Preprint

A Range Space with Constant VC Dimension for All-pairs Shortest Paths in Graphs

Alane M. de Lima [1,‡], André L. Vignatti [1,‡] and Murilo V. G. da Silva [1,‡]

[1] Federal University of Paraná, Brazil; amlima@inf.ufpr.br, vignatti@inf.ufpr.br, murilo@inf.ufpr.br
‡ These authors contributed equally to this work.

Abstract: Let G be an undirected graph with non-negative edge weights and let S be a subset of its shortest paths such that, for every pair (u, v) of distinct vertices, S contains exactly one shortest path between u and v. In this paper we define a range space associated with S and prove that its VC dimension is 2. As a consequence, we show a bound for the number of shortest paths trees required to be sampled in order to solve a relaxed version of the All-pairs Shortest Paths problem (APSP) in G. In this version of the problem we are interested in computing all shortest paths with "centrality" at least ε, where this centrality measure is a certain generalization of the betweenness centrality. Given any $0 < \varepsilon, \delta < 1$, we propose a sampling algorithm that outputs with probability $1 - \delta$ the (exact) distance and the shortest path between every pair of vertices (u, v) that has centrality at least ε. The bound that we obtain for the sample size depends only on ε and δ, and do not depend on the size of the graph.

Keywords: All-pairs Shortest Paths; Sample Complexity; Sampling Algorithm

check for updates

Citation: Lima, A. M. de; Vignatti, A.L.; da Silva, M. V. G. A Range Space with Constant VC Dimension for All-pairs Shortest Paths in Graphs. *Preprints* **2022**, *1*, 0. https://doi.org/

1. Introduction

The All-pairs Shortest Path (APSP) is the problem of computing a path with the minimum length between every pair of vertices in a weighted graph. The APSP problem is very well studied and there has been recent results for a variety of assumptions for the input graph (directed/undirected, integer/real edge weights, etc) [1–4]. In this paper we assume that the input is an undirected graph G with n vertices and m edges with non-negative weights.

In our scenario, the fastest known exact algorithms are the algorithm proposed by Williams (2014) [1], which runs in $\mathcal{O}\left(\frac{n^3}{2^c\sqrt{\log n}}\right)$ time, for some constant $c > 0$, and by Pettie and Ramachandram (2002) [5] for the case of sparse graphs, which runs in $\mathcal{O}(nm \log \alpha(m, n))$ time, where $\alpha(m, n)$ is the Tarjan's inverse-Ackermann function. If no assumption is taken about the sparsity of the graph, then it is an open question whether the APSP problem can be solved in *strictly* subcubic time, i.e. $\mathcal{O}(n^{3-c})$, for any $c > 0$, even when the edge weights are natural numbers.

Recent results in fine-grained complexity indicate that the complexity time for the APSP is tight [6–8], reinforcing the hypothesis that there is no strictly subcubic algorithm for such task [9]. Since the exact computation of this version is expensive for large graphs, especially the dense ones, it is natural dealing with alternative versions of the problem, whether they are approximate [10,11] or applied to restricted scenarios [12]. In this paper, we follow this line of work, dealing with a relaxation of the problem in the sense that the classical APSP is a special case for a given adjustable parameter. More specifically, we aim to compute, with high probability, all the shortest paths that meet a certain "centrality" requirement. The idea is that the centrality of a shortest path P is higher when a large number of shortest paths has P as a subpath. The precise definition of this centrality measure is given in Section 2.

In this relaxed version of the APSP, given constant parameters $0 < \varepsilon, \delta < 1$, we propose a sampling algorithm that outputs, with probability at least $1 - \delta$, the (exact) distance and a shortest path between every pair of vertices that admits a shortest path with centrality at least ε. The central idea of the algorithm is to sample roots of shortest paths trees. In order to give a bound for the sample size that is sufficient to meet the input parameters, we use sample complexity tools, namely, Vapnik–Chervonenkis (VC) dimension theory and the ε-net theorem. We define a range space associated with a set of *canonical shortest paths* in G between every pair of distinct vertices. One of the main results that we prove is that the VC dimension of such range space is 2 and that the bound for the sample size is $r = \lceil \frac{c}{\varepsilon}\left(2\ln(\frac{1}{\varepsilon}) + \ln\frac{1}{\delta}\right)\rceil$, where c is a constant around $\frac{1}{2}$ [13]. This result is interesting, since it does not depend neither on the size of the input n, which is the case if one uses standard union-bound techniques, nor on the topological structure of the graph that may vary with n in many cases. As a consequence of this bound for the sample size, we obtain a sampling algorithm for our problem with running time $\mathcal{O}(m + n\log n + (\text{Diam}_V(G))^2)$, where $\text{Diam}_V(G)$ is the vertex-diameter of the input graph (i.e. the maximum number of vertices in a shortest path in G), for any constant ε.

If one sets ε as a function of n, in the limit case, when $\varepsilon(n) = \frac{1}{n(n-1)}$, our algorithm solves – with high probability – the classical APSP problem, but with time complexity exceeding the running time of the exact algorithms from the literature [5,14]. However, it is still an interesting problem to know for which functions $\varepsilon(n)$ we still have a strictly subcubic sampling algorithm. We show that our algorithm runs in $\mathcal{O}(n^{3-c})$ time if $\varepsilon(n)$ is any $\Omega\left(\frac{W_0(n')}{n'}\right)$ function, where $n' = n^{1-c}$ (for a constant $c > 0$) and $W_0(n')$ is the branch 0 of the Lambert-W function defined for $n' \geq 0$, a non-algebraic value such that $W_0(n') = \ln n' - \ln\ln n' + \Theta\left(\frac{\ln\ln n'}{\ln n'}\right)$, which holds for $n' \geq e$.

2. Shortest Paths, Canonical Paths, and Shortest Paths Trees

Let $G = (V, E)$ be an undirected graph, with $n = |V|$ and $m = |E|$, and let ω be a function of edge *weights* from E to an enumerable subset of $\mathbb{R}_{\geq 0}$. W.l.o.g., we assume that G is connected, since our results can be applied to the connected components when a graph is disconnected. Even though G is undirected, for convenience we use the notation (u, v) for an edge of G. A *path* is a sequence of vertices $P = (v_1, v_2, \ldots, v_k)$ such that $v_i \neq v_{i+1}$ and $(v_i, v_{i+1}) \in E$, for $1 \leq i < k$. If $u = v_1$ and $v = v_k$, such path is referred to as a (u, v)-*path*. We define E_P as the set of edges of P. The *shortest path* from u to v in G is the (u, v)-path such that the sum of the weights of the edges in E_P is minimized. In this case we denote such value $d(u, v)$, also called the *distance* from u to v.

The set of all shortest paths from u to v in G is denoted $\mathcal{C}_{uv}$. For a given path $P \in \mathcal{C}_{uv}$, let $\text{Inn}(P)$ be the set of *inner* vertices of P, that is, $\text{Inn}(P) = \{w \in P : w \notin \{u, v\}\}$. Consider a shortest (u, v)-path P, and let u' and v' be two vertices of P, with u' closer to u and v' closer to v. The subpath of P starting in u' and ending in v' is called a (u', v')-*subpath of* P. The (immediate) *predecessor* of v in a shortest (u, v)-path P, denoted $\text{pred}_P(v)$, is the vertex $w \in \text{Inn}(P)$ such that $(w, v) \in E_P$. The *diameter* of G, denoted Diam_G, is the size of the largest shortest path in G. The *vertex-diameter*, denoted $\text{Diam}_V(G)$, is the maximum number of vertices in a shortest path of G.

Let $\sigma : V \to \{1, \ldots, n\}$ be an arbitrary vertex ordering of G. Consider the set of shortest paths $\mathcal{L}_{uv} = \{P \in \mathcal{C}_{uv} : \sigma(\text{pred}_P(v)) \text{ is minimum}\}$. Note that there is only one vertex w that satisfies the property "$\sigma(\text{pred}_P(v))$ is minimum", so even if there are several paths in $\mathcal{L}_{uv}$, the last edge (w, v) is the same for all of them. Next, we introduce the definition of a *canonical path* with respect to σ.

Definition 1 (Canonical path (CP))**.** *Consider a pair of vertices $(u, v) \in V^2$ in G. The* canonical path *(CP) from u to v, denoted P, is recursively defined as the shortest path in $\mathcal{C}_{uv}$ such that*

case 1: $|\mathcal{L}_{uv}| = 1$**.** *Then $P \in \mathcal{L}_{uv}$ is the canonical path from u to v.*

case 2: $|\mathcal{L}_{uv}| > 1$**.** *Let w be the (unique) predecessor of v in the shortest paths of $\mathcal{L}_{uv}$. Then, the canonical path from u to v corresponds to the canonical path from u to w plus the edge (w, v).*

Fact 1. *Given a pair of vertices $(u, v) \in V^2$, the CP from u to v exists and it is unique.*

To see that Fact 1 holds, note that at each recursive step, there is only one vertex w satisfying the property that defines $\mathcal{L}_{uv}$, and there is only one canonical path from u to w. Besides, the recursion presented above always stop in the base case, since the distance between a pair of vertices in a recursive step is smaller than the distance of a pair of vertices analyzed in the previous step. The base is the one where there is only one (u, u')-subpath which is the shortest path from u to u', for $u' \in \mathrm{Inn}(P)$. Another important observation about canonical paths is that the canonical path from u to v is not necessarily the same as the canonical path from v to u.

A *shortest paths tree (SPT)* of a vertex u is a spanning tree of G such that the path from u to every other vertex of this tree is a shortest path in G. There might be many SPTs for a given vertex. In this paper we are interested in fixing one canonical SPT T_u, for every vertex u of G. More precisely, for a given (arbitrary) vertex ordering σ, the canonical SPT T_u is defined such that, for every vertex v, the shortest path from u to v in T_u is a canonical path. In Section 4.1 we give more details on the computation of T_u, but, briefly speaking, this tree is the one computed by a modification on Dijkstra's algorithm where σ is used as a tie-breaking criterion. We also call T_u the *Dijkstra tree* of u.

A shortest path that starts at the root of a Dijkstra tree is also called a *branch* of G. More formally, given T_u, for every $v \neq u$, the shortest path from u to v is a branch, denoted $\mathcal{B}_{uv}$. In addition, every subpath of $\mathcal{B}_{uv}$ is also a shortest path in G, and we denote such set of subpaths (including $\mathcal{B}_{uv}$) as $S(\mathcal{B}_{uv})$.

We introduce the *shortest path centrality* of a pair of vertices (u, v). The idea, intuitively, is that a shortest path is "central" if several other shortest paths pass through it. This is a similar idea that is used in the well-known betweenness metric for a vertex [15], where a vertex has high betweenness if many shortest paths pass through it.

In order to formally define the shortest path centrality we first need the following. Let t_{uv} be the number of canonical paths that contain a shortest path from u to v as subpath, defined as

$$t_{uv} = \sum_{(a,b) \in V^2 : a \neq b} \mathbb{1}_{uv}(\mathcal{B}_{ab}),$$

where $\mathbb{1}_{uv}(\mathcal{B}_{ab})$ is the indicator function that returns 1 if there is some shortest path from u to v as subpath of the branch $\mathcal{B}_{ab}$ (and 0 otherwise).

Definition 2 (Shortest Path Centrality)**.** *Given a pair $(u, v) \in V^2$, the* shortest path centrality *of (u, v) is defined as*

$$c(u, v) = \frac{t_{uv}}{n(n-1)}, \quad \textit{where } n = |V|.$$

2.1. Key Results on Canonical Paths

Before we present the main results of this paper in Section 3.1, we need first a key technical result concerning canonical paths. We show in Theorem 1 that any subpath of a canonical path is also a canonical path.

Lemma 1. *Given a pair of vertices $(u, v) \in V^2$, let P be the CP from u to v in G. If $|\mathcal{L}_{uv}| = 1$, then every subpath of P is also a CP.*

Proof. Let P' be a (u', v')-subpath of P. Suppose by contradiction that P' is not a CP. Let $Q' \neq P'$ be the shortest path $Q' = (u', \ldots, v')$ in G which the CP from u' to v'.

Case 1: $v' \neq v$. Let S_1 be a (u, u')-subpath and S_2 be a (v', v)-subpath, both from P. Let Q be the concatenation of S_1, Q', and S_2. Note that P' and Q' have the same length (since both are shortest paths), and so does P and Q. Since P and Q have the same vertices from v' to v, then the predecessor of v in both paths is the same. Hence, P and Q are in $\mathcal{L}_{uv}$. But then $|\mathcal{L}_{uv}| > 1$, a contradiction.

Case 2: $v' = v$. Let w and w' be the predecessors of v in P' and Q', respectively. Note that $w \neq w'$. Thus, since $\{w', v\}$ is the last edge of Q', by the definition of CP, $\sigma(w') < \sigma(w)$.

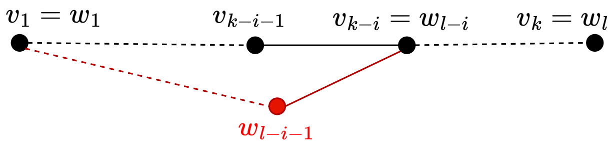

Figure 1. Illustration of vertex v_{k-i} in the shortest paths P (depicted in black color) and Z (depicted in red color).

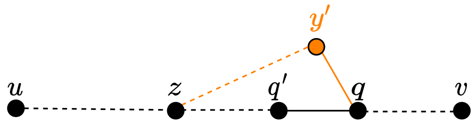

Figure 2. Illustration of the shortest path from z to q (in orange), denoted Q', in the proof of Lemma 4.

But then in the edge (w, v) of P, vertex w does not have the minimum index among all possible predecessors of v, contradicting the fact that P is a CP. $\square$

Lemma 2. *Given a pair of vertices $(u, v) \in V^2$, let P be the CP from u to v in G. Let w be the predecessor of v in P. Then the (u, w)-subpath of P is the CP from u to w.*

Proof. Let P' be the (u, w)-subpath of P. In the case of $|\mathcal{L}_{uv}| = 1$, then from Lemma 1 we have that P' is the CP from u to w. Otherwise, by Definition 1 (case 2) applied to P, it must hold that P' is the CP from u to w. $\square$

Lemma 3. *Given a pair of vertices $(u, v) \in V^2$, let P be the CP from u to v in G. Then for each $z \in Inn(P)$, the (u, z)-subpath of P is the CP from u to z.*

Proof. Let P' be the (u, z)-subpath of P and w be the predecessor of v in P. We prove our claim by induction on the number of edges from z to v. The base case is the one where $z = w$ (i.e. P' is the (u, w)-subpath of P). This holds from Lemma 2.

Let z' be the predecessor of z in P and let P'' be the (u, z')-subpath of P. For the induction step, we show that if P' is CP from u to z, then P'' is the CP from u to z'.

By Definition 1 applied to P', there are two cases to consider: $|\mathcal{L}_{uz}| = 1$ (case 1) and $|\mathcal{L}_{uz}| > 1$ (case 2). In case 1, by Lemma 1 applied to P', the shortest path P'' must be the CP from u to z'. In case 2, by Definition 1 (case 2) applied to P', the CP from u to z' is P''. $\square$

Lemma 4. *Given a pair of vertices $(u, v) \in V^2$, let P be the CP from u to v in G. Then for each $z \in Inn(P)$, the (z, v)-subpath of P is the CP from z to v.*

Proof. Let Q be the (z, v)-subpath of P. We prove by contradiction supposing that Q is not the CP from z to v in G. Then there is a shortest path Y which is the CP from z to v in G. Consider the subpath of P from u to z concatenated with Y, and denote such concatenation as Z. Note that, even though the number of vertices of Q and Y may be different, the length of Q and Y is the same, since both are shortest paths. The same applies to P and Z.

Denote the vertices in P and Z as $P = (u = v_1, \ldots, v = v_k)$ and $Z = (u = w_1, \ldots, v = w_l)$. Let v_{k-i} be the vertex of P such that i is maximum, $0 \le i < k$, and such that the following holds: for all $1 \le j \le i$, the vertex v_{k-j} in P is the same as the vertex w_{l-j} in Z (Figure 1). For simplicity, denote v_{k-i} as q, v_{k-i-1} as q', and w_{l-i-1} as y'. Note that the edges in the (q, v)-subpaths of P and Z are the same, but (q', q) and (y', q) is not the same edge.

Let Q' and Y' be the (z, q)-subpaths of P and Y, respectively (Figure 2). Since we are assuming that Y is the CP from z to v in G, then by Lemma 3, Y' is the CP from z to q in G. Note that $Q' \neq Y'$ (since $Q \neq Y$), and hence, Q' is not the CP from z to q in G. Thus, $\sigma(q') > \sigma(y')$.

From Lemma 3 applied to P, the (u,q)-subpath of P is a CP. But this path is a shortest path such that q' is not the vertex with minimum index among all possible predecessors of q (recall that $\sigma(q') > \sigma(y')$), a contradiction. □

Theorem 1. *Given a pair of vertices $(u,v) \in V^2$, let P the CP from u to v in G. Then for each $(u',v') \in V^2$, the (u',v')-subpath of P is the CP from u' to v' in G.*

Proof. Let P' be the (u',v')-subpath of P. From Lemma 4, the (u',v)-subpath of P, denoted Q, is a CP. From Lemma 3, since Q is the CP from u' to v in G, then P' is the CP from u' to v' in G. □

3. Sample Complexity and VC Dimension

In sampling algorithms, typically the aim is the estimation of a certain quantity according to given parameters of quality and confidence using a random sample of size as small as possible. A central concept in sample complexity theory is the Vapnik–Chervonenkis Theory (VC dimension), in particular, the idea of finding an upper bound for the VC dimension of a class of binary functions related to the sampling problem at hand. In our context, for instance, we may consider a binary function that takes a branch and outputs 1 if such branch contains a shortest path for a given set. Generally speaking, from the upper bound for the VC dimension of the given class of binary functions we can derive an upper bound to the sample size for the sampling algorithm.

We present in this section the main definitions and results from sample complexity theory used in this paper. An in-depth exposition of the VC dimension theory and the ε-net theorem can be found in the books of Shalev-Schwartz and Ben-David (2014) [16], Mitzenmacher and Upfal (2017) [17], Anthony and Bartlett (2009) [18], and Mohri et al. (2012) [19].

Definition 3 (Range Space)**.** *A* range space *is a pair $\mathcal{R} = (U, \mathcal{I})$, where U is a domain (finite or infinite) and $\mathcal{I}$ is a collection of subsets of U, called* ranges*.*

For a given $S \subseteq U$, the *projection* of $\mathcal{I}$ on S is the set $\mathcal{I}_S = \{S \cap I : I \in \mathcal{I}\}$. If $|\mathcal{I}_S| = 2^{|S|}$ then we say S is *shattered* by $\mathcal{I}$. The VC dimension of a range space is the size of the largest subset S that can be shattered by $\mathcal{I}$, i.e.

Definition 4 (VC dimension)**.** *The VC dimension of a range space $\mathcal{R} = (U, \mathcal{I})$, denoted $VCDim(\mathcal{R})$, is*

$$VCDim(\mathcal{R}) = \max\{k : \exists S \subseteq U \text{ such that } |S| = k \text{ and } |\mathcal{I}_S| = 2^k\}.$$

The following combinatorial object, called ε-net, is useful when one wants to find a sample $S \subseteq U$ that intersects every range in $\mathcal{I}$ of a sufficient size.

Definition 5 (ε-net)**.** *Let $\mathcal{R} = (U, \mathcal{I})$ be a range space and π be a probability distribution on U. Given $0 < \varepsilon < 1$, a set S is called ε-net w.r.t. $\mathcal{R}$ if*

$$\forall I \in \mathcal{I},\ \Pr_{\pi}(I) \geq \varepsilon \quad \Rightarrow \quad |I \cap S| \geq 1.$$

When computing ε-nets for a given range space $\mathcal{R} = (U, \mathcal{I})$, we typically build a sample S from elements of U. One can obtain lower bounds for the size of S via standard union bound. However, these bounds usually overestimate $|S|$ since they only take into account the number of points in U or the number of ranges in $\mathcal{R}$. This issue can be overcame if the VC dimension of the range space that models the problem at hand, denoted k, is finite. The next theorem, proven by Har–Peled and Sharir (2011) [20], states a lower bound for $|S|$ based on k.

Theorem 2 (see [20], Theorem 2.12)**.** *Given $0 < \varepsilon, \delta < 1$, let $\mathcal{R} = (U, \mathcal{I})$ be a range space with $VCDim(\mathcal{R}) \leq k$, let π be a probability distribution on the domain U, and let c be a universal positive constant.*

A collection of elements $S \subseteq U$ sampled w.r.t. π with $|S| = \frac{c}{\varepsilon}\left(k \ln \frac{1}{\varepsilon} + \ln \frac{1}{\delta}\right)$ is an ε-net with probability at least $1 - \delta$.

As pointed by Löffler and Phillips (2009) [13], c is around $\frac{1}{2}$, but in this paper we leave c as an unspecified constant.

Some of the techniques used in our sampling strategy described in Sections 3.1 and 4 were developed by Riondato and Kornaropoulos (2016) and Riondato and Upfal (2018) [21, 22], where the authors used VC dimension theory, the ε-sample theorem, and Rademacher averages for the estimation of betweenness centrality in a graph. The work of Lima et al. [23,24] showed how to use sample complexity tools for the estimation of the percolation centrality, which is a generalization of the betweenness centrality. More recently, Cousins et al. (2021) [25] showed improved bounds for the betweenness centrality approximation using Monte–Carlo empirical Rademacher averages, and Lima et al. (2022) [26] used sample complexity tools in the design of a sampling algorithm for the local clustering coefficient of every vertex of a graph.

3.1. Range Space and VC Dimension Results

In this section, we first define the problem in terms of a range space, and then we show that the VC dimension of the range space that models the problem is constant, which directly impacts in the size of the sample to be used by our algorithms. In fact, we show that this sample size only depends on the parameters of quality and confidence, ε and δ, respectively.

Let $n = |V|$ and $\mathcal{T}$ be the set of n Dijkstra trees of G. Recall that such trees are, by definition, composed by canonical paths. The universe U is defined for the set of all branches of Dijkstra trees, i.e.

$$U = \bigcup_{(a,b) \in V^2 : b \neq a} \mathcal{B}_{ab}.$$

For each pair $(u, v) \in V^2$, let p_{uv} be the canonical path from u to v, according to Definition 1. Each range τ_{uv} is defined as $\tau_{uv} = \{\mathcal{B}_{ab} \in U : p_{uv} \in S(\mathcal{B}_{ab})\}$. In other words, we can say that $\mathcal{B}_{ab}$ is in the range of (u, v) if $\mathcal{B}_{ab}$ "passes" through a canonical path between u and v. Let $\mathcal{I} = \{\tau_{uv} : (u, v) \in V^2\}$ be the rangeset. So, $\mathcal{R} = (U, \mathcal{I})$ is the range space defined for our problem.

Now we show how to plug our range space $\mathcal{R}$ with Definition 5 so we can use Theorem 2 to bound the sample size that is tight enough for the task that we are tackling. We first show in Theorem 3 that $c(u, v) = \Pr_\pi(\tau_{uv})$. For this result, we have that each tree $T_a \in \mathcal{T}$ is sampled with probability $\pi(T_a) = \frac{1}{n}$ and each branch $\mathcal{B}_{ab} \in T_a$ is sampled with probability $\frac{1}{n-1}$, leading to the probability distribution $\pi(\mathcal{B}_{ab}) = \frac{1}{n(n-1)}$ (which is a proper distribution as the sum is equal to 1). Let $\mathbb{1}_{uv}(\mathcal{B}_{ab})$ be the indicator function that returns 1 if there is some canonical path from u to v as subpath of $\mathcal{B}_{ab}$, i.e. $\mathcal{B}_{ab} \in \tau_{uv}$, and 0 otherwise.

Theorem 3. *For $(u, v) \in V^2$, $\Pr_\pi(\tau_{uv}) = c(u, v)$.*

Proof. For fixed $(u,v) \in V^2$ and considering that a branch $\mathcal{B}_{ab} \in U$ is sampled with probability $\pi(\mathcal{B}_{ab}) = \frac{1}{n(n-1)}$, we have

$$\begin{aligned}
\Pr_{\pi}(\tau_{uv}) &= \sum_{T_a \in \mathcal{T}} \sum_{\mathcal{B}_{ab} \in T_a} \pi(\mathcal{B}_{ab}) \mathbb{1}_{uv}(\mathcal{B}_{ab}) \\
&= \frac{1}{n(n-1)} \sum_{T_a \in \mathcal{T}} \sum_{\mathcal{B}_{ab} \in T_a} \mathbb{1}_{uv}(\mathcal{B}_{ab}) \\
&= \frac{1}{n(n-1)} \sum_{a \in V} \sum_{b \in V : b \neq a} \mathbb{1}_{uv}(\mathcal{B}_{ab}) \\
&= \frac{t_{uv}}{n(n-1)} = c(u,v).
\end{aligned}$$

The first equality follows from the fact that the probability that a branch lies on the range τ_{uv} is equal to counting the individual probabilities of each branch that is in τ_{uv}. □

For problems involving shortest paths, such as the ones in [21,24], it is possible to find a bound for the sample size using VC dimension theory. The referred work typically apply the same proof structure, having a bound based on the vertex-diameter of a graph G, denoted $\mathrm{Diam}_V(G)$, as in Theorem 4 (we present such proof for the sake of completeness). Even though $\mathrm{Diam}_V(G)$ might be as large as n, in particular, this bound is exponentially smaller for graphs with logarithmic vertex-diameter, which may be common in practice.

Although the bound presented in Theorem 4 depends on a combinatorial structure of G, in this work we present an improvement to this result in Theorems 5 and 6, giving a bound that depends only on the desired quality and confidence parameters of the solution. More specifically, for these two theorems we have that $\mathrm{VCDim}(G) = 2$ for a given graph G with respect to a fixed vertex ordering σ, where $\mathrm{VCDim}(G)$ denotes the VC dimension of the range space $\mathcal{R} = (U, \mathcal{I})$ related to a graph G.

Theorem 4. *For a given graph $G = (V,E)$,*

$$VCDim(G) \leq \lfloor 2 \lg Diam_V(G) + 1 \rfloor.$$

Proof. Let $\mathrm{VCDim}(G) = k$, where $k \in \mathbb{N}$. Then, there is $S \subseteq U$ such that $|S| = k$ and S is shattered by $\mathcal{I}$. Each $\mathcal{B}_{ab} \in S$ must appear in 2^{k-1} different ranges in $\mathcal{I}$, from the definition of shattering. On the other hand, $\mathcal{B}_{ab}$ has length at most $\mathrm{Diam}_V(G)$. Then the maximum number of subpaths of $\mathcal{B}_{ab}$, denoted $|S(\mathcal{B}_{ab})|$, is $\mathrm{Diam}_V(G) \cdot (\mathrm{Diam}_V(G) - 1)$. Thus, the branch $\mathcal{B}_{ab}$ lies in at most $|S(\mathcal{B}_{ab})|$ ranges, and therefore,

$$2^{k-1} \leq |S(\mathcal{B}_{ab})| \leq \mathrm{Diam}_V(G) \cdot (\mathrm{Diam}_V(G) - 1) \leq \mathrm{Diam}_V(G)^2.$$

Solving for k, $\mathrm{VCDim}(G) = k \leq \lfloor 2 \lg \mathrm{Diam}_V(G) + 1 \rfloor$. □

For Theorems 5 and 6, we introduce the definition of *meeting path* between two canonical paths P_1 and P_2, and in Lemma 5 we prove that there is only one such path between P_1 and P_2. We use this fact to prove that $\mathrm{VCDim}(G) \leq 2$ in Theorem 5.

Definition 6. *Consider two different canonical paths P_1 and P_2. We say that a canonical path $Z = (z, \ldots, z')$ is a* meeting path *between P_1 and P_2 if Z is a maximal (z,z')-subpath of P_1 and P_2.*

Lemma 5. *Consider two different canonical paths P_1 and P_2. Let Z be a meeting path between P_1 and P_2. Then Z is the only meeting path between both paths in G.*

Proof. Let $P_1 = (x, \ldots, x')$, $P_2 = (y, \ldots, y')$, and $Z = (z, \ldots, z')$. Suppose that Z is a meeting path between P_1 and P_2 and suppose that it is not unique. Let $W = (w, \ldots, w')$ be

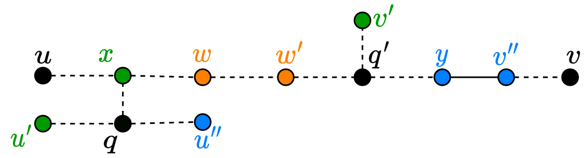

Figure 3. Case where $\tau_{xw} \cap S = \{P_1, P_2, P_3\}$, for $P_1 = (u, \ldots, v)$, $P_2 = (u', \ldots, v')$, and $P_3 = (u'', \ldots, v'')$.

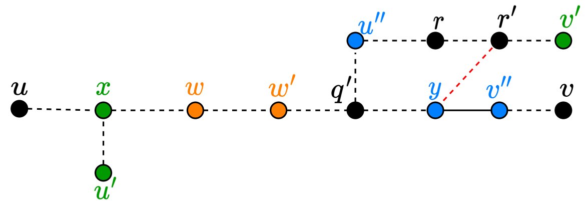

Figure 4. Case where $\tau_{q'y} \cap S = \{P_1\}$, for $P_1 = (u, \ldots, v)$, $P_2 = (u', \ldots, v')$, and $P_3 = (u'', \ldots, v'')$. The red dashed path correspond to a shortest path that cannot happen.

another meeting path in G. Note that Z and W are disjoint, otherwise the concatenation of both paths would contradict the maximality of Z and W. Without loss of generality, we may assume the following:

- Z is contained in the (x, z')-subpath of P_1 and in the (y, z')-subpath of P_2, with z' closer to x and to y in P_1 and P_2, respectively;
- W is contained in the (w, x')-subpath of P_1 and in the (w, y')-subpath of P_2, with w closer to x' and to y' in P_1 and P_2, respectively.

Let D be the CP from z' to w in G. Since P_1 and P_2 are canonical paths, by Theorem 1, the (z', w)-subpath of P_1 and the (z', w)-subpath of P_2 must be equal do D. Let Z' be the concatenation of Z, D, and W. Then Z' is a meeting path between P_1 and P_2 that contradicts the maximality of Z. □

Theorem 5. *For a given graph $G = (V, E)$ and a fixed ordering σ over V,*

$$VCDim(G) \leq 2.$$

Proof. Suppose that $\text{VCDim}(G) > 2$. Then there is a set of canonical paths $\mathcal{S} = \{P_1, P_2, P_3\}$ that is shattered by $\mathcal{I}$. These paths are described as $P_1 = \{u, \ldots, v\}$, $P_2 = \{u', \ldots, v'\}$, and $P_3 = \{u'', \ldots, v''\}$. Let W be the (w, w')-subpath of P_1 that is also contained in P_2 and P_3. From the definition of shattering, this path must exist so that $\tau_{ww'} \cap S = \{P_1, P_2, P_3\}$. Let x be the farthest predecessor of w in P_1 such that, w.l.o.g., the (x, w)-subpath of P_1, denoted X, is also contained in P_2 (but not in P_3). Let y be the farthest successor of P_1 such that a (q', y)-subpath of P_1, denoted Y, is also contained in P_3 (but not in P_2). Note that X and Y must exist so that $\tau_{xw} \cap S = \{P_1, P_2\}$ and $\tau_{q'y} \cap S = \{P_1, P_3\}$.

Suppose that there is a (q, x)-subpath of P_2 that is contained in P_3 but not in P_1, as depicted in Figure 3. Since the CP from u' to v' is not the same as the one from v' to u' (and correspondingly for u'' and v''), and P_2 and P_3 must pass through W, then q is not contained in X. From Lemma 5, all the vertices from q to w' must be the same in P_2 and P_3. Hence, P_3 goes through x, and from our initial assumption, P_2 does not have any intersection with a vertex that comes before x in P_1. Besides, P_3 goes through q' and Y. Therefore, any subpath of P_2 starting in q is also a subpath of P_3. This contradicts that $\tau_{xw} \cap S = \{P_1, P_2\}$ since $\tau_{xw} \cap S = \{P_1, P_2, P_3\}$.

Consider now the (q', v')-subpath of P_2, denoted P_2'. Suppose that P_3 has an intersection with a (r, r')-subpath of P_2' (Figure 4). From our initial assumption, P_3 goes through W and Y, so it passes through q', and q' reaches r. Hence, from Lemma 5, all the vertices from q' to r' must be the same in P_2 and P_3. In this case, P_3 does not contain a (r', w)-subpath, otherwise P_1 and P_3 would form a cycle starting and ending in r'. Besides, P_3 does not

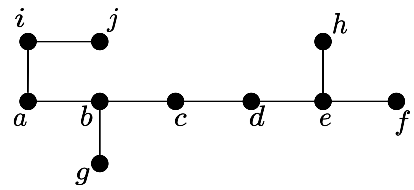

Figure 5. Graph with $\text{VCdim}(G) \geq 2$.

have a (r', y)-subpath or a (r', y')-subpath, for any $y' \in \text{Inn}(Y)$, otherwise that would be two different CPs from r' to y'. Hence, P_3 does not pass through the (q', y)-subpath of P_1, contradicting that $\tau_{q'y} \cap S = \{P_1, P_3\}$ since $\tau_{q'y} \cap S = \{P_1\}$. □

Theorem 6. *For a given graph $G = (V, E)$ and a fixed ordering σ over V,*

$$VCDim(G) \geq 2.$$

Proof. Consider the graph as in Figure 5. Then, for $P_1 = (a, b, c, d, e, f)$, $P_2 = (g, b, c, d, e, h)$, and $S = \{P_1, P_2\}$, we have: $\tau_{ac} = \{P_1\}$, $\tau_{gc} = \{P_2\}$, $\tau_{cd} = \{P_1, P_2\}$, and $\tau_{aj} = \emptyset$. □

4. Algorithms

For an undirected graph $G = (V, E)$ with non-negative edges weights, with $n = |V|$ and $m = |E|$, we first present in Section 4.1 a modified version of Dijkstra's algorithm which takes into consideration a given vertex ordering σ, and then we show that the shortest paths in the SPT computed by the algorithm are canonical paths. Then, in Section 4.2 we present an algorithm for the relaxed APSP problem that returns, with probability at least $1 - \delta$, the shortest paths with centrality least ε.

4.1. Modified Dijkstra

In this section we present a modification of Dijkstra's algorithm presented in [27]. Dijkstra's algorithm, for a given vertex s, outputs a SPT, denoted T_s, rooted in s. This algorithm maintains in every step a set S such that every vertex in S has its distance from s already computed. At every step, a vertex v in $V \setminus S$ with minimum estimated distance from s is selected to be added in S. An edge $(u, w) \in E$ is *relaxed* if the minimum distance from s to u plus the weight of (u, w) improves the minimum distance from s to w.

The main difference between the modified algorithm that we present here and the original one is the tie-breaking criterion for the selection of edges to be added in a shortest path. In a given step of the modified Dijkstra, if there are multiple vertices in $V \setminus S$ with the same estimation for minimum distance from s, then the one with minimum index in σ is chosen to be added in S. Additionally, let u be a vertex that has been just inserted in T_s in a given iteration. For every neighbor y of u in $V \setminus S$ for which the algorithm relax the edge (u, y), the ordering is taken into consideration so that if $d(s, u) + \omega(u, y) = d(s, u') + \omega(u', y)$, for some u' in S, then the tie-breaking for the shortest (s, y)-path depends on which vertex between u and u' has the minimum index in σ.

Theorem 7 shows that the modified Dijkstra's algorithm correctly computes all the canonical paths from a source s to any other vertex in V with respect to σ. Note that S is a priority queue that is also modified to give higher priority to vertices with lowest indexes in σ in the case of ties in the vertices selection. We observe, however, that these modifications do not increase the running time of the priority queue operations.

Theorem 7. *All shortest paths computed by a modified Dijkstra's algorithm with respect to a given vertex ordering σ are canonical paths.*

Proof. (Sketch) Similar to the proof of correctness of the original Dijkstra's algorithm presented in [27] (Theorem 22.6), the proof is by induction on the size of S.

Let s be the source vertex. For each $u \in V$, let $\tilde{d}(s,u)$ the estimated minimum distance from s to u in a given step of the algorithm. For $|S| = 0$, the set S is empty and then this base is trivially true. For the base where $|S| = 1$, we have $S = \{s\}$, and then $\tilde{d}(s,s) = d(s,s) = 0$. Besides, s does not have a predecessor, since it is the source, so the base is also true for this case. For the inductive step, we have the following hypothesis: for all $v \in S$, we have that $\tilde{d}(s,v) = d(s,v)$ and the predecessor of v in the Dijkstra tree of s is the one with minimum index in σ. Proving that $\tilde{d}(s,v) = d(s,v)$ follow the same arguments of the proof of correctness in [27] for the original Dijkstra's algorithm.

In order to prove that the predecessor of v in the Dijkstra tree of s, denoted v', is the one with minimum index in σ among all possible predecessors of v, we prove that all edges (z,v) where $\tilde{d}(s,v) = \tilde{d}(s,z) + \omega(z,v)$ were examined when the edge (v',v) were relaxed. Consider, by contradiction, that there is some vertex u' that has the minimum index in σ among all possible predecessors of v, but that the edge (u',v) was not examined before vertex v is added to S. If the edge (u',v) was not examined, then v was added in S before u'. In this case, this happened either because $\tilde{d}(s,v) < \tilde{d}(s,u')$ or because $\tilde{d}(s,v) = \tilde{d}(s,u')$ and $\sigma(v) < \sigma(u')$. However, in both cases, then u' could not be the predecessor of v, since $\tilde{d}(s,u')$ should be strictly smaller than $\tilde{d}(s,v)$ to be considered as a possible predecessor of v. Hence, all $y \in S$ with $\tilde{d}(s,y) < \tilde{d}(s,v)$ should have been examined before v, and hence, v' is the predecessor of v with minimum index in σ among all such vertices. This value never changes again once v is added in S. □

4.2. Computing Shortest Paths with High Centrality

Given $0 < \varepsilon, \delta < 1$, Algorithm 1 computes, with probability $1 - \delta$, the distances between pair of vertices with centrality at least ε. We also briefly describe the necessary modifications on the algorithm so that the shortest path associated to such distances be also computed.

Algorithm 1: PROBABILISTICALLPAIRSSHORTESTPATHS(G,ε,δ)

input : weighted graph $G = (V, E)$ with $n = |V|$, parameters $0 < \varepsilon, \delta < 1$.
output: distance d_{uv}, for each $(u,v) \in V^2$ s.t. $c(u,v) > \varepsilon$, with probability $1 - \delta$.

1 **for** $i \leftarrow 1$ **to** $\left\lceil \frac{c}{\varepsilon}\left(2\ln\frac{1}{\varepsilon} + \ln\frac{1}{\delta}\right)\right\rceil$ **do**
2   sample $a \in V$ with probability $1/n$
3     $T_a \leftarrow$ SINGLESOURCESHORTESTPATHS(a)   /* modified Dijkstra */
4   sample $b \in V \setminus \{a\}$ with probability $1/(n-1)$
5     $\mathcal{B}_{ab} \leftarrow$ shortest path from a to b in T_a
6   **for each** $(u,v) \in \mathcal{B}_{ab} \times \mathcal{B}_{ab}$ **do**   /* u closer to a, v closer to b */
7         $d_{uv} \leftarrow d_{av} - d_{au}$ /* d_{au} and d_{av} come from T_a */
8 **return** each d_{uv} in the distances table

Theorem 8. *Consider a (u,v)-path such that $c(u,v) \geq \varepsilon$. Algorithm 1 computes the exact distance between u and v with probability $1 - \delta$.*

Proof. Algorithm 1 samples several branches and we first assume that such samples are an ε-net (we show later that this is indeed true). Recalling the range space modeling (Section 4.2), the sample of branches is denoted by S and the (u,v)-path is related to a range τ_{uv}.

As, by lines 2 and 4, the branch is sampled with probability $1/n(n-1)$ then, by Theorem 3, we have that $c(u,v) = \Pr(\tau_{uv})$. Thus, as $c(u,v) \geq \varepsilon$, so $\Pr(\tau_{uv}) \geq \varepsilon$. As we are assuming that the sample is an ε-net, by Definition 5, then $|\tau_{uv} \cap S| \geq 1$ for all τ_{uv} such that $\Pr(\tau_{uv}) \geq \varepsilon$. That is, since $c(u,v) \geq \varepsilon$ then at least one branch of the sample S contains the (u,v)-path. If a branch $\mathcal{B}_{ab}$ in S contains the (u,v)-path, then in line 3 the exact distance between u and v is computed, since the (u,v)-path which is a subpath of the shortest path from a to b is also minimal, so its distance d_{uv} can be computed as $d_{av} - d_{au}$.

Now it remains to prove that the sample S is indeed an ε-net. Note that in lines 1–7, the loop is executed $k = \left\lceil \frac{c}{\varepsilon}\left(2\ln\frac{1}{\varepsilon} + \ln\frac{1}{\delta}\right)\right\rceil$ times, so our sample has at least size k. By Theorems 2, 5, and 6, this sample size is sufficient for it to be an ε-net with probability at least $1-\delta$. $\square$

Theorem 9. *Algorithm 1 has running time* $\mathcal{O}(m + n\log n + (Diam_V(G))^2)$.

Proof. Lines 2, 4 and 5 takes linear time. Line 3 (the modified Dijkstra) runs in $\mathcal{O}(m + n\log n)$, as the modifications do not change the running time of the original Dijkstra's algorithm. The loop in line 6 takes time $\mathcal{O}((\mathrm{Diam}_V(G))^2)$ since the length of $\mathcal{B}_{ab}$ cannot be greater than the vertex diameter of the graph. The distances returned by Dijkstra's algorithm in line 3 are stored in a table d. Since operations of insertion, deletion, and search on this data structure take time $\mathcal{O}(1)$, then updating table d takes time $\mathcal{O}(1)$. Assuming that ε and δ are constants, the number of loop iterations in lines 1–7 is constant, and the result follows. $\square$

As it is common to APSP and search algorithms, Algorithm 1 also constructs a data structure from which, for all vertices (u, w), a shortest path from u to w can be retrieved. We can store the predecessors of each vertex that is in $\mathcal{B}_{ab}$ so that a (u, v)-subpath of $\mathcal{B}_{ab}$ can be retrieved by a backward traversing from v to u on these predecessors. This modification does not change the execution time of the original algorithm.

In the remainder of this section we are interested in determining the smallest value of ε for which our algorithm would still perform on strictly subcubic time. For this, we drop the assumption that ε is constant and therefore write it as a function of n, denoted by $\varepsilon(n)$.

Let k be the sample size (which impacts on the number of times line 1 of Algorithm 1 is executed). Then $k = \mathcal{O}\left(\frac{1}{\varepsilon(n)}\ln\frac{1}{\varepsilon(n)}\right)$, and the running time of Algorithm 1 becomes $\mathcal{O}(k\cdot(m + n\log n + (\mathrm{Diam}_V(G))^2))$. In the worst case $m = \mathcal{O}(n^2)$ and then its running time is $\mathcal{O}(k\cdot n^2)$. As the best conjectured time is $\mathcal{O}(n^{3-c})$, for a constant $c > 0$ [14], then we are looking for the value of $\varepsilon(n)$ such that the time of our algorithm is upper bounded by $\mathcal{O}(n^{3-c})$, i.e. $\mathcal{O}(k\cdot n^2) = \mathcal{O}(n^{3-c})$. Thus $k = n^{1-c}$, i.e.

$$\frac{1}{\varepsilon(n)}\ln\frac{1}{\varepsilon(n)} = n^{1-c}.$$

Solving for $\varepsilon(n)$, we have $\varepsilon(n) = \frac{W_0(n^{1-c})}{n^{1-c}}$, where $W_0(n^{1-c})$ is the branch 0 of the Lambert-W function [28]. To simplify the notation, let $n' = n^{1-c}$. If $n' \geq e$, then a known bound [29] for $W_0(n')$ is $W_0(n') = \ln n' - \ln\ln n' + \Theta\left(\frac{\ln\ln n'}{\ln n'}\right)$. Therefore $\varepsilon(n) = \frac{\ln n' - \ln\ln n' + \Theta\left(\frac{\ln\ln n'}{\ln n'}\right)}{n'}$.

Note that the smallest value for the centrality of a path is $1/n(n-1)$, which is the case for a path that is not strictly contained in any other path. So, to compute the distance of paths with such small centrality, we have to use ε so small that the execution time exceeds that of the best existing algorithms [5,14]. Nevertheless, by the reasoning above, we note that we can set ε as small as $\Theta\left(\frac{\ln n'}{n'}\right)$.

5. Estimating the Shortest Path Centrality

The main objective of our paper is the computation of shortest paths with high centrality. However, one might be interested in computing the value of the centrality of such shortest paths. In this section we give the outline of how to adapt our algorithm so that the centrality of each $(u, v) \in V^2$ can be estimated within ε error, with probability at least $1-\delta$, for $0 < \varepsilon, \delta < 1$. For this task we can use the more general result of Theorem 2 applied to the notion of *ε-sample*, which states that a collection of elements $S \subseteq U$ sampled with respect to π with $|S| = \frac{c}{\varepsilon^2}\left(k + \ln\frac{1}{\delta}\right)$ is an ε-sample with probability at least $1-\delta$. More precisely, an ε-sample generalizes an ε-net in the sense that not only it intersects ranges of a sufficient size but it also guarantees the right relative frequency of each range in $\mathcal{I}$ within the sample S. That is, given $0 < \varepsilon < 1$, a set S is called ε-sample with respect to a range space $\mathcal{R} = (U, \mathcal{I})$, and a probability distribution π on U if $\forall I \in \mathcal{I}$, $|\Pr_\pi(I) - \frac{|S \cap I|}{|S|}| \le \varepsilon$.

The idea is that after building a sample of size $r = \lceil \frac{c}{\varepsilon^2}(2 + \ln \frac{1}{\delta}) \rceil$, we build a counting table $\tilde{t}$ to estimate the number of branches that contain a certain canonical path as a subpath. More specifically, the entry $\tilde{t}_{uv}$ estimates the value t_{uv} (recall Definition 2 in Section 2) for the pair of vertices (u, v). For this, the value $\tilde{t}_{uv}$ is incremented by $1/r$ in lines 6 and 7 if the branch $\mathcal{B}_{ab}$ contains the canonical path between u and v as subpath. At the end of the algorithm, if a pair of vertices (u, v) is not included in the table, the estimation for the centrality is assumed to be zero. This modification does not change the asymptotic running time of Algorithm 1. We state that in Corollary 1.

Corollary 1. *Given an undirected graph $G = (V, E)$ with non-negative edge weights, with $n = |V|$, and a sample of size $r = \lceil \frac{c}{\varepsilon^2}(2 + \ln \frac{1}{\delta}) \rceil$, Algorithm 2 has running time $\mathcal{O}(m + n \log n + (\mathrm{Diam}_V(G))^2)$ for the computation of a table from which the centrality estimation of each $u, v \in V^2$ can be retrieved.*

6. Concluding Remarks

In this paper we present a range space having the domain composed by the shortest paths of a graph G where there is one shortest path for each pair of vertices in G.We show that the VC dimension of such range space is 2. We show that this result can be applied to bound the sample size required for an approximation algorithm for a relaxed version of the All-pairs shortest path problem (APSP). In this version, we compute, with probability at least $1 - \delta$, the shortest paths of G having centrality at least ε, for $0 < \varepsilon, \delta < 1$. We present a $\mathcal{O}(m + n \log n + (\mathrm{Diam}_V(G))^2)$ running time algorithm for this task. We show that a sample of shortest paths of size $\lceil \frac{c}{\varepsilon}\left(2 \ln \frac{1}{\varepsilon} + \ln \frac{1}{\delta}\right) \rceil$ is sufficient for achieving the desired result. So, in an application where one might be interested only in computing "central" shortest paths the algorithm is rather efficient and it depends only on the parameters ε and δ (classical approaches in literature based in union bound, for example, typical require sample sizes that depend on the size of the input).

An open question that we are particularly interested is the connection between ε and n or $\mathrm{Diam}_V(G)$ for specific input distributions. For the general case, trivially setting $\varepsilon = \frac{1}{n(n-1)}$, we have a guarantee that every shortest path in G is computed with probability $1 - \delta$, but that would yield an algorithm with running time exceeding $\mathcal{O}(n^3)$. This may not be a surprise since APSP may not admit a strictly subcubic algorithm. Nevertheless, we show that if ε is at least $\frac{\ln n' - \ln \ln n' + \Theta\left(\frac{\ln \ln n'}{\ln n'}\right)}{n'}$, where $n' = 1 - c$, the running time of our algorithm is $\mathcal{O}(n^{3-c})$, for $c > 0$.

References

1. Williams, R. Faster All-pairs Shortest Paths via Circuit Complexity. In Proceedings of the 46-th Annual ACM Symposium on Theory of Computing; ACM: New York, 2014; STOC'14, pp. 664–673.
2. Chan, T.M. All-Pairs Shortest Paths for Unweighted Undirected Graphs in o(Mn) Time. *ACM Trans. Algorithms* **2012**, *8*.
3. Eirinakis, P.; Williamson, M.D.; Subramani, K. On the Shoshan-Zwick Algorithm for the All-Pairs Shortest Path Problem. *Journal of Graph Algorithms and Applications* **2017**, *21*, 177–181.
4. Brodnik, A.; Grgurovič, M. Solving all-pairs shortest path by single-source computations: Theory and practice. *Discrete applied mathematics* **2017**, *231*, 119–130.
5. Pettie, S.; Ramachandran, V. Computing Shortest Paths with Comparisons and Additions. In Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2002; SODA '02, pp. 267–276.
6. Roditty, L.; Vassilevska Williams, V. Fast approximation algorithms for the diameter and radius of sparse graphs. In Proceedings of the 45th annual ACM symposium on Theory of computing, 2013, pp. 515–524.
7. Abboud, A.; Williams, V. Popular Conjectures Imply Strong Lower Bounds for Dynamic Problems. In Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science, 2014, pp. 434–443.

8. Abboud, A.; Williams, V.; Yu, H. Matching Triangles and Basing Hardness on an Extremely Popular Conjecture. *SIAM Journal on Computing* **2018**, *47*, 1098–1122.
9. Vassilevska Williams, V. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis (invited talk). In Proceedings of the 10th International Symposium on Parameterized and Exact Computation (IPEC 2015). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.
10. Dor, D.; Halperin, S.; Zwick, U. All-pairs almost shortest paths. *SIAM Journal on Computing* **2000**, *29*, 1740–1759.
11. Roditty, L.; Shapira, A. All-pairs shortest paths with a sublinear additive error. *ACM Transactions on Algorithms (TALG)* **2011**, *7*, 1–12.
12. Shoshan, A.; Zwick, U. All pairs shortest paths in undirected graphs with integer weights. In Proceedings of the 40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039), 1999, pp. 605–614.
13. Löffler, M.; Phillips, J.M. Shape fitting on point sets with probability distributions. In Proceedings of the European symposium on algorithms. Springer, 2009, pp. 313–324.
14. Williams, R. Faster All-Pairs Shortest Paths via Circuit Complexity. *SIAM Journal on Computing* **2018**, *47*, 1965–1985.
15. Freeman, L.C. A set of measures of centrality based on betweenness. *Sociometry* **1977**, pp. 35–41.
16. Shalev-Shwartz, S.; Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*; Cambridge University Press: New York, 2014.
17. Mitzenmacher, M.; Upfal, E. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*, 2nd ed.; Cambridge University Press: New York, 2017.
18. Anthony, M.; Bartlett, P.L. *Neural Network Learning: Theoretical Foundations*, 1st ed.; Cambridge University Press: New York, 2009.
19. Mohri, M.; Rostamizadeh, A.; Talwalkar, A. *Foundations of Machine Learning*; The MIT Press: Cambridge, 2012.
20. Har-Peled, S.; Sharir, M. Relative (p, ε)-approximations in geometry. *Discrete & Computational Geometry* **2011**, *45*, 462–496.
21. Riondato, M.; Kornaropoulos, E.M. Fast approximation of betweenness centrality through sampling. *Data Mining and Knowledge Discovery* **2016**, *30*, 438–475.
22. Riondato, M.; Upfal, E. ABRA: Approximating Betweenness Centrality in Static and Dynamic Graphs with Rademacher Averages. *ACM Trans. Knowl. Discov. Data* **2018**, *12*, 61:1–61:38.
23. Lima, A.M.; da Silva, M.V.; Vignatti, A.L. Estimating the Percolation Centrality of Large Networks through Pseudo-dimension Theory. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1839–1847.
24. Lima, A.M.; da Silva, M.V.; Vignatti, A. Percolation centrality via Rademacher Complexity. *Discrete Applied Mathematics* **2021**.
25. Cousins, C.; Wohlgemuth, C.; Riondato, M. Bavarian: Betweenness Centrality Approximation with Variance-Aware Rademacher Averages. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 196–206.
26. Lima, A.M.; da Silva, M.V.G.; Vignatti, A.L. Estimating the Clustering Coefficient Using Sample Complexity Analysis. In Proceedings of the LATIN 2022: Theoretical Informatics; Springer International Publishing: Cham, 2022; pp. 328–341.
27. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. *Introduction to algorithms*; MIT press, 2022.
28. Weisstein, E. Lambert W-Function, from Wolfram Math-World, 2013.
29. Hoorfar, A.; Hassani, M. Inequalities on the Lambert W function and hyperpower function. *J. Inequal. Pure and Appl. Math* **2008**, *9*, 5–9.