Maximal Quasi-Cliques Mining in Uncertain Graphs

Lianpeng Qiao, Rong-Hua Li, Zhiwei Zhang, Ye Yuan, Guoren Wang, and Hongchao Qin

Abstract—Cohesive subgraph mining is a fundamental problem in the field of graph data analysis. Many existing cohesive graph mining algorithms are mainly tailored to deterministic graphs. Real-world graphs, however, are often not deterministic, but uncertain in nature. Applications of such uncertain graphs include protein-protein interactions networks with experimentally inferred links and sensor networks with uncertain connectivity links. In this article, we study the problem of mining cohesive subgraphs from an uncertain graph. Specifically, we introduce a new (α, γ)-quasi-clique model to model the cohesive subgraphs in an uncertain graph, and propose a basic enumeration algorithm to find all maximal (α, γ)-quasi-cliques. We also develop an advanced enumeration algorithm based on several novel pruning rules, including early termination and candidate set reduction. To further improve the efficiency, we propose several optimization techniques. Extensive experiments on five real-world datasets demonstrate that our solutions are almost three times faster than the baseline approach.

Index Terms—Maximal (α, γ)-quasi-clique, uncertain graphs, cohesive subgraphs, enumeration algorithm

1 Introduction

Real-world graphs, such as social networks, protein-protein interaction (PPI) networks, and communication networks, often contain cohesive subgraph structures. Mining cohesive subgraphs from a graph is an important problem in the field of network analysis which has attracted much attention[1], [2], [3], [4] [5], [6], [7], [8], [9]. Among all the cohesive subgraphs, clique is the densest one, which requires that each node must connect to all the other nodes [10], [11]. However, the constraint of clique is too restrictive combined with the fact that most real-world datasets are incomplete. Considering this, γ-quasi-cliques are proposed as it requires all the nodes in the subgraph are adjacent to at least $\lceil \gamma \cdot (n-1) \rceil$ other nodes, where $\gamma \in (0, 1]$ and n denotes the number of nodes in the graph.

Many real-world graphs, however, are uncertain in nature where each edge is associated with a probability as shown in Fig. 1. The uncertain graph has been widely used in many applications to represent the uncertain connectivity links between objects, such as PPI networks with experimentally inferred links, social networks with uncertain links, and sensor networks with uncertain connectivity links. Some cohesive subgraph mining problems have recently been studied on uncertain graphs including the core decomposition [12], the truss decomposition [13] and the maximal clique enumeration problems [14], [15], [16].

Challenges and Contributions. In this paper, we propose a maximal (α, γ)-quasi-clique model to represent a maximal γ-quasi-clique in an uncertain graph. Specially, for an uncertain graph G, we call a set of nodes $C(V, E)$ a maximal (α, γ)-quasi-clique if (1) C is a γ-quasi-clique and the probability of each node's degree in C being larger than or equal to $\lceil \gamma \cdot (|V|-1) \rceil$ is not less than α, and (2) C is a maximal node set satisfying (1). For an uncertain graph G, each edge is associated with a probability, so the degree of each node in G is also associated with a probability. There are two main reasons that affect the efficiency of the maximal quasi-clique mining problem on an uncertain graph. The first one is that some nodes are not likely to be included into some quasi-cliques, but we still need to check whether these nodes satisfy the constraints of quasi-cliques in an uncertain graph. The other one is that the time cost of probability calculation and updating of nodes is often very expensive. In fact, even simple problems can become complex in the context of the uncertain graph. For example, to determine whether there is a path of length k between two given nodes in a deterministic graph, we can solve the problem within a polynomial time. However, in an uncertain graph, the problem becomes a #P-complete problem. Sanei-Mehri *et al.* [17] proved that even the problem of maximality checking of a quasi-clique in a deterministic graph is NP-hard. Enumerating all the maximal (α, γ)-quasi-cliques from an uncertain graph is an harder problem which demonstrates that our problem is also a #P-complete problem.

To tackle these challenges, we propose several novel and efficient algorithms to find all the quasi-cliques. Specifically, with the degree measurement of nodes and the (α, γ)-quasi-clique definition, we first design a basic enumeration algorithm to find all the maximal (α, γ)-quasi-cliques. By analyzing the properties of (α, γ)-quasi-clique in an uncertain

- *Lianpeng Qiao is with the Department of Computer Science, Northeastern University, Shenyang, Liaoning 110004, China. E-mail: qiaolp@stumail.neu.edu.cn.*
- *Rong-Hua Li, Zhiwei Zhang, Ye Yuan, Guoren Wang, and Hongchao Qin are with the Department of Computer Science, Beijing Institute of Technology, Beijing 100081, China. E-mail: {lironghuascut, qhc.neu}@gmail.com, cszwzhang@comp.hkbu.edu.hk, yuan-ye@bit.edu.cn, wanggrbit@126.com.*

Manuscript received 17 June 2020; revised 21 May 2021; accepted 22 June 2021. Date of publication 29 June 2021; date of current version 16 January 2023. (Corresponding author: Guoren Wang.)
Recommended for acceptance by M. Piccardi.
Digital Object Identifier no. 10.1109/TBDATA.2021.3093355

2332-7790 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.
Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

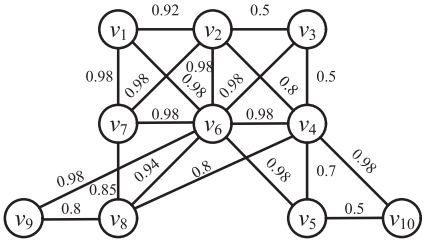


Fig. 1. An uncertain graph.

graph, we then propose an advanced algorithm with several carefully-designed pruning techniques, which can significantly reduce the search space and terminate the enumeration procedure early. Since all the proposed algorithmcs rely on the probability calculation for the nodes, we further reduce the cost by designing a dynamic programming algorithm to update the probabilities. Furthermore, a series of optimizations are also proposed to speed up the enumeration procedure. Our contributions are summarized as follows.

- We formalize the problem to mine maximal (α, γ)-quasi-cliques from an uncertain graph, where a probabilistic function is designed to measure the degree of the nodes in uncertain graphs. (Section 2)
- By analyzing the properties of (α, γ)-quasi-clique, we first propose a basic enumeration algorithm to find all maximal (α, γ)-quasi-cliques (Section 3.1). Then, to improve the efficiency, we develop an advanced enumeration algorithm with several effective pruning methods which include early termination and candidate set reduction (Sections 3.2, 3.3, and 3.4). For candidate set reduction, the nodes that do not belong to any (α, γ)-quasi-cliques will be removed without enumerating the graphs with them. For early termination, the enumeration will stop if the current node set cannot form an (α, γ)-quasi-clique. In addition, for the probability computation, we propose a new probability update method based on a dynamic programming framework (Section 4).
- We conduct extensive experiments on several real datasets, and the results show that the proposed techniques can find the (α, γ)-quasi-cliques effectively and efficiently (Section 6).

Organization. Section 2 introduces the model of (α, γ)-quasi-clique and formulates our problem. The pruning techniques and algorithms for mining maximal (α, γ)-quasi-cliques from an uncertain graph are proposed in Section 3. Probability calculation and update methods are proposed in Section 4. Section 5 introduces several pruning methods used by existing works to further speed up the algorithm 3. Note that since there are several methods that are not fully applicable to uncertain graph, we have made appropriate adjustments and still described here as contributions to existing work. Experimental studies are presented in Section 6. We review the related work in Section 7, and conclude this work in Section 8.

2 Preliminaries

In this section, we first introduce some useful notations, and then formulate our problem. Table 1 lists the main symbols used in this paper and their descriptions.

TABLE 1
Notations

Notations	Descriptions
G	$G = (V, E, p)$, an uncertain graph
$\tilde{G}$	the deterministic graph of G
$G(X)$	the induced graph of X in G
$\tilde{G}(X)$	the induced graph of X in $\tilde{G}$
$N_G(v)$	the set of neighbours of node v in G
$deg_G(v)$	the degree of node v in G
$\mathcal{G}_p$	the set of the possible worlds of G
$G_p^k(v)$	the set of the possible worlds of G where v's degree equals k
$Pr_G(deg(v) = k)$	the probability that the degree of v equals k in G
$P_G(v, k)$	the probability that the degree of v is greater than or equals k in G
$N_k^G(v)$	the set of nodes that are within a distance of k from node v in G
$indeg^X(v)$	the degree of node v in X
$indeg_{min}(X)$	the smallest $indeg^X$ in X
L_{min}	the lower bound of the number of nodes that can be added to X to form an (α, γ)-quasi-clique

Let $G(V, E, p)$ be an uncertain graph, where V denotes the set of nodes, E is the set of edges, and p is a function that assigns the probability of existence to each edge $e \in E$. For a node set $X \subseteq V$, $G(X) = (X, E_X, p)$ is an induced uncertain subgraph of X in G if $E_X = \{(u, v) | (u, v) \in E, u, v \in X\}$. The deterministic graph of G, denoted as $\tilde{G}$, is obtained by ignoring all probabilities of the edges in G. Let $N_{\tilde{G}}(u)$ be the set of neighbors of node u in $\tilde{G}$, and $deg_{\tilde{G}}(u) = |N_{\tilde{G}}(u)|$ is the degree of node u in $\tilde{G}$.

For an uncertain graph $G(V, E, p)$, we suppose that the existence of different edges in E is mutually independent. The possible world graphs of G are the deterministic graphs which contain all the nodes in V and the edges sampling from E based on the function p. Given an uncertain graph $G(V, E, p)$ and $m = |E|$. Possible world graphs of G can be regarded as $\mathcal{G}_p = \{\mathcal{G}_p^1, \mathcal{G}_p^2, ..., \mathcal{G}_p^{2^m}\}$. The probability of getting a possible world graphs $\mathcal{G}_p^i$ where $i \in [1, 2^m]$ from the uncertain graph G is:

$$P(\mathcal{G}_p^i) = \prod_{e \in E_{\mathcal{G}_p^i}} p(e) \prod_{e \in E \setminus E_{\mathcal{G}_p^i}} (1 - p(e)). \quad (1)$$

The γ-Quasi-Clique Model in a Deterministic Graph. Here we first introduce the problem of mining γ-quasi-cliques from a deterministic graph.

Definition 1 (γ-quasi-clique). *Given a deterministic graph $\tilde{G}(V, E)$ and a node set X, $\tilde{G}(X)$ is a γ-quasi-clique $(0 < \gamma < 1)$ if $\tilde{G}(X)$ is connected, and for every node $v \in X$, $deg_{\tilde{G}(X)}(v) \geq \lceil \gamma \cdot (|X| - 1) \rceil$.*

According to Definition 1, $\tilde{G}(X)$ is a maximal γ-quasi-clique of $\tilde{G}$ if $\tilde{G}(X)$ is a γ-quasi-clique, and there does not exist another node set Y such that $Y \supset X$ and $\tilde{G}(Y)$ is a γ-quasi-clique.

The γ-Quasi-Clique Model in an Uncertain Graph. Combining the definition of possible world graphs of an uncertain

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

graph G, finding a γ-quasi-clique in G is equivalent to identify the set of all possible world graphs in which the node set is a γ-quasi-clique. The probability of the γ-quasi-clique in an uncertain graph is the sum of the probabilities of these identified possible world graphs.

Given an uncertain graph $G(V, E, p)$, we can get at most $2^{|E|}$ possible worlds that are eligible for a γ-quasi-clique in G based on the possible-world semantics. In accordance with upon content, we can get that the complexity of the number of the qualified possible world graphs in the uncertain graph with m edges is $O(2^m)$. Clearly, given an uncertain graph $G(V, E, p)$, finding all eligible possible worlds is impractical as the number of the qualified possible world graphs in the uncertain graph is too large. The general turnaround adopted is to assign a score to each node based on the probability of the node to be part of a special subgraph structure, and then return the maximal node sets that satisfy the constraints, such as k-core [12]. Similar to [12], [18], we refer that $P_G(v, k)$ is the sum of probabilities of possible world graphs of uncertain graph G in which the degree of v is no smaller than k, and $Pr_G(deg(v) = k)$ to denote the probability of v's degree being equal to k in G. The detailed definition is as follows.

Definition 2 (k-probability). *Given an uncertain graph $G = (V, E, p)$ and a node $v \in V$, we call $P_G(v, k)$ as v's k-probability in G, which is the sum of the probabilities of possible world graphs of uncertain graph G in which the degree of v is no less than k.*

Definition 3 ((α, γ)-quasi-clique). *Given an uncertain graph $G(V, E, p)$, parameters α and γ. An induced subgraph $H(V_H, E_H, p)$ is an (α, γ)-quasi-clique in G if the probability, such that each node $v \in V_H$ has a degree no less than $\lceil \gamma \cdot (|V_H| - 1) \rceil$ inside H, is no less than α, i.e., $\forall v \in V_H$, $P_H(v, \lceil \gamma \cdot (|V_H| - 1) \rceil) \geq \alpha$.*

According to Definition 3, H is a maximal (α, γ)-quasi-clique if H is an (α, γ)-quasi-clique and there does not exist another node set Y such that $V_H \subset Y$ and $G(Y)$ is an (α, γ)-quasi-clique.

The state-of-the-art definition of the uncertain clique is (k, τ)-clique proposed by Li *et al*. [16]. There are two constraints in this definition. One of the constraints is the total number of the vertices in a (k, τ)-clique must be no smaller than k, the other is that the product of the probabilities of all edges in (k, τ)-clique must be no smaller than τ. Combining with the definitions of (k, τ)-clique and (α, γ)-quasi-clique, we can find that the uncertain clique is a special case of our uncertain quasi-clique. When γ is set to 1, the maximal (α, γ)-quasi-cliques we get in the uncertain graph are actually the maximal uncertain cliques. But because uncertain quasi-clique does not have the downward closure property as uncertain clique and the state-of-the-art definition of the uncertain clique is fundamentally different from Definition 3 when $\gamma \neq 1$. The method of uncertain clique mining cannot be simply applied to the problem of uncertain quasi-clique mining.

Combining the content of Definition 3, it will boil down to the state-of-the-art definition of γ-quasi-clique in the deterministic graph if we remove the probabilities of the edges in the uncertain graph G. Compared with γ-quasi-clique, the uncertain quasi-clique mining spends much time in probability calculation and update. Therefore, the uncertain quasi-clique mining is more expensive than γ-quasi-clique. The method of γ-quasi-clique mining can be applied to the uncertain quasi-clique mining problem after adjustment. The process of the method is as follows: (1) find all the γ-quasi-cliques in the resulting deterministic graph by ignoring all edge probabilities of the given uncertain graph, and (2) get the maximal (α, γ)-quasi-cliques by filtering out the γ-quasi-cliques that do not satisfy Definition 3. We compare this method with the other three methods as another baseline method and finally decide not to use this method as the baseline method. For details, please refer to the first paragraph of Section 6.3.

For an uncertain graph, many (α, γ)-quasi-cliques are often very small and may be of no practical use. Similar to the problem of mining quasi-cliques from deterministic graphs [8], [9], it will be more useful to find large (α, γ)-quasi-cliques in uncertain graphs.

Problem Statement. Given an uncertain graph G and three parameters α, γ and min_s, mining the (α, γ)-quasi-cliques from G is equivalent to derive all the maximal (α, γ)-quasi-cliques of G in which every maximal (α, γ)-quasi-clique H satisfies $|V(H)| \geq min_s$.

Example 1. Consider an uncertain graph $G(V, E, p)$ shown in Fig. 1. Let $\alpha = 0.8$, $\gamma = 0.6$ and $min_s = 3$. Then, we can see that the induced graph of $X = \{v_1, v_2, v_6, v_7\}$ is a maximal (α, γ)-quasi-clique. With $k = \lceil \gamma \cdot (|X| - 1) \rceil = 2$, it has $P_{G(X)}(v_1, k) = P_{G(X)}(v_2, k) = 0.9965$, and $P_{G(X)}(v_6, k) = P_{G(X)}(v_7, k) = 0.9988$. Also, there does not exist a node set $Y \supset X$ such that $P_{G(Y)}(u, \lceil \gamma \cdot (|Y| - 1) \rceil) \geq 0.8$ for each node $u \in Y$.

3 The Proposed Algorithms

3.1 Basic Approach to Mine (α, γ)-Quasi-Clique

To show how to mine an (α, γ)-quasi-clique from the uncertain graph $G = (V, E, p)$, we first focus on the computation of $P_G(v, \lceil \gamma \cdot (|V| - 1) \rceil)$ for each node $v \in V$. As $P_G(v, \lceil \gamma \cdot (|V| - 1) \rceil)$ is the $\lceil \gamma \cdot (|V| - 1) \rceil$-probability of v in G, then we have the following equation:

$$P_G(v, \lceil \gamma \cdot (|V| - 1) \rceil) = \sum_{i=\lceil \gamma \cdot (|V|-1) \rceil}^{deg_{\tilde{G}}(v)} Pr_G(deg_G(v) = i) = \sum_{G' \subseteq G_p^k(v)} Pr(G'). \tag{2}$$

In Equation (2), $G_p^k(v)$ is the set of possible world graphs of G where v's degree is equal to k, i.e., $G_p^k(v) = \{G' | G' \subseteq \mathcal{G}_p, deg_{G'}(v) = k\}$.

A basic approach to finding all (α, γ)-quasi-cliques is to enumerate all the candidate subgraphs. Fig. 2 shows the enumeration procedure for G. Note that we use $X, cand(X)$ to represent the nodes that we have searched and the candidate nodes that can form an (α, γ)-quasi-clique. In order to avoid duplication, we sort the nodes in order and extend the set X by this order.

For example, in the enumeration procedure shown in Fig. 2, nodes are sorted in lexicographic order. In each iteration, a node v_i is selected, and the subgraph with or without

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

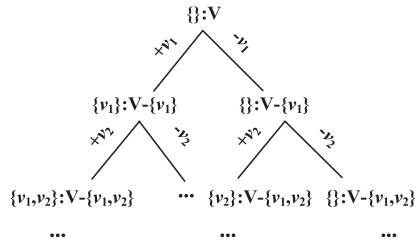


Fig. 2. Illustration of the enumeration procedure.

v_i will be checked. The pseudo-code of this algorithm is shown in Algorithm 1 and Algorithm 2.

Algorithm 1. Baseline Approach

Input: $G(V, E, p)$ is the uncertain graph; γ is the minimum degree threshold; α is the minimum probability threshold; min_s is the minimum size threshold.
Output: the node set R_{all}.

1: Remove the vertices whose degree is less than $\lceil \gamma \cdot (min_s - 1) \rceil$ to get a new node set $V^{'}$;
2: **if** $|V'| < min_s$ **then**
3: return NULL;
4: Compute the $P_{G(V')}(v, \lceil \gamma \cdot (|V^{'}| - 1) \rceil)$ values $\forall v \in V^{'}$;
5: **if** $G(V^{'})$ *is an* (α, γ)*-quasi-clique* **then**
6: $R_{all}.push(V^{'})$;
7: return R_{all};
8: $R_{all} \leftarrow NaiveEnum(\emptyset, V, \gamma, \alpha, min_s)$;
9: **Return** R_{all};

Algorithm 2. NaiveEnum(X, $cand(X)$, γ , α, min_s)

Input: X is the initial node set; $cand(X)$ is the candidate extension of X; γ is the minimum degree threshold; α is the minimum probability threshold; min_s is the minimum size threshold.
Output: the node set R.

1: **if** $|X| + |cand(X)| < min_s$ **then**
2: return R;
3: Compute the $P_{G(X)}(v, \lceil \gamma \cdot (|X| - 1) \rceil)$ values $\forall v \in X$;
4: **if** $|cand(X)| = \emptyset$ *and* $G(X)$ *is an* (α, γ)*-quasi-clique* **then**
5: $R \leftarrow R \cup X$ if $\nexists Y \in R$, such that $X \subset Y$;
6: return R;
7: $u \leftarrow$ choose the first node in $cand(X)$;
8: NaiveEnum($X, cand(X) \backslash \{u\}, \gamma, \alpha, min_s$);
9: $X \leftarrow X \cup \{u\}$, $cand(X) \leftarrow cand(X) \backslash \{u\}$;
10: NaiveEnum($X, cand(X), \gamma, \alpha, min_s$);
11: **Return** R;

First, Algorithm 1 removes the nodes with a degree smaller than $\lceil \gamma \cdot (|V| - 1) \rceil$ and gets the new node set $V^{'}$ (line 1). Before using a node v to extend X, Algorithm 1 uses the minimum size constraint to reduce the search space (lines 2-3). Then, $P_{G(V')}(v, \lceil \gamma \cdot (|V^{'}| - 1) \rceil)$ is computed for $v \in V^{'}$ (line 4). If $G(V^{'})$ is an (α, γ)-quasi-clique, it must be a maximal (α, γ)-quasi-clique, and the nodes are pushed into R_{all} (lines 5-7). After removing redundant nodes, Algorithm 2 is invoked to enumerate all maximal (α, γ)-quasi-cliques (line 8).

Algorithm 2 first judges whether $|X| + |cand(X)|$ meets the constraint of the minimum size threshold. If the constraint is not met, the algorithm can be terminated early (lines 1-2). Then Algorithm 2 judges whether $G(X)$ is an (α, γ)-quasi-clique. If there is no $Y \supset X$ in R, we push X into R and return R (lines 3-6). If $cand(X)$ is not empty, the algorithm will use the nodes in $cand(X)$ to extend X and recursively invokes itself until the result set is returned (lines 7-11). Note that we use a prefix tree to help us remove non-maximal quasi-cliques. This method has been used previously in paper [9]. The update complexity of prefix-tree is $O(n)$. The complexity of probability computing is $O(\gamma nm) = O(nm)$ as shown in Section 4.1. The time complexity of Algorithm 2 should be $O(2^n n^2 m)$ where n is the number of the nodes in the uncertain graph, and m is the number of the edges in the uncertain graph.

Example 2. Consider the uncertain graph G in Fig. 1. Let α = 0.8, γ = 0.6 and min_s = 3. First, there is no node removed from V, since the degrees of all the nodes are larger than $\lceil 0.6 \cdot (3-1) \rceil = 2$. Then we calculate $P_H(v_i, \lceil \gamma \cdot (|V(H)| - 1) \rceil)$ for $v_i \in V(H)$ based on Equation (2). Consider the node v_6, we can derive that $P_G(v_6, 6) = 0.0146 + 0.1687 + 0.816 = 0.9701 > 0.8$. Similarly, for v_1, it has $P_G(v_1, 6) = 0 < 0.8$. The graph G is not an (α, γ)-quasi-clique. X is initialized as empty and $cand(X)$ is set as V to invoke Algorithm 2. Assume v_1 is chosen as line 7 in Algorithm 2. So, it holds $P_{G(X)}(v, \lceil \gamma \cdot (|X| - 1) \rceil) = 0$ for $X \cup \{v_1\}$. The procedure is recursively invoked until X contains $\{v_1, v_2, v_6, v_7\}$. $P_{G(X)}(v_i, \lceil \gamma \cdot (|X| - 1) \rceil)$ for nodes in X with $i \in \{1, 2, 6, 7\}$ are 0.9965, 0.9965, 0.9988, and 0.9988 respectively. Thus, a maximal (α, γ)-quasi-clique is found.

3.2 Advanced Approach to Find (α, γ)-Quasi-Cliques

In this section, we present several pruning techniques that can significantly reduce the size of the graph. Given an uncertain graph G, intuitively, for each node v, it needs to consider two conditions if v belongs to a (α, γ)-quasi-clique. The first is the degree condition. If the degree of v in G is smaller than $\lceil \gamma \cdot (min_s - 1) \rceil$ in $\tilde{G}$, then v can not appear in any γ-quasi-cliques in $\tilde{G}$. In this scenario, v cannot belong to any (α, γ)-quasi-cliques in G. The other condition is the possibility of the edges. That is, for a node v whose degree equals $\lceil \gamma \cdot (min_s - 1) \rceil$, if the minimum probability of the edges connected to v is less than α or the product of the probability values of all the edges connected to v is less than α, then v would not be contained in any (α, γ)-quasi-cliques.

In the following, we will introduce several pruning techniques, which can be divided into two categories: Early Termination and Candidate Set Reduction.

3.3 Early Termination

In this section, we introduce the conditions to early terminate the enumeration procedure. Assume that X is a node set that is generated during the enumeration procedure as shown in Algorithm 2. For a node v in the graph $G(V, E, p)$, each edge (u_i, v) has an associated possibility $p(u_i, v)$, which indicates the possibility of the existence of this edge. We use $p_m(v)$ to denote the maximal possibility among all the edges (u, v) for the node v. Then, if v belongs to an (α, γ)-quasi-clique H, it indicates that $P_H(v, k) \geq \alpha$ for $k = \lceil \gamma \cdot (|V_H| - 1) \rceil$. Also, as shown in Equation (2), if $p(u_i, v)$ increases, $P_G(v, k)$ will also increase, and the corresponding (α, γ)-quasi-cliques may

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

contain more nodes. Based on this, we can derive the following results.

Theorem 1. *Given two uncertain graphs $G(V, E, p)$ and $G'(V, E, p')$ with $p'(e) \geq p(e)$ for each $e \in E$. Then for a maximal (α, γ)-quasi-clique H in the graph G, there exists at least one maximal (α, γ)-quasi-clique H' in G', such that $V_H \subseteq V_{H'}$.*

Based on Theorem 1, with a larger possibility associated with each edge, the (α, γ)-quasi-cliques may contain more nodes and edges. Given an uncertain graph $G(V, E, p)$, let $G'(V, E, p')$ be a graph which contains the same node set and edge set as G. The only difference is that for each edge (u, v) in G', the possibility associated with (u, v) is larger than that in G. Then, it is easy to derive that every (α, γ)-quasi-clique in G is a subset of at least one (α, γ)-quasi-clique in G'. If an (α, γ)-quasi-clique $G(S)$ in G' is found, it is the upper bound of the (α, γ)-quasi-clique $G(C)$ in G which satisfies $C \subseteq S$.

The Lemma 1 in [9] is based on the property that the diameter of the γ-quasi-clique in the deterministic graph is no more than 2 if $\gamma \in [0.5, 1]$ to prune the candidate set of the initial vertex set X. Note that we use [0.5,1] as the value range of γ by default, and the specific content is explained in Section 5. Inspired by Lemma 1 in [9], we propose our Lemma 1. Different from that, given an uncertain graph G, our Lemma 1 prunes the candidate vertex set of X based on the upper bound of the degree k_{max} that all the vertices $v \in X$ satisfy $P_G(v, k_{max}) \geq \alpha$.

Lemma 1. *Given an uncertain graph $G(V, E, p)$. For a node set $X \subset V$, if there exists a node set $Y \subset V$ such that $X \subset Y$ and $G(Y)$ is an (α, γ)-quasi-clique, it has $|Y| \leq \frac{k_{max}}{\gamma} + 1$, where k_{max} is the maximum k satisfies $P_G(v, k) \geq \alpha$ for each $v \in X$.*

Proof. As we know $X \subseteq Y$ and $G(Y)$ is an (α, γ)-quasi-clique. Then, for every node $v \in X$, we have $P_{G(Y)}(v, k) \geq \alpha$ where $k = \lceil \gamma \cdot (|Y| - 1) \rceil$. Since Y is a subset of V, we can derive the following inequality based on Theorem 1.

$$P_{G(Y)}(v, k) \leq P_G(v, k) \leq \sum_{i=k}^{|N_G(v)|} \binom{|N_G(v)|}{i} \cdot (p_m(v))^i \cdot (1 - p_m(v))^{|N_G(v)|-i}$$

Since $P_Y(v, k) \geq \alpha$, we have the following inequality:

$$\sum_{i=k}^{|N_G(v)|} \binom{|N_G(v)|}{i} \cdot (p_m(v))^i \cdot (1 - p_m(v))^{|N_G(v)|-i} \geq \alpha$$

Let L_{min} be the minimum number of nodes that can be added to X to form an (α, γ)-quasi-clique. Next, we can get the maximal k, i.e., k_{max}, satisfying the above inequality for all $v \in X$ where $k \in [\lceil \gamma \cdot (|X| + L_{min} - 1) \rceil,\ |N_G(v)|]$. And we are able to obtain that $\lceil \gamma \cdot (|Y| - 1) \rceil \leq k_{max}$. Then, we have $\lfloor k_{max}/\gamma \rfloor \geq \lfloor \lceil \gamma \cdot (|Y| - 1) \rceil / \gamma \rfloor \geq \lfloor \gamma \cdot (|Y| - 1)/\gamma \rfloor = |Y| - 1$. Therefore, we have $|Y| \leq \lfloor k_{max}/\gamma \rfloor + 1$. □

In Lemma 1, we consider the upper bound of the size for the (α, γ)-quasi-clique in G. Considering the situation that we aim to find an (α, γ)-quasi-clique in which it contains a node set X in the graph G, we use $indeg^X(u)$ to denote the number of nodes in X which are linked to u. If there exists a node $u \in X$, such that $indeg^X(u) < \lceil \gamma \cdot (|X| - 1) \rceil$, then at least one node should be added to X to increase the degree of u to form an (α, γ)-quasi-clique. We use $indeg_{min}(X)$ to denote the smallest $indeg^X(u)$ for all $u \in X$ and at least t nodes should be added to X to increase the degree of the nodes in X, then it has $L_{min} = min\{t | indeg_{min}(X) + t \geq \lceil \gamma \cdot (|X| + t - 1) \rceil\}$. Thus, we have the following property.

Property 1: Given an uncertain graph $G(V, E, p)$ and a node set $X \subset V$. If $L_{min} + |X| > \frac{k_{max}}{\gamma} + 1$, there does not exist an (α, γ)-quasi-clique containing X.

Property 1 is inspired by the Lemma 10 in [9]. Given an initial vertex set X, the candidate vertex set Y and L_{min}, which means the lower bound of the number of vertices that can be added to X. The Lemma 10 in [9] is based on the fact that the degree of each vertex $v \in X$ within $X \cup Y$ should be no small than $\lceil \gamma \cdot (|X| + L_{min} - 1) \rceil$. Different from that, our Property 1 is based on the fact that the upper bound of the degree k_{max} that all the vertices $v \in X$ satisfy $P_G(v, k_{max}) \geq \alpha$ should not satisfy $L_{min} + |X| > \frac{k_{max}}{\gamma} + 1$.

Example 3. Consider the uncertain graph G shown in Fig. 1. Let $\alpha = 0.95$, $\gamma = 0.8$ and $min_s = 3$, and assume that the initial node set $X = \{v_1, v_2\}$. Then, we can derive the extension candidate set of X that is $cand(X) = \{v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}\}$. It has $|N_G(v_1)| = 3$ and $|N_G(v_2)| = 4$. Then, we find that when $k = 3$, the inequality $\sum_{i=k}^{|N_G(v)|} \binom{|N_G(v)|}{i} \cdot (p_m(v))^i \cdot (1 - p_m(v))^{|N_G(v)|-i} \geq \alpha$ does not hold for all nodes in X, since $\sum_{i=k}^{|N_G(v_1)|} \binom{|N_G(v_1)|}{i} \cdot (p_m(v_1))^i \cdot (1 - p_m(v_1))^{|N_G(v_1)|-i} = \binom{3}{3} \cdot (0.98)^3 \cdot (0.02)^0 = 0.9412 < 0.95$. When $k = 2$, the inequality holds for all the nodes in X and the values are 0.99882 and 0.99997 respectively.

3.4 Candidate Set Reduction

In this section, we introduce the optimization techniques to reduce the candidate set to generate (α, γ)-quasi-cliques. Given an uncertain graph G and a node set X, we aim to enumerate all the node sets containing X and check whether they are (α, γ)-quasi-cliques or not. We use $cand(X)$ to denote the set of nodes that can form (α, γ)-quasi-cliques with X. As shown in Theorem 1, with the larger probability values of the edges incident to v, $Pr(v, k)$ will increase as well. Considering all the edges incident to the node v, $p_m(v)$ denotes the maximum possibility of all the edges incident to v in G. Then we have the following theorem.

Theorem 2. *Given an uncertain graph $G(V, E, p)$ and a node set X. Assume that there exists an (α, γ)-quasi-clique $G(Y)$ and $X \subset Y$. If a node v satisfies $P_G(v, k) < \alpha$ where $k = \lceil \gamma \cdot (|L_{min} + |X|| - 1) \rceil$ and $L_{min} = min\{t | indeg_{min}(X) + t \geq \lceil \gamma \cdot (|X| + t - 1) \rceil\}$, then we have $v \notin Y$.*

Proof. For a node $v \in cand(X)$, $P_G(v, k)$ is the k-probability of node v in G where $k = \lceil \gamma \cdot (|L_{min} + |X| - 1) \rceil$. We can get that $P_G(v, k) \leq \sum_k^{|N_G(v)|} \binom{|N_G(v)|}{k} (p_m(v))^i (1 - p_m(v))^{|N_G(v)|-k}$ where $N_G(v)$ is the set of neighbours of v in G and $p_m(v)$ is the maximal probability among the edges incident to v in G. If $\sum_k^{|N_G(v)|} \binom{|N_G(v)|}{k} (p_m(v))^i (1 - p_m(v))^{|N_G(v)|-k} < \alpha$, then the node v can be removed from the candidate set $cand(X)$.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

The nodes in the candidate set should be updated after removing v from $cand(X)$. We remove the nodes iteratively from $cand(X)$ until there is no node can be removed further. □

Theorem 2 prunes a vertex based on the upper bound of the probability that the degree of the vertex is no smaller than the minimum threshold L_{min} of the degree. L_{min} is the lower bound of the number of vertices that can be added to the initial vertex set X to form a γ-quasi-clique proposed in [19].

Algorithm 3. AdvEnum(X, $cand(X)$, γ, α, min_s)

Input: X is the initial node set; $cand(X)$ is the candidate extension of X; γ is the minimum degree threshold; α is the minimum probability threshold; min_s is the minimum size threshold.
Output: the node set R.

1: **if** $|cand(X)| = \emptyset$ *and* $G(X)$ *is an* (α, γ)*-quasi-clique* **then**
2: $R \leftarrow R \cup X$ if $\nexists Y \in R$, such that $X \subset Y$;
3: return R;
4: **if** $G(X \cup cand(X))$ *is an* (α, γ)*-quasi-clique* **then**
5: $R \leftarrow R \cup X \cup cand(X)$ if $\nexists Y \in R$, such that $X \cup cand(X) \subset Y$;
6: return R;
7: $L_{min} \leftarrow min\{t | indeg_{min}(X) + t \geq \lceil \gamma \cdot (|X| + t - 1) \rceil\}$;
8: remove v from $cand(X)$ if $P_{G(X \cup cand(X))}(v, k) < \alpha$; [Theorem 2]
9: compute $P_{G(X \cup cand(X))}(v, \lceil \gamma \cdot (|X| + |cand(X)| - 1) \rceil), \forall v \in X \cup cand(X)$;
10: **for** *each node* $v \in X$ **do**
11: k_v is the maximum value such that $P_{G(X \cup cand(X))}(v, k_v) \geq \alpha$;
12: $k_{max} \leftarrow \min_{v \in X} k_v$;
13: **if** $|X| + L_{min} \leq \frac{k_{max}}{\gamma} + 1$ **then**
14: $u \leftarrow$ choose a node in $cand(X)$;
15: $cand(X') \leftarrow cand(X) \backslash \{u\}$;
16: compute $P_{G(X \cup cand(X'))}(v, \lceil \gamma \cdot (|X| + |cand(X)| - 1) \rceil), \forall v \in X \cup cand(X)$;
17: AdvEnum($X, cand(X'), \gamma, \alpha, min_s$);
18: $X' \leftarrow X \cup \{u\}$;
19: Compute $P_{G(X')}(v, \lceil \gamma \cdot (|X| - 1) \rceil), \forall v \in X'$;
20: AdvEnum($X', cand(X'), \gamma, \alpha, min_s$);
21: **Return** R;

Example 4. Consider the uncertain graph $G(V, E, p)$ shown in Fig. 1. Let $\alpha = 0.8$, $\gamma = 0.8$ and $min_s = 3$. Let the initial node set $X = \{v_1, v_2\}$ and $cand(X) = \{v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}\}$. Then, we have $L_{min} = 0$ and $k = \lceil 0.8 \cdot (0 + 2 - 1) \rceil = 1$. For the node $v_{10} \in cand(X)$, we get that $P_G(v_{10}, 1) = 2 \cdot 0.5^2 + 0.5^2 = 0.75 < 0.8$. Thus, we can remove v_{10} from $cand(X)$.

Armed with the Early Termination and Candidate Set Reduction optimization techniques, we come up with Algorithm 3. We first check if X or $X \cup cand(X)$ is a maximal (α, γ)-quasi-clique. If it is, we put X into the result set R and return R (lines 1-6). Then, we remove some nodes that are not contained by any (α, γ)-quasi-cliques based on Theorem 2 (lines 7-9). Next, we derive the upper bound of the (α, γ)-quasi-clique's size which contains X based on Lemma 1 and stop the expansion of X based on Property 1 (lines 10-12). If $|X| + L_{min} \leq \frac{k_{max}}{\gamma} + 1$, the algorithm recursively calls itself to expand X until there is a result set R being returned (lines 13-21). We store all the (α, γ)-quasi-cliques in a prefix-tree. (α, γ)-quasi-cliques presented by internal nodes cannot be maximal. For the (α, γ)-quasi-clique represented by a leaf node, we will mark its subsets in the search tree as non-maximal quasi-cliques. Finally, the (α, γ)-quasi-cliques represented by the leaf nodes and without being marked as non-maximal quasi-cliques are valid results that will be put into the result set R. The time complexity of Algorithm 3 is $O(2^n n^2 m)$ where n is the number of the nodes in the uncertain graph, and m is the number of the edges in the uncertain graph.

Example 5. Consider the uncertain graph G shown in Fig. 1. Let $\alpha = 0.95$, $\gamma = 0.9$ and $min_s = 3$. The initial node set $X = \{v_1, v_2, v_6, v_7\}$ and the corresponding $cand(X) = \{v_8, v_9, v_{10}\}$. First, we compute L_{min} of X based on Theorem 2, which is $L_{min} = min\{t | 3 + t \geq \lceil 0.9 \cdot (4 + t - 1) \rceil\} = 0$ and $k = \lceil 0.9 \cdot (0 + 4 - 1) \rceil = 3$. For the node v_{10}, we get that $P_{G(X \cup cand(X))}(v_{10}, 3) = 0$. Then, we remove v_{10} from $cand(X)$. We can derive that $P_{G(X \cup cand(X))}(v, 2) \geq 0.95$ holds for all nodes $v \in X$ and $P_{G(X \cup cand(X))}(v, 3) \geq 0.9$ does not hold, because $P_{G(X \cup cand(X))}(v_2, 3) \leq \binom{3}{3} \cdot 0.98^3 \cdot 0.02^0 = 0.9412 < 0.95$. We have $k_{max} = 2$ and the upper bound of the (α, γ)-quasi-clique's size which contains X is $\lfloor \frac{k_{max}}{\gamma} \rfloor + 1 = \lfloor \frac{2}{0.9} \rfloor + 1 = 3$. Clearly, $|X| + L_{min}$ is equal to 4. According to Property 1, there is no need to extend X with the nodes in $cand(X)$, since we have $|X| + L_{min} > \lfloor \frac{k_{max}}{\gamma} \rfloor + 1$ (lines 9-13).

Note that Algorithm 3 involves many probability calculations. In order to reduce the cost of probability calculations, we introduce a dynamic programming method and propose a new probability update approach, which will be introduced in the following section.

4 Probability Calculation

In this section, we first introduce an efficient method to compute $P_G(v, \lceil \gamma \cdot (V - 1) \rceil)$ for $v \in V$ and then propose an incremental updating approach to maintain them.

4.1 Computing Node Probability

Given an uncertain graph $G = (V, E, p)$ where $n = |V|$ and $m = |E|$. For one node $v \in V$, the $\lceil \gamma \cdot (|V| - 1) \rceil$-probability of v in G can be expressed as:

$$P_G(v, \lceil \gamma \cdot (|V| - 1) \rceil) = \sum_{i = \lceil \gamma \cdot (|V| - 1) \rceil}^{deg_{\tilde{G}}(v)} Pr_G(deg_G(v) = i) = 1 - \sum_{i=0}^{\lceil \gamma \cdot (|V| - 1) \rceil - 1} Pr_G(deg_G(v) = i) \quad (3)$$

Note that $P_G(v, \lceil \gamma \cdot (|V| - 1) \rceil)$ is the sum of $Pr(deg(v) = i)$ where $i \in [\lceil \gamma \cdot (|V| - 1) \rceil, |N(v)|]$. Let $E_G(v) = \{e_1, e_2, ..., e_{d_{\tilde{G}}(v)}\}$ be the set of edges incident to node v in the uncertain graph G, and $E^h_{\tilde{G}}(v) = \{e_1, e_2, ..., e_h\} (0 \leq h \leq d_{\tilde{G}}(v))$ is a subset of $E_G(v)$. Let $G_h = \{V, E \backslash (E_G(v) \backslash E^h_{\tilde{G}}(v)), P\}$ be the uncertain subgraph of G without the edges in $E_G(v) \backslash E^h_{\tilde{G}}(v)$.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

Since $f(h,i) = Pr(d_{G_h}(v) = i)$, then for $h \in [1,\ d_{\tilde{G}}(v)]$ we have

$$f_v(h,i) = p_{e_h} f_v(h-1, i-1) + (1 - p_{e_h}) f_v(h-1, i) \tag{4}$$

We need to compute all $f_v(h,i)$ values for all $h \in [1,\ d_{\tilde{G}}(v)]$, $i \in [0,\ \lceil \gamma \cdot (|V|-1) \rceil]$ so as to get the final $f_v(d_{\tilde{G}}(v), i)$, which corresponds to $Pr(d_{G_h}(v) = i)$. It will be more efficient to update the $v's$ i-probability in G_h with an increasing order of i. Based on Equation (3), we can easily get that $P_G(v, \lceil \gamma \cdot (|V|-1) \rceil)$, and it can be derived in $O(\lceil \gamma \cdot (|V|-1) \rceil d_{\tilde{G}}(v))$ time. Computing the $\lceil \gamma \cdot (|V|-1) \rceil$-probabilities cfor all nodes $v \in V$ in G will cost $O(\sum_{v \in V} \lceil \gamma \cdot (|V|-1) \rceil\ d_{\tilde{G}}(v))$. Therefore, the complexity can be more compactly expressed as $O(\gamma nm)$.

For a node u in the initial node set X, we can immediately get the $P_X(u,k)$ for $k \in [\lceil \gamma \cdot (|X|-1) \rceil,\ d_{\tilde{G}(X)}(u)]$ based on the following equation.

$$\mathcal{P}_u(h,j) = p_{e_h} \mathcal{P}_u(h-1, j-1) + (1 - p_{e_h}) \mathcal{P}_u(h-1, j) \tag{5}$$

We can also observe that $\mathcal{P}_u(h,j) = P(u|e_1, ..., e_h, j) = Pr(deg(u|e_1, ..., e_h) \geq j)$ for $h \in [0,\ d_X(u)]$, and $\{e_1, ..., e_h\}$ is the set of edges incident to u in $G(X)$. Since $\mathcal{P}_u(h,0) = 1$ for $h \in [0,\ d_X(u)]$ and $\mathcal{P}_u(0,j) = 0$ for $j \in [1,\ d_X(u)]$, we can get $\mathcal{P}_u(h,j)$ for $h \in [1,\ d_X(u)]$, $j \in [0,\ h]$ based on Equation (5). Then, we can calculate $P_X(u, \lceil \gamma \cdot (|X|-1) \rceil)$ for all $u \in X$. Two methods are given for the probability calculation of the vertex in X an the vertex in $V \backslash X$, as shown in Equation (5) and Equation (4) respectively. Equation (4) refers to the probability calculation method proposed in [12].

4.2 Updating the Probability Value

Given an uncertain graph $G(V,E,p)$ and a node $v \in V$. We now consider how to update the $\lceil \gamma \cdot (|V|-1) \rceil$-probability of v in G when an edge incident to v was removed. We can get the following equation that $P_G(v,k) = 1 - \sum_{i=0}^{k-1} Pr_G(deg(v) = i)$ where $k = \lceil \gamma \cdot (|V|-1) \rceil$. If we remove a node u from V, we update $P_{G\backslash\{u\}}(v,k)$ for all $v \in V \backslash \{u\}$. As we know, $Pr_G(deg(v) = k) = Pr_{G\backslash\{u\}}(deg(v) = k) \cdot (1 - P(e)) + Pr_{G\backslash\{u\}}(deg(v) = k-1) \cdot P(e)$ where u is the node removed from the uncertain graph G and e is the edge between v and u. We can update $Pr_{G\backslash\{u\}}(deg(v) = k)$ with the following equation.

$$Pr_{G\backslash\{u\}}(deg(v) = k) = \frac{Pr_G(deg(v) = k) - p(e)(Pr_{G\backslash\{u\}}(deg(v) = k-1))}{1 - p(e)} \tag{6}$$

We can set $Pr_{G\backslash\{u\}}(deg(v) = 0) = \frac{Pr_G(deg(v)=0)}{1-p(e)}$ and apply Equation (6) to compute the remaining $Pr_{G\backslash\{u\}}(deg(v) = i)$ for $i \in [1,\ \lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil - 1]$. Then we can get $P_{G\backslash\{u\}}(v, \lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil)$ based on Equation (3). Updating the $\lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil$-probability of v in $G\backslash\{u\}$ globally takes $O(\lceil \gamma \cdot (|V|-1) \rceil)$, which is better than recalculating. Therefore, the running time of updating the probabilities of the nodes with removing the connected node u is $O(\lceil \gamma \cdot (|V|-1) \rceil |N_G(u)|)$.

For the problem of the k-core mining on the uncertain graph [12], when vertex u is removed, the probability of all the other vertices connected to u with a degree no less than k should be updated again. Different from that, for our problem of quasi-clique in the uncertain graph, if $C(V,E)$ is the current subgraph, we need to verify whether the probability that each vertex has a degree being no less than $\lceil \gamma \cdot (|V|-1) \rceil$ is greater than or equal to α. If there is a vertex u that does not satisfy the condition, we should remove u from C. Combing with Equation (7), we can find that the probabilities of some vertices' degree being greater than or equal to $\lceil \gamma \cdot (|V|-1) \rceil$ increase after we removing u. Therefore, for those vertices that have met the probability requirement before u is removed from C, even if we remove u, they still meet the requirement. Therefore, there is no need to update the probabilities of those vertices in this case. Based on this property, we propose a new updating strategy below to further reduce unnecessary probability calculations.

Algorithm 4. $UpdatePr(X, cand(X), V_{del}, G)$

Input: X is a node set; $cand(X)$ is the candidate extension of X; V_{del} is the set of removing nodes; $G(V,E,p)$ is an uncertain graph.

```
1: s ← |X| + |cand(X)|;
2:    for u ∈ V_del do
3: for (u,v) ∈ E(G) do
4:       b ← ⌈γ·(s+|V_del|−1)⌉ ≠ ⌈γ·(s+|V_del|−2)⌉ ;
5:       if delay[v] = NULL and b = true and
         P_{G'}(v, ⌈γ·(s+|V_del|−1)⌉) ≥ α OR
         delay[v] ≠ NULL and b = true then
6:          delay[v].push[p(e_{u,v})];
7:       else
8:          delay[v].push[p(e_{u,v})];
9:          update the Pr(deg(v) = k) for k ∈ [0, ⌈γ·(s+|V_del|−
            2)⌉ − 1] combining with p' for p' ∈ delay[v] based on
            Equation (6);
10:   G ← G − {u}, V_del ← V_del − {u};
```

First, let us introduce some observations. Given an uncertain graph $G(V,E,p)$ and node $v \in V$. Let $e = (u,v)$ be the edge between u and v. Let $G' = (V\backslash\{u\}, E\backslash\{e\}, p)$ be the subgraph of G by removing u from V. If $\lceil \gamma \cdot (|V|-1) \rceil \neq \lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil$. We can get the following inequality.

$$\begin{aligned} &P_G(v, \lceil \gamma \cdot (|V|-1) \rceil) = \\ &\quad P_{G'}(v, \lceil \gamma \cdot (|V|-1) \rceil) + p(e) \cdot Pr_{G'}(deg_{G'}(v) = \\ &\lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil) \leq P_{G'} \cdot (v, \lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil) \end{aligned} \tag{7}$$

For node v linked to the removing node u and it satisfies that $\lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil \neq \lceil \gamma \cdot (|V|-1) \rceil$, there holds $P_{G\backslash\{u\}}(v, \gamma \cdot (|V\backslash\{u\}|-1) \rceil) > P_G(v, \gamma \cdot (|V|-1) \rceil)$ based on Inequality (4). If $P_G(v, \lceil \gamma \cdot (|V|-1) \rceil) \geq \alpha$, then $P_{G\backslash\{u\}}(v, \lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil) \geq P_G(v, \lceil \gamma \cdot (|V|-1) \rceil) \geq \alpha$. So there is no need to update the probability of v immediately since the $\lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil$-probability of v in $G\backslash\{u\}$ still satisfies the constraints of (α,γ)-quasi-clique even if u was removed.

If node v satisfies $\lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil = \lceil \gamma \cdot (|V|-1) \rceil$, we need to update the $Pr_{G\backslash\{u\}}(deg(v) = i)$ of v for $i \in [0,\ \lceil \gamma \cdot (|V\backslash\{u\}|-1) \rceil]$ no matter node v satisfies $P_G(v, \lceil \gamma \cdot (|V|-1) \rceil) \geq \alpha$ or $P_G(v, \lceil \gamma \cdot (|V|-1) \rceil) < \alpha$.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

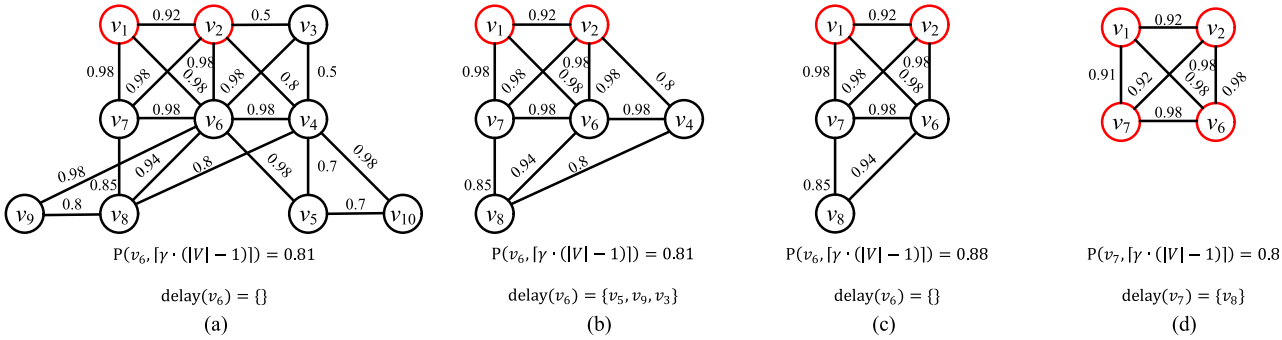


Fig. 3. Illustration of the probability updating procedure.

For node v does not link to u, we can always get $P_{G\backslash\{u\}}(v, \lceil\gamma \cdot (|V\backslash\{u\}| - 1)\rceil) \geq P_G(v, \lceil\gamma \cdot (|V| - 1)\rceil)$. If $P_G(v, \lceil\gamma \cdot (|V| - 1)\rceil) \geq \alpha$, the $\lceil\gamma \cdot (|V\backslash\{u\}| - 1\rceil$-probability of v in $G\backslash\{u\}$ always satisfies the constraints of (α, γ)-quasi-clique.

Let u be the node that is inserted into the initial node set X. For node v incident to u in X, we can update $P_X(v, \lceil\gamma \cdot (|X| - 1)\rceil)$ based on Equation (5). For node w that does not link to u in X, we can calculate $P_X(w, \lceil\gamma \cdot (|X| - 1)\rceil)$ immediately based on the previously calculation.

Algorithm 4 shows the pseudo-codes of updating the probabilities. For a node set X and its extensible candidate set $cand(X)$, we need update the probability values for all nodes in $X \cup cand(X)$ utilizing the set of removing nodes V_{del}. Let v be the node in $X \cup cand(X)$. For node $u \in V_{del}$, there are two updating strategies: *(a)* there is an edge between u and v in G; *(b)* there is no edge between u and v in G'. For *(a)*, we need check whether $delay[v]$, the set of nodes incident to v, is empty. If $delay[v]$ is empty, $\lceil\gamma \cdot (|V| - 1)\rceil \neq \lceil\gamma \cdot (|V\backslash\{u\}| - 1)\rceil$, $P_G(v, \lceil\gamma \cdot (|V)| - 1)\rceil) \geq \alpha$ or $dealy[v]$ is not empty, $\lceil\gamma \cdot (|V| - 1)\rceil \neq \lceil\gamma \cdot (|V\backslash\{u\}| - 1)\rceil$, we can delay the updating of v's probability and push $p(e_{u,v})$ into $delay[v]$ (lines 3-6). Else, we push $p(e_{u,v})$ into $delay[v]$ and update the probability of v combining all the probability values in $delay[v]$ based on Equation (6) (lines 7-9). For *(b)*, if $P_G(v, \lceil\gamma \cdot (|V| - 1)\rceil) < \alpha$, we can immediately get $P_G(v, \lceil\gamma \cdot (|V\backslash\{u\}| - 1)\rceil)$ since we have stored $Pr_G(deg(v) = k)$ for $k \in [0, \lceil\gamma \cdot (|V| - 1)\rceil - 1]$. Suppose that Δ_u is the number of the neighbors of u in G. Therefore, the process of updating the probabilities after removing u takes $O(\Delta_u \lceil\gamma \cdot (|V| - 1)\rceil)$, and it is no more than $O(\lceil\gamma \cdot (|V| - 1)\rceil |N_G(u)|)$.

Example 6. Let $\alpha = 0.8$, $\gamma = 0.8$ and $min_s = 3$. In this example, we use the example in Fig. 3to illustrate the efficiency of Algorithm 4. Fig. 3a illustrates an uncertain graph with 10 nodes. First, we can get the $\lceil\gamma \cdot (|V| - 1)\rceil$-probability of v_6 in G which is $P_G(v_6, \lceil 0.8 \cdot (10 - 1)\rceil) = 0.81 > 0.8$. Assume $X = \{v_1, v_2\}$ and $cand(X) = \{v_{1-10}\}$, we can have that L_{min} and k of v_{10} are 0 and 1 respectively. Then, we can remove v_{10} from $cand(X)$ based on Theorem 2. We can observe that the degrees of v_5 and v_9 are 2, which is equal to $\lceil 0.8 \cdot (3 - 1)\rceil$ and the probability product of all the edges connected to v_5 (or v_9) is less than 0.8. So, we can remove $\{v_5, v_9\}$ and push $\{v_5, v_9\}$ into $delay[v_6]$ since $\lceil 0.8 \cdot (9 - 1)\rceil \neq \lceil 0.8 \cdot (8 - 1)\rceil$ and $\lceil 0.8 \cdot (8 - 1)\rceil \neq \lceil 0.8 \cdot (7 - 1)\rceil$. The current $X = \{v_1, v_2\}$ and $cand(X) = \{v_3, v_4, v_6, v_7, v_8\}$. Next, we push v_3 into X and we find that there is no (α, γ)-quasi-clique which contains the current X which means that we can remove v_3 from $cand(X)$. Fig. 3b shows the remaining graph with $X = \{v_1, v_2\}$ and $cand(X) = \{v_4, v_6, v_7, v_8\}$. Since $\lceil 0.8 \cdot (7 - 1)\rceil \neq \lceil 0.8 \cdot (6 - 1)\rceil$, we push v_3 into $delay[v_6]$.

After that, We find that v_4 cannot be contained by any (α, γ)-quasi-clique like v_3, which means we can remove v_4 from $cand(X)$. Since $\lceil 0.8 \cdot (5 - 1)\rceil = \lceil 0.8 \cdot (6 - 1)\rceil$, we should update $Pr(deg(v) = k)$ for $k \in [0, 3]$ with $p(e(v_5, v_6))$, $p(e(v_9, v_6))$, $p(e(v_3, v_6))$ and $p(e(v_4, v_6))$ based on the Equation (6). If we immediately update the k-probability values of v_6 when the other nodes are removed, the range of the k we need to update are $[0, 5]$, $[0, 4]$, $[0, 3]$ and $[0, 3]$ respectively. We can find that Algorithm 4 reduces the range of k we need to update.

The remaining graph is shown in Fig. 3c, we find v_8 could be removed from $cand(X)$ based on the new pruning method used in the probabilistic graph. Since $\lceil 0.8 \cdot (5 - 1)\rceil \neq \lceil 0.8 \cdot (4 - 1)\rceil$ and $P_{G(X \cup cand(X))}(v_7, \lceil 0.8 \cdot (5 - 1)\rceil) = P_{G(X \cup cand(X))}(v_7, 4) = 0.8$, we can remove v_8 without updating the k-probability of v_7 with $k = \lceil 0.8 \cdot (4 - 1)\rceil = 3$. Finally, We get the (α, γ)-quasi-clique as shown in Fig. 3d, which is $\{v_1, v_2, v_6, v_7\}$.

5 Optimizations

In this section, we describe several pruning techniques used by existing works. Note that because there are several methods that are not fully applicable to uncertain graphs, we have made appropriate adjustments and still described here as contributions to the existing work. The optimizations can speed up the enumeration procedure by inserting a set of nodes directly to the node set X and reduces the number of iterations.

Theorem 3. *Let $H = (X, E_X, p)$ be an (α, γ)-quasi-clique we have found in the uncertain graph G. Then for the node set X, the induced graph $\tilde{G}(X)$ in $\tilde{G}$ is a γ-quasi-clique.*

Proof. According to Definition 3, we know that $P(v, \lceil\gamma \cdot (|X| - 1)\rceil) \geq \alpha$ holds for all $v \in X$. Since $\alpha \in (0, 1]$, we can get $deg_{\tilde{G}}(v) \geq \lceil\gamma \cdot (|X| - 1)\rceil$ for all $v \in X$, which confirms that $\tilde{G}(X)$ is a γ-quasi-clique. □

In [8], Pei *et al.* developed an upper bound of the diameter of a γ-quasi-clique based on γ. Consider a deterministic graph $\tilde{G} = (V, E)$ and two node sets $X \subset Y \subseteq V$. If $\tilde{G}(Y)$ is a γ-quasi-clique, for every node $v \in (Y - X)$, we have $v \in \cap_{v \in X} N_k^{\tilde{G}}(v)$ where k is the upper bound of the diameter of a γ-quasi-clique. The nodes that are not in $\cap_{v \in X} N_k^{\tilde{G}}(v)$ can be removed from $cand(X)$. The relationship between the

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

diameter of γ-quasi-clique and γ is shown in the following formula:

$$diam(\tilde{G})\begin{cases}=1 & if 1 \geq \gamma > \frac{n-2}{n-1}\\ \leq 2 & if \frac{n-2}{n-1} \geq \gamma \geq \frac{1}{2}\\ \leq 3\lfloor\frac{n}{\gamma(n-1)+1}\rfloor - 3 & if \frac{1}{2} > \gamma \geq \frac{2}{n-1} and\\ & nmod(\gamma(n-1)+1) = 0\\ \leq 3\lfloor\frac{n}{\gamma(n-1)+1}\rfloor - 2 & if \frac{1}{2} > \gamma \geq \frac{2}{n-1} and\\ & nmod(\gamma(n-1)+1) = 1\\ \leq 3\lfloor\frac{n}{\gamma(n-1)+1}\rfloor - 1 & if \frac{1}{2} > \gamma \geq \frac{2}{n-1} and\\ & nmod(\gamma(n-1)+1) \geq 2\\ \leq n-1 & if \gamma = \frac{1}{n-1}.\end{cases} \tag{8}$$

According to [8], we may be interested in γ-quasi-cliques with a reasonably-large γ, since the diameter of the γ-quasi-clique changes dramatically with respect to γ [8]. Therefore, Pei *et al.* [8] suggested that γ should be bounded in [0.5,1]. In this paper, we use [0.5,1] as the value range of γ by default.

Given an uncertain graph $G = (V, E, p)$ and a node set $X \subset V$. Based on Theorem 3, Pei *et al.* used the diameter of a γ-quasi-clique being no more than 2 to get the candidate set of X with the following formula:

$$cand(X) = \bigcap_{v \in X} N_2^G(v). \tag{9}$$

The following three lemmas are the versions after appropriate adjustments to the pruning methods in the paper [9], and the main ideas are still the same as the corresponding pruning methods in paper [9]. Below, we first introduce the optimization method to reduce the cost of extending X with $cand(X)$.

Lemma 2. *Given an uncertain graph $G(V, E, p)$ and a node set $X \subset V$, assume that there exists an (α, γ)-quasi-clique $G(Y)$ in which $G(Y) \subset G$ and $X \subset Y$. For $v \in X$, if $degree(v) = \lceil\gamma \cdot (|X| + L_{min} - 1)\rceil$ and $P_{G(Y)}(v, \lceil\gamma \cdot (|X| + L_{min} - 1)\rceil) \geq \alpha$, then $N(v) \subset Y$.*

Proof. Let $v \in X$ satisfies that $degree(v) = \lceil\gamma \cdot (|X| + L_{min} - 1)\rceil$ and $P_{G(Y)}(v, \lceil\gamma \cdot (|X| + L_{min} - 1)\rceil) \geq \alpha$. Node u is a node such that $u \in cand(X)$ and $(u, v) \in E$. Suppose that $u \notin Y$, then $P_Y(v, \lceil\gamma \cdot (|X| + L_{min} - 1)\rceil) = 0 < \alpha$. It contradicts the fact that Y is an (α, γ)-quasi-clique. □

Similar to Lemma 2, we consider the probability constraint for an (α, γ)-quasi-clique. Given an uncertain graph G and a candidate node set X for (α, γ)-quasi-clique, consider a node $v \in X$ and another node set Y as $X \cup N(v)$. If $P_{G(Y)}(v, \lceil\gamma \cdot (|X| + L_{min} - 1)\rceil) = \alpha$, it indicates that all the neighbors of v should be included in the candidate node set if v satisfies the probability constraint for (α, γ)-quasi-clique. If any neighbor does not appear in the candidate node set with v, then v should be removed from the candidates. Therefore, we have the following lemma.

Lemma 3. *Given an uncertain graph $G(V, E, p)$ and a node set $X \subset V$, assume that there exists an (α, γ)-quasi-clique $G(Y)$ in which $G(Y) \subset G$ and $X \subset Y$. Then, for a node $v \in X$, if $degree(v) > \lceil\gamma \cdot (|X| + L_{min} - 1)\rceil$ and $P_{G(Y)}(v, \lceil\gamma \cdot (|X| + L_{min} - 1)\rceil) = \alpha$, then $N(v) \subset Y$.*

Proof. Let node u be a node such that $u \in cand(X)$ and $(u, v) \in E$. Suppose that $u \notin Y$, then $P_{G(Y)}(v, \lceil\gamma \cdot (|X| + L_{max} - 1)\rceil) < P_{G(X \cup cand(X))}(v, \lceil\gamma \cdot (|X| + L_{max} - 1)\rceil) = \alpha$. It contradicts the fact that Y is an (α, γ)-quasi-clique. □

For nodes satisfy Lemma 2 or Lemma 3, we refer to them as *key nodes*. We need to get all the key nodes for a node set X. If there is a key node v, then we can add the nodes (in $cand(X)$) that are adjacent to v into the set X.

Algorithm 5. AdvEnumOpt(X, $cand(X)$, γ, α, min_s)

Input: X is the initial node set; $cand(X)$ is the candidate extension of X based on Equation 9; γ is the minimum degree threshold; α is the minimum probability threshold; min_s is the minimum size threshold.
Output: the node set R.

1: **if** $|cand(X)| = \emptyset$ *and* $G(X)$ *is an* (α, γ)*-quasi-clique* **then**
2: $R \leftarrow R \cup X$ if $\nexists Y \in R$, such that $X \subset Y$;
3: return R;
4: **if** $G(X \cup cand(X))$ *is an* (α, γ)*-quasi-clique* **then**
5: $R \leftarrow R \cup X \cup cand(X)$ if $\nexists Y \in R$, such that $X \cup cand(X) \subset Y$;
6: return R;
7: $L_{min} \leftarrow min\{t | indeg_{min}(X) + t \geq \lceil\gamma \cdot (|X| + t - 1)\rceil\}$;
8: $G' \leftarrow G(X \cup cand(X))$;
9: Remove v from $cand(X)$ and put v into V_{del} if $P_{G(X \cup cand(X))}(v, k) < \alpha$; [Theorem 2]
10: $UpdatePr(X, cand(X), V_{del}, G')$;
11: **for** *each node* $v \in X$ **do**
12: k_v is the maximum value such that $P_{G(X \cup cand(X))}(v, k_v) \geq \alpha$;
13: $k_{max} \leftarrow \min_{v \in X} k_v$;
14: **for** $v \in X$ **do**
15: **if** v *is a key node* **then**
16: $X \leftarrow X \cup N(v)$; [Lemma 2, Lemma 3]
17: Find the useless node u of X; Sort the nodes in $cand(X)$ such that nodes in $U_X(u)$ are after all the other nodes; [Lemma 4]
18: Update L_{min} and k_{max};
19: **if** $|X| + L_{min} \leq \frac{k_{max}}{\gamma} + 1$ **then**
20: **for** *all* $w \in cand(X) \backslash U_X(u)$ **do**
21: $X' \leftarrow X \cup \{w\}$, $cand(X) \leftarrow cand(X) \backslash \{w\}$;
22: $cand(X') \leftarrow \cap_{v \in X} N_2^{G(X \cup cand(X))}(w)$;
23: $V_{del} \leftarrow V_{del} \cup (cand(X) \backslash cand(X'))$;
24: Update $P_{G(X')}(v, \lceil\gamma \cdot (|X'| - 1)\rceil), \forall v \in X'$ and $UpdatePr(X', cand(X'), V_{del}, G')$;
25: AdvEnumOpt($X', cand(X'), \gamma, \alpha, min_s$);
26: **Return** R;

Lemma 4. *Given an uncertain graph $G(V, E, p)$. Let X be the initial node set and $cand(X)$ be the candidate extension of X. If $\lceil\gamma \cdot (i-1)\rceil \neq \lceil\gamma \cdot i\rceil$ holds for all $i \in [|L_{mim}| + |X| + 1, \frac{k_{max}}{\gamma} + 1]$, then we can get node $u \in cand(X)$ and $P_{G(X \cup cand(X))}(u, \lceil\gamma \cdot (|X| + |cand(X)| - 1)\rceil) \geq \alpha$. If $\forall v \in X$ and $(v, u) \notin E$, then $P_{G(X \cup cand(X))}(v, \lceil\gamma \cdot (|X| + |cand(X)| - 1)\rceil) \geq \alpha$. For a node set Y such that $G(Y)$ is an (α, γ)-quasi-clique and $Y \subset (X \cup (cand(X) \cap N_{\tilde{G}}(u) \cap (\cap_{v \in X and (u,v) \notin E}(N_{\tilde{G}}(v)))))$, $G(Y)$ is not a maximal (α, γ)-quasi-clique.*

Proof. Since $G(Y)$ is an (α, γ)-quasi-clique, $P_{G(Y)}(v, \lceil\gamma \cdot (|Y| - 1)\rceil) \geq \alpha$ holds for all $v \in Y$. Then we will discuss whether $G(Y') = G(Y \cup \{u\})$ is an (α, γ)-quasi-clique. For

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

the node u, we can have $P_{G(Y')}(u, \lceil \gamma \cdot (|Y'|-1) \rceil) \geq P_{G(X \cup cand(X))}(u, \lceil \gamma \cdot (|X|+|cand(X)|-1) \rceil) \geq \alpha$ based on Equation (7). For the node v such that $(u,v) \notin E$ we can have $P_{G(Y')}(v, \lceil \gamma \cdot (|Y'|-1) \rceil) \geq P_{G(X \cup cand(X))}(v, \lceil \gamma \cdot (|X|+|cand(X)|-1) \rceil) \geq \alpha$. The same result holds for v such that $(u,v) \in E$ and $v \in Y - X$. □

Let $U_X(u)$ denote the set of nodes with respect to X, and $U_X(u) = cand(X) \cap N_{\hat{G}}(u) \cap (\cap_{v \in X and (u,v) \notin E}(N_{\hat{G}}(v)))$. We can find a node u that maximize the size of $U_X(u)$, and u is an useless node. Then, we can make use of the nodes in $cand(X) \backslash U_X(u)$ to extend X.

Based on the above lemmas, we can obtain our optimized algorithm which is shown in Algorithm 5. We first check if X or $X \cup cand(X)$ is a maximal (α, γ)-quasi-clique. If it is, we put X into the result set R and return R (lines 1-6). Then, we remove some nodes that are not contained by any (α, γ)-quasi-cliques based on Theorem 2 (lines 7-10). Next, we derive the upper bound of the (α, γ)-quasi-clique's size which contains X based on Lemma 1 and stop the expansion of X based on Property 1 (lines 11-13). According to Lemmas 2 and 3, we can find all key nodes in X and put all the neighbors of the key nodes into X (lines 14-16). For the vertex set X, we finds its useless node u, and put the nodes in $U_X(u)$ after all the other nodes in $cand(X)$ (line 15). Only the nodes in $cand(X) \backslash U_X(u)$ are used to extend X. If $|X| + L_{min} \leq \frac{k_{max}}{\gamma} + 1$, the algorithm recursively calls itself to expand X until there is a result set R being returned (lines 19-26). The time complexity of Algorithm 5 is $O(n^3)$ where n is the number of the nodes in the uncertain graph.

6 Experiments

6.1 Experimental setup

To enumerate all maximal (α, γ)-quasi-cliques, we implement three algorithms called Baseline, SeBaseline, AdvEnum, and AdvEnum+. Baseline is based on NaiveEnum algorithm, which is shown in Algorithm 1 and Algorithm 2. SeBaseline is another baseline approach with the following steps (1) Find all the γ-quasi-cliques in the resulting deterministic graph by ignoring all edge probabilities of the given uncertain graph; (2) Process all γ-quasi-clique and filter out all γ-quasi-clique that do not satisfy Definition 3; (3) Return all the maximal γ-quasi-clique as the maximal (α, γ)-quasi-cliques. AdvEnum is the approach with the AdvEnum algorithm, as shown in Algorithm 3. AdvEnum+ is AdvEnumOpt with the probability update method proposed in Section 5.

All algorithms are implemented in C++. All the experiments are conducted on a server with two Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz CPUs and 256 GB main memory. CeontOS 7.4 X86_64 operating system with Linux kernel 3.10.0.

Datasets. As shown in Table 2, we use five real-world graphs to evaluate the efficiency of all the algorithms in the experiments. The detailed information of these datasets are described as follows. CORE is a protein-protein interaction (PPI) network provided by Krogan *et al.* [20]. The data set contains 2,708 nodes and 7,123 edges, where the node denotes the protein and the edge denotes the interaction between two proteins. Each edge is associated with a probability, which represents the probability of a connection between the corresponding two proteins. The probabilities of edges are distributed in the $[0.27,\ 0.98]$. There are around 20 percent edges that have a probability no less than 0.98, an there is no edge with probability less than 0.27. Bitcoin and AskUbuntu can be downloaded from the Stanford network dataset collection (snap.stanford.edu). Bitcoin is a weighted graph and AskUbuntu is a temporal graph. For these two datasets, we adopt a standard method used in [21], [22] to generate the probabilistic graphs for Bitcoin and AskUbuntu. In particular, for each edge (u,v), we make use of an exponential cumulative distribution with mean $\lambda = 2$ to the weight of (u,v) to generate a probability (i.e., $p(u,v) = 1 - exp(-w_{uv}/\lambda)$ [21], [22]). Hyves is an unweighted network graph which can be downloaded from the Koblenz Network Collection (http://konect.cc/networks/). For this unweighted network, we generate a probability for each edge following a uniform distribution. The statistic information of our all datasets are provided in Table 2.

TABLE 2
Datasets

Dataset	n	m	d_{max}
CORE	2,708	7,123	141
Bitcoin	5,881	35,592	795
AskUbuntu	157,522	455,691	5,401
Amazon	334,863	925,872	549
Hyves	1,402,673	2,777,419	31,883

Parameters. There are three parameters in our algorithms: α, γ and min_s. The parameter α is chosen from the interval $[0.6,\ 0.9]$ with a default value of $\alpha = 0.7$. γ is selected from the interval $[0.6,\ 0.9]$ with a default value of $\gamma = 0.8$. min_s is a positive integer with a default value of 8. Unless otherwise specified, the values of the other parameters are set to the default value when varying a parameter in experiments.

6.2 Effectiveness testing

In this section, we conduct a case study on a protein-protein interaction (PPI) network to evaluate the effectiveness of the proposed algorithms. Following [23], we used a PPI network CORE which is an uncertain graph provided by Krogan *et al.* [20], because we can obtain the ground truth clustering results on the CORE dataset on the basis of the MIPS protein database [23]. CORE contains 2,708 nodes and 7,123 edges where a node represents a protein and each edge represents the interaction between two proteins. Based on the ground truth, we are capable of computing the number of the true positive (TP), the number of the false positive (FP), as well as the precision (PR = TP/(TP+FP)) obtained by a variety of algorithms. More specifically, TP represents the number of correctly matched interaction in predicted complexes with that in MIPS, and FP represents the total number of interactions in predicted complexes minus TP. We compute TP, FP and PR with the method used by Kollios *et al.* [23]. We compare the proposed algorithm (AdvEnum+) with two state-of-art protein complex clustering algorithms USCAN and PCluster based on the TP, FP and PR metrics. For USCAN and PCluster, we adopt the default parameter values as used in their original experiments. The

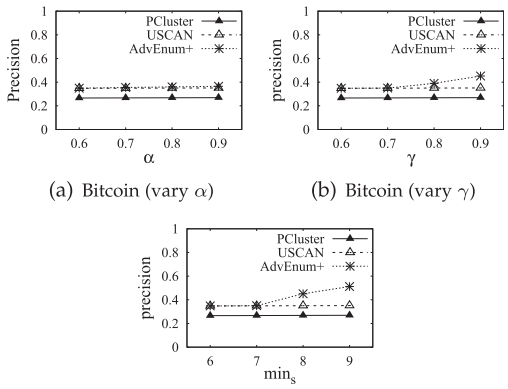
Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

TABLE 3
Precision of Different Algorithms

Algorithms	Results	TP	FP	PR
USCAN	456	1,086	2,037	0.348
PCluster	475	1,027	3,021	0.266
AdvEnum+	105	2,151	6,051	0.355

parameters of AdvEnum are also set to default values (i.e., $\alpha = 0.7$, $\gamma = 0.8$ and min_s = 8). Table 3 shows the results of different algorithms. As can be seen, our approaches to find maximal (α, γ)-quasi-cliques performs better than the other two baseline algorithms in terms of TP, FP and PR. For example, the precision of AdvEnum+ is 0.355, while the precision of USCAN and PCluster is 0.348 and 0.266. The reason is that each complex may be a small cohesive subgraph, which can be well characterized by a maximal (α, γ)-quasi-clique. Also both USCAN and PCluster are clustering-based algorithms which may generate large-size clusters, thus their precision is smaller than ours.

Precision With Varying Parameters. Here we study how the parameters affect the clustering qualities of our algorithm. Fig. 4 shows the precision of our algorithm with varying parameters on the CORE dataset. As we can see from the Fig. 4a, the precision of our algorithm is relatively robust with varying α. This is because the probabilities of the edges in CORE are very large, thereby the maximal (α, γ)-quasi-cliques that we are looking for have high probabilities. Thus, the maximal (α, γ)-quasi-cliques cannot be significantly affected by the parameter α. From Figs. 4b and 4c, we can see that the precision of our algorithm increases with γ (or min_s) increasing. This is because with a larger γ or min_s, we may prune more nodes, and therefore the maximal (α, γ)-quasi-cliques will be highly reliable. The results shown in Fig. 4 confirm that our approach to find maximal (α, γ)-quasi-cliques has a good performance for protein complexes detection.



(a) Bitcoin (vary α) (b) Bitcoin (vary γ)

(c) Bitcoin (vary min_s)

Fig. 4. Precision of different algorithms with varying parameters.

6.3 Efficiency testing

In this section, we evaluate the efficiency of our algorithms by considering running time, pruning effect, the effect of probability distribution, and scalability. Combining the running time of the four approaches shown in Fig. 5 on the data sets CORE and Bitcoin, we can find that SeBaseline and Baseline have the same performance. Neither of them can get results. In addition, the last two approaches AdvEnum and AdvEnum+, both calculate and update the probability during the operation of the algorithms, and Baseline is also performed in this order. So we do not consider SeBaseline in the experimental evaluation.

Runtime of Baseline, AdvEnum and AdvEnum+. We evaluate the runtime of Baseline, AdvEnum and AdvEnum+ for enumerating all the maximal (α, γ)-quasi-cliques in our experiments. Note that we choose the parameter min_s from the interval $[4, 28]$ with a default value 26 for AskUbuntu and Hyves since these two datasets are too dense to deal with a small min_s. Fig. 5 shows the runtime of these three algorithms on all datasets with varying values for α, γ and min_s respectively. As can be seen, AdvEnum is significantly faster than Baseline, and AdvEnum+ is consistently faster than AdvEnum with all parameters. These results confirm that our pruning techniques and probability update method are effective in enumerating maximal (α, γ)-quasi-cliques on uncertain graphs. In general, the runtime of AdvEnum and AdvEnum+ decrease as α, γ or $\min_s$ increases. This is because when these three parameters change, the AdvEnum and AdvEnum+ algorithms can remove much more nodes from the uncertain graph or prune the search space earlier during the enumeration procedure. However, the runtime of AdvEnum and AdvEnum+ in CORE show different trends. We can see that the running time of AdvEnum and AdvEnum+ increases as α increasing. This is because the probability associated with the edges in CORE is very high. In this scenario, during the enumeration procedure, some (α, γ)-quasi-cliques contains a large portion of nodes in CORE. Thus, the enumeration procedure can terminate early. This is why the runtime of AdvEnum and AdvEnum+ increase as α increases. As shown in Fig. 5, we can see that the execution time gap between AdvEnum and AdvEnum+ becomes smaller as these three parameters increase. This is because as these parameters increase, the number of remaining nodes after pruning increases, which suggests that a fewer nodes can be removed with our optimization techniques during the enumeration procedure. As a result, the running time of AdvEnum and AdvEnum+ are gradually approaching as these parameters increase.

Pruning Effect of Baseline and AdvEnum. In this section, we evaluate the number of remaining nodes after we prune nodes with the pruning methods in Baseline and AdvEnum respectively. Fig. 5 shows the number of remaining nodes of NaiveEunm and AdvEnum on Amazon with varying values of α, γ and min_s respectively. As shown in Figs. 6b and 6c, we can see that when $\gamma \geq 0.8$ or $min_s \geq 8$, quite a few nodes can be removed after we prune nodes using the pruning methods in Baseline and AdvEnum respectively. This is why the runtime of three algorithms has a cliff-like fall in Fig. 5d, 5i, and 5n. Additionally, we can also observe that the pruning performance of AdvEnum is much better than Baseline on Amazon, which is consistent with our previous results.

Effect of Different Probability Distributions. Here we study the performance of our algorithms with different probability distributions. As introduced in the previous experiments, we generate the probability on each edge by an exponential

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

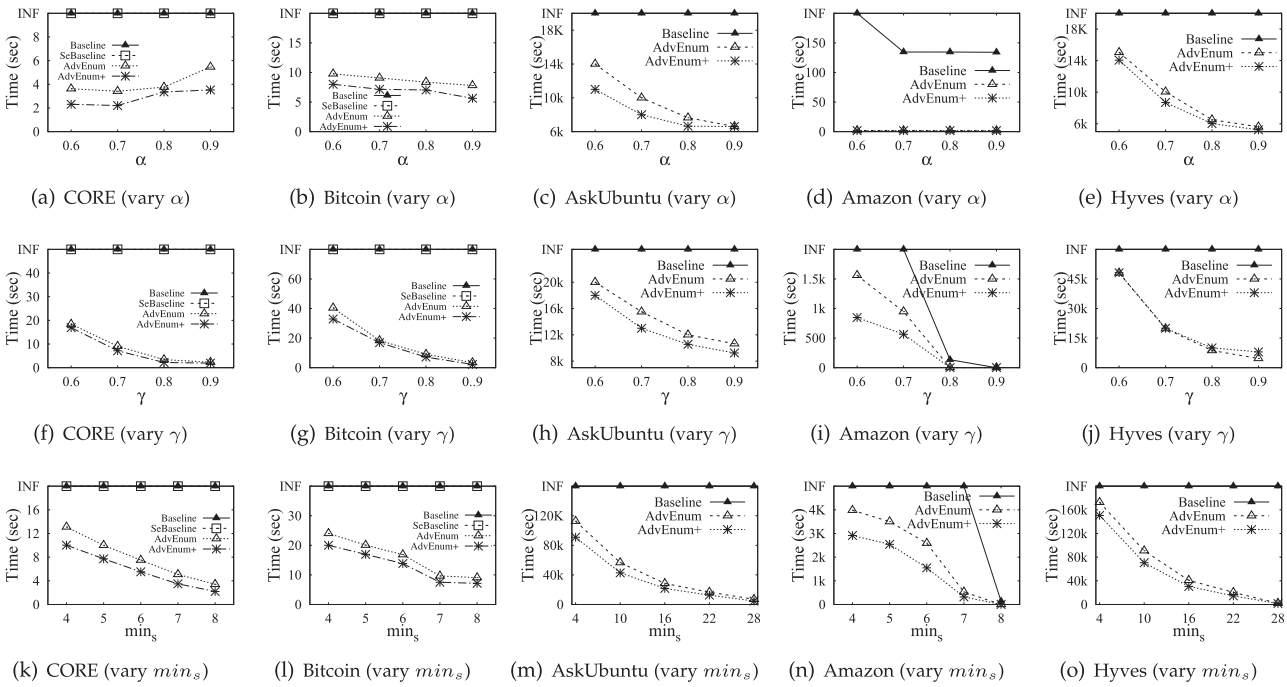


Fig. 5. Runtime of different algorithms for mining all maximal (α, γ)-quasi-cliques.

distribution with a parameter λ. We first investigate the impact of parameter λ for Baseline, AdvEnum and AdvEnum + (vary λ from 2 to 5), respectively. Second, we generate the edge probabilities for Bitcoin with a uniform $[0, 1]$ distribution and evaluate the performance of the three algorithms on this dataset. In Fig. 7a, we can find that the number of remaining nodes obtained by AdvEnum decreases as λ increases. This is because the probability of edge decreases as λ increases, thus more nodes can be removed with the pruning methods in AdvEnum. In Fig. 7b, we are able to observe that the runtime of AdvEnum (or AdvEnum+) decreases with an increasing λ, due to the probabilities of edges reducing. In addition, Fig. 7b also shows that the runtime gap is getting larger as λ increases. This is because the number of nodes that are removed during the enumeration procedure increases with λ increasing.

Scalability Testings. We use the Bitcoin and Hyves datasets to evaluate the scalability of all the algorithms. We generate several subgraphs by randomly sampling 20-80 percent of the nodes (or edges) from these two datasets and evaluate the time overheads of our algorithms on these subgraphs. We set the parameters for all algorithms with the default values. As shown in Fig. 8, we can see that the running time of AdvEnum and AdvEnum+ increase smoothly with respect to $|V|$ and $|E|$, which indicates that our algorithms are scalable when handling real-world graphs.

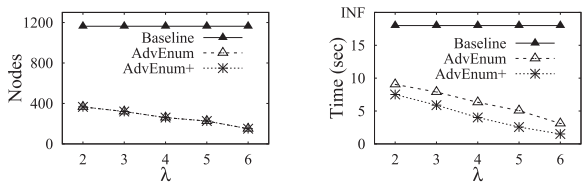


Fig. 7. Effect of different probability distribution (Bitcoin).

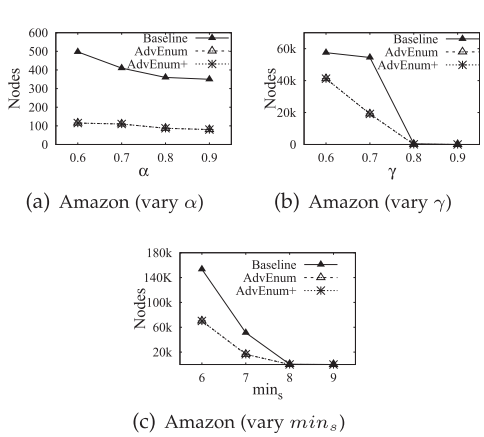


Fig. 6. Number of remaining nodes.

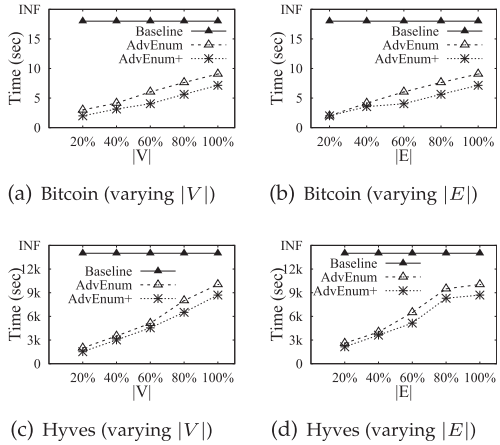


Fig. 8. Scalability of various algorithms.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

7 Related Work

Uncertain Graph Mining. Mining uncertain graphs has attracted much attention in the database and data mining communities [12], [13], [15], [16], [24], [25], [26], [27], [28], [29]. Zou *et al.* [27] proposed an approximate algorithm to mine frequent subgraphs from an uncertain graph database. Yuan *et al.* [28] proposed an efficient subgraph search method on large uncertain graphs. Lin *et al.* [25] proposed an algorithm to find reliable clusters in an uncertain graph. Bonchi *et al.* [12] studied the k-core decomposition problem on an uncertain graph. Huang *et al.* [13] studied the k-truss mining problem on an uncertain graph by proposing a new concept (k, γ)-truss. Gao *et al.* [26] proposed a new solution to find RkNN in an uncertain graph. Qiu *et al.* [19] studied the problem of graph structural clustering on an uncertain graph. Li *et al.* [16] study the maximal clique search problem on an uncertain graph. However, the problem of mining quasi-cliques from an uncertain graph has not been studied previously.

γ-Quasi-Clique Mining. γ-quasi-clique mining is an interesting problem in the field of graph mining. As we know, the first study of the quasi-clique mining problem on a deterministic graph is conducted by Matsuda *et al.*[30] who introduce a subgraph structure called p-quasi complete graph, which is the same as the current definition of γ-quasi-clique in a deterministic graph. They proposed an approximation algorithm to get all the nodes with a minimum number of p-quasi complete graphs. Abello *et al.* defined a γ-clique in a graph which is a connected subgraph with edge density no less than γ in [31]. An approximation algorithm was proposed by them to find all the γ-cliques. All the above studies mine quasi-cliques from a single graph. There are some studies that mine subgraph patterns from a graph database which includes a set of graphs. Yan *et al.* [32] investigated the problem of mining frequent graph patterns with connectivity constraints from a graph database. Different from the above work, some papers mine the quasi-cliques from multiple graphs. Pei *et al.* [8] proposed an algorithm called Crochet which exploits several interesting and effective techniques to efficiently mine cross-graph quasi-cliques. Wang *et al.* [33] investigated the frequent closed clique mining problem from a graph database. They developed an algorithm called CLAN to compute all the frequent closed cliques. Since cliques have the downward closure property, therefore mining cliques is much easier than mining quasi-cliques. Zeng *et al.* [34] investigated the frequent closed quasi-clique mining problem from graph databases. They proposed an efficient algorithm called Cocain to solve the problem with some interesting pruning techniques. Liu *et al.* [9] proposed an algorithm called *Quick* with several interesting pruning methods to find maximal quasi-cliques from undirected graphs. They also proposed several effective pruning techniques to reduce the search space. Although these algorithms are very efficient in practical, they are only work on deterministic graphs, and they cannot be directly used for uncertain graphs.

8 Conclusion

In this paper, we study the problem of mining maximal (α, γ)-quasi-cliques from an uncertain graph. We propose a basic enumeration approach to find all the maximal (α, γ)-quasi-cliques, and a more efficient algorithm with early termination and candidate set reduction pruning techniques. We also propose a dynamic programming framework to update the probability, as well as several optimization techniques to further improve the efficiency. Extensive experiments on large real networks demonstrate the effectiveness and efficiency of our solutions. However, the performance of our algorithm is not good enough in the dense uncertain graphs. Our future works are as follows: (1) Enumerating the top-k (α, γ)-quasi-cliques on uncertain graphsc combining the scoring mechanism; (2) Mining the subgraph with another dense subgraph model on uncertain graphs, such as k-plex.

Acknowledgments

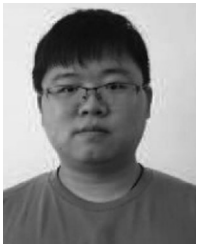
This work was supported in part by the NSFC under Grants 61772346, 61732003, U1809206, 61932004, and 61702435 and in part by the Fundamental Research Funds for the Central Universities under Grant N181605012.

References

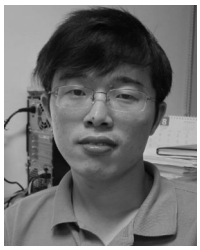
[1] V. Batagelj and M. Zaversnik, "An o(m) algorithm for cores decomposition of networks," 2003, *arXiv:cs/0310049*.
[2] R.-H. Li, J. X. Yu, and R. Mao, "Efficient core maintenance in large dynamic graphs," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 10, pp. 2453–2465, Oct. 2014.
[3] X. Huang, H. Cheng, L. Qin, W. Tian, and J. X. Yu, "Querying k-truss community in large and dynamic graphs," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 1311–1322.
[4] L. Qin, R.-H. Li, L. Chang, and C. Zhang, "Locally densest subgraph discovery," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 965–974.
[5] R.-H. Li, L. Qin, J. X. Yu, and R. Mao, "Influential community search in large networks," *Proc. VLDB Endowment*, vol. 8, no. 5, pp. 509–520, 2015.
[6] C. Lu, J. X. Yu, H. Wei, and Y. Zhang, "Finding the maximum clique in massive graphs," *Proc. VLDB Endowment*, vol. 10, no. 11, pp. 1538–1549, 2017.
[7] R.-H. Li *et al.*, "Skyline community search in multi-valued networks," in *Proc. Int. Conf. Manage. Data*, 2018, pp. 457–472.
[8] J. Pei, D. Jiang, and A. Zhang, "On mining cross-graph quasi-cliques," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2005, pp. 228–238.
[9] G. Liu and L. Wong, "Effective pruning techniques for mining quasi-cliques," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, 2008, pp. 33–49.
[10] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques," in *Proc. Int. Comput. Combinatorics Conf.*, 2004, pp. 161–170.
[11] J. Cheng, L. Zhu, Y. Ke, and S. Chu, "Fast algorithms for maximal clique enumeration with limited memory," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2012, pp. 1240–1248.
[12] F. Bonchi, F. Gullo, A. Kaltenbrunner, and Y. Volkovich, "Core decomposition of uncertain graphs," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 1316–1325.
[13] X. Huang, W. Lu, and L. V. S. Lakshmanan, "Truss decomposition of probabilistic graphs: Semantics and algorithms," in *Proc. Int. Conf. Manage. Data*, 2016, pp. 77–90.
[14] A. P. Mukherjee, P. Xu, and S. Tirthapura, "Mining maximal cliques from an uncertain graph," in *Proc. IEEE 31st Int. Conf. Data Eng.*, 2015, pp. 243–254.
[15] Z. Zou, J. Li, H. Gao, and S. Zhang, "Finding top-k maximal cliques in an uncertain graph," in *Proc. IEEE 26th Int. Conf. Data Eng.*, 2010, pp. 649–652.
[16] 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 *Proc. IEEE 35th Int. Conf. Data Eng.*, 2019, pp. 1178–1189.
[17] S. V. Sanei-Mehri, A. Das, and S. Tirthapura, "Enumerating top-k quasi-cliques," in *Proc. IEEE Int. Conf. Big Data*, 2018, pp. 1107–1112.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.

[18] N. N. Dalvi and D. Suciu, "Efficient query evaluation on probabilistic databases," in *Proc. 13th Int. Conf. Very Large Data Bases*, 2004, pp. 864–875.
[19] Y.-X. Qiu *et al.*, "Efficient structural clustering on probabilistic graphs," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 10, pp. 1954–1968, Oct. 2019.
[20] N. J. Krogan *et al.*, "Global landscape of protein complexes in the yeast saccharomyces cerevisiae," *Nature*, vol. 440, no. 7084, pp. 637–643, 2006.
[21] M. Potamias, F. Bonchi, A. Gionis, and G. Kollios, "k-nearest neighbors in uncertain graphs," *Proc. VLDB Endowment*, vol. 3, no. 1–2, pp. 997–1008, 2010.
[22] R. Jin, L. Liu, B. Ding, and H. Wang, "Distance-constraint reachability computation in uncertain graphs," *Proc. VLDB Endowment*, vol. 4, no. 9, pp. 551–562, 2011.
[23] G. Kollios, M. Potamias, and E. Terzi, "Clustering large probabilistic graphs," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 2, pp. 325–336, Feb. 2013.
[24] R. Jin, L. Liu, and C. C. Aggarwal, "Discovering highly reliable subgraphs in uncertain graphs," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2011, pp. 992–1000.
[25] L. Lin, R. Jin, C. Aggarwal, and Y. Shen, "Reliable clustering on uncertain graphs," in *Proc. IEEE 12th Int. Conf. Data Mining*, 2012, pp. 459–468.
[26] Y. Gao, X. Miao, G. Chen, B. Zheng, D. Cai, and H. Cui, "On efficiently finding reverse k-nearest neighbors over uncertain graphs," *VLDB J.*, vol. 26, no. 4, pp. 467–492, 2017.
[27] Z. Zou, H. Gao, and J. Li, "Discovering frequent subgraphs over uncertain graph databases under probabilistic semantics," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2010, pp. 633–642.
[28] Y. Yuan, G. Wang, H. Wang, and L. Chen, "Efficient subgraph search over large uncertain graphs," *Proc. VLDB Endowment*, vol. 4, no. 11, pp. 876–886, 2011.
[29] Y. Yuan, G. Wang, L. Chen, and H. Wang, "Efficient subgraph similarity search on large probabilistic graph databases," *Proc. VLDB Endowment*, vol. 5, no. 9, pp. 800–811, 2012.
[30] H. Matsuda, T. Ishihara, and A. Hashimoto, "Classifying molecular sequences using a linkage graph with their pairwise similarities," *Theor. Comput. Sci.*, vol. 210, no. 2, pp. 305–325, 1999.
[31] J. Abello, M. G. C. Resende, and S. Sudarsky, "Massive quasi-clique detection," in *Proc. Latin Amer. Symp. Theor. Inform.*, 2002, pp. 598–612.
[32] X. Yan, X. J. Zhou, and J. Han, "Mining closed relational graphs with connectivity constraints," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2005, pp. 324–333.
[33] J. Wang, Z. Zeng, and L. Zhou, "CLAN: An algorithm for mining closed cliques from large dense graph databases," in *Proc. 22nd Int. Conf. Data Eng.*, 2006, p. 73.
[34] Z. Zeng, J. Wang, L. Zhou, and G. Karypis, "Coherent closed quasi-clique discovery from large dense graph databases," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2006, pp. 797–802.



Lianpeng Qiao received the BS and ME degrees, in 2014 and 2017, respectively, in computer science from Northeastern University, China, where he is currently working toward the PhD degree. His research interests include social network analysis and data-driven graph mining.

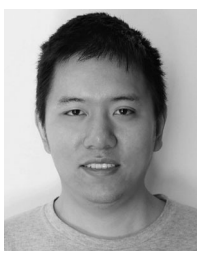


Rong-Hua Li received the PhD degree from the Chinese University of Hong Kong in 2013. He is currently an associate professor with the Beijing Institute of Technology, Beijing, China. His research interests include graph data management and mining, social network analysis, graph computation systems, and graph-based machine learning.

Zhiwei Zhang received the PhD degree from the Chinese University of Hong Kong in 2014. He is currently a professor with the Beijing Institute of Technology, Beijing, China. His research interests include blockchain, social network analysis, distributed systems, and graph-based machine learning.

Ye Yuan received the BS, MS, and PhD degrees in computer science from Northeastern University, in 2004, 2007, and 2011, respectively. He is currently a professor with the Department of Computer Science, Beijing Institute of Technology, Beijing, China. His research interests include graph databases, probabilistic databases, and social network analysis.

Guoren Wang received the BSc, MSc, and PhD degrees from the Department of Computer Science, Northeastern University, China, in 1988, 1991, and 1996, respectively. He is currently a professor with the Department of Computer Science, Beijing Institute of Technology, Beijing, China. He has authored or coauthored more than 100 research papers. His research interests include query processing and optimization, bioinformatics, high dimensional indexing, parallel database systems, and cloud data management.



Hongchao Qin received the BS degree in mathematics and ME degree in computer science, in 2013 and 2015, respectively, from Northeastern University, China, where he is currently working toward the PhD degree. His research interests include social network analysis and data-driven graph mining.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on January 23,2024 at 02:56:09 UTC from IEEE Xplore. Restrictions apply.