Estimate and Reduce Uncertainty in Uncertain Graphs

Naheed Anjum Arafat*, Ehsan Bonabi Mobaraki+, Arijit Khan+, Yllka Velaj#, Francesco Bonchi$

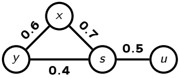
*Nanyang Technological University (Singapore),
+Aalborg University (Denmark),
#Univ. of Vienna (Austria),
$CENTAI (Italy)

DSAA 2024

Introduction

Uncertain Graph

An **Uncertain Graph** is a graph where every edge has an independent probability of existence (encapsulating real-world uncertainty).



Uncertain Graph (Edge Uncertainty)

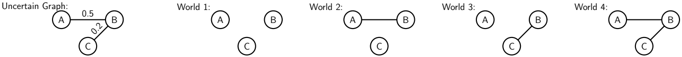
Examples

- **Sensor networks:** Edge probability encapsulates packet delivery probability via the corresponding link.
- **Protein-protein interaction networks:** Edge probability encapsulates the probability of interaction between two proteins established through noisy and error-prone experiments.
- **Social Networks:** Edge probability encapsulates the probability that some action by a node (u) will be adopted by another node (v) due to the corresponding link (u, v).
- Traffic networks, Knowledge bases constructed from diverse sources, etc.

Structural properties of Uncertain Graphs have uncertainty

Possible world semantics

An uncertain graph $\mathcal{G}$ with $|E| = m$ edges leads to 2^m possible worlds. Every possible world G has the probability $Pr(G) = \prod_{e \in E_G} p(e) \prod_{e' \in E \setminus E_G} (1 - p(e'))$



An uncertain graph and its possible worlds. Pr(W2) = 0.5 * (1-0.2) = 0.4

Structural properties: Reachability, #Triangles, Length of a shortest path, Node label
Distribution induced by a structural property:

- Prob(Reach(A,C) = 1) = Pr(W4), because C is reachable from A in only W4.
- Prob(Reach(A,C) = 0) = Pr(W1) + Pr(W2) + Pr(W3) = 1 - Pr(W4), because C is not reachable from A in W1, W2 and W3.

This distribution has uncertainty

Problem Statement & Contributions

1. How to measure the uncertainty induced by structural properties of an Uncertain graph?
    - We proposed to use **entropy** to measure uncertainty.
    - We proposed a **Monte-Carlo algorithm with theoretical guarantee** on the estimate of entropy.
2. Given a limited budget of k, how to select at most k uncertain edges updating whose probabilities maximally reduces uncertainty?

$$\arg\max_{E_1 \subseteq E, |E_1| \leq k} \left\{ \Delta H(E_1) = H(\mathbf{\Omega}) - H(\mathbf{\Omega'}) \right\} \tag{1}$$

where $\mathbf{\Omega}$ is the random variable indicating property values, e.g. $\mathbf{\Omega} \in \{0, 1\}$ for reachability. $E_1 \subseteq E$ indicates the uncertain edges whose probabilities are to be updated. $H(\mathbf{\Omega})$ and $H(\mathbf{\Omega'})$ indicates the entropy before and after edge probability updates.

- We proposed a **greedy** subgraph selection-based efficient **algorithm**.

Challenges

Hardness

- Uncertainty estimation and reduction of network properties are hard since, computing the underlying properties such as reliability, shortest path, triangle count, etc. over uncertain graphs themselves are #P-hard.
- The objective function for uncertainty reduction is not monotonic, neither submodular, nor supermodular. An exact approach for selecting the k-best edges has exponential time complexity.

Generality and adaptability Existing methods for uncertainty reduction work in a limited setting, e.g., for reliability query and crowdsourcing-based edge cleaning. They ignore other graph properties, ML model outputs, and diverse kinds of edge probability updates. We considered 2 types of updates:

1. $\mathcal{U}_1(p(e)) : (0,1) \rightarrow 1$: The resulting edge probability is known apriori.
2. $\mathcal{U}_2(p(e)) : (0,1) \rightarrow \{0,1\}$: The resulting edge probability is revealed only after the update (e.g., based on crowdsourcing results).

Measuring Uncertainty

Estimating $Pr(\mathbf{\Omega} = \Omega)$. The true probability distribution $Pr(\mathbf{\Omega} = \Omega)$ is approximated via MC-sampling using the sample distribution $\hat{P}r(\mathbf{\Omega})$.

$$\hat{P}r(\mathbf{\Omega} = \Omega) = \frac{\sum_{i=1}^{T} I(P(G_i) = \Omega)}{T} \quad (2)$$

where $\{G_1, G_2, \ldots, G_T\}$ are T possible worlds sampled via independent sampling of edges as per their probabilities. $I()$ is indicator function. $P(G_i)$ is the value of the graph structural property on possible world G_i.

Theorem: $\hat{P}r$ is unbiased

$\hat{P}r(.)$ is an unbiased estimator of $Pr(.)$

Estimating Entropy.

$$\hat{H} = -\sum_{\Omega \in \hat{Sup}(P, \mathcal{G})} \hat{P}r(\mathbf{\Omega} = \Omega) \log \hat{P}r(\mathbf{\Omega} = \Omega) \quad (3)$$

Compute N independent entropy estimates $\hat{H}_1, \hat{H}_2, \ldots, \hat{H}_N$, and return the average of those N estimates.

Generality: The algorithm works for any real-valued function P

Reducing Uncertainty

Naive algorithm Enumerate all subsets of E of size up to k, compute the updated entropy and select the subset whose update reduces the initial entropy maximally.

- Exact algorithm.
- Issues: Exponential (in $|E|$) time-complexity.

Greedy algorithm At every iteration, greedily select the edge that reduces the entropy maximally.

- Approximate entropy $H(\mathbf{\Omega}) - H(\mathbf{\Omega'})$ using MC sampling. Hence time-complexity is linear (in $|E|$).
- Cold-start problem: A locally-best solution at earlier rounds may lead to a globally sub-optimal solution.

Greedy+subgraph algorithm Rank **subgraphs of interest** based on the network function, update operation, and the **entropy of subgraphs**. Select the best subgraph in terms of subgraph entropy.

- Subgraphs of interest: Shortest path between the s-t pair (reachability and Shortest Path query), the triangles in $\mathcal{G}$ (#Triangles query), the explanation subgraphs in a node's neighborhood (Node classification).
- Entropy of a subgraph S,

$$H(S) = -Pr(S)\log Pr(S) - (1 - Pr(S))\log(1 - Pr(S)) \tag{4}$$

where $Pr(S) = \Pi_{e \in S} p(e)$.

Experiments: Datasets

Table 1: Statistics of datasets. Reach: reachability, SP: shortest path distance, #Tri: triangle counting, NC: node classification

graph	**type**	**queries shown**	**#nodes**	**#edges**	**edge prob.**
ER	synthetic	reach, SP, #Tri	15	22	0.27 ± 0.21
Biomine	biological	reach, SP	1 008 201	13 485 878	0.27 ± 0.21
Flickr	social	#Tri	78 322	10 171 509	0.09 ± 0.06
Products	crowdsourced	reach	2 173	37 641	0.17 ± 0.09
Papers	crowdsourced	#Tri	995	152 731	0.26 ± 0.23
DBLP	collaboration	Node Class.	632 870	3 301 970	0.46 ± 0.14

Experiments: Uncertanty estimation

Comparison w.r.t. to an exact method:

Table 2: Entropy estimate: comparison with **Exact** method (*ER*)

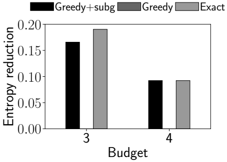
algorithm	**avg. running time (sec)**			**avg. error**		
	Reach	SP	#Tri	Reach	SP	#Tri
Exact	177.4	190.9	815.8	0	0	0
MC	0.039	0.096	0.368	0.088	0.086	0.058

Variants of MC algorithm:

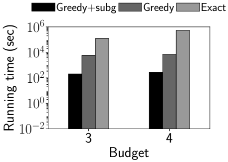
Table 3: Entropy estimate for reachability (*Biomine*)

algorithm	**avg. running time (sec)**	**avg. error**	**avg. peak mem. (GB)**
MC	32849.5	**0**	**4.0**
MC+BFS	2742.2	0.005	**4.0**
ProbTree-MC	18515.1	0.008	8.6
ProbTree-MC+BFS	2257.1	0.049	8.6
RSS	1672.1	0.100	**4.0**
ProbTree-RSS	**1342.8**	0.300	8.6

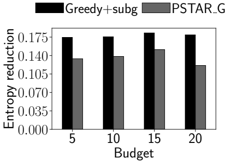
Experiments: Uncertainty reduction



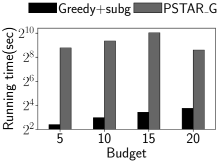
(a) **Reachability**



(b) **Reachability**



(c) **Reachability**



(d) **Reachability**

(a-b): Comparison among **Exact**, **Greedy**, and **Greedy+subgraph**, *ER* dataset, update $\mathcal{U}_1$. Each *s*-*t* pair is 3-hops away when budget = 3, and 4-hops away when budget = 4. **Greedy** often does not reduce entropy at all due to the cold-start problem. (c-d): Comparison between our Greedy+subgraph with baseline **PSTAR_G**[1], *Products* dataset, update $\mathcal{U}_2$. Our algorithm is more effective + 32-128X more efficient.

[1] **Lin et al. Human-powered data cleaning for probabilistic reachability queries on uncertain graphs. TKDE (2017).**

ML Application: Strategic Collaboration Problem

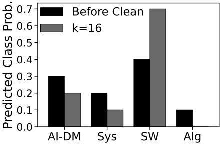
Find co-authors to collaborate often such that someone's profile is more prominently classified in a specific research domain.

Training We generate 100 possible worlds from the uncertain *DBLP* graph and train a 3-layer vanilla GCN on the labeled nodes from these possible worlds in a supervised manner.

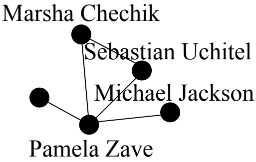
Testing For a test node (Pamela Zave), we obtain its predicted class labels across 10 possible worlds. Based on this, we obtain the frequency distribution of the predicted classes.

Finding Subgraph of Interest (S) For a test node, we find its majority predicted class and apply SubgraphX on each possible world to obtain an explanation subgraph that explains the majority class prediction in that possible world.

Reducing Uncertainty We apply **Greedy+subgraph** on the explanation subgraphs and clean the edges to 1.



(a) Pamela Zave ($H(\Omega) = 1.36, H(\Omega') = 1.16$)



(b) Recommended collaborations

Figure 2: (a) The distributions of predicted class for Pamela Zave (a senior researcher in software requirement engineering) before and after cleaning top-16 uncertain edges. (b) The recommended future collaborations (among her co-authors) such that she is more prominently classified into SW.

Conclusion

- We studied estimating and reducing uncertainty of computing network functions over uncertain graphs.
- For uncertainty estimation, we proposed an approximation algorithm with an (ϵ, δ)-type guarantee.
- For uncertainty reduction, we designed a practical greedy subgraphs selection algorithm that reduces the cold start problem of greedy approaches.
- Based on empirical results, our algorithms coupled with indexing and smart sampling strategies achieve the best accuracy and efficiency.
- Our case study depicted an application of uncertainty reduction for node classification in the strategic collaboration problem.

Future work.

Extending our solution to network functions generating multiple outputs, e.g., all subgraphs satisfying an input constraint, all nodes reachable within a limited number of hops, all nodes classified in a specific label, etc.

Q&A[2]

[2] I am on the job market. I would be happy to discuss collaborations and job opportunities.

Supplementary Slides

Algorithm

Estimate Entropy

Input: positive integers N, T, function $P : G \to \mathbb{R}$, uncertain graph $\mathcal{G} = (V, E, p)$

1: **for all** $i = 1, 2, \ldots, N$ **do**

2: Compute sample distribution: $\hat{Pr}_i, \hat{Sup}_i(P, \mathcal{G}) \leftarrow$ **Estimate PrSupport**$(T, P, \mathcal{G})$

3: Compute sample distribution Entropy: $\hat{H}_i \leftarrow -\sum_{\Omega \in \hat{Sup}_i(P,\mathcal{G})} \hat{Pr}_i(\Omega) \log \hat{Pr}_i(\Omega)$

return $\frac{1}{N} \sum_{i=1}^{N} \hat{H}_i$

Estimate PrSupport

Input: positive integer T, function $P : G \to \mathbb{R}$, uncertain graph $\mathcal{G} = (V, E, p)$

1: initialize $\hat{Pr}(.) \leftarrow 0$, $\hat{Sup}(P, \mathcal{G}) \leftarrow \phi$

2: **for all** $i = 1, 2, \ldots, T$ **do**

3: Sample a world $G_i \sqsubseteq \mathcal{G}$ via independent sampling of edges based on their probabilities

4: Compute observed function value: $\Omega = P(G_i)$

5: $\hat{Pr}(\Omega) \leftarrow \hat{Pr}(\Omega) + \frac{1}{T}$

6: $\hat{Sup}(P, \mathcal{G}) \leftarrow \hat{Sup}(P, \mathcal{G}) \cup \{\Omega\}$

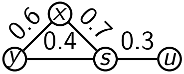
return $\hat{Pr}$, $\hat{Sup}(P, \mathcal{G})$

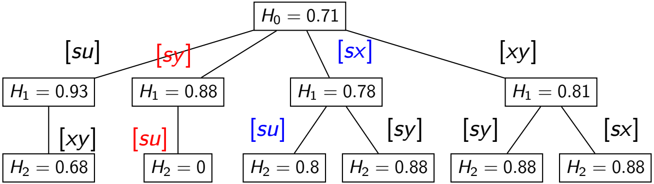
Sample complexity

$$Pr\left(\frac{\sum \hat{H}_i}{N} - H(\mathbf{\Omega}) \geq \frac{|Sup(P,\mathcal{G})|-1}{2T} + \epsilon\right) \leq 2e^{\frac{-2N\epsilon^2}{\log^2 |Sup(P,\mathcal{G})|}} \quad (5)$$

- we refer to $\frac{|Sup(P,\mathcal{G})|-1}{2T} + \epsilon$ as the margin of error.
- We observe that larger T decreases the margin of error.
- In contrast, N has little impact on the margin of error; however, the probability that our error estimate crosses that margin increases as we reduce N.
- When the support size $|Sup(P,\mathcal{G})|$ increases, both the margin of error and the probability that our error estimate crosses that margin increase. Thus, a lower support size $|Sup(P,\mathcal{G})|$ is preferred.

Cold-start problem





The edges that should be chosen for cleaning as per $\mathcal{U}_1$ to reach global optima are colored red ($[sy], [su]$). The edges chosen by Greedy are colored blue. **Greedy selects the edge $[sx]$ at round 1 because it is locally best at round 1. However, this leads to a globally sub-optimal selection ($[sx], [su]$).**