ISSN 1346-5597

National Institute of Informatics

NII Technical Report

Approximate Shortest Path Queries in Graphs Using Voronoi Duals

Christian Sommer, Michael E. Houle, Martin Wolff and Shinichi Honiden

NII-2008-007E
Aug. 2008

URL: http://www.nii.ac.jp/ Tokyo 101-8430, JAPAN +81-3-4212-2000 FAX: +81-3-3556-1916

Approximate Shortest Path Queries in Graphs Using Voronoi Duals

Christian Sommer, Michael E. Houle, Martin Wolff, and Shinichi Honiden
{sommer,meh,wolff,honiden}@nii.ac.jp

National Institute of Informatics, Tokyo, Japan

Abstract. We propose an approximation method to answer shortest path queries in graphs, based on hierarchical random sampling and Voronoi duals. The lowest level of the hierarchy stores the initial graph. At each higher level, we compute a simplification of the graph on the level below, by selecting a constant fraction of nodes. Edges are generated as the Voronoi dual within the lower level, using the selected nodes as Voronoi sites. This hierarchy allows for fast computation of approximate shortest paths for general graphs. The time–quality tradeoff decision can be made at query time. We provide bounds on the approximation ratio of the path lengths.

Keywords: shortest path, approximation, graph algorithms, Voronoi

1 Introduction

Classical algorithms, such as Dijkstra's [6], can efficiently answer shortest path queries (for graphs with non-negative edge weights). The implementation using Fibonacci heaps [11] has worst-case running time $O(m + n \log n)$ (where n denotes the number of nodes and m the number of edges). Bi-directional search [20] in practice improves running time. Goldberg [13] provides an expected linear time algorithm if edge weights come from a natural probability distribution. However, for huge graphs even linear time is still too time-consuming.

Among others, large road networks, social networks, and the web graph contain practical problems of challenging size. At query time, only a small portion of the input graph can be considered. If preprocessing is allowed, queries can be answered much faster. Various optimization techniques reduce the graph relevant for a query, and thus the problem size, resulting in substantial speed-ups at query time. The tradeoff between preprocessing time and query time depends on the needs of the application.

1.1 Related work

Hierarchical methods [4, 12, 24, 25] provide an efficient framework, especially in the case of road networks. Sanders and Schultes [21–24] contributes a method to compute shortest paths in 'almost constant time' with a carefully designed structure consisting of precomputed shortest paths [16]. Their solution is tailored to perform exceptionally well for road networks, where graphs are almost planar and nodes have small constant degrees. Precomputation is time- and space-consuming; however, it is still manageable in practice, and allows for extremely fast query times. Holzer et al. [15] evaluates four different heuristics for exact shortest path queries, including their own multi-level approach [25]. The arc flag method [17, 18] divides the graph into regions and marks edges lying on at least one shortest path into a certain region. Although preprocessing is slow, at query time, once the target region is known, only edges leading into that region have to be considered. Combined

with heuristic techniques such as highway hierarchies and shortcuts [4], this proves to be a very efficient method in practice. A^* search [7, 14] is a general search method trying to influence the search towards the target. The recent progress is impressive: for the road networks of Europe or the USA, a speed-up of several orders of magnitudes compared to Dijkstra's algorithm can be achieved with a preprocessing time in the tens of minutes on a high-performance computer [23]. Unfortunately, theoretical bounds on both query time and preprocessing time are hard to obtain. However, even though road networks constitute the most common and popular application to date, other scenarios are possible. Computer networks, social networks, or the web graph exhibit different degree and structural properties, and contain hundreds of millions or even billions of nodes. In specific cases, a user might be willing to trade preprocessing time against exactness both due to the vast size of the data or due to restricted processing power. These senarios allow to make use of a fast approximation method.

Using Voronoi diagrams for shortest paths has been extensively studied; however, to the best of our knowledge this has only been considered for the geodesic in spatial settings and not for general graphs [2].

1.2 Contribution

We propose an approximation method to answer shortest path queries in graphs, based on random sampling and Voronoi duals [9, 19]. In preprocessing, every node is selected as a Voronoi node independently at random with probability p, and the Voronoi dual is computed (sec. 2). This can be applied recursively, leading to a hierarchy with graphs of smaller size on every level (sec. 4). At query time, search for the shortest path from source s to target t can be done faster in the Voronoi dual because it is of smaller size. This shortest path in the Voronoi dual guides the search for an approximate shortest path in the original graph. The expected approximation ratio is at most logarithmic in the number of nodes on the actual shortest path (sec. 3).

Our hierarchy approach is closely related to the hierarchical structure of Schultes and Sanders [24]. However, the computations for every level are rather different: (1) instead of promoting nodes according to their importance (graph centrality), we promote nodes using random sampling, as a consequence (2) we only approximate shortest paths, and (3) we do not focus on almost planar graphs such as road networks. Our method allows for much faster preprocessing, which can be analyzed in terms of complexity.

2 Graph Voronoi Diagram

In this section we present the construction of the graph Voronoi dual, and show how a shortest path in the dual can be used to find an approximation of the shortest path in the original graph. In the hierarchical approximation method described in sec. 4 the substitution of a graph by its Voronoi dual serves as the mechanism by which upper levels of the hierarchy are generated from lower levels. In the following, unless indicated otherwise, we consider only undirected, connected graphs. First, we introduce the notions and terminology needed in this paper.

2.1 Preliminaries

A *graph* $G = (V, E)$ consists of a set of *vertices* V and *edges* $E \subseteq \binom{V}{2}$. A *directed graph* $G = (V, E)$ consists of a set of *vertices* V and directed *edges* $E \subseteq V \times V$. A graph $G' = (V', E')$ is a *subgraph* of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. An *induced subgraph* is a subset of the vertices of a graph G together with any edges whose endpoints are both in this subset. Two nodes $u, v \in V$ are called

adjacent (or *neighbors*) iff they are connected by an edge, i.e., $(u,v) \in E \vee (v,u) \in E$. The *in-neighbors* of v are the members of $N_{\text{in}} = \{s : (s,v) \in E\}$, and the *out-neighbors* of v are the members of $N_{\text{out}}(v) = \{s : (v,s) \in E\}$. The *neighbors* of v, $N(v) = N_{\text{in}}(v) \cup N_{\text{out}}(v)$, are the union of the in-neighbors and out-neighbors of v. The *degree* of a node v, $\deg(v) = |N(v)|$, is the number of its neighbors. The *in-degree* of a node v, $\deg_{\text{in}}(v) := |N_{\text{in}}(v)|$, is the number of its incoming edges, and the *out-degree* of a node v $\deg_{\text{out}}(v) := |N_{\text{out}}(v)|$, is the number of its outgoing edges. A *weighted* graph $G = (V, E, \omega)$ consists of a graph (V, E) together with a *weight function* $\omega : E \to \mathbb{R}$. We assume non-negative edge weights; that is, $\omega : E \to \mathbb{R}^{\geq 0}$.

In a graph $G = (V, E, \omega)$, a (directed) *path* from $s = u_0 \in V$ to $t = u_h \in V$ is a sequence of nodes $(u_0, u_1, \ldots, u_h)$ for which $(u_i, u_{i+1}) \in E$ for all $i \in \{0, 1, \ldots h-1\}$. The *length* of a path P is the sum of its edge weights $|P| := \sum_{i=0}^{h-1} \omega(u_i, u_{i+1})$. The *hop length* of a path P is the number of its edges h. A subpath P' of a path P is a subsequence of its nodes $P' = (u_i, u_{i+1}, \ldots u_j)$, $0 \leq i < j \leq h$. A *simple path* is a path without repeated vertices. Let $\mathcal{P}_G(u,v)$ denote the set of paths from u to v in a graph G. The (directed) *distance* $d(u,v)$ between two nodes u, v is the length of a shortest path from u to v; that is, $d(u,v) = \min_{P \in \mathcal{P}(u,v)} |P|$. If $\mathcal{P}(u,v) = \emptyset$ then $d(u,v) := \infty$. Let $\mathcal{SP}_G(s,t) := \{P \in \mathcal{P} : |P| = d_G(s,t)\}$ denote the set of all shortest paths between s and t and $SP_G(s,t) \in \mathcal{SP}_G(s,t)$ an arbitrary shortest path from s to t. A directed graph is *connected* if for any two vertices u and v there exists an undirected path from u to v, and *strongly connected* if a directed path exists in each direction.

2.2 Graph Voronoi Diagram

The classical Voronoi diagram is a distance-based decomposition of a metric space relative to a discrete set, the Voronoi sites. Given a set of points (the Voronoi sites), the Voronoi decomposition leads to regions (the Voronoi regions) consisting of all points that are closest to a specific site. Mehlhorn [19] and Erwig [9] proposed an analogous decomposition, the *Graph Voronoi Diagram*, for undirected and directed graphs respectively.

Definition 1 (Graph Voronoi Diagram [9, 19]). *In a graph $G = (V, E, \omega)$, the* Voronoi diagram *for a set of nodes $K = \{v_1, \ldots, v_k\} \subseteq V$ is a partition $\mathsf{Vor}_{(G,K)} := \{V_1, \ldots, V_k\}$ of V such that for each node $u \in V_i$, $d(u, v_i) \leq d(u, v_j)$ for all $j \in \{1, \ldots, k\}$.*

The V_i are called *Voronoi regions*. Let $\mathsf{vor}(u)$ denote a node's corresponding Voronoi region i; that is, $\mathsf{vor}(u) = i \Leftrightarrow u \in V_i$.

Analogously to the Delaunay triangulation dual for classical Voronoi diagrams of point sets, we define the Voronoi dual for graphs.

Definition 2. *Let $G = (V, E, \omega_G)$ be a weighted graph and $\mathsf{Vor}_{G,K}$ its Voronoi diagram. To each Voronoi node v_j we associate a corresponding dual node, denoted by v_j^*. The* Voronoi dual *is the graph $G^* = (K^*, E_{G^*}, \omega_{G^*})$ with node set $K^* = \{v_i^* : v_i \in K\}$, edges $E_{G^*} := \{(v_i^*, v_j^*) : v_i^*, v_j^* \in K^* \wedge \exists u \in V_i \wedge \exists w \in V_j : (u,w) \in E\}$, and edge weights $\omega_{G^*}(u^*, v^*) := d_G(u,v)$.*

Situations where a node u has the same distance to more than one Voronoi nodes can be resolved by arbitrarily assigning it to one of them. Erwig [9, Thm. 2] shows that the graph Voronoi diagram can be constructed with a single (parallel) Dijkstra search in time $O(m + n \cdot \log n)$. We slightly modify this construction of the Voronoi *diagram* [9, sec. 3.1] to compute the Voronoi *dual* — that is, to also compute E_{G^*} and ω_{G^*}. Whenever a node u is settled in the Dijkstra search (and thereby assigned to a Voronoi region $V_{\mathsf{vor}(u)}$) for all its settled neighbors u' of different Voronoi regions ($\mathsf{vor}(u) \neq \mathsf{vor}(u')$) we add the edge $(v^*_{\mathsf{vor}(u)}, v^*_{\mathsf{vor}(u')})$ with weight $\omega_{G^*}(v^*_{\mathsf{vor}(u)}, v^*_{\mathsf{vor}(u')}) = d_G(v_{\mathsf{vor}(u)}, u) + \omega_G(u, u') + d_G(u', v_{\mathsf{vor}(u')})$, or decrease its length if there already is an edge in G^*. The final edge

weight $\omega_{G^*}(v^*_{\mathsf{vor}(u)}, v^*_{\mathsf{vor}(u')})$ equals the length of the shortest path that crosses the Voronoi border, which may be larger than the actual distance $d_G(v_{\mathsf{vor}(u)}, v_{\mathsf{vor}(u')})$. This increases the time complexity by at most a constant factor.

Definition 3. *For a path $P = (u_0, u_1, \ldots, u_h)$ in a graph G, the corresponding* Voronoi path *in the Voronoi dual G^* is the path $P^* = (v^*_{\mathsf{vor}(u_0)}, v^*_{\mathsf{vor}(u_1)}, \ldots, v^*_{\mathsf{vor}(u_h)})$.*

Using this definition, multiple consecutive occurrences of nodes $v_{\mathsf{vor}(u_i)^*}$ are possible in P^*. They are treated as a single occurrence, and such paths are equivalent. Note that this path P^* may not necessary be simple.

Lemma 1. *For any path $P = (u_0, \ldots, u_h)$ in an undirected graph $G = (V, E, \omega)$, the corresponding Voronoi path P^* exists and is unique.*

Proof. Suppose that there is no such path P^* in G^*. This implies that there exist pairs of nodes u_k, u_{k+1} on the path P for which $v^*_{\mathsf{vor}(u_k)} \neq v^*_{\mathsf{vor}(u_{k+1})}$ and $(v^*_{\mathsf{vor}(u_k)}, v^*_{\mathsf{vor}(u_{k+1})}) \notin E^*$. As u_k, u_{k+1} are consecutive nodes on the path P, we know that $(u_k, u_{k+1}) \in E$. This contradicts the definition of the Voronoi dual (Def. 2) as $(u_k, u_{k+1}) \in E$ and $v^*_{\mathsf{vor}(u_k)} \neq v^*_{\mathsf{vor}(u_{k+1})}$ implies that $(v^*_{\mathsf{vor}(u_k)}, v^*_{\mathsf{vor}(u_{k+1})}) \in E^*$. This path is unique since nodes u_k on the path belong to exactly one Voronoi region, corresponding to exactly one Voronoi node $v^*_{\mathsf{vor}(u_k)}$. □

Definition 4. *For a path P^* in the Voronoi dual G^* of a graph G, the* Voronoi sleeve *is the subgraph of G induced by the nodes in the union of all Voronoi regions V_i for which v^*_i lies on P^*, i.e., $\mathsf{Sleeve}_{(G,G^*)}(P^*) := G\left[\bigcup_{v^*_i \in P^*} V_i\right]$.*

With the definitions at hand we can now state the approximation method.

2.3 Approximation Algorithm

Given a graph G and its Voronoi dual G^* we answer (approximate) shortest path queries between source s and target t using the following algorithm. The algorithm first searches a shortest path $SP_{G^*}(v^*_{\mathsf{vor}(s)}, v^*_{\mathsf{vor}(t)})$ in the smaller Voronoi dual G^*. This path guides the search for an approximate shortest path in the original graph, as it defines the subgraph $\mathcal{S} = \mathsf{Sleeve}(SP_{G^*}(v^*_{\mathsf{vor}(s)}, v^*_{\mathsf{vor}(t)}))$ in which the Dijkstra search is performed to compute the approximate shortest path $SP_{\mathcal{S}}(s, t)$.

Algorithm 1 – **Construction** *Input: Graph $G = (V, E, \omega)$, Sampling Rate $p \in [0, 1]$ Output: Voronoi dual G^* with Voronoi nodes selected independently at random*

1. *Random sampling: Every node $v \in V$ is selected as Voronoi node independently at random with probability $0 < p < 1$: $\forall v \in V : Pr[v^* \in K] = p$.*
2. *Compute Voronoi dual $\mathsf{Vor}_{G,K} = G^* = (K^*, E_{G^*}, \omega_{G^*})$ using the modified version of [9, sec. 3.1].*
3. *Return G^**

– **Query** *Input: Graph G, Voronoi dual G^*, Source s, Target t, Output: an approximate shortest path P*

1. *Find Voronoi source $v^*_{\mathsf{vor}(s)}$ and Voronoi target $v^*_{\mathsf{vor}(t)}$*
2. *Compute the shortest path from $v^*_{\mathsf{vor}(s)}$ to $v^*_{\mathsf{vor}(t)}$ in the Voronoi dual G^*: $SP_{G^*}(v^*_{\mathsf{vor}(s)}, v^*_{\mathsf{vor}(t)})$*
3. *Compute the Voronoi sleeve $\mathcal{S} := \mathsf{Sleeve}(SP_{G^*}(v^*_{\mathsf{vor}(s)}, v^*_{\mathsf{vor}(t)}))$*
4. *Compute the shortest path from s to t in the Voronoi sleeve $\mathcal{S}$: $SP_{\mathcal{S}}(s, t)$*
5. *Return $P = SP_{\mathcal{S}}(s, t)$*

In the next section, we prove that the expected path length approximation ratio is logarithmic in the number of hops of an exact shortest path.

Theorem 1. *For shortest paths of h hops, Algorithm 1 with sampling rate p has expected approximation ratio $O(\log_{1/(1-p)} h)$.*

3 Proof of Theorem 1

The path $SP_{\mathcal{S}}(s,t)$ found by the algorithm is an approximation because it is possible that no actual shortest path $SP_G(s,t)$ lies entirely within the Voronoi sleeve $\mathcal{S}$. We explain how this is possible and give an upper bound on the expected length $|SP_{\mathcal{S}}(s,t)|$. For this purpose, we prove length relations between simple paths P and their corresponding Voronoi paths P^*. The dilation of a path P^* depends on the number of Voronoi nodes on the path P and their distribution on the path P. In particular, it depends linearly on the largest interval between two Voronoi nodes on the path.

Definition 5. *For a path $P = (u_0, u_1, \ldots, u_h)$ in a graph $G = (V, E, \omega)$, and a set of Voronoi nodes $K \subseteq V$, two Voronoi v_i, v_j nodes on P are called* consecutive *if the subpath between v_i and v_j does not contain another Voronoi node. The* gap *g between two consecutive Voronoi nodes on the path is defined as the hop length of this subpath. The* largest gap *of a path is the maximum over all gaps between two consecutive Voronoi nodes on the path.*

For the analysis, we assume that s and t are Voronoi nodes. We want to prove that the dilation is at most the largest gap $\bar{h}$ between two Voronoi nodes on the path $SP_G(s,t)$. For the analysis we fix a shortest path $SP_G(s,t)$. Let h denote the number of hops, and let $u_1, \ldots, u_{h-1}$) be the intermediate nodes of the shortest path. If the corresponding Voronoi path $(SP_G(s,t))^*$ of this shortest path $SP_G(s,t)$ is a shortest path from s to t in the Voronoi dual, i.e., $(SP_G(s,t))^* \in \mathcal{SP}_{G^*}(s^*, t^*)$, the Voronoi sleeve $\mathcal{S}$ also contains $SP_G(s,t)$, i.e. $SP_G(s,t) \in \mathcal{SP}_{\mathcal{S}}(s,t)$. Figure 1 gives an example under which conditions $(SP_G(s,t))^* \notin \mathcal{SP}_{G^*}(s^*, t^*)$ and Lemma 2 and 3 give bounds on the dilation.

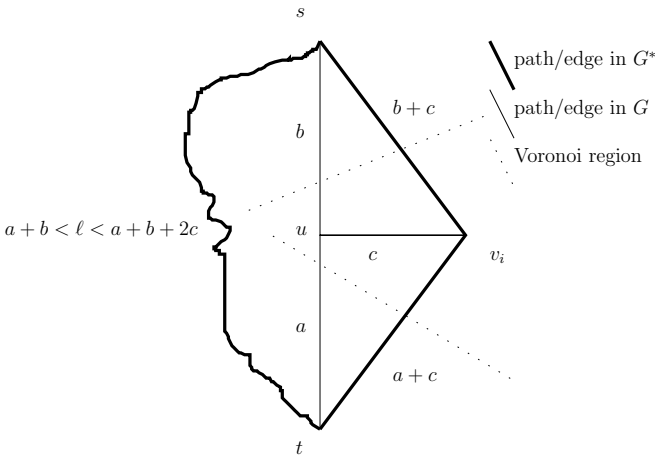

Fig. 1. s, t, and v_i are Voronoi nodes. The shortest path from s to t leads through u, which is in v_i's Voronoi region (if $c < a$ and $c < b$) and paths in the Voronoi dual route through v_i^*. Therefore, the shortest path in the Voronoi dual SP_{G^*} takes another route (left) and the Voronoi sleeve $\mathcal{S}$ does not contain u.

For any simple path P, in Lemma 2 we first give a worst-case bound on the length of the corresponding Voronoi path P^*. The path suffers from maximal dilation if there is no Voronoi node

among the intermediate nodes and the corresponding Voronoi nodes have maximal distance (while still satisfying the Voronoi condition). Lemma 3 gives a better bound for the case where there are other Voronoi nodes on the shortest path. It is a simple composition of Lemma 2. Figure 2 gives an illustration of the scenario.

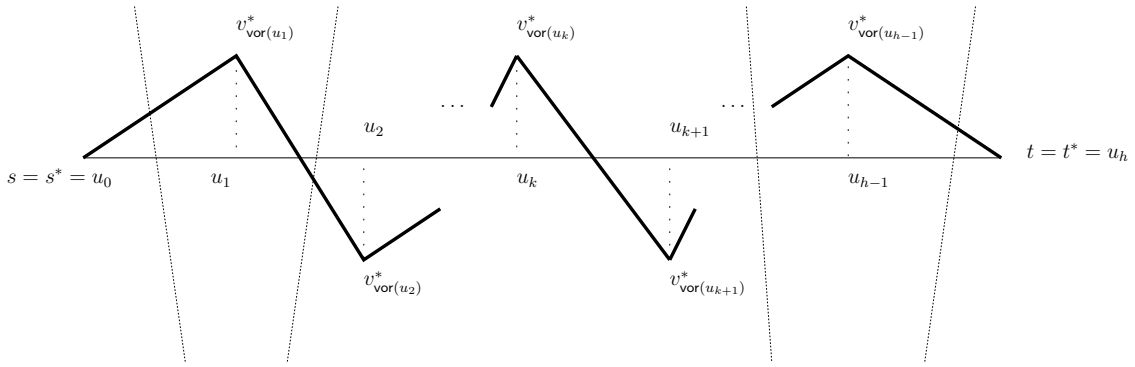

Fig. 2. The shortest path between two Voronoi nodes s and t with $h-1$ intermediate nodes $u_1, \ldots, u_{h-1}$. The distance between two Voronoi nodes that are adjacent in the Voronoi dual is at most $\omega_{G^*}(v^*_{\mathsf{vor}(u_k)}, v^*_{\mathsf{vor}(u_{k+1})}) \leq d_G(v_{\mathsf{vor}(u_k)}, u_k) + \omega_G(u_k, u_{k+1}) + d_G(u_{k+1}, v_{\mathsf{vor}(u_{k+1})})$.

Lemma 2. *Given a simple path $P = (v_i, u_1, \ldots, u_{h-1}, v_j)$ between two Voronoi nodes $v_i = u_0$ and $v_j = u_h$ with h hops and length $|P|$. The corresponding Voronoi path P^* in the Voronoi dual G^* has at most length $|P^*| \leq h \cdot |P|$. This is tight.*

Proof. There are $h-1$ nodes on an h-hop path and, therefore, the path may lead through at most $h+1$ different Voronoi regions, whereof at most $h-1$ regions are 'interfering' regions, meaning that the original shortest path does not lead through the corresponding Voronoi nodes but the Voronoi path dilates to this nodes. The path length $|P|$ in the original graph is the sum of the edge weights $|P| := d_G(s,t) = \sum_{k=0}^{h-1} \omega_G(u_k, u_{k+1})$. The edge between two Voronoi nodes on the path has at most the following length (see also Fig. 2):

$$d_{G^*}(v_{\mathsf{vor}}(u_k), v_{\mathsf{vor}}(u_{k+1})) \leq d_G(v_{\mathsf{vor}}(u_k), u_k) + \omega_G(u_k, u_{k+1}) + d_G(u_{k+1}, v_{\mathsf{vor}}(u_{k+1}))$$

The Voronoi condition holds: intermediate nodes u_k are closest to 'their' corresponding Voronoi node $v_{\mathsf{vor}(u_k)}$, i.e., $\forall j : d_G(u_k, v_{\mathsf{vor}(u_k)}) \leq d_G(u_k, v_{\mathsf{vor}(u_j)})$. Therefore, the corresponding Voronoi nodes also must be closer than the source and target. That is,

$$\begin{aligned} d_G(u_k, v_{\mathsf{vor}(u_k)}) &\leq d_G(s, u_k) \\ d_G(u_k, v_{\mathsf{vor}(u_k)}) = d_G(v_{\mathsf{vor}(u_k)}, u_k) &\leq d_G(u_k, t) \end{aligned}$$

This can be applied in the following:

$$\begin{aligned} |P^*| \leq d_{G^*}(s,t) = d_{G^*}(s, v_{\mathsf{vor}(u_1)}) & \\ + \sum_{k=1}^{h-2} d_G(v_{\mathsf{vor}(u_k)}, u_k) + \omega_G(u_k, u_{k+1}) + d_G(u_{k+1}, v_{\mathsf{vor}(u_{k+1})}) & \\ + d_{G^*}(v_{\mathsf{vor}(u_{h-1})}, t) & \end{aligned}$$

$$
\begin{aligned}
&\leq \omega_G(s,u_1) + d_G(u_1, v_{\mathsf{vor}(u_1)}) \\
&\quad + \sum_{k=1}^{h-2} d_G(v_{\mathsf{vor}(u_k)}, u_k) + d_G(u_{k+1}, v_{\mathsf{vor}(u_{k+1})}) \\
&\quad + \sum_{k=1}^{h-2} \omega_G(u_k, u_{k+1}) \\
&\quad + d_G(v_{\mathsf{vor}(u_{h-1})}, u_{h-1}) + \omega_G(u_{h-1}, t) \\
&= d_G(s,t) \\
&\quad + \sum_{k=1}^{h-1} d_G(v_{\mathsf{vor}(u_k)}, u_k) + d_G(u_k, v_{\mathsf{vor}(u_k)}) \\
&\leq d_G(s,t) + \sum_{k=1}^{h-1} d_G(s,u_k) + d_G(u_k,t) \\
&= h \cdot |P|
\end{aligned}
$$

If all $h-1$ intermediate nodes belong to different interfering Voronoi regions at maximum distance, this bound is tight. The bound is achieved when (for example) $d_G(u_k, v_{\mathsf{vor}(u_k)}) = a - \epsilon$, $\omega(u_k, u_{k+1}) = 2 \cdot \epsilon$, and $\omega(s,u_1) = \omega(u_{h-1}, t) = a$. □

The maximum dilation is guaranteed to be smaller if there is additional Voronoi nodes on the (shortest) path. It is then proportional to the largest gap of the path, which we prove in the following lemma. Lemma 4 gives an upper bound on the expected largest gap.

Lemma 3. *Given a simple path $P = (v_i, u_1, \ldots, u_{h-1}, v_j)$ between two Voronoi nodes $v_i = u_0$ and $v_j = u_h$ with h hops and length $|P|$. Let $\bar{h}$ denote the largest gap of P. The corresponding Voronoi path P^* in the Voronoi dual G^* has at most length $|P^*| \leq \bar{h} \cdot |P|$. This is tight.*

Proof. Suppose there is $2+\nu$ Voronoi nodes on the path, i.e., $u_k = v_{\mathsf{vor}(u_k)}$. The remaining $h-1-\nu$ nodes are non-Voronoi nodes. We cut the path P into subpaths P_k between Voronoi nodes. Let h_k denote the gap between two consecutive Voronoi nodes. The Voronoi path is composed of $1+\nu$ segments P_k of h_k hops between Voronoi nodes ($\sum_{k=0}^{\nu} |P_k| = P$, $\sum_{k=0}^{\nu} h_k = h$, $\forall k : h_k \leq \bar{h}$). Composition of Lemma 2 leads to the following bound on the path length:

$$
\sum_{k=0}^{\nu} h_k |P_k| \leq \sum_{k=0}^{\nu} \max_{\kappa \in \{0,\ldots,\nu\}} h_\kappa |P_k| \leq \bar{h} \cdot |P|.
$$

Tightness holds due to Lemma 2. □

Lemma 4. *In a random bit-string of length $h-1$, where each bit is 1 independently at random with probability p, the longest sequence of 0's is of expected length at most $O(\log_{1/(1-p)} h)$.*

Proof. We give a loose upper bound. A closer bound is the expectation of the maximum of N independent geometric random variables with probability p and sum $h-1-N$ (N itself being a random variable). This problem is known as the Longest Success-Run [8, ch. 8.5]. We neglect the sum condition, which only increases the maximum, and instead of N random variables, we take the maximum value out of $h \geq N$ random variables, which also only increases the maximum. The latter bound is proven by Szpankowski and Rego [26, eq. (2.12)] to be of order $O(\log_{1/(1-p)} h)$. □

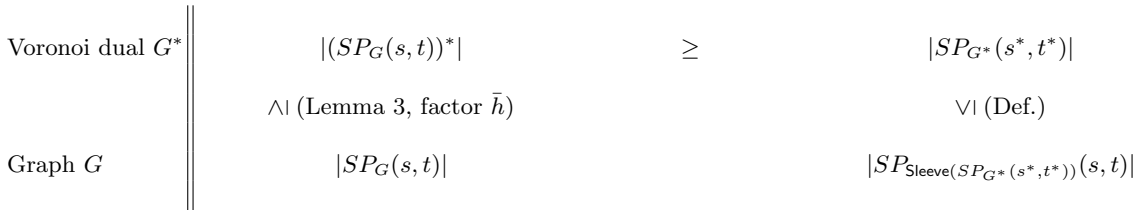

Fig. 3. Outline of the proof of Theorem 1. Lemma 3 shows the relationship between paths in the actual graph and paths in the Voronoi dual. The corresponding path is longer than the shortest path in the dual. Finally, the shortest path of the dual is mapped back to the original graph.

We now combine Lemma 2, 3, and 4.

Let $\bar{h}$ denote the largest gap of the path P. The corresponding Voronoi path has length at most $\bar{h} \cdot |SP_G(s,t)|$ by Lemma 3. Trivially, the shortest path in the Voronoi dual is shorter or of equal length, $|(SP_G(s,t))^*| \geq |SP_{G^*}(s^*,t^*)|$. By definition of the Voronoi dual, distances in the Voronoi dual cannot be smaller than in the original graph. Therefore, there is a path between s and t in the Voronoi sleeve with length at most $\bar{h} \cdot |SP_G(s,t)|$.

For a simple path of h hops, the expected largest gap $\bar{h}$ (nodes are independently selected as Voronoi nodes with sampling rate p) is at most $O(\log_{1/(1-p)} h)$ by Lemma 4 (due to [26]). This concludes the proof. □

4 Hierarchical Composition

Recursively computing the Voronoi dual (sec. 2) leads to a hierarchical search structure. Recursion stops as soon as the resulting graph is sufficiently small — that is, when it contains the cube-root fraction of the original graph. For the remaining graph, a cubic-time all-pair shortest path algorithm [10, 28] can be executed in overall linear time, and its output can be stored in linear space.

At query time, given source s and target t, the algorithm first ascends the hierarchy recursively using the corresponding Voronoi nodes $v^*_{\mathsf{vor}(s)}, v^*_{\mathsf{vor}(t)}$ as new source and target, then computes the shortest path on the highest level. Finally, the algorithm descends the hierarchy refining the path on every level, making use of the approximate shortest path computed relative to the level above. If the graph is 'nice', the graph Voronoi construction generalizes to multiple levels of the hierarchy. In this case, the maximal dilation is bounded by $O(\log^{\log h} h)$. However, due to arbitrary edge lengths, the shortest path on higher levels could contain more nodes and thus more hops instead of the expected $p \cdot h$ hops.

5 Conclusion

The Voronoi diagram is of use for path computations in general (undirected) graphs as well. If Voronoi nodes are chosen at random with constant probability, shortest paths with h hops can be approximated efficiently with expected approximation ratio $O(\log h)$. Recursive construction of the Voronoi dual leads to a hierarchical structure, wherein approximate shortest paths can be computed very efficiently.

References

1. 9th DIMACS Implementation Challenge. Shortest paths. `http://www.dis.uniroma1.it/challenge9/`, 2006.
2. Sang Won Bae and Kyung-Yong Chwa. Shortest paths and Voronoi diagrams with transportation networks under general distances. In *Algorithms and Computation, 16th International Symposium*, pages 1007–1018, 2005.
3. Holger Bast, Stefan Funke, Domagoj Matijevic, Peter Sanders, and Dominik Schultes. In transit to constant time shortest-path queries in road networks. In *Proceedings of the Workshop on Algorithm Engineering and Experiments*, 2007.
4. Reinhard Bauer and Daniel Delling. SHARC: Fast and robust unidirectional routing. In *Proceedings of the 10th Workshop on Algorithm Engineering and Experiments*, pages 13–26, 2008.
5. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 2nd edition, 2001.
6. Edsger W. Dijkstra. A note on two problems in connection with graphs. *Numerische Math.*, 1:269–271, 1959.
7. Jim E. Doran. An approach to automatic problem-solving. *Machine Intelligence*, 1:105–124, 1967.
8. Paul Embrechts, Thomas Mikosch, and Claudia Klüppelberg. *Modelling extremal events: for insurance and finance*. Springer-Verlag, London, UK, 1997.
9. Martin Erwig. The graph Voronoi diagram with applications. *Networks*, 36(3):156–163, 2000.
10. Robert W. Floyd. Algorithm 97: Shortest path. *Commun. ACM*, 5(6):345, 1962.
11. Michael L. Fredman and Robert E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM*, 34(3):596–615, 1987.
12. Robert Geisberger, Peter Sanders, Dominik Schultes, and Daniel Delling. Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In *Experimental Algorithms, 7th International Workshop*, pages 319–333, 2008.
13. Andrew V. Goldberg. A practical shortest path algorithm with linear expected time. *SIAM J. Comput.*, 37(5):1637–1655, 2008.
14. Andrew V. Goldberg and Chris Harrelson. Computing the shortest path: A* search meets graph theory. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 156–165, 2005.
15. Martin Holzer, Frank Schulz, Dorothea Wagner, and Thomas Willhalm. Combining speed-up techniques for shortest-path computations. *ACM Journal of Experimental Algorithms*, 10, 2005.
16. Sebastian Knopp, Peter Sanders, Dominik Schultes, Frank Schulz, and Dorothea Wagner. Computing many-to-many shortest paths using highway hierarchies. In *Proceedings of the Workshop on Algorithm Engineering and Experiments*, 2007.
17. Ulrich Lauther. Slow preprocessing of graphs for extremely fast shortest path calculations. Workshop on Computational Integer Programming, 11 1997.
18. Ulrich Lauther. An extremely fast, exact algorithm for finding shortest paths in static networks with geographical background. In *Geoinformation und Mobilität - von der Forschung zur praktischen Anwendung*, volume 22, pages 219–230, 2004.
19. Kurt Mehlhorn. A faster approximation algorithm for the Steiner problem in graphs. *Inf. Process. Lett.*, 27(3):125–128, 1988.
20. Ira S. Pohl. Bi-directional search. *Machine Intelligence*, 6:127–140, 1971.
21. Peter Sanders and Dominik Schultes. Highway hierarchies hasten exact shortest path queries. In *Algorithms - ESA 2005, 13th Annual European Symposium*, pages 568–579, 2005.
22. Peter Sanders and Dominik Schultes. Engineering highway hierarchies. In *Algorithms - ESA 2006, 14th Annual European Symposium*, pages 804–816, 2006.
23. Peter Sanders and Dominik Schultes. Engineering fast route planning algorithms. In *Experimental Algorithms, 6th International Workshop*, pages 23–36, 2007.
24. Dominik Schultes and Peter Sanders. Dynamic highway-node routing. In *Experimental Algorithms, 6th International Workshop*, pages 66–79, 2007.
25. Frank Schulz, Dorothea Wagner, and Christos D. Zaroliagis. Using multi-level graphs for timetable information in railway systems. In *Algorithm Engineering and Experiments, 4th International Workshop*, pages 43–59, 2002.
26. Wojciech Szpankowski and Vernon Rego. Yet another application of a binomial recurrence. order statistics. *Computing*, 43(4):401–410, 1990.
27. U.S. Census Bureau, Washington, DC. UA Census 2000 TIGER/Line Files. `http://www.census.gov/geo/www/tiger/tigerua/uatgr2k.html`, 2002.
28. Stephen Warshall. A theorem on boolean matrices. *J. ACM*, 9(1):11–12, 1962.