtain that the probability of $|T| \geq 2 \cdot E(|T|) = 2pn$ is at most

$$e^{-2p^2n} = e^{-2 \cdot \mathcal{O}(1/r^2 \log^2 n/r) \cdot n} = e^{-\mathcal{O}(\log^2 n)} = n^{-\mathcal{O}(1)},$$

where we use that $r \in \mathcal{O}(\sqrt{n})$. Moreover, a union bound yields that with high probability, the total size of all access node sets is bounded by $\mathcal{O}(nr \log n)$, and hence, the stated bound on the space consumption also holds with high probability. \square

Note that our approach always computes the correct shortest path distance, as in the bounded growth model we can handle local queries efficiently through the algorithm of Dijkstra. The above Theorem allows to choose r such that the total space consumption is minimized.

Lemma 12 *In bounded growth graphs, TN can be computed in expected 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 $r^3 = 2n \log n$. It follows that 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 3 concludes the proof. \square

We observe that under the assumption $h \in \Theta(\sqrt{n})$, our new bound of $\mathcal{O}(n^{4/3} \log^{4/3} n)$ 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 13 *In bounded growth graphs, ϵ -net based TN can be computed in expected 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 $\mathcal{O}(n^{1/2} \log^2 n \log^2 n) = \mathcal{O}(\sqrt{n} \log^4 n)$. \square

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 $\sqrt{n}/\log^2 n$ 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 3 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 3.

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 $r \in \Theta(n^{1/4} \log n)$. Moreover, for all radii from the Corollary, the concentration bounds from Lemma 11 hold.

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 $h \in \Theta(\sqrt{n})$.

6.3 Graphs with low skeleton dimension

To compute a transit node set T for a graph with skeleton dimension k , we can also choose an ϵ -net for $\epsilon n = 2/3r$. Again, we construct T by selecting a random sample of size $\mathcal{O}(n/r \log n/r)$. The access nodes $AN(v)$ of a node v can now be restricted to the nodes $u \in T \cap B_{2/3r}(v)$ that have a descendant w in v 's shortest path tree satisfying $d_u(w) \geq \frac{1}{3}r$. This guarantees that all shortest paths of length at least r are hit. Every node $u \in B_{2/3r}(v)$ having such a descendant w with $d_u(w) \geq \frac{1}{3}r \geq \frac{1}{2}d_v(u)$ is contained in the skeleton T_v^* , so their number is bounded by $\sum_{i=0}^{2/3r} k \in \mathcal{O}(rk)$. As the probability for some node to be in T is $d/r \log n/r$, the expected number of access nodes is $E(|AN(v)|) = \mathcal{O}(k \log n/r)$. This gives us the following Theorem.

Theorem 4 In graphs with skeleton dimension k , TN can be computed in expected $\mathcal{O}(n^2 \log n)$ time. The expected space consumption is $\mathcal{O}(n^2/r^2 \log^2 n/r + nk \log n/r)$ and the expected query time for long queries is in $\mathcal{O}(k^2 \log^2 n/r)$.

Proof Choosing a random sample of size $\mathcal{O}(n/r \log n/r)$ and checking its validity via n Dijkstra runs can be performed in $\mathcal{O}(n^2 \log n)$ time. In expectation we need to repeat this step a constant number of times until we obtain a valid ϵ -net. The computation of the distances from every node to all of its access nodes and between all transit nodes can be done through n Dijkstra runs in $\mathcal{O}(n^2 \log n)$ time. The expected space consumption for the transit and access nodes is $|T|^2 + \sum_{v \in V} |AN(v)| = \mathcal{O}(n^2/r^2 \log^2 n/r + nk \log n/r)$ and the expected query time for long queries is $(E(|AN(v)|))^2 = \mathcal{O}(k^2 \log^2 n/r)$. \square

To treat short queries, we need an alternative to the algorithm of Dijkstra as it might require $\Omega(n \log n)$ time. A possible solution is to adopt the skeleton-based hub label approach by Kosowski and Viennot (2017), but to consider only pairs of nodes at a distance of at most r , i.e., in the preprocessing step we choose $L(u) = \{\eta(u, v) \mid v \in V: d_u(v) \leq r\}$ for every node u (c.f. Sect. 4.1). Following the analysis from Kosowski and Viennot (2017), the average size of every such ‘local’ label set can be bounded by $\sum_{i=1}^r k/i \in \mathcal{O}(k \log r)$. Therefore, this approach adds a term of $\mathcal{O}(nk \log r)$ to the expected space consumption.

For a skeleton dimension $k \in \mathcal{O}(n/\log n)$, the expected space consumption of TN is $\mathcal{O}(nk \log n/r)$. In order to beat the space consumption of the skeleton-based (global) HL from Kosowski and Viennot (2017), which has an expected space consumption

of $\mathcal{O}(nk \log D)$ for a graph diameter of D , we need to choose $r \in \Omega(n/D)$, which is only useful if $D \in \Omega(\sqrt{n})$. For a radius of $r = n/D$, we obtain an expected query time of $\mathcal{O}(k^2 \log^2 D)$, which is worse than the HL approach by a factor of $k \log D$. Better query times than for HL can only be achieved if the skeleton dimension k is bounded by $\mathcal{O}(\log D)$. In this case we need to choose a radius $r \in \Omega\left(n/e^{\sqrt{1/k \log D}}\right)$ which implies that $r \in \Omega(n/D^\epsilon)$ for any $\epsilon > 0$. As a radius of $r > D$ is pointless, it follows that we require $D \in \Omega\left(n^{1/(1+\epsilon)}\right)$ for any $\epsilon > 0$. Hence, we can only beat the query times of HL on very special networks for large radii.

This complies also with the experimental results from Blum and Storandt (2018b), which suggest that on large networks (CH-based) TN is dominated by (skeleton-based) HL. On real-world road networks with at most two million nodes, however, it was shown that TN still exhibits a lower space consumption than HL at the cost of higher query times, which might be due to hidden constants not included in our analysis.

Note that in Gupta et al. (2019), the skeleton dimension was used for the analysis of so called 3-hopsets, where a graph is augmented with shortcut edges, such that for any pair of nodes there is a shortest path consisting of at most 3 edges. For graphs with skeleton dimension k , the authors could show bounds of $\mathcal{O}(nk \log k \log \log n)$ and $\mathcal{O}(k^2 \log^2 k \log^2 \log n)$ on the space consumption and search space sizes, respectively. We would like to emphasize that the notion of a 3-hopset differs however from TN, as in the former only relevant ‘middle hops’ are stored whereas in the latter approach we store the pairwise distances between all transit nodes. Therefore, any TN data structure is also a valid 3-hopset, but not vice versa.

7 Weighted graphs

In this section we discuss how our algorithms can be applied to graphs $G(V, E)$ that have non-uniform integer edge costs $\ell : E \rightarrow \mathbb{N}$. A naïve approach is to subdivide every edge e into $\ell(e)$ edges of unit length which results in an unweighted graph $H(V \cup V^+, E_H)$, where V^+ are the non-terminal nodes inserted during this process. On real-world road networks this increases the graph size only by a constant factor as the average edge length is usually quite small. Alternatively, one might modify the algorithms as follows in order to avoid the introduction of many unnecessary nodes.

7.1 Hub labels

Our results for HL carry immediately over to weighted graphs as the algorithm from Kosowski and Viennot (2017) does not require an unweighted input graph.

7.2 Contraction hierarchies

One key observation of our CH analysis in unweighted bounded growth graphs is that any shortest path of length r contains $r + 1$ nodes. In the weighted setting this is no

Table 2 Currently known theoretical results for Contraction Hierarchies (CH), Transit Nodes (TN) and Hub Labels (HL) in dependency of highway dimension, skeleton dimension, treewidth and for bounded growth graphs

	Highway dimension	Skeleton dimension	Treewidth	Bounded growth
CH	Abraham et al. (2011a, 2013)	OPEN	Bauer et al. (2013)	Theorem 2
TN	Abraham et al. (2011a)	Theorem 4	OPEN	Lemma 13
HL	Abraham et al. (2011a)	Kosowski and Viennot (2017)	Gavoille et al. (2004), Corollary 2	Theorem 1

longer true, but if we denote the maximum edge length of a given weighted graph by ℓ_{\max} , it holds that the number of nodes of any shortest path of length r is at least r/ℓ_{\max} . If we incorporate this into the analysis of Sect. 5, we obtain that on weighted bounded growth graphs, a random contraction order yields an CH overlay graph with an expected number of shortcuts of $\mathcal{O}(\ell_{\max}^2 n \log D)$ and expected search space sizes of $\mathcal{O}(\ell_{\max} \sqrt{n} \log n)$.

7.3 Transit nodes

Like in the case of CH, we can also extend our analysis of TN to weighted graphs by introducing the maximum edge length ℓ_{\max} as an additional parameter. The central idea of our TN preprocessing is to compute an ϵ -net in order to find a set of nodes that hit all shortest paths of length at least r . On graphs of maximum edge length ℓ_{\max} , any such path contains at least r/ℓ_{\max} many vertices. Hence, we can choose the transit node set T is an ϵ -net for $\epsilon = r/(n\ell_{\max})$. This means that we can compute a transit node set such that the expected space consumption is $\mathcal{O}(\ell_{\max}^2 n^2/r^2 \log^2 n + \ell_{\max} nr \log n)$ and the expected time for a long query is $\mathcal{O}(\ell_{\max}^2 r^2 \log^2 n)$.

8 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 $\Theta(\sqrt{n})$ —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.

Moreover, for TN and HL we could show new bounds in dependency of the skeleton dimension and the treewidth, respectively. Table 2 shows a compilation of theoretical results for the three state-of-the-art route planning techniques. So far, no bounds for CH in dependency of the skeleton dimension or for TN in dependency of the treewidth are known.

In terms of analyzing contraction hierarchies, it is still open to prove a subquadratic relation between the query time and the search space size. Furthermore, lower bounds for query times and space consumption when assuming bounded growth would be of interest to see if our analyses are tight.

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted

by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Abraham I, Fiat A, Goldberg AV, Werneck RF (2010) Highway dimension, shortest paths, and provably efficient algorithms. In: Proceedings of the 21st annual ACM-SIAM symposium on discrete algorithms (SODA), pp 782–793. SIAM . <https://doi.org/10.1137/1.9781611973075.64>
- Abraham I, Delling D, Fiat A, Goldberg AV, Werneck RFF (2011a) VC-dimension and shortest path algorithms. In: Proceedings of the 38th international colloquium on automata, languages and programming, (ICALP), lecture notes in computer science, vol 6755, pp 690–699. Springer. https://doi.org/10.1007/978-3-642-22006-7_58
- Abraham I, Delling D, Goldberg AV, Werneck RF (2011b) A hub-based labeling algorithm for shortest paths in road networks. In: Proceedings of the 10th international conference on experimental algorithms (SEA), lecture notes in computer science, vol 6630, pp 230–241. Springer. https://doi.org/10.1007/978-3-642-20662-7_20
- Abraham I, Delling D, Fiat A, Goldberg AV, Werneck RF (2013) Highway dimension and provably efficient shortest path algorithms. Tech. Rep. MSR-TR-2013-91, Microsoft Research. <https://www.microsoft.com/en-us/research/wp-content/uploads/2013/09/tr-msr-2013-91-rev.pdf>
- Antsfeld L, Harabor DD, Kilby P, Walsh T, et al. (2012) Transit routing on video game maps. In: Proceedings of the 8th AAAI conference on artificial intelligence and interactive digital entertainment (AIIDE). AAAI Press. <http://www.aaai.org/ocs/index.php/AIIDE/AIIDE12/paper/view/5459>
- Bast H, Funke S, Sanders P, Schultes D (2007) Fast routing in road networks with transit nodes. *Science* 316(5824):566–566. <https://doi.org/10.1126/science.1137521>
- Bauer R, Columbus T, Rutter I, Wagner D (2013) Search-space size in contraction hierarchies. In: Proceedings of the 40th international colloquium on automata, languages, and programming (ICALP), lecture notes in computer science, vol 6124, pp 93–104. Springer. <https://doi.org/10.1016/j.tcs.2016.07.003>
- Blum J (2019) Hierarchy of transportation network parameters and hardness results. In: Proceedings of the 14th international symposium on parameterized and exact computation (IPEC), LIPIcs, vol 148, pp 4:1–4:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. <https://doi.org/10.4230/LIPIcs.IPEC.2019.4>
- Blum J, Storandt S (2018a) Computation and growth of road network dimensions. In: Proceedings of the 24th international computing and combinatorics conference (COCOON), lecture notes in computer science, vol 10976, pp 230–241. Springer. https://doi.org/10.1007/978-3-319-94776-1_20
- Blum J, Storandt S (2018b) Scalability of route planning techniques. In: Proceedings of the 28th international conference on automated planning and scheduling (ICAPS), pp 20–28. AAAI Press. <https://aaai.org/ocs/index.php/ICAPS/ICAPS18/paper/view/17741>
- Blum J, Funke S, Storandt S (2018) Sublinear search spaces for shortest path planning in grid and road networks. In: Proceedings of the 32nd AAAI conference on artificial intelligence (AAAI), pp 6119–6126. AAAI Press. <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16271>
- Brönnimann H, Goodrich MT (1995) Almost optimal set covers in finite vc-dimension. *Discrete Comput Geom* 14(4):463–479. <https://doi.org/10.1007/bf02570718>
- Delling D, Goldberg AV, Pajor T, Werneck RF (2014) Robust exact distance queries on massive networks. Tech. Rep. MSR-TR-2014-12, Microsoft Research. <https://www.microsoft.com/en-us/research/wp-content/uploads/2014/07/complexTR-rev2.pdf>
- Dibbelt J, Strasser B, Wagner D (2014) Customizable contraction hierarchies. In: Proceedings of the 13th international symposium on experimental algorithms (SEA), lecture notes in computer science, vol 8504, pp 271–282. Springer. <https://doi.org/10.1145/2886843>
- Eisner J, Funke S (2012) Transit nodes—lower bounds and refined construction. In: Proceedings of the 14th workshop on algorithm engineering and experiments (ALENEX), pp 141–149. SIAM/Omnipress. <https://doi.org/10.1137/1.9781611972924.14>
- Funke S, Storandt S (2015) Provable efficiency of contraction hierarchies with randomized preprocessing. In: Proceedings of the 26th international symposium on algorithms and computation (ISAAC), lecture notes in computer science, vol 9472, pp 479–490. Springer. https://doi.org/10.1007/978-3-662-48971-0_41

- Funke S, Nusser A, Storandt S (2014) On k-path covers and their applications. *Proc VLDB Endow* 7(10):893–902. <https://doi.org/10.14778/2732951.2732963>
- Gavoille C, Peleg D, Pérennes S, Raz R (2004) Distance labeling in graphs. *J. Algorithms* 53(1):85–112. <https://doi.org/10.1016/j.jalgor.2004.05.002>
- Geisberger R, Sanders P, Schultes D, Vetter C (2012) Exact routing in large road networks using contraction hierarchies. *Transport Sci* 46(3):388–404. <https://doi.org/10.1287/trsc.1110.0401>
- Gupta S, Kosowski A, Viennot L (2019) Exploiting hopsets: improved distance oracles for graphs of constant highway dimension and beyond. In: Proceedings of the 46th international colloquium on automata, languages, and programming (ICALP), LIPIcs, vol 132, pp 143:1–143:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik). <https://doi.org/10.4230/LIPIcs.ICALP.2019.143>
- Hausser D, Welzl E (1986) Epsilon-nets and simplex range queries. In: Proceedings of the 2nd annual ACM SIGACT/SIGGRAPH symposium on computational geometry (SCG), pp 61–71. ACM. <https://doi.org/10.1145/10515.10522>
- Kosowski A, Viennot L (2017) Beyond highway dimension: small distance labels using tree skeletons. In: Proceedings of the 28th annual ACM-SIAM symposium on discrete algorithms (SODA), pp 1462–1478. SIAM. <http://dl.acm.org/citation.cfm?id=3039686.3039781>
- Kuhn F, Moscibroda T, Nieberg T, Wattenhofer R (2005) Fast deterministic distributed maximal independent set computation on growth-bounded graphs. In: Proceedings of the 19th international conference on distributed computing (DISC), lecture notes in computer science, vol 3724, pp 273–287. Springer. https://doi.org/10.1007/11561927_21
- Milosavljević N (2012) On optimal preprocessing for contraction hierarchies. In: Proceedings of the 5th ACM SIGSPATIAL international workshop on computational transportation science, pp 33–38. ACM. <https://doi.org/10.1145/2442942.2442949>
- Pugh W (1990) Skip lists: a probabilistic alternative to balanced trees. *Commun ACM* 33(6):668–676
- Robertson N, Seymour PD (1986) Graph minors. v. excluding a planar graph. *J Comb Theory Ser B* 41(1):92–114. [https://doi.org/10.1016/0095-8956\(86\)90030-4](https://doi.org/10.1016/0095-8956(86)90030-4)
- Storandt S (2013) Contraction hierarchies on grid graphs. In: Proceedings of the 36th annual german conference on AI (KI), lecture notes in computer science, vol 8077, pp 236–247. Springer. https://doi.org/10.1007/978-3-642-40942-4_21
- Tao Y, Sheng C, Pei J (2011) On k-skip shortest paths. In: Proceedings of the ACM SIGMOD international conference on management of data, pp 421–432. ACM. <https://doi.org/10.1145/1989323.1989368>
- Vapnik VN, Chervonenkis AY (2015) On the uniform convergence of relative frequencies of events to their probabilities. In: Measures of complexity, pp 11–30. Springer
- White C (2015) Lower bounds in the preprocessing and query phases of routing algorithms. In: Proceedings of the 23rd annual European symposium on algorithms (ESA), lecture notes in computer science, vol 9294, pp 1013–1024. Springer. https://doi.org/10.1007/978-3-662-48350-3_84