Most Probable Densest Subgraphs

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

Xiangyu Ke
ZJU, China
xiangyu.ke@zju.edu.cn

Arijit Khan
AAU, Denmark
arijitk@cs.aau.dk

Cheng Long
NTU, Singapore
c.long@ntu.edu.sg

***Abstract*—Computing the densest subgraph is a primitive graph operation with critical applications in detecting communities, events, and anomalies in biological, social, Web, and financial networks. In this paper, we study the novel problem of Most Probable Densest Subgraph (MPDS) discovery in uncertain graphs: Find the node set that is the *most likely* to induce a densest subgraph in an uncertain graph. We further extend our problem by considering various notions of density, e.g., clique and pattern densities, studying the top-k MPDSs, and finding the node set with the largest containment probability within densest subgraphs. We show that it is #P-hard to compute the probability of a node set inducing a densest subgraph. We then devise sampling-based efficient algorithms, with end-to-end accuracy guarantees, to compute the MPDS. Our thorough experimental results and real-world case studies on brain and social networks validate the effectiveness, efficiency, and usefulness of our solution.**

***Index Terms*—uncertain graphs, densest subgraphs**

I. Introduction

The discovery of dense subgraphs has attracted extensive attention in the data management community [1], [2], [3], [4], [5]. They may correspond to communities [6], filter bubbles and echo chambers [7], [8] in social networks, brain regions responding to stimuli [9] or related to diseases [10], and commercial value motifs in financial domains [11]. They also have wide applications in graph compression and visualization [12], [13], [14], indexing for reachability and distance queries [15], [16], and social piggybacking [17]. Densest subgraphs usually maximize some notion of density in a given graph, e.g., the *edge density* [1], defined as the ratio of the number of induced edges to the number of nodes in a subgraph. Although there are an exponential number of subgraphs, a densest subgraph can be found both exactly and approximately in polynomial time [1], [2]. There also exist many other density metrics [18], such as the edge ratio, edge surplus, discounted average degree, triangle density, clique density, pattern density, etc. Their algorithms are developed in [19], [20], [21], [5].

Uncertainty is intrinsic in large graphs due to errors in measurements [22], edge imputation using inference and prediction models [23], [24], and explicit manipulation including privacy reasons [25]. An uncertain graph, where every edge is associated with a probability of existence, is an expressive data model that has prompted a great deal of research [26], [27], [28]. Uncertain graphs are prevalent in many applications, such as biological networks [29], knowledge bases [30], social networks [31], influence maximization [32], road networks [33], and crowd sourcing [34]. Recently, researchers have extended several classic network problems to uncertain graphs, e.g., nearest neighbors [35], shortest paths and centrality [36], cliques [37], [38], [39], core and truss decomposition [40], [41], clustering [42], and motif counting [43]. Surprisingly, except for maximum *expected edge density* [44], [45], the study of densest subgraph discovery on uncertain graphs is still absent. The expected edge density of an uncertain graph is defined as the expectation of the edge density value of a possible world (i.e., a deterministic graph) of the uncertain graph, chosen at random [44]. However, a subgraph of the uncertain graph having the maximum expected edge density may induce densest subgraphs *only in a few* (even zero) possible worlds of that uncertain graph. Such a subgraph can be large with many low-probability edges, or having nodes that are loosely connected (see Example 1). This defeats the purpose of finding a densest subgraph. Instead, many applications would require a densest subgraph with a high precision, such as being the densest with a high probability. Specifically, given an uncertain graph $\mathcal{G}$, *our goal is to find the node set that is the most likely to induce a densest subgraph in* $\mathcal{G}$, i.e., maximize the sum of the probabilities of those possible worlds of $\mathcal{G}$ in which this node set induces a densest subgraph. We refer to the uncertain subgraph induced by this node set as the *most probable densest subgraph* (MPDS). To the best of our knowledge, computing the MPDS is a novel problem. We demonstrate real-world applications and case studies of our problem on uncertain brain (§ VI-F) and social (§ VI-E) networks, where our proposed MPDS distinguishes healthy brains from those with autism and identifies meaningful communities in a social network.

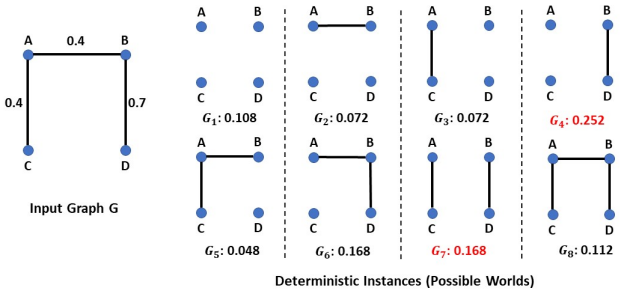


Fig. 1: Possible worlds of an uncertain graph and their probabilities

Example 1. *Figure 1 shows all possible worlds of an uncertain graph with their existence probabilities. It can be verified*

Xiangyu Ke is the corresponding author. Arijit Khan acknowledges support from the Novo Nordisk Foundation grant NNF22OC0072415. Cheng Long is supported by the Ministry of Education, Singapore, under its Academic Research Fund (Tier 1 Award (RG77/21)). Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the funding agencies.

TABLE I: Edge densities in possible worlds (PWs), *expected edge densities* (EEDs) and *densest subgraph probabilities* (DSPs) of node sets in the uncertain graph in Figure 1. The EED of a node set U is the sum of the edge densities of the subgraphs induced by U across all PWs, weighted by their probabilities. The DSP of U is the sum of the probabilities of those PWs where U induces a densest subgraph.

PW:Pr.	{A,B}	{A,C}	{B,D}	{A,B,C}	{A,B,D}	{A,B,C,D}
G_1:0.11	0	0	0	0	0	0
G_2:0.07	**0.5**	0	0	0.33	0.33	0.25
G_3:0.07	0	**0.5**	0	0.33	0	0.25
G_4:0.25	0	0	**0.5**	0	0.33	0.25
G_5:0.05	0.5	0.5	0	**0.67**	0.33	0.5
G_6:0.17	0.5	0	0.5	0.33	**0.67**	0.5
G_7:0.17	0	**0.5**	**0.5**	0.33	0.33	**0.5**
G_8:0.11	0.5	0.5	0.5	0.67	0.67	**0.75**
EED	0.2	0.2	0.35	0.27	0.37	**0.38**
DSP	0.07	0.24	**0.42**	0.05	0.17	0.28

that, in each world, the connected component is also the densest subgraph. As Table I shows, the node set {A,B,C,D} has the maximum expected density (0.38), but it induces a densest subgraph only *in possible worlds G_7 and G_8 with low existence probabilities (0.168 and 0.112). Thus, the probability of {A,B,C,D} inducing a densest subgraph is only 0.28. In contrast, the node set {B,D} has a lower expected density (0.35), but its probability of inducing a densest subgraph is much higher (0.42), since it induces a densest subgraph in possible worlds G_4 and G_7 with high existence probabilities.*

Challenges and our contributions. We formulate and study the novel problem of *most probable densest subgraph* (MPDS) discovery in uncertain graphs: Given an uncertain graph $\mathcal{G}$, find the node set that is the most likely to induce a densest subgraph in $\mathcal{G}$. Our contributions are summarized below.

• **Novel problems:** To the best of our knowledge, the densest subgraph discovery in uncertain graphs has not been investigated before, other than expected edge density [44], [45]. Based on edge density, clique density, and pattern density [5], we propose *densest subgraph probability* as a more sophisticated density metric. We prove that computing the densest subgraph probability is $\#\mathbf{P}$-hard. *We formulate and study the following novel problems of* MPDS *discovery in uncertain graphs* (§ II): MPDS based on edge density, clique density, and pattern density; for each of them, their top-k variants and nucleus densest subgraph (NDS) variants. Real-world applications and case studies demonstrate the usefulness of our novel problems.

• **Efficient approximate solution with end-to-end accuracy guarantees:** In spite of the $\#\mathbf{P}$-hardness of computing the densest subgraph probability, we design an efficient approximation algorithm for the top-k densest subgraphs discovery, with an end-to-end accuracy guarantee. Our solution for edge density-based MPDS is built on independent sampling of possible worlds (e.g., via Monte-Carlo sampling) and, in each of them, efficient enumeration of all edge-densest subgraphs (via [46]). We provide time and space complexity analyses and theoretical accuracy guarantees of our method (§III-A).

• **Extension to other density notions:** Besides edge density, our algorithm can be extended well to clique and pattern densities, while ensuring the same accuracy guarantee. We notice that, while there exist efficient algorithms to find *one* clique-densest and *one* pattern-densest subgraph in a deterministic graph [5], the problems of enumerating *all* clique-densest and *all* pattern-densest subgraphs in a deterministic graph have not been studied earlier. However, such enumerations are required in our overall solution framework. Therefore, as additional technical contributions, we develop novel, *exact* algorithms for efficiently enumerating *all* clique-densest and *all* pattern-densest subgraphs in a deterministic graph (§ III-B, III-C).

• **Practical nucleus densest subgraphs** (NDS)**:** In large graphs, we find that the densest subgraph probability of every possible node set may be quite small, due to the existence of many possible worlds, each having a smaller probability; and any two worlds might not have exactly the same densest subgraph. This defeats our purpose of identifying a node set that induces a *densest* subgraph with a *high probability*. In such cases, we propose to find those nodes which are most likely to form the "nucleus" of various densest subgraphs, i.e., whose containment probability within a densest subgraph is maximized. We develop an approximate solution and present theoretical analyses about its accuracy-efficiency trade-offs. *The novelty of our solution is that, by finding the maximum-sized densest subgraph in each sampled world, we reduce this problem to the closed frequent itemset mining problem, for which efficient algorithms like TFP [47] exist* (§ IV).

• **Experiments and case studies:** Our rigorous experiments (§ VI) show that our MPDS and NDS are different from existing notions of dense subgraphs in uncertain graphs (§ VI-B). Also, our methods are very efficient even on large graphs (§ VI-G) and return reasonably accurate results when compared to the exact methods (§ VI-H). Moreover, our case studies on brain (§ VI-F) and social (§ VI-E) networks demonstrate useful real-world applications of the MPDS.

II. Preliminaries

An uncertain graph $\mathcal{G}$ is a triple (V, E, p), where V is a set of n nodes, $E \subseteq V \times V$ is a set of m undirected unweighted edges, and the function $p : E \rightarrow (0, 1]$ assigns a probability of existence to each edge. Following the bulk of the literature on uncertain graphs [26], [35], [36], [48], [49], [50], we assume that the edges exist independent of each other: The uncertain graph $\mathcal{G}$ can be interpreted as a probability distribution over 2^m deterministic instances (possible worlds) $G = (V, E_G) \sqsubseteq \mathcal{G}$ obtained by independently sampling the edges. The probability of a possible world $G = (V, E_G)$ being observed is:

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

In the rest of this section, we first revisit several notions of *graph density* [4], [19], [5] in deterministic graphs (§II-A). We next extend these notions to uncertain graphs based on the possible world semantics (§II-B). Then, our novel Most Probable Densest Subgraph (MPDS) problem is formally introduced, together with several practical variants (§II-C). Finally, we discuss the hardness of our problem (§II-D).

A. Density Notions over Deterministic Graphs

1) Edge Density: The edge density [1] measures the average degree per node, which can be used for community detection [51], [3] in social networks.

Definition 1 (Edge Density [1])**.** *The edge density ρ_e of a deterministic graph $G = (V, E)$ is defined as:*

$$\rho_e(G) = \frac{|E|}{|V|} \tag{2}$$

2) Clique Density: An h-clique ($h \geq 2$) is a complete graph of h nodes. The clique density is formally defined below.

Definition 2 (h-Clique Density [19])**.** *Given an integer $h \geq 2$, the h-clique density ρ_h of a deterministic graph $G = (V, E)$, with the number of h-cliques $\mu_h(G)$, is defined as:*

$$\rho_h(G) = \frac{\mu_h(G)}{|V|} \tag{3}$$

Notice that a 2-clique is an edge. Thus, edge density is a special case of clique density when $h = 2$. Clique density benefits in higher-order community discovery and finding subgraphs which are large near-cliques [52], [19].

3) Pattern Density: Given an arbitrary pattern graph, the pattern density measures the average number of such patterns per node occurring in a subgraph.

Definition 3 (Pattern Density [5])**.** *Given a pattern graph ψ, the pattern density ρ_ψ (w.r.t. ψ) of a deterministic graph $G = (V, E)$, with the number of ψ-instances $\mu_\psi(G)$, is defined as:*

$$\rho_\psi(G) = \frac{\mu_\psi(G)}{|V|} \tag{4}$$

Clearly, clique density is a special case of pattern density when the input pattern is a clique. Pattern density can be more expressive in real-world applications. For instance, in the LinkedIn social network, the "employer" nodes (e.g., companies) cannot directly link to the "education" nodes (e.g., universities). Thus, "employer" nodes and "education" nodes never form a clique. However, they can be connected via nodes representing "employee". A subgraph which is dense w.r.t. the "employer-employee1-education-employee2-employer" diamond pattern may identify a group of employees with common work experiences and educational backgrounds.

B. Extending Density Notions to Uncertain Graphs

We define the probability of a node set inducing a densest subgraph in an uncertain graph using possible world semantics.

Definition 4 (Densest Subgraph Probability)**.** *Given an uncertain graph $\mathcal{G} = (V, E, p)$ and a node set $U \subseteq V$, the densest subgraph probability of U, denoted by $\tau(U)$, is the sum of the probabilities of all possible worlds where the subgraph induced by U has the largest density. Formally,*

$$\tau(U) = \sum_{G \sqsubseteq \mathcal{G}} \Pr(G) \times \mathbb{1}\left[\rho(G[U]) = \max_{W \subseteq V} \rho(G[W])\right] \tag{5}$$

The above equation verifies, in each possible world of the uncertain graph $\mathcal{G}$, whether the node set U induces a subgraph with the maximum density. $G[W] = (W, E_G[W])$ denotes the subgraph of G induced by a node set $W \subseteq V$, where

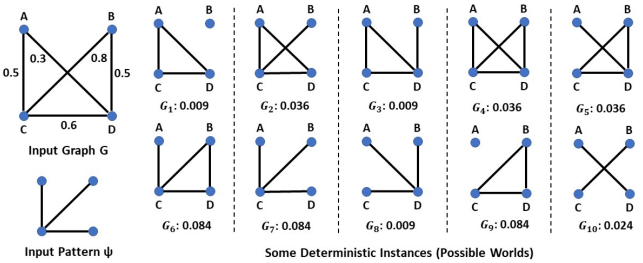


Fig. 2: 3-Clique and pattern densities in an uncertain graph

$E_G[W] = \{(u, v) \in E_G : u \in W, v \in W\}$. The indicator function $\mathbb{1}[\cdot]$ returns 1 if the inner condition is true (i.e., the subgraph induced by U has the largest density in G), and 0 otherwise. Note that the node set whose induced subgraph has the maximum edge density in G may not be unique. The density metric ρ can follow any of the density notions (§ II-A) based on the real application demand. In the following, without loss of generality, *densest subgraph probability* is coupled with edge density ρ_e by default. For h-clique density ρ_h and pattern density ρ_ψ, we refer to *h-clique densest subgraph probability* τ_h and *pattern densest subgraph probability* τ_ψ respectively.

Example 2. *Figure 2 shows an input uncertain graph and some of its possible worlds. The subgraphs induced by the node set $\{A, C, D\}$ contain a 3-clique (i.e., triangle) only in possible worlds G_1, G_2, G_3, and G_4. In fact, $\{A, C, D\}$ induces 3-clique densest subgraphs in possible worlds G_1, G_2, and G_3, thus the 3-clique densest subgraph probability of $\{A, C, D\}$ is 0.009 + 0.036 + 0.009 = 0.054.*

For the pattern ψ, we notice that the subgraphs induced by node set $\{A, B, C, D\}$ contain ψ only in possible worlds G_2, G_3, G_4, G_5, G_6, G_7, and G_8. Moreover, $\{A, B, C, D\}$ induces the ψ-densest subgraphs in all these six possible worlds, thus the ψ-densest subgraph probability of $\{A, B, C, D\}$ is 0.294.

C. Problem Formulations

We study the novel problem of finding the node set which is the most likely to induce a densest subgraph in an uncertain graph, formally defined as follows.

Problem 1 (Most Probable Densest Subgraph (MPDS))**.** *Given an uncertain graph $\mathcal{G} = (V, E, p)$, find the node set $U^* \subseteq V$ that has the highest densest subgraph probability.*

$$U^* = \arg\max_{U \subseteq V} \tau(U) \tag{6}$$

The h-Clique-MPDS and Pattern-MPDS problems can be defined analogously. In the following, we provide two other variants of our MPDS problem. First, the user may be interested in exploring more possible choices besides the best node set. Thus, we provide the top-k variant below.

Problem 2 (Top-k Most Probable Densest Subgraphs (Top-k MPDSs))**.** *Given an uncertain graph $\mathcal{G} = (V, E, p)$ and a positive integer k, find the top-k distinct node sets $U_1^*, U_2^*, \ldots, U_k^*$ (where each $U_i^* \subseteq V$, $1 \leq i \leq k$) having the highest densest subgraph probabilities. Formally,*

$$\begin{aligned} &\tau(U_i^*) \geq \tau(U_{i+1}^*) \quad \forall i \in [1, k) \\ &\tau(U_i^*) \geq \tau(U)\ \forall i \in [1, k]\ \&\ \forall U \subseteq V \setminus \{U_1^*, \ldots, U_k^*\} \end{aligned} \tag{7}$$

Second, in large graphs, we observe that the densest subgraph probability of every node set can be quite small, e.g., below 3.91×10^{-5} in Homo Sapiens, Biomine, and Twitter (§ VI). In such cases, reporting MPDSs contradicts our goal of identifying node sets that can induce densest subgraphs *with high probabilities*. Instead, we find those sets with the highest containment probabilities within a densest subgraph. Such node sets form the "nuclei" of various densest subgraphs across different possible worlds of the uncertain graph.

Definition 5 (Densest Subgraph Containment Probability)**.** *The densest subgraph containment probability $\gamma(U)$ of $U \subseteq V$ is the sum of the probabilities of all possible worlds G of the uncertain graph $\mathcal{G} = (V, E, p)$ such that U is contained in a densest subgraph of G. Formally,*

$$\gamma(U) = \sum_{G \sqsubseteq \mathcal{G}} \Pr(G) \times \mathbb{1}\left[\exists\, U' \supseteq U \,:\, \rho(G[U']) = \max_{W \subseteq V} \rho(G[W])\right]$$

Example 3. *Considering the input uncertain graph in Figure 1, the node set $\{B, D\}$ can induce a densest subgraph with probability 0.42 (in possible worlds G_4 and G_7). We notice that $\{B, D\}$ is also contained in densest subgraphs of other possible worlds (G_6 and G_8), even though $\{B, D\}$ alone does not induce densest subgraphs in these worlds. The overall densest subgraph containment probability of $\{B, D\}$ is 0.7 (due to G_4, G_6, G_7, and G_8), which implies that this node set is a critical component in forming densest subgraphs.*

We aim to find the top-k node sets having the highest densest subgraph containment probabilities. However, there are two caveats. First, a very small node set (as small as a singleton) can also have a high containment probability; however, such sets do not represent meaningful graph communities. Second, if a node set U and one of its supersets U' have equal containment probabilities $\gamma(\cdot)$, then it makes more sense to report U' only (in order to avoid redundancy in the result). To mitigate these, we respectively impose the restrictions that all returned node sets must have a minimum specified size and must be closed w.r.t. $\gamma(\cdot)$. A node set is closed w.r.t. $\gamma(\cdot)$ if none of its supersets has the same value of $\gamma(\cdot)$. We now define our NDS problem.

Problem 3 (Top-k Nucleus Densest Subgraphs (NDSs))**.** *Given an uncertain graph $\mathcal{G} = (V, E, p)$ and positive integers k and l_m, let $\mathcal{V}_c^{\geq l_m}$ denote the set of all node sets of size at least l_m that are closed w.r.t. $\gamma(\cdot)$. Find the top-k closed node sets $U_1^*, U_2^*, \ldots, U_k^*$ (where $U_i^* \in \mathcal{V}_c^{\geq l_m}$, $i \in [1, k]$) having the highest densest subgraph containment probabilities. Formally,*

$$\gamma(U_i^*) \geq \gamma(U_{i+1}^*) \quad \forall i \in [1, k)$$
$$\gamma(U_i^*) \geq \gamma(U) \quad \forall i \in [1, k] \;\&\; \forall U \in \mathcal{V}_c^{\geq l_m} \setminus \{U_1^*, \ldots, U_k^*\} \quad (8)$$

Notice that the Top-k MPDS and the NDS problems can be analogously extended to their clique and pattern versions.

D. Hardness

Theorem 1. *Computing the densest subgraph probability of a node set U in an uncertain graph $\mathcal{G} = (V, E, p)$ is $\#\mathbf{P}$-hard.*

Proof. We prove by a reduction from the $\#\mathbf{P}$-hard problem of finding the number of matchings in a graph [48]. A matching in a deterministic graph $G = (V, E)$ is an edge set $M \subseteq E$ without any common nodes.

Consider a deterministic graph $G = (V, E)$. This graph is transformed, by adding two new nodes v_1 and v_2 along with an edge between them, into an uncertain graph $\mathcal{G} = (V \cup \{v_1, v_2\}, E \cup \{(v_1, v_2)\}, p)$, where the probability of each edge is 0.5, except the new edge (v_1, v_2) which has probability 1. Clearly, this reduction takes $\mathcal{O}(|E|)$ time, which is polynomial in the size of G. It can be shown that: • Any possible world $G' \sqsubseteq \mathcal{G}$ with non-zero probability has $\Pr(G') = (0.5)^{|E|}$. • There is a bijection between the set of subsets of E and the set of possible worlds of $\mathcal{G}$ with non-zero probability. • The node set $\{v_1, v_2\}$ induces a densest subgraph in a possible world iff every node has degree at most 1 in that world, i.e., the edges in the world excluding (v_1, v_2) form a matching in G. Thus,

$$\begin{aligned}
\tau(\{v_1, v_2\}) &= \sum_{G' \sqsubseteq \mathcal{G}} \Pr(G') \times \mathbb{1}\left[\{v_1, v_2\} = \arg\max_{W \subseteq V} \rho(G'[W])\right] \\
&= \sum_{G' \sqsubseteq \mathcal{G}} \Pr(G') \times \mathbb{1}\left[\text{each node has degree at most 1 in } G'\right] \\
&= (0.5)^{|E|} \sum_{G' \sqsubseteq \mathcal{G} : \Pr(G') \neq 0} \mathbb{1}\left[\text{each node has degree at most 1 in } G'\right] \\
&= (0.5)^{|E|} \sum_{M \subseteq E} \mathbb{1}\left[\text{M is a matching in } G\right]
\end{aligned}$$

The sum in the last line above is the number of matchings in G. Thus, a solution to our problem on $\mathcal{G}$ provides a solution to the matching counting problem on G. □

Since the computation of $\tau(U)$, for a given U, is $\#\mathbf{P}$-hard, the computations of its generalizations $\tau_h(U)$ and $\tau_\psi(U)$ are also $\#\mathbf{P}$-hard. Thus, finding the node sets with the top-k densest subgraph probabilities, as well as computing the NDS, are also very difficult. *Given such computational challenges, we design approximate algorithms, with end-to-end accuracy guarantees, to find the most probable densest subgraphs in an uncertain graph, based on various graph density notions.*

III. Approximate Solutions for Densest Subgraphs

In this section, we develop approximation algorithms for detecting the top-k MPDSs, along with end-to-end theoretical accuracy guarantees. Our **technical contributions** are as follows: **(1)** We design *novel approximation methods to compute the top-k MPDSs in an uncertain graph for all density notions: edge (§ III-A), clique (§ III-B), and pattern (§ III-C).* **(2)** As building blocks of the algorithms for clique and pattern densities, we also design *novel algorithms to discover all clique and pattern densest subgraphs in a deterministic graph*[1][2](§ III-B and § III-C). **(3)** Additionally, we use these methods to design *approximation algorithms, with end-to-end theoretical quality guarantees, for computing the corresponding NDS (§ IV).*

A. Top-k MPDS: Approximate Algorithm

[1]Due to the additional nodes for cliques/patterns in flow network construction, the definitions and theorems do not trivially follow [46], more details can be found in the remark, § III-B

[2]Our empirical study (§ VI-D) validates that considering *all* densest subgraphs can significantly outperform (e.g., up to 20× in LastFM) the method that considers only one randomly chosen densest subgraph.

Algorithm 1 Top-k MPDS estimation

Input: Uncertain graph $\mathcal{G} = (V, E, p)$, positive integer k, and number of samples θ
Output: (Approximate) Top-k MPDS
1: **for all** $U \subseteq V$ **do**
2: $\quad\widehat{\tau}(U) \leftarrow 0$
3: **for** $i = 1$ **to** θ **do**
4: $\quad$Sample a possible world $G \sqsubseteq \mathcal{G}$
5: $\quad S \leftarrow$ All densest subgraphs in G via [46]
6: $\quad$**for all** $U \in S$ **do**
7: $\quad\quad\widehat{\tau}(U) \leftarrow \widehat{\tau}(U) + \frac{1}{\theta}$
8: **return** Top-k U's having the highest $\widehat{\tau}(U)$

The proposed solution (Algorithm 1) runs θ independent iterations as follows: Sample a possible world $G \sqsubseteq \mathcal{G}$ and find *all* the node sets inducing the densest subgraphs in G (Line 5). $\widehat{\tau}(U)$ denotes the estimated densest subgraph probability, which is computed as the average frequency that a node set U induces a densest subgraph across θ rounds. Finally, we return the top-k node sets having the highest $\widehat{\tau}(\cdot)$.

Lemma 1. *$\widehat{\tau}(U)$ is an unbiased estimator for $\tau(U)$. Formally, $\mathbb{E}[\widehat{\tau}(U)] = \tau(U)$.*

Proof. Let $X_i(U)$ be a binary random variable denoting whether U induces a densest subgraph in the i^{th} possible world; thus $\widehat{\tau}(U) = \frac{1}{\theta}\sum_{i=1}^{\theta} X_i(U)$. Clearly, $\mathbb{E}[X_i(U)] = \Pr(X_i[U] = 1) = \tau(U)$, and hence $\mathbb{E}[\widehat{\tau}(U)] = \tau(U)$. □

The unbiasedness ensures that the estimated $\widehat{\tau}(U)$ goes closer to the true value $\tau(U)$ as the sample size θ increases.

The technique in [46] for computing all densest subgraphs in a deterministic graph (Line 5) involves reducing the graph to its $\lceil\tilde{\rho}\rceil$-core [53], where $\tilde{\rho}$ is a lower bound on the maximum edge density ρ_e^* of any subgraph. This is followed by computing ρ_e^* using the state-of-the-art Goldberg's algorithm [1]. During each iteration of its binary search, the Goldberg's algorithm tries to find a subgraph with density larger than a guessed value α by computing the minimum cut in a flow network parameterized by α. Once ρ_e^* is found, all densest subgraphs are enumerated by traversing the strongly connected components (SCCs) in the residual graph (under a maximum flow) of the flow network with $\alpha = \rho_e^*$; the details can be found in [46] and in the example below.

Example 4. *We shall compute all densest subgraphs in a possible world G (Figure 3(b)) of an uncertain graph $\mathcal{G}$ ((Figure 3(a)). A flow network G_α (Figure 3(c)) is constructed as follows. (1) Add a source node s and a sink node t. (2) If an edge (u, v) exists in G, add an edge from u to v and one from v to u in G_α, both with capacity 1. (3) Add an edge from s to each node v in G with capacity equal to the degree of v in G. (4) Add an edge from each node in G to t with capacity 2α. Goldberg's algorithm [1] conducts a binary search with $[0, m]$ as the initial range of α. In each iteration, it guesses an α and computes the maximum flow (minimum cut) in the flow network. Once terminated, the optimal density is assigned to be $\rho^* = \alpha$. In this example, we get $\rho^* = 1$ and a densest subgraph $\{A, B, C, D\}$ (which corresponds to the minimum cut). To find the other densest subgraphs, we create the residual graph ((Figure 3(d)) by removing all the edges with zero residual capacity. The densest subgraphs are enumerated by exploring the SCCs in this residual graph. In this case, we find one additional densest subgraph $\{B, C, D\}$.*

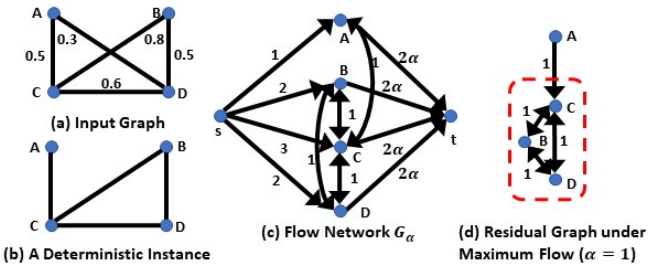


Fig. 3: Finding all densest subgraphs in a possible world

Space complexity. The majority of the memory is consumed by the flow network in each iteration of the binary search, and hence the overall space complexity is $\mathcal{O}(m + n)$.

Time complexity. Recall that we sample θ possible worlds. The computation of the $\lceil\tilde{\rho}\rceil$-core of each possible world takes $\mathcal{O}(m)$ time [53]. Let n_c and m_c denote the number of nodes and edges respectively in the $\lceil\tilde{\rho}\rceil$-core of a sampled possible world $G \sqsubseteq \mathcal{G}$. As shown in [46], the computation of ρ_e^* takes $\mathcal{O}\left(n_c m_c \log\left(\frac{n_c^2}{m_c}\right)\right)$ time, while the enumeration of the densest subgraphs takes $\mathcal{O}(L)$ time per subgraph, where L is the number of nodes in that densest subgraph (note that $L \leq n$). Denoting respectively by n_c^*, m_c^*, d^*, and L^* the maximum number of nodes, edges, densest subgraphs, and nodes in a densest subgraph in any possible world, the overall time complexity of our algorithm is $\mathcal{O}\left(\theta\left(m + n_c^* m_c^* \log\left(\frac{n_c^{*2}}{m_c^*}\right) + d^* L^*\right)\right)$. Practically, $n_c^* \ll n$, $m_c^* \ll m$ and $L^* \ll n$, as validated in our experiments (§ VI). For instance, $\frac{n_c^*}{n} = 2.45 \times 10^{-4}$, $\frac{L^*}{n} = 1.8 \times 10^{-4}$, $\frac{m_c^*}{m} = 5.43 \times 10^{-3}$, and $d^* = 1$ in our large Twitter dataset, making Algorithm 1 efficient even with large-scale graphs.

Accuracy guarantee. We theoretically analyze the sample size θ to return the true top-k node sets with a high probability. First, the true top-k node sets are proved likely to be included in the candidate node sets[3] after θ rounds (Theorem 2). Second, we ensure that those sets indeed have the k largest *estimated* densest subgraph probabilities (Theorem 3).

Theorem 2. *Let $V_1, \ldots, V_k$ denote the true top-k node sets having the highest densest subgraph probabilities, and let CV denote the set of candidate node sets after θ rounds. Then,*

$$\Pr(\{V_1, \ldots, V_k\} \subseteq CV) \geq 1 - \sum_{i=1}^{k} (1 - \tau(V_i))^\theta \qquad (9)$$

Proof. Since we compute all densest subgraphs in each of θ independently sampled possible worlds, $\Pr(V_i \notin CV) = (1 - \tau(V_i))^\theta \ \forall i \in \{1, \ldots, k\}$. From the union bound,

$$\Pr(\{V_1, \ldots, V_k\} \subseteq CV) = 1 - \Pr(\exists i \in \{1, \ldots, k\} : V_i \notin CV)$$
$$\geq 1 - \sum_{i=1}^{k} \Pr(V_i \notin CV) = 1 - \sum_{i=1}^{k} (1 - \tau(V_i))^\theta$$

From Theorem 2, if the densest subgraph probabilities of the true top-k node sets and the sample size θ are reasonably

[3] A node set U is called a candidate node set if its estimated densest subgraph probability $\widehat{\tau}(U)$ after θ rounds is non-zero, i.e., U induces a densest subgraph in at least one of θ possible world samples of $\mathcal{G}$.

large, they are all highly likely to be included in the candidate node set, which is necessary for all of them to be returned.

Theorem 3. *Let $V_1, \ldots, V_{k+1}$ denote the true top-$(k+1)$ node sets having the highest densest subgraph probabilities, and let CV denote the set of candidate node sets after θ rounds. Define $mid = \frac{1}{2}\left[\tau\left(V_k\right) + \tau\left(V_{k+1}\right)\right]$ and*

$$d_U = \begin{cases} \tau(U) - mid & \text{if } U \in \{V_1, \ldots, V_k\} \\ mid - \tau(U) & \text{otherwise} \end{cases} \tag{10}$$

Then, the probability that $V_1, \ldots, V_k$ are returned by Algorithm 1 is at least

$$\left[1 - \sum_{i=1}^{k} \left(1 - \tau(V_i)\right)^{\theta}\right] \left[1 - \sum_{U \in CV} \exp\left(-2 d_U^2 \theta\right)\right] \tag{11}$$

Proof. Let $CV' = CV \setminus \{V_1, \ldots, V_k\}$. We have:

$$\begin{aligned} &\Pr(V_1, \ldots V_k \text{ are returned}) \geq \Pr(\{V_1, \ldots, V_k\} \subseteq CV) \times \\ &\Pr\left(\left(\bigwedge_{U \in \{V_1, \ldots, V_k\}} \widehat{\tau}(U) > mid\right) \wedge \left(\bigwedge_{U \in CV'} \widehat{\tau}(U) < mid\right)\right) \end{aligned} \tag{12}$$

Now, using the union bound and Hoeffding's inequality,

$$\begin{aligned} &\Pr\left(\left(\bigwedge_{U \in \{V_1, \ldots, V_k\}} \widehat{\tau}(U) > mid\right) \wedge \left(\bigwedge_{U \in CV'} \widehat{\tau}(U) < mid\right)\right) \\ &= 1 - \Pr\left(\left(\bigvee_{U \in \{V_1, \ldots, V_k\}} \widehat{\tau}(U) \leq mid\right) \vee \left(\bigvee_{U \in CV'} \widehat{\tau}(U) \geq mid\right)\right) \\ &\geq 1 - \sum_{U \in \{V_1, \ldots, V_k\}} \Pr\left(\widehat{\tau}(U) \leq mid\right) - \sum_{U \in CV'} \Pr\left(\widehat{\tau}(U) \geq mid\right) \\ &= 1 - \sum_{U \in \{V_1, \ldots, V_k\}} \Pr\left(\widehat{\tau}(U) - \tau(U) \leq -d_U\right) - \sum_{U \in CV'} \Pr\left(\widehat{\tau}(U) - \tau(U) \geq d_U\right) \\ &\geq 1 - \sum_{U \in CV} \exp\left(-2 d_U^2 \theta\right) \end{aligned} \tag{13}$$

Finally, plugging (9) and (13) into (12), we obtain (11). □

From Theorem 3, if the densest subgraph probabilities of the true top-k node sets are reasonably large in contrast to the others and if the sample size θ is sufficiently large, the true top-k sets are returned by Algorithm 1 with a high probability.

Remarks. (1) Notice that the algorithmic framework and accuracy guarantees can be easily adapted to solve the top-k Clique-MPDS (resp. Pattern-MPDS) problems. However, we need to develop an efficient algorithm for detecting *all* clique (resp. pattern)-densest subgraphs in each sampled possible world (Line 5 of Algorithm 1), which is our *novel technical contribution* in § III-B (resp. § III-C). **(2)** The analyses in this section are based on the assumption that we use *Monte Carlo* to sample possible worlds. There also exist other sampling techniques such as *Lazy Propagation* [54] and *Recursive Stratified Sampling* [55]. We empirically show that, for our problem, these three sampling strategies result in *similar sample sizes* θ and have *comparable running times*, while *Monte Carlo* consumes *much less memory* (§ VI-G).

Algorithm 2 Find all clique-densest subgraphs

Input: Deterministic graph $G = (V, E)$, positive integer h
Output: All h-clique densest subgraphs in G
1: $\tilde{\rho} \leftarrow$ Density returned by the peeling method [19], [5]
2: $G_c \leftarrow (\lceil \tilde{\rho} \rceil, h)$-core of G [5]
3: $\Lambda \leftarrow$ All $(h-1)$-cliques contained in h-cliques in G_c [56]
4: $\rho_h^* \leftarrow \max_{S \subseteq V} \rho_h(S)$ [57]
5: $\mathcal{H} \leftarrow$ Algorithm 6 (G_c, Λ, ρ_h^*) [20], [5]
6: $f^* \leftarrow$ Maximum flow in $\mathcal{H}$
7: $\mathcal{C} \leftarrow$ SCCs of the residual graph $\mathcal{H}_{f^*}$, excluding those of s and t
8: **return** Algorithm 3 ($\emptyset$, $\mathcal{C}$, V)

B. h-Clique-MPDS: Approximate Algorithm

Inspired by [46], we develop a novel, exact, and efficient solution to discover *all* clique-densest subgraphs in a deterministic graph (Algorithm 2). *This is a novel problem, and no existing work has studied it. Therefore, Algorithm 2 is one of our novel technical contributions*. In the following, we first revisit the concepts of clique degree (Definition 6) and clique-based core (Definition 7) in deterministic graphs. Then, we illustrate the technical details and the intuitions of our algorithm, together with a running example. Finally, we provide theoretical analyses about its efficiency and correctness.

Definition 6 (h-Clique Degree [5])**.** *The h-clique degree ($h \geq 2$) of a node v in a deterministic graph G, denoted by $deg_G(v, h)$, is the number of h-cliques in G containing v.*

Definition 7 ((k,h)-Core [5])**.** *Given a deterministic graph G and two integers $h \geq 2$ and $k \geq 0$, the (k,h)-core of G, denoted by $\mathcal{R}_k$, is the largest subgraph of G such that, for every node v in $\mathcal{R}_k$, $deg_{\mathcal{R}_k}(v, h) \geq k$.*

Armed with these definitions, we proceed to the details of Algorithm 2, which consists of two general steps: **(1)** The technique in [57] is applied to compute the maximum density of any subgraph of G. **(2)** A flow network $\mathcal{H}$ is constructed following [20], [46]. The SCCs of the residual graph under a maximum flow in $\mathcal{H}$ indicate all densest subgraphs of G.

In Line 1, it runs the peeling method of [19], which iteratively removes the node with the smallest h-clique degree and returns the maximum density among all the resultant subgraphs, denoted by $\tilde{\rho}$. Then, in Line 2, it replaces G with its $(\lceil \tilde{\rho} \rceil, h)$-core, i.e., the subgraph induced by those nodes which have h-clique degree at least $\tilde{\rho}$. After that, Line 3 computes the set Λ of all $(h-1)$-cliques contained in h-cliques in G, which are enumerated using the method in [56]. Line 4 computes ρ_h^*, the maximum h-clique density of any subgraph, by the method in [57], which iteratively computes a (predicted) clique-densest subgraph via optimizing a convex program, till the computed subgraph is deemed to be indeed clique-densest. After that, the clique density of the computed subgraph is returned.

Next, Line 5 constructs a flow network $\mathcal{H}$ following [20], [5], which contains one node for each $(h-1)$-clique in Λ and one for each node in V, in addition to a source node s and a sink node t (see Appendix A for the pseudocode). Once $\mathcal{H}$ is constructed, Algorithm 2 computes a maximum flow f^* in $\mathcal{H}$ (Line 6) and then identifies the strongly connected components (SCCs) of the residual graph $\mathcal{H}_{f^*}$ of $\mathcal{H}$ under f^* (Line 7) after removing the edges with zero residual capacity.

Algorithm 3 Enumerate all clique densest subgraphs

Input: Component sets $\mathcal{C}_1$ and $\mathcal{C}_2$, node set V
Output: All clique-densest subgraphs
1: $R \leftarrow \emptyset$
2: **if** $\mathcal{C}_1 \neq \emptyset$ **then**
3: $\quad R \leftarrow R \cup \left(\bigcup_{C \in \mathcal{C}_1 \cup des(\mathcal{C}_1)} C \cap V\right)$
4: **for all** $C \in \mathcal{C}_2$ **do**
5: $\quad$ **if** $C \cap V \neq \emptyset$ **then**
6: $\quad\quad \mathcal{C}_2 \leftarrow \mathcal{C}_2 \setminus \{C\}$
7: $\quad\quad S \leftarrow$ Algorithm 3 $(\mathcal{C}_1 \cup \{C\}, \mathcal{C}_2 \setminus (des(C) \cup anc(C)), V)$
8: $\quad\quad R \leftarrow R \cup S$
9: **return** R

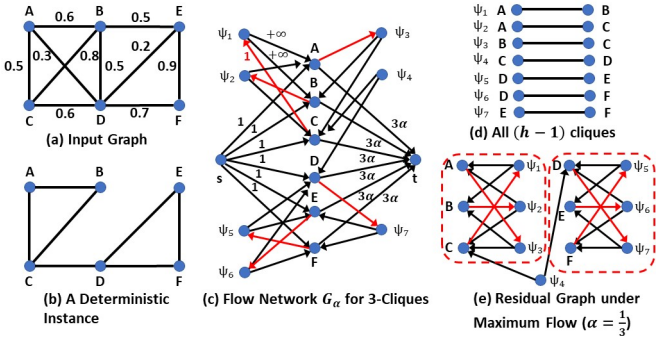


Fig. 4: Finding all 3-clique densest subgraphs in a possible world

Finally, in Line 8, Algorithm 3 enumerates one densest subgraph of G for every value of $C \cup des(C)$, where C is an SCC of $\mathcal{H}_{f^*}$, and the set $des(C)$ (resp. $anc(C)$) denotes the set of SCCs having a directed path from (resp. to) C in the SCC graph of $\mathcal{H}_{f^*}$. The detailed theoretical analyses for Algorithms 2 and 3 are given in Appendix A.

Example 5. *We shall compute all h-clique densest subgraphs ($h = 3$) in a possible world G (Figure 4(b)) of an uncertain graph $\mathcal{G}$ ((Figure 4(a)). The maximum 3-clique density ρ_h^* of any subgraph of G can be easily computed as $\frac{1}{3}$. A flow network G_α is constructed as in Figure 4(c). Instead of directly adding edges between nodes as for edge density (Example 4), a new set of nodes representing the $(h-1)$-cliques is added. A $(h-1)$-clique node ψ_i has a directed edge to each node contained in this $(h-1)$-clique, with infinite capacity. If a node v forms an h-clique with the $(h-1)$-clique ψ_i, a directed edge from v to ψ_i is added with capacity 1. For simplicity, we only show the capacities of the edges entering and leaving ψ_1 in Figure 4(c). In this example, the $(h-1)$-cliques are all edges in the possible world, as shown in Figure 4(d). After plugging in $\rho_h^* = \frac{1}{3}$ and computing the maximum flow, we identify a 3-clique densest subgraph $\{A, B, C, D, E, F\}$, and obtain a residual graph as shown in Figure 4(e). The remaining 3-clique densest subgraphs are $\{A, B, C\}$ and $\{D, E, F\}$. Each of them corresponds to an SCC of the residual graph.*

Remark. Our Algorithm 2 for computing all clique-densest subgraphs has similarities to the method of computing all edge-densest subgraphs in [46]. *However, there are also major differences from [46]:* As demonstrated in Example 5, our flow network $\mathcal{H}$ has one additional node for each $(h-1)$-clique in Λ (in addition to one for each node in V as in [46]), and the edges are from nodes in V to nodes in Λ (unlike [46], where the edges only exist between nodes in V). Thus, some non-trivial additions[4] need to be incorporated into the definitions and proofs in [46] to prove the correctness of our Algorithms 2 and 3. *This forms one of our novel technical contributions.*

Space complexity. Let G_c be the $(\lceil\tilde{\rho}\rceil, h)$-core in Lines 1-2 of Algorithm 2, with n_c and m_c the corresponding node and edge counts respectively. As each h-clique of G_c contains h distinct $(h-1)$-cliques, $|\Lambda| = \mathcal{O}(h\mu_h(G_c))$, where $\mu_h(G_c)$ is the number of h-cliques in G_c. Thus, the number of nodes in $\mathcal{H}$ is $\mathcal{O}(n_c + h\mu_h(G_c))$ and the number of edges is $\mathcal{O}\left(n_c + \sum_{v \in V} deg_{G_c}(v, h) + (h-1)h\mu_h(G_c)\right) = \mathcal{O}\left(n_c + h^2\mu_h(G_c)\right)$. Since these consume the most memory, the space complexity of Algorithm 2 is $\mathcal{O}\left(n_c + h^2\mu_h(G_c)\right)$.

When we adapt Algorithm 1 for h-Clique-MPDS, in addition to the memory required for storing the uncertain graph, the majority of the memory is consumed by Line 5, which invokes Algorithm 2. Denoting by μ_h^* the maximum number of h-cliques in the $(\lceil\tilde{\rho}\rceil, h)$-core of any possible world of $\mathcal{G}$, the overall space complexity of our method is $\mathcal{O}\left(m + n + h^2\mu_h^*\right)$.

Time complexity. For Lines 1-3 in Algorithm 2, the major step is enumerating all h-cliques in G, which takes $\mathcal{O}\left(hm\left(\frac{1}{2} \cdot \mu_h(G)\right)^{h-2}\right)$ time, where $\mu_h(G)$ is the number of h-cliques in G [56]. If T denotes the number of iterations involved in computing ρ_h^* using [57], the running time of Line 4 consists of optimizing the convex program and computing the maximum flow in each iteration [57], and requires $\mathcal{O}\left(2^T h\mu_h(G_c) + T\left(n_c h\mu_h(G_c) + n_c^3\right)\right)$ time, where n_c is as defined above. Lines 5-6 can be done in $\mathcal{O}\left(\left(n_c h\mu_h(G_c) + n_c^3\right)\right)$ time [5]. Line 7 takes $\mathcal{O}\left(n_c + h^2\mu_h(G_c)\right)$ time, as it involves finding the SCCs of $\mathcal{H}_{f^*}$, whose node and edge counts are stated above. In Line 8, each densest subgraph is enumerated exactly once (as proved in Appendix A), and each subgraph enumeration takes time linear in the corresponding number of nodes.

Denoting respectively by n_c^*, d^*, L^*, and T^*, the maximum number (in the $(\lceil\tilde{\rho}\rceil, h)$-core of any possible world) of nodes, densest subgraphs, nodes in a densest subgraph and iterations of [57] to compute ρ_h^*, and by μ_h^* the maximum number of h-cliques in any possible world, the overall time complexity of our method is $\mathcal{O}\left(\theta\left(hm\left(\frac{1}{2} \cdot \mu_h^*\right)^{h-2} + 2^{T^*} h\mu_h^* + T^*\left(n_c^* h\mu_h^* + n_c^{*3}\right) + d^* L^*\right)\right)$. Practically, $n_c^* \ll n$ and T^* is very small, as validated in our experiments (§ VI). For instance, $T^* = 11$, $\frac{n_c^*}{n} = 4.731 \times 10^{-4}$, $\frac{L^*}{n} = 2.049 \times 10^{-5}$, $\frac{\mu_h^*}{m} = 0.181$, and $d^* = 1$ for our large Twitter dataset, making our algorithm quite efficient with large-scale graphs.

Accuracy guarantee. We show (briefly in the following and in details in Appendix A) that Algorithm 2 correctly computes all clique-densest subgraphs in a deterministic graph. Thus, our overall accuracy guarantees for finding the top-k h-Clique-MPDSs in an uncertain graph remain the same as in §III-A.

Theorem 4. *Algorithm 2 enumerates each h-clique densest subgraph of a deterministic graph G exactly once.*

[4] In Appendix A, Definitions 10-11 and the proofs of Lemmas 9-10 are different from their counterparts in [46]. Moreover, Lemma 7 is newly derived and serves as a critical reason for not considering SCCs of the sink node.

Algorithm 4 Find all pattern-densest subgraphs

Input: Deterministic graph $G = (V, E)$, pattern $\psi = (V_\psi, E_\psi)$
Output: All pattern-densest subgraphs w.r.t. ψ in G
1: $\tilde{\rho} \leftarrow$ Density returned by the peeling method [5]
2: $G_c \leftarrow (\lceil \tilde{\rho} \rceil, \psi)$-core of G [5]
3: $\Lambda \leftarrow$ All ψ-instances in G_c [58]
4: $\rho^*_\psi \leftarrow \max_{S \subseteq V} \rho_\psi(S)$ [57]
5: $\mathcal{H} \leftarrow$ Algorithm 7 (G_c, ψ, Λ, ρ^*_h) [5]
6: $f^* \leftarrow$ Maximum flow in $\mathcal{H}$
7: $\mathcal{C} \leftarrow$ SCCs of the residual graph $\mathcal{H}_{f^*}$, excluding those of s and t
8: **return** Algorithm 3 ($\emptyset$, $\mathcal{C}$, V)

Proof Sketch. Lines 1-2 of Algorithm 2 are justified by the fact that the densest subgraph is in the $(\lceil \rho^*_h \rceil, h)$-core, and hence in the $(\lceil \tilde{\rho} \rceil, h)$-core [5]. The correctness of Line 4 follows from [57]. The main idea behind the remaining lines is that all h-clique densest subgraphs of G are hidden in the SCCs of the residual graph $\mathcal{H}_{f^*}$ of $\mathcal{H}$ under a maximum flow f^*. Any h-clique densest subgraph of G constitutes a minimum s-t cut in $\mathcal{H}$. Thus, this subgraph has no outgoing edge in $\mathcal{H}_{f^*}$. □

C. Pattern-MPDS: Approximate Algorithm

Algorithm 4 for finding *all* pattern-densest subgraphs (w.r.t. a given pattern $\psi = (V_\psi, E_\psi)$) is inspired by [46]. As earlier, *finding all pattern-densest subgraphs in a deterministic graph is a novel problem, and Algorithm 4 is a novel contribution.*

Algorithm 4 is similar to Algorithm 2 (§ III-B); in fact, the (k, h)-core (Definition 7) can be easily extended to the (k, ψ)-core (Line 2), and ρ^*_ψ can be computed in Line 4 by extending [57]. In Line 3, Λ refers to the set of all ψ-instances in G instead of cliques, which can be enumerated using the method in [58]. Moreover, the flow network $\mathcal{H}$ (constructed as in [5]; the pseudocode is included in Appendix B) contains one node for each group of ψ-instances with a common node set, instead of one for each instance, in order to reduce the memory footprint and running time. This forms the main difference between our method and the one in [46], resulting in some non-trivial additions which constitute one of our novel contributions (see Appendix B).

Accuracy guarantee. By a similar analysis as in Appendix A, it can be shown that Algorithm 4 correctly computes pattern-densest subgraphs w.r.t. ψ in a deterministic graph. The main difference is in the derivation of the capacity of a minimum cut in $\mathcal{H}$, which is quite different from the one in § III-B as mentioned above; this derivation is shown in Appendix B. Finally, since Algorithm 4 correctly computes all pattern-densest subgraphs w.r.t. ψ in a deterministic graph, our overall accuracy guarantees for finding the top-k Pattern-MPDS in an uncertain graph remain the same as in §III-A.

Space complexity. Let G_c be the $(\lceil \tilde{\rho} \rceil, \psi)$-core in Lines 1-2 of Algorithm 4, with n_c and m_c the corresponding node and edge counts respectively. Clearly $|\Lambda| = \mathcal{O}(\mu_\psi(G_c))$, where $\mu_\psi(G_c)$ is the number of ψ-instances in G_c. Thus, the number of nodes in $\mathcal{H}$ is $\mathcal{O}(n_c + \mu_\psi(G_c))$ and the number of edges is $\mathcal{O}(n_c + |V_\psi|\, \mu_\psi(G_c))$. Also, computing ρ^*_ψ (Line 4) requires $\mathcal{O}(n_c + |\Lambda|) = \mathcal{O}(n_c + \mu_\psi(G_c))$ space [5]. Since these constitute most of the memory consumed, the total space complexity of Algorithm 4 is $\mathcal{O}(n_c + |V_\psi|\, \mu_\psi(G_c))$.

Algorithm 5 Estimate all NDS in an uncertain graph

Input: Uncertain graph $\mathcal{G} = (V, E, p)$, positive integers k and l_m, no. of samples θ
Output: (Approximate) NDS
1: $CV \leftarrow \emptyset$
2: **for** $i = 1$ **to** θ **do**
3: Sample a possible world $G \sqsubseteq \mathcal{G}$
4: $S \leftarrow$ Maximum-sized densest subgraph in G
5: $CV \leftarrow CV \cup \{S\}$
6: **return** TFP(CV, k, l_m) [47]

When we adapt Algorithm 1 for Pattern-MPDS, in addition to the memory required for storing the uncertain graph, the majority of the memory is consumed by Line 5, which invokes Algorithm 4. Denoting by μ^*_ψ the maximum number of ψ-instances in the $(\lceil \tilde{\rho} \rceil, h)$-core of any possible world of $\mathcal{G}$, the overall space complexity is $\mathcal{O}\left(m + n + |V_\psi|\, \mu^*_\psi\right)$.

Time complexity. Assume that the enumeration of all ψ-instances in a possible world takes $\mathcal{O}(t_\psi)$ time. By a similar analysis as in § III-B, denoting respectively by n^*_c, d^*, L^*, μ^*_ψ, and T^*, the maximum number (in the $(\lceil \tilde{\rho} \rceil, \psi)$-core of any possible world) of nodes, densest subgraphs, nodes in a densest subgraph, ψ-instances, and iterations of [57] to compute ρ^*_ψ, the overall time complexity of our method is $\mathcal{O}\left(\theta\left(t_\psi + 2^{T^*}\, |V_\psi|\, \mu^*_\psi + T^*\left(n^*_c \mu^*_\psi + {n^*_c}^3\right) + d^* L^*\right)\right)$.

Remark. For larger graphs and bigger patterns ψ, we find that the enumeration of all ψ-instances (which is necessary to compute the densest subgraphs in a possible world) can be expensive. In such cases, we resort to a heuristic method in which we enumerate some reasonably dense subgraphs (instead of all densest ones) using [5]. Specifically, we use a method different from Algorithm 4, which runs core decomposition w.r.t. ψ. If k_{max} denotes the maximum core number, then the (k_{max}, ψ)-core is a reasonably dense subgraph. In particular, the (k_{max}, ψ)-core's density is at least $\frac{1}{|V_\psi|}$ times the maximum density of any subgraph [5]. Based on this, we return the (k_{max}, ψ)-core and all intermediate subgraphs (obtained during core decomposition) having greater densities. Experimental results (§ VI-G) show that this heuristic method yields good-quality solutions with higher efficiency.

IV. Approximate Solution for Nucleus Densest Subgraphs

In this section, we *convert the* NDS *(nucleus densest subgraphs) problem into the widely-studied closed frequent itemset mining problem* and develop an *approximate method* (Algorithm 5) to find the top-k NDS for all three notions of density: edge, clique, and pattern density.

The algorithm first runs θ independent rounds: Sample a possible world $G \sqsubseteq \mathcal{G}$ and insert, into the set of candidate node sets CV, the maximum-sized densest subgraph of G[5]. For a node set $U \subseteq V$, let $\widehat{\gamma}(U)$ denote the estimated densest subgraph containment probability of U, which is computed as the fraction of node sets in CV which contain U. Then a

[5]By a trivial generalization of [59] to all density notions, the union of the node sets of all densest subgraphs of a deterministic graph G induces the maximum-sized (w.r.t. node count) densest subgraph in G. Thus, a node set is contained in a densest subgraph of G if and only if it is contained in the maximum-sized densest subgraph of G.

closed frequent itemset mining algorithm (e.g., TFP [47]) is applied to compute the top-k closed node sets in CV of size at least l_m having the largest values of $\widehat{\gamma}(\cdot)$. A node set is said to be closed w.r.t. $\widehat{\gamma}(\cdot)$ if it has no superset with the same value of $\widehat{\gamma}(\cdot)$ (i.e., is contained in the same number of node sets in CV). Here, l_m is a user input decided based on the minimum desired size of a returned subgraph.

The maximum-sized densest subgraph in a deterministic graph (Line 4) can be computed using parts of the methods in § III. For edge density, we terminate after computing the SCCs of the residual graph under a maximum flow [46]. For clique and pattern densities, we terminate after computing the value of the maximum density of a subgraph, since we also get the maximum-sized densest subgraph in this process [57].

Space complexity. The major memory cost is due to the flow network in each iteration of Algorithm 5. Therefore, the overall space complexity is the same as in § III.

Time complexity. The time complexity is similar to the ones in § III, plus that for computing the closed frequent node sets by TFP, which is reasonable in our experiments (§ VI-I). Note that, for Pattern-NDS on our larger graphs, we use the heuristic method of § III-C in Line 4 of Algorithm 5.

Accuracy guarantee. We theoretically analyze the sample size θ to return the true top-k node sets with a high probability. First, we prove that the true top-k node sets are likely to be closed w.r.t. $\widehat{\gamma}(\cdot)$ after θ rounds (Theorem 5), which is necessary (but not sufficient) for them to be finally returned. Second, we ensure that those sets indeed have the k largest *estimated* densest subgraph containment probabilities (Theorem 6), which guarantees that they are finally returned.

Theorem 5. *Given an uncertain graph $\mathcal{G}$ and positive integers k and l_m, let $V_1, \ldots, V_k$ denote the true top-k closed node sets w.r.t. $\gamma(\cdot)$ of size at least l_m having the highest densest subgraph containment probabilities. For each $i \in [1, k]$, let $\mathbf{G}(V_i)$ denote the set of all possible worlds of $\mathcal{G}$ whose densest subgraphs contain V_i. Define $\mathbb{G} = \bigcup_{i=1}^{k} \mathbf{G}(V_i)$. Then*

$$\Pr\left(V_1, \ldots, V_k \text{ are closed w.r.t. } \widehat{\gamma}(\cdot)\right) \geq 1 - \sum_{G \in \mathbb{G}} \left(1 - \Pr(G)\right)^{\theta} \quad (14)$$

Proof. For each $i \in [1, k]$, if all possible worlds in $\mathbf{G}(V_i)$ are sampled at least once, then for each set $V' \supset V_i$, there is at least one node set (densest subgraph) in CV containing V_i but not V', since V_i is closed w.r.t. $\gamma(\cdot)$. In that case, V_i is also closed w.r.t. $\widehat{\gamma}(\cdot)$. Thus, using this and the union bound,

$$\Pr\left(V_1, \ldots, V_k \text{ are closed w.r.t. } \widehat{\gamma}(\cdot)\right) \geq \Pr\left(\bigwedge_{G \in \mathbb{G}} G \text{ is sampled}\right)$$

$$= 1 - \Pr\left(\bigvee_{G \in \mathbb{G}} G \text{ is never sampled}\right) \geq 1 - \sum_{G \in \mathbb{G}} \left(1 - \Pr(G)\right)^{\theta}$$

□

From Theorem 5, if the existence probabilities of the possible worlds whose densest subgraphs contain the true top-k node sets and the sample size θ are reasonably large, they are all highly likely to be closed w.r.t. $\widehat{\gamma}(\cdot)$, which is necessary for all of them to be returned.

Theorem 6. *Let $V_1, \ldots, V_{k+1}$ denote the true top-$(k+1)$ closed node sets of size at least l_m having the highest densest subgraph containment probabilities, and let $\mathcal{CV}$ denote the set of candidate node sets after θ rounds. Define $mid = \frac{1}{2}\left[\gamma\left(V_k\right) + \gamma\left(V_{k+1}\right)\right]$ and*

$$d_U = \begin{cases} \gamma(U) - mid & \text{if } U \in \{V_1, \ldots, V_k\} \\ mid - \gamma(U) & \text{otherwise} \end{cases} \quad (15)$$

For each $i \in [1, k]$, let $\mathbf{G}(V_i)$ denote the set of all possible worlds of $\mathcal{G}$ whose densest subgraphs contain V_i. Define $\mathbb{G} = \bigcup_{i=1}^{k} \mathbf{G}(V_i)$ and $\mathcal{CV}$ as the set of all closed node sets w.r.t. $\widehat{\gamma}(\cdot)$ of size at least l_m. Then, the probability that $V_1, \ldots, V_k$ are returned by Algorithm 5 is at least

$$\left[1 - \sum_{G \in \mathbb{G}} \left(1 - \Pr(G)\right)^{\theta}\right] \left[1 - \sum_{U \in \mathcal{CV}} \exp\left(-2 d_U^2 \theta\right)\right] \quad (16)$$

Proof. Let $\mathcal{CV}' = \mathcal{CV} \setminus \{V_1, \ldots, V_k\}$. We have:

$$\Pr(V_1, \ldots V_k \text{ are returned}) \geq \Pr(V_1, \ldots, V_k \text{ are closed w.r.t. } \widehat{\gamma}(\cdot)) \times$$
$$\Pr\left(\left(\bigwedge_{U \in \{V_1, \ldots, V_k\}} \widehat{\gamma}(U) > mid\right) \wedge \left(\bigwedge_{U \in \mathcal{CV}'} \widehat{\gamma}(U) < mid\right)\right) \quad (17)$$

Similar to Lemma 1, it can be proved that $\mathbb{E}[\widehat{\gamma}(U)] = \gamma(U)$ $\forall U \subseteq V$. Using the union bound and Hoeffding's inequality,

$$\Pr\left(\left(\bigwedge_{U \in \{V_1, \ldots, V_k\}} \widehat{\gamma}(U) > mid\right) \wedge \left(\bigwedge_{U \in \mathcal{CV}'} \widehat{\gamma}(U) < mid\right)\right)$$
$$= 1 - \Pr\left(\left(\bigvee_{U \in \{V_1, \ldots, V_k\}} \widehat{\gamma}(U) \leq mid\right) \vee \left(\bigvee_{U \in \mathcal{CV}'} \widehat{\gamma}(U) \geq mid\right)\right)$$
$$\geq 1 - \sum_{U \in \{V_1, \ldots, V_k\}} \Pr\left(\widehat{\gamma}(U) \leq mid\right) - \sum_{U \in \mathcal{CV}'} \Pr\left(\widehat{\gamma}(U) \geq mid\right)$$
$$= 1 - \sum_{U \in \{V_1, \ldots, V_k\}} \Pr\left(\widehat{\gamma}(U) - \gamma(U) \leq -d_U\right) - \sum_{U \in \mathcal{CV}'} \Pr\left(\widehat{\gamma}(U) - \gamma(U) \geq d_U\right)$$
$$\geq 1 - \sum_{U \in \mathcal{CV}} \exp\left(-2 d_U^2 \theta\right) \quad (18)$$

Finally, plugging (14) and (18) into (17), we obtain (16). □

From Theorem 6, if the densest subgraph containment probabilities of the true top-k node sets are reasonably larger than the other closed node sets w.r.t. $\widehat{\gamma}(\cdot)$ of size at least l_m, and if the sample size θ is sufficiently large, the true top-k sets are returned by Algorithm 5 with a high probability.

V. Related Work

In this section, we first revisit the densest subgraph discovery problem in deterministic graphs. Then, we discuss the existing attempt that extends the edge density to uncertain graphs by considering the maximum expected density. Finally, we state several close notions for cohesive and dense substructures in uncertain graphs, including core, truss, maximal cliques, clustering, and highly reliable subgraphs.

Densest subgraph in a deterministic graph. Given an undirected, unweighted, deterministic graph, the original densest subgraph (DS) problem [1] finds a subgraph with the highest edge-density, exact solutions to which are based on min-cuts in flow-networks [1], [5] and linear programming [1], [2]. Since the computation of maximum flow has a high time

complexity, researchers proposed approximate algorithms with theoretical guarantees, e.g., [2], [5], [60], [61]. Variants of the edge-density-based DS problem were also studied, such as triangle-density, clique-density, pattern-density, and edge-surplus based DS [5], [20], [57], [19], [3], densest k-connected subgraph [62] and size-bounded DS [63], top-k overlapping DS [64], top-k DS maintenance on dynamic graphs [65], [60], [61], locally DS [66], robust DS [67], density-friendly graph decomposition [68], [69], DS on directed [70], bipartite [71], multilayer graphs [72] etc. For surveys and tutorials, we refer to [73], [4]. Enumerating all the densest subgraphs based on edge-density in a deterministic graph has been recently studied in [46], which we use as a subroutine to find the MPDS of an uncertain graph in our problem. We notice that the problems of enumerating all clique-DS and pattern-DS in a deterministic graph were not studied in the literature. Thus, as additional technical contributions, we develop novel, exact algorithms for efficiently enumerating all clique-DS and all pattern-DS in a deterministic graph, and use them as subroutines to respectively find the h-Clique-MPDS and the Pattern-MPDS of an uncertain graph in our problem.

Expected edge densest subgraph. The expected edge density of a node set U in an uncertain graph is the expectation of the edge density of the subgraph induced by U across all possible worlds. Zou [44] designed a polynomial-time algorithm to find the subgraph with the maximum expected edge density in an uncertain graph using maximum flow techniques. In a graph where each edge weight distribution follows a given mean (reward) and variance (risk), Tsourakakis et al. [45] find a node set whose induced subgraph has high average reward (i.e., expected density) and low average risk. As shown in Example 1 and our experiments (§VI-B and §VI-E), the expected edge densest subgraph is different from the MPDS: a subgraph of an uncertain graph having the maximum expected edge density may induce densest subgraphs *only in a few* possible worlds of the graph. Such a subgraph can be large with many low-probability edges or loosely connected nodes.

Core and truss decompositions in an uncertain graph. As cohesive and dense substructures finding, core and truss decompositions are popular. The k-core (resp. k-truss) of a graph is a maximal subgraph in which every node is connected to at least k other nodes (resp. each edge participates in at least $(k-2)$ triangles) within that subgraph. These notions have been extended to uncertain graphs, returning those subgraphs satisfying the above conditions with probability at least a threshold [40], [74], [75], [76], [41], [77], [78]. The innermost cores and trusses have been used in applications such as task-driven team formation due to their higher densities [41], [40]. However, they are different from the MPDS (§VI-B and §VI-E). Unlike the MPDS, they do not find the node set most likely to induce a densest subgraph in the uncertain graph.

Top-k maximal cliques in an uncertain graph. A clique is a set of nodes with each pair connected by an edge. [37], [38], [39], [79] study enumeration of maximal cliques in uncertain graphs. Densest subgraphs are not necessarily cliques.

TABLE II: Characteristics of our datasets

Name	n	m	Type	Edge Prob: Mean, St. Dev., Quart.
Karate Club	34	78	Social	0.25, 0.09, {0.18, 0.26, 0.33}
Intel Lab	54	969	Device	0.33, 0.19, {0.16, 0.27, 0.44}
LastFM	6 899	23 696	Social	0.33, 0.19, {0.16, 0.27, 0.44}
Homo Sapiens	18 384	995 916	Bio	0.32, 0.21, {0.18, 0.24, 0.34}
Biomine	1 045 414	6 742 939	Bio	0.27, 0.21, {0.12, 0.22, 0.36}
Twitter	6 294 565	11 063 034	Social	0.14, 0.10, {0.10, 0.10, 0.19}
Friendster	65 608 366	1 806 067 135	Social	0.005, 0.013, {0.001, 0.003, 0.005}

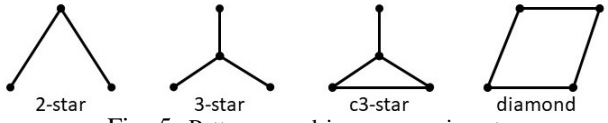


Fig. 5: Patterns used in our experiments

Node clustering in an uncertain graph. k-means and k-medians approaches have been extended to node clustering in uncertain graphs [80], [42], [81]: Partition the nodes into k clusters such that some notion of path-based connection probability (e.g., reliability) between each node and its cluster's center (or, across each pair of nodes within every cluster) is maximized. Notice that clustering methods optimize the path-based connection probability between nodes inside a cluster, and do not optimize the density of a subgraph.

Highly reliable subgraphs in an uncertain graph. Jin et al. [82] have developed a mining-based approach to discover highly reliable subgraphs in an uncertain graph. A highly reliable subgraph denotes a node set that has a high probability of remaining connected via some paths in possible worlds of the uncertain graph. This method deals with path-based connectivity between nodes, and is hence not suitable to identify densest subgraphs, such as MPDS in our work.

VI. Experimental Results

We run experiments to demonstrate the effectiveness and efficiency of our methods. Our C++ code [83] is executed on one core of a 512GB, 2.4GHz Xeon server running Ubuntu.

A. Experimental Setup

Datasets. We conduct experimental evaluations on six real-world graphs (Table II). **(1)** Karate Club [84] is a social network of a university karate club. Nodes are club members and edges denote member interactions. **(2)** Intel Lab [85] is a collection of sensor communication data with 54 sensors deployed in the Intel Berkeley Research Lab between February 28 and April 5, 2004. **(3)** LastFM [86] is a musical social network where users listen to music and share with friends. An edge between two users exists if they communicate at least once. **(4)** Homo Sapiens [87] is a protein-protein interaction network, where nodes are proteins and edges are interactions between proteins. **(5)** Biomine [88] is constructed by integrating cross-references from several biological databases. Nodes represent biological concepts such as genes, proteins etc., and edges denote real-world phenomena between two nodes, e.g.,

TABLE III: Densest subgraph containment probabilities of the NDS, the expected densest subgraph (EDS), the innermost η-core and γ-truss ($\eta = \gamma = 0.1$); Expected densities of the NDS and EDS

Dataset	Containment Probability				Expected Density	
	NDS	EDS	Core	Truss	NDS	EDS
Homo Sapiens	**1**	0.05	**1**	**1**	54	**54.62**
Biomine	**1**	0.01	0.99	0	46.45	**48.02**
Twitter	**1**	0	0.95	0	37.65	**38.64**

TABLE IV: Densest subgraph probabilities of the MPDS, the expected densest subgraph (EDS), the innermost η-core and γ-truss ($\eta = \gamma = 0.1$); Expected densities of the MPDS and EDS

Dataset	Densest Subgraph Probability				Expected Density	
	MPDS	EDS	Core	Truss	MPDS	EDS
Karate Club	**0.012**	0	0	0	0.703	**0.75**
Intel Lab	**0.078**	0.01	0.01	0	3.246	**3.25**
LastFM	**0.075**	0	0.04	0.02	0.667	**0.86**

a gene "code" for a protein. **(6)** Twitter [89] is a widely used social network where nodes are users and edges are retweets. **(7)** Friendster [90] is a social network where nodes are users and edges denote friendships.

Edge probability models. We adopt various widely used models for generating the edge probabilities in our evaluation. In **(1)** Karate Club, **(6)** Twitter and **(7)** Friendster, we assign the probability of an edge as $1-e^{-\frac{t}{\mu}}$, which is an exponential cdf of mean μ to the number t of communications between the two users. We set $\mu = 20$ [91]. In **(2)** Intel Lab, a (real) edge probability denotes the fraction of messages from the sender that successfully reached the receiver [85], [36]. In **(3)** LastFM, the probability of any edge is the reciprocal of the larger of the out-degrees of its source and target nodes [91]. In **(4)** Homo Sapiens, an edge probability represents the confidence on the existence of the corresponding interaction, based on real biological experiments [87]. In **(5)** Biomine [88], an edge probability quantifies the existence of a phenomenon between the two endpoints, which was a combination of three criteria: relevance, informativeness, and confidence on the existence of a specific relationship [35], [40], [41].

Methods compared. We compare our MPDS and NDS algorithms with those for computing the expected densest subgraph [44], (k, η)-core [40], and (k, γ)-truss [41] (§ VI-B). As discussed in § II-C, we show the results for MPDS on the three smaller datasets and NDS on the three larger ones. Additionally, on some small synthetic graphs, we compare our MPDS approximation algorithms with the corresponding exact ones (§ VI-H). For sampling possible worlds, we compare our employed *Monte Carlo (MC)* method with *Lazy Propagation* [54] and *Recursive Stratified Sampling* [55] (§ VI-I).

Parameters. • h **for Clique-MPDS/NDS**: We vary $h \in \{3, 4, 5\}$ [20]. Notice that $h = 2$ denotes an edge. • ψ **for Pattern-MPDS/NDS**: We vary $\psi \in \{$2-star, 3-star, c3-star, diamond$\}$ [5], as shown in Figure 5. • **Top-k MPDSs**: We vary $k \in \{1, 5, 10\}$, with default value 1. • **Top-k NDSs**: We vary $k \in \{1, 5, 10, 50, 100\}$. • **Minimum size threshold** l_m: We vary $l_m \in [1, 750]$. Beyond that range, no NDS is returned for any of our datasets. • **Number of sampled possible worlds** θ: We vary $\theta \in \{2^0 \times 10, 2^1 \times 10, \ldots, 2^8 \times 10\}$. The default value is chosen as in § VI-I.

TABLE V: Probabilistic density of our proposed subgraphs (MPDS for the two smaller datasets and NDS for the two larger ones) and of existing dense subgraphs in uncertain graphs

Dataset	Probabilistic Density			
	MPDS/NDS	EDS	Core	Truss
Karate Club	**0.281**	0.095	0.073	0.134
LastFM	**0.333**	0.007	0.008	0.013
Biomine	**0.546**	0.191	0.212	0.538
Twitter	**0.789**	0.042	0.121	0.781

TABLE VI: Probabilistic clustering coefficient of our proposed subgraphs (MPDS for the two smaller datasets and NDS for the two larger ones) and of existing dense subgraphs in uncertain graphs

Dataset	Probabilistic Clustering Coefficient			
	MPDS/NDS	EDS	Core	Truss
Karate Club	**0.284**	0.150	0.094	0.158
LastFM	**0.333**	0.002	0.022	0.257
Biomine	**0.546**	0.203	0.217	0.539
Twitter	**0.775**	0.142	0.253	0.768

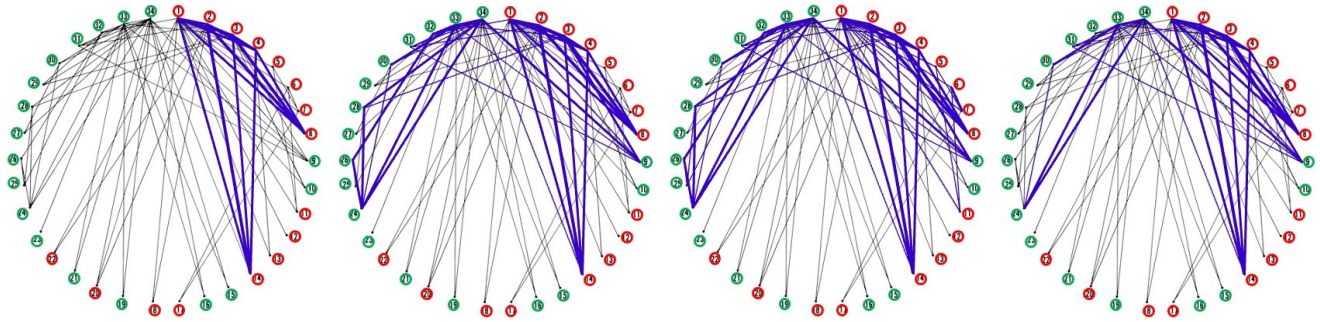
B. Comparison with Expected Density, Core, and Truss Decompositions in Uncertain Graphs

For each of our larger datasets, we compare our NDS with some existing or close notions of densest subgraphs in uncertain graphs: expected densest subgraph [44], innermost core [40], and innermost truss [41]. Specifically, we compare the (approximate) densest subgraph containment probabilities of the following: • The (maximal) node set returned by Algorithm 5 with the highest frequency of being contained in the densest subgraphs of the generated possible worlds; • the expected densest subgraph (EDS); • for a given η, the innermost η-core, i.e., the (k, η)-core with the largest value of k; • for a given γ, the innermost γ-truss (analogous to cores). As shown in Table III, the containment probability of the η-core is comparable to (yet not greater than) that of the NDS for all datasets, in contrast to the EDS and the γ-truss. This makes sense for the following reason. The innermost η-core is likely to be an innermost core (and hence a reasonably dense subgraph [5]) of a possible world of the input graph. However, the same cannot be said about the other subgraphs. Table III demonstrates that our solution produces the most optimal node set with the highest densest subgraph containment probability compared to the other approaches.

In addition, for each of our smaller datasets, we compare the densest subgraph probability of the MPDS with those of the EDS, innermost η-core and innermost γ-truss. As shown in Table IV, the MPDS outperforms the other subgraphs.

Since the EDS performs very poorly for all datasets, for fairness, we compare the expected densities of the EDS and our NDS/MPDS. Tables III and IV show that our solutions produce subgraphs with expected densities comparable to the optimal values, thereby showing that our returned subgraphs are good even with respect to expected density.

We also consider two *external* evaluation metrics: • *Probabilistic Density* $PD(U)$ [41] for capturing the cohesiveness of a probabilistic subgraph U, which is defined as the weighted sum of existing edges divided by the maximum number of possible edges this subgraph can have (Equation 19); and • *Probabilistic Clustering Coefficient* $PCC(U)$ [92] for measuring how well the nodes in a probabilistic subgraph U cluster together, which is computed as three times the weighted sum of all possible triangles divided by the weighted sum of all

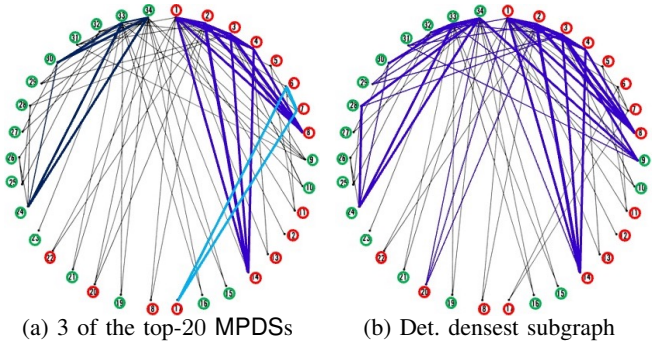


(a) MPDS (b) Expected Densest Subgraph (c) Innermost core (d) Innermost truss

Fig. 6: Case studies to compare with existing dense subgraphs in uncertain graphs on the Karate Club dataset. The relevant subgraphs are in blue, while the colour of a node denotes its ground-truth community [84]. The thickness of each edge is proportional to its probability.

TABLE VII: Densest subgraph probabilities of the MPDS and the deterministic densest subgraph (DDS)

Subgraph	Karate Club	Intel Lab	LastFM
MPDS	**0.012**	**0.078**	**0.075**
DDS	≈ 0	0.044	≈ 0



(a) 3 of the top-20 MPDSs (b) Det. densest subgraph

Fig. 7: 3 MPDSs (different colors) and the densest subgraph in the deterministic version of the Karate Club dataset. The color of a node denotes the community to which it belongs [84]. We only show 3 of the top-20 MPDSs for better visualization; the others have node sets that are subsets of those shown. The thickness of each edge is proportional to its probability.

neighboring edge pairs. The weights are existence probabilities of edges/triangles/neighboring edge pairs, assuming independence among edges (Equation 20). V_U, E_U, and Δ_U denote the set of nodes, edges, and triangles in U, respectively.

$$PD(U) = \frac{2\sum_{e\in E_U} p(e)}{|V_U|(|V_U|-1)} \tag{19}$$

$$PCC(U) = \frac{3\sum_{\triangle_{uvw}\in\Delta_U} p(u,v)p(u,w)p(v,w)}{\sum_{(u,v),(u,w)\in E_U, v\neq w} p(u,v)p(u,w)} \tag{20}$$

Tables V-VI demonstrate that our NDS/MPDS significantly outperforms other dense subgraph notions based on both probabilistic density and probabilistic clustering coefficient, implying that the NDS/MPDS is much more cohesive, i.e., most of the possible edges induced by the NDS/MPDS node set tend to exist, and the nodes in the NDS/MPDS cluster together. Only the innermost truss achieves slightly lower results on the two larger datasets.

C. Comparison between MPDS and the Densest Subgraph in the Deterministic Version of Input Graph

As shown in Table VII, the (estimated) densest subgraph probability of the MPDS is much higher than that of the densest subgraph in the deterministic version of an uncertain graph (denoted by DDS). The reason is that the rich information encoded in the edge probabilities is ignored by the DDS. In practice, nodes may be densely connected via low-probability edges, e.g., due to noise. Our proposed MPDS can capture and filter out this, leading to more useful results in uncertain graphs compared to the DDS.

TABLE VIII: Distribution (mean, standard deviation, quartiles) of the number of densest subgraphs (edge, 3-clique, diamond) across all sampling rounds in our MPDS algorithm

Notion	Karate Club	LastFM
Edge	1.12, 0.54, {1, 1, 1}	2613.24, 22825.66, {15, 127, 1023}
3-Clique	1.35, 0.91, {1, 1, 1}	1880.74, 22134, {31, 127, 511}
Diamond	1.18, 0.71, {1, 1, 1}	3.52, 9.6, {1, 1, 3}

TABLE IX: Average densest subgraph probabilities of the top-10 MPDSs returned by computing all vs only one densest subgraph in each sampled world

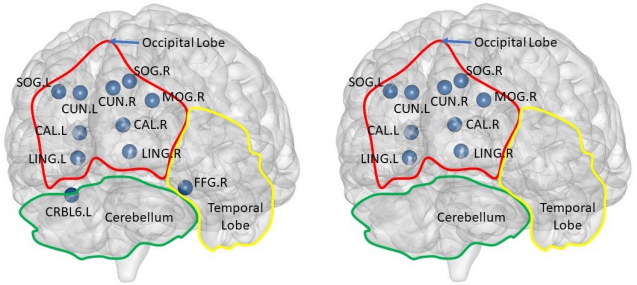
Dataset	Edge		3-Clique		Diamond	
	All	One	All	One	All	One
Karate Club	**0.006**	0.005	**0.019**	0.018	**0.011**	0.01
LastFM	**0.054**	0.004	**0.08**	0.004	**0.009**	0.007

TABLE X: Purity (§ VI-E) of the node sets in the top-k MPDSs and in the existing notions of dense subgraphs over Karate Club uncertain graph. There are only two cores and two trusses in this graph; thus the entries for $k > 2$ are empty for those subgraphs.

Top-k	Purity			
	MPDS	EDS	Core	Truss
1	**1**	0.6	0.5	0.538
2	**1**	0.6	0.515	0.536
5	**1**	0.749	-	-
10	**1**	0.699	-	-

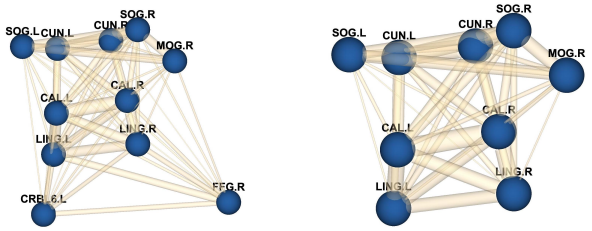
D. Considering All vs. One Densest Subgraph(s) in Each Sampled World

As shown in Table VIII, the number of densest subgraphs in a deterministic sample can be very large in practice (e.g., in LastFM). Thus, if we compute only one densest subgraph (instead of all such subgraphs) in each world, the frequency of a particular subgraph in the candidate set (and hence its estimated densest subgraph probability) will reduce. Table IX presents that the average (estimated) densest subgraph probability of the top-10 results reduces if we compute only one densest subgraph. This gap can be up to 20× when the number of densest subgraphs is huge (LastFM as per



(a) Typically developed (b) ASD-affected

Fig. 8: Node sets of 3-clique MPDSs in brain networks. The colored boundaries denote the cerebellum, occipital, and temporal lobes.



(a) Typically developed (b) ASD-affected

Fig. 9: 3-clique MPDSs in brain networks. The thickness of each edge is proportional to its probability.

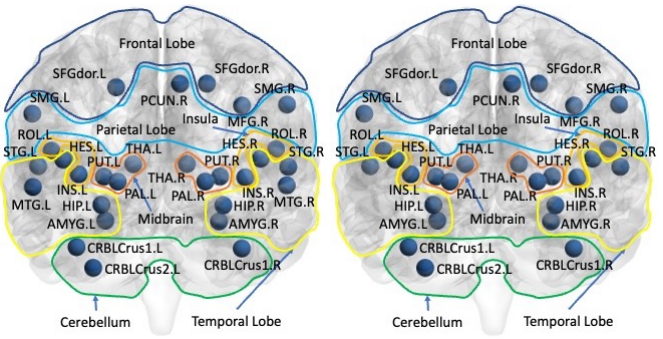
Table VIII). Therefore, computing all densest subgraphs in each sampled world is critical in finding MPDSs.

E. Case Studies: Karate Club Network

Densest subgraphs in social networks can correspond to communities [6], filter bubbles and echo chambers [7], [8].

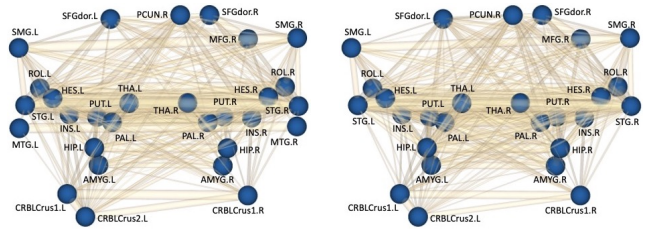
Comparison of the MPDS with the deterministic densest subgraph. Figure 7 shows that the densest subgraph given by the deterministic version of the Karate Club dataset has a much larger size with many low-probability edges and contains nodes from both ground-truth communities [84]. On the contrary, each of the MPDSs contains nodes from only one ground-truth community and has edges with higher probabilities. This demonstrates that MPDS is more powerful in detecting communities in uncertain graphs than simply considering the densest subgraph in the deterministic version.

Comparison with other notions of densest subgraphs in uncertain graphs. Table VI already shows that the MPDS of the Karate Club dataset has a much higher probabilistic clustering coefficient than the other existing dense subgraph notions. We show that the MPDS also represents a more meaningful and concise community (of club members) than the other subgraphs. All four subgraphs are shown in Figure 6. Notice that the MPDS only contains nodes from one single ground-truth community [84] and has edges with higher probabilities, in contrast to all the other subgraphs which contain nodes from both communities and have many low-probability edges. Moreover, in Table X, we report the average purity (i.e., highest fraction of nodes from the same ground-truth community [84] in a node set) of the top-k (up to $k = 10$) subgraphs returned by each notion, and observe that MPDSs always achieve 100% purity. Thus, users can retrieve the top-k MPDSs to identify high-quality communities. This case study



(a) Typically developed (b) ASD-affected

Fig. 10: Node sets of 3-clique EDSs in brain networks. The colored boundaries denote various brain regions as shown.



(a) Typically developed (b) ASD-affected

Fig. 11: 3-clique EDSs in brain networks. The thickness of each edge is proportional to its probability.

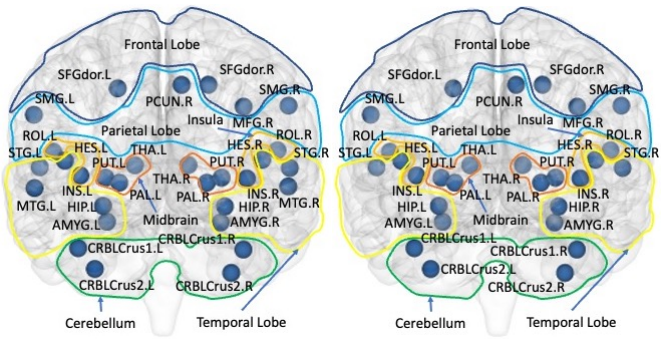
highlights the importance of computing MPDSs despite other notions of dense subgraphs in uncertain graphs.

F. Case Studies: Brain Networks

A brain network can be defined as an uncertain graph where nodes are brain regions of interest (ROIs), an edge indicates co-activation between two ROIs, and an edge probability indicates the strength of the co-activation signal. Dense subgraphs in brain networks can represent brain regions responding together to stimuli [9] or related to diseases [10].

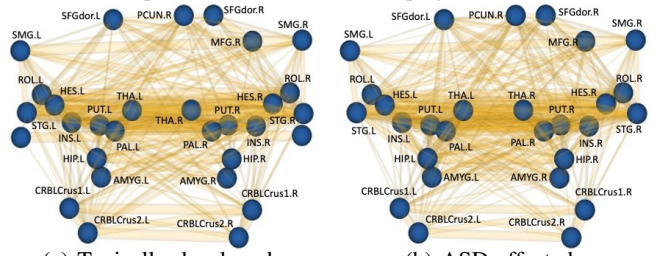
The dataset we use [93] contains data of 52 *Typically Developed* (TD) children and 49 children suffering from *Autism Spectrum Disorder* (ASD). Each subject is represented as a graph over 116 nodes (ROIs). $\mathcal{G}_{ASD}$ and $\mathcal{G}_{TD}$ are uncertain graphs, defined over the same set of nodes as the original ones, while the probability of each edge is the average of those of the same edge across all graphs in the ASD and TD groups.

Using BrainNet Viewer [94], we show the 3-clique MPDSs for both $\mathcal{G}_{TD}$ and $\mathcal{G}_{ASD}$ in Figures 8 and 9. The MPDS in $\mathcal{G}_{ASD}$ lies entirely in the occipital lobe, in contrast to that in $\mathcal{G}_{TD}$, which also contains one node in the temporal lobe and one in the cerebellum. Besides, the MPDS in $\mathcal{G}_{ASD}$ is more symmetrical than that in $\mathcal{G}_{TD}$, since the former has only one node (MOG.R) without its counterpart in the other hemisphere, while the latter has two more such nodes (CRBL6.L and FFG.R). This is consistent with the results of different works in neuroscience indicating that, in contrast to typically developed brains, those affected by ASD are characterized by under-connectivity between distant brain regions and over-connectivity between closer ones [95], [96], and that the hemispheres of ASD-affected brains are more symmetrical than those of typically developed ones [97]. Our consistent findings



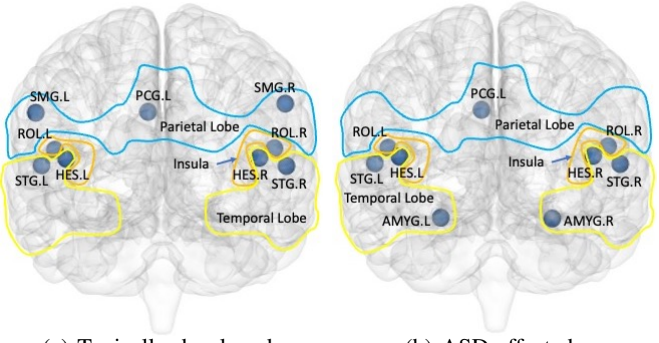
(a) Typically developed (b) ASD-affected

Fig. 12: Node sets of innermost cores in brain networks. The colored boundaries denote various brain regions as shown. Not all nodes are shown due to space constraints in the 2D projection of the 3D brain.



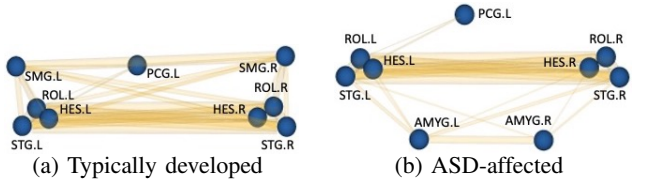
(a) Typically developed (b) ASD-affected

Fig. 13: Innermost cores in brain networks. The thickness of each edge is proportional to its probability. Not all nodes are shown due to space constraints in the 2D projection of the 3D brain.



(a) Typically developed (b) ASD-affected

Fig. 14: Node sets of innermost trusses in brain networks. The colored boundaries denote various brain regions as shown.



(a) Typically developed (b) ASD-affected

Fig. 15: Innermost trusses in brain networks. The thickness of each edge is proportional to its probability.

underline the importance of finding MPDSs in uncertain brain networks that can differentiate healthy and autistic brains.

We now show that the expected (3-clique) densest subgraph (EDS) cannot characterize and distinguish autistic brains, unlike our proposed MPDS. Note that in the existing literature, the EDS notion was defined only for edge density [44]; however, we show that it can be extended to clique and pattern densities (in Appendix C). Figures 10 and 11

TABLE XI: Densest subgraph containment probabilities and running times of approximate and heuristic Pattern-NDS; Karate Club

Pattern	Containment Probability		Running Time (seconds)	
	Approx.	Heuristic	Approx.	Heuristic
2-star	**0.625**	0.6	0.0561	**0.0129**
3-star	**0.55**	0.525	0.0242	**0.0101**
c3-star	**0.3313**	0.262	0.0244	**0.0109**
diamond	**0.8**	0.7687	0.0212	**0.0093**

TABLE XII: Densest subgraph containment probabilities and running times of approximate and heuristic Edge-NDS; Friendster

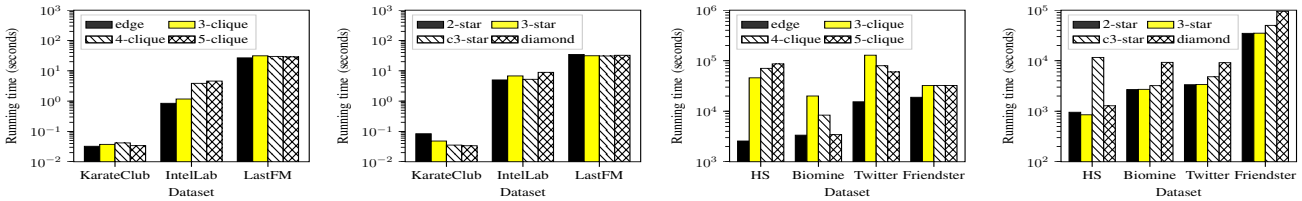
Method	Containment Probability	Running time (hours)
Approximate	**0.025**	21.216
Heuristic	0.021	**4.97**

present 3-clique EDSs in both $\mathcal{G}_{TD}$ and $\mathcal{G}_{ASD}$. Both of them span as many as 9 brain regions and are similar (w.r.t. symmetry) since both only have the same 3 nodes (PCUN.R, MFG.R, CRBLCrus2.L) without their counterparts in the other hemispheres. These contradict the characteristics of autistic brains in the biological literature [95], [96], [97] and fail to distinguish them from normal ones. This is consistent with our observation in Example 1 that EDSs tend to be very large and span several unimportant nodes, and hence can be less meaningful in real-world applications. Our MPDS is more powerful than the existing expectation-based notion EDS in analyzing autistic brains. Also, as shown in Figures 12-13 (resp. 14-15), the innermost cores (resp. trusses) for both brains span multiple brain regions and are similar (w.r.t. symmetry) since they have the same nodes PCUN.R, MFG.R (resp. PCG.L) without their counterparts in the other hemispheres. Thus, even the innermost cores and trusses cannot well-characterize and distinguish autistic brains.

G. Efficiency

We report the running times of our methods in Figure 16. As shown in Figures 16(a) and 16(c), the running times for edge density are smaller than those for clique density. This is because the flow networks involved in computing edge densest subgraphs are much smaller; they only contain nodes for each node in the sampled possible worlds (§ III-A), in contrast to those for h-clique densest subgraphs which also contain nodes for each $(h-1)$-clique contained in h-cliques in the sampled possible worlds (§ III-B). However, there is no clear winner among 3-clique, 4-clique, and 5-clique. This is because, even if larger cliques take a longer time to enumerate, smaller cliques can be more in number, thereby increasing the size of the flow network and hence the running time. Similar arguments can be made for the four patterns (Figures 16(b) and 16(d)).

For patterns on the larger graphs (Figure 16(d)), we use our heuristic Pattern-NDS method in place of our approximate one (§ IV). In fact, the approximate Pattern-NDS method takes more than 3 days to run on these datasets. To show the quality of the results returned by our heuristic method, Table XI compares the running times and the (approximate) densest subgraph containment probabilities (as in § VI-B) of the solutions returned by both methods on the smaller Karate Club dataset. Clearly, for all patterns, the heuristic method



(a) Edge and Clique-MPDS (b) Pattern-MPDS (c) Edge and Clique-NDS (d) Heuristic Pattern-NDS

Fig. 16: Running times of our proposed methods; MPDS for the smaller datasets and NDS for the larger ones; HS denotes HomoSapiens

TABLE XIII: Sampling methods' comparison; MPDS; Intel Lab

Method	θ	Running Time (seconds)	Memory (MB)
MC	160	2.233	**2.016**
LP	160	2.164	2.656
RSS	**120**	**2.111**	3.281

TABLE XIV: Sampling methods' comparison; NDS; Biomine

Method	θ	Running Time (seconds)	Memory (MB)
MC	640	2248	**781**
LP	640	2178	1029
RSS	**600**	**2027**	1516

returns solutions of comparable quality to those returned by the approximate one, while having a lower running time.

For our largest dataset Friendster, our approximate methods require higher running times. Thus, we adopt a similar heuristic as in § III-C to edge and clique densities (which returns, in each sampled world, all subgraphs denser than the innermost core), and run these heuristic methods on Friendster. As shown in Table XII, this heuristic method yields reasonably good results, while significantly reducing the running time.

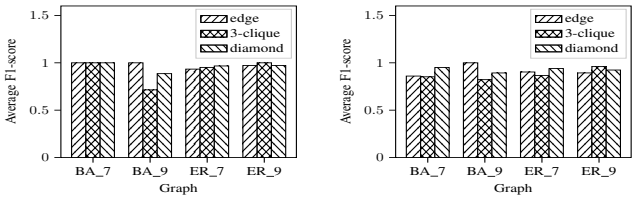
Varying sampling strategies. For sampling possible worlds, we compare our employed *Monte Carlo (MC)* with *Lazy Propagation (LP)* and *Recursive Stratified Sampling (RSS)* (§ III-A). Tables XIII and XIV show that all strategies result in *similar sample sizes* θ at convergence (§ VI-I) and have *comparable running times*, while MC consumes much less memory. The other datasets exhibit similar trends and are omitted due to space constraints. Our MPDS and NDS solutions require sampling *all* edges in the graph. When applying LP, the visit frequencies of *all* edges need to be stored and updated. This reduces the speedup while increasing the memory usage. RSS has a lower variance than MC; the difference depends on the variances of the estimates in various strata [55]. Unlike reliability queries, our solutions can hardly benefit from the BFS-based edge selection strategy (starting from high-degree nodes), since we need to consider the states of *all* edges. Ergo, the estimation variance is not reduced much, leading to a similar (but slightly smaller) sample size θ [98]. Thus, there is limited speedup on top of the memory overhead due to recursion. Hence, we adopt MC as the default strategy.

H. Comparison with the Exact Algorithms

We compare the effectiveness and efficiency of our proposed approximate top-k MPDS algorithms with those of the corresponding exact algorithms. Since the computation is $\#\mathbf{P}$-hard (II-D), the exact top-k MPDS methods need to compute the densest subgraphs in all 2^m possible worlds, which is infeasible for the datasets in Table II. Thus, we generate four small synthetic graphs according to the Erdös-Rényi (ER) [99] and Barabási-Albert (BA) [100] models, and assign edge probabilities uniformly at random. We denote by ER_n (resp. BA_n) the ER (resp. BA) graph with n nodes. The number m of edges of the generated datasets, along with the running times of the exact and our proposed algorithms for edge, 3-clique, and diamond MPDS, are shown in Table XV. Clearly, for each notion of density on each dataset, the exact method takes some orders of magnitude longer time than our method.

TABLE XV: Running times (seconds) of the exact and our approx. MPDS methods on synthetic graphs with number of edges m

Graph	m	Edge		3-Clique		Diamond	
		Exact	Ours	Exact	Ours	Exact	Ours
BA_7	13	0.172	**0.02**	0.225	**0.025**	0.349	**0.025**
BA_9	21	58.08	**0.04**	77.264	**0.042**	93.095	**0.045**
ER_7	20	71.39	**0.033**	78.919	**0.036**	140.361	**0.04**
ER_9	30	97413	**0.048**	123253	**0.054**	273557	**0.064**



(a) $k = 5$ (b) $k = 10$

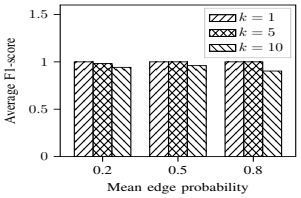
Fig. 17: F1-score (averaged across all ranks from 1 to k) of the top-k node sets returned by our MPDS methods w.r.t. the exact ones

We next show the effectiveness. For $k = 1$, in all cases, our method returns the same result as the exact one. For $k \in \{5, 10\}$, we compute, for each rank, the F1-score of the node set returned by our method with respect to the exact one as the ground truth. Figure 17 shows these scores averaged across all ranks from 1 to k. Clearly, the scores are reasonably high in all cases, which implies a high accuracy for each method.

Varying edge probabilities. To show the impact of the edge probabilities on the effectiveness of our method, we generate 3 synthetic graphs by assigning normally distributed edge probabilities with means 0.2, 0.5 and 0.8 to ER_7. For each graph, we run both the exact and our approximate MPDS methods. Figure 18 above shows that our method returns reasonably good results for all edge probability distributions. The running time of our method is longer for larger values of the mean, since the edge probabilities and hence the possible worlds sampled in each round are larger.

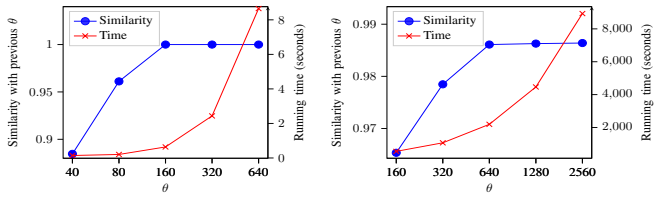
Mean	Running Time (sec.)
0.2	0.0899594
0.5	0.102649
0.8	0.121513

(a) Running Time



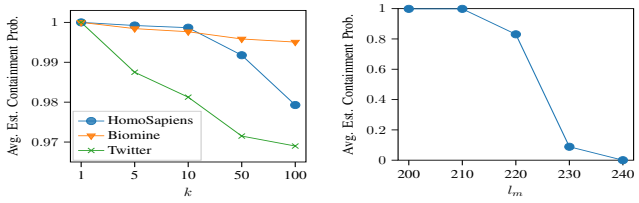
(b) Average F1-score

Fig. 18: Running time of our MPDS method and F1-score (averaged across all ranks from 1 to k) of the top-k node sets returned by our MPDS method w.r.t. the exact ones for synthetic edge probabilities



(a) MPDS; Intel Lab (b) NDS; Biomine

Fig. 19: Variation, w.r.t. θ, of the running time and the similarity of the returned node sets to those for the previous value of θ



(a) Varying k (b) Varying l_m; Homo Sapiens

Fig. 20: Variation of the average estimated densest subgraph containment probability with k and l_m for NDS queries

I. Parameter Sensitivity

Varying θ. We study the effects of the variation of θ in Figure 19. As in § VI-G, although we show the results for only two datasets, the other datasets exhibit similar trends. For MPDS on the Intel Lab dataset (Figure 19(a)), increasing θ steadily increases the similarity of the returned node sets to those for the previous value of θ till a certain point ($\theta = 160$), after which it converges, while the running time keeps increasing. Similar effects can be observed at $\theta = 640$ for NDS on the Biomine dataset (Figure 19(b)). We choose such values of θ as default for the respective datasets in our experiments.

Varying k for top-k NDSs. Figure 20(a) shows that increasing the value of k results in the reduction of the average estimated densest subgraph containment probability. This shows that increasing the value of k too much results in the returned node sets being of lower quality.

Varying l_m. Figure 20(b) shows the variation of the average estimated densest subgraph containment probability with l_m. Initially, till a certain value of l_m, the probability remains constant; since the returned node sets should be closed, our algorithm avoids reporting too small node sets even for smaller values of l_m. After that, the probability keeps decreasing with l_m till a certain value beyond which it remains 0 as there is no larger closed node set. This helps to choose a feasible upper bound on l_m. Note that Figure 20(b) shows the results for the Homo Sapiens dataset; the other datasets exhibit similar trends but over different ranges of values of l_m, and are omitted due to space constraints.

TABLE XVI: Table of notations in appendix

Notation	Description
Λ	Set of all $(h-1)$-cliques contained in h-cliques in G
$\mu_h(G)$	Number of h-cliques in G
ρ_h^*	Maximum h-clique density of any subgraph of G
$\mathcal{H}$	The critical flow network (Algorithm 6)
$c(\mathcal{S},\mathcal{T})$	Capacity of s-t cut $(\mathcal{S},\mathcal{T})$ in $\mathcal{H}$
f^*	A maximum flow in $\mathcal{H}$
$\mathcal{H}_{f^*}$	Residual graph of $\mathcal{H}$ under f^*
$\mathcal{H}^C$	Graph of strongly connected components of $\mathcal{H}_{f^*}$

VII. Conclusions

We studied the novel problem of finding the Most Probable Densest Subgraph (MPDS) in an uncertain graph, according to edge, clique and pattern densities. We proved that computing the densest subgraph probability for any given node set is #**P**-hard. We proposed a solution which returns the most frequent densest subgraphs from some sampled possible worlds, with theoretical accuracy guarantees. As building blocks, we designed novel algorithms to compute all clique- and pattern-densest subgraphs in a deterministic graph. We then extended our algorithm to compute the Nucleus Densest Subgraphs (NDS) via reduction to closed frequent itemset mining. Our experiments on large real-world graphs showed that our methods are efficient. We depicted that our methods are reasonably accurate compared to the exact ones, while being orders of magnitude faster, using some small synthetic graphs. Moreover, we showed that our MPDS is significantly different from existing notions of dense subgraphs in uncertain graphs. Our case studies showcased the usefulness of the MPDS in differentiating autistic brains from healthy ones and in detecting useful communities in social networks.

Appendix A
Proof of Correctness of Algorithm 2

We prove that Algorithm 2 correctly computes all h-clique densest subgraphs of G as below:

(1) Lines 1-2 are justified by Lemma 2 and the fact that $\tilde{\rho}$ is a lower bound on ρ_h^*, the optimal h-clique density of a subgraph in G [5].

(2) Line 6 of Algorithm 2 computes a maximum flow f^* in $\mathcal{H}$. We show that the capacity of such flow can capture the number of h-cliques in G (Lemma 3 and Corollary 1). Thus, it facilitates the densest subgraph finding (Lemma 4).

(3) We show the properties of $\mathcal{H}_{f^*}$ regarding the source node s (Lemma 5) and the sink node t (Lemmas 6, 7), along with their SCCs (Lemma 8). These illustrate why we do not consider the SCCs of s and t in Line 7 of Algorithm 2.

(4) We prove that all densest subgraphs can be determined exactly once via enumerating all independent sets (Corollary 2), along with several definitions (Definitions 8-11) and auxiliary lemmas (Lemmas 9-10).

Lemma 2 ([5]). *Given a graph $G = (V, E)$ and an integer $h > 0$, the h-clique densest subgraph of G is contained in its $(\lceil \rho_h^* \rceil, h)$-core, where $\rho_h^* = \max_{S \subseteq V} \rho_h(S)$.*

Algorithm 6 Construct the flow network for clique density [20]

Input: Graph $G = (V, E)$, set of $(h-1)$-cliques Λ, parameter α
Output: Flow network $\mathcal{H} = (V_{\mathcal{H}}, E_{\mathcal{H}}, c)$
1: $V_{\mathcal{H}} \leftarrow V \cup \Lambda \cup \{s, t\}$, $E_{\mathcal{H}} \leftarrow \emptyset$
2: **for all** $v \in V$ **do**
3: $\quad E_{\mathcal{H}} \leftarrow E_{\mathcal{H}} \cup \{(s, v), (v, t), (v, s), (t, v)\}$
4: $\quad$ Set $c(s, v) = deg_G(v, h)$, $c(v, t) = h\alpha$, $c(v, s) = c(t, v) = 0$
5: **for all** $\lambda \in \Lambda$ **do**
6: $\quad$ **for all** $v \in \lambda$ **do**
7: $\quad\quad E_{\mathcal{H}} \leftarrow E_{\mathcal{H}} \cup \{(\lambda, v), (v, \lambda)\}$
8: $\quad\quad$ Set $c(\lambda, v) = +\infty$, $c(v, \lambda) = 0$
9: $\quad$ **for all** $v \in V$ **do**
10: $\quad\quad$ **if** λ and v form an h-clique **then**
11: $\quad\quad\quad E_{\mathcal{H}} \leftarrow E_{\mathcal{H}} \cup \{(v, \lambda), (\lambda, v)\}$
12: $\quad\quad\quad$ Set $c(v, \lambda) = 1$, $c(\lambda, v) = 0$
13: **return** $(V_{\mathcal{H}}, E_{\mathcal{H}}, c)$

Line 5 of Algorithm 2 constructs a flow network by the method in [20]. For the sake of completeness, the pseudocode is shown in Algorithm 6. Specifically, $\mathcal{H}$ has a source node s, a sink node t, and one node for each $v \in V$ and each $\lambda \in \Lambda$. Each edge e in $\mathcal{H}$ has a non-negative capacity $c(e)$. The edges can be classified into four categories, as shown in Line 3, Algorithm 6. A flow in $\mathcal{H}$ is a function $f : E_{\mathcal{H}} \to \mathbb{R}$ that satisfies the following properties:

- **Capacity:** $f(e) \leq c(e)\ \forall e \in E_{\mathcal{H}}$
- **Antisymmetry:** $f(v, u) = -f(u, v)\ \forall (u, v) \in E_{\mathcal{H}}$
- **Conservation:** For each $u \in V_{\mathcal{H}} \setminus \{s, t\}$, $\sum_{(u,v)\in E_{\mathcal{H}}} f(u, v) = \sum_{(v,u)\in E_{\mathcal{H}}} f(v, u) = 0$

The value of a flow f is defined as the total flow on the edges leaving s, i.e., $\sum_{v\in V_{\mathcal{H}} : (s,v)\in E_{\mathcal{H}}} f(s, v)$.

Line 6 of Algorithm 2 computes a maximum flow f^* in $\mathcal{H}$. We show in Lemma 4 that this facilitates the h-clique densest subgraph detection. To this end, we first compute the value of a maximum flow (and hence the capacity of a minimum s-t cut) in $\mathcal{H}$. We prove this in Corollary 1 using Lemma 3 as a building block. Lemma 3 computes, for a given node set $V_1 \subseteq V$, the minimum capacity of an s-t cut in $\mathcal{H}$ with all nodes in V_1 (resp. $V \setminus V_1$) on the same side as s (resp. t).

Lemma 3. *For a set $V_1 \subseteq V$, the minimum value of an s-t cut $(\mathcal{S}, V_{\mathcal{H}} \setminus \mathcal{S})$ of $\mathcal{H}$ having $\mathcal{S} \cap V = V_1$ is*

$$\min_{\mathcal{S}:s\in\mathcal{S},t\notin\mathcal{S},\mathcal{S}\cap V=V_1} c\left(\mathcal{S}, V_{\mathcal{H}} \setminus \mathcal{S}\right) = h\left[\mu_h(G) + [\rho_h^* - \rho_h(V_1)]\,|V_1|\right]$$

Moreover, one value of $\mathcal{S}$ achieving the minimum above has $\Lambda_1 = \mathcal{S} \cap \Lambda$ as the set of all $(h-1)$-cliques in Λ which consist only of nodes in V_1.

Proof. Similar to that of Theorem 2 in [20]. □

Corollary 1. *The capacity of a minimum s-t cut $(\mathcal{S}^*, V_{\mathcal{H}} \setminus \mathcal{S}^*)$ of $\mathcal{H}$ is $h\mu_h(G)$.*

Proof. From Lemma 3, and noticing that $\rho_h(V_1) \leq \rho_h^*$ for any $V_1 \subseteq V$, the capacity of a minimum s-t cut $(\mathcal{S}^*, V_{\mathcal{H}} \setminus \mathcal{S}^*)$ of $\mathcal{H}$ is

$$\begin{aligned} c(\mathcal{S}^*, V_{\mathcal{H}} \setminus \mathcal{S}^*) &= \min_{V_1\subseteq V} \min_{\mathcal{S}:s\in\mathcal{S},t\notin\mathcal{S},\mathcal{S}\cap V=V_1} c\left(\mathcal{S}, V_{\mathcal{H}} \setminus \mathcal{S}\right) \\ &= h\left[\mu_h(G) + \min_{V_1\subseteq V} [\rho_h^* - \rho_h(V_1)]\,|V_1|\right] \\ &= h\mu_h(G) \end{aligned}$$

□

We now show the relationship between a maximum flow in $\mathcal{H}$ (Line 6 of Algorithm 2) and the densest subgraphs in G.

Lemma 4. *Let V_1 be a non-empty subset of V. Denote by Λ_1 the set of all $(h-1)$-cliques in Λ consisting only of nodes in V_1, and let $\mathcal{S} = \{s\} \cup V_1 \cup \Lambda_1$. The following statements are equivalent.*

1) *V_1 induces a densest subgraph in G.*
2) *$(\mathcal{S}, V_{\mathcal{H}} \setminus \mathcal{S})$ is a minimum s-t cut in $\mathcal{H}$.*
3) *There is no edge from $\mathcal{S}$ to $V_{\mathcal{H}} \setminus \mathcal{S}$ in $\mathcal{H}_{f^*}$.*

Proof. Similar to that of Lemma 3 in [46]. □

To enumerate the densest subgraphs in G following Lemma 4, we decompose the directed graph $\mathcal{H}_{f^*}$ into strongly connected components (SCCs). By contracting each SCC of $\mathcal{H}_{f^*}$ into a super-node, we obtain $\mathcal{H}^C$, the SCC graph of $\mathcal{H}_{f^*}$, which is a directed acyclic graph. However, as mentioned in Line 7 of Algorithm 2, we do not consider the SCCs of s and t. In the following, we justify this based on some properties of $\mathcal{H}_{f^*}$ regarding the source node s (Lemma 5) and the sink node t (Lemmas 6 and 7), along with their SCCs (Lemma 8).

Lemma 5. *In the residual graph $\mathcal{H}_{f^*}$, there is no edge from the source node s to any $u \in V_{\mathcal{H}} \setminus \{s\}$, and there is an edge from every $v \in V$ to s.*

Proof. Similar to that of Lemma 4 in [46]. □

Lemma 6. *In the residual graph $\mathcal{H}_{f^*}$, there is a path from the sink node t to every $v \in V$.*

Proof. Similar to that of Lemma 5 in [46]. □

Lemma 7. *In the residual graph $\mathcal{H}_{f^*}$, there is a path from the sink node t to every $\lambda \in \Lambda$.*

Proof. Consider any arbitrary $\lambda \in \Lambda$. Suppose, by way of contradiction, that there is no path from t to λ in $\mathcal{H}_{f^*}$. In that case, there is no edge in $\mathcal{H}_{f^*}$ entering λ, i.e., the corresponding edge in $\mathcal{H}$ is saturated, as such edges can only be from nodes $v \in V$, and from Lemma 6, there is a path from t to every node $v \in V$ in $\mathcal{H}_{f^*}$. Now, for every $v \in \lambda$, $c(v, \lambda) = 0$, implying that $f^*(v, \lambda) = 0$. Also, for every $v \in V$ that forms an h-clique with λ, $c(v, \lambda) = 1$, implying that $f^*(v, \lambda) = 1$. (Note that there must exist such $v \in V$, since Λ contains only those $(h-1)$-cliques that are contained in h-cliques.) This means the total flow entering λ is positive, which violates the conservation property of a flow, leading to a contradiction. □

Lemma 8. *Let $scc(s)$ and $scc(t)$ denote the SCCs of $\mathcal{H}_{f^*}$ containing s and t, respectively. The SCC graph $\mathcal{H}^C$ of $\mathcal{H}_{f^*}$ has the following properties.*

- *$scc(s) = \{s\}$*
- *$scc(s)$ has no outgoing edge*
- *$scc(t)$ has no incoming edge*

Proof. The first two properties can be proved directly from Lemma 5. We prove the third by contradiction. Suppose $scc(t)$

has an incoming edge in $\mathcal{H}^C$ from another SCC C. Consider any node $u \in C$. Then $u \neq t$ and there is a path from u to t in $\mathcal{H}_{f^*}$. Clearly, $u \neq s$, which means $u \in V \cup \Lambda$. From Lemmas 6 and 7, there is a path from t to u in $\mathcal{H}_{f^*}$. In that case, u and t are in the same SCC, which is a contradiction. □

A densest subgraph of G cannot have any node in the SCC of s (Lemma 8) or t (Lemma 4). Moreover, removing the SCCs of s and t from $\mathcal{H}^C$ does not affect the connections between the other SCCs (Lemma 8), which can contain a densest subgraph. Therefore, we can focus entirely on the other SCCs, as in line 7 of Algorithm 2. We denote them as non-trivial components (Definition 8).

Definition 8 (Non-trivial Component)**.** *A non-trivial component is a strongly connected component of $\mathcal{H}_{f^*}$ which does not include the source node s or the target node t.*

To enumerate all densest subgraphs of G, we define the descendants and ancestors of non-trivial components (Definition 9) and introduce the independence among such components (Definition 10).

Definition 9 (Descendants and Ancestors)**.** *For a non-trivial component C, the set of descendants (resp. ancestors) of C, denoted by $des(C)$ (resp. $anc(C)$), is defined as the set of those non-trivial components C' such that there exists a directed path from C to C' (resp. from C' to C) in $\mathcal{H}^C$. For a set $\mathcal{C}$ of non-trivial components, the sets of descendants and ancestors of $\mathcal{C}$, denoted by $des(\mathcal{C})$ and $anc(\mathcal{C})$, respectively, are defined as $des(\mathcal{C}) = \cup_{C \in \mathcal{C}} des(C)$ and $anc(\mathcal{C}) = \cup_{C \in \mathcal{C}} anc(C)$.*

Definition 10 (Independent Component Set)**.** *An independent component set is a set $\mathcal{C}$ of non-trivial components such that:*

- *For all $C \in \mathcal{C}$, $C \cap V \neq \emptyset$.*
- *For all $C_1, C_2 \in \mathcal{C}$, $C_1 \notin des(C_2)$ and $C_2 \notin des(C_1)$.*

It turns out that enumerating all independent component sets is equivalent to enumerating all densest subgraphs in G (Corollary 2). To prove this, we first define the concept of d-closed component sets (Definition 11).

Definition 11 (d-Closed Component Set)**.** *A d-closed component set is a set $\mathcal{C}$ of non-trivial components such that:*

- *For all $C \in \mathcal{C}$ where $C \cap V = \emptyset$, there is at least one incoming edge in $\mathcal{H}^C$ to C from a component $C' \in \mathcal{C}$.*
- *$des(\mathcal{C}) \subseteq \mathcal{C}$.*

We now show the existence of a bijection between independent sets and d-closed component sets (Lemma 9), as well as one between d-closed component sets and densest subgraphs in G (Lemma 10), thereby proving Corollary 2.

Lemma 9. *For an independent component set $\mathcal{C}$, define $g(\mathcal{C}) = \mathcal{C} \cup des(\mathcal{C})$. Then g is a bijection between the set of independent component sets and the set of d-closed component sets of $\mathcal{H}^C$.*

Proof. Notice that the function g is well-defined, since $g(\mathcal{C})$ is a d-closed component set uniquely determined by the independent component set $\mathcal{C}$. It suffices to show that, for each d-closed component set $\mathcal{C}_1$, there is a unique independent component set $\mathcal{C}$ such that $\mathcal{C}_1 = g(\mathcal{C}) = \mathcal{C} \cup des(\mathcal{C})$.

Let $\mathcal{C}_2$ be the set of components in $\mathcal{C}_1$ with no incoming edges in $\mathcal{H}^C$ from any component in $\mathcal{C}_1$. We shall show that $\mathcal{C}_2$ is our required set by proving that **(1)** $g(\mathcal{C}_2) = \mathcal{C}_1$; **(2)** $\mathcal{C}_2$ is independent; and **(3)** $\mathcal{C}_2$ is the unique set satisfying the above.

To see why $g(\mathcal{C}_2) = \mathcal{C}_2 \cup des(\mathcal{C}_2) = \mathcal{C}_1$, note that $\mathcal{C}_2 \subseteq \mathcal{C}_1$ and $\mathcal{C}_2 \cap des(\mathcal{C}_2) = \emptyset$ by construction. Thus, it is enough to show that $des(\mathcal{C}_2) = \mathcal{C}_1 \setminus \mathcal{C}_2$. If $C \in des(\mathcal{C}_2)$, then $C \in \mathcal{C}_1 \setminus \mathcal{C}_2$ since $\mathcal{C}_1$ is d-closed; thus $des(\mathcal{C}_2) \subseteq \mathcal{C}_1 \setminus \mathcal{C}_2$. Also, if $C \in \mathcal{C}_1 \setminus \mathcal{C}_2$, then C has an incoming edge from a component $C' \in \mathcal{C}_1$. If $C' \in \mathcal{C}_2$, then clearly $C \in des(\mathcal{C}_2)$; otherwise C' has an incoming edge from a component $C'' \in \mathcal{C}_1$. Continuing in this way, since $\mathcal{H}^C$ is acyclic, C has a path from a component in $\mathcal{C}_2$, which means $C \in des(\mathcal{C}_2)$, and hence $\mathcal{C}_1 \setminus \mathcal{C}_2 \subseteq des(\mathcal{C}_2)$.

Now we prove that $\mathcal{C}_2$ is independent. First note that $\mathcal{C}_2$ only consists of non-trivial components C satisfying $C \cap V \neq \emptyset$. To see why, consider $C \in \mathcal{C}_1$ such that $C \cap V = \emptyset$. Since $\mathcal{C}_1$ is d-closed, there is at least one incoming edge in $\mathcal{H}^C$ to C from a component in $\mathcal{C}_1$, which means $C \notin \mathcal{C}_2$ by construction. Now suppose, by way of contradiction, that there exist $C', C'' \in \mathcal{C}_2 \subseteq \mathcal{C}_1$ where $C' \in des(C_2)$. Then there is a directed path from C'' to C' in $\mathcal{H}^C$. Let C''' be the immediate predecessor of C' on that path. Since $\mathcal{C}_1$ is d-closed, $C''' \in \mathcal{C}_1$. In that case, the edge from C''' to C' means that, by construction, $C' \notin \mathcal{C}_2$, which is a contradiction.

It remains to be proved that $\mathcal{C}_2$ is the unique set satisfying the above properties. Let $\mathcal{C}_3$ be another independent component set such that $g(\mathcal{C}_3) = \mathcal{C}_1$. We shall show that $\mathcal{C}_2 = \mathcal{C}_3$.

If $C \in \mathcal{C}_2 \subseteq \mathcal{C}_1$, suppose, by way of contradiction, that $C \notin \mathcal{C}_3$. Since $\mathcal{C}_1 = \mathcal{C}_3 \cup des(\mathcal{C}_3)$, $C \in des(\mathcal{C}_3)$. By definition, there is a path in $\mathcal{H}^C$ to C from a component $C' \in \mathcal{C}_3 \subseteq \mathcal{C}_1$. Let C'' be the immediate predecessor of C on that path. Since $\mathcal{C}_3$ is independent, $C'' \notin \mathcal{C}_3$. Since $\mathcal{C}_1$ is d-closed, $C'' \in \mathcal{C}_1$. Thus, there exists an edge in $\mathcal{H}^C$ to $C \in \mathcal{C}_2$ from $C'' \in \mathcal{C}_1$, which contradicts the construction of $\mathcal{C}_2$. Thus $C \in \mathcal{C}_3$, and hence $\mathcal{C}_2 \subseteq \mathcal{C}_3$.

If $C \notin \mathcal{C}_2$, suppose, by way of contradiction, that $C \in \mathcal{C}_3 \subseteq \mathcal{C}_1$. By definition, C has an incoming edge in $\mathcal{H}^C$ from a component $C' \in \mathcal{C}_1$. Clearly, C' is an ancestor of C. Since $\mathcal{C}_1 = \mathcal{C}_3 \cup des(\mathcal{C}_3)$, $C' \in \mathcal{C}_3$, which contradicts the assumption that $\mathcal{C}_3$ is independent. Thus $C \notin \mathcal{C}_3$, and hence $\mathcal{C}_3 \subseteq \mathcal{C}_2$. □

Lemma 10. *For a d-closed component set $\mathcal{C}$, define $g(\mathcal{C}) = \bigcup_{C \in \mathcal{C}} C \cap V$. Then g is a bijection between the set of d-closed component sets of $\mathcal{H}^C$ and the set of densest subgraphs of G.*

Proof. For any node $u \in V_{\mathcal{H}}$, let $scc(u)$ denote the unique strongly connected component in $\mathcal{H}^C$ containing u.

We first prove that the function g is well defined. Clearly, $g(\mathcal{C})$ is uniquely determined by $\mathcal{C}$. Since $\mathcal{C}$ is d-closed, there is no edge in $\mathcal{H}^C$ from a non-trivial component in $\mathcal{C}$ to any non-trivial component not in $\mathcal{C}$. Also, from Lemma 8, $scc(t)$ has no incoming edge and $scc(s)$ has no outgoing edge. In other words, there is no edge in $\mathcal{H}_{f^*}$ from any node in $\{s\} \cup$

$\left(\bigcup_{C\in\mathcal{C}} C\right)$ to any other node. Thus, from Lemma 4, $g(\mathcal{C})$ induces a densest subgraph in G.

Next we show that, for any $V_1 \subseteq V$ inducing a densest subgraph in G, there is a unique d-closed component set $\mathcal{C}$ such that $V_1 = g(\mathcal{C}) = \bigcup_{C\in\mathcal{C}} C \cap V$. Denote by Λ_1 the set of all $(h-1)$-cliques in Λ consisting only of nodes in V_1. Define $\mathcal{C}_1 = \bigcup_{u\in V_1\cup\Lambda_1} scc(u)$, and $\mathcal{C}_2 \subseteq \mathcal{C}_1$ as the set $\mathcal{C}_1$ excluding those components $C \in \mathcal{C}_1$ with $C \cap V = \emptyset$ and no edge in $\mathcal{H}^C$ to C from any component in $\mathcal{C}_1$. We shall show that $\mathcal{C}_2$ is our required component set by proving that **(1)** $g(\mathcal{C}_2) = V_1$; **(2)** $\mathcal{C}_2$ is d-closed; and **(3)** $\mathcal{C}_2$ is the unique set satisfying the above properties.

We first prove that $g(\mathcal{C}_2) = V_1$. Notice that every $v \in V_1$ is contained in some component in $\mathcal{C}_1$. Also, no node $v \in V \setminus V_1$ is contained in any component in $\mathcal{C}_1$ (see Claim 1). Since $\mathcal{C}_2 \subseteq \mathcal{C}_1$ by construction, the same holds for $\mathcal{C}_2$ as well. Now it is easy to see that $\bigcup_{C\in\mathcal{C}_2} C \cap V = V_1$.

Second, we prove that $\mathcal{C}_2$ is d-closed. Note that by construction, for all $C \in \mathcal{C}_2$ where $C \cap V = \emptyset$, there is at least one incoming edge in $\mathcal{H}^C$ to C from a component $C' \in \mathcal{C}_2$. Now, for showing that $des(\mathcal{C}_2) \subseteq \mathcal{C}_2$, it suffices to show that there is no edge (and hence no path) in $\mathcal{H}^C$ from a component in $C \in \mathcal{C}_2$ to a non-trivial component $C' \notin \mathcal{C}_2$. All components in $\mathcal{C}_1$ (and hence $\mathcal{C}_2$, since $\mathcal{C}_2 \subseteq \mathcal{C}_1$) consist only of nodes in $V_1 \cup \Lambda_1$, and from Lemma 4, there is no edge from such nodes to those contained in components not in $\mathcal{C}_1$. Also, by construction, no component in $\mathcal{C}_1$ (and hence $\mathcal{C}_2$) has any outgoing edge to any component in $\mathcal{C}_1 \setminus \mathcal{C}_2$.

Finally, we prove that $\mathcal{C}_2$ is the unique set satisfying the above two properties. Let $\mathcal{C}_3$ be another set such that $g(\mathcal{C}_3) = V_1$ and $\mathcal{C}_3$ is d-closed; we shall show that $\mathcal{C}_2 = \mathcal{C}_3$. For $i \in \{2, 3\}$, define $\mathcal{C}_i^{(V)} = \{C \in \mathcal{C}_i : C \cap V \neq \emptyset\}$ and $\mathcal{C}_i^{(\Lambda)} = \mathcal{C}_i \setminus \mathcal{C}_i^{(V)} = \{C \in \mathcal{C}_i : C \cap V = \emptyset\}$. Clearly, it suffices to prove that $\mathcal{C}_2^{(V)} = \mathcal{C}_3^{(V)}$ and $\mathcal{C}_2^{(\Lambda)} = \mathcal{C}_3^{(\Lambda)}$. The first one is trivial, since $\bigcup_{C\in\mathcal{C}_2} C \cap V = \bigcup_{C\in\mathcal{C}_3} C \cap V = V_1$. For the second, we prove that $\mathcal{C}_2^{(\Lambda)} \subseteq \mathcal{C}_3^{(\Lambda)}$ and $\mathcal{C}_3^{(\Lambda)} \subseteq \mathcal{C}_2^{(\Lambda)}$.

If $C \in \mathcal{C}_2^{(\Lambda)}$, then C must have an incoming edge from a component $C' \in \mathcal{C}_2$ (since $\mathcal{C}_2$ is d-closed) with $C' \cap V \neq \emptyset$ (by the construction of the flow network $\mathcal{H}$). In that case, $C' \in \mathcal{C}_2^{(V)} = \mathcal{C}_3^{(V)} \subseteq \mathcal{C}_3$. Since $\mathcal{C}_3$ is d-closed, $C \in \mathcal{C}_3$; and as $C \notin \mathcal{C}_2^{(V)} = \mathcal{C}_3^{(V)}$, $C \in \mathcal{C}_3^{(\Lambda)}$. Hence $\mathcal{C}_2^{(\Lambda)} \subseteq \mathcal{C}_3^{(\Lambda)}$.

If $C \notin \mathcal{C}_2^{(\Lambda)}$, suppose, by way of contradiction, that $C \in \mathcal{C}_3^{(\Lambda)}$. By definition, $C \subseteq \mathcal{C}_3$ and $C \subseteq \Lambda$. Since $\mathcal{C}_3$ is d-closed, C must have an incoming edge from a component $C' \in \mathcal{C}_3$. Also, by the construction of the flow network $\mathcal{H}$, $C' \cap V \neq \emptyset$, which means $C' \in \mathcal{C}_3^{(V)} = \mathcal{C}_2^{(V)} \subseteq \mathcal{C}_2$. Since $\mathcal{C}_2$ is d-closed, $C \in \mathcal{C}_2$. As $C \cap V = \emptyset$, $C \in \mathcal{C}_2^{(\Lambda)}$, which is a contradiction. Thus $C \notin \mathcal{C}_3^{(\Lambda)}$, and hence $\mathcal{C}_3^{(\Lambda)} \subseteq \mathcal{C}_2^{(\Lambda)}$. □

Claim 1. *For $V_1 \subseteq V$ inducing a densest subgraph in G, denote by Λ_1 the set of all $(h-1)$-cliques in Λ consisting only of nodes in V_1. For any non-trivial component C, either $C \subseteq V_1 \cup \Lambda_1$ or $C \cap (V_1 \cup \Lambda_1) = \emptyset$.*

Proof. Suppose, by way of contradiction, that C has nodes $u_1 \in V_1 \cup \Lambda_1$ and $u_2 \in V_{\mathcal{H}} \setminus (V_1 \cup \Lambda_1 \cup \{s, t\})$. Since C is strongly connected, there must exist a path from u_1 to u_2 in $\mathcal{H}_{f^*}$. Then there is an edge from a node in $V_1 \cup \Lambda_1$ to one in $V_{\mathcal{H}} \setminus (V_1 \cup \Lambda_1 \cup \{s, t\})$, which is not possible (Lemma 4). □

Algorithm 7 Construct the flow network for pattern density [5]

Input: Graph $G = (V, E)$, pattern $\psi = (V_\psi, E_\psi)$, set of ψ-instances Λ, parameter α
Output: Flow network $\mathcal{H} = (V_{\mathcal{H}}, E_{\mathcal{H}}, c)$
1: $V_{\mathcal{H}} \leftarrow V \cup \{s, t\}$, $E_{\mathcal{H}} \leftarrow \emptyset$
2: **for all** $v \in V$ **do**
3: $E_{\mathcal{H}} \leftarrow E_{\mathcal{H}} \cup \{(s, v), (v, t), (v, s), (t, v)\}$
4: Set $c(s, v) = deg_G(v, \psi)$, $c(v, t) = |V_\psi|\alpha$, $c(v, s) = c(t, v) = 0$
5: $\Lambda' \leftarrow$ Set of groups g of ψ-instances in Λ with the same node set
6: **for all** $g' \in \Lambda$ **do**
7: $\lambda' \leftarrow$ Node set of instances in g
8: $V_{\mathcal{H}} \leftarrow V_{\mathcal{H}} \cup \{\lambda'\}$
9: **for all** $v' \in \lambda'$ **do**
10: $E_{\mathcal{H}} \leftarrow E_{\mathcal{H}} \cup \{(\lambda', v'), (v', \lambda')\}$
11: Set $c(\lambda', v') = |g|\,(|V_\psi| - 1)$, $c(v', \lambda') = |g|$
12: **return** $(V_{\mathcal{H}}, E_{\mathcal{H}}, c)$

Corollary 2. *For an independent component set $\mathcal{C}$, define $g(\mathcal{C}) = \bigcup_{C\in\mathcal{C}\cup des(\mathcal{C})} C \cap V$. Then g is a bijection between the set of independent component sets and the set of densest subgraphs of G.*

Proof. Let g_1 and g_2 be the bijections in Lemmas 9 and 10 respectively. Clearly $g_2(g_1(\mathcal{C})) = g(\mathcal{C})$ is a bijection. □

Following Corollary 2, we enumerate all densest subgraphs (Line 8 of Algorithm 2) by exploring all independent component sets, which invokes Algorithm 3 by taking an empty set $\emptyset$, the set of all non-trivial components in $\mathcal{H}^C$, and V as inputs. Specifically, Algorithm 3 recursively selects an independent component set $\mathcal{C}_1$ while storing in $\mathcal{C}_2$, the components that are independent with all components in $\mathcal{C}_1$. When a component C is moved from $\mathcal{C}_2$ to $\mathcal{C}_1$, all descendants and ancestors of C along with C itself are removed from $\mathcal{C}_2$ and then a recursion is invoked (line 7); this guarantees that $\mathcal{C}_1$ is independent. Note that the removal of C itself (line 6) ensures that each independent component set, and hence each densest subgraph, is enumerated exactly once.

APPENDIX B
PROOF OF CORRECTNESS OF ALGORITHM 4

The correctness of Algorithm 4 can be proved in a way similar to that of Algorithm 2 (Appendix A). The main difference lies in the construction of the flow network $\mathcal{H}$, which is done using the method in [5]. For reference, its pseudocode is shown in Algorithm 7. Note that this algorithm does not construct a node for every ψ-instance; rather, it constructs one node for each group g of ψ-instances with the same set of nodes in V. Hereafter, we denote by Λ' the set of all such ψ-instance groups. Since this algorithm is different from its analogue in Appendix A, so is the derivation of the value of a maximum flow in the constructed network $\mathcal{H}$. This is shown in Lemma 11.

Lemma 11. *For a set $V_1 \subseteq V$, the minimum value of an s-t cut $(\mathcal{S}, V_{\mathcal{H}} \setminus \mathcal{S})$ of $\mathcal{H}$ having $\mathcal{S} \cap V = V_1$ is*

$$\min_{\mathcal{S}: s\in\mathcal{S}, t\notin\mathcal{S}, \mathcal{S}\cap V = V_1} c(\mathcal{S}, V_{\mathcal{H}} \setminus \mathcal{S}) = |V_\psi| \left[\mu_\psi(G) + \left[\rho_\psi^* - \rho_\psi(V_1)\right] |V_1|\right]$$

Moreover, one value of $\mathcal{S}$ achieving the minimum above has $\Lambda'_1 = \mathcal{S} \cap \Lambda'$ as the set of all ψ-instance groups in Λ' which consist only of nodes in V_1.

Proof. Similar to that of Lemma 13 in [5]. □

APPENDIX C
EXTENSION OF EXPECTED DENSEST SUBGRAPHS TO CLIQUE AND PATTERN DENSITIES

The notion of expected densest subgraphs in uncertain graphs, which has been studied for edge density [44], can be extended to clique and pattern densities. The following discussion is for pattern density, and hence also holds for the special case of clique density.

Definition 12 (Expected Pattern Density)**.** *Given an uncertain graph $\mathcal{G} = (V, E, p)$ and a pattern ψ, the expected pattern density (w.r.t. ψ) of $\mathcal{G}$ is the expectation of the pattern density of all possible worlds of $\mathcal{G}$. Formally,*

$$\overline{\rho}_\psi(\mathcal{G}) = \mathbb{E}\left[\rho_\psi(G)\right] = \sum_{G \sqsubseteq \mathcal{G}} \Pr(G) \times \rho_\psi(G) \tag{21}$$

As shown in Theorem 7, the expected pattern density of a subgraph w.r.t. any pattern can be expressed as the weighted pattern density of that subgraph, which is ratio of the sum of the weights of all pattern instances in the deterministic version of the subgraph to the number of nodes in the subgraph. Here the weight of a pattern instance is its existence probability, i.e., the product of its edge probabilities.

Theorem 7. *Given an uncertain graph $\mathcal{G} = (V, E, p)$ and a pattern ψ, the expected pattern density (w.r.t. ψ) of a subgraph $\mathcal{G}' = (V', E', p)$ can be computed as*

$$\overline{\rho}_\psi(\mathcal{G}') = \frac{1}{|V'|} \sum_{\omega \in \Omega} \prod_{e \in E_\omega} p(e) \tag{22}$$

where Ω is the set of all instances $\omega = (V_\omega, E_\omega)$ of ψ in the deterministic version of $\mathcal{G}'$.

Proof. Consider a possible world $G' \sqsubseteq \mathcal{G}'$. For each $\omega \in \Omega$, let X_ω be a binary random variable denoting whether ω exists in G'. Then $\mathbb{E}[X_\omega] = \Pr(X_\omega = 1) = \prod_{e \in E_\omega} p(e)$ and the number of ψ-instances in G' is $\sum_{\omega \in \Omega} X_\omega$. Thus the pattern density (w.r.t. ψ) of G' is

$$\rho_\psi(G') = \frac{1}{|V'|} \sum_{\omega \in \Omega} X_\omega$$

From Definition 12 and the linearity of expectation, we have

$$\overline{\rho}_\psi(\mathcal{G}') = \mathbb{E}\left[\rho_\psi(G')\right] = \frac{1}{|V'|} \sum_{\omega \in \Omega} \mathbb{E}[X_\omega] = \frac{1}{|V'|} \sum_{\omega \in \Omega} \prod_{e \in E_\omega} p(e)$$

□

Since the expected pattern density is equivalent to the weighted pattern density of a particular weighted deterministic graph, the expected pattern densest subgraph in an uncertain graph can be computed using existing methods like [57].

REFERENCES

[1] A. V. Goldberg, *Finding a Maximum Density Subgraph*. University of California Berkeley, 1984.

[2] M. Charikar, "Greedy approximation algorithms for finding dense components in a graph," in *International Workshop on Approximation Algorithms for Combinatorial Optimization*. Berlin, Heidelberg: Springer, 2000, pp. 84–95.

[3] C. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarli, "Denser than the densest subgraph: Extracting optimal quasi-cliques with quality guarantees," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013, p. 104–112.

[4] A. Gionis and C. E. Tsourakakis, "Dense subgraph discovery," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, p. 2313–2314.

[5] Y. Fang, K. Yu, R. Cheng, L. V. Lakshmanan, and X. Lin, "Efficient algorithms for densest subgraph discovery," *Proceedings of the VLDB Endowment*, vol. 12, no. 11, p. 1719–1732, 2019.

[6] Y. Dourisboure, F. Geraci, and M. Pellegrini, "Extraction and classification of dense implicit communities in the web graph," *ACM Transactions on the Web*, vol. 3, no. 2, pp. 1–36, 2009.

[7] K. Asatani, H. Yamano, T. Sakaki, and I. Sakata, "Dense and influential core promotion of daily viral information spread in political echo chambers," *Scientific reports*, vol. 11, no. 1, pp. 1–10, 2021.

[8] L. V. Lakshmanan, "On a quest for combating filter bubbles and misinformation," in *Proceedings of the 2022 ACM SIGMOD International Conference on Management of Data*, 2022, p. 2.

[9] R. Legenstein, W. Maass, C. H. Papadimitriou, and S. S. Vempala, "Long term memory and the densest k-subgraph problem," in *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 94. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, pp. 57:1–57:15.

[10] Q. Wu, X. Huang, A. J. Culbreth, J. A. Waltz, L. E. Hong, and S. Chen, "Extracting brain disease-related connectome subgraphs by adaptive dense subgraph discovery," *Biometrics*, 2021.

[11] X. Du, R. Jin, L. Ding, V. E. Lee, and J. H. Thornton, "Migration motif: A spatial-temporal pattern mining approach for financial markets," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, p. 1135–1144.

[12] G. Buehrer and K. Chellapilla, "A scalable pattern mining approach to web graph compression with communities," in *Proceedings of the 2008 International Conference on Web Search and Data Mining*, 2008, p. 95–106.

[13] Y. Zhang and S. Parthasarathy, "Extracting analyzing and visualizing triangle k-core motifs within networks," in *IEEE 28th international conference on data engineering*, 2012, pp. 1049–1060.

[14] F. Zhao and A. K. H. Tung, "Large scale cohesive subgraphs discovery for social network visual analysis," *Proceedings of the VLDB Endowment*, vol. 6, no. 2, p. 85–96, 2012.

[15] E. Cohen, E. Halperin, H. Kaplan, and U. Zwick, "Reachability and distance queries via 2-hop labels," in *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*. USA: Society for Industrial and Applied Mathematics, 2002, p. 937–946.

[16] R. Jin, Y. Xiang, N. Ruan, and D. Fuhry, "3-hop: A high-compression indexing scheme for reachability query," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, 2009, p. 813–826.

[17] A. Gionis, F. Junqueira, V. Leroy, M. Serafini, and I. Weber, "Piggybacking on social networks," *Proceedings of the VLDB Endowment*, vol. 6, no. 6, p. 409–420, 2013.

[18] A. Faragó and Z. R. Mojaveri, "In search of the densest subgraph," *Algorithms*, vol. 12, no. 8, p. 157, 2019.

[19] C. E. Tsourakakis, "The k-clique densest subgraph problem," in *Proceedings of the 24th International Conference on World Wide Web*. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2015, p. 1122–1132.

[20] M. Mitzenmacher, J. Pachocki, R. Peng, C. E. Tsourakakis, and S. C. Xu, "Scalable large near-clique detection in large-scale networks via sampling," in *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, p. 815–824.

[21] H. Yanagisawa and S. Hara, "Discounted average degree density metric and new algorithms for the densest subgraph problem," *Networks*, vol. 71, no. 1, pp. 3–15, 2018.

[22] C. C. Aggarwal, *Managing and Mining Uncertain Data*, ser. Advances in Database Systems. Boston, MA, USA: Springer, 2009, vol. 35.

[23] E. Adar and C. Ré, "Managing uncertainty in social networks," *IEEE Data Engineering Bulletin*, vol. 30, no. 2, pp. 15–22, 2007.

[24] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," in *Proceedings of the 12th International Conference on Information and Knowledge Management*, 2003, p. 556–559.

[25] P. Boldi, F. Bonchi, A. Gionis, and T. Tassa, "Injecting uncertainty in graphs for identity obfuscation," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, p. 1376–1387, 2012.

[26] A. Khan, Y. Ye, and L. Chen, *On Uncertain Graphs*, ser. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2018, vol. 10.

[27] V. Kassiano, A. Gounaris, A. N. Papadopoulos, and K. Tsichlas, "Mining uncertain graphs: An overview," in *International Workshop of Algorithmic Aspects of Cloud Computing*. Cham: Springer, 2016, pp. 87–116.

[28] A. Khan and L. Chen, "On uncertain graphs modeling and queries," *Proceedings of the VLDB Endowment*, vol. 8, no. 12, p. 2042–2043, 2015.

[29] P. Sevon, L. Eronen, P. Hintsanen, K. Kulovesi, and H. Toivonen, "Link discovery in graphs derived from biological databases," in *International Workshop on Data Integration in the Life Sciences*. Berlin, Heidelberg: Springer, 2006, pp. 35–49.

[30] C. Zhang, C. Ré, M. Cafarella, C. De Sa, A. Ratner, J. Shin, F. Wang, and S. Wu, "Deepdive: Declarative knowledge base construction," *Communications of the ACM*, vol. 60, no. 5, p. 93–102, 2017.

[31] Z. Zou, H. Gao, and J. Li, "Discovering frequent subgraphs over uncertain graph databases under probabilistic semantics," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010, p. 633–642.

[32] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003, p. 137–146.

[33] M. Hua and J. Pei, "Probabilistic path queries in road networks: Traffic uncertainty aware path selection," in *Proceedings of the 13th International Conference on Extending Database Technology*, 2010, p. 347–358.

[34] V. K. Yalavarthi, X. Ke, and A. Khan, "Select your questions wisely: For entity resolution with crowd errors," in *Proceedings of the 2017 ACM Conference on Information and Knowledge Management*, 2017, p. 317–326.

[35] M. Potamias, F. Bonchi, A. Gionis, and G. Kollios, "k-nearest neighbors in uncertain graphs," *Proceedings of the VLDB Endowment*, vol. 3, no. 1–2, p. 997–1008, 2010.

[36] A. Saha, R. Brokkelkamp, Y. Velaj, A. Khan, and F. Bonchi, "Shortest paths and centrality in uncertain networks," *Proceedings of the VLDB Endowment*, vol. 14, no. 7, p. 1188–1201, 2021.

[37] A. P. Mukherjee, P. Xu, and S. Tirthapura, "Enumeration of maximal cliques from an uncertain graph," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 3, pp. 543–555, 2016.

[38] Z. Zou, J. Li, H. Gao, and S. Zhang, "Finding top-k maximal cliques in an uncertain graph," in *IEEE 26th International Conference on Data Engineering*, 2010, pp. 649–652.

[39] R.-H. Li, Q. Dai, G. Wang, Z. Ming, L. Qin, and J. X. Yu, "Improved algorithms for maximal clique search in uncertain networks," in *IEEE 35th International Conference on Data Engineering*, 2019, pp. 1178–1189.

[40] F. Bonchi, F. Gullo, A. Kaltenbrunner, and Y. Volkovich, "Core decomposition of uncertain graphs," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, p. 1316–1325.

[41] X. Huang, W. Lu, and L. V. Lakshmanan, "Truss decomposition of probabilistic graphs: Semantics and algorithms," in *Proceedings of the 2016 ACM SIGMOD International Conference on Management of Data*, 2016, p. 77–90.

[42] K. Han, F. Gui, X. Xiao, J. Tang, Y. He, Z. Cao, and H. Huang, "Efficient and effective algorithms for clustering uncertain graphs," *Proceedings of the VLDB Endowment*, vol. 12, no. 6, p. 667–680, 2019.

[43] C. Ma, R. Cheng, L. V. Lakshmanan, T. Grubenmann, Y. Fang, and X. Li, "Linc: A motif counting algorithm for uncertain graphs," *Proceedings of the VLDB Endowment*, vol. 13, no. 2, p. 155–168, 2019.

[44] Z. Zou, "Polynomial-time algorithm for finding densest subgraphs in uncertain graphs," in *Proceedings of the 11th Workshop on Mining and Learning with Graphs*, 2013.

[45] C. E. Tsourakakis, T. Chen, N. Kakimura, and J. Pachocki, "Novel dense subgraph discovery primitives: Risk aversion and exclusion queries," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2020, pp. 378–394.

[46] L. Chang and M. Qiao, "Deconstruct densest subgraphs," in *Proceedings of the 29th International Conference on World Wide Web*, 2020, p. 2747–2753.

[47] J. Wang, J. Han, Y. Lu, and P. Tzvetkov, "Tfp: an efficient algorithm for mining top-k frequent closed itemsets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 5, pp. 652–663, 2005.

[48] L. G. Valiant, "The complexity of enumeration and reliability problems," *SIAM Journal on Computing*, vol. 8, no. 3, pp. 410–421, 1979.

[49] M. O. Ball, "Computational complexity of network reliability analysis: An overview," *IEEE Transactions on Reliability*, vol. 35, no. 3, pp. 230–239, 1986.

[50] A. Khan, F. Bonchi, F. Gullo, and A. Nufer, "Conditional reliability in uncertain graphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 11, pp. 2078–2092, 2018.

[51] J. Chen and Y. Saad, "Dense subgraph extraction with application to community detection," *TKDE*, vol. 24, no. 7, pp. 1216–1230, 2012.

[52] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2017, pp. 555–564.

[53] V. Batagelj and M. Zaversnik, "An o(m) algorithm for cores decomposition of networks," *CoRR*, vol. cs.DS/0310049, 2003.

[54] Y. Li, J. Fan, D. Zhang, and K.-L. Tan, "Discovering your selling points: Personalized social influential tags exploration," in *Proceedings of the 2017 ACM SIGMOD International Conference on Management of Data*, 2017, p. 619–634.

[55] R.-H. Li, J. X. Yu, R. Mao, and T. Jin, "Recursive stratified sampling: A new framework for query evaluation on uncertain graphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 2, pp. 468–482, 2016.

[56] M. Danisch, O. D. Balalau, and M. Sozio, "Listing k-cliques in sparse real-world graphs," in *Proceedings of the 2018 World Wide Web Conference*. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2018, p. 589–598.

[57] B. Sun, M. Danisch, T.-H. H. Chan, and M. Sozio, "Kclist++: A simple algorithm for finding k-clique densest subgraphs in large graphs," *Proceedings of the VLDB Endowment*, vol. 13, no. 10, p. 1628–1640, 2020.

[58] M. Qiao, H. Zhang, and H. Cheng, "Subgraph matching: On compression and computation," *Proceedings of the VLDB Endowment*, vol. 11, no. 2, p. 176–188, 2017.

[59] O. D. Balalau, F. Bonchi, T.-H. H. Chan, F. Gullo, and M. Sozio, "Finding subgraphs with maximum total density and limited overlap," in *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, 2015, p. 379–388.

[60] B. Bahmani, R. Kumar, and S. Vassilvitskii, "Densest subgraph in streaming and mapreduce," *Proc. VLDB Endow.*, vol. 5, no. 5, p. 454–465, jan 2012.

[61] S. Sawlani and J. Wang, "Near-optimal fully dynamic densest subgraph," in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, ser. STOC 2020. New York, NY, USA: Association for Computing Machinery, 2020, p. 181–193.

[62] F. Bonchi, D. García-Soriano, A. Miyauchi, and C. E. Tsourakakis, "Finding densest k-connected subgraphs," *Discrete Appl. Math.*, vol. 305, no. C, p. 34–47, dec 2022.

[63] R. Andersen and K. Chellapilla, "Finding dense subgraphs with size bounds," in *Algorithms and Models for the Web-Graph*, K. Avrachenkov, D. Donato, and N. Litvak, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 25–37.

[64] E. Galbrun, A. Gionis, and N. Tatti, "Top-k overlapping densest subgraphs," *Data Min. Knowl. Discov.*, vol. 30, no. 5, p. 1134–1165, sep 2016.

[65] M. A. U. Nasir, A. Gionis, G. D. F. Morales, and S. Girdzijauskas, "Fully dynamic algorithm for top-k densest subgraphs," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, ser. CIKM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1817–1826.

[66] L. Qin, R.-H. Li, L. Chang, and C. Zhang, "Locally densest subgraph discovery," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 965–974.

[67] A. Miyauchi and A. Takeda, "Robust densest subgraph discovery," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 1188–1193.

[68] M. Danisch, T.-H. H. Chan, and M. Sozio, "Large scale density-friendly graph decomposition via convex programming," in *Proceedings of the 26th International Conference on World Wide Web*, ser. WWW '17. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2017, p. 233–242.

[69] N. Tatti, "Density-friendly graph decomposition," *ACM Trans. Knowl. Discov. Data*, vol. 13, no. 5, sep 2019.

[70] C. Ma, Y. Fang, R. Cheng, L. V. Lakshmanan, W. Zhang, and X. Lin, "Efficient algorithms for densest subgraph discovery on large directed graphs," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1051–1066.

[71] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "Fraudar: Bounding graph fraud in the face of camouflage," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 895–904.

[72] E. Galimberti, F. Bonchi, F. Gullo, and T. Lanciano, "Core decomposition in multilayer networks: Theory, algorithms, and applications," *ACM Trans. Knowl. Discov. Data*, vol. 14, no. 1, jan 2020.

[73] Y. Fang, W. Luo, and C. Ma, "Densest subgraph discovery on large graphs: Applications, challenges, and techniques," *Proc. VLDB Endow.*, vol. 15, no. 12, pp. 3766–3769, 2022.

[74] F. Esfahani, J. Wu, V. Srinivasan, A. Thomo, and K. Wu, "Fast truss decomposition in large-scale probabilistic graphs," in *Advances in Database Technology - 22nd International Conference on Extending Database Technology*. OpenProceedings.org, 2019, pp. 722–725.

[75] D. Seux, F. Malliaros, A. Papadopoulos, and M. Vazirgiannis, "Core decomposition of uncertain graphs using representative instances," in *6th International Conference on Complex Networks and Their Applications*, 2017.

[76] Q. Dai, R. Li, G. Wang, R. Mao, Z. Zhang, and Y. Yuan, "Core decomposition on uncertain graphs revisited," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[77] Z. Sun, X. Huang, J. Xu, and F. Bonchi, "Efficient probabilistic truss indexing on uncertain graphs," in *The Web Conference*. ACM / IW3C2, 2021, pp. 354–366.

[78] Z. Zou and R. Zhu, "Truss decomposition of uncertain graphs," *Knowledge and Information Systems*, vol. 50, no. 1, pp. 197–230, 2017.

[79] Q. Dai, R.-H. Li, M. Liao, H. Chen, and G. Wang, "Fast maximal clique enumeration on uncertain graphs: A pivot-based approach," in *Proceedings of the 2022 ACM SIGMOD International Conference on Management of Data*, 2022, p. 2034–2047.

[80] L. Liu, R. Jin, C. C. Aggarwal, and Y. Shen, "Reliable clustering on uncertain graphs," in *IEEE International Conference on Data Mining*, 2012, pp. 459–468.

[81] M. Ceccarello, C. Fantozzi, A. Pietracaprina, G. Pucci, and F. Vandin, "Clustering uncertain graphs," *Proceedings of the VLDB Endowment*, vol. 11, no. 4, pp. 472–484, 2017.

[82] R. Jin, L. Liu, and C. C. Aggarwal, "Discovering highly reliable subgraphs in uncertain graphs," in *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining*. ACM, 2011, pp. 992–1000.

[83] A. Saha, X. Ke, A. Khan, and C. Long, "Most probable densest subgraphs: Code and data," 2022. [Online]. Available: https://github.com/ArkaSaha/MPDS

[84] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of anthropological research*, vol. 33, no. 4, pp. 452–473, 1977.

[85] S. Madden, "Intel lab data," 2004. [Online]. Available: http://db.csail.mit.edu/labdata/labdata.html

[86] "Lastfm api," 2002. [Online]. Available: https://www.last.fm

[87] D. Szklarczyk, A. L. Gable, K. C. Nastou, D. Lyon, R. Kirsch, S. Pyysalo, N. T. Doncheva, M. Legeay, T. Fang, P. Bork *et al.*, "The string database in 2021: Customizable protein–protein networks, and functional characterization of user-uploaded gene/measurement sets," *Nucleic Acids Research*, vol. 49, no. D1, pp. D605–D612, 2021.

[88] L. Eronen and H. Toivonen, "Biomine: Predicting links between biological entities using network models of heterogeneous databases," *BMC Bioinformatics*, vol. 13, no. 1, 2012.

[89] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, 2014.

[90] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowledge and Information Systems*, vol. 42, pp. 181–213, 2015.

[91] X. Ke, A. Khan, M. Al Hasan, and R. Rezvansangsari, "Reliability maximization in uncertain graphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 2, pp. 894–913, 2022.

[92] J. J. Pfeiffer III and J. Neville, "Methods to determine node centrality and clustering in graphs with uncertain structure," in *Fifth International AAAI Conference on Weblogs and Social Media*, 2011.

[93] C. Craddock, Y. Benhajali, C. Chu, F. Chouinard, A. Evans, A. Jakab, B. S. Khundrakpam, J. D. Lewis, Q. Li, M. Milham, C. Yan, and P. Bellec, "The neuro bureau preprocessing initiative: Open sharing of preprocessed neuroimaging data and derivatives," *Frontiers in Neuroinformatics*, 2013.

[94] M. Xia, J. Wang, and Y. He, "Brainnet viewer: a network visualization tool for human brain connectomics," *PloS one*, vol. 8, no. 7, p. e68910, 2013.

[95] A. Di Martino, C. Kelly, R. Grzadzinski, X.-N. Zuo, M. Mennes, M. Mairena, C. Lord, F. Castellanos, and M. Milham, "Aberrant striatal functional connectivity in children with autism," *Biological Psychiatry*, vol. 69, no. 9, pp. 847–56, 12 2010.

[96] S. Noonan, F. Haist, and R.-A. Müller, "Aberrant functional connectivity in autism: Evidence from low-frequency bold signal fluctuations," *Brain Research*, vol. 1262, pp. 48–63, 02 2009.

[97] M. Postema, D. Van Rooij, E. Anagnostou, C. Arango, G. Auzias, M. Behrmann, G. Busatto, S. Calderoni, R. Calvo, E. Daly, C. Deruelle, A. Di Martino, I. Dinstein, F. Duran, S. Durston, C. Ecker, S. Ehrlich, D. Fair, J. Fedor, and C. Francks, "Altered structural brain asymmetry in autism spectrum disorder in a study of 54 datasets," *Nature Communications*, vol. 10, 12 2019.

[98] X. Ke, A. Khan, and L. L. H. Quan, "An in-depth comparison of s-t reliability algorithms over uncertain graphs," *Proceedings of the VLDB Endowment*, vol. 12, no. 8, p. 864–876, 2019.

[99] P. Erdös and A. Rényi, "On random graphs," *Publicationes Mathematicae Debrecen*, vol. 6, pp. 290–297, 1959.

[100] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.