Shortest Paths and Centrality in Uncertain Networks

Arkaprava Saha
NTU Singapore
saha0003@e.ntu.edu.sg

Ruben Brokkelkamp
CWI Amsterdam, The Netherlands
ruben.brokkelkamp@cwi.nl

Yllka Velaj
University of Vienna, Austria
yllka.velaj@univie.ac.at

Arijit Khan
NTU Singapore
arijit.khan@ntu.edu.sg

Francesco Bonchi
ISI Foundation, Turin, Italy
francesco.bonchi@isi.it

ABSTRACT

Computing the shortest path between a pair of nodes is a fundamental graph primitive, which has critical applications in vehicle routing, finding functional pathways in biological networks, survivable network design, among many others. In this work, we study shortest-path queries over uncertain networks, i.e., graphs where every edge is associated with a probability of existence. We show that, for a given path, it is #**P**-hard to compute the probability of it being the shortest path, and we also derive other interesting properties highlighting the complexity of computing the Most Probable Shortest Paths (MPSPs). We thus devise sampling-based efficient algorithms, with end-to-end accuracy guarantees, to compute the MPSP. As a concrete application, we show how to compute a novel concept of betweenness centrality in an uncertain graph using MPSPs. Our thorough experimental results and rich real-world case studies on sensor networks and brain networks validate the effectiveness, efficiency, scalability, and usefulness of our solution.

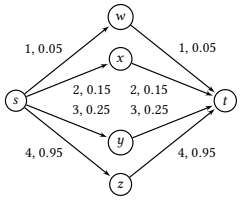
PVLDB Reference Format:
Arkaprava Saha, Ruben Brokkelkamp, Yllka Velaj, Arijit Khan, and Francesco Bonchi. Shortest Paths and Centrality in Uncertain Networks. PVLDB, 14(7): 1188-1201, 2021.
doi:10.14778/3450980.3450988

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/ArkaSaha/MPSP-Centrality.

1 INTRODUCTION

Uncertain networks, i.e., graphs where each edge is associated with a probability of existence, have received a great deal of attention thanks to their expressivity and applicability in many real world contexts. Researchers have studied k-nearest neighbor queries [39, 52], reachability queries [31], clustering [23], sampling [48], network design [30], and embedding [24], just to mention a few. Uncertainty in a network might arise due to noisy measurements [2], edge imputation using inference and prediction models [1, 40], and explicit manipulation of edges, e.g., for privacy purposes [7].

Shortest-path queries [8, 17, 27] are one of the fundamental graph primitives with a plethora of applications, e.g., traffic routing, finding functional pathways in biological networks. A critical application of shortest paths is the computation of *betweenness centrality* [10, 19, 43, 54], a measure of importance of a node based on its effectiveness in connecting pairs of other nodes via shortest paths.



Path P	Length	$Pr(\mathrm{Sh}_s^t(P))$
$P_1 : (s, w, t)$	2	0.0025
$P_2 : (s, x, t)$	4	0.0224
$P_3 : (s, y, t)$	6	0.0609
$P_4 : (s, z, t)$	8	0.8250

Figure 1: Example of paths in an uncertain graph: $Pr(\mathrm{Sh}_s^t(P))$ denotes the probability that path P is the shortest path from s to t.

In this paper, we first study the fundamental problem of computing shortest-path queries in uncertain networks, then we build over it a measure of betweeness centrality. The notion of shortest path in an uncertain graph should consider not only the length of a path but also the probability of existence of all edges on the path. Specifically, given an uncertain graph $\mathcal{G}$, a source node s, and a target node t, our goal is to find the path P from s to t with the highest probability of being the shortest path (SP), i.e., the probability with which P exists and no path shorter than P exists. We refer to such a path as the *Most Probable Shortest Path* (MPSP) from s to t.

Example 1. *Each edge in the uncertain graph in Figure 1 is annotated with its length and its probability of existence. For the source s and target t, although the path $P_1 = (s, w, t)$ is the shortest (when not considering probabilities), the one having the highest probability of being the shortest path, i.e., the MPSP from s to t, is $P_4 = (s, z, t)$, which is also the longest path (when not considering probabilities).*

Computing MPSPs is useful in many applications. Road networks are modeled as uncertain graphs because of unexpected traffic jams [25], where a vehicle driver may find the MPSP to the nearest gas station or restaurant. MPSPs are also useful in routing over wireless sensor networks, where links between sensor nodes have a probability of failure. Many applications not only require the shortest route, but also one with a high precision [22, 33], such as being the shortest with a high probability. Brain networks are often represented as weighted uncertain graphs, where nodes are the brain regions of interest (ROIs), edges indicate potential co-activation between ROIs, edge distance represents physical distance between ROIs, and edge probability indicates the strength of the co-activation signal [15]. Finding MPSPs between different ROIs of the brain could differentiate healthy brains from those with diseases, such as autism [16, 20]. In our experiments, we present

Proceedings of the VLDB Endowment, Vol. 14, No. 7 ISSN 2150-8097.
doi:10.14778/3450980.3450988

two concrete use cases of MPSPs on sensor networks (§ 5.7) and brain networks (§ 5.8).

1.1 Related Work

Several variants of shortest-path queries over uncertain graphs have been studied in the literature. Work in [12, 13, 62] investigates *threshold-based shortest-path queries* in uncertain graphs, i.e., the problem of finding all paths having shortest-path probability larger than a predefined threshold. In particular, [12, 13] consider a different uncertain graph model with correlation. The work closest to ours is probably [63], which considers MPSP queries as we do, but it does not provide any hardness result or any accuracy guarantee.

In [63], similar to [13, 62], a *filtering-and-verification* framework is used, which enumerates paths between the two given nodes in increasing order of length, without considering edges' probabilities, till a termination criterion is achieved. Among the candidate paths generated, a sampling method is applied (e.g., the Luby-Karp algorithm [28]) to approximately measure each candidate path's probability of being the shortest path. However, it may happen that the MPSP (the path we are looking for) is not one of the shortest few paths when one does not consider probabilities (as in Example 1). In this case, a filtering-and-verification approach would have to enumerate a large number of paths before including the real MPSP in the candidate set. Thus we ask the question: *can we quickly include the MPSP in the candidate set, without requiring to enumerate all paths shorter than the MPSP?* To address this, we combine Monte Carlo (MC) sampling with Dijkstra's algorithm (referred to as Dijkstra+MC) from the source node. That is, when a node is reached via Dijkstra's algorithm, its outgoing edges are sampled according to their probabilities, and only the sampled edges are considered for choosing the next node. As formally proved in § 3.3, our method will need only a small number (≈ 20) of Dijkstra+MC runs to include the MPSP in the candidate set with a high probability. We demonstrate this with an example.

Example 2. *In Figure 1, there are four paths from s to t. The path P_4 (the longest path) is the MPSP. The probabilities of the edges in P_4 are much larger than those of the edges in the other paths. Hence, a run of Dijkstra+MC on this graph produces the path P_4 with a higher probability, since the other edges are highly* unlikely *to be sampled. Thus, we need only a small number of such runs (maybe 1 or 2) to include P_4 in the candidate set. On the other hand, the method in [63] requires all the three remaining paths to be enumerated before P_4, which is clearly more time-consuming.*

The idea of Dijkstra+MC (or BFS+MC in an uncertain graph that does not consider edge lengths) has been extensively used in probabilistic reachability queries [26, 31, 33] and to build reverse-reachable sketches for the influence maximization problem [9, 57].

The work in [12], discussed before, also employs a form of Dijkstra+MC, followed by the Horvitz-Thompson (HT) unequal probability estimator, to compute the probability of being the shortest path *heuristically*, without any accuracy guarantee. While we employ Dijkstra+MC for effective and faster candidate generation, we then apply the Luby-Karp sampling to find the MPSP in this candidate set. Unlike [12], we provide end-to-end accuracy guarantees of our method, and we also experimentally demonstrate the superiority of our approach over [12].

1.2 Contributions and Roadmap

We formally define the concept of the Most Probable Shortest Path (MPSP) in an uncertain graph (§ 2), prove that our problem is #**P**-hard, and also derive other interesting properties highlighting the complexity of computing MPSPs (§ 2.1). We discuss an earlier baseline solution [63], together with its shortcomings (§ 2.2). In § 3, we propose our sampling based efficient algorithms, with end-to-end accuracy guarantees, to compute the MPSP.

We then focus on three important generalizations of our problem: first we study top-k MPSP queries for $k > 1$ (§ 3.2); followed by single-source and single-target MPSP queries (§ 3.4); then MPSPs over uncertain multi-graphs (§ 3.5). The last one provides a general data model, since it allows one to model the uncertainty as a probability distribution of the length of an edge: for instance, in road networks, it can model the probability distribution of travel times on specific road segments. Furthermore, we study MPSP-Betweenness-Centrality and develop efficient sampling strategies to compute the top-k central nodes, with theoretical quality guarantees (§ 4).

Finally, we conduct thorough experiments (§ 5) showing scalability over large-scale datasets and performance improvements against state-of-the-art methods [12, 63]. We also develop interesting case studies with sensor (§ 5.7) and brain (§ 5.8) networks.

2 PRELIMINARIES

Let $\mathcal{G} = (V, E, W, p)$ be a probabilistic (or uncertain) directed graph, where $W : E \rightarrow \mathbb{R}_{\geq 0}$ defines non-negative edge length, and $p : E \rightarrow (0, 1]$ is a function that assigns a probability of existence to each edge. Following the bulk of the literature on uncertain graphs [5, 26, 34, 35, 52, 59, 62, 63], we adopt the well-established *possible world* semantics and assume that edge probabilities are independent of each other: the uncertain graph $\mathcal{G}$ is interpreted as a probability distribution over the $2^{|E|}$ deterministic graphs (possible worlds) $G = (V, E_G, W) \sqsubseteq \mathcal{G}$ obtained by sampling each edge $e \in E$ independently at random with probability $p(e)$. That is, the probability of observing the possible world $G = (V, E_G, W)$ is:

$$Pr(G) = \prod_{e \in E_G} p(e) \prod_{e \in E \setminus E_G} (1 - p(e)) \tag{1}$$

Given a pair of distinct nodes $(s, t) \in V \times V$, a (simple) path P from s to t is an ordered sequence of edges denoted by $P = (e_1, e_2, \ldots, e_n)$, such that $e_i = (u_i, u_{i+1}) \in E$ for all $i \in \{1, 2, \ldots, n\}$, $u_1 = s$, $u_{n+1} = t$ and $u_i \neq u_j$ for $i \neq j$. For this path, the nodes s and t are called the *source* and *target* nodes respectively, while the remaining ones constitute the set $Int(P)$ of *internal nodes*. The *length* of the path P is the sum of lengths of its edges: $W(P) = \sum_{i=1}^{n} W(e_i)$. A *shortest path* from s to t in a deterministic graph G is one having the minimum length, and we denote by $SP(G, s, t)$ the set of all such paths.

In an uncertain graph $\mathcal{G}$, let $\mathcal{P}(\mathcal{G}, s, t)$ denote the set of all paths from s to t. Given a path P, the event that P exists (resp. does not exist) is denoted by $\mathbf{X}(P)$ (resp. $\overline{\mathbf{X}(P)}$), and $Pr\big(\mathbf{X}(P)\big) = \prod_{i=1}^{n} p(e_i) = 1 - Pr\big(\overline{\mathbf{X}(P)}\big)$. We also denote by $\mathbf{Sh}_s^t(P)$ the event that P happens to be a shortest path from s to t, whose probability is:

$$Pr(\mathbf{Sh}_s^t(P)) = \sum_{G \sqsubseteq \mathcal{G}} Pr(G) \times \mathbb{1}[P \in SP(G, s, t)] \tag{2}$$

where $\mathbb{1}[.]$ is the indicator function.

The main problem studied in this paper is as follows.

Problem 1 (Most Probable Shortest Path (MPSP)). *Given an uncertain graph $\mathcal{G} = (V, E, W, p)$ and two nodes $s, t \in V$, find the Most Probable Shortest Path (MPSP) from s to t. Formally:*

$$MPSP(\mathcal{G}, s, t) = \underset{P \in \mathcal{P}(\mathcal{G}, s, t)}{\arg\max}\ Pr(\mathbf{Sh}_s^t(P)) \tag{3}$$

2.1 Hardness of the Problem

One factor that makes Problem 1 challenging is that even computing the probability of being the shortest path between two given nodes, for a given path, is hard.

Theorem 1. *Given an uncertain graph $\mathcal{G} = (V, E, W, p)$ and a path $P \in \mathcal{P}(\mathcal{G}, s, t)$, the problem of computing the probability of P being a shortest path from s to t in $\mathcal{G}$ is #**P**-hard.*

Proof. We prove the #**P**-hardness by polynomial-time reduction from the *s-t connectedness* problem, which is known to be #**P**-hard [59]. Given a certain (deterministic) graph $G = (V, E)$, and two nodes s and t, the goal of the s-t connectedness problem is to find the number of subgraphs of G in which there is a path from s to t.

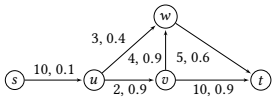
Consider an arbitrary instance of the s-t connectedness problem with inputs $G = (V_1, E_1)$ and two nodes $s, t \in V_1$. Let $n = |V_1|$. The deterministic graph G is converted to an uncertain graph $\mathcal{G} = (V_1 \cup V_2, E_1 \cup E_2, W, p)$, where $V_2 = \{v_1, v_2, \ldots, v_n\}$ is a set of n new nodes and $E_2 = \{(s, v_1), (v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n), (v_n, t)\}$. In other words, G is augmented with a new path P from s to t. The functions W and p are defined $\forall e \in E_1 \cup E_2$ as $W(e) = 1$ and

$$p(e) = \begin{cases} \frac{1}{2} & \text{if } e \in E_1 \\ 1 & \text{if } e \in E_2 \end{cases}$$

We make three observations. (*i*) Every possible world $G' \sqsubseteq \mathcal{G}$ for which $Pr(G') > 0$ contains the path P and satisfies $Pr(G') = \left(\frac{1}{2}\right)^{|E_1|}$. (*ii*) There is a bijection between the set of subgraphs of G and the set of possible worlds of $\mathcal{G}$ with non-zero probability. A subgraph $G'' = (V'', E'')$ of G can be mapped to the possible world $G' = (V'' \cup V_2, E'' \cup E_2, W)$ of $\mathcal{G}$. This mapping is clearly one-to-one, since $V'' \cap V_2 = \phi$ and $E'' \cap E_2 = \phi$ by definition. To see why it is onto, note that any possible world of $\mathcal{G}$, that exists with positive probability, must contain all edges in E_2, since $p(e) = 1$ $\forall e \in E_2$. Hence, given a possible world $G' = (V', E', W)$ of $\mathcal{G}$, there exists a subgraph $G'' = (V' \setminus V_2, E' \setminus E_2, W)$ which is the pre-image of G' under the mapping. (*iii*) For a subgraph $G'' = (V'', E'')$ of G and its corresponding possible world $G' = (V'' \cup V_2, E'' \cup E_2, W)$ of $\mathcal{G}$, P is the shortest path from s to t in G' if and only if s and t are disconnected in G''. The *'if'* part is trivial. The *'only if'* part follows since $W(P) = n + 1$ and $W(P') \leq n - 1$, where P' denotes any path from s to t in G''.

Putting together the above observations, we obtain the following:

$$\begin{aligned} Pr(\mathbf{Sh}_s^t(P)) &= \sum_{G' \sqsubseteq \mathcal{G}} Pr(G') \times \mathbb{1}[P \in SP(G', s, t)] \\ &= 1 - \sum_{G' \sqsubseteq \mathcal{G}} Pr(G') \times \mathbb{1}[P \notin SP(G', s, t)] \\ &= 1 - \left(\frac{1}{2}\right)^{|E_1|} \sum_{G' \sqsubseteq \mathcal{G} : Pr(G') > 0} \mathbb{1}[P \notin SP(G', s, t)] \end{aligned}$$



Path P	W(P)	$Pr(\mathbf{Sh}_s^t(P))$
(s, u, w, t)	18	0.024
(s, u, v, w, t)	21	0.029
(s, u, v, t)	22	0.035

Figure 2: An example to demonstrate properties of MPSP

From observation (*iii*), the summation term in the last line is exactly the number of subgraphs of G in which the nodes s and t are connected. Thus, a solution to our problem on $\mathcal{G}$ provides a solution to the s-t connectedness problem on G. This reduction involves $O(n)$ node and edge additions to G, and hence takes time polynomial in the size of G. □

In addition to #**P**-hardness, there are some other properties of MPSPs that make our problem hard. *Many of the classical properties of shortest paths over deterministic graphs no longer hold for MPSPs in uncertain graphs.* For instance, the concatenation of two MPSPs, and a subpath of an MPSP, are not necessarily MPSPs. We demonstrate these properties next, using the uncertain graph $\mathcal{G}$ in Figure 2.

In the following, we denote by $\mathcal{M}(\mathcal{G}, s, t)$ the set of MPSPs from s to t, and by $\mathbb{M}(\mathcal{G})$ the set of all MPSPs between all pairs of nodes, i.e,. $\mathbb{M}(\mathcal{G}) = \bigcup_{(s,t) \in V \times V} \mathcal{M}(\mathcal{G}, s, t)$.

Observation 1. *Given an uncertain graph $\mathcal{G} = (V, E, W, p)$, an MPSP $P \in \mathbb{M}(\mathcal{G})$ and a subpath Q of P, it is possible that $Q \notin \mathbb{M}(\mathcal{G})$.*

Consider the path $(s, u, v, t) \in \mathcal{M}(\mathcal{G}, s, t)$ and its subpath (v, t). The probabilities of being a shortest path from v to t turn out to be $Pr(\mathbf{Sh}_v^t(v, t)) = 0.414$ and $Pr(\mathbf{Sh}_v^t(v, w, t)) = 0.540$, so that (v, t) is not even the MPSP from v to t. The observation follows.

Given two paths $P = (e_1, \ldots, (u, v))$ and $Q = ((v, w), \ldots, e_n)$, the *concatenation* of P and Q, denoted by $P \cdot Q$, is defined as the sequence $P \cdot Q = (e_1, \ldots, (u, v), (v, w), \ldots, e_n)$. Note that the concatenation of two paths P and Q is defined only when the target node of P is the same as the source node of Q. The next observation states that the concatenation of two MPSPs is not necessarily an MPSP.

Observation 2. *Given an uncertain graph $\mathcal{G} = (V, E, W, p)$ and two MPSPs $P, Q \in \mathbb{M}(\mathcal{G})$, such that the target node of P is the same as the source node of Q, it is possible that $P \cdot Q \notin \mathbb{M}(\mathcal{G})$.*

Notice that since $P = (s, u, v)$ is the only path from s to v, it is clear that $\mathcal{M}(\mathcal{G}, s, v) = \{(s, u, v)\}$. Also, as shown in Observation 1, $Q = (v, w, t) \in \mathcal{M}(\mathcal{G}, v, t)$. However, $P \cdot Q = (s, u, v, w, t) = P_2 \notin \mathcal{M}(\mathcal{G}, s, t)$, and hence $P \cdot Q \notin \mathbb{M}(\mathcal{G})$.

2.2 Baseline: Filtering-and-Verification

In our experiments (§5) we use as a baseline the filtering-and-verification approach of [63]. This method consists of two steps: generating a set of candidate paths containing the MPSP, and using Luby-Karp sampling to find the MPSP in this set.

For step 1, given a source s and a target t, Yen's algorithm [61] is used to progressively generate s-t paths $P_1, P_2, P_3, \ldots$ in ascending order of lengths. For any i, using paths $P_1, \ldots, P_i$, a lower bound $LB(P_i)$ and an upper bound $UB(P_i)$ on the probability that the path P_i is the SP is computed. The upper bound is monotonically decreasing in i, and hence, if $UB(P_i) < \epsilon$ for some $\epsilon > 0$, $UB(P_j) < \epsilon$ for all $j > i$. For including the MPSP in the candidate set, the algorithm

continues to generate paths until $UB(P_{i+1}) < \max_{j\in[1,i]}\{LB(P_j)\}$ for some i. This gives the candidate set $\{P_1, \ldots, P_i\}$.

Step 2 consists of running the Luby-Karp algorithm [28] to approximate the probability that each path in the candidate set is the MPSP. It returns the path with the highest such probability.

Two major shortcomings have an influence on the performance of this method. First, the number of candidates generated can be very large, even exponential in the input size. For both lower bounds LB given in [63], it holds that $LB(P_j) \leq Pr(\mathbf{X}(P_j))$. The upper bound on the probability of path P_i being the SP is computed as $UB(P_i) = 1 - \sum_{j=1}^{i-1} LB(P_j)$. If the probability of existence of the MPSP is low, then those of the other shorter paths would generally be low. Hence, the upper bound will decrease very slowly, and it can take a lot of time before the candidate generation terminates.

The second shortcoming is the computational cost of candidate generation. Assume that we generate k paths before the candidate generation terminates. This step has time complexity $O(k|V|(|E| + |V|\log|V|))$. As mentioned in the first shortcoming, the number of candidates k can become very large, and even if it is small, we have the $|V||E|$ factor. Empirically (§5) we find that the candidate generation does not finish in one hour for our synthetic datasets.

3 PROPOSED SOLUTION

We propose a two-phase algorithm to approximate the MPSP between two nodes in an uncertain graph. In the first phase we compute paths that are candidates for being the MPSP (via Dijkstra+MC), and in the second phase we approximate the probability of each candidate path being the shortest path (via Luby-Karp algorithm). Our method is described in § 3.1 and theoretical guarantees on the quality of the returned path are provided in § 3.3.

Dijkstra+MC is simple, yet effective and efficient for candidate generation as we argued in Example 2 (§ 1.1). Our novel algorithmic contributions include pairing up Dijkstra+MC with the Luby-Karp algorithm for ultimately finding the MPSP approximately, with accuracy guarantees. Empirical results show that our algorithm has better accuracy and scalability over the baseline [63] (§2.2), and over more advanced sampling approaches, e.g., Horvitz-Thompson unequal probability estimator (we demonstrate it in §5.4). Among other novel algorithmic contributions, we extend our method to find the top-k MPSPs for $k > 1$ (§ 3.2), single-source and single-target MPSP queries (§ 3.4), and to compute the MPSPs in uncertain multi-graphs (§ 3.5). Our final technical contribution is to define a novel MPSP-Betweenness-Centrality as a concrete application (§ 4); we then develop efficient sampling strategies to compute the top-k central nodes, with theoretical quality guarantees.

3.1 Two-Phase Algorithm

In Algorithm 1 we describe our two-phase approach.

Phase 1: Dijkstra+MC. Given an uncertain graph $\mathcal{G} = (V, E, W, p)$ and two nodes $(s, t) \in V \times V$, the first phase involves computing paths that are candidates for being the MPSP from s to t. This is done by performing m independent runs of Dijkstra's algorithm on $\mathcal{G}$, where m is a hyperparameter (lines 2 to 7 of Algorithm 1). Dijkstra's algorithm on an uncertain graph is similar to the classic algorithm on deterministic graphs, except that when the algorithm reaches a node in the uncertain graph, its outgoing edges are sampled according to their respective probabilities (Algorithm 2). At any stage, only the sampled edges are considered for choosing the next node. This is equivalent to running Dijkstra's algorithm on a possible world $G \sqsubseteq \mathcal{G}$. If t is reachable from s in the sampled possible world G, then Dijkstra's algorithm on G results in an s-t path which is added to the set of candidate paths denoted by CP. Otherwise, if t is not reachable, then an empty path (denoted by P_ϕ in Algorithms 1 and 2) is returned.

Algorithm 1 Approximating the MPSP from s to t

Input: Uncertain graph $\mathcal{G} = (V, E, W, p)$, source s, target t, positive integers m and N
Output: An (approximate) MPSP from s to t
1: $CP \leftarrow \phi$
2: **for** $i = 1$ to m **do**
3: $P \leftarrow$ Alg. 2 $(\mathcal{G}, s, t)$
4: **if** $P \neq P_\phi$ **then**
5: $CP \leftarrow CP \cup \{P\}$
6: **end if**
7: **end for**
8: $LP \leftarrow$ All paths in CP in increasing order of length
9: **for** $i = 1$ to $|LP|$ **do**
10: $\widehat{\mathbf{p}}(LP[i]) \leftarrow$ Alg. 3 $(\mathcal{G}, s, t, LP[i], \{LP[1], \ldots, LP[i-1]\}, N)$
11: **end for**
12: **return** $\arg\max_{P\in LP}[\widehat{\mathbf{p}}(P)]$

Algorithm 2 Candidate Generation with Dijkstra+MC

Input: Uncertain graph $\mathcal{G} = (V, E, W, p)$, source s, target t
Output: A path from s to t
1: $u \leftarrow s, vis \leftarrow \{s\}, \mathbf{P}[v] \leftarrow P_\phi \ \forall v \in V$
2: **repeat**
3: **for all** $e = (u, v) \in E$ s.t. $v \notin vis$ **do**
4: **if** $W(\mathbf{P}[v]) > W(\mathbf{P}[u]) + W(e)$ **then**
5: With probability $p(e)$, $\mathbf{P}[v] \leftarrow \mathbf{P}[u] \cdot (e)$
6: **end if**
7: **end for**
8: $u \leftarrow \arg\min_{v\in V\setminus vis} W(\mathbf{P}[v])$
9: $vis \leftarrow vis \cup \{u\}$
10: **until** $u = t$ **or** $\mathbf{P}[u] = P_\phi$
11: **return** $\mathbf{P}[t]$

Algorithm 3 Estimate $Pr(\mathbf{Sh}_s^t(P))$ for a path P from s to t

Input: Uncertain graph $\mathcal{G} = (V, E, W, p)$, source s, target t, s-t paths P and $\{P_1, \ldots, P_n\}$ shorter than P, positive integer N
Output: An estimate of $Pr(\mathbf{Sh}_s^t(P))$
1: $C \leftarrow 0, S \leftarrow \sum_{i=1}^{n} Pr(\mathbf{X}(P_i \setminus P))$
2: **for** $r = 1$ to N **do**
3: Sample $i \in [1, n]$ with probability $\frac{Pr(\mathbf{X}(P_i\setminus P))}{S}$
4: Sample $G = (V, E_G, W) \sqsubseteq \mathcal{G}$ such that $(P_i \setminus P) \subseteq E_G$
5: **if** $\forall (j < i)\,[(P_j \setminus P) \nsubseteq E_G]$ **then**
6: $C \leftarrow C + 1$
7: **end if**
8: **end for**
9: $\widehat{p} \leftarrow \frac{C}{N} \times S$
10: **return** $(1 - \widehat{p}) \times Pr(\mathbf{X}(P))$

Phase 2: Probability Approximation. In the second phase, the Luby-Karp algorithm (Algorithm 3) is employed to compute an approximation of the probability of each candidate path being the shortest s-t path in $\mathcal{G}$. Intuitively, given a path P and some other shorter paths from s to t, along with a hyperparameter N, the algorithm first estimates the probability $\widehat{p}$ of existence of any of the paths shorter than P by generating N suitable possible worlds via Monte Carlo sampling, and then it returns the value $(1 - \widehat{p}) \times Pr(\mathbf{X}(P))$ as an estimate of $Pr(\mathbf{Sh}_s^t(P))$.

Notice that in order to approximate the probability of a path P being the shortest path in $\mathcal{G}$, the Luby-Karp algorithm, as described in [63], requires as input all the paths that are shorter than

P. Although the set of candidate paths computed after m runs of Algorithm 2 does not necessarily include all such paths, we shall show in § 3.3 that we can still provide good approximation guarantees.

Time Complexity. In Phase 1, we perform m Dijkstra's runs on the uncertain graph $\mathcal{G}$, which has time complexity $O\big(m(|E| + |V|\log|V|)\big)$. However, due to sampling of edges, Dijkstra is run on a smaller graph than the original uncertain graph, thus practically it is even more efficient. In Phase 2, first we need to sort (at most) m distinct candidate paths. This step requires $O(m \log m)$ time. Then, we run Algorithm 3 for each candidate path, which has time complexity $O(N|E|)$, N being the number of Monte Carlo (MC)-runs in the Luby-Karp algorithm. Therefore, the overall time complexity of our method is: $O\left(m\left(N|E| + |V|\log|V| + \log m\right)\right)$.

Space Complexity. Both Dijkstra+MC and Luby-Karp have lower memory footprints, and do not have much additional overhead other than storing the graph, which is $O(|E|+|V|)$ via adjacency list. Additionally, Dijkstra+MC generates at most m candidate paths, which require at most $O(m|E|)$ storage, but practically it is less since a path generally has fewer than $|E|$ edges. Thus, the space complexity of our method is: $O(m|E| + |V|)$.

3.2 Extension to Top-k MPSPs

The method presented in §3.1 can be easily extended to compute the top-k MPSPs where $k > 1$. We notice that if the number of candidate paths is smaller than or equal to k, we return all the candidate paths. Otherwise, we modify Algorithm 1 so that it stores every candidate path P and the estimate of $Pr(\mathbf{Sh}_s^t(P))$ in decreasing order of the probabilities, and then it returns the top-k elements.

We provide theoretical guarantees that *with a high probability, the true top-k shortest paths are the ones returned by our algorithm.*

3.3 Accuracy Guarantees

As a first step, notice that an s-t path P is returned after one run of Algorithm 2 if and only if Algorithm 2 samples a possible world of $\mathcal{G}$ in which P is a shortest path from s to t. Thus, the probability of the former is equal to that of the latter, which, by definition, is equal to $Pr(\mathbf{Sh}_s^t(P))$. Extending this to m runs of Algorithm 2, denoting by CP the set of all (candidate) paths returned, for any given path P, we have $Pr(P \in CP) = 1 - \big(1 - Pr(\mathbf{Sh}_s^t(P))\big)^m$. Further extending to k paths, the probability of any given set $\{P_1, \ldots, P_k\}$ of k s-t paths being included in CP is, by the inclusion-exclusion principle:

$$\begin{aligned} Pr(\{P_1,\ldots,P_k\} \subseteq CP) &= Pr\left(\bigwedge_{i=1}^{k} P_i \in CP\right) = 1 - Pr\left(\bigvee_{i=1}^{k} P_i \notin CP\right) \\ &= \sum_{i=0}^{k} (-1)^i \sum_{S \subseteq \{P_1,\ldots,P_k\} : |S|=i} \left(1 - \sum_{P \in S} Pr(\mathbf{Sh}_s^t(P))\right)^m \end{aligned} \tag{4}$$

A key observation is that, for an MPSP P^*, $Pr(P^* \in CP)$ is very high for a reasonably large value of $Pr(\mathbf{Sh}_s^t(P^*))$, even for small m. For example, consider the MPSP P_4 in the graph in Figure 1 for which $Pr(\mathbf{Sh}_s^t(P_4)) = 0.825$. Setting $m = 20$ yields $Pr(P_4 \in CP) > 0.999$. Also, in our experiments, the path P returned by our method for most of the synthetic networks and the road networks for the smaller hop queries satisfies $Pr(\mathbf{Sh}_s^t(P^*)) > 0.06$, and hence $Pr(P^* \in CP) > 0.7$ with $m = 20$.

Before proceeding, we define some useful notations that we will use throughout the remainder of the section. Given an uncertain graph $\mathcal{G} = (V, E, W, p)$, a source node s, a target node t, a set of s-t paths CP, and any path $P \in CP$, we use the following notation:

- $\mathbf{A}(P)$: Set of all paths in $\mathcal{G}$ that are shorter than P.
- $\mathbf{C}(P)$: Set of all paths in CP shorter than P, i.e., $CP \cap \mathbf{A}(P)$.
- $\mathbf{M}(P) = \mathbf{A}(P) \setminus \mathbf{C}(P)$.
- $\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big)$: Probability that P exists and no path in $\mathbf{C}(P)$ exists, i.e., $Pr\big(\mathbf{X}(P)\big)\big[1 - Pr\big(\bigcup_{Q \in \mathbf{C}(P)} \mathbf{X}(Q \setminus P)\big)\big]$ where $Q \setminus P$ is the set of all edges in Q that are not in P. Clearly, $\mathbf{p_{ne}}\big(P, \mathbf{A}(P)\big) = Pr(\mathbf{Sh}_s^t(P))$.
- $\mathbf{p_m}\big(P, \mathbf{C}(P)\big)$: Sum (over all paths Q shorter than P and missing from CP) of the probability that Q is the shortest s-t path and that P exists, i.e., $\sum_{Q \in \mathbf{M}(P)} Pr(\mathbf{Sh}_s^t(Q) \wedge \mathbf{X}(P))$.
- $\widehat{\mathbf{p}}\big(P, \mathbf{C}(P)\big)$: Output of Alg. 3 ($\mathcal{G}$, s, t, P, $\mathbf{C}(P)$, N).

Even if the true top-k MPSPs are included in CP, the probability of them being the paths finally returned depends on the quality of the approximation computed in Algorithm 3 for every single path in CP. Fortunately, there is a guarantee on this quality [29, 63].

Theorem 2 ([29, 63]). *Given an uncertain graph $\mathcal{G} = (V, E, W, p)$, a source node s and a target node t, a set of s-t paths CP, and a path $P \in CP$, $\widehat{\mathbf{p}}\big(P, \mathbf{C}(P)\big)$ is an accurate estimate of $\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big)$ with a high probability. More formally, for all $\epsilon \in [0, 2]$,*

$$Pr\Big(\big|\widehat{\mathbf{p}}\big(P, \mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big)\big| \geq \epsilon\Big) \leq 2\exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right) \tag{5}$$

However, as mentioned in § 3.1, the quality of approximating $Pr(\mathbf{Sh}_s^t(P))$ could be hampered because the set CP computed after m runs of Algorithm 2 may not include all paths shorter than the path in question. We shall show that, even then, the approximation made by Algorithm 3 is very accurate with a high probability. To this end, we first provide a lower and an upper bound on the difference in the SP probability resulting from missing out some shorter paths.

Theorem 3. *Given an uncertain graph $\mathcal{G} = (V, E, W, p)$, a source node s, and a target node t, let CP denote a set of paths from s to t. Consider a path $P \in CP$. Then*

$$0 \leq \mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) \leq \mathbf{p_m}\big(P, \mathbf{C}(P)\big) \tag{6}$$

Proof. We have, by definition, the following:

$$\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) = Pr\big(\mathbf{X}(P)\big)\left[1 - Pr\left(\bigcup_{Q \in \mathbf{C}(P)} \mathbf{X}(Q \setminus P)\right)\right]$$

$$Pr(\mathbf{Sh}_s^t(P)) = Pr\big(\mathbf{X}(P)\big)\left[1 - Pr\left(\bigcup_{Q \in \mathbf{A}(P)} \mathbf{X}(Q \setminus P)\right)\right]$$

Let us define:

$$D_A = Pr\left(\bigcup_{Q \in \mathbf{A}(P)} \mathbf{X}(Q \setminus P)\right), \quad D_C = Pr\left(\bigcup_{Q \in \mathbf{C}(P)} \mathbf{X}(Q \setminus P)\right)$$

This means that

$$\begin{aligned} &\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) \\ &= Pr\big(\mathbf{X}(P)\big)\left[Pr\left(\bigcup_{Q \in \mathbf{A}(P)} \mathbf{X}(Q \setminus P)\right) - Pr\left(\bigcup_{Q \in \mathbf{C}(P)} \mathbf{X}(Q \setminus P)\right)\right] \\ &= Pr\big(\mathbf{X}(P)\big) \times (D_A - D_C) \end{aligned} \tag{7}$$

By definition, $\mathbf{C}(P) \subseteq \mathbf{A}(P)$. Thus it holds that $D_A - D_C \geq 0$. Now, observe that any path $Q \in \mathbf{A}(P)$ is shorter than P. Since $\mathbf{A}(P)$ contains all paths in $\mathcal{G}$ that are shorter than P, the set of all paths

in $\mathbf{A}(P)$ shorter than Q is exactly equal to that of all paths in $\mathcal{G}$ that are shorter than Q, which is, by definition, equal to $\mathbf{A}(Q)$. Hence,

$$D_A = Pr\left(\bigcup_{Q\in\mathbf{A}(P)} \mathbf{X}(Q\setminus P)\right) = \sum_{Q\in\mathbf{A}(P)}\left[Pr\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - Pr\left(\bigcup_{R\in\mathbf{A}(Q)}\mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right] \quad (8)$$

By a similar reasoning, the set of all paths in $\mathbf{C}(P)$ shorter than Q is exactly equal to $\mathbf{C}(Q)$. Hence,

$$D_C = Pr\left(\bigcup_{Q\in\mathbf{C}(P)} \mathbf{X}(Q\setminus P)\right) = \sum_{Q\in\mathbf{C}(P)}\left[Pr\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - Pr\left(\bigcup_{R\in\mathbf{C}(Q)}\mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right] \geq \sum_{Q\in\mathbf{C}(P)}\left[Pr\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - Pr\left(\bigcup_{R\in\mathbf{A}(Q)}\mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right] \quad (9)$$

where (9) follows because $\mathbf{C}(Q)\subseteq\mathbf{A}(Q)$ by definition.
Note that (8) and (9) are summations of the same term across all paths Q in $\mathbf{A}(P)$ and $\mathbf{C}(P)$ respectively. Since $\mathbf{C}(P)\subseteq\mathbf{A}(P)$ and $\mathbf{A}(P)\setminus\mathbf{C}(P) = \mathbf{M}(P)$ by definition,

$$D_A - D_C \leq \sum_{Q\in\mathbf{M}(P)}\left[Pr\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - Pr\left(\bigcup_{R\in\mathbf{A}(Q)}\mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right] \quad (10)$$

Plugging (10) into (7), and using $D_A - D_C \geq 0$, we have

$$0 \leq \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) = Pr\big(\mathbf{X}(P)\big)\times\big(D_A - D_C\big) \leq Pr\big(\mathbf{X}(P)\big)\sum_{Q\in\mathbf{M}(P)}\left[Pr\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - Pr\left(\bigcup_{R\in\mathbf{A}(Q)}\mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right] = \sum_{Q\in\mathbf{M}(P)} Pr(\mathbf{Sh}_s^t(Q)\wedge X(P)) = \mathbf{p_m}\big(P,\mathbf{C}(P)\big) \quad \text{(by definition)}$$

This completes the proof. □

Note that from (4), we can say that for every s-t path missing from CP (not returned in any run of Algorithm 2), it is highly likely that the probability of that path being a shortest s-t path is extremely small. Thus, for any s-t path $P\in CP$, the sum of the shortest-path probabilities of all paths shorter than P and missing from CP is also very small, and hence $\mathbf{p_m}\big(P,\mathbf{C}(P)\big)$, which also includes the condition that P exists, is even smaller.

Using Theorems 2 and 3, we can provide a quality guarantee for Algorithm 3 on a single path even with some shorter paths missing.

Theorem 4. *Consider an uncertain graph $\mathcal{G} = (V,E,W,p)$, a source node s and a target node t, a set CP of s-t paths and a path $P\in CP$. Then, $\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big)$ is an accurate estimate of $Pr(\mathbf{Sh}_s^t(P))$ with a high probability. Formally, assuming $\mathbf{p_m}\big(P,\mathbf{C}(P)\big)\in[0,1]$, for all $\epsilon\in[0,1]$, the following holds.*

$$Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) - \mathbf{p_m}\big(P,\mathbf{C}(P)\big)\geq\epsilon\Big) \leq \exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right) \quad (11)$$

$$Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) \leq -\epsilon\Big) \leq \exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right) \quad (12)$$

Proof. Note that $\mathbf{p_m}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) + \Pr(\mathbf{Sh}_s^t(P))\in[0,1]$ from Theorem 3. Thus $\epsilon + \mathbf{p_m}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) + Pr(\mathbf{Sh}_s^t(P))\in[0,2]$. Applying Theorem 2,

$$\begin{aligned} &Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) - \mathbf{p_m}\big(P,\mathbf{C}(P)\big)\geq\epsilon\Big)\\ &= Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big)\geq\epsilon + \mathbf{p_m}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) + Pr(\mathbf{Sh}_s^t(P))\Big)\\ &\leq \exp\left(-\frac{N\Big(\epsilon + \mathbf{p_m}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) + Pr(\mathbf{Sh}_s^t(P))\Big)^2}{4|\mathbf{C}(P)|}\right)\\ &\leq \exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right)\end{aligned}$$

Similarly, $\mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P))\in[0,1]$. Theorem 2 gives

$$\begin{aligned} &Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) \leq -\epsilon\Big)\\ &= Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) \leq -\epsilon - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) + Pr(\mathbf{Sh}_s^t(P))\Big)\\ &\leq \exp\left(-\frac{N\Big(\epsilon + \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P))\Big)^2}{4|\mathbf{C}(P)|}\right) \leq \exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right)\end{aligned}$$

Hence, the theorem. □

We now prove the accuracy guarantee of our top-k MPSP method.

Theorem 5. *Given an uncertain graph $\mathcal{G} = (V,E,W,p)$, a source node s, a target node t, and an integer k, let $P_1,\ldots,P_{k+1}$ denote the true top $k+1$ MPSPs (in order) from s to t. Then, $P_1,\ldots,P_k$ are indeed the paths returned by our method with a high probability. Formally, define: $mid = \frac{1}{2}[Pr(\mathbf{Sh}_s^t(P_k)) + Pr(\mathbf{Sh}_s^t(P_{k+1})) + \mathbf{p_m}(P_{k+1},\mathbf{C}(P_{k+1}))]$, a set of s-t candidate paths CP, and for all $P\in CP$,*

$$d_P = \begin{cases} Pr(\mathbf{Sh}_s^t(P)) - mid & \text{if } P\in\{P_1,\ldots,P_k\}\\ mid - Pr(\mathbf{Sh}_s^t(P)) - \mathbf{p_m}(P,\mathbf{C}(P)) & \text{otherwise}\end{cases}$$

and assume that $d_P\in[0,1]$. This assumption is reasonable since, as noted earlier, $\mathbf{p_m}(P,\mathbf{C}(P))$ is very small. Then the probability that $P_1,\ldots,P_k$ are the paths returned is at least

$$Pr(\{P_1,\ldots,P_k\}\subseteq CP)\prod_{P\in CP}\left[1-\exp\left(-\frac{Nd_P^2}{4|\mathbf{C}(P)|}\right)\right]$$

Proof. The random variables $\widehat{\mathbf{p}}(P,\mathbf{C}(P))$ for all $P\in CP$ are independent, since the Monte Carlo rounds of Algorithm 3 on input path P do not depend on each other. Hence, the probability that $P_1,\ldots,P_k$ are the paths returned is at least

$$\begin{aligned} &Pr(\{P_1,\ldots,P_k\}\subseteq CP)\times\prod_{P\in\{P_1,\ldots,P_k\}}Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) > mid\Big)\\ &\quad\times\prod_{P\in CP\setminus\{P_1,\ldots,P_k\}}Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) < mid\Big)\\ &= Pr(\{P_1,\ldots,P_k\}\subseteq CP)\\ &\quad\times\prod_{P\in\{P_1,\ldots,P_k\}}\Big[1 - Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) \leq -d_P\Big)\Big]\\ &\quad\times\prod_{P\in CP\setminus\{P_1,\ldots,P_k\}}\Big[1 - Pr\Big(\widehat{\mathbf{p}}\big(P,\mathbf{C}(P)\big) - Pr(\mathbf{Sh}_s^t(P)) - \mathbf{p_m}\big(P,\mathbf{C}(P)\big)\geq d_P\Big)\Big]\\ &\geq Pr(\{P_1,\ldots,P_k\}\subseteq CP)\prod_{P\in CP}\left[1-\exp\left(-\frac{Nd_P^2}{4|\mathbf{C}(P)|}\right)\right]\end{aligned}$$

□

3.4 Single-Source and Single-Target MPSPs

Our approach for generating the MPSP from a single source to a single target can be easily extended to compute MPSPs from a single source to all other nodes in the graph. Phase 1 continues running Dijkstra+MC on the entire graph until all edges are sampled, or no new target nodes can be reached. Phase 2 runs separately for each individual target (i.e., each source-target pair). A similar strategy can be applied for computing MPSPs to a single target from all other nodes: we need to use the same method on the graph with the edges reversed. Since Phase 1 is not run separately for each source-target pair, this helps in reducing the running time of this phase from $|V|$ times that of a single source-target pair to a smaller value. This is demonstrated empirically in § 5.6.

3.5 Extension to Uncertain Multi-Graphs

An uncertain multi-graph is a quadruple (V, E, W, p), where V is a set of nodes and $E \subseteq V \times V \times \mathbb{R}_{\geq 0} \times (0, 1]$ is a set of directed edges with lengths (W) and probabilities of existence (p), such that every pair of nodes can be connected by zero, one, or more edges, called *parallel edges*, with a distinct combination of length and probability of existence. This more general data model can be used, e.g., to incorporate a probability distribution of travel times on a segment of a road network, depending on the traffic conditions.

Given a pair of nodes $(s, t) \in V \times V$, a (simple) path in an uncertain multi-graph is an ordered sequence of edges $(e_1, e_2, \ldots, e_n)$ where $e_i = (u_i, u_{i+1}, w_i, p_i) \in E$, $u_1 = s$, $u_{n+1} = t$ and $u_i \neq u_j$ for $i \neq j$. Our algorithm, described in § 3.1, can be easily adapted to find MPSPs in uncertain multi-graphs. The main difference lies in the generation of the candidate paths. In Phase 1, when we reach a node in the uncertain graph, its outgoing edges are sampled with their respective probabilities, and only one sampled edge from the current node to each adjacent node (having the minimum length among all sampled edges from the current node to that adjacent node) is considered for updating the paths in line 5 of Algorithm 2.

4 MPSP-BETWEENNESS CENTRALITY

We next define MPSP-Betweenness Centrality in uncertain graphs.

In a deterministic directed graph $G = (V, E, W)$, the *betweenness centrality* of a node $v \in V$ is defined as

$$b_G(v) = \frac{1}{|V|(|V|-1)} \sum_{\substack{(s,t) \in V \times V \\ s \neq v \neq t, \sigma(s,t) \neq 0}} \frac{\sigma(s, t|v)}{\sigma(s, t)} \tag{13}$$

$\sigma(s, t)$ denotes the number of shortest paths from s to t, and $\sigma(s, t|v)$ the number of such paths P that contain v as an internal node.

In our work, we naturally extend this definition to betweenness centrality in an uncertain graph $\mathcal{G} = (V, E, W, p)$ for most probable shortest paths by replacing $\sigma(s, t)$ with $|\mathcal{M}(\mathcal{G}, s, t)|$ and $\sigma(s, t|v)$ with $|\mathcal{M}(\mathcal{G}, s, t|v)|$, where $\mathcal{M}(\mathcal{G}, s, t|v)$ consists of the paths $P \in \mathcal{M}(\mathcal{G}, s, t)$ that have v as an internal node.

Definition 1 (MPSP-Betweenness Centrality). *In an uncertain graph $\mathcal{G} = (V, E, W, p)$, we define the betweenness centrality of a node $v \in V$ based on most probable shortest paths as*

$$b_{\mathcal{G}}(v) = \frac{1}{|V|(|V|-1)} \sum_{\substack{(s,t) \in V \times V \\ s \neq v \neq t, \mathcal{M}(\mathcal{G},s,t) \neq \emptyset}} \frac{|\mathcal{M}(\mathcal{G}, s, t|v)|}{|\mathcal{M}(\mathcal{G}, s, t)|} \tag{14}$$

A different definition of betweenness centrality for uncertain graphs is given in [49, 60] and it is referred to as *expected betweenness centrality*. The expected betweenness of a node is the weighted average of its betweenness over all possible worlds.

$$\mathbb{E}_{G \sim \mathcal{G}}[b_G(v)] = \sum_{G \sqsubseteq \mathcal{G}} Pr(G) \times b_G(v) \tag{15}$$

Either of these notions can be meaningful, depending on the application. For instance, the notion of expected centrality is worth studying when the application concerns broadcasting of a message from one node to another, in which the message can be propagated over different possible paths. On the other hand, the notion of MPSP-Betweenness Centrality gives a more accurate picture when the application concerns routing or route recommendation, in which the path(s) need to be fixed beforehand and we can only use a single path to go from the origin to the destination.

Another notion of betweenness centrality is based on *possible shortest paths* [60] and it is called PSP-Betweenness Centrality.

In our experiments in § 5.9, we see that these different notions of betweenness yield slightly different rankings when ordering the nodes based on their betweenness values. Moreover, exploiting the results in § 4.1, we are able to compute the MPSP-Betweenness Centrality much faster than the expected and PSP-betweenness.

4.1 Efficient s-t Pair Sampling

The naive method of computing the MPSP-Betweenness Centrality of a node by considering all the s-t pairs and then computing the MPSPs is infeasible for large uncertain graphs. Moreover, designing an efficient algorithm for this task is challenging in our setting. As observed in § 2.1, in uncertain graphs, a sub-path of an MPSP is not necessarily an MPSP. Therefore, we cannot decompose a shortest path into two smaller shortest sub-paths or concatenate two shortest sub-paths to get a larger shortest path. For these reasons, we can neither apply optimization techniques such as those exploited in Brandes' algorithm [10], nor apply techniques based on node sampling where a small set of nodes is sampled and their contributions to the betweenness centralities are accumulated to estimate the betweenness of other nodes [4, 11, 21].

Therefore, we design a novel algorithm based on efficient s-t path sampling instead of node sampling. In the following, for simplicity we assume that there is only one MPSP for every pair of nodes. Thanks to this assumption, choosing an MPSP uniformly at random is equivalent to finding the unique MPSP between them using Algorithm 1. However, if there are multiple MPSPs for a pair of nodes, we can identify all of them using our top-k approach in § 3.2, and then select one among them uniformly at random.

Our proposed method, whose pseudocode is shown in Algorithm 4, samples r s-t pairs, for each of which it computes the MPSP P and then increments the betweenness centrality of every internal node of P by $\frac{1}{r}$. The main question that now arises is: *How many samples are needed to produce a very accurate estimate of the betweenness centrality of every node with a high probability?* In the remainder of this section, we provide an answer to this question. Specifically, given $\epsilon, \delta > 0$, we find a lower bound on the number of samples r so that, with probability at least $1 - \delta$, the difference between the approximate and the exact centrality of every node is at most ϵ.

Algorithm 4 Approximating MPSP-Betweenness-Centrality

Input: Uncertain graph $\mathcal{G} = (V, E, W, p)$, number of samples r, positive integers m and N.
Output: $\widehat{b_{\mathcal{G}}} : V \rightarrow \mathbb{R}$.

```
1: b̂_G(v) ← 0 ∀v ∈ V.
2: for i = 1 to r do
3:     Sample distinct nodes s and t
4:     P ← Alg. 1 (G, s, t, m, N)
5:     for all v ∈ Int(P) do
6:         b̂_G(v) ← b̂_G(v) + 1/r
7:     end for
8: end for
9: return b̂_G
```

Following the ideas in [43, Proof of Lemma 1], we can obtain the following lower bound on the required number of samples (proof omitted owing to space limits).

Theorem 6. *Given an uncertain graph $\mathcal{G} = (V, E, W, p)$ and $\epsilon, \delta > 0$, assuming that Algorithm 1 returns the correct MPSP and that there is a unique MPSP between every pair of nodes, the output of Algorithm 4 when using $r \geq \frac{1}{2\epsilon^2} \ln \frac{2|V|}{\delta}$ samples satisfies*

$$\Pr\left(\left|\widehat{b_{\mathcal{G}}}(v) - b_{\mathcal{G}}(v)\right| < \epsilon \ \forall v \in V\right) > 1 - \delta$$

Computational Complexity. The space and time complexities of Algorithm 4 are dominated by those of Algorithm 1 (line 4 of Algorithm 4). Hence, it follows from § 3.1 that the complexities are: $O(m|E| + |V|)$ and $O(rm(|E| + |V| \log |V| + \log m))$, respectively.

Parallel Implementation. In Algorithm 4, the computations performed on the r sampled s-t pairs are independent of each other. Hence, these computations can be implemented in parallel, e.g., via multiple threads. We experimentally demonstrate the effect of the number of threads on the running time of our algorithm in § 5.9.

5 EXPERIMENTAL RESULTS

We assess the efficiency and effectiveness of our proposal, and compare it against previous work [12, 63] on synthetic (§ 5.2) and road networks (§ 5.3). We also analyze the effect of each phase of our method on the performance (§ 5.4), parameter sensitivity (§ 5.5), and single-source and single-target queries (§ 5.6). Finally, we present use cases on sensor (§ 5.7) and brain networks (§ 5.8), and application to network centrality (§ 5.9).

5.1 Experimental Setup

Experiments are conducted on a single core (except when testing our parallel implementation) of a server with a 3.7 GHz Xeon processor and 256 GB RAM. Our C++ code is available in [55].

Queries. For each uncertain graph, we generate four categories of source-target pairs as queries. The first three categories constitute randomly chosen node pairs that are 2, 4, and 6 hops away. The last category comprises pairs of randomly chosen connected nodes. The result for each category is an average over 100 s-t pairs.

Parameters. • **# Dijkstra+MC-runs in Phase 1 (m)**: A small m is sufficient for our purpose (§ 3.3). We vary $m \in \{5, 10, 20, 50, 100\}$, with the default value 20. • **# MC-samples in Phase 2 (N)**: We vary $N \in \{10^1, 10^2, 10^3, 10^4, 10^5\}$, with the default value 10^3. • **Top-k MPSPs**: We vary $k \in \{1, 5, 10\}$, with the default value 1.

Methods Compared. We employ the filtering-and-verification based method [63] as the baseline (§ 2.2). Furthermore, in §5.4 we compare against [12] that used Dijkstra+MC (first phase of our method) to compute the probability of being the shortest path heuristically, without any accuracy guarantee.

5.2 Results on Synthetic Networks

We generate synthetic, uncertain (directed) graphs according to two classic models. (i) The *Erdos-Renyi* (ER) model [18] generates a random graph with $|V|$ nodes and $|E|$ directed edges chosen uniformly at random from $|V|(|V|-1)$ possible edges; (ii) The *Barabasi-Albert* (BA) model [6] generates a graph with $|V|$ nodes and $|E|$ edges satisfying a power law (in)degree distribution. Starting with a single node and no edge, a new node is added in every time step along with $|E|/|V|$ edges directed from the new node to an existing node, such that the probability of choosing an existing node i (as target), with its current in-degree d_i, is proportional to d_i.

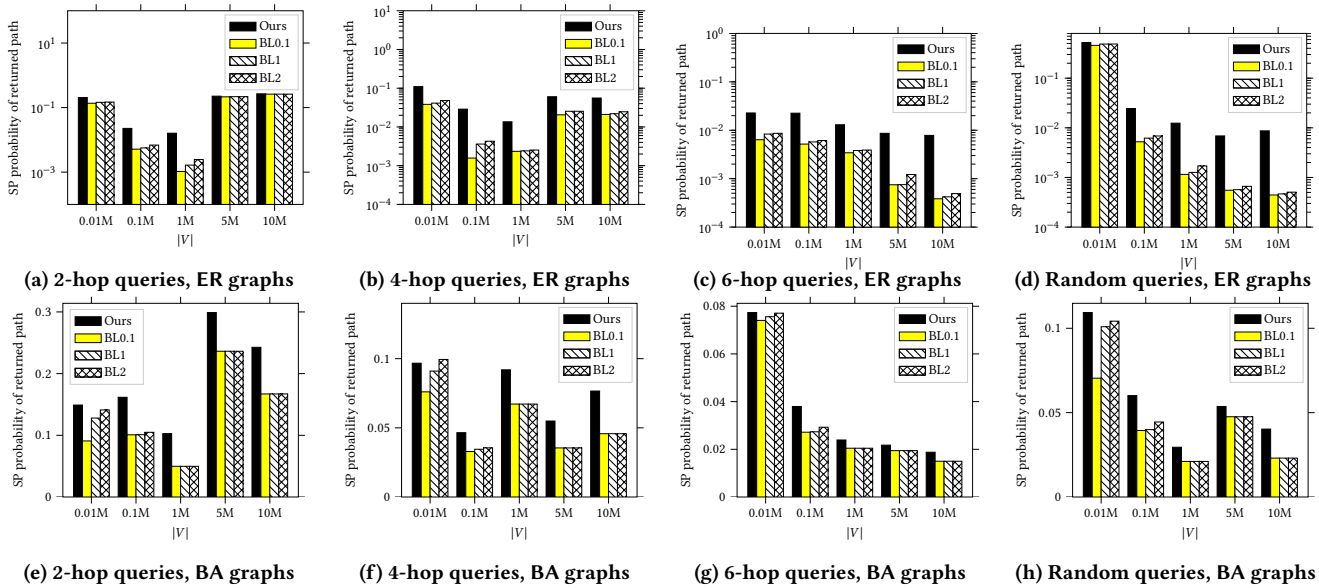
For both models, we vary $|V|$ in {0.01M, 0.1M, 1M, 5M, 10M}, and for every value of $|V|$, we vary the value of $|E|/|V|$ in {2, 6, 10}. In each synthetic graph, the probability of every edge is a uniform random real number in the interval $(0, 1]$, and the length of every edge is a uniform random integer in the interval $(0, 1000]$.

Figure 3 reports the comparison of quality (expressed as the probability of the returned path being a shortest path) against the baseline method [63]. Given that the candidate generation phase (§ 2.2) of [63] does not finish in one hour over our synthetic datasets, to make the comparison feasible, we place an upper limit on the candidate generation time of [63]. Notice that increasing this time limit leads to more candidate paths, hence the possibility of higher-quality returned paths is also increased. However, once an MPSP is included in the candidate set, increasing the time threshold further would not lead to better-quality solutions.

Following this observation, if T denotes the candidate generation time of our method for a given query, we compare the effectiveness of our algorithm against three variants of the baseline, when we terminate the baseline's candidate generation at time cT, with $c \in \{0.1, 1, 2\}$. We denote these three sets of baselines as $BL0.1$, $BL1$, $BL2$, as shown in Figure 3. Intuitively, $BL2$ could result in higher-quality returned paths compared to those via $BL0.1$ and $BL1$, however at the cost of higher running time, i.e., about 2 times higher running time than $BL1$ and 20 times more than $BL0.1$. $BL2$ is also about 2 times more time consuming than ours.

Quality results in Figure 3 show that *in most of the cases our method outperforms all variants of the baseline*. For 6-hop and random queries over larger ER graphs, the SP probability returned by our solution is up to one order of magnitude better than those returned by the baselines (Figures 3 (c, d)).

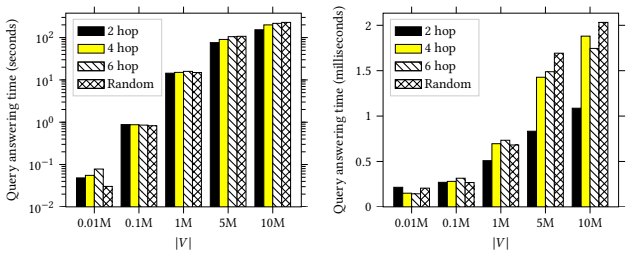
We show the efficiency of our method in Figure 4 for different query categories. Since the time limit of the baseline is set by us, we do not compare its running time with that of ours. We observe that our running times are less sensitive to different query categories. However, the running times on ER graphs are some orders of magnitude larger than those on BA graphs. This can be explained based on how these graphs are constructed: each node in BA graphs has out-degree at most 10. On the other hand, in ER graphs, there are several nodes with out-degrees more than 15~20. This implies that Dijkstra's algorithm visits higher out-degree nodes a lot more in ER graphs, requiring longer running times.



(a) 2-hop queries, ER graphs (b) 4-hop queries, ER graphs (c) 6-hop queries, ER graphs (d) Random queries, ER graphs

(e) 2-hop queries, BA graphs (f) 4-hop queries, BA graphs (g) 6-hop queries, BA graphs (h) Random queries, BA graphs

Figure 3: Comparison of quality (probability $Pr(\mathrm{Sh}_s^t(P))$ of the returned path P being a shortest path) on synthetic graphs with $|E|/|V| = 10$.



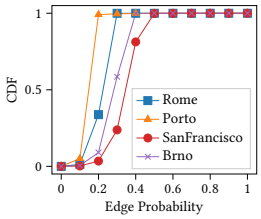
(a) ER graphs (b) BA graphs

Figure 4: Running time on synthetic graphs with $|E|/|V| = 10$.

5.3 Results on Road Networks

We construct uncertain (directed) graphs from four real-world road networks obtained via OpenStreetMap [47], along with recorded taxi trajectory data for each network (see the table in Figure 5). The nodes denote locations, while the edges denote road segments. The length of an edge is measured as its spatial length. We *map-match* every trajectory to the corresponding map using the open-source software OSRM [41], thus obtaining the road segments involved in each trajectory along with the speed on each segment. However, there are road segments in each network which are not traversed by any trajectory. We synthetically assign a speed to each such segment following [14], by sampling from a normal distribution with mean equal to the speed limit on that segment and standard deviation equal to a quarter of the mean. Since commuters are more likely to prefer those roads on which they can travel at a higher speed, we assign the probability of an edge (road segment) proportional to its average speed across all trajectories. The number of nodes, edges, and distribution of the edge probabilities in the resultant graphs are shown in Figure 5.

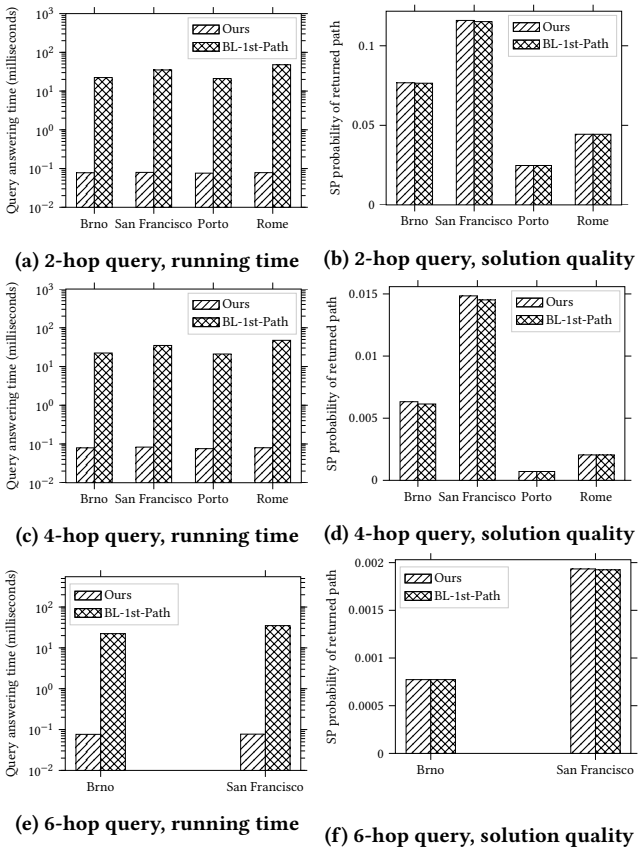
In our experiments on road networks, varying time thresholds for the baseline does not result in any quality difference. This is because the road networks are sparse, and the MPSP is often the shortest path in the certain (deterministic) version of the network. Hence, we terminate the baseline's candidate generation as soon as only the first s-t path is obtained, which is essentially the shortest s-t path considering the deterministic version of the network. We refer to this variant of the baseline as *BL-1st-Path*.



City	$\|V\|$	$\|E\|$	Trajectory
Brno	1.9M	4.0M	[53]
Porto	1.8M	3.7M	[45]
Rome	4.0M	8.0M	[3]
SF	3.0M	6.2M	[50]

Figure 5: Properties of road networks: SF denotes San Francisco.

Figures 6 (b, d, f) compare the quality of our method against the baseline. Both methods return similar results in terms of quality. As stated earlier, the returned path (by both methods) for almost every query is also the shortest path in the certain version of the graph. Notice that the entries in the 6-hop query category are vacant for the *Porto* and *Rome* road networks. This implies that, for these graphs, running Dijkstra+MC for queries in the 6-hop category resulted in an empty path. This can be attributed to the fact that the edge probabilities of these graphs are smaller compared to those of the other graphs, which is evident from Figure 5. In general, due to the sparseness of road networks and relatively smaller edge probabilities, MPSP queries are more meaningful here for nearby s-t pairs (e.g., find the MPSP to the nearest gas station or restaurant).



(a) 2-hop query, running time (b) 2-hop query, solution quality

(c) 4-hop query, running time (d) 4-hop query, solution quality

(e) 6-hop query, running time (f) 6-hop query, solution quality

Figure 6: Results on road networks

On the other hand, our method takes *up to 2∼3 orders of magnitude less time than the baseline*, as shown in Figures 6 (a, c, e). This is because the baseline approach essentially uses Dijkstra's algorithm on the certain version of the graph to retrieve the shortest path, which has to visit every node closer to the source than the target. In contrast, Dijkstra+MC in our candidate generation may end up not visiting many nodes since the corresponding edges are not sampled.

5.4 Effect of Each Phase on the Performance

Our method (§3.1) consists of two phases: Dijkstra+MC for efficient candidate path generation (Phase 1), followed by the Luby-Karp algorithm to select the MPSP among them (Phase 2). We analyze the benefits of both phases by comparing the two-phased method against a method based only on Phase 1, followed by a selection by majority, i.e., the path that has been sampled most times by Dijkstra+MC is returned as the candidate MPSP.

Table 1 shows that the two-phased method *never produces worse-quality results, and can return better MPSPs for up to 59% of the queries.* To understand this result, assume that there are two s-t paths P_1 and P_2 such that the SP probability of P_1 is slightly higher than that of P_2. Then, it could happen that P_2 is sampled a larger number of times (i.e., with a higher frequency) than P_1, due to randomness of the Dijkstra+MC sampling. Then, according to majority, the estimate for P_2 is higher than that of P_1. However, the Luby-Karp algorithm in our second phase does not care about the sampling frequency at all; it only needs to know if P_1 and P_2 are present in the sampled candidate set (at least once), thereby reporting the correct MPSP. These results demonstrate the usefulness of Phase 2.

Table 1: Percentage of queries for which our method finds better MPSPs compared to (a) only Phase 1 of our method (Dijkstra+MC) followed by selection via majority, and (b) Phase 1 of our method followed by HT-estimator. ER graph with $|V| = 10^4$, $|E| = 10^5$.

Query type	% of queries our method finds better MPSPs	
	vs. Phase 1 + Majority	vs. Phase 1 + HT-estimator
2-hop	36%	12%
4-hop	59%	5%
Random	11%	6%

We also compare with the case in which Phase 1 is augmented with an unequal probability estimator, e.g., Horvitz-Thompson (HT) inspired by [12]. Recall (§ 1.1) that [12] deals with a different problem (i.e., threshold-based shortest-path queries) and adopts a different uncertain data model. However, their heuristic approach can be adapted for our purposes. Although the HT-estimator is useful in reducing the variance of Dijkstra+MC sampling, the Luby-Karp algorithm in our second phase still outperforms it, for the reason stated above. In particular, *our method never produces worse results and it produces better MPSPs for up to 12% of the queries.*

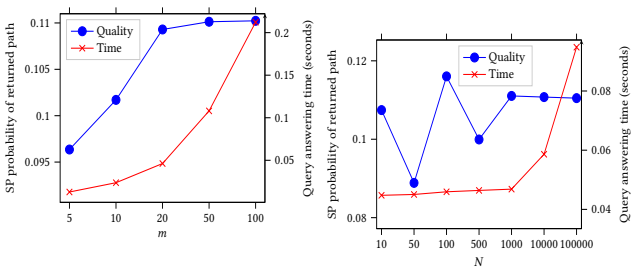
5.5 Parameter Sensitivity Analysis

Impact of m and N. We vary the number $m \in \{5, 10, 20, 50, 100\}$ of Dijkstra+MC runs (Phase 1), and the number of MC samples $N \in \{10^1, 10^2, 10^3, 10^4, 10^5\}$ for the Luby-Karp algorithm (Phase 2). The results are shown in Figure 7. Owing to space constraints, we only show the results of 4-hop queries on the ER graph with $|V| = 10^4$ and $|E| = 10^5$. For Dijkstra+MC, we observe that increasing m till its default value ($m = 20$) steadily increases the SP probabilities of returned paths. This indicates that we need about $m = 20$ runs of Dijkstra+MC to include the MPSP in the candidate set. For the Luby-Karp algorithm, on the other hand, increasing N till its default value ($N = 10^3$) shows fluctuation of the SP probabilities returned, implying that the sampling method has not converged yet. The returned SP probabilities stabilize around these default parameter values. We further notice that increasing these parameter values beyond their default values of $m = 20$ (resp. $N = 10^3$) returns SP probabilities of paths (resp. Luby-Karp estimates) having nearly the same value, but the running time is significantly increased. This justifies the selection of our default parameter values.

Top-k MPSPs. We find the top-k MPSPs with $k \in \{1, 5, 10\}$. The results for $k = 1$ have already been shown in Figures 3 and 4. For $k \in \{5, 10\}$, the running times are nearly the same as with $k = 1$; thus, we only show the SP probability of our solution (averaged over the k paths returned for each query) in Figure 8. Notice that our algorithm returns better top-k paths compared to the baseline.

5.6 Single-Source and Single-Target Queries

Figure 9 (left) shows running times of single-source multi-target queries (§ 3.4) on ER graphs. The y-axis is logarithmic and the query answering time is the aggregated time required for both phases 1 and 2. Notice that the running time of Phase 2 is much higher than that of Phase 1. Moreover, the running time of Phase 1 is increased



(a) Variation of m ($N = 10^3$) (b) Variation of N ($m = 20$)

Figure 7: Running times and quality of returned paths; ER graph with $|V| = 10^4$, $|E| = 10^5$; 4-hop queries.

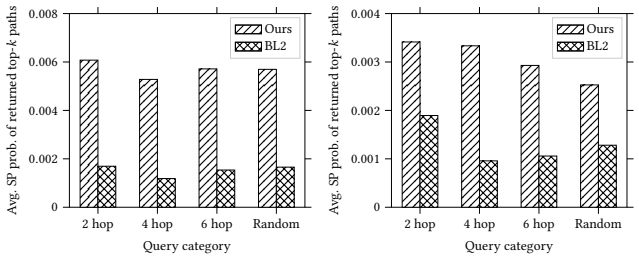


Figure 8: Quality of our solution for the top-k MPSPs; ER graph with $|V| = 10^5$, $|E| = 10^6$, $k = 5$ (left) and $k = 10$ (right).

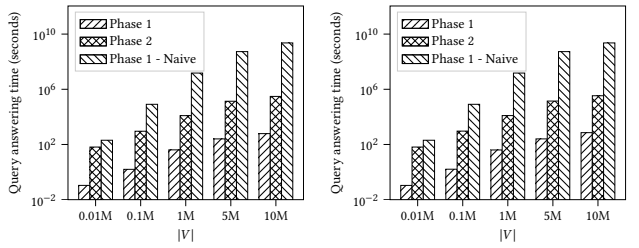


Figure 9: Running time of single-source (left) and single-target (right) queries; ER graphs with $|E|/|V| = 10$

by a small factor with increase in the number of nodes, because Dijkstra is not run separately for individual target nodes. Our Phase 1 is several orders of magnitude faster than *Phase 1-Naive*, which is running Phase 1 separately for each target. Figure 9 (right) shows similar improved efficiency for multi-source single-target queries.

5.7 Case Studies: Sensor Network

Intel Lab Data [42] is a collection of sensor communication data with 54 sensors deployed in the Intel Berkeley Research Lab between February 28 and April 5, 2004. The probabilities on (directed) edges denote the percentages of messages from a sender successfully reached to a receiver. The edge length is the spatial distance (in metres) between the co-ordinates of the two sensors.

We show MPSPs from node 48 to 22 in Figure 10. We observe that the MPSP is the *sixth* shortest path in the certain version of the graph. The first few shortest paths have smaller probabilities of existence, showcasing the usefulness of MPSPs in uncertain graphs.

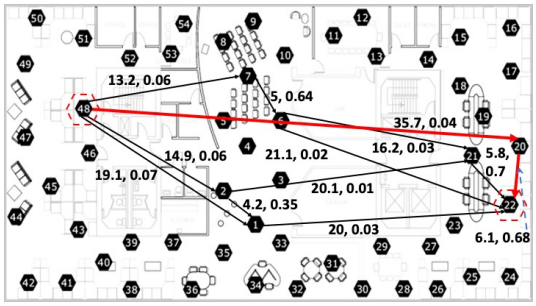


Figure 10: Case studies on sensor network. Paths from node 48 to node 22 in the sensor network. The node sequences of the top 6 shortest paths (in ascending order of length) are $(48, 1, 22)$, $(48, 2, 1, 22)$, $(48, 7, 6, 22)$, $(48, 7, 6, 21, 22)$, $(48, 2, 21, 22)$, $(48, 20, 22)$. The 6^{th} shortest path, shown in red, is the MPSP.

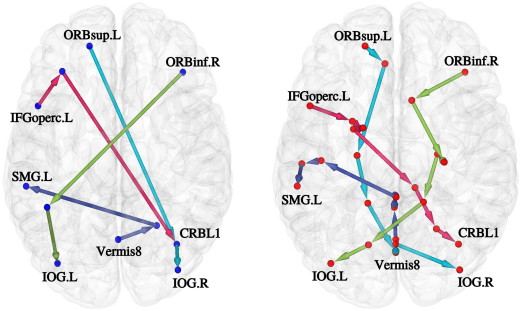


Figure 11: MPSPs for TD group (left) and ASD group (right) of brain networks. Each of 4 MPSPs is represented by edges of same colour.

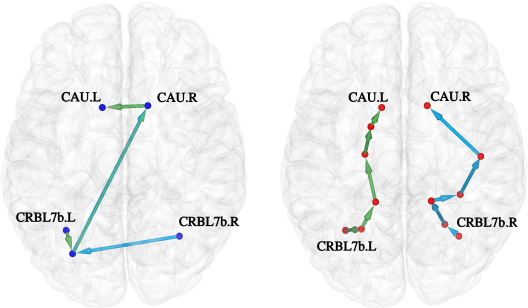


Figure 12: MPSPs for TD group (left) and ASD group (right) of brain networks. Each of 2 MPSPs is represented by edges of same colour.

5.8 Case Studies: Brain Networks

A brain network can be defined as a weighted uncertain graph, where nodes are brain regions of interest (ROIs), (bi-directed) edges indicate co-activation between ROIs, edge distance represents physical distance between ROIs, and edge probability indicates the strength of the co-activation signal (i.e., the pairwise Pearson correlation between the time series of each pair of ROIs). We use a publicly available dataset from the Autism Brain Imaging Data Exchange (ABIDE) project [15]. The dataset contains data of 52 *Typically Developed* (TD) children and 49 children suffering from *Autism Spectrum Disorder* (ASD) whose age is at most 9 years [36, 37, 44, 58]: each subject corresponds to a graph over 116 nodes (ROIs).

$\mathcal{G}_{ASD}$ and $\mathcal{G}_{TD}$ are weighted uncertain graphs, defined over the same set of nodes as the original graphs, while the weight and probability of each edge are the averages of the respective values of the same edge across all graphs in the ASD and TD groups.

In Figures 11 and 12, we show the MPSPs for 6 s-t pairs of both $\mathcal{G}_{TD}$ (left) and $\mathcal{G}_{ASD}$ (right). Consider the pink path in Figure 11 from the *inferior frontal gyrus, opercular part* (IFGoperc.L) to the *cerebellum* (CRBL1). The MPSP in $\mathcal{G}_{TD}$ is a path with 2 hops over a longer distance, compared to that in $\mathcal{G}_{ASD}$ with 6 shorter hops. This is consistent with the results of different works in neuroscience [16, 46] indicating that ASD is characterized by underconnectivity between distant brain regions and overconnectivity between closer ones. Moreover, children with ASD have brains that are overly connected compared to typically developed children [20, 32, 56]. In addition, the hemispheres in ASD group are more symmetrical than those of the TD group [51]. We highlight this in Figure 12: the MPSPs in the left and right cerebral hemispheres of the brain are indeed more similar and symmetrical in children with autism, while in the TD group the paths can cross the hemispheres and also span the same regions. Our consistent findings underline the importance of MPSPs in uncertain graphs.

5.9 Application: Network Centrality

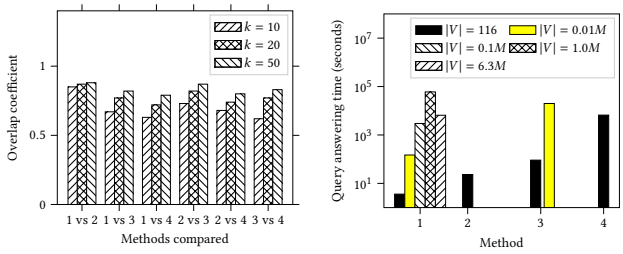
We compare the top-k most central nodes according to four centrality computation methods (introduced in § 4): (1) MPSP-Betweenness Centrality with sampled s-t pairs, (2) MPSP-Betweenness Centrality with all s-t pairs, (3) expected betweenness centrality [49, 60] (by sampling possible worlds and using [54] for every sampled world) , and (4) PSP-Betweenness Centrality [60].

We first run these methods on six different brain graphs (randomly selected from 52 TD brains), each with 116 nodes. Following [54], we set $\epsilon = 0.05$ and $\delta = 0.1$ for all the methods when required. For every method, we compute the betweenness centrality of all nodes and rank them in descending order of centrality. Given a value of $k \in \{10, 20, 50\}$, for each of the $\binom{4}{2} = 6$ possible pairs of methods, we compare the similarity of the sets of top-k nodes returned by both methods using the overlap coefficient. The overlap coefficient of two sets A and B, each of size k, is defined as $\frac{|A \cap B|}{k}$.

We report these results averaged over six graphs in Figure 13(a). For every value of k, methods 1 and 2 (both of which deal with MPSP-Betweenness-Centrality) produce very similar results showing that *our sampling based method yields good approximation.* The overlap with other methods is a bit lower indicating that there is a slight difference in the top-k nodes produced by each method.

Next, to assess the efficiency and scalability of our method, we compute the centrality ranking for the six brain graphs ($|V| = 116$), a Twitter graph ($|V| = 6.3$M, $|E| = 11.1$M), and the ER graphs with $|V| \in \{0.01M, 0.1M, 1M\}$ and $|E| = 10|V|$. Twitter [38] is a social network where users post new tweets or retweet those of other users. This data is used to construct a directed graph in which nodes are users and edges are retweets. Each edge has weight one, and the probability is given by $1 - \exp(-t/\mu)$, where t is the number of retweets between the corresponding users. We set $\mu = 10$.

The sequential running times are shown in Figure 13(b). A missing bar means that the run did not terminate within a day. It turns out that *only our method (1) terminates within a reasonable time for all graphs.* Notice that even for the $1M$ node graph, our method finishes within 17 hours. Although the Twitter graph (6.3M nodes) is larger than the ER graph with 1M nodes, the running time on Twitter is less than that on ER, because the former is more sparse.



(a) Top-k central node similarity **(b) Running time (sequential)**

(c) Parallelization: Our method (method 1)'s running time

# Threads	Twitter	ER, $\|V\| = 10$M, $\|E\| = 100$M
1	6 520.65 sec	$>$ 2 days
10	930.98 sec	$>$ 2 days
20	795.91 sec	125 603.60 sec
40	666.76 sec	61 668.80 sec

Figure 13: Centrality results; 4 methods are described in § 5.9

Finally, we run the parallel implementation of our method, i.e., method 1, on our two largest graphs: Twitter ($|V| = 6.3$M, $|E| = 11.1$M) and the ER graph with $|V| = 10$M and $|E| = 100$M. All 40 cores of the server are used and up to 40 threads are employed for parallelization via POSIX threads. Figure 13(c) shows that increasing the number of threads leads to shorter running times. *With 40 threads, centrality computation on Twitter ($|V| = 6.3M$, $|E| = 11.1M$) requires only 11 minutes, and on ER ($|V| = 10M$, $|E| = 100M$), it finishes in 18 hours. These results demonstrate good parallelizability and scalability of our algorithm over large graphs.*

6 CONCLUSIONS

In this paper, we investigated the problem of finding the Most Probable Shortest Path (MPSP) between two nodes in an uncertain graph. We proved that the problem is #**P**-hard, and also derived some other properties of MPSPs that make our problem challenging. Our proposed solution proceeds in two phases: efficient and effective sampling of some candidate paths using Dijkstra+MC, followed by approximating the SP probability of each candidate path using the Luby-Karp algorithm. We provided theoretical quality guarantees of our algorithm, as well as extended it to find the top-k MPSPs. We also illustrated an application of our algorithm by defining and efficiently computing a new concept of betweenness centrality in an uncertain graph. The experimental results validate the effectiveness, efficiency, and scalability of our methods, and rich real-world case studies demonstrate the usefulness of our problem.

ACKNOWLEDGMENTS

Arijit Khan is funded by MOE grants RG117/19 and MOE2019-T2-2-042. Ruben Brokkelkamp is supported by the Netherlands Organisation for Scientific Research (NWO) through Gravitation-grant NETWORKS-024.002.003.

REFERENCES

[1] Eytan Adar and Christopher Re. 2007. Managing uncertainty in social networks. *IEEE Data Engineering Bulletin* 30, 2 (2007), 15–22.

[2] Charu C. Aggarwal. 2009. *Managing and Mining Uncertain Data*. Advances in Database Systems, Vol. 35. Springer, Boston, MA, USA.

[3] Raul Amici, Marco Bonola, Lorenzo Bracciale, Antonello Rabuffi, Pierpaolo Loreti, and Giuseppe Bianchi. 2014. Performance assessment of an epidemic protocol in VANET using real traces. In *MoWNet*. Elsevier, 92–99.

[4] David A. Bader, Shiva Kintali, Kamesh Madduri, and Milena Mihail. 2007. Approximating betweenness centrality. In *Algorithms and Models for the Web-Graph*, Anthony Bonato and Fan R. K. Chung (Eds.). Springer, Berlin, Heidelberg, 124–137.

[5] Michael O. Ball. 1986. Computational complexity of network reliability analysis: An overview. *IEEE Transactions on Reliability* 35, 3 (1986), 230–239.

[6] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.

[7] Paolo Boldi, Francesco Bonchi, Aristides Gionis, and Tamir Tassa. 2012. Injecting uncertainty in graphs for identity obfuscation. *Proceedings of the VLDB Endowment (PVLDB)* 5, 11 (July 2012), 1376–1387. https://doi.org/10.14778/2350229.2350254

[8] Francesco Bonchi, Aristides Gionis, Francesco Gullo, and Antti Ukkonen. 2014. Distance oracles in edge-labeled graphs. In *Proceedings of the 17th International Conference on Extending Database Technology (EDBT 2014)*. 547–558.

[9] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. 2014. Maximizing social influence in nearly optimal time. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2014)*. SIAM, 946–957.

[10] Ulrik Brandes. 2001. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology* 25, 2 (2001), 163–177.

[11] M. H. Chehreghani. 2014. An efficient algorithm for approximate betweenness centrality computation. *Comput. J.* 57, 9 (2014), 1371–1382.

[12] Yurong Cheng, Ye Yuan, Guoren Wang, Baiyou Qiao, and Zhiqiong Wang. 2014. Efficient sampling methods for shortest path query over uncertain graphs. In *International Conference on Database Systems for Advanced Applications (DASFAA 2014)*. Springer, Cham, 124–140.

[13] Yu-Rong Cheng, Ye Yuan, Lei Chen, and Guo-Ren Wang. 2015. Threshold-based shortest path query over large correlated uncertain graphs. *Journal of Computer Science and Technology* 30, 4 (2015), 762–780.

[14] Camila F. Costa, Mario A. Nascimento, José A.F. Macêdo, Yannis Theodoridis, Nikos Pelekis, and Javam Machado. 2015. Optimal time-dependent sequenced route queries in road networks. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*. Association for Computing Machinery, New York, NY, USA, 1–4.

[15] Cameron Craddock, Yassine Benhajali, Carlton Chu, Francois Chouinard, Alan Evans, András Jakab, Budhachandra Singh Khundrakpam, John David Lewis, Qingyang Li, Michael Milham, Chaogan Yan, and Pierre Bellec. 2013. The neuro bureau preprocessing initiative: Open sharing of preprocessed neuroimaging data and derivatives. *Frontiers in Neuroinformatics* (2013). https://doi.org/10.3389/conf.fninf.2013.09.00041

[16] Adriana Di Martino, Clare Kelly, Rebecca Grzadzinski, Xi-Nian Zuo, Maarten Mennes, María Mairena, Catherine Lord, Francisco Castellanos, and Michael Milham. 2010. Aberrant striatal functional connectivity in children with autism. *Biological Psychiatry* 69, 9 (12 2010), 847–56. https://doi.org/10.1016/j.biopsych.2010.10.029

[17] David Eppstein. 1998. Finding the k shortest paths. *SIAM J. Comput.* 28, 2 (1998), 652–673.

[18] Paul Erdős and Alfréd Rényi. 1959. On random graphs. *Publicationes Mathematicae Debrecen* 6 (1959), 290–297.

[19] Linton C. Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* 40, 1 (1977), 35–41.

[20] Luis García Domínguez, Jim Stieben, José Luis Pérez Velázquez, and Stuart Shanker. 2013. The imaginary part of coherency in autism: Differences in cortical functional connectivity in preschool children. *PLOS ONE* 8, 10 (10 2013), 1–13. https://doi.org/10.1371/journal.pone.0075941

[21] Robert Geisberger, Peter Sanders, and Dominik Schultes. 2008. Better approximation of betweenness centrality. In *Proceedings of the Meeting on Algorithm Engineering and Expermiments* (San Francisco, California). Society for Industrial and Applied Mathematics, USA, 90–100.

[22] Joy Ghosh, Hung Q. Ngo, Seokhoon Yoon, and Chunming Qiao. 2007. On a routing problem within probabilistic graphs and its application to intermittently connected networks. In *IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications*. IEEE, 1721–1729.

[23] Kai Han, Fei Gui, Xiaokui Xiao, Jing Tang, Yuntian He, Zongmai Cao, and He Huang. 2019. Efficient and effective algorithms for clustering uncertain graphs. *Proceedings of the VLDB Endowment (PVLDB)* 12, 6 (2019), 667–680.

[24] Jiafeng Hu, Reynold Cheng, Zhipeng Huang, Yixang Fang, and Siqiang Luo. 2017. On embedding uncertain graphs. In *Proceedings of the 2017 ACM International Conference on Information and Knowledge Management (CIKM 2017)*. Association for Computing Machinery, New York, NY, USA, 157–166.

[25] Ming Hua and Jian Pei. 2010. Probabilistic path queries in road networks: Traffic uncertainty aware path selection. In *Proceedings of the 13th International Conference on Extending Database Technology*. 347–358.

[26] Ruoming Jin, Lin Liu, Bolin Ding, and Haixun Wang. 2011. Distance-constraint reachability computation in uncertain graphs. *Proceedings of the VLDB Endowment (PVLDB)* 4, 9 (2011), 551–562.

[27] Donald B. Johnson. 1977. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM (JACM)* 24, 1 (1977), 1–13.

[28] Richard M. Karp and Michael Luby. 1983. Monte-Carlo algorithms for enumeration and reliability problems. In *24th Annual Symposium on Foundations of Computer Science (SFCS 1983)*. IEEE, 56–64.

[29] Richard M. Karp, Michael Luby, and Neal Madras. 1989. Monte-Carlo approximation algorithms for enumeration problems. *Journal of Algorithms* 10, 3 (1989), 429–448.

[30] Xiangyu Ke, Arijit Khan, Mohammad Al Hasan, and Rojin Rezvansangsari. 2020. Reliability maximization in uncertain graphs. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[31] Xiangyu Ke, Arijit Khan, and Leroy Lim Hong Quan. 2019. An in-depth comparison of s-t reliability algorithms over uncertain graphs. *Proceedings of the VLDB Endowment (PVLDB)* 12, 8 (April 2019), 864–876. https://doi.org/10.14778/3324301.3324304

[32] Christopher Keown, Patricia Shih, Aarti Nair, Nick Peterson, Mark Mulvey, and Ralph-Axel Müller. 2013. Local functional overconnectivity in posterior brain regions is associated with symptom severity in autism spectrum disorders. *Cell Reports* 5 (11 2013), 567–572. https://doi.org/10.1016/j.celrep.2013.10.003

[33] Arijit Khan, Francesco Bonchi, Aristides Gionis, and Francesco Gullo. 2014. Fast reliability search in uncertain graphs.. In *Proceedings of the 17th International Conference on Extending Database Technology (EDBT 2014)*. 535–546.

[34] Arijit Khan, Francesco Bonchi, Francesco Gullo, and Andreas Nufer. 2018. Conditional reliability in uncertain graphs. *IEEE Transactions on Knowledge and Data Engineering* 30, 1 (2018), 2078–2092.

[35] Arijit Khan, Yuan Ye, and Lei Chen. 2018. *On Uncertain Graphs*. Synthesis Lectures on Data Management, Vol. 10. Morgan & Claypool Publishers, San Rafael, CA, USA.

[36] Sadamori Kojaku and Naoki Masuda. 2019. Constructing networks by filtering correlation matrices: A null model approach. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 475, 2231 (2019), 20190578. https://doi.org/10.1098/rspa.2019.0578

[37] Tommaso Lanciano, Francesco Bonchi, and Aristides Gionis. 2020. Explainable classification of brain networks via contrast subgraphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 11. https://doi.org/10.1145/3394486.3403383

[38] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[39] Xiaodong Li, Reynold Cheng, Yixiang Fang, Jiafeng Hu, and Silviu Maniu. 2018. Scalable evaluation of k-NN queries on large uncertain graphs. In *Proceedings of the 21st International Conference on Extending Database Technology (EDBT 2018)*. Vienna, Austria, 181–192. https://doi.org/10.5441/002/edbt.2018.17

[40] David Liben-Nowell and Jon Kleinberg. 2003. The link prediction problem for social networks. In *Proceedings of the Twelfth ACM International Conference on Information and Knowledge Management* (New Orleans, LA, USA) *(CIKM '03)*. Association for Computing Machinery, New York, NY, USA, 556–559. https://doi.org/10.1145/956863.956972

[41] Dennis Luxen and Christian Vetter. 2011. Real-time routing with OpenStreetMap data. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. Association for Computing Machinery, New York, NY, USA, 513–516.

[42] Samuel Madden. 2004. Intel lab data. http://db.csail.mit.edu/labdata/labdata.html

[43] Ahmad Mahmoody, Charalampos E Tsourakakis, and Eli Upfal. 2016. Scalable betweenness centrality maximization via sampling. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, New York, NY, USA, 1765–1773.

[44] Naoki Masuda, Sadamori Kojaku, and Yukie Sano. 2018. Configuration model for correlation matrices preserving the node strength. *Physical Review E* 98 (Jul 2018), 012312. Issue 1. https://doi.org/10.1103/PhysRevE.98.012312

[45] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* 14, 3 (2013), 1393–1402.

[46] Sarah Noonan, Frank Haist, and Ralph-Axel Müller. 2009. Aberrant functional connectivity in autism: Evidence from low-frequency BOLD signal fluctuations. *Brain Research* 1262 (02 2009), 48–63. https://doi.org/10.1016/j.brainres.2008.12.076

[47] OpenStreetMap contributors. 2020. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

[48] Panos Parchas, Francesco Gullo, Dimitris Papadias, and Franceseco Bonchi. 2014. The pursuit of a good possible world: Extracting representative instances of uncertain graphs. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 967–978.

[49] Joseph John Pfeiffer and Jennifer Neville. 2011. Methods to determine node centrality and clustering in graphs with uncertain structure. In *Fifth International AAAI Conference on Weblogs and Social Media*. 590–593.

[50] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. 2009. CRAWDAD dataset epfl/mobility (v. 2009-02-24). https://crawdad.org/epfl/mobility/20090224/cab.

[51] Merel Postema, Daan Van Rooij, Evdokia Anagnostou, Celso Arango, Guillaume Auzias, Marlene Behrmann, Geraldo Busatto, Sara Calderoni, Rosa Calvo, Eileen Daly, Christine Deruelle, Adriana Di Martino, Ilan Dinstein, Fabio Duran, Sarah Durston, Christine Ecker, Stefan Ehrlich, Damien Fair, Jennifer Fedor, and Clyde Francks. 2019. Altered structural brain asymmetry in autism spectrum disorder in a study of 54 datasets. *Nature Communications* 10 (12 2019). https://doi.org/10.1038/s41467-019-13005-8

[52] Michalis Potamias, Francesco Bonchi, Aristides Gionis, and George Kollios. 2010. K-nearest neighbors in uncertain graphs. *Proceedings of the VLDB Endowment (PVLDB)* 3, 1-2 (2010), 997–1008.

[53] Vit Ptošek, Lukáš Rapant, and Jan Martinovič. 2018. Floating car data collection for processing and benchmarking. https://doi.org/10.5281/zenodo.2250119

[54] Matteo Riondato and Evgenios M. Kornaropoulos. 2016. Fast approximation of betweenness centrality through sampling. *Data Mining and Knowledge Discovery* 30, 2 (2016), 438–475.

[55] Arkaprava Saha, Ruben Brokkelkamp, Yllka Velaj, Arijit Khan, and Francesco Bonchi. 2020. Shortest paths and centrality in uncertain networks: Code and data. https://github.com/ArkaSaha/MPSP-Centrality.

[56] Kaustubh Supekar, Lucina Q. Uddin, Amirah Khouzam, Jennifer Phillips, William D. Gaillard, Lauren E. Kenworthy, Benjamin E. Yerys, Chandan J. Vaidya, and Vinod Menon. 2013. Brain hyperconnectivity in children with autism and its links to social deficits. *Cell Reports* 5, 3 (2013), 738 – 747. https://doi.org/10.1016/j.celrep.2013.10.001

[57] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 75–86.

[58] Nathalie Tzourio-Mazoyer, Brigitte Landeau, Dimitri Papathanassiou, Fabrice Crivello, Olivier Etard, Nicolas Delcroix, Bernard Mazoyer, and Marc Joliot. 2002. Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain. *NeuroImage* 15, 1 (2002), 273 – 289. https://doi.org/10.1006/nimg.2001.0978

[59] Leslie G. Valiant. 1979. The complexity of enumeration and reliability problems. *SIAM J. Comput.* 8, 3 (1979), 410–421.

[60] Chenxu Wang and Ziyuan Lin. 2019. An efficient approximation of betweenness centrality for uncertain graphs. *IEEE Access* 7 (2019), 61259–61272.

[61] Jin Y. Yen. 1971. Finding the k shortest loopless paths in a network. *Management Science* 17, 11 (1971), 712–716.

[62] Ye Yuan, Lei Chen, and Guoren Wang. 2010. Efficiently answering probability threshold-based shortest path queries over uncertain graphs. In *International Conference on Database Systems for Advanced Applications (DASFAA 2010)*. Springer, Berlin, Heidelberg, 155–170.

[63] Lei Zou, Peng Peng, and Dongyan Zhao. 2011. Top-K possible shortest path query over a large uncertain graph. In *International Conference on Web Information Systems Engineering (WISE 2011)*. Springer, Berlin, Heidelberg, 72–86.