Data Science and Engineering (2024) 9:220–237
https://doi.org/10.1007/s41019-024-00244-z

RESEARCH PAPER

# Leveraging Semantic Information for Enhanced Community Search in Heterogeneous Graphs

Yuqi Li[1] · Guosheng Zang[1] · Chunyao Song[1] · Xiaojie Yuan[1] · Tingjian Ge[2]

Received: 17 October 2023 / Revised: 27 December 2023 / Accepted: 28 January 2024 / Published online: 28 February 2024

**Abstract**
Community search (CS) is a vital research area in network science that focuses on discovering personalized communities for query vertices from graphs. However, existing CS methods mainly concentrate on homogeneous or simple attributed graphs, often disregarding complex semantic information and rich contents carried by entities in heterogeneous graphs (HGs). In this paper, we propose a novel problem, namely the "Semantic Network Oriented Community Search with Meta-Structures in Heterogeneous Graphs (SNCS)," which aims to find dense communities that contain the query vertex, with vertices of the same type sharing similar topics. In response to this new problem, we present a novel approach, also named SNCS, representing the first solution employing meta-structures and topic constraints to tackle community search, leveraging both topological and latent features. To overcome the high-time complexity challenge posed by searching through meta-structures, we introduce a unique graph reconstruction technique. Our proposed method's superiority is validated through extensive evaluations on real-world datasets. The results demonstrate a significant improvement in the quality of the obtained communities, with increases of 3.5–4.4% in clustering coefficient and 5–11% in density while requiring only 4–46% of the running time when compared with the state-of-the-art methods.

**Keywords** Heterogeneous graphs · Community search · Meta-structures · Graph reconstruction

## 1 Introduction

With the explosive growth of data generated from everyday applications, graphs have emerged as an effective model for describing real-world data, in such an environment, research on graphs has developed rapidly, especially heterogeneous graphs [1, 2], including bibliographic networks, knowledge graphs, shopping networks and others. For instance, an Amazon dataset [3] was used to extract a movie review network, showcasing relationships between various entities, such as moviegoers, movies, categories, and tags (see Fig. 1). The toy network includes six moviegoers (i.e., $u_1, u_2, \dots, u_6$), six movies (i.e., $m_1, m_2, \dots, m_6$), one category ($c$), and two tags ($t_1, t_2$) represented as vertices. Semantic relationships between different entities are shown by undirected edges. Moreover, each movie node includes text depicting moviegoers' comments. Similarly, other heterogeneous graph (HG) entities often contain additional descriptive text as well, as demonstrated in this movie network.

Community search has recently gained considerable attention [4–8] due to its deployment in real-world applications, such as temporary grouping, event organization, friend and advertisement recommendations. The primary goal of community search is to obtain a dense subgraph that satisfies the user's requirements quickly [9, 10]. To the best of our knowledge, current studies mainly concentrate on homogeneous and attributed graphs. Community search on HGs is an emerging problem, with only limited studies focusing solely on capturing the heterogeneous information via meta-paths [11–14]. However, to better align with the user's intent in a semantically rich world, it is necessary to consider not only the structural correlations among entities, but also integrates the unstructured textual information of the entities. For instance, in Fig. 1, if reviewer $u_1$ wishes to find a group with similar taste in movies, $u_1$ might anticipate that the group's reviewers are not only connected but also share high content similarity. Fortunately, meta-structures

✉ Chunyao Song
chunyao.song@nankai.edu.cn

1 College of Computer Science, DISSec, NDST, TMCC, Nankai University, Tianjin, China

2 Department of Computer Science, University of Massachusetts Lowell, Lowell, USA

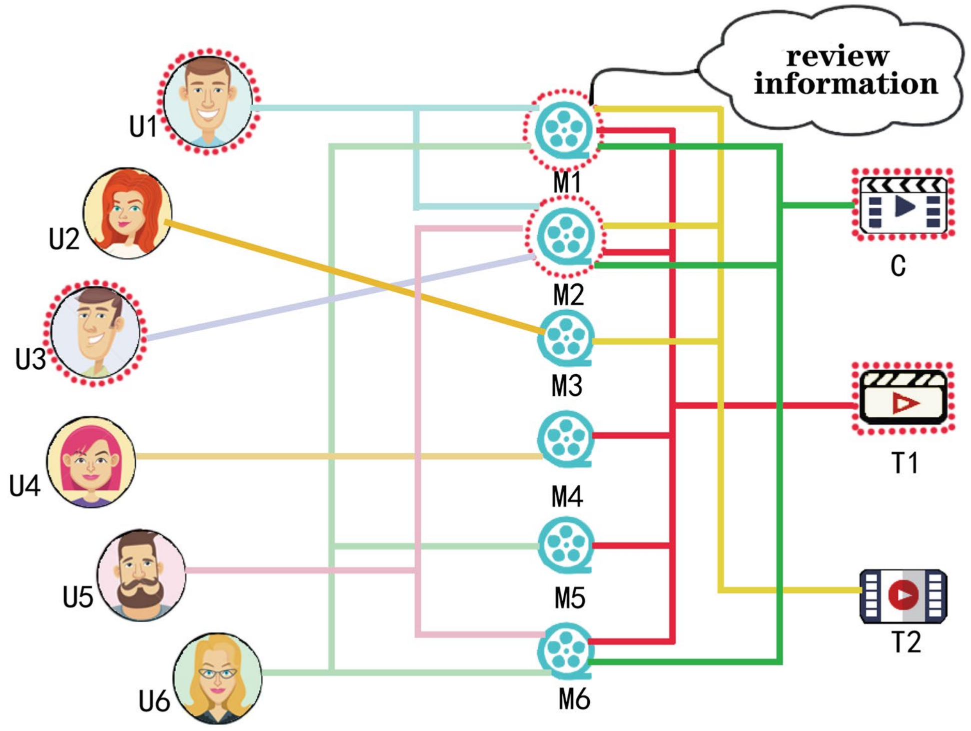


**Fig. 1** An example of heterogeneous graphs

have been shown to be more effective than meta-paths in computing entity similarities [15]. However, searching through meta-structures is highly computationally expensive. To obtain a community with higher quality and lower computational complexity, we propose SNCS, a semantic network oriented community search by meta-structures in heterogeneous graphs with the graph reconstruction algorithm. In this paper, we present a detailed approach that provides personalized communities for query vertices. Here is a realistic application scenario of finding potential members for a movie club.

***Example 1*** **Movie club members.** Fig. 1 depicts a movie review network with different vertex types and edge relationships. Suppose a reviewer $u_1$ aims to form a movie club with like-minded individuals. Using the meta-path $P = U \rightarrow M \rightarrow C \rightarrow M \rightarrow U$ results in a community of $\{u_1, u_3, u_5, u_6\}$. However, this community is not tailored to the personalized outcomes of reviewer $u_1$, to learn the movie reviewers' preferences; we utilize the latent topic model [16] to extract their topic information. Subsequently, we search for the movie reviewers based on a pre-specified meta-structure ($MS = U \rightarrow M \rightarrow C\&\&T \rightarrow M \rightarrow U$). As illustrated in Fig. 1, it depicts an instance of the meta-structure marked by dotted contour lines. The meta-structure pattern $u_1 \rightarrow m_1 \rightarrow c\&\&t_1 \rightarrow m_2 \rightarrow u_3$ is an example, where we can add $u_3$, along with other potential reviewers, to the candidate reviewer list that complies with this meta-structure pattern. To ensure the community's cohesion and exclude irrelevant entities (e.g., $u_3$), every reviewer within the same community must connect to at least $k$ other edge-disjoint meta-structure-based neighbors (detailed in Sect. 3). Taking $u_1$ as the query vertex, the SNCS algorithm results in the community composed of $u_1$, $u_5$, and $u_6$, where each vertex has at least two meta-structure-based edge-disjoint neighbors, with the exclusion of $u_3$. SNCS exhibits the following notable advantages compared to meta-path methods: (a) SNCS can differentiate movie reviewers' preferences by utilizing topic models to consider the content information of movies. (b) SNCS can reveal the relevance of movie reviewers and identify potential 'spurious movie lovers.' For instance, $u_3$ has watched and reviewed only one movie ($m_2$), and it should not be included in this community, but meta-path includes it. (c) SNCS can further constrain the category of movies, allowing for the identification of films belonging to multiple categories simultaneously (e.g., $(m_1, m_2)$), whereas meta-paths can only consider movies belonging to a single category. Additionally, in the end of Sect. 3, we have provided a detailed comparison between Meta-Paths and Meta-Structures.

In this paper, we propose a novel approach to the problem of community search, namely SNCS. Specifically, our contributions are summarized as follows:

- We present SNCS, a novel community search approach that utilizes topic models to account for content preferences, which has not been explored before (Sect. 4).

- Our study is the first to utilize meta-structures to improve the quality of community search in heterogeneous graphs. Additionally, we develop an efficient graph reconstruction algorithm to tackle the high time complexity associated with searching through meta-structures. The algorithm is based on cutting-edge community search techniques and is proved to be effective in our analysis (Sect. 4).
- We conduct comprehensive experiments on real HGs to demonstrate the efficiency and effectiveness of our proposed SNCS approach. We also perform an in-depth comparison and analysis of the experimental parameters. Finally, to illustrate a practical application of the proposed method, we provide a case study on a real dataset (Sect. 5).

## 2 Related Work

With the initial introduction of community search [17], there has been a proliferation of variants in recent years. These variants are applied to homogeneous graphs, attribute graphs, heterogeneous graphs, and dynamic graphs, but their core focus remains on discovering tightly connected communities that satisfy specific constraints. Community detection and community search represent two main categories of studies related to finding communities on graphs [18–20].

**Community detection** algorithms have been studied extensively on both homogeneous and heterogeneous graphs [21–24]. The number of communities retrieved by community detection algorithms is typically uncertain, but vertices within the same community are closely related [25]. The algorithms implemented on heterogeneous graphs typically utilize meta-paths to leverage relation information, which can be categorized into four types: graph neural networks [26, 27], graph division [28], module optimization [29], and label transmission [30, 31]-based methods. Regrettably, community detection algorithms operate on the entire graph, which hinders their ability to search for communities specific to a given set of vertices, thus posing challenges in the pursuit of personalized community identification.

**Community search** [4] offers highly personalized queries for communities based on the given query vertex $q$. Numerous community search variations have been proposed since its inception [32–34]. One can specify one or more vertices to search for the designed communities that contain the query vertices [17]. Most methods typically return only one community, while some yield several different communities [35]. One of the most significant advantages of community search is its high personalization [36], efficiency [5], and robust support for dynamic graphs [4]. Community search studies mainly focus on homogeneous graphs and attributed graphs [18]. The study of homogeneous graphs focuses on topology, finding communities that satisfy $k$-clique [37], $k$-core [17], or $k$-truss [38] constraints, and contain the query vertices. The study on attributed graph [10, 39] focuses on both topology and vertices' properties to find a community that is closer in attributes. Research on community search in heterogeneous graphs is limited, which are meta-path-based methods. For instance, Batch-Ecore [11] utilizes meta-paths to extract topologically dense communities in heterogeneous networks; ISCH [40] defines an influential community by considering the significance of vertices, and SACS-HIN [41] defines new spatially aware communities by utilizing the geographic location of vertices. Some topic-based community search methods use vertices' attributes to personalize the resulting communities. The heterogeneous graph oriented method $k\mathcal{KP}$-core [13] consider the vertex keywords in the search process, proposing dense connection communities based on keywords. However, none of these methods fully exploit the rich information available in the graphs, especially in the complex semantic networks, as the information extracted from meta-path-based methods is limited. On the other hand, the use of keywords in existing methods during community search is limited in terms of flexibility, as they require manual selection of an appropriate set of keywords to identify appropriate communities. In the context of community searches on heterogeneous graphs [20, 33, 40, 41], the "community" to be searched for is typically tailored to address specific problem definitions and lacks generality. Certain NLP topic models [42, 43] can derive an m-dimensional vector $z$ for each user, representing the distribution of topics for that user, such as non-negative Matrix Factorization (NMF) [44], Latent Dirichlet Allocation (LDA) [45], Neural LDA [46], Embedded Topic Models (ETM) [47], among others. In order to flexibly identify topic-related communities, it is essential to consider the use of topic models to incorporate content information during the search.

## 3 Preliminaries

Table 1 presents a summary of the symbols used in this paper. In the following subsections, we will introduce the Heterogeneous Graphs (HGs) model and compare the concepts of meta-paths and meta-structures as described in the literature.

**Definition 1** (*Community Search* [17]) Since its inception, Community Search has undergone various variants, but its definition can be generalized as follows: Given an undirected (connected) graph $G = (V, E)$ and a set of query nodes $Q \in V$, subject to constraints defined by the problem, we aim to identify an induced subgraph $H = (V_H, E_H)$ of $G$ that satisfies at least the following conditions: (a) $H$ contains $Q$;

**Table 1** Symbols used in this paper

| Symbol | Description |
|---|---|
| $G(V, E)$ | A Heterogeneous Graph with vertex set V and edge set E |
| $T_G = (\mathcal{L}, \mathcal{R})$ | The Heterogeneous Graph schema |
| $\mathcal{L}, \mathcal{R}$ | The vertex and edge type set |
| $\mathcal{P}, \mathcal{S}$ | The meta-path and meta-structure |
| $p, s$ | The instance of meta-path and meta-structure |
| $\phi(v)(\psi(e))$ | The vertex(edge) type of $v(e)$ |
| $\theta$ | The threshold for topic similarity |
| $u.t$ | The topic vector of node $u$ |
| $Nbr(u)$ | The neighbors directly connected to vertex $u$ |
| $\beta(v, C)$ | The edge-disjoint meta-structure instances which Start from vertex $v$ and end at vertex in set $\mathcal{C}$ |
| $Sim(u, v)$ | The topic similarity between $u$ and $v$ |
| $\mathcal{S}_{nbr}(v)$ | The meta-structure-based neighbor of $v$ |
| $E_k(\mathcal{S})$ | An edge-disjoint $(k, \mathcal{S})$-core |

(b) $H$ is connected; (c) it adheres to the constraints defined by the problem.

**Definition 2** (*Heterogeneous Graph (HG)*) [48, 49], also called a Heterogeneous Information Network, is a graph data structure represented by $G = (V, E)$ that contains multiple types of vertices and edges. Additionally, it has type mapping functions that include a vertex type mapping function $\phi : V \rightarrow \mathcal{L}$ and a link type mapping function $\psi : E \rightarrow \mathcal{R}$, satisfying $|\mathcal{L}| + |\mathcal{R}| > 2$. Each object $v$ belongs to an object type $\phi(v) \in \mathcal{L}$, and each link $e$ belongs to a relation type $\psi(e) \in \mathcal{R}$. Moreover, certain vertices in this model can hold descriptive text information. Heterogeneous graph is a prevalent data type that can be found in various scenarios, such as in knowledge graphs and social networks [50].

**Definition 3** (*Heterogeneous Graph Schema* [48, 49]) is a directed graph $T_G = (\mathcal{L}, \mathcal{R})$ which is based on an Heterogeneous Graph $G = (V, E)$. The vertex types and edge types of $G$ correspond to the vertices and edges of $T_G$, respectively. This schema serves to reveal semantic information between different types of vertices.

**Definition 4** (*Meta-Path* [51]) A meta-path, denoted by $\mathcal{P}$, is a path defined on the Heterogeneous Graph schema $T_G$. The vertices and edges of the meta-path correspond to the vertices and edges on the HG schema, which represent the relationships from one vertex type to another. For every meta-path, there must exist an opposite path on the HG schema, which we denote by $P^{-1}$.

**Definition 5** (*Meta-Structure* [15]) A meta-structure, denoted by $\mathcal{S}$, is a directed acyclic graph (DAG) with a single source node $n_s$ and a single sink (target) node $n_t$, defined on the HG schema $T_G = (\mathcal{L}, \mathcal{R})$. The reverse meta-structure of $\mathcal{S}$ in HG, with all edges of $\mathcal{S}$ reversed, is denoted by $\mathcal{S}^{-1}$, representing the relationship from the sink node $n_t$ to the source node $n_s$. Moreover, a meta-structure consists of vertical layers in level order, and we define any vertical layer with the number of vertices greater than one as a bifurcation.

As an example, we can refer to the schema of HG displayed in Fig. 2a. Figure 2c, d shows instances of the corresponding meta-path and meta-structure presented in Fig. 2b, respectively. Particularly, in Fig. 2d, the vertical layer containing $c$ and $t_2$ is a bifurcation, which is enclosed by a dashed line.

**Comparison between Meta-Paths and Meta-Structures** Meta-paths have been widely used in the analysis of Heterogeneous Graphs [52, 53]. However, there are limitations in their ability to describe complex patterns. For instance, consider the complex relationship ($s = u_1 \rightarrow m_1 \rightarrow c\&\&t_1 \rightarrow m_2 \rightarrow u_3$) between film reviewers $u_1$ and $u_3$, as shown in Fig. 2d. This pattern can be captured accurately only by a meta-structure and not by a meta-path. One possible solution is to decompose $s$ into two meta-paths [52, 53], $p_1$ and $p_2$. However, this approach has two drawbacks: (a) Certain vertices are shared by two or more edges. In this example, $e(u_1, m_1)$ and $e(m_2, u_3)$ occur only once under the meta-structure, but they will be included multiple times in the meta-path, leading to inaccurate semantic information. (b) In different instances of the meta-path, certain types of vertices cannot be restricted to a specific vertex. For instance, in Fig. 1, we can connect $u_1$ and $u_6$ using $s = u_1 \rightarrow m_1 \rightarrow c\&\&t_1 \rightarrow m_6 \rightarrow u_6$, where $m$ denotes any movie in the meta-structure. However, when the meta-structure is decomposed into meta-paths ($p_1$ and $p_2$), the corresponding node $m$ may refer to different movies. Thus, $u_1$ and $u_6$ can be connected by either $p_1 = u_1 \rightarrow m_1 \rightarrow t_1 \rightarrow m_6 \rightarrow u_6$ or by $p_2 = u_1 \rightarrow m_2 \rightarrow t_1 \rightarrow m_5 \rightarrow u_6$. Here, $m_5$ only mentions the tag $t_1$ and does not belong to category $c$, thereby weakening the semantic relationship constrained by the meta-structure.

## 4 Problem Definition

This section proposes a new problem called the Semantic Network Oriented Community Search with Meta-Structures in Heterogeneous Graphs (SNCS). It extends the k-core [54] measure of community quality to a meta-structure enabled pattern. We extended the concept of $k$-core, introducing a new definition known as ($k$,$\mathcal{S}$)-core. Subsequently, we

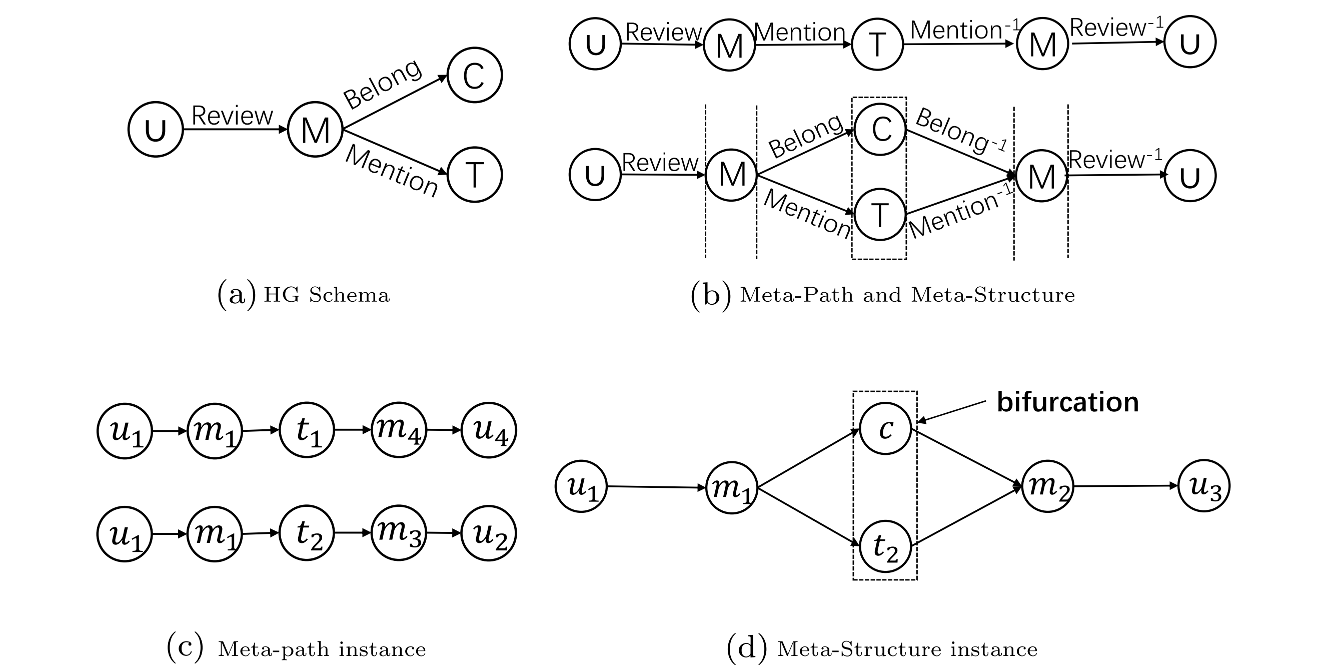

U
Review
M
Belong
C
Mention
T
(a) HG Schema
Review
Mention
Mention-1
Review-1
Belong-1
Mention-1
(b) Meta-Path and Meta-Structure
$u_1$ $m_1$ $t_1$ $m_4$ $u_4$
$u_1$ $m_1$ $t_2$ $m_3$ $u_2$
(c) Meta-path instance
bifurcation
$u_1$ $m_1$ $c$ $t_2$ $m_2$ $u_3$
(d) Meta-Structure instance

**Fig. 2** Schema, Meta-Path, Meta-Structure and instances

explore the conditions allowing for the repetition of edge types and elaborate on the maximal community in community search. Furthermore, based on the meta-structure, we provide a definition for the maximum community. Finally, we present the formal definition of SNCS.

**Definition 6** ($(k, \mathcal{S})$-core) A $k$-core [55] is a maximal connected subgraph within a homogeneous graph, in which each vertex has at least $k$ neighbors. However, since connected vertices in a HG diverse in type, the same $k$-core cannot be applied directly for searching communities in HG. Let $\mathcal{S}$ be a meta-structure and $\mathcal{S}_{nbr}(v)$ be the set of vertices connected to $v$. A $(k, \mathcal{S})$-core, $E_k(\mathcal{S})$, is a set of vertices $v \in HG$ such that the size of $\mathcal{S}_{nbr}(v)$ is no less than $k$. The vertices in $E_k(\mathcal{S})$ are of the same type as the source/target nodes of $\mathcal{S}$.

**Definition 7** (*Maximal community*) Previously, works for community search on HGs based on meta-paths [11], where a maximal community consists of all vertices and their connections, such that each vertex belongs to some community which satisfies the query constraint. We define the maximal community as a set of all vertices and their connections where each vertex belongs to a community that satisfies both the requirement of the $(k,\mathcal{S})$-core and the topic constraints based on the given meta-structure.

Figure 3a shows an example of a simple heterogeneous graph. Given the meta-structure $\mathcal{S} = U \rightarrow M \rightarrow C\&\&T \rightarrow M \rightarrow U$, $\mathcal{S}_{nbr}(u_1)$ contains users $u_2$ and $u_3$. Under the edge-disjoint constraint, edges $e_1(m_2, c)$ and $e_2(m_2, t)$ can be used only once during the search phase, which means that $u_1$ and $u_2$ cannot be connected. However, this is unreasonable since $u_1$ and $u_2$ watched and commented on the same film, belonged to category $c$, and contained tag $t$, indicating a strong association. Therefore, we suggest a partial type edge-disjoint

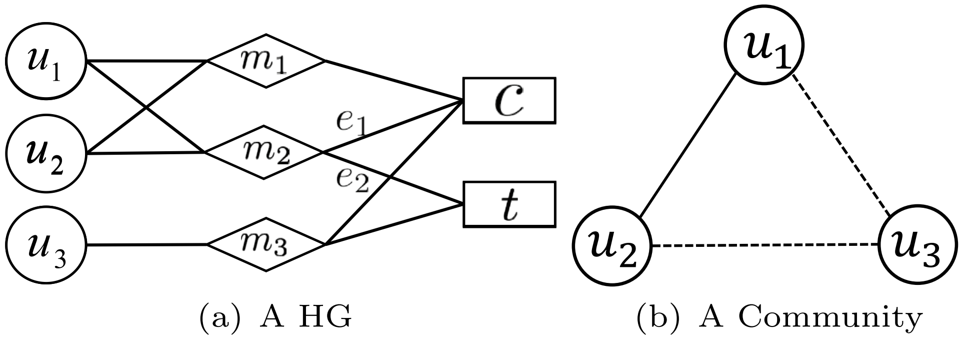


**Fig. 3** An Heterogeneous Graphs (HG) and A community

approach for practical purposes, e.g., edges $\psi(e(m_2, c))$ and $\psi(e(m_2, t))$ can be repeatable during the community search. Figure 3b depicts an example community obtained from Fig. 3a. Vertices $u_1$, $u_2$, and $u_3$ are connected to each other if all edges can be repeated during the search. However, the size of $E_k(s)$ is not two when using $u_3$ as the query vertex and adopting certain types of edge-disjoint constraints (i.e., edges from User type vertices to Movie type vertices are non-repeatable). We aim to find the largest community for $u_3$, not a subset of it. The constraint of non-repeated edges is used to ensure that the $E_k(s)$ calculation is correct. Taking $u_3$ as the query vertex, we can obtain two communities under the same constraint: $u_1$,$u_3$ and $u_2$,$u_3$, as $u_3$ is connected to both $u_1$ and $u_2$ through the meta-structure $\mathcal{S}$. For instance, in Fig. 3b, the solid line between $u_1$ and $u_2$ indicates a connection through the meta-structure, while the dashed lines connecting $u_3$ with $u_1$ and $u_2$ signify that, under the constraint of disallowing edge repetition, $u_3$ cannot simultaneously connect with $u_1$ and $u_2$. Although both communities are significant, we are interested in finding the union between them, i.e., $\{u_1, u_2, u_3\}$. In this community, $u_1$, $u_2$, and $u_3$ are all potentially connected to each other through the meta-structure, but these three vertices cannot be connected simultaneously due to the edge-disjoint constraint. Consequently, the numbers of neighbors of $u_1$ and $u_2$ are both 1. Thus, this community is a $(1, \mathcal{S})$-core.

**Definition 8** (*Semantic Network Oriented Community Search with Meta-Structures in Heterogeneous Graphs (SNCS)*) SNCS is a variant of community search specifically tailored for heterogeneous graphs, given a HG with vertices having possible descriptive textual information, a query vertex $q$, a topic vector, a meta-structure $\mathcal{S}$, a threshold $\theta$ and a parameter $k$ (as in $(k, \mathcal{S})$-core), the SNCS retrieves a maximal set $\mathcal{C}$ as a community, where the vertices in $\mathcal{C}$ should satisfy the following constraints:

- $q \in \mathcal{C}$;
- $\forall v \in \mathcal{C}$, $\phi(v)$=$\phi(q)$;
- $\forall v \in \mathcal{C}$, $Sim(v, q) \geq \theta$, e.g., $(\mathbf{v} \cdot \mathbf{q})/(|\mathbf{v}| \times |\mathbf{q}|) \geq \theta$, where $\mathbf{v}$ and $\mathbf{q}$ represent the feature vectors of vertices $v$ and $q$, respectively;
- $\forall v \in \mathcal{C}$, the size of $\mathcal{S}_{nbr}(v)$ is no less than $k$.

Specifically, the term 'Oriented' emphasizes the integration and utilization of semantic information through meta-structures, enabling a nuanced exploration of community structures in heterogeneous graphs. It focuses on enhancing the representation and discovery of communities by seamlessly integrating semantic information into the community search process.

## 5 Method

This section provides a detailed description of the SNCS, which includes the topic extraction algorithm and the graph reconstruction algorithm. The pipeline for SNCS is illustrated in Fig. 4.

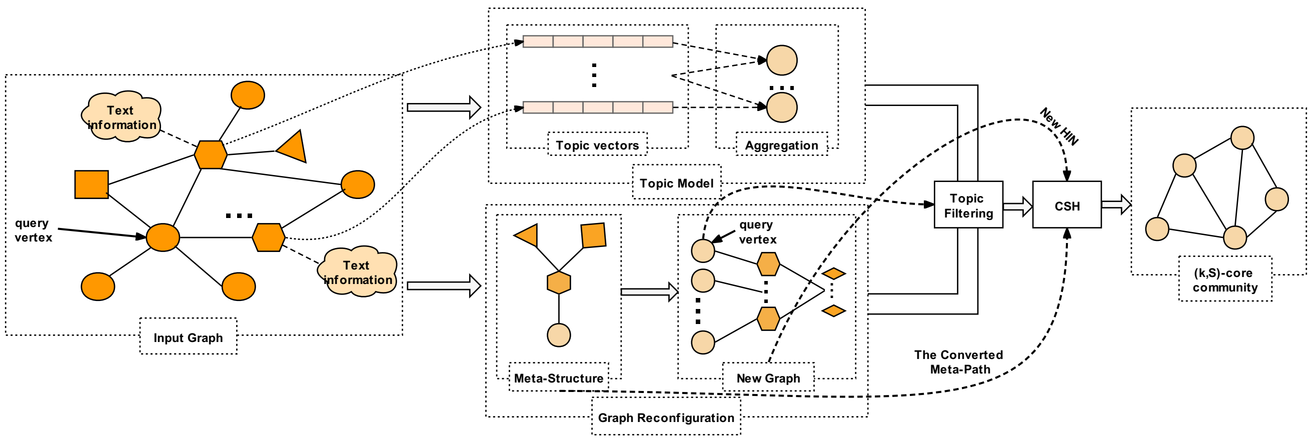


**Fig. 4** The pipeline of SNCS. For the input heterogeneous graph (HG), we extract topics and reconstruct the graph in parallel. We aggregate topic vectors from the neighbors of vertices that have the same type as the query vertex $q$. Following the given meta-structure, we reconstruct a new HG from the input HG and return the meta-paths that can represent this structure on the newly reconstructed graph. Subsequently, we filter the topics to obtain a specific set of vertices, which is further used to search for communities on the newly reconstructed HG by the converted meta-paths (e.g., conducting community search by the meta-path-based BatchEcore [11] algorithm)

### 5.1 Topic Model

A recent paper [16] offers an excellent examination of comparing and optimizing topic models. Based on this research, we adopted the most widely used LDA approach [45] to extract the optimal topic from the messages conveyed by the vertices. LDA, an acronym for Latent Dirichlet Allocation [41], is a typical bag-of-words model, where each document is represented as a set of words without any inherent order. The purpose of LDA is to provide a probabilistic distribution of topics for each document in a collection, enabling the extraction of topics (distributions) from a set of documents through analysis. In our study, we treat the text associated with all vertices as the document collection, with each vertex's associated text considered as an individual document. The OCTIS [16] framework integrates a range of topic models, offering preprocessing methods and evaluation metrics. Additionally, optimal hyperparameters are estimated using a Bayesian Optimization approach. With its assistance, we extract a topic vector for each vertex based on the textual information.

We aggregated the topic to the vertices of the query type in the community search. For example, in Fig. 1, we extracted the topics of each film from its textual reviews and aggregated them to the users. The formula used for this is $u_i.t = Softmax(\sum_{m_i \in Nbr(u_i)} m_i.t)$. To extract the weight of each user on $k$ topics by LDA, we calculated the probability of preference for each topic through the softmax function. For evaluating the topic similarity of two vertices, we use the cosine function: $Sim(u,v) = cos(u,v) = \frac{u \cdot v}{|u| * |v|}$.

### 5.2 Graph Reconstruction Algorithm

Searching for communities through meta-structures is a challenging task, since vertices have many neighbors, and various combinations will appear when the meta-structure has lots of bifurcations. For one vertex, if the meta-structure has $m$ branches, then there would be $\prod_i^m n_i$ combinations, where $n_i$ is the number of neighbors of that vertex of the $i$th type. In large networks where each vertex has numerous neighbors, combining neighbors in the searching process results in a high time complexity. Fortunately, the searching meta-structures are typically static, while query vertices and constraints may change. Therefore, by constructing a new graph from the original heterogeneous graph with the static meta-structure, we can perform community search through meta-path on the new graph. This process is equivalent to searching through meta-structures, a claim that we will prove later. Based on this, we design an efficient algorithm for performing graph reconstruction in batch.

Figure 5a displays a simple heterogeneous graph. If we do a community search through the meta-structure $\mathcal{S} = U \rightarrow M \rightarrow C\&\&T \rightarrow M \rightarrow U$, we can first find all the combinations of vertices with types C and T and then, connect the vertices of type $M$ and vertices of the combination type. As seen in Fig. 5b, vertices $v_1, v_2, v_3$, and $v_4$ represent the combinations of $(c1, t1)$, $(c1, t2)$, $(c2, t1)$, and $(c2, t2)$, respectively. In the reconstructed graph, we retained only the vertices *appearing in any meta-structure instance*. However, it is crucial to note that some edges will be repeated when finding combinations of vertices. For example, as seen in Fig. 5b, $m_1$ connects to both $v_3$ and $v_4$, and the edge $e_1$ $(m_1 \rightarrow c_2)$ is used twice. We allowed this type of edge to repeat since it appears at bifurcations. Based on the above description, we propose a method for reconstructing the HG for a given meta-structure, thus converting the high time-consuming meta-structure-based search into an efficient meta-path-based search. Algorithm 1 is the proposed method, which permits edge repetition for vertices in meta-structures with multiple bifurcations.

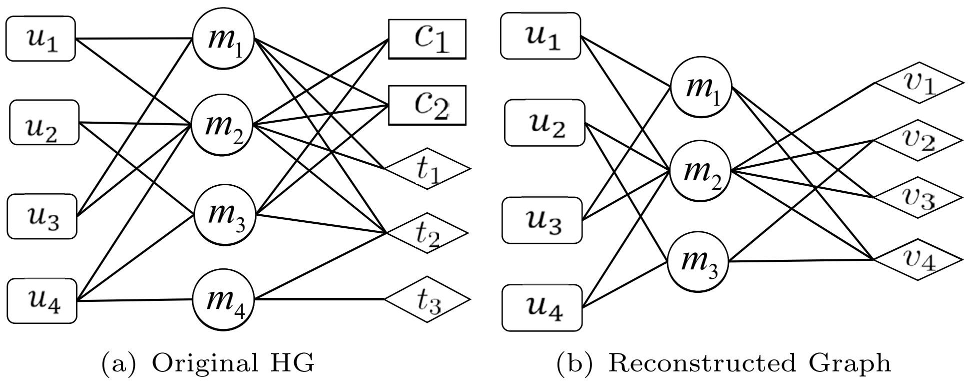


**Fig. 5** Original HG and the Reconstructed Graph

**Algorithm 1** Graph reconstruction algorithm

```
   Input: HG ($\mathcal{H}$), meta-structure($\mathcal{S}$)
   Output: A new graph ($\mathcal{H}'$)
1  initialize $l$ = the diameter of $\mathcal{S}$
   for i ← 1 to $l$ do
2      if $len(\mathcal{S}[i]) \geq 2$ then
3          initialize a map $\mathcal{M}$, a set $\mathcal{B}$.
           collect a set $\mathcal{A}$ of vertices with the $\mathcal{S}[i-1].v$
           decompose $\mathcal{S}[i-1:i+1]$ to meta-paths, $\mathcal{P}$
           for each vertex $u \in \mathcal{A}$ do
4              using DFS to find the target vertex $w$ located at $\mathcal{S}[i+1]$
               if $w$ is connected to $u$ by $p \in \mathcal{P}$ then
5                  $\mathcal{M}[u]$.add (vertices connected to $u$ and $w$)
                   $\mathcal{B}$.add (vertices connected to $u$ and $w$)
6          end
7          set $\mathcal{C}$ : all vertex combinations in $\mathcal{B}$
           for $u$ in $\mathcal{M}.keys()$ do
8              set $\mathcal{C}[u]$ : all vertices combinations in $\mathcal{M}$.get($u$)
               for each combined vertex $v$ in $\mathcal{C}[u]$ do
9                  connect $u$ and $v$
                   update the type of $e(u, v)$ and $v$
10             end
11         end
12 end
13 return: a new graph $\mathcal{H}'$ with new index and type
```

We use $l$ to denote the number of vertical layers (i.e., levels) of the meta-structure and traverse each layer of the meta-structure. For layers containing multiple vertex types, where bifurcation appear, we initialize a Map $\mathcal{M}$ and a set $\mathcal{B}$ on lines 3–4. We collect the set of all vertices where the bifurcation begins, denoted as $\mathcal{A}$ (e.g., all vertices of type $M$), and then decompose this meta-structure into a set $\mathcal{P}$ of meta-paths on lines 5–6. We traverse every vertex $u$ of the set $\mathcal{A}$, finding the meta-path instances in $\mathcal{P}$ using the depth first search (DFS). If the path from vertex $u$ to a vertex $w$ is a meta-path instance, then, we add the vertex connecting $u$ and $w$ to both $\mathcal{M}[u]$ and $\mathcal{B}$ on lines 7–11. $\mathcal{M}[u_i]$ stores the valid vertices in instances where the bifurcations occur at $u_i$, and $\mathcal{B}$ stores all the valid vertices that bifurcate. Note that valid vertices are defined as vertices appearing in instances of the set $\mathcal{P}$ of meta-paths. After this step, the vertices in the set $\mathcal{B}$ are combined according to the vertex types in $s[i]$ on lines 13–14. Next, each vertex in the keyset of Map $\mathcal{M}$ is connected to the combined vertex belonging to it, and the type of new edges is updated on lines 15–21. Finally, we return a new graph with the given meta-structure on line 23. Given a meta-structure, we can convert the original Heterogeneous Graph into a new graph by merging the vertices at the bifurcation point. This yields a novel meta-path $\mathcal{P}_s$. The upper bound for the time complexity of Algorithm 1 is determined to be $O(l \cdot (|V| + |E|) \cdot |A| + |A|)$, where $l$ represents the length of the meta-structure, typically a single-digit value. Here, $A$ denotes the neighbors of each vertex during the algorithm's iteration, usually with $|A| \ll |V|$. The upper bound for the space complexity is $O(|V| \cdot |A|^2 + |E|)$, primarily attributed to recording the upper bounds generated by $\mathcal{C}$, and typically $|A| \ll |V|$. With the state-of-the-art meta-path-based community search algorithm BatchEcore [11], we can efficiently utilize complex semantic information, i.e., meta-structure, to do community search on heterogeneous graphs.

**Theorem 1** *To get the maximum community, the BatchEcore algorithm can be used on the reconstructed graph with the transformed meta-paths. This is equivalent to performing* *meta-structure-based community search on the original HG and enables edges to be repeated at a bifurcation point in the meta-structure.*

***Proof*** BatchEcore [11] first proposed to do community search on HGs and exclusively utilized meta-paths for this purpose. To address the low engagement problem (i.e., the "spurious viewer" $u_3$ in Fig. 1), [11] proposed three different models to find the designated community in HG, which are edge and vertex repeatable, edge-disjoint, and vertex-disjoint, respectively. As introduced in the INTRODUCTION, the edge and vertex repeatable model often results in a significant number of "spurious viewers" who, in reality, do not exhibit a high level of engagement. The vertex-disjoint model may lead to many unconnected vertices, as the number of vertices in bifurcated meta-structures is often limited. BatchEcore is an algorithm that requires edge-disjoint meta-structures, removing vertices with core numbers less than $k$ in a batch. It computes the number of meta-path-based neighbors of a vertex using the Max Flow algorithm [56]. Referring to BatchEcore, we define $\beta(v, \mathcal{C})$ as the edge-disjoint meta-structure instances which start from vertex $v$ and end at vertex in set $\mathcal{C}$. Our objective is to find a community $\mathcal{C}$ such that $\forall v \in \mathcal{C}, \beta(v, \mathcal{C}) \geq k$. Inspired by this, given the HGs and the searching meta-structure, we reconstruct the original HG into a new graph. It is perfectly feasible to apply the BatchEcore method directly on the reconstructed graph, not only to allow edges that connect the bifurcations to be repeated, but also to efficiently compute the exact $\beta(v, \mathcal{C})$. The requirement that edges at a bifurcation can be repeated is equivalent to the maximum community with edge-disjoint searched by BatchECore.

BatchEcore algorithm aims to find the largest community, so the $\beta(v, S)$ of each vertex in the community $\mathcal{C}$ is computed independently by the maximum flow algorithm. This makes it possible to reuse certain edges when computing $\beta(v', S)$ for different vertices $v'$, but it does not lead to an increase in the value of $k$. It can be seen from Fig. 6 that the addition of combinatorial vertices causes changes to the flow network. But this does not change the maximum flow of the network. The max flow min-cut theorem [56] states that in a flow network, if there is a maximum flow $f$ in the network, the flow value of $f$ is equal to the capacity of the minimum cut. An s-t cut $C = (S, T)$ is a division of the vertices of the network into two parts $S$, $T$, and $s \in S$ and $t \in T$. We use $c$ to denote the number of edges that connect the source part of the cut to the sink part. For example, in Fig. 6, $c1$, $c2$, $c3$ and so on are all cuts of this flow network, and we denote the smallest cut as $c_{min}$.

For community search, we construct a flow network with vertex $u$ as the source node and specify the capacity of each edge as 1. There is only one vertex in the layer of the source node $u$. So if we take $u$ as $S$ and the other vertices as $T$, the size of the cut, denote as $c_1$, is the number of vertices connected to $u$. We use $f$ to denote the maximum flow of this network, obviously, $c \geq f$. The addition of combinatorial vertices will bring a change in the cuts of the two layers,

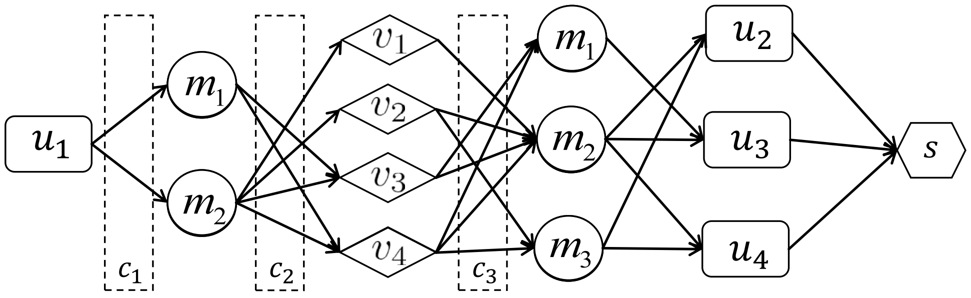


**Fig. 6** The flow network

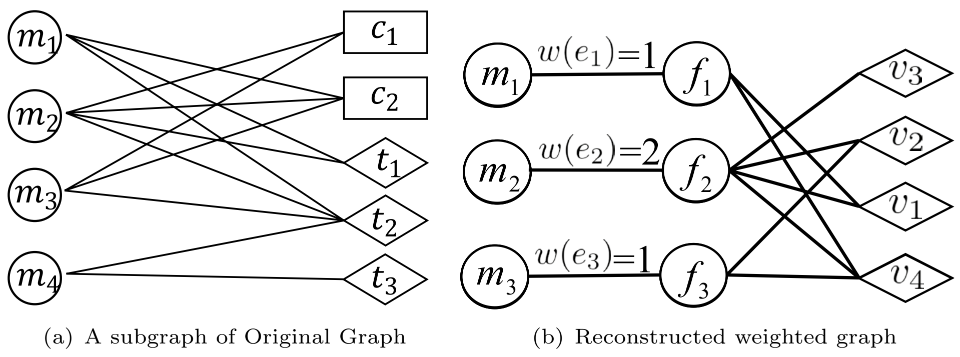


**Fig. 7** The reconstructed edge-weight graph

for example, $c_2$ and $c_3$ in Fig. 6. Combinatorial vertices are virtual vertices used to connect different vertices that can be connected by a meta-structural bifurcation. Therefore, the minimum value of $c_2$ and $c_3$ must be no less than $c_1$, which can ultimately be formalized as: $min(c_2, c_3) \geq c_1 \geq f$. It can be seen that the addition of combined vertex layers does not change the max flow $f$. So, we can directly use the BatchEcore method on the reconstructed graph and can achieve the goal of allowing the edges at the bifurcation to be repeated. □

## 5.3 Optimization

We propose an optimization of the graph reconstruction algorithm to improve the accuracy and generality. Specifically, Algorithm 1 may be inaccurate in counting meta-structure instances in certain problems, as exemplified by Fig. 7a, which shows a subgraph with only $M$, $C$, and $T$ vertex types derived from Fig. 5a. For example, if $m_2$ is connected to both the combination vertices $(c_1, t_1)$ and $(c_2, t_2)$, the edge $e(m_1, c_2)$ would be counted twice, which can result in an incorrect $\beta(k, S)$. To address this issue, we propose an optimized edge-weighted graph reconstruction algorithm that can record the number of instances of meta-structures. Figure 7b shows the edge-weighted graph reconstructed from Fig. 7a, which includes a new vertex type $f$ called the "flow constraint vertex" and assigns weights to each edge. To conserve storage, we only store the weights for edges of type $\psi(e(m,f))$, while the default weight for other types is 1. The weight of edges in type $\psi(e(m,f))$ is determined by counting the number of instances of the sub-meta-structure where the meta-structure bifurcations. As an example, $m_2$ from Fig. 7a has four possible meta-structure instances. However, without repeated edges, there are only two meta-structure instances ($s_1 = m_2 \rightarrow c_2 \&\& t_1 \rightarrow m_1$, $s_2 = m_2 \rightarrow c_1 \&\& t_2 \rightarrow m_3$) connected to vertices of type $\phi(m_2)$ from $m_2$, so the weight of edge $e_2(m_2, f_2)$ is 2 (e.g., $w(e_2) = 2$). Based on the above discussion, we introduce "flow constrained vertices" in Algorithm 2 to capture the number of instances of the meta-structure.

**Algorithm 2** Edge-weighted graph reconstruction algorithm

```
  Input: HG ($\mathcal{H}$), meta-structure($\mathcal{S}$)
  Output: A new graph $\mathcal{H}''$ with weight
1 //Replace lines 8-13 in Alg. 1 by the following:
   for $u$ in $\mathcal{M}.keys()$ do
2     set $C_u$ : all vertices combinations in $\mathcal{M}$.get($u$)
      $min_s$ : number of instances of meta-structure with disjoint edges
      create a new vertex $f_i$ and connect it to $u$
      $w(e(f_i, u)) = min_s$
      for each vertex $v$ in $C_u$ do
3        connect $f_i$ and $v$
           update the type of $e(u, f_i)$, $e(f_i, v)$, $v$, $f_i$
4     end
5 end
6 return: a new edge weighted graph $\mathcal{H}''$
```

We replace lines 15–21 in Algorithm 1 by the following: For each vertex $u$ in the key set of Map $\mathcal{M}$, we capture the number of meta-structure bifurcation instances connected to $u$ as follows: First, we combine its neighbors based on their vertex types at the bifurcation (lines 2–3). Next, we compute $min_s$, which is the minimum number of vertices present in each bifurcation type that vertex $u$ is connected to (line 4). To reflect this information, we introduce a new flow-constrained vertex, $f_i$, for each $u$ and connect it to $u$. The edge weight of $e(f_i, u)$ is set to $min_s$ (lines 5–6). Then, we connect $f_i$ to the combinatorial vertices related to vertex $u$ and update the types of $e(u,f_i)$, $e(f_i, v)$, $v$, and $f_i$ (lines 7–9). It is worth noting that $u$ is not directly linked to the combinatorial vertices. Algorithm 2 is an improvement over Algorithm 1, with the most significant difference being the inclusion of edge-weight computation. Consequently, the upper bound for time complexity

**Table 2** Statistics of real datasets

| DataSet | Vertices | Edges | Vertex Types | Edge Types |
|---|---|---|---|---|
| AMRN | 174,014 | 1,727,449 | 4 | 3 |
| ASN | 318,663 | 569,195 | 4 | 3 |
| DBLP-4-Area | 10,669 | 44128 | 4 | 3 |

is $O(l \cdot (|V| + |E|) \cdot |A| + |A| + |E|)$, and the upper bound for space complexity is $O(|V| \cdot |A|^2 + 2 \times |E|)$. As stated in Theorem 1, our proposed graph reconstruction algorithm allows us to efficiently and accurately utilize the complex semantic relationships present in heterogeneous graphs (e.g., conducting community search through meta-structures).

## 6 Experiments

### 6.1 Experimental Setup

**Datasets.** We use three real datasets with textual information at vertices and a moderate number of types: the Amazon Movie review network dataset (AMRN) [3], the academic social network dataset (ASN) [57], and a bibliographic network dataset DBLP-4-Area [48]. The statistics of the three datasets are shown in Table 2. The DBLP-4-Area dataset is the same dataset used in work that calculating node similarities by meta-structures [15], which has the same characteristics as the ASN dataset. The meta-path-based methods take a lot of time to process due to the large degree of the AMRN dataset. Thus, we selected the ASN and DBLP-4-Area datasets for comparison experiments. For the topic constrained experiment, we selected the DBLP-4-area dataset to ensure a fair comparison (i.e., consistent with the data type used in the baseline).

**Community Quality Metrics.** Since the vertices in the searched community are not directly connected, the existing quality evaluation metrics become inappropriate; thus, we propose an extension to overcome this challenge and expand the evaluation metrics to measure the community quality by treating the two end vertices connected by a meta-structure instance as connected and defining the distance as one. The specific evaluation metrics, namely community diameter, average path length (APL), density of connection, and clustering coefficient (CC). Specifically, we assume that any two end vertices connected by a meta-structure instance are connected and set the distance between them to one. We extend the following evaluation metrics to measure the quality of the searched community: 1. Closeness of Communities: The closeness of communities is measured using two widely accepted metrics [58], namely the community diameter and the average path length (APL). The community diameter is calculated by finding the largest shortest distance between any pair of vertices in the community, while the average path length is calculated as the average of the shortest distance between all possible pairs of vertices. Although the former may be influenced by individual vertices, the latter provides a more accurate reflection of the distance between pairs of vertices. To redefine "the distance," we set the length of a meta-structure instance to one. If the distance between two vertices is greater than one, it means that the meta-structure cannot connect them directly. Smaller values of these two metrics indicate higher overall connectedness of the community.

2. **Density of Connection:** The density of connection is defined as the number of edges over the number of vertices [59]. However, this may cause a problem that the density increases with the number of vertices, such as in complete graphs, take the complete graph as an example. According to the traditional definition, the density $D$ should be $D = n \times (n-1)/2n = (n-1)/2$. The density increases with the number of vertices, but obviously the complete graph is already a very tightly connected community independent of the number of vertices. A good evaluation metric should minimize the impact of graph size. Therefore, we define the connection density as the average degree over the number of vertices, which can be used to reduce the effect of graph size.

3. **Clustering Coefficient:** Clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together [60]. Graphs with higher clustering coefficients are found to have significant modular structures, and the average distance between different vertex pairs is smaller.

In short, a smaller diameter and average path length signifies closer relationships between searched communities; larger density of connection and clustering coefficient also indicate closer relationships.

**Baselines.** To demonstrate SNCS's superiority in various aspects, we compare it with the SOTA methods for community search on HGs: BatchEcore [11] and $k\mathcal{KP}$-core [13]. BatchEcore conducts community search on HGs through meta-paths while requiring non-repeated edges. $k\mathcal{KP}$-core redefines a $k\mathcal{KP}$-core in HGs, i.e., every vertex has at least one $\mathcal{KP}$-neighbor and $k$ path instances. Here, one $\mathcal{KP}$-neighbor refers to a meta-path instance that covers all specified keywords set. $k\mathcal{KP}$-core is a community search algorithm that utilizes meta-paths, permits edge duplicates, and incorporates keyword constraints. However, it defines a new community and cannot be directly compared to our method; thus, we made a simple modification (discussed in Table 4) for a comparison with SNCS. Furthermore, to validate the effectiveness of meta-structure-based search and topic constraints, we introduced several variants. For instance, we added topic constraints to the meta-path-based community search methods, denoted as $P_1 + T$ and $P_2 + T$. We also take the intersection of the results obtained by separately searching with two meta-path, denoted as $P_1 \cap P_2$. Additionally, we conducted a variant of the SNCS method by searching solely with structural motifs while eliminating topic constraints, referred to as SNCS/T.

**Evaluation and Query Settings.** In line with existing works about meta-structures [15], we focus on meta-structures with diameters at most four and select

**Table 3** Evaluation metrics on five experiments

| datasets / metrics / methods | time(s) | | size | | Diameter | |
|---|---|---|---|---|---|---|
| | DBLP | ASN | DBLP | ASN | DBLP | ASN |
| $P_1$ | 0.0450 | 86.7889 | 45 | 2029 | 2 | 4 |
| $P_2$ | 0.0440 | 61.6213 | 44 | 2111 | 2 | 5 |
| $P_1 \cap P_2$ | 0.0890 | 148.4102 | 44 | 2029 | 2 | 4 |
| SNCS/T | 0.0390 | 67.3393 | 39 | 2024 | 2 | 4 |
| $P_1$+T | 0.0297 | 4.3244 | 21.4 | 239 | 2 | 3.75 |
| $P_2$+T | 0.031 | 2.2875 | 18.9 | 191 | 2.4 | 3 |
| SNCS | **0.0215** | **2.3123** | **26** | **235** | **2** | **3** |
| **datasets / metrics / methods** | **APL** | | **CC** | | **density** | |
| | DBLP | ASN | DBLP | ASN | DBLP | ASN |
| $P_1$ | 1.2616 | 1.2862 | 0.8504 | 0.8884 | 0.7220 | 0.7228 |
| $P_2$ | 1.2336 | 1.3140 | 0.8556 | 0.8864 | 0.7490 | 0.7020 |
| $P_1 \cap P_2$ | 1.2336 | 1.2862 | 0.8556 | 0.8884 | 0.7490 | 0.7228 |
| SNCS/T | 1.1943 | 1.2861 | 0.8760 | 0.8886 | 0.7850 | 0.7230 |
| $P_1$+T | 1.2088 | 1.2197 | 0.8674 | 0.8919 | 0.7471 | 0.7716 |
| $P_2$+T | 1.2852 | 1.2718 | 0.8445 | 0.8493 | 0.6740 | 0.6659 |
| SNCS | **1.1545** | **1.1481** | **0.8998** | **0.9221** | **0.8049** | **0.8359** |

The bold font signifies the best performance under each column

meta-structures with more connected vertices as expert suggest, so as to ensure that our query is meaningful. We apply the "Edge-weighted graph reconstruction algorithm" to reconstruct a new graph and generated over 1000 queries for each dataset. The reported values in all experiments are the average values. To ensure a fair comparison, we first randomly selected multiple query vertices and specified the searching meta-structure. Using the ASN dataset as an example, the meta-structure was defined as $\mathcal{S} = A \to P \to L\&\&Y \to P \to A$, where the vertex types $A$, $P$, $L$, and $Y$ represent author, paper, publication, and year, respectively. The communities retrieved by SNCS were denoted as $\mathcal{C}_T$. To compare them with the algorithm for meta-path-based community search, we decomposed the meta-structure $S$ into two meta-paths $\mathcal{P}_1 = APLPA$ and $\mathcal{P}_2 = APYPA$. The community retrieved by BatchEcore with $\mathcal{P}_1$ and $\mathcal{P}_2$ was labeled as $\mathcal{C}_{p_1}$ and $\mathcal{C}_{p_2}$, respectively. As previously discussed, meta-structures exhibit better semantic associations than meta-paths and are not merely a combination of meta-paths. To validate this point, we defined the community $\mathcal{C}_m$ as the common vertices of $\mathcal{C}_{p_1}$ and $\mathcal{C}_{p_2}$, i.e., $\mathcal{C}_m = \mathcal{C}_{p_1} \cap \mathcal{C}_{p_2}$. Additionally, we conducted ablation experiments to demonstrate the effectiveness of topic constraint in community search, namely SNCS/T, and the communities are labeled as $\mathcal{C}_S$.

Our dataset contains four vertex types that, coincidentally, constitute a meta-structure. To ensure the validity of our experiments, we randomly selected 20 vertices as the set of query vertices. The topic similarity threshold $\theta$ is set from 0.5 to 0.95, and the $k$ is set from 1 to 12. In the results reported in the following, each data point is the average result for these queries. From the parameter analysis section in paper, we have analyzed that a larger $k$ and $\theta$ mean denser topology. In order to identify intermediate-sized dense communities surrounding query vertices in the DBLP and ASN datasets, we determined appropriate parameter configurations to be $k = 9$ and $\theta = 0.60$, and 0.95, respectively (the settings of Table 3 in paper). For the parameters settings of Table 4 in paper, we utilized the same vertices set as query vertices and set $k$ equal to 9 for $k$-core. For $k\mathcal{KP}$-core, we employed the exact keywords set that corresponded with the query vertices, while for SNCS, we used the topic vectors that had been extracted from these vertices. Topic constraints for $k\mathcal{KP}$-core and SNCS were established at 0.08 and 0.60, respectively. Setting topic constraints for $k\mathcal{KP}$-core that covers a certain proportion of the keyword set are challenging, so we increased $\theta$ starting from 0.01 in increments of 0.01 until the optimal community was achieved. Our codes and all the detailed settings are open in the repository, https://github.com/LIyvqi/SNCS.

## 6.2 Experimental Performance

In this subsection, we emphasize the superior performance of SNCS. Firstly, we introduce the efficiency of the SNCS' graph reconstruction algorithm. Following that, we compare SNCS with baselines to demonstrate the high quality of the

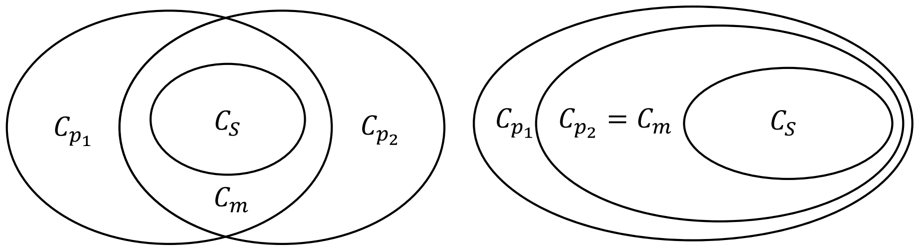


**Fig. 8** The relationships between communities

**Table 4** Comparison of topic constrained methods

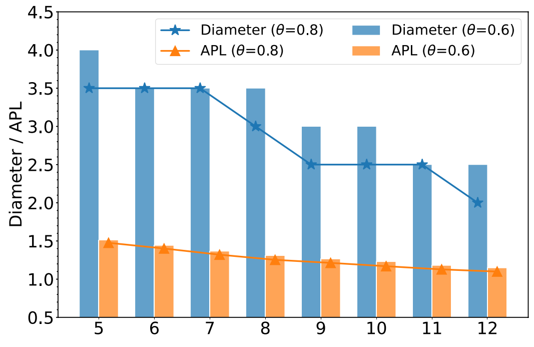
| Evaluation metrics | Diameter | APL | Clustering coefficient |
|---|---|---|---|
| $k\mathcal{KP}$-core | 7.5 | 2.7607 | 0.9050 |
| SNCS | 2.0 | 1.1048 | 0.9285 |

communities retrieved by SNCS. We further validate the effectiveness of meta-structures and the topic model through ablation experiments. Finally, we present the highlights of the topic constraints. Table 3 provides a report on the comparative experiments, including their respective search times and searched community evaluations. And table 4 provides a comparative analysis between SNCS and other methods utilizing keyword-based community search.

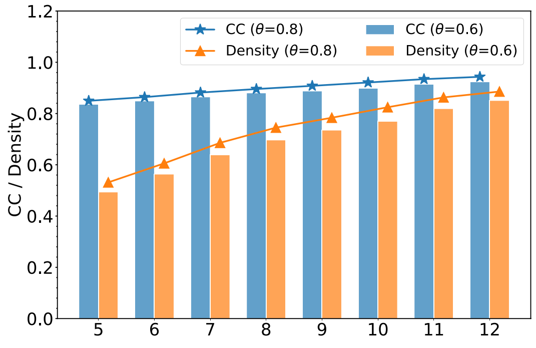
**Efficiency.** Given the HGs and the meta-structure, SNCS only need to reconstruct the HG to the edge-weighted graph once and it is very efficient. Reconstructing AMRN, ASN, and DBLP-4-Area required 2.6s, 3056.0s and 4.0s, respectively. This is because the algorithm becomes computationally more complex to compute the combination of adjacent vertices as the number of vertices under each vertex type increases. The inclusion of topic constraints significantly reduces the size of searches, especially on larger datasets such as ASN. As shown in Table 3, SNCS requires no more than 4% of the time for meta-path search. The SNCS/T algorithm is more efficient than meta-path-based searches because the graph reconstruction algorithm reduces graph size. In addition, it is generally more efficient than community search through meta-paths. If there is uncertainty regarding which meta-path to select, employing a meta-structure that includes multiple meta-paths can accelerate the search for high-quality communities.

**Community Quality.** For the retrieved communities, smaller size indicates stronger vertex connections. SNCS and SNCS/T can retrieve more closely connected communities by removing weakly correlated vertices while meta-paths cannot filter out these weak correlations. Table 3 reports the order of retrieved community sizes as $\mathcal{C}_t < \mathcal{C}_S < \mathcal{C}_m \leq min\{\mathcal{C}_{p_1}, \mathcal{C}_{p_2}\}$, showing that using a meta-structure avoids the drawbacks of methods that combine meta-paths. In practice, ASN and DBLP have only four vertex types, which together form one meta-structure. In academic citation networks, the meta-path $P_2 = APYPA$ connects more vertices than $P_1 = APLPA$, leading to a community relationship: $\mathcal{C}_S \in \mathcal{C}_m = min\{\mathcal{C}_{p_1}, \mathcal{C}_{p_2}\} \in max\{\mathcal{C}_{p_1}, \mathcal{C}_{p_2}\}$. Theoretically, the communities retrieved by different methods should have the relationship as in Fig. 8a. However, ASN and DBLP-4-Area have only four vertex types, which exactly constitute one meta-structure. In academic citation networks, the meta-path $P_2 = APYPA$ can connect more vertices than $P_1 = APLPA$, which leads to the relationships between communities retrieved as shown in Fig. 8b, which is a special case of Fig. 8a. In such datasets, it is difficult to search for tightly connected communities by meta-paths alone.

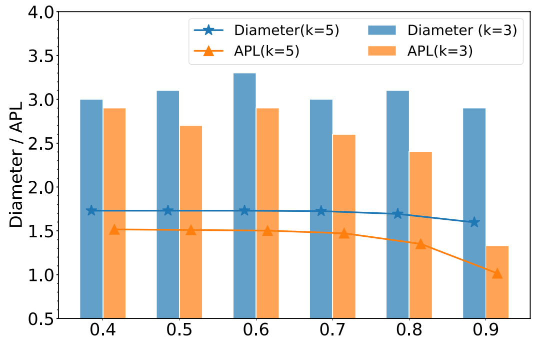
**Highlights of topic constraints.** To the best of our knowledge, currently, $k\mathcal{KP}$-core is the only approach that performs community search on HGs while considering thematic constraints (i.e., specified keyword set). This approach is based on meta-paths and mandates that searched communities include at least one path instance that comprises the full specified keywords. However, specifying keywords are challenging and does not guarantee that communities can always be found. We have fine-tuned the topic constraints of $k\mathcal{KP}$-core to achieve a specific keywords coverage ratio $\theta$. This fine-tuning allows for effective comparisons between communities. We report the comparative results with the same $k$ in Table 4 and select $\theta$ that produced the highest quality communities. SNCS detects significantly higher quality communities than $k\mathcal{KP}$-core and has an advantage over using specified keywords in terms of convenience. Table 3 shows that incorporating topic constraints can enhance both model efficiency and performance. An interesting phenomenon is that identical communities could be found for different query vertices if we ignore topic constraints, which is a limitation of existing works on community search. SNCS overcomes the limitations of topological similarity by introducing a topic model that creates a customized community for the query vertex $q$. Although community quality is evaluated through topology, the communities retrieved by SNCS outperform those obtained using other methods.



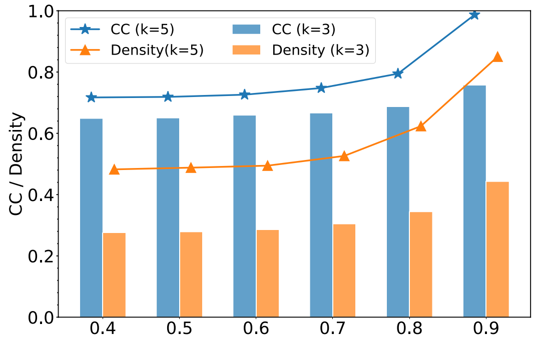
(a) Diameter and APL on ASN



(b) CC and Density on ASN

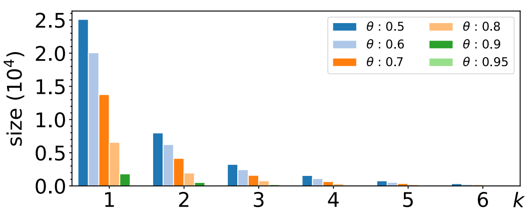


(c) Diameter and APL on DBLP

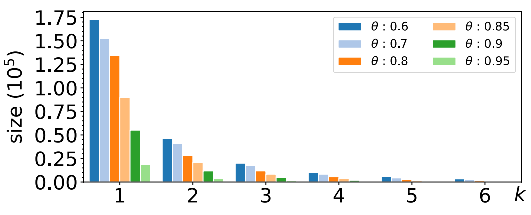


(d) CC and Density on DBLP

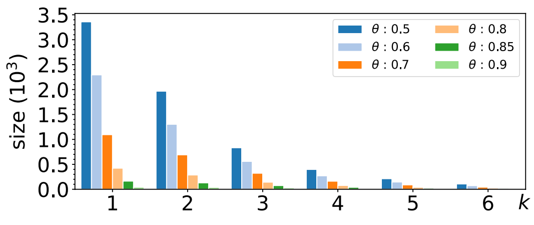
**Fig. 9** Diameter, APL, CC and Density of the retrieved community



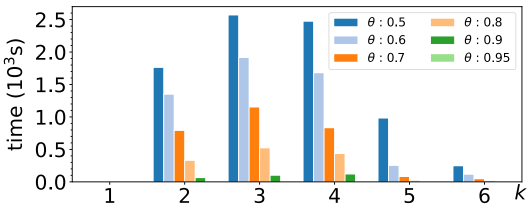
(a) The size of AMRN



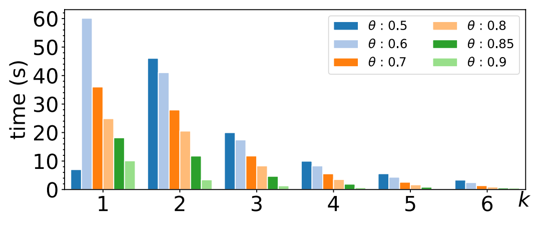
(b) The size of ASN



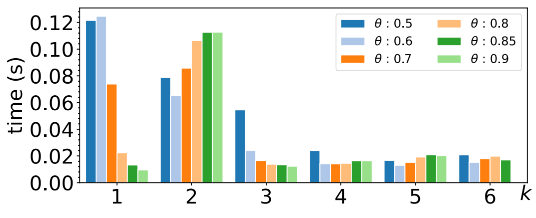
(c) The size of DBLP-4-Area



(d) Running time of AMRN



(e) Running time of ASN



(f) Running time of DBLP-4-Area

**Fig. 10** Size and Running time of "SNCS" with different $k$ core number and $\theta$

### 6.3 Parameter Analysis

We analyze the performance of SNCS in terms of community quality, community size, and runtime under varying parameter settings.

**Quality.** The community obtained by SNCS exhibits a high level of connectivity. As demonstrated in Fig. 9, we evaluated the quality of the communities on two datasets, ASN and DBLP-4-Area, under varying values of $k$ (ranging from 5 to 12) and $\theta$ (from 0.4 to 0.9). The results demonstrate that the community obtained by SNCS represents a dense subgraph, with an increasing level of connectivity as $k$ and $\theta$ increase. For example, on the ASN dataset, the diameters of the graphs are no greater than 3; the average path length is less than 2; the clustering coefficients are above 0.7, and the density is greater than 0.8, particularly. The density of the community is becoming larger as $k$ or $\theta$ increase. For example, when analyzing the ASN dataset, we observed that the diameter of the graphs was less than or equal to 3; the average path length was below 2; the clustering coefficients were higher than 0.7, and the density exceeded 0.8. Moreover, as depicted in Fig. 9a, b, we utilized bars and lines of the same color to indicate the same quality evaluation metric at $\theta$ values of 0.6 and 0.8, respectively. Our findings suggest that, as $k$ increases, the influence of increasing $\theta$ on community density decreases. Similarly, we employed the same color of bars and lines to represent the same quality evaluation metric at $k = 3$ and $k = 5$ for Fig. 9c, d, respectively. Our results demonstrate that, in the same $(k, \mathcal{S})$-core, the obtained communities become increasingly cohesive as $\theta$ increases, particularly when $k$ is small.

**Size.** This analysis examines the $(k, \mathcal{S})$-cores size distribution for various $\theta$ values to evaluate our proposed SNCS. Figure 10a–c displays the results of the analysis for three datasets, where $k$ values range from 1 to 6 along the horizontal axis, the number of vertices in the community is shown on the vertical axis, and different colors represent various topic similarity thresholds $\theta$. As shown in the figure, the number of vertices in the community diminishes as $k$ increases while remaining under the same similarity threshold. Similarly, the number of vertices in the same $(k, \mathcal{S})$-core decreases as $\theta$ increases. Notably, although an increase in $\theta$ leads to a reduction in the number of vertices across the community, the extent of decrease varies among different $(k, \mathcal{S})$-cores. As the value of $k$ increases, the decrease in the number of vertices caused by a similar increase in $\theta$ reduces. This finding suggests that vertices within higher $(k, \mathcal{S})$-cores exhibit greater similarity. Referring to the analysis of Quality, we can find that on the same heterogeneous graph, given the same query vertex, a smaller community means that the vertices in the community are more tightly connected.

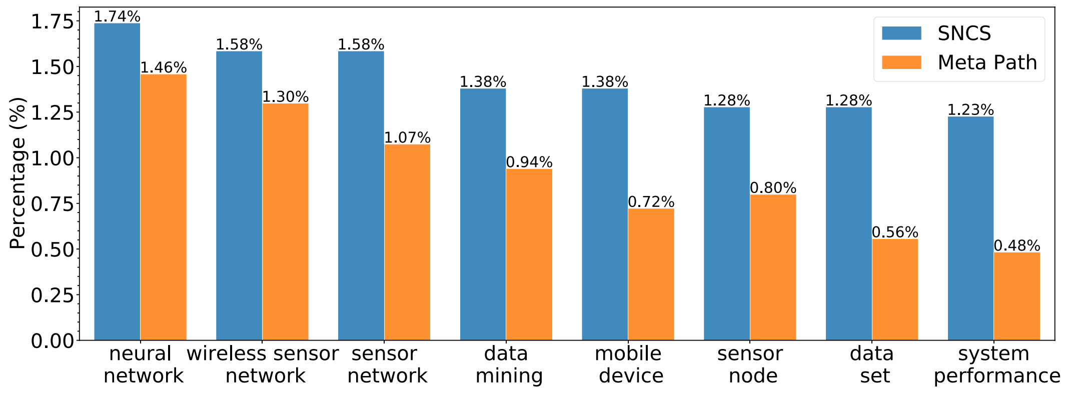
**Efficiency.** The efficiency of the search process on the reconstructed graph is presented in Fig. 10d–f, which shows the running time at different $k$ and $\theta$. Notably, with the same $(k, \mathcal{S})$-core value, it can be observed that the running time decreases as the similarity threshold $\theta$ increases. For different datasets, the $(k, \mathcal{S})$-core corresponding to the longest run time is different, and as $k$ increases, the run time tends to rise and then fall. As we mentioned earlier, the similarity between vertices is high in a $(k, \mathcal{S})$-core with a large value of $k$, but larger $k$ may lead to increased time consuming. We can increase efficiency by reasonably increasing the $\theta$.

In summary, increasing $k$ and $\theta$ results in smaller communities with more interconnected vertices. The time for community search decreases as $\theta$ increases in a monotonic trend. However, as $k$ increases, the search time tends to increase and then decrease. Denoting the inflection point as $k_t$, if we select a $(k, \mathcal{S})$-core with a $k$ smaller than or near $k_t$, we can increase $\theta$ to obtain a more semantically relevant community. This will not only improve the efficiency, but also break the topological constraints. When $k$ is larger than $k_t$, we can choose a smaller $\theta$ (at around $\theta = 0.75$), which improves search efficiency and enhances semantic association.

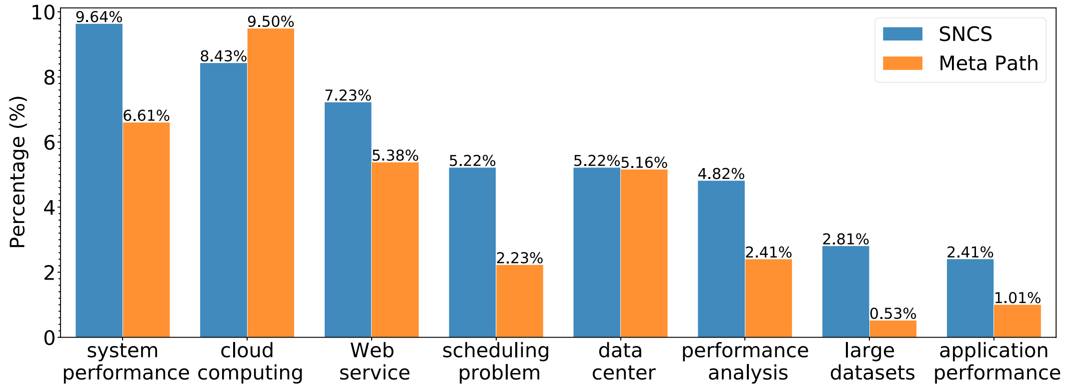
### 6.4 Case Study

We conducted a case study to compare the effectiveness of SNCS and BatchEcore (a meta-path-based approach) [11] on the ASN dataset. In this dataset, each researcher is associated with keywords related to their field of study. Assume that the query vertex $q$ is Prof. Michael R. Lyu (a researcher in computer science); the meta-structure is $\mathcal{S} = A \rightarrow P \rightarrow L\&\&Y \rightarrow P \rightarrow A$; the meta-path is $\mathcal{P} = A \rightarrow P \rightarrow L \rightarrow P \rightarrow A$, while $k = 14$ and $\theta = 0.7$. There are 101 and 621 scholars searched by SNCS and BatchEcore, respectively. Figure 11a displays the distribution of keywords in the community, with horizontal coordinates indicating the most frequent keywords and vertical coordinates indicating the relative frequency of keywords.

In contrast to the meta-path-based method, the community obtained by SNCS is highly thematically relevant, and its keywords are distributed more centrally. Figure 11b reveals the advantages of personalization in SNCS, where the vertical coordinate indicates the percentage of people containing the keyword to the total number of people in the community, denoted as $f_{key}$. SNCS has a better $f_{key}$ than the meta-path-based approach for most keywords, especially on the keywords with a smaller $f_{key}$. This result is in line with our expectation. If someone wants to organize a seminar, the members of the community found by SNCS not only has a concentrated keyword distribution, but also have a higher semantic similarity with the query vertex $q$.



(a) Distribution of keywords in the community



(b) Percentage of people in the community with the query vertex's keywords

**Fig. 11** Keywords statistics in the searched community

## 7 Conclusion

In this paper, in order to improve the quality of community search and find a community with closely related vertices and rich semantic meanings, we have formalized and solved a new problem, SNCS. To tackle this problem, we first propose to use meta-structures for community search and design an efficient graph reconstruction algorithm with partial edge-disjoint constraints, to better meet the practical needs. We also add topic constraints to find a result community, where vertices are not only topologically closely related, but also having high semantic similarities. Through extensive experiments and case study analysis, we have demonstrated that the resulting communities found by our proposed method are tightly cohesive and have good semantic similarities. When comparing the quality of the communities found by different methods, SNCS is better on all metrics than the competing methods. Moreover, our proposed method is much more efficient, especially on large datasets. In summary, meta-structures can express more complex relationships among vertices, which cannot be achieved by simply combining different meta-paths. Adding topic constraints can filter out vertices which may have high topological similarity, but low semantic similarity to the given query vertex $q$, resulting in a tightly cohesive community from the aspects of both topology and semantic meaning.

**Acknowledgements** This work is supported by the National Natural Science Foundation of China (Grant Number 62172237, U1936206).

## Declarations

**Conflict of interest** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

1. Li L, Duan L, Wang J, He C, Chen Z, Xie G, Deng S, Luo Z (2023) Memory-enhanced transformer for representation learning on temporal heterogeneous graphs. Data Sci Eng 8(2):98–111. https://doi.org/10.1007/S41019-023-00207-W
2. Tuteja S, Kumar R (2022) A unification of heterogeneous data sources into a graph model in e-commerce. Data Sci Eng 7(1):57–70. https://doi.org/10.1007/S41019-021-00174-0
3. Ni J, Li J, McAuley J (2019) Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), pp 188–197
4. Cui W, Xiao Y, Wang H, Lu Y, Wang W (2013) Online search of overlapping communities. In: Proceedings of the 2013 ACM SIGMOD international conference on management of data, pp 277–288
5. Cui W, Xiao Y, Wang H, Wang W (2014) Local search of communities in large graphs. In: Proceedings of the 2014 ACM SIGMOD international conference on management of data, pp 991–1002
6. Fang Y, Wang Z, Cheng R, Li X, Luo S, Hu J, Chen X (2018) On spatial-aware community search. IEEE Trans Knowl Data Eng 31(4):783–798
7. Armenatzoglou N, Papadopoulos S, Papadias D (2013) A general framework for geo-social query processing. Proc VLDB Endow 6(10):913–924. https://doi.org/10.14778/2536206.2536218
8. Li R-H, Qin L, Ye F, Yu JX, Xiao X, Xiao N, Zheng Z (2018) Skyline community search in multi-valued networks. In: Proceedings of the 2018 international conference on management of data, pp 457–472
9. Fang Y, Cheng R, Luo S, Hu J (2016) Effective community search for large attributed graphs. Proc VLDB Endow 9(12):1233–1244
10. Huang X, Lakshmanan LV (2017) Attribute-driven community search. Proc VLDB Endow 10(9):949–960
11. Fang Y, Yang Y, Zhang W, Lin X, Cao X (2020) Effective and efficient community search over large heterogeneous information networks. Proc VLDB Endow 13(6):854–867
12. Yang Y, Fang Y, Lin X, Zhang W (2020) Effective and efficient truss computation over large heterogeneous information networks. In: 2020 IEEE 36th international conference on data engineering (ICDE), pp 901–912. IEEE
13. Qiao L, Zhang Z, Yuan Y, Chen C, Wang G (2021) Keyword-centric community search over large heterogeneous information networks. In: Jensen CS, Lim E-P, Yang D-N, Lee W-C, Tseng VS, Kalogeraki V, Huang J-W, Shen C-Y (eds) Database systems for advanced applications. Springer, Cham, pp 158–173
14. Yang F, Ma H, Gao W, Li Z (2022) Community search over heterogeneous information networks via weighting strategy and query replacement. Front Comput Sci 16(4):164345. https://doi.org/10.1007/s11704-022-1329-9
15. Huang Z, Zheng Y, Cheng R, Sun Y, Mamoulis N, Li X (2016) Meta structure: Computing relevance in large heterogeneous information networks. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1595–1604
16. Terragni S, Fersini E, Galuzzi BG, Tropeano P, Candelieri A (2021) OCTIS: comparing and optimizing topic models is simple! In: Proceedings of the 16th conference of the European chapter of the association for computational linguistics: system demonstrations, pp 263–270
17. Sozio M, Gionis A (2010) The community-search problem and how to plan a successful cocktail party. In: Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining, pp 939–948
18. Fang Y, Huang X, Qin L, Zhang Y, Zhang W, Cheng R, Lin X (2020) A survey of community search over big graphs. VLDB J 29(1):353–392
19. Huang X, Chen D, Ren T, Wang D (2021) A survey of community detection methods in multilayer networks. Data Min Knowl Disc 35(1):1–45
20. Jin D, Yu Z, Jiao P, Pan S, He D, Wu J, Yu PS, Zhang W (2023) A survey of community detection approaches: from statistical modeling to deep learning. IEEE Trans Knowl Data Eng 35(2):1149–1170. https://doi.org/10.1109/TKDE.2021.3104155
21. Sun Y, Tang J, Han J, Gupta M, Zhao B (2010) Community evolution detection in dynamic heterogeneous information networks. In: Proceedings of the eighth workshop on mining and learning with graphs, pp 137–146
22. Basu S, Shekhar S, Kumar N, Mukherjee S, Pan I (2017) A particle swarm modelforstatic community detection based on homogeneous features. In: 2017 2nd IEEE international conference on recent trends in electronics, information communication technology (RTEICT). IEEE, pp 1507–1510
23. Wu Y, Fu Y, Xu J, Yin H, Zhou Q, Liu D (2023) Heterogeneous question answering community detection based on graph neural network. Inf Sci 621:652–671. https://doi.org/10.1016/j.ins.2022.10.126
24. Liu J, Shao Y, Su S (2021) Multiple local community detection via high-quality seed identification over both static and dynamic networks. Data Sci Eng 6(3):249–264. https://doi.org/10.1007/S41019-021-00160-6
25. Moscato V, Sperlì G (2021) A survey about community detection over on-line social and heterogeneous information networks. Knowl Based Syst 224:107112. https://doi.org/10.1016/j.knosys.2021.107112
26. Luo L, Fang Y, Cao X, Zhang X, Zhang W (2021) Detecting communities from heterogeneous graphs: a context path-based graph neural network model. In: Proceedings of the 30th ACM international conference on information & knowledge management. CIKM ’21. Association for Computing Machinery, New York, NY, USA, pp 1170–1180. https://doi.org/10.1145/3459637.3482250
27. Zheng Y, Zhang X, Chen S, Zhang X, Yang X, Wang D (2023) When convolutional network meets temporal heterogeneous graphs: an effective community detection method. IEEE Trans Knowl Data Eng 35(2):2173–2178. https://doi.org/10.1109/TKDE.2021.3096122
28. Kernighan BW, Lin S (1970) An efficient heuristic procedure for partitioning graphs. Bell Syst Tech J 49(2):291–307
29. Newman ME, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69(2):026113
30. Raghavan UN, Albert R, Kumara S (2007) Near linear time algorithm to detect community structures in large-scale networks. Phys Rev E 76(3):036106
31. Yue Y, Wang G, Hu J, Li Y (2023) An improved label propagation algorithm based on community core node and label importance for community detection in sparse network. Appl Intell 1–17
32. Al-Baghdadi A, Lian X (2020) Topic-based community search over spatial-social networks. Proc VLDB Endow 13(11):2104–2117
33. Liu H, Ma H, Li Z, Chang L (2023) Adaptive target community search with sample expansion. Knowl Based Syst 259:110077. https://doi.org/10.1016/j.knosys.2022.110077
34. Wu Y, Zhao J, Sun R, Chen C, Wang X (2021) Efficient personalized influential community search in large networks. Data Sci Eng 6(3):310–322. https://doi.org/10.1007/S41019-021-00163-3

35. Huang X, Cheng H, Qin L, Tian W, Yu JX (2014) Querying k-truss community in large and dynamic graphs. In: Proceedings of the 2014 ACM SIGMOD international conference on management of data. SIGMOD '14. Association for Computing Machinery, New York, NY, USA, pp 1311–1322. https://doi.org/10.1145/2588555.2610495
36. Fang Y, Cheng R, Li X, Luo S, Hu J (2017) Effective community search over large spatial graphs. Proc VLDB Endow 10(6):709–720
37. Luce RD, Perry AD (1949) A method of matrix analysis of group structure. Psychometrika 14(2):95–116
38. Cohen J (2008) Trusses: Cohesive subgraphs for social network analysis. National security agency technical report 16(3.1)
39. Islam MS, Ali ME, Kang Y, Sellis T, Choudhury FM, Roy S (2022) Keyword aware influential community search in large attributed graphs. Inf Syst 104:101914. https://doi.org/10.1016/j.is.2021.101914
40. Zhou Y, Fang Y, Luo W, Ye Y (2023) Influential community search over large heterogeneous information networks. Proc VLDB Endow 16(8):2047–2060
41. Zhou Y, Zhou L, Wang J, Wang L, Kong B (2023) Spatial-aware community search over heterogeneous information networks. In: Meng X, Li X, Xu J, Zhang X, Fang Y, Zheng B, Li Y (eds) Spatial data and intelligence. Springer, Cham, pp 103–114
42. Barbieri N, Bonchi F, Manco G (2013) Topic-aware social influence propagation models. Knowl Inf Syst 37(3):555–584
43. Papadimitriou CH, Raghavan P, Tamaki H, Vempala S (2000) Latent semantic indexing: a probabilistic analysis. J Comput Syst Sci 61(2):217–235
44. Févotte C, Idier J (2011) Algorithms for nonnegative matrix factorization with the $\beta$-divergence. Neural Comput 23(9):2421–2456
45. Blei DM, Ng AY, Jordan MI (2003) Latent Dirichlet allocation. J Mach Learn Res 3:993–1022
46. Srivastava A, Sutton C (2017) Autoencoding variational inference for topic models. arXiv preprint arXiv:1703.01488
47. Dieng AB, Ruiz FJ, Blei DM (2020) Topic modeling in embedding spaces. Trans Assoc Comput Linguist 8:439–453
48. Sun Y, Han J, Yan X, Yu PS, Wu T (2011) PathSim: meta path-based top-k similarity search in heterogeneous information networks. Proc VLDB Endow 4(11):992–1003
49. Liu X, Yu Y, Guo C, Sun Y (2014) Meta-path-based ranking with pseudo relevance feedback on heterogeneous graph for citation recommendation. In: Proceedings of the 23rd ACM international conference on conference on information and knowledge management, pp. 121–130
50. Zheng Y, Shi C, Cao X, Li X, Wu B (2017) Entity set expansion with meta path in knowledge graph. In: Kim J, Shim K, Cao L, Lee J, Lin X, Moon Y (eds) Advances in knowledge discovery and data mining—21st Pacific-Asia conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I. Lecture notes in computer science, vol 10234, pp 317–329. https://doi.org/10.1007/978-3-319-57454-7_25
51. Lao N, Cohen WW (2010) Relational retrieval using a combination of path-constrained random walks. Mach Learn 81(1):53–67
52. Li J, Ge B, Yang K, Chen Y, Tan Y (2017) Meta-path based heterogeneous combat network link prediction. Physica A 482:507–523
53. Ji H, Shi C, Wang B (2018) Attention based meta path fusion for heterogeneous information network embedding. In: Pacific rim international conference on artificial intelligence, pp 348–360. Springer, Berlin
54. Kabir H, Madduri K (2017) Parallel k-core decomposition on multicore platforms. In: 2017 IEEE international parallel and distributed processing symposium workshops (IPDPSW). IEEE, pp 1482–1491
55. Kong Y-X, Shi G-Y, Wu R-J, Zhang Y-C (2019) K-core: theories and applications. Phys Rep 832:1–32
56. Seymour PD (1977) The matroids with the max-flow min-cut property. J Combin Theory Ser B 23(2):189–222. https://doi.org/10.1016/0095-8956(77)90031-4
57. Tang J, Zhang J, Yao L, Li J, Zhang L, Su Z (2008) Arnetminer: extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, pp 990–998
58. Huang X, Lakshmanan LVS, Yu JX, Cheng H (2015) Approximate closest community search in networks. Proc VLDB Endow 9(4):276–287. https://doi.org/10.14778/2856318.2856323
59. Wu Y, Jin R, Li J, Zhang X (2015) Robust local community detection: on free rider effect and its elimination. Proc VLDB Endow 8(7):798–809
60. Holland PW, Leinhardt S (1971) Transitivity in structural models of small groups. Comp Group Stud 2(2):107–124