Journal of Southeast University (English Edition) Vol. 42, No. 1, pp. 74-79 Mar. 2026 ISSN 1003-7985

Research on heuristic approximation algorithm of the densest *k*-subgraph discovery in large-scale dynamic graphs

HAN Tao[1], TIAN Yuxi[1], ZHAO Jianwei[1], WANG Senzhang[2]

（1. National Center for Public Credit and Geospatial Information, Beijing 100059, China; 2. School of Computer Science and Engineering, Central South University, Changsha 410083, China）

Abstract: To address the issue that static densest subgraph mining algorithms often exhibit low efficiency when handling large scale dynamic graphs, this paper proposes a heuristic approximation algorithm. The algorithm approximates the densest *k*-subgraphs of the entire graph through four steps: partitioning the large-scale dynamic graph, constructing a partial set of the densest *k*-subgraphs, heuristically merging the subgraph sets, and finally extracting the densest *k*-subgraphs. This approach significantly reduces the computational time for large-scale dynamic graphs while simultaneously improving the quality of the resulting subgraphs. This algorithm is applicable to various definitions of “density” and can accommodate diverse requirements on the number of edges. When integrated with existing static densest subgraph detection algorithms, it achieves scalability and computational efficiency. Theoretical analysis demonstrates that the optimal density of the densest *k*-subgraphs extracted by the proposed algorithm reaches 0. 9. To evaluate the performance of the algorithm, experiments were conducted on four billion-scale datasets: Friendster, Orkut, YouTube, and DBLP. The results indicate that the proposed algorithm outperforms static methods in both runtime efficiency and subgraph quality on large-scale dynamic graphs.

Key words: the densest *k*-subgraph; features; heuristic approximation algorithm; optimal density

DOI: 10. 3969/j. issn. 1003-7985. 2026. 01. 007

With advancements in web technologies and artificial intelligence, applications in e-commerce, intelligent transportation[1-2], multiradar systems[3], and social networks generate vast amounts of data on users' travel patterns, social interactions, and purchasing behaviors. These domains, such as friendships and interactions (e. g. , liking and reposting messages) in social networks or spatiotemporal information from human interactions with IoT devices, can be represented as graphs. The densest subgraph is a significant feature of a graph[4], with broad applications in social network analysis, spatiotemporal protein-protein interaction networks in biology[5-7], and event detection in spatiotemporal networks[8]. The densest *k*-subgraph (DkS) problem is a fundamental problem in graph mining[9], which identifies a subset of *k* nodes with the maximum number of edges. The DkS problem has been extensively studied from both data mining and theoretical computer science perspectives[10-18]. In data mining, the focus is on improving the runtime efficiency and quality of the densest *k*-subgraphs while also demonstrating the applicability of the proposed methods. Theoretical computer science investigations focus on the relaxed conditions of the problem, computational complexity, and approximation algorithms for overlapping densest *k*-subgraphs. Convex programming and approximation methods for directed, weighted, and multilayer graphs have been explored in other studies[19-23]. Given these diverse approaches, the definition of a “densest subgraph” can vary significantly depending on the specific problem characteristic. These definitions, such as average density[24], *k*-core[25], and clique, primarily focus on the relationship between nodes and degrees. However, the subgraphs identified by these algorithms often vary significantly in size, making subsequent visualization and further research challenging. Dynamic graphs further increase the difficulty of discovering the densest subgraphs in large-scale graphs. First, memory limitations prevent the loading of large-scale graphs into memory. Second, although existing algorithms can verify the discovery of the densest *k*-subgraphs in polynomial time, the computation time is infeasible because of the large-scale graphs. Finally, graphs typically evolve, with nodes and edges continuously changing. Traditional static graph algorithms cannot efficiently handle dynamic graphs, posing new challenges to conventional static densest *k*-subgraph discov-

---

Received 2025-04-17, **Revised** 2025-10-14.

Biography: HAN Tao (1984—), female, doctor, senior engineer, hantao@creditchina. gov. cn.

Foundation item: The National Natural Science Foundation of China (No. 62172443).

Citation: HAN Tao, TIAN Yuxi, ZHAO Jianwei, et al. Research on heuristic approximation algorithm of the densest k-subgraph discovery in large-scale dynamic graphs[J]. Journal of Southeast University (English Edition), 2026, 42(1): 74-79. DOI: 10. 3969/j. issn. 1003-7985. 2026. 01. 007.

ery algorithms.

1 Definition of the Densest *k*-Subgraph Problem

The DkS is a fundamental graph optimization problem that identifies a subgraph K consisting of fewer than k vertices, such that the number of edges within K' is maximized. Given an undirected graph $G = (V, E)$ and a positive integer $\lambda|G| \leqslant |V|$, the objective is to find a subset $K' \subseteq V$ with $|K'| \leqslant \lambda|G|$ that maximizes the edge density. In dynamic settings, identifying the DkS within a time window can be expressed as follows:

$$K' = \text{density}(G) \approx \text{density}\left(\bigcup_{i=1}^{w} \text{density}(G_i)\right)$$

$$\text{s.t.} \quad |K'| \leqslant \lambda|G| \tag{1}$$

This problem is NP-hard, and extensive research has focused on large-scale graph approximation algorithms, heuristics, and scalable solutions.

2 Framework of Heuristic Approximation of DkSs Detection

Under a sliding time window W, the graph G is partitioned into a series of subgraphs G_i. The function density (G_i) is used to identify the DkS within each G_i. Subsequently, a heuristic merging algorithm (denoted as $\bigcup_{i=1}^{w}$) is applied to aggregate the local DkS set $(K'_1, K'_2, \cdots, K'_i, \cdots, K'_w)$ into a candidate graph for the DkS. Finally, the DkS K' is extracted from this candidate graph under the constraint of edges $|K'| \leqslant \lambda|G|$.

The large-scale dynamic graph is segmented into subgraphs G_i using a fixed time window W by leveraging the locality property of social network communities. Each G_i is then processed to generate a set of local DkSs, which are merged via a heuristic algorithm to construct a partial candidate graph (PCG). The DkS K' is mined from the PCG.

The proposed algorithm comprises four key steps (Fig. 1): (1) subgraph partitioning, which divides the dynamic graph into temporal subgraphs; (2) PCG construction, which extracts local DkSs from each partition; (3) heuristic merging, which combines local results into a unified candidate graph; (4) DkS extraction, which identifies the global solution K' under edge constraints. The algorithmic framework is shown below in Algorithm 1.

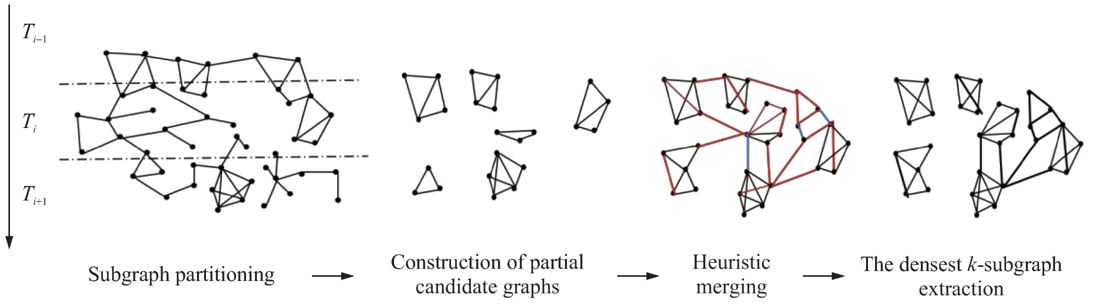


Fig. 1 Four steps of heuristic approximation of the densest *k*-subgraphs detection

Algorithm 1 Heuristic approximation of the densest *k*-subgraphs detection

```
Input: $G, \lambda$.
Output: $K'$.
Init PCG = $\varnothing$, ECG = $\varnothing$, pruned_edges = $\varnothing$;
While $i$ is not larger than $x$,
    $G_i = \{(v_{i1}, v_{i2}), (v_{i3}, v_{i4}), \cdots, (v_{ip}, v_{i(p+1)})\}$;
    $K'_i$ = density($G_i$);
    If $|K'_i| < \gamma|G_i|$,
        $S_i = K'_i \cup \gamma|G_i|$;
    else $S_i = K'_i$;
    pruned_edges = pruned_edges $\cup (G_i - S_i)$;
    PCG = PCG $\cup S_i$;
    ECG = Approximate(PCG, pruned_edges, $\gamma$);
    pruned_edges = $\varnothing$;
$K'$ = density(ECG).
```

2.1 Subgraph partitioning

In this step, the edges of the large-scale dynamic graph are initially partitioned. The edges within a specific time interval are grouped into the same time window to form a subgraph. As time progresses, a series of subgraphs $G_1, G_2, \cdots, G_i, \cdots, G_x$ are obtained.

2.2 Construction of PCG

For each partitioned subgraph, a PCG is formed by extracting the local densest subgraphs that meet the constraint $|S_i| \geqslant \gamma|G_i|$ and condition $S_i \approx \text{density}(G_i)$. The parameter γ governs the size threshold of candidate subgraphs.

$$\text{PCG} = \text{PCG} \cup S_i \approx \text{PCG} \cup \text{density}(G_i)$$

$$\text{s.t.} \quad |S_i| \geqslant \gamma|G_i| \tag{2}$$

Constraints differentiating PCG construction from traditional densest subgraph problems: (1) Oversized subgraphs. If a candidate densest subgraph K_i satisfies $|K_i| \geqslant \gamma |G_i|$, then $S_i = K_i$. (2) Undersized subgraphs. If $|K'|$ is smaller than the scale defined by parameter γ, the algorithm expands the result by selectively incorporating additional vertices and edges from G_i until the scale requirement is met, while preserving density properties.

$$S_i = \begin{cases} K_i & |K_i| \geqslant \gamma |G_i| \\ K_i \cup \gamma |G_i| & \text{other} \end{cases} \tag{3}$$

2.3 Heuristic merging

Following the aforementioned two steps, PCG is obtained. In this phase, the PCG is incrementally merged with newly generated subgraphs S_i to construct the entire candidate graph (ECG). The core of this process is a heuristic approximation algorithm that efficiently merges subgraphs. This procedure achieves two key objectives without recomputational overhead and two situations could be concerned: addition operation and recovery operation.

(1) Addition operation. New subgraph S_i is incorporated into the current PCG based on shared edges and vertices with existing densest k-subgraph candidates.

(2) Recovery operation. This addresses graph fragmentation from temporal window partitioning by reconstructing coherent dense subgraphs across discontinuous time intervals. The algorithm avoids recalculating density metrics during merging by leveraging precomputed topological relationships. Formal pseudocode is provided in Algorithm 2.

Algorithm 2 Heuristic approximation algorithm (HAA)

```
Input: PCG, $\gamma$.
Output: ECG.
Init ECG = $\varnothing$;
For $S_i$ in PCG,
    If $i$>1,
        If $S_i \cap S_0 \neq \varnothing$,
            If density($S_i \cap S_0$) ≥ density($S_0$),
                $S_i \leftarrow S_i \cap S_0$;
            Else $S_0 \leftarrow S_i \cup S_0$;
        Else $S_0 \leftarrow S_1$;
Return $S_0$.
```

2.4 The densest k-subgraph extraction

This phase identifies the densest k-subgraph from the ECG by optimizing the density metric. The density function density(·) is evaluated using two complementary approaches: average density maximization and k-clique detection. The average density maximization defines subgraph density as the maximum average degree. This metric prioritizes subgraphs with the highest edge-to-vertex ratio, capturing globally dense regions. k-clique detection identifies all k-vertex cliques (complete subgraphs with exactly k nodes and $(k-1)(k-2)/2$ edges), ensuring strong connectivity because every pair of nodes must be directly adjacent.

3 Optimal Density Analysis

This section provides a theoretical analysis of the accuracy lower bound of the proposed algorithm. We first derive the minimum density threshold for edge increase when adding a node to subgraphs under our method[13], then extend it to the merged global graph. Let V denote the vertex set, $V = V_1 \cup V_2 \cup \cdots \cup V_t$, where each vertex subset has nt−1 nodes. $H(\rho)$ quantifies the information loss during merging, then $H(\rho) = -[\rho \ln\rho + (1-\rho)\ln(1-\rho)]$, where $\ln\rho$ denotes the logarithm with base 2, and $1-\rho$ means density.

3.1 Minimum density threshold for node addition

Assume adding the i-th node v to the subgraph S_i. Let $H(\rho_i) \geqslant 1 - 1/i\ln n/[2k\ln(\ln n)]$ denotes the Shannon entropy function for the connectivity distribution of node v. The probability that inserting v introduces at least $(1-\rho_i)i$ edges to S_i is given by $\Pr[|E(v, S_i)| \geqslant (1-\rho_i)i]$. When $k < \ln n$ edges related to v are added to S_{k-1}, then

$$\Pr[|E(v, S_i)| \geqslant (1-\rho_i)i] = \sum_{i}^{t=(1-\rho_i)i} \binom{i}{t} 2^{-i} = 2^{(H(\rho_i)+o(1)-1)i} \tag{4}$$

Similarly, when a node v is added to the subgraph S_i, the probability that v connects with fewer than $(1-\rho_i)i$ edges is expressed as

$$\Pr[|E(v, S_i)| < (1-\rho_i)i \quad \forall v \in V_{i+1}] \leqslant [1 - 2^{(H(\rho_i)-1)i}]^{nk^{-1}} \leqslant \left[1 - \frac{2k\ln(\ln n)}{n}\right]^{nk^{-1}} \leqslant \frac{1}{\ln^2 n} \tag{5}$$

Therefore, we derive

$$\Pr[|E(v, S_i)| \geqslant (1-\rho_i)i] \geqslant 1 - \frac{1}{\ln^2 n}$$

3.2 Probability threshold for node merging across multiple subgraphs

When all the node sets $V = V_1 \cup V_2 \cup \cdots \cup V_t$ are merged, the probability for the node set union is

$$\Pr\left[|E(S, S)| \geqslant \sum_{i=0}^{k-1}(1-\rho_i)i\right] \geqslant \prod_{i=0}^{k-1} \Pr[|E(v, S_i)| \geqslant (1-\rho_i)i] \geqslant \left(1 - \frac{1}{\ln^2 n}\right)^{k-1} \geqslant \mathrm{e}^{\frac{1}{\ln n}} \quad k = 2\ln n \tag{6}$$

By the average density criterion, the optimal density is achieved when $k = 2\ln n$:

$$|E(S, S)| \geqslant \sum_{i=0}^{k-1}(1-\rho_i)i \geqslant \sum_{i=0}^{k-1} i - \sum_{i=\ln m}^{k-1} iH^i\left(1 - \frac{\ln m}{i}\right) \tag{7}$$

Therefore,

$$\text{density}(S)=\frac{|E(S,S)|}{\binom{k}{2}}\geqslant$$

$$1-\frac{1}{2}(1+o(1))\int_{0}^{\alpha}(1+x)H^{-1}\left(1+\frac{1}{1+x}\right)>0.9 \tag{8}$$

$$\alpha=\frac{k}{\ln m}-1=1+o(1),\ m=\frac{n}{2k\ln(\ln n)}$$

Thus, the density of the densest k-subgraph computed using the proposed method exhibits rigorous theoretical foundations. Under subgraph independence conditions, the achieved optimal density converges to 0.9 with high probability.

4 Experiments and Analysis

4.1 Experimental setting

Four datasets from SNAP[26] were selected: Friendster, Orkut, YouTube, and DBLP. Table 1 presents the statistics of these datasets, including the number of nodes, edges, and triangles. To validate the compatibility of the proposed method with various densest subgraph discovery algorithms, we employed two density functions: average density (maximizing edge-to-vertex ratio) and k-clique (extracting fully connected subgraphs with exactly k nodes). The size parameter k was set to 3 and 5, corresponding to complete graphs (cliques) of 3 and 5 nodes, respectively.

Table 1 Statistics of datasets

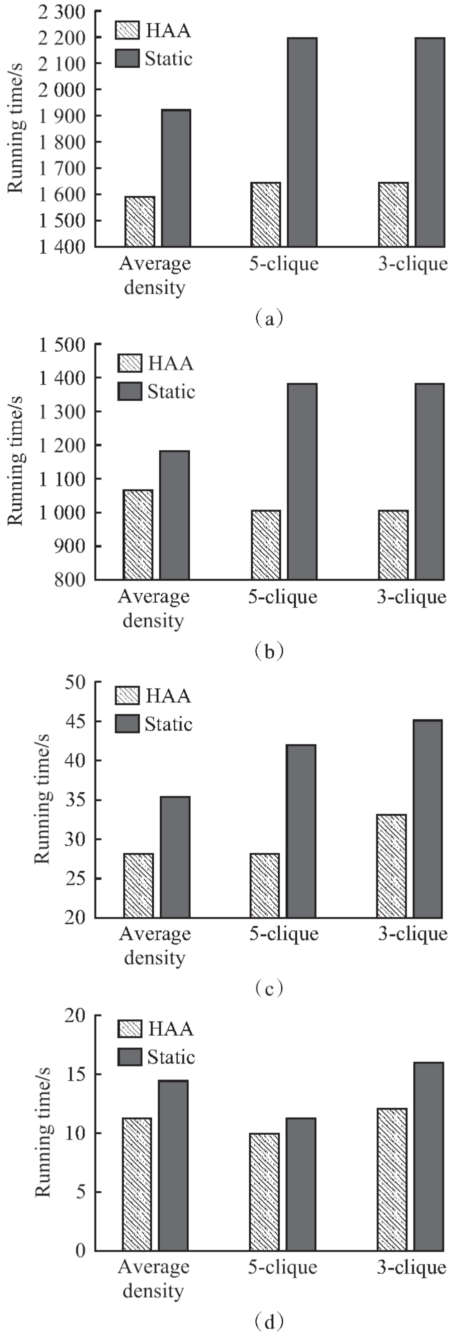
Dataset	Number of graph		
	Nodes	Edges	Triangles
Friendster	65 608 366	1 806 067 135	4 173 724 142
Orkut	3 072 441	117 185 083	627 584 181
YouTube	1 134 890	2 987 624	3 056 386
DBLP	317 080	1 049 866	2 224 385

4.2 Efficiency evaluation

The running times of the algorithms on the Friendster, Orkut, YouTube, and DBLP datasets are shown in Table 2 and Fig. 2. On the Friendster dataset, the proposed algorithm exhibits a running time of 1 568-1 640 s compared with the original algorithm (1 921-2 195 s). On the Orkut dataset, the proposed algorithm runs in 1 005-1 066 s compared with the original algorithm (1 182-1 380 s). On the YouTube dataset, the proposed algorithm has a running time of 28.02-33.03 s compared with the original algorithm (35.31-45.11 s). On the DBLP dataset, the proposed algorithm runs in 9.88-12.03 s compared to the original algorithm (11.20-15.93 s). Overall, the proposed algorithm achieves significantly lower running times than the original algorithm across different datasets, with an average reduction of 29%.

Table 2 Running time of algorithms s

Dataset	Average density		5-clique		3-clique	
	HAA	Static	HAA	Static	HAA	Static
Friendster	1 586	1 921	1 640	2 195	1 640	2 195
Orkut	1 066	1 182	1 005	1 380	1 005	1 380
YouTube	28.02	35.31	28.03	42.02	33.03	45.11
DBLP	11.21	14.43	9.88	11.20	12.03	15.93



(a)

(b)

(c)

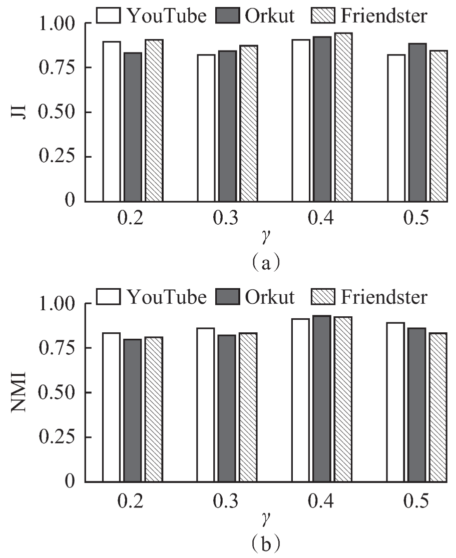
(d)

Fig. 2 Running time of various approaches on four datasets. (a)Friendster; (b)Orkut; (c)YouTube; (d)DBLP

4.3 Subgraph quality evaluation

Two commonly used metrics were employed to evaluate the quality of the densest subgraph: the Jaccard index (JI) and normalized mutual information (NMI). The JI measures similarity by calculating the classification of node pairs (edge endpoints), specifically, whether the two endpoints of an edge are assigned to the same or different subgraphs. The closer the JI is to 1, the higher the similarity between the subgraphs. NMI is an information-theoretic measure that quantifies the structural consistency of the densest subgraph. A higher NMI value indicates a more accurate partitioning of the densest subgraph. A maximum value of 1 indicates perfect structural consistency of the densest subgraph with the original graph, indicating optimal algorithm performance. Information entropy is utilized to measure the difference between the predicted and ground-truth community structures. A higher value indicates a more accurate community structure partitioning. A maximum value of 1 indicates perfect consistency with the ground-truth community structure, representing optimal algorithm performance.

The results for both metrics are shown in Fig. 3. The JI and NMI range from 0. 80 to 0. 92 and 0. 82 to 0. 94, respectively. Both ranges are close to the maximum value of 1, indicating that the structure of the densest subgraph identified by the proposed algorithm is highly consistent with that of the original graph. The proposed algorithm has been integrated with several established densest subgraph discovery algorithms. The consistent results from these integrations confirm the high scalability and stability of the algorithm.



(a)

(b)

Fig. 3 JI & NMI on the densest subgraph with various γ. (a) JI; (b) NMI

By setting the parameters to 0. 2, 0. 3, 0. 4, and 0. 5, and the densest subgraph size to 20%, 30%, 40%, and 50% of the original graph, respectively, the JI and NMI of the resulting subgraphs both exceed 0. 80. This indicates that these parameter values exhibit a relatively small impact on the stability of the proposed algorithm.

5 Conclusions

This study proposes a heuristic approximate algorithm for mining the densest *k*-subgraphs in large-scale dynamic graphs using a sliding time window. The proposed method continuously identifies the densest subgraphs that meet certain size constraints. The proposed approach reduces the computational overhead, decreases the overall running time, and improves the quality of the results by partitioning the graph into continuous subgraphs within the time window and iteratively processing them. The proposed method also resolves the memory limitation issue inherent in processing large-scale graphs that cannot be entirely loaded into memory. It is worth noting that the proposed method preserves local densest *k*-subgraphs and reduces computational effort by eliminating nodes and edges with fewer connections. The key contributions of this study are as follows:

(1) A novel heuristic algorithm that effectively mines the densest *k*-subgraphs by merging local densest subgraphs is proposed.

(2) A theoretical analysis demonstrates that the optimal density of the proposed method is 0. 9.

(3) Extensive experiments on multiple datasets confirm the effectiveness of the proposed algorithm.

The proposed heuristic approximation algorithm, combined with two density functions and multiple parameters, is validated on three large-scale datasets. The results demonstrate its effectiveness and accuracy in identifying the densest subgraphs in dynamic graphs, confirming its significant value for large-scale dynamic graph analysis.

References

[1] Bao Q, Wu Z, Chen J M, et al. Comprehensive assessment of traffic operation risk based on vehicle trajectory prediction and macro-micro indicator aggregation for road segments approaching intersections[J]. Journal of Southeast University (Natural Science Edition), 2025, 55(1): 230-238. (in Chinese)

[2] LU H X, XU H Y, LIU S C, et al. Design of an intelligent online street view big data analysis system: A case study on sky view factor extraction[J]. Journal of Southeast University (Natural Science Edition), 2025, 55 (1): 290-296. (in Chinese)

[3] LIU L, JIANG L Y, ZHANG B L. Multi-radar deployment based on improved simulated annealing with optimal neighborhood search[J]. Journal of Southeast University (Natural Science Edition), 2024, 54(5): 1322-1329. (in Chinese)

[4] DAS SARMA A, Deshpande A, Kannan R. Finding dense subgraphs in G (n, 1/2) [C]//7th International Workshop on Approximation and Online Algorithms. Copenhagen, Denmark, 2009, 5893: 98.

［5］ FOX J，NGUYEN T，SCOTT A，et al. Induced subgraph density. Ⅱ. Sparse and dense sets in cographs［J］. European Journal of Combinatorics，2025，124：104075.

［6］ XU X J，LIU H Y，LÜ X W，et al. An efficient and exact algorithm for locally *h*-clique densest subgraph discovery［J］. Proceedings of the ACM on Management of Data，2024，2（6）：1-26.

［7］ HOSSEINZADEH M M. Dense subgraphs in biological networks［C］//46th International Conference on Current Trends in Theory and Practice of Informatics（SOFSEM）. Limassol，Cyprus，2020，12011：711-719.

［8］ TU S J，STANKOVIC A，NEUMANN S，et al. Optirefine：Densest subgraphs and maximum cuts with *k* refinements［J］. Data Mining and Knowledge Discovery，2025，39（6）：82.

［9］ CALUDE C S，DINNEEN M J，HUA R. Quantum solutions for densest *k*-subgraph problems［J］. Journal of Membrane Computing，2020，2（1）：26-41.

［10］ LU Q H，SIDIROPOULOS N D，KONAR A. Densest *k*-subgraph mining via a provably tight relaxation［J］. Proceedings of the AAAI Conference on Artificial Intelligence，2025，39（12）：12291-12299.

［11］ KHANNA Y，LOUIS A. Planted models for the densest *k*-subgraph problem［C］//40th Conference on Foundations of Software Technology and Theoretical Computer Science-FSTTCS. India，2020，182：27.

［12］ BONCHI F，GARCÍA-SORIANO D，MIYAUCHI A，et al. Finding densest *k*-connected subgraphs［J］. Discrete Applied Mathematics，2021，305：34-47.

［13］ Das Sarma A，Deshpande A，Kannan R. Finding dense subgraphs in $G(n, 1/2)$［C］//Approximation and Online Algorithms. Berlin，Germany：Springer，2010：98-103.

［14］ CHEN T Y，MIYAUCHI A，TSOURAKAKIS C E. Q-DISCO：Query-centric densest subgraphs in networks with opinion information［C］//Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining. Hannover，Germany，2025：194-203.

［15］ LANCIANO T，MIYAUCHI A，FAZZONE A，et al. A survey on the densest subgraph problem and its variants［J］. ACM Computing Surveys，2024，56（8）：1-40.

［16］ DONDI R，HOSSEINZADEH M M，MAURI G，et al. Top-*k* overlapping densest subgraphs：Approximation algorithms and computational complexity［J］. Journal of Combinatorial Optimization，2021，41（1）：80-104.

［17］ DONDI R，HOSSEINZADEH M M，GUZZI P H. A novel algorithm for finding top-*k* weighted overlapping densest connected subgraphs in dual networks［J］. Applied Network Science，2021，6（1）：40.

［18］ MEGHERBI W，HADDAD M，SEBA H. DeepDense：Enabling node embedding to dense subgraph mining［J］. Expert Systems with Applications，2024，238：121816.

［19］ MA C H，FANG Y X，CHENG R，et al. A convex-programming approach for efficient directed densest subgraph discovery［C］//Proceedings of the 2022 International Conference on Management of Data. Philadelphia，PA，USA，2022：845-859.

［20］ MA C H，FANG Y X，CHENG R，et al. Efficient directed densest subgraph discovery［J］. Sigmod Record，2021，50（1）：33-40.

［21］ KOANA T，KOMUSIEWICZ C，SOMMER F. Computing dense and sparse subgraphs of weakly closed graphs［J］. Algorithmica，2023，85（7）：2156-2187.

［22］ MA C H，FANG Y X，CHENG R，et al. On directed densest subgraph discovery［J］. ACM Transactions on Database Systems，2021，46（4）：13.

［23］ DONDI R，HERMELIN D. Computing the *k* densest subgraphs of a graph［J］. Information Processing Letters，2023，179：106316.

［24］ XU Y C，MA C H，FANG Y X，et al. Efficient and effective algorithms for densest subgraph discovery and maintenance［J］. The VLDB Journal，2024，33（5）：1427-1452.

［25］ BASU S，PAUL-PENA D，QIAN K，et al. Covering a graph with dense subgraph families，via triangle-rich sets［C］//Proceedings of the 33rd ACM International Conference on Information and Knowledge Management. Boise，ID，USA，2024：109-119.

［26］ LESKOVEC J，SOSIC R. SNAP：A general-purpose network analysis and graph-mining library［J］. ACM Transactions on Intelligent Systems and Technology，2016，8（1）：1–20.

大规模动态图的*k*-最密集子图启发式近似发现算法研究

韩涛[1]，田玉玺[1]，赵建伟[1]，王森章[2]

（1.国家公共信用和地理空间信息中心，北京 100059；2.中南大学计算机学院，长沙 410083）

摘要：为了解决静态最密集子图挖掘算法处理大规模动态图效率较低问题，本文提出了一个启发式近似算法。该算法通过切分大规模动态图、构建部分*k*-最密集子图集、启发式合并子图集、挖掘*k*-最密集子图4个步骤近似计算整体图的*k*-最密集子图，大大减少大规模动态图的计算时间，同时提升结果子图质量。此算法适用于多种“密度”定义和不同边数要求，与现有静态最密集子图检测算法相结合实现算法的可扩展性和计算高效性。理论分析证明该算法提取的*k*-最密集子图的最优密度达到0.9。为了评估所提算法的性能，在4个十亿节点的数据集Friendster、Orkut、YouTube、DBLP上进行实验，结果表明所提算法在大规模和动态图上的运行时间、子图质量均优于静态方法。

关键词：*k*-最密集子图；密集特征；启发式近似算法；最优密度

中图分类号：TP311