Efficient Algorithms for Densest Subgraph Discovery

Author: Yixiang Fang, Kaiqiang Yu, Reynold Cheng,
Laks V.S. Lakshmanan, Xuemin Lin

2019 Proc. VLDB Endow

Xiaojia Xu

Problem

- **The densest subgraph discovery (DSD) problem**

Given a graph G with n vertices and m edges, the densest subgraph discovery (DSD) is the problem of discovering a "dense" subgraph from G.

[11] J. Chen and Y. Saad. Dense subgraph extraction with application to community detection. *TKDE*, 24(7):1216–1230, 2012.

[28] E. Fratkin, B. T. Naughton, D. L. Brutlag, and S. Batzoglou. Motifcut: regulatory motifs finding with maximum density subgraphs. *Bioinformatics*, 22(14):e150–e157, 2006.

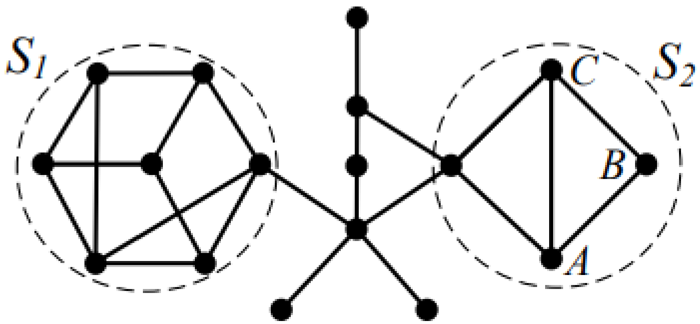
[14] E. Cohen, E. Halperin, H. Kaplan, and U. Zwick. Reachability and distance queries via 2-hop labels. *SIAM Journal on Computing*, 32(5):1338–1355, 2003.

[66] C. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarli. Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees. In *KDD*, pages 104–112. ACM, 2013.

Problem

- **The densest subgraph discovery (DSD) problem**

- **Edge-based Densest Subgraph (EDS) Problem**
- **h-clique Densest Subgraph (CDS) Problem**



(a) An example graph

[32] A. V. Goldberg. *Finding a maximum density subgraph*. UC Berkeley, 1984.

[49] M. Mitzenmacher, J. Pachocki, R. Peng, C. Tsourakakis, and S. C. Xu. Scalable large near-clique detection in large-scale networks via sampling. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 815–824. ACM, 2015.

[65] C. Tsourakakis. The k-clique densest subgraph problem. In *WWW*, pages 1122–1132, 2015.

Problem

- **This paper focus on:**

PROBLEM 1 (CDS PROBLEM [65, 49]). *Given a graph $G(V, E)$ and an h-clique $\Psi(V_\Psi, E_\Psi)$ ($h \geq 2$), return the subgraph D of $G(V, E)$, whose h-clique-density $\rho(D, \Psi)$ is the highest.*

PROBLEM 2 (PDS PROBLEM). *Given a graph $G(V, E)$ and a pattern $\Psi(V_\Psi, E_\Psi)$, return the subgraph D of $G(V, E)$, whose pattern-density $\rho(D, \Psi)$ is the highest.*

[49] M. Mitzenmacher, J. Pachocki, R. Peng, C. Tsourakakis, and S. C. Xu. Scalable large near-clique detection in large-scale networks via sampling. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 815–824. ACM, 2015.

[65] C. Tsourakakis. The k-clique densest subgraph problem. In *WWW*, pages 1122–1132, 2015.

Applications

- network science

[11] J. Chen and Y. Saad. Dense subgraph extraction with application to community detection. *TKDE*, 24(7):1216–1230, 2012.

[66] C. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarli. Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees. In *KDD*, pages 104–112. ACM, 2013.

- biological analysis

[28] E. Fratkin, B. T. Naughton, D. L. Brutlag, and S. Batzoglou. Motifcut: regulatory motifs finding with maximum density subgraphs. *Bioinformatics*, 22(14):e150–e157, 2006.

[55] B. Saha, A. Hoch, S. Khuller, L. Raschid, and X.-N. Zhang. Dense subgraphs with restrictions and applications to gene annotation graphs. In *RECOMB*, volume 6044, pages 456–472, 2010.

- graph databases

[41] R. Jin, Y. Xiang, N. Ruan, and D. Fuhry. 3-hop: a high-compression indexing scheme for reachability query. In *SIGMOD*, pages 813–826. ACM, 2009.

[71] Y. Zhang and S. Parthasarathy. Extracting analyzing and visualizing triangle k-core motifs within networks. In *ICDE*, pages 1049–1060, 2012.

- system optimization

[30] A. Gionis, F. Junqueira, V. Leroy, M. Serafini, and I. Weber. Piggybacking on social networks. *PVLDB*, 6(6):409–420, 2013.

[31] A. Gionis and C. E. Tsourakakis. Dense subgraph discovery: Kdd 2015 tutorial. In *SIGKDD*, pages 2313–2314, NY, USA, 2015. ACM.

Methods

- They propose the $(k, \Psi)-core$ by incorporating an $h-clique$ Ψ where $h \geq 2$.

- They further establish the lower and upper bounds of densities for $(k, \Psi)-cores$.

- Based on the $(k, \Psi)-cores$, they develop fast exact and approximation DSD algorithms w.r.t. $h-clique-density$.

- They generalize $h-clique-density$ to $pattern-density$ and adapt their solutions to solving DSD w.r.t. $pattern-density$.

Fundamentals

DEFINITION 1 (EDGE-DENSITY [32, 29]). *Given a graph G(V, E), its edge-density is* $\tau(G) = \frac{|E|}{|V|}$.

DEFINITION 2 (CLIQUE INSTANCE). *Given a graph* $G(V, E)$ *and an integer* $h \geq 2$, *we say a set of* h *vertices,* $S \in V$, *is an* h*-clique instance, if each pair of vertices* $u, v \in S$ *is connected by an edge.*

DEFINITION 3 (CLIQUE-DEGREE). *Given a graph* $G(V, E)$ *and an* h*-clique* Ψ, *the clique-degree of a vertex* v *in* G, *or* $deg_G(v, \Psi)$, *is the number of clique instances containing* v.

[32] A. V. Goldberg. *Finding a maximum density subgraph*. UC Berkeley, 1984.

[29] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1):30–55, 1989.

Fundamentals

DEFINITION 4 (h-CLIQUE-DENSITY [65]). *Given a graph $G(V, E)$ and an h-clique $\Psi(V_\Psi, E_\Psi)$ with $h \geq 2$, the h-clique-density of G w.r.t. Ψ is*

$$\rho(G, \Psi) = \frac{\mu(G, \Psi)}{|V|}, \tag{1}$$

where $\mu(G, \Psi)$ is the number of clique instances of Ψ in G.

[65] C. Tsourakakis. The k-clique densest subgraph problem. In *WWW*, pages 1122–1132, 2015.

Fundamentals

PROBLEM 1 (CDS PROBLEM [65, 49]). *Given a graph $G(V,E)$ and an h-clique $\Psi(V_\Psi, E_\Psi)$ $(h \geq 2)$, return the subgraph D of $G(V,E)$, whose h-clique-density $\rho(D,\Psi)$ is the highest.*

[49] M. Mitzenmacher, J. Pachocki, R. Peng, C. Tsourakakis, and S. C. Xu. Scalable large near-clique detection in large-scale networks via sampling. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 815–824. ACM, 2015.

[65] C. Tsourakakis. The k-clique densest subgraph problem. In *WWW*, pages 1122–1132, 2015.

Fundamentals

Algorithm 1: The algorithm: `Exact`.

Input: $G(V, E)$, $\Psi(V_\Psi, E_\Psi)$;
Output: The CDS $D(V_D, E_D)$;

initialize $l \leftarrow 0$, $u \leftarrow \max\limits_{v \in V} deg_G(v, \Psi)$;
initialize $\Lambda \leftarrow$all the instances of $(h$–$1)$-clique in G, $D \leftarrow \emptyset$;
while $u - l \geq \frac{1}{n(n-1)}$ **do**
  $\alpha \leftarrow \frac{l+u}{2}$;
  $V_\mathcal{F} \leftarrow \{s\} \cup V \cup \Lambda \cup \{t\}$; `// build a flow network`
  for *each vertex* $v \in V$ **do**
    add an edge $s \rightarrow v$ with capacity $deg_G(v, \Psi)$;
    add an edge $v \rightarrow t$ with capacity $\alpha|V_\Psi|$;
  for *each (h–1)-clique* $\psi \in \Lambda$ **do**
    for *each vertex* $v \in \psi$ **do**
      add an edge $\psi \rightarrow v$ with capacity $+\infty$;
  for *each (h–1)-clique* $\psi \in \Lambda$ **do**
    for *each vertex* $v \in V$ **do**
      if ψ *and* v *form an h-clique* **then**
        add an edge $v \rightarrow \psi$ with capacity 1;
  find minimum st-cut $(\mathcal{S}, \mathcal{T})$ from the flow network $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;
  if $\mathcal{S}$=$\{s\}$ **then** $u \leftarrow \alpha$;
  else $l \leftarrow \alpha$, $D \leftarrow$ the subgraph induced by $\mathcal{S} \backslash \{s\}$;
return D;

[49] M. Mitzenmacher, J. Pachocki, R. Peng, C. Tsourakakis, and S. C. Xu. Scalable large near-clique detection in large-scale networks via sampling. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 815–824. ACM, 2015.

Fundamentals

Algorithm 1: The algorithm: `Exact`.

Input: $G(V,E)$, $\Psi(V_\Psi, E_\Psi)$;
Output: The CDS $D(V_D, E_D)$;

initialize $l \leftarrow 0$, $u \leftarrow \max_{v \in V} deg_G(v, \Psi)$;
initialize $\Lambda \leftarrow$all the instances of $(h$–$1)$-clique in G, $D \leftarrow \emptyset$;
while $u - l \geq \frac{1}{n(n-1)}$ **do**
  $\alpha \leftarrow \frac{l+u}{2}$;
  $V_\mathcal{F} \leftarrow \{s\} \cup V \cup \Lambda \cup \{t\}$; `// build a flow network`
  for *each vertex* $v \in V$ **do**
    add an edge $s \rightarrow v$ with capacity $deg_G(v, \Psi)$;
    add an edge $v \rightarrow t$ with capacity $\alpha|V_\Psi|$;
  for *each (h–1)-clique* $\psi \in \Lambda$ **do**
    for *each vertex* $v \in \psi$ **do**
      add an edge $\psi \rightarrow v$ with capacity $+\infty$;
  for *each (h–1)-clique* $\psi \in \Lambda$ **do**
    for *each vertex* $v \in V$ **do**
      if *ψ and v form an h-clique* **then**
        add an edge $v \rightarrow \psi$ with capacity 1;
  find minimum st-cut $(\mathcal{S}, \mathcal{T})$ from the flow network $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;
  if $\mathcal{S}$=$\{s\}$ **then** $u \leftarrow \alpha$;
  else $l \leftarrow \alpha$, $D \leftarrow$ the subgraph induced by $\mathcal{S} \backslash \{s\}$;
return D;

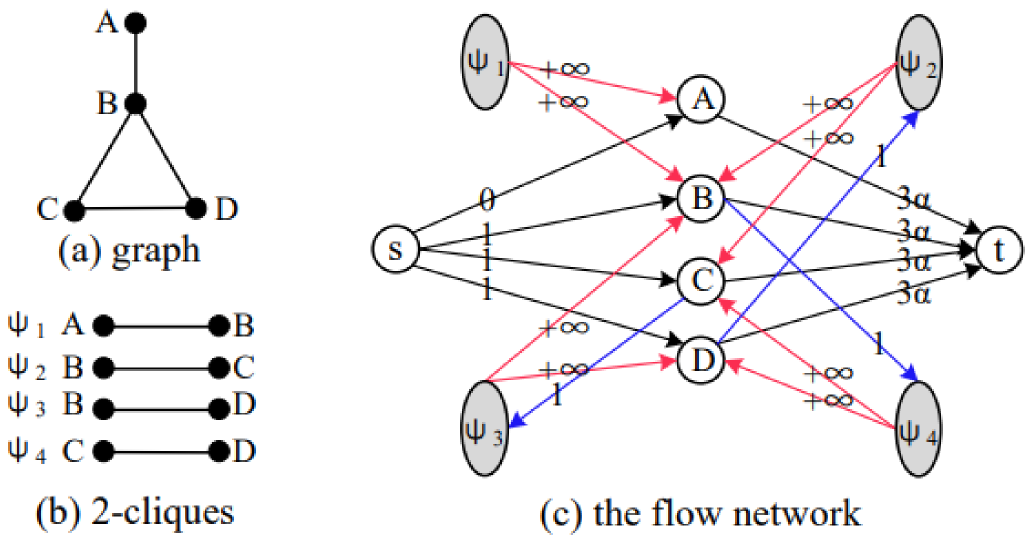


Figure 2: Illustrating the flow network (Ψ is a triangle).

LEMMA 1. *Given a graph $G(V,E)$ and an h-clique $\Psi(V_\Psi, E_\Psi)$,* `Exact` *takes* $\mathcal{O}\left(n \cdot \binom{d-1}{h-1} + (n|\Lambda| + \min(n, |\Lambda|)^3) \log n\right)$ *time and* $\mathcal{O}(n + |\Lambda|)$ *space, where Λ is set of (h–1)-clique instances in G [65].*

[65] C. Tsourakakis. The k-clique densest subgraph problem. In *WWW*, pages 1122–1132, 2015.

Fundamentals

Algorithm 2: The algorithm: `PeelApp`.

Input: $G(V, E)$, $\Psi(V_\Psi, E_\Psi)$;
Output: A subgraph S^*;
initialize $S \leftarrow G$, $S^* \leftarrow \emptyset$;
compute the clique-degree for each vertex of G;
while $S \neq \emptyset$ **do**
  $v \leftarrow$ the vertex with the minimum clique-degree in S;
  $S \leftarrow$ remove the vertex v from S;
  if $\rho(S, \Psi) > \rho(S^*, \Psi)$ **then** $S^* \leftarrow S$;
return S^*;

LEMMA 2. *Given a graph G and an h-clique $\Psi(V_\Psi, E_\Psi)$, then* `PeelApp` *takes $\mathcal{O}\left(n \cdot \binom{d-1}{h-1}\right)$ time and $\mathcal{O}(m)$ space [65].*

[65] C. Tsourakakis. The k-clique densest subgraph problem. In *WWW*, pages 1122–1132, 2015.

Methods

- They propose the $(k, \Psi) - core$ by incorporating an $h - clique$ Ψ where $h \geq 2$.

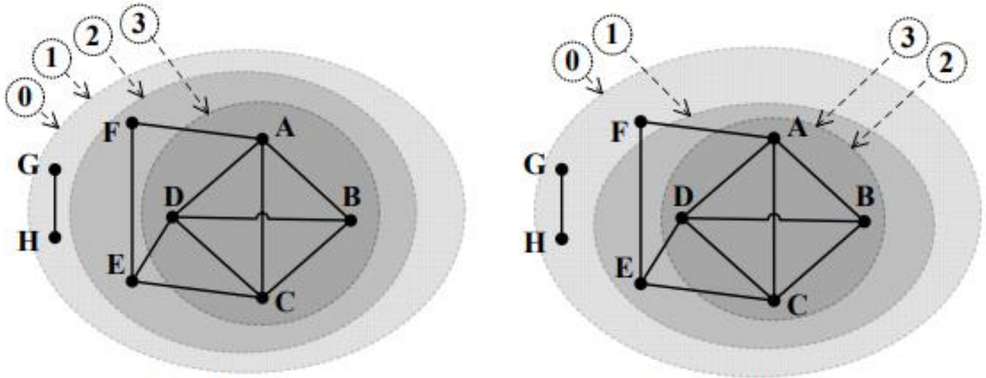
- They further establish the lower and upper bounds of densities for $(k, \Psi) - cores$.

- Based on the $(k, \Psi) - cores$, they develop fast exact and approximation DSD algorithms w.r.t. $h - clique - density$.

- They generalize $h - clique - density$ to $pattern - density$ and adapt their solutions to solving DSD w.r.t. $pattern - density$.

THE CLIQUE-BASED CORES

DEFINITION 5 (k-CORE [62, 7]). *Given a graph G and an integer k ($k \geq 0$), the k-core, denoted by $\mathcal{H}_k$, is the largest subgraph of G, such that $\forall v \in \mathcal{H}_k$, $deg_{\mathcal{H}_k}(v) \geq k$.*



(a) k-cores (b) (k, Ψ)-cores

Figure 3: k-core, and (k, Ψ)-core (Ψ is a triangle).

DEFINITION 6 ((k, Ψ)-CORE). *Given a graph G, an integer k ($k \geq 0$), and an h-clique Ψ, the (k, Ψ)-core, denoted by $\mathcal{R}_k$, is the largest subgraph of G such that $\forall v \in \mathcal{R}_k$, $deg_{\mathcal{R}_k}(v, \Psi) \geq k$.*

[62] S. B. Seidman. Network structure and minimum degree. *Social networks*, 1983.

[7] V. Batagelj and M. Zaversnik. An o(m) algorithm for cores decomposition of networks. *arXiv preprint cs/0310049*, 2003.

Algorithms

THEOREM 1. *Given a graph G and an h-clique $\Psi(V_\Psi, E_\Psi)$, let $\mathcal{R}_k$ be a (k, Ψ)-core of G. Then, the h-clique-density of $\mathcal{R}_k$ satisfies*

$$\frac{k}{|V_\Psi|} \leq \rho(\mathcal{R}_k, \Psi) \leq k_{\max}. \tag{2}$$

To prove this theorem, we develop the following lemmas.

LEMMA 3. *Given a graph G and an h-clique Ψ, the connected components of CDS D have the same clique-density.*

PROOF SKETCH. The lemma can be proved by contradiction. □

$$\rho(D \backslash U, \Psi) = \frac{\mu(D \backslash U, \Psi)}{|V_D| - |U|} > \frac{\rho_{opt}|V_D| - \rho_{opt}|U|}{|V_D| - |U|} = \rho_{opt}. \tag{3}$$

LEMMA 4. *Given a graph $G(V, E)$, an h-clique $\Psi(V_\Psi, E_\Psi)$, and the CDS $D(V_D, E_D)$, for any subset U of V_D, removing U from D will result in the removal of at least $\rho_{opt} \times |U|$ clique instances from D.*

LEMMA 5. *Given a graph G, an h-clique $\Psi(V_\Psi, E_\Psi)$, and its maximum clique-core number $k_{\max}$, we have:*

$$\rho_{opt} \leq k_{\max}. \tag{4}$$

Algorithms

Proof of THEOREM 1: The upper bound follows by Lemma 5. Let us focus on the lower bound. Let r_k be the number of vertices in $\mathcal{R}_k$. By Definition 6, since $\mathcal{R}_k$ is a (k, Ψ)-core, each vertex v of $\mathcal{R}_k$ participates in at least k clique instances. Meanwhile, each clique instance involves $|V_\Psi|$ vertices. As a result, there are at least $\frac{k \times r_k}{|V_\Psi|}$ clique instances in $\mathcal{R}_k$. Thus, we have $\rho(\mathcal{R}_k, \Psi) \geq \frac{k}{|V_\Psi|}$. □

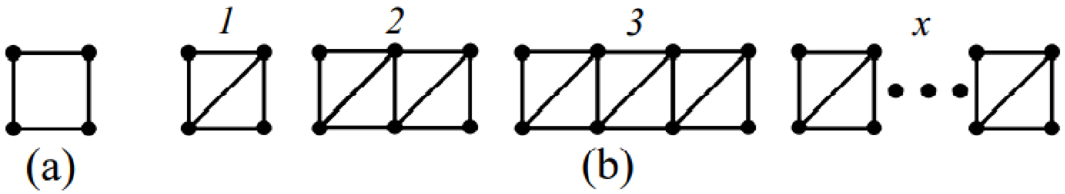


Figure 4: Illustrating the lower and upper bounds.

Algorithms

Algorithm 3: (k, Ψ)-core decomposition.

Input: $G(V, E)$, $\Psi(V_\Psi, E_\Psi)$;
Output: The clique-core number of each vertex;

initialize $core[\,]$ $\leftarrow$ an array with n entries;
for *each vertex* $v \in V$ **do** compute its clique-degree $deg_G(v, \Psi)$;
sort vertices of V in increasing order of their clique-degrees;
while V *is not empty* **do**
    $core[v] \leftarrow deg_G(v, \Psi)$ where v has the minimum clique-degree;
    for *each clique instance* ψ *containing* v **do**
        for *each vertex* u *in* ψ **do**
            if $deg_G(u, \Psi) > deg_G(v, \Psi)$ **then**
                decrease u's clique-degree;
    update G by removing v and its incident edges;
    resort the vertices in V;
return the array $core[\,]$;

[7] V. Batagelj and M. Zaversnik. An o(m) algorithm for cores decomposition of networks. *arXiv preprint cs/0310049*, 2003.
[17] M. Danisch, O. Balalau, and M. Sozio. Listing k-cliques in sparse real-world graphs. In *WWW*, pages 589–598, 2018.

LEMMA 6. *Given a graph G and an h-clique $\Psi(V_\Psi, E_\Psi)$, the core decomposition algorithm above completes in $\mathcal{O}\left(n \cdot \binom{d-1}{h-1}\right)$ time and $\mathcal{O}(m)$ space.*

Extension and Discussion

The k-clique-core can be extended to k-pattern-core by incorporating a general pattern (e.g., star, loop, etc.). Let Ψ be a pattern. Then, the (k, Ψ)-core is the largest subgraph of G, in which each vertex participates in at least k instances of Ψ. The properties of k-clique-cores also hold for k-pattern-core. Besides, for any two patterns Ψ and Ψ', if $|V_\Psi|=|V_{\Psi'}|$ and $\Psi \subseteq \Psi'$, i.e., Ψ is a subpattern of Ψ', then the (k, Ψ')-core is a subgraph of the (k, Ψ)-core. Algorithm 3 can also be extended for decomposing k-pattern-cores. We skip the details due to the space limitation.

[58] A. E. Sariyüce and A. Pinar. Fast hierarchy construction for dense subgraphs. *PVLDB*, 10(3):97–108, 2016.

[59] A. E. Sariyüce, C. Seshadhri, and A. Pinar. Local algorithms 12(1):43–56, 2018.

[60] A. E. Sariyüce, C. Seshadhri, A. Pinar, and U. V. Catalyurek. Finding the hierarchy of dense subgraphs using nucleus decompositions. In *WWW*, pages 927–937, 2015.

Methods

- They propose the $(k, \Psi) - core$ by incorporating an $h - clique$ Ψ where $h \geq 2$.

- They further establish the lower and upper bounds of densities for $(k, \Psi) - cores$.

- Based on the $(k, \Psi) - cores$, they develop fast exact and approximation DSD algorithms w.r.t. $h - clique - density$.

- They generalize $h - clique - density$ to $pattern - density$ and adapt their solutions to solving DSD w.r.t. $pattern - density$.

The Core-Based Exact Method

- Three optimization techniques for boosting the effi
  met

- **Tighter bounds on α.**

$\rho(\mathcal{R}_{k_{\max}}, \Psi) \geq \frac{k_{\max}}{|V_\Psi|} \rightarrow \rho_{opt} \geq \frac{k_{\max}}{|V_\Psi|} \rightarrow$ the lower bound of α is $\frac{k_{\max}}{|V_\Psi|}$

Lemma 5 $\rightarrow \rho_{opt} \leq k_{\max} \rightarrow$ the upper bound of α is $k_{\max}$

- **Locating the CDS in**

• *Pruning1:* The CDS is in the (k', Ψ)-core, where $k' = \lceil \rho' \rceil$ and ρ' is the highest h-clique-density of all residual graphs. The correctness directly follows Lemma 7, since $\rho' \leq \rho_{opt}$.

Since the (k', Ψ)-core may be disconnected and some connected components may be denser than others, we can further locate the CDS in a core with a larger core number, using *Pruning2*.

• *Pruning2:* For each connected component of the (k', Ψ)-core, we compute its h-clique-density. Let ρ'' be the maximum h-clique-density of these connected components. If $\lceil \rho'' \rceil > k'$, we increase k' to $k'' = \lceil \rho'' \rceil$ and the CDS is in the (k'', Ψ)-core. The correctness holds by Lemma 7, since $\rho' \leq \rho'' \leq \rho_{opt}$.

• *Pruning3:* After locating the CDS in a connected component $C(V_C, E_C)$, we can change the stopping criterion of binary search to "$u - l < \frac{1}{|V_C|(|V_C|-1)}$". Since $C(V_C, E_C)$ contains the CDS and the flow network is built using $C(V_C, E_C)$, the pruning is correct by following Algorithm 1.

- **The flow network gr**

Algorithms

Algorithm 4: The algorithm: `CoreExact`.

Input: $G(V, E)$, $\Psi(V_\Psi, E_\Psi)$;
Output: The CDS $D(V_D, E_D)$;

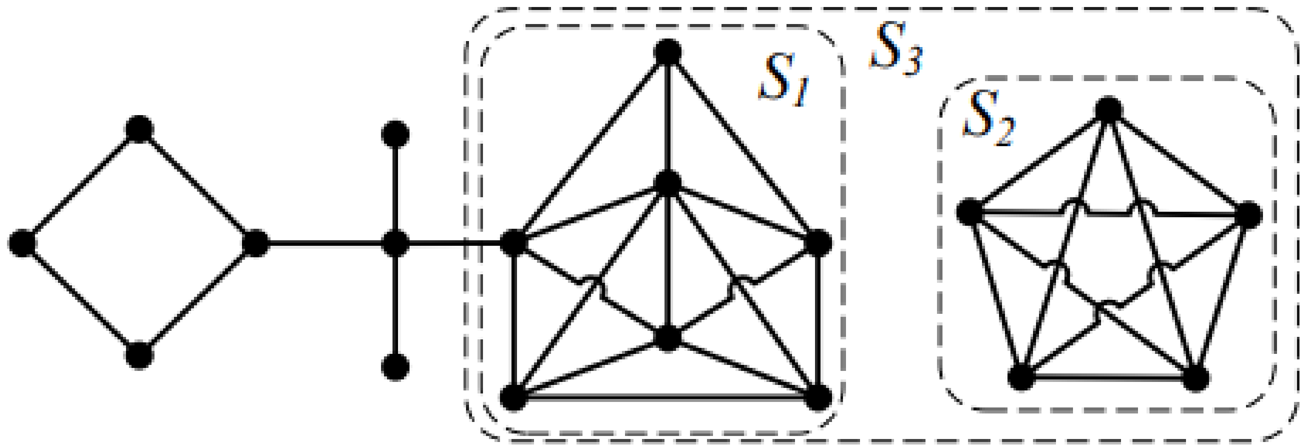
perform core decomposition using Algorithm 3;
locate the (k'', Ψ)-core using pruning criteria;
initialize $\mathcal{C} \leftarrow \emptyset$, $D \leftarrow \emptyset$, $U \leftarrow \emptyset$, $l \leftarrow \rho''$, $u \leftarrow k_{\max}$;
put all the connected components of (k'', Ψ)-core into $\mathcal{C}$;
for *each connected component* $C(V_C, E_C) \in \mathcal{C}$ **do**
    if $l > k''$ **then** $C(V_C, E_C) \leftarrow C \cap (\lceil l \rceil, \Psi)$-core;
    build a flow network $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$ by lines 5-15 of Algorithm 1;
    find minimum st-cut $(\mathcal{S}, \mathcal{T})$ from $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;
    if $\mathcal{S} = \emptyset$ **then** continue;
    while $u - l \geq \frac{1}{|V_C|(|V_C|-1)}$ **do**
        $\alpha \leftarrow \frac{l+u}{2}$;
        build $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$ by lines 5-15 of Algorithm 1;
        find minimum st-cut $(\mathcal{S}, \mathcal{T})$ from $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;
        if $\mathcal{S} = \{s\}$ **then**
            $u \leftarrow \alpha$;
        else
            if $\alpha > \lceil l \rceil$ **then** remove some vertices from C;
            $l \leftarrow \alpha$;
            $U \leftarrow \mathcal{S} \setminus \{s\}$;
    if $\rho(G[U], \Psi) > \rho(D, \Psi)$ **then** $D \leftarrow G[U]$;
return D;

Algorithms

Algorithm 4: The algorithm: `CoreExact`.

Input: $G(V,E)$, $\Psi(V_\Psi, E_\Psi)$;
Output: The CDS $D(V_D, E_D)$;

perform core decomposition using Algorithm 3;
locate the (k'', Ψ)-core using pruning criteria;
initialize $\mathcal{C} \leftarrow \emptyset$, $D \leftarrow \emptyset$, $U \leftarrow \emptyset$, $l \leftarrow \rho''$, $u \leftarrow k_{\max}$;
put all the connected components of (k'', Ψ)-core into $\mathcal{C}$;
for *each connected component* $C(V_C, E_C) \in \mathcal{C}$ **do**
    if $l > k''$ **then** $C(V_C, E_C) \leftarrow C \cap (\lceil l \rceil, \Psi)$-core;
    build a flow network $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$ by lines 5-15 of Algorithm 1;
    find minimum st-cut $(\mathcal{S}, \mathcal{T})$ from $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;
    if $\mathcal{S}=\emptyset$ **then** continue;
    while $u - l \geq \frac{1}{|V_C|(|V_C|-1)}$ **do**
        $\alpha \leftarrow \frac{l+u}{2}$;
        build $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$ by lines 5-15 of Algorithm 1;
        find minimum st-cut $(\mathcal{S}, \mathcal{T})$ from $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;
        if $\mathcal{S}=\{s\}$ **then**
            $u \leftarrow \alpha$;
        else
            if $\alpha > \lceil l \rceil$ **then** remove some vertices from C;
            $l \leftarrow \alpha$;
            $U \leftarrow \mathcal{S} \backslash \{s\}$;
    if $\rho(G[U], \Psi) > \rho(D, \Psi)$ **then** $D \leftarrow G[U]$;
return D;



The Core-Based Approximation Methods

LEMMA 8. *Given a graph G and an h-clique $\Psi(V_\Psi, E_\Psi)$, the $(k_{\max}, \Psi)$-core is a $\frac{1}{|V_\Psi|}$-approximation solution to CDS problem.*

PROOF. By Theorem 1, we have $\frac{k_{\max}}{|V_\Psi|} \leq \rho(\mathcal{R}_{k_{\max}}, \Psi) \leq k_{\max}$. Using the fact that $\rho_{opt} \leq k_{\max}$, we have

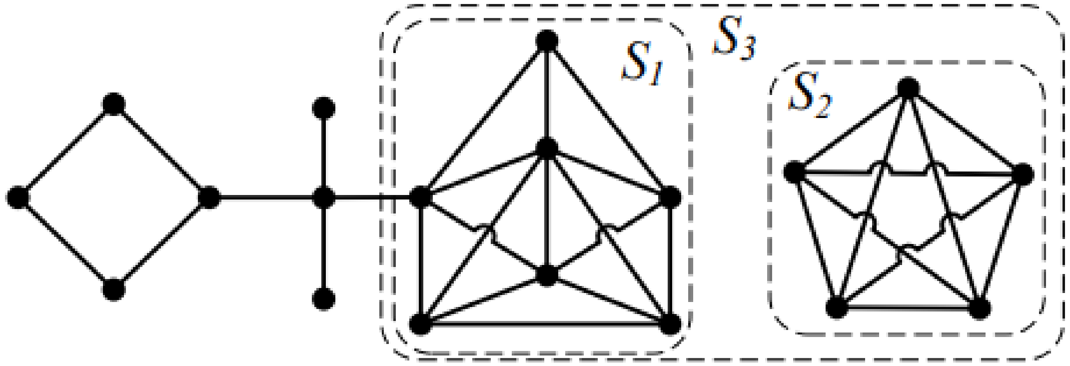
$$\frac{\rho(\mathcal{R}_{k_{\max}}, \Psi)}{\rho_{opt}} \geq \frac{k_{\max}/|V_\Psi|}{k_{\max}} = \frac{1}{|V_\Psi|}. \quad (5)$$

The lemma follows. □

The Core-Based Approximation Methods

Algorithm 5: The algorithm: `IncApp`.

Input: $G(V, E)$, $\Psi(V_\Psi, E_\Psi)$;
Output: The $(k_{\max}, \Psi)$-core;
run (k, Ψ)-core decomposition algorithm (Section 5.3);
return the $(k_{\max}, \Psi)$-core;



We remark that for h-cliques where $h \geq 3$, computing the clique-degree $deg_G(v, \Psi)$ may be costly. Instead, we replace it by an upper bound $\gamma(v, \Psi)$, which can be computed more efficiently. Specifically, we run the k-core decomposition algorithm [7], and for each vertex v in an x-core, we set $\gamma(v, \Psi) = \binom{x}{h-1}$.

[7] V. Batagelj and M. Zaversnik. An o(m) algorithm for cores decomposition of networks. *arXiv preprint cs/0310049*, 2003.

The Core-Based Approximation Methods

Algorithm 6: The algorithm: `CoreApp`.

Input: $G(V, E)$, $\Psi(V_\Psi, E_\Psi)$;
Output: The $(k_{\max}, \Psi)$-core;

for $\forall v \in V$ **do** compute $\gamma(v, \Psi)$ of $deg_G(v, \Psi)$;
sort vertices of V in decreasing order of their $\gamma(v, \Psi)$ values;
initialize W, $k_{\max} \leftarrow 0$, $S^* \leftarrow \emptyset$;
while $\max_{v \in V \setminus W} \gamma(v, \Psi) \geq k_{\max}$ **do**
  for $\forall v \in W$ **do** compute $deg_{G[W]}(v, \Psi)$;
  $k_l \leftarrow \min_{v \in W} deg_{G[W]}(v, \Psi)$, $k_u \leftarrow \max_{v \in W} deg_{G[W]}(v, \Psi)$;
  $k \leftarrow \max\{k_l, k_{\max} + 1\}$;
  while $k \leq k_u$ *and* $|W|>0$ **do**
    while $(\exists v \in W, deg_{G[W]}<k)$ **do**
      delete v from W and decrease clique-degrees;
    if $|W|>0$ **then**
      if $k>k_{\max}$ **then**
        $k_{\max} \leftarrow k$, $S^* \leftarrow G[W]$;
      $k \leftarrow k+1$;
  $W \leftarrow$ top-$(2\times|W|)$ vertices in V;
return S^*;

LEMMA 9. *The time and space complexities of* `CoreApp` *are* $\mathcal{O}\left(n \cdot \binom{d-1}{h-1}\right)$ *and* $\mathcal{O}(m)$ *respectively.*

[13] J. Cheng, Y. Ke, S. Chu, and M. T. Özsu. Efficient core decomposition in massive networks. In *ICDE*, pages 51–62, 2011.

Discussion

Parallelizability. The existing parallel k-core decomposition algorithms [50, 48, 59] can be easily extended for decomposing (k, Ψ)-cores, so our approximation solutions, which rely on the ($k_{\max}$, Ψ)-core, can be computed in parallel. Moreover, for the exact solution `CoreExact`, the main overhead comes from the step of computing the minimum st-cut. The parallel algorithms of computing the minimum st-cut have been studied extensively [42, 52], so our exact algorithm can also be easily parallelized.

[50] A. Montresor, F. De Pellegrini, and D. Miorandi. Distributed k-core decomposition. *IEEE Transactions on parallel and distributed systems*, 24(2):288–300, 2013.

[48] A. Mandal and M. Al Hasan. A distributed k-core decomposition algorithm on spark. In *International Conference on Big Data*, pages 976–981. IEEE, 2017.

[59] A. E. Sariyüce, C. Seshadhri, and A. Pinar. Local algorithms 12(1):43–56, 2018.

[42] D. B. Johnson. Parallel algorithms for minimum cuts and maximum flows in planar networks. *Journal of the ACM (JACM)*, 34(4):950–967, 1987.

[52] T. L. Pham, I. Lavallee, M. Bui, and S. H. Do. A distributed algorithm for the maximum flow problem. In *International Symposium on Parallel and Distributed Computing (ISPDC)*, pages 131–138. IEEE, 2005.

Methods

- They propose the $(k, \Psi) - core$ by incorporating an $h - clique$ Ψ where $h \geq 2$.

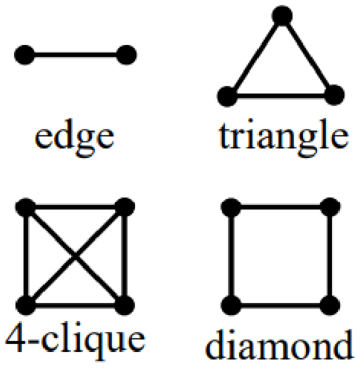
- They further establish the lower and upper bounds of densities for $(k, \Psi) - cores$.

- Based on the $(k, \Psi) - cores$, they develop fast exact and approximation DSD algorithms w.r.t. $h - clique - density$.

- They generalize $h - clique - density$ to $pattern - density$ and adapt their solutions to solving DSD w.r.t. $pattern - density$.

THE PDS PROBLEM

PROBLEM 2 (PDS PROBLEM). *Given a graph $G(V, E)$ and a pattern $\Psi(V_\Psi, E_\Psi)$, return the subgraph D of $G(V, E)$, whose pattern-density $\rho(D, \Psi)$ is the highest.*



(b) Cliques and pattern

[34] J. Hu, R. Cheng, K. C.-C. Chang, A. Sankar, Y. Fang, and B. Y. Lam. Discovering maximal motif cliques in large heterogeneous information networks. In *IEEE International Conference on Data Engineering (ICDE)*, pages 746–757. IEEE, 2019.

[70] S. Wuchty, Z. N. Oltvai, and A.-L. Barabási. Evolutionary conservation of motif constituents within the yeast protein interaction network. *Nature Genetics*, 35:176–179, 2003.

THE PDS PROBLEM

DEFINITION 7 (SUBGRAPH ISOMORPHISM). *A graph* $G(V, E)$ *is subgraph isomorphic to a pattern* $\Psi(V_\Psi, E_\Psi)$ *if there exists an injection* $\phi: V_\Psi \to V$, *such that for all* $v, v' \in V_\Psi$, *if* $(v, v') \in E_\Psi$, *then* $(\phi(v), \phi(v')) \in E$.

DEFINITION 8 (PATTERN INSTANCE). *Given a graph* $G(V, E)$ *and a pattern* $\Psi(V_\Psi, E_\Psi)$, *a subgraph* $S(V_S, E_S) \subseteq G$ *is a pattern instance of* Ψ, *if* S *is isomorphic to* Ψ.

DEFINITION 9 (PATTERN-DEGREE). *Given a graph* $G(V, E)$ *and a pattern* Ψ, *the pattern-degree of a vertex* v, *or* $deg_G(v, \Psi)$, *is the number of pattern instances of* Ψ *containing* v.

DEFINITION 10 (PATTERN-DENSITY). *Given a graph* $G(V, E)$ *and a pattern* $\Psi(V_\Psi, E_\Psi)$, *the pattern-density of* G *w.r.t.* Ψ *is* $\rho(G, \Psi) = \frac{\mu(G,\Psi)}{|V|}$, *where* $\mu(G, \Psi)$ *is the number of pattern instances of* Ψ *in* G.

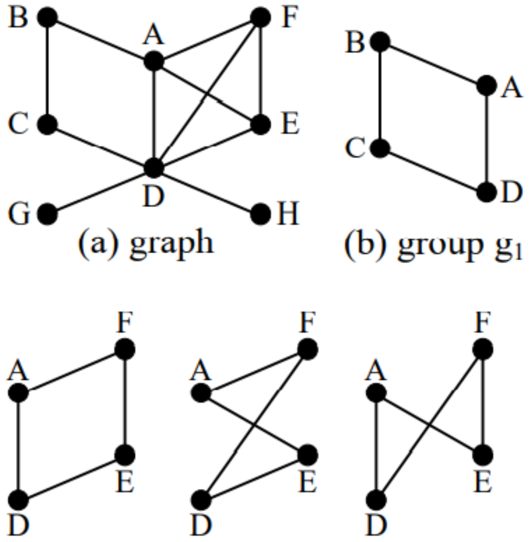
Algorithms

LEMMA 10. *Given a graph G and a pattern $\Psi(V_\Psi, E_\Psi)$, the subgraph S^* returned by* `PeelApp` *is a $\frac{1}{|V_\Psi|}$-approximation solution to the PDS problem w.r.t. pattern-density for pattern Ψ.*

THEOREM 2. *Given a graph G and a pattern $\Psi(V_\Psi, E_\Psi)$, the algorithm* `PExact` *correctly finds the PDS of G w.r.t. pattern-density of Ψ.*

[65] C. Tsourakakis. The k-clique densest subgraph problem. In *WWW*, pages 1122–1132, 2015.

Algorithms



(a) graph (b) group g_1 (c) group g_2

Algorithm 7: `construct+`(G, Ψ, α).

Input: $G(V, E)$, $\Psi(V_\Psi, E_\Psi)$, α;
Output: The flow network $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;
$\Lambda \leftarrow$ all the pattern instances of Ψ in G;
$\Lambda'=\{g_1, g_2, \cdots, g_{|\Lambda'|}\} \leftarrow$ group the pattern instances in Λ;
$V_\mathcal{F} \leftarrow \{s\} \cup V \cup \Lambda' \cup \{t\}$;
$\forall v \in V$, add an edge $s \rightarrow v$ with capacity $deg_G(v, \Psi)$;
$\forall v \in V$, add an edge $v \rightarrow t$ with capacity $\alpha|V_\Psi|$;
$\forall v \in V$, if it appears in a group $g \in \Lambda'$, add an edge $v \rightarrow g$ with capacity $|g|$;
$\forall g \in \Lambda'$, if it contains a vertex v, add an edge $g \rightarrow v$ with capacity $|g|(|V_\Psi| - 1)$;
return $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;

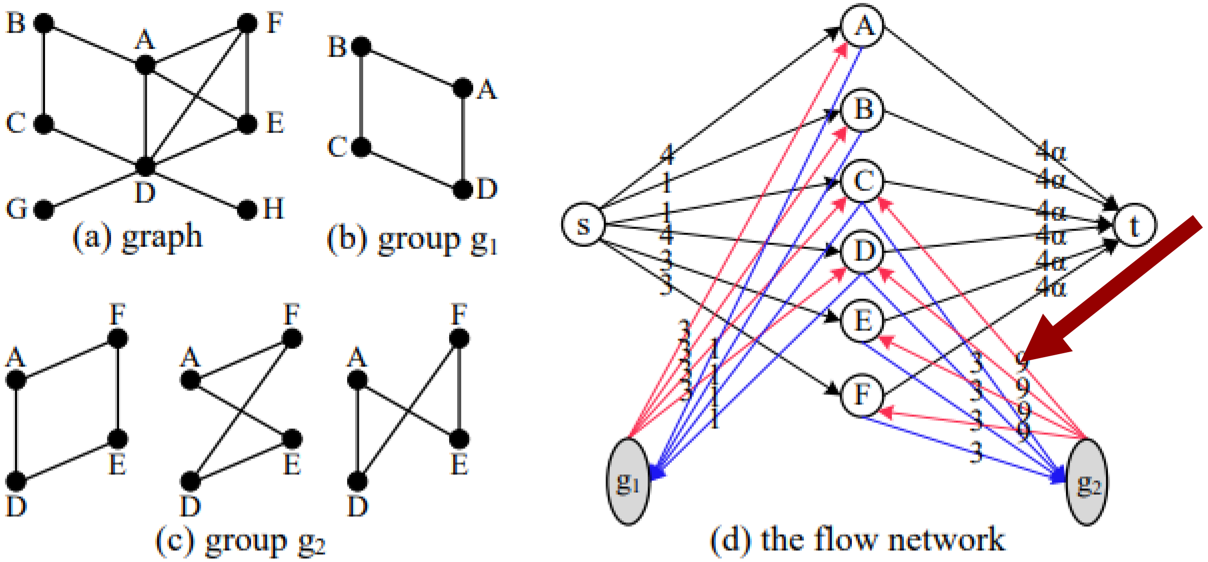
LEMMA 11. *Given a graph G, a pattern $\Psi(V_\Psi, E_\Psi)$, the flow networks built by* `PExact` *(lines 5-12) and* `construct+` *have the same capacity for their minimum st-cut.*

Algorithms

Algorithm 7: `construct+`(G, Ψ, α).

Input: $G(V, E)$, $\Psi(V_\Psi, E_\Psi)$, α;
Output: The flow network $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;

$\Lambda \leftarrow$ all the pattern instances of Ψ in G;
$\Lambda'=\{g_1, g_2, \cdots, g_{|\Lambda'|}\}\leftarrow$ group the pattern instances in Λ;
$V_\mathcal{F} \leftarrow \{s\} \cup V \cup \Lambda' \cup \{t\}$;
$\forall v \in V$, add an edge $s \rightarrow v$ with capacity $deg_G(v, \Psi)$;
$\forall v \in V$, add an edge $v \rightarrow t$ with capacity $\alpha|V_\Psi|$;
$\forall v \in V$, if it appears in a group $g \in \Lambda'$, add an edge $v \rightarrow g$ with capacity $|g|$;
$\forall g \in \Lambda'$, if it contains a vertex v, add an edge $g \rightarrow v$ with capacity $|g|(|V_\Psi| - 1)$;
return $\mathcal{F}(V_\mathcal{F}, E_\mathcal{F})$;

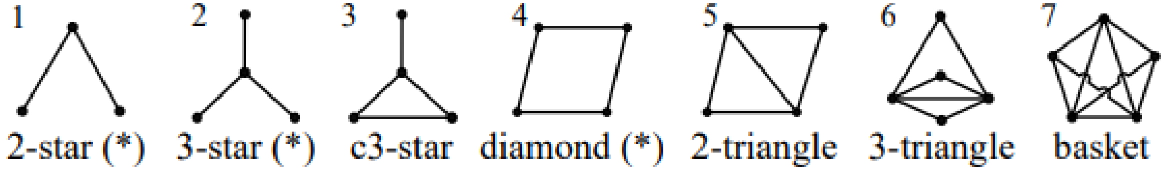


[53] M. Qiao, H. Zhang, and H. Cheng. Subgraph matching: on compression and computation. *PVLDB*, 11(2):176–188, 2017.

Evaluation

Table 2: Datasets used in our experiments.

Graph	Name	Vertices	Edges
Real small graphs (all algo.)	Yeast	1,116	2,148
	Netscience	1,589	2,742
	As-733	1,486	3,172
	Ca-HepTh	9,877	25,998
	As-Caida	26,475	106,762
Real large graphs (approx. algo.)	DBLP	425,957	1,049,866
	Cit-Patents	3,774,768	16,518,948
	Friendster	20,145,325	106,570,765
	Enwiki-2017	5,409,498	122,008,994
	UK-2002	18,520,486	298,113,762
Synthetic random graphs	SSCA	100,000	3,405,676
	ER	100,000	4,837,534
	R-MAT	100,000	2,571,986



[45] L. Lai, L. Qin, X. Lin, Y. Zhang, L. Chang, and S. Yang. Scalable distributed subgraph enumeration. *PVLDB*, 10(3):217–228, 2016.

[46] J. Leskovec, A. Singh, and J. Kleinberg. Patterns of influence in a recommendation network. In *PAKDD*, pages 380–389. Springer, 2006.

[70] S. Wuchty, Z. N. Oltvai, and A.-L. Barabási. Evolutionary conservation of motif constituents within the yeast protein interaction network. *Nature Genetics*, 35:176–179, 2003.

Evaluation

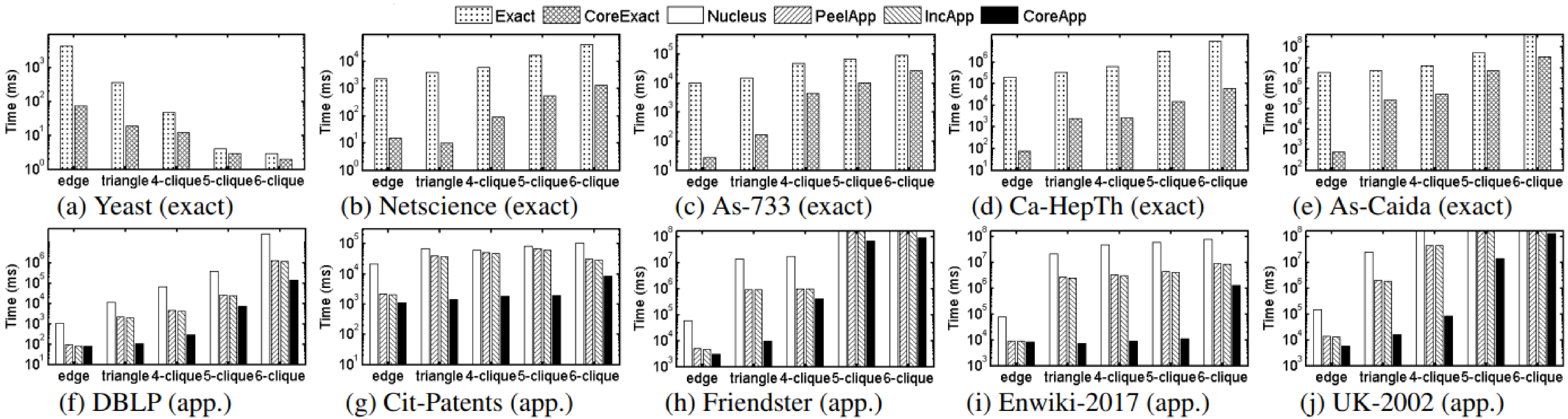


Figure 8: Efficiency of exact and approximation CDS algorithms.

Evaluation

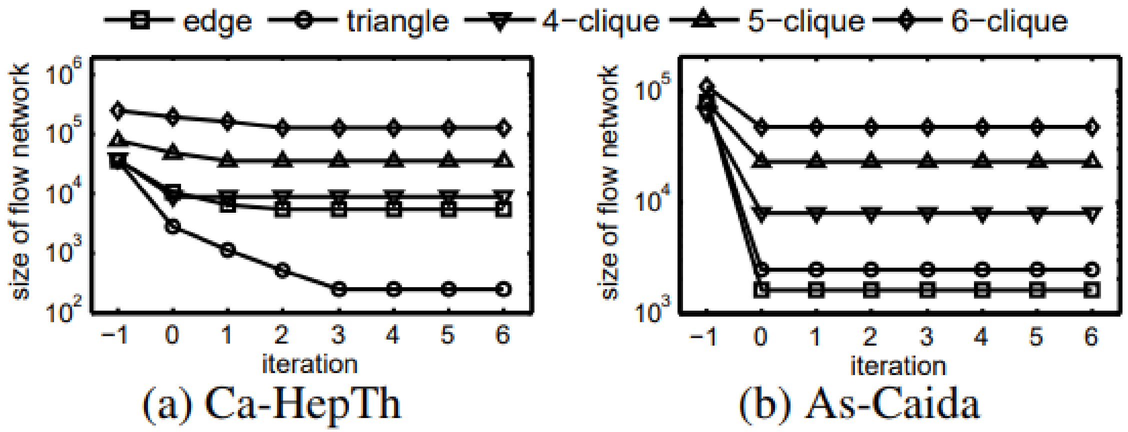


Figure 9: Flow network sizes in `CoreExact`.

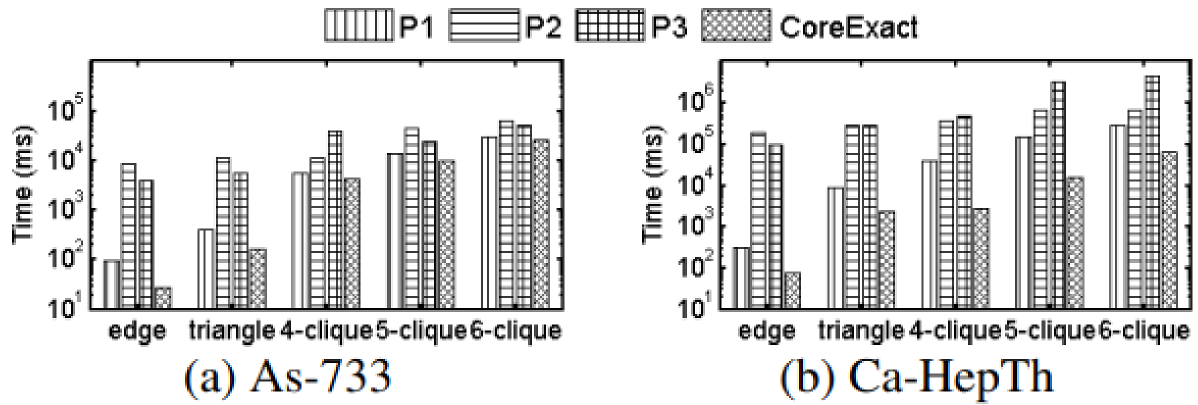


Figure 10: The effect of pruning criteria in `CoreExact`.

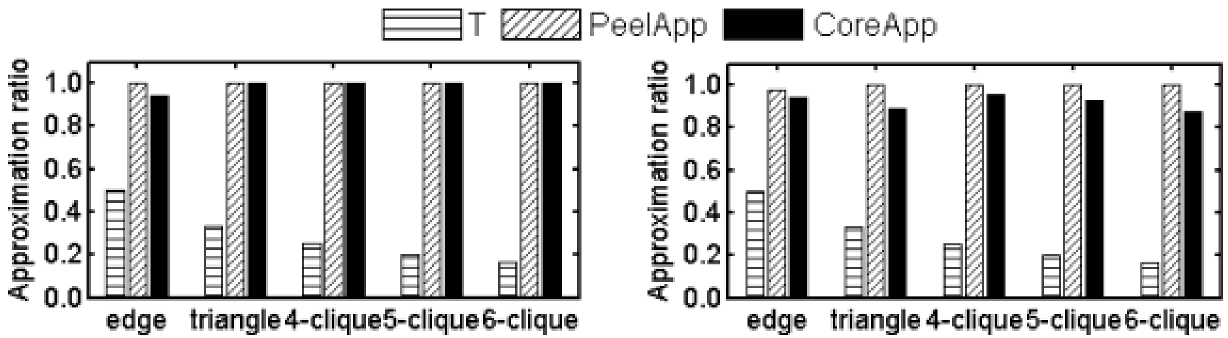
Table 3: % of time cost of core decomposition.

Dataset	edge	triangle	4-clique	5-clique	6-clique
As-733	57.14%	8.28%	0.31%	0.09%	0.04%
Ca-HepTh	69.74%	6.01%	2.32%	0.87%	0.65%

Evaluation

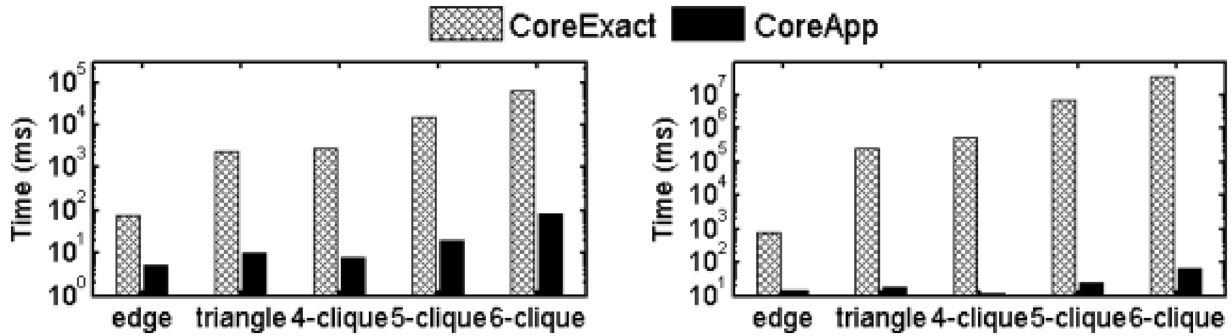
Table 4: Efficiency of `EMcore` and `CoreApp` (seconds).

Algo.	DBLP	CitPatents	FriendSter	Enwiki-2017	UK-2002
EMcore	0.091	1.132	3.143	8.543	7.543
CoreApp	0.077	1.021	2.986	8.139	5.825



(a) Netscience (b) As-Caida

Figure 11: Approximation ratio.



(a) Ca-HepTh (b) As-Caida

Figure 12: `CoreExact` **and** `CoreApp`**.**

Evaluation

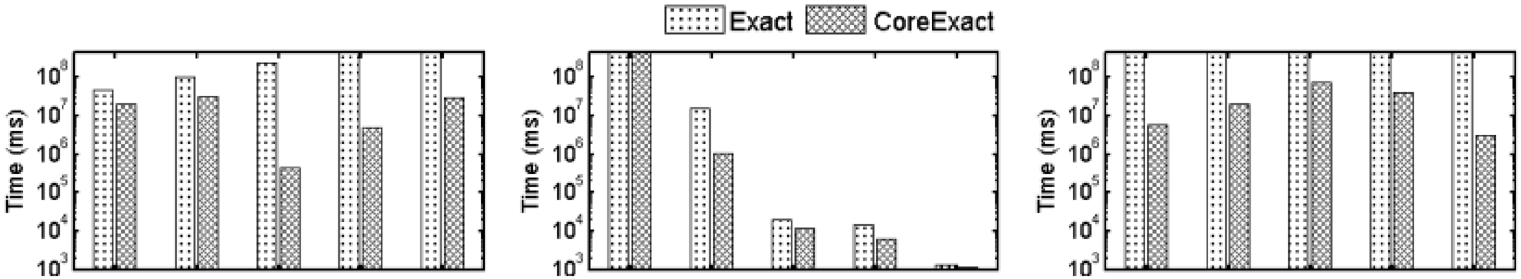
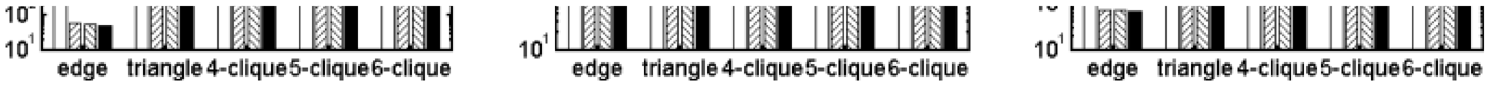


Table 5: The edge-densities and clique-densities (pattern-densities) of CDS's (PDS's).

Dataset	edge	triangle		4-clique		5-clique		6-clique		2-star		diamond	
	ρ_{opt}	ρ_{opt}	$\rho(\text{EDS},\Psi)$	ρ_{opt}	$\rho(\text{EDS},\Psi)$	ρ_{opt}	$\rho(\text{EDS},\Psi)$	ρ_{opt}	$\rho(\text{EDS},\Psi)$	ρ_{opt}	$\rho(\text{EDS},\Psi)$	ρ_{opt}	$\rho(\text{EDS},\Psi)$
S-DBLP	6	22	22	55	55	99	99	132	132	73.5	66	165	165
Yeast	3.13	2.11	0.467	0.67	0.0	0.0	0.0	0.0	0.0	111.3	18.13	20	19.2
Netscience	9.50	57.25	57.25	242.3	242.3	775.2	775.2	1938	1938	171	171	726.8	726.8
As-733	8.19	31.43	31.35	68.67	67.94	92.78	90.23	79.37	75.13	826.3	153.8	3376	437.7



(a) SSCA (b) ER (c) R-MAT

Figure 14: Efficiency of approximation CDS algorithms on random graphs.

Evaluation

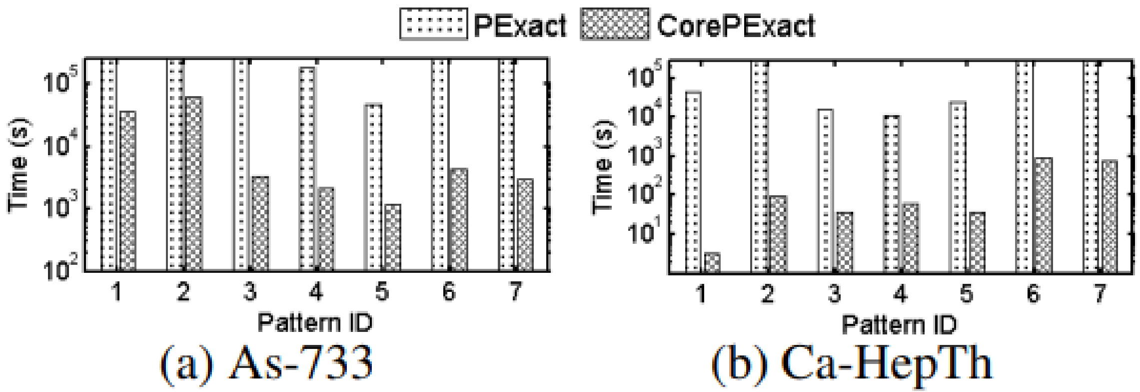


Figure 15: Efficiency of exact PDS algorithms.

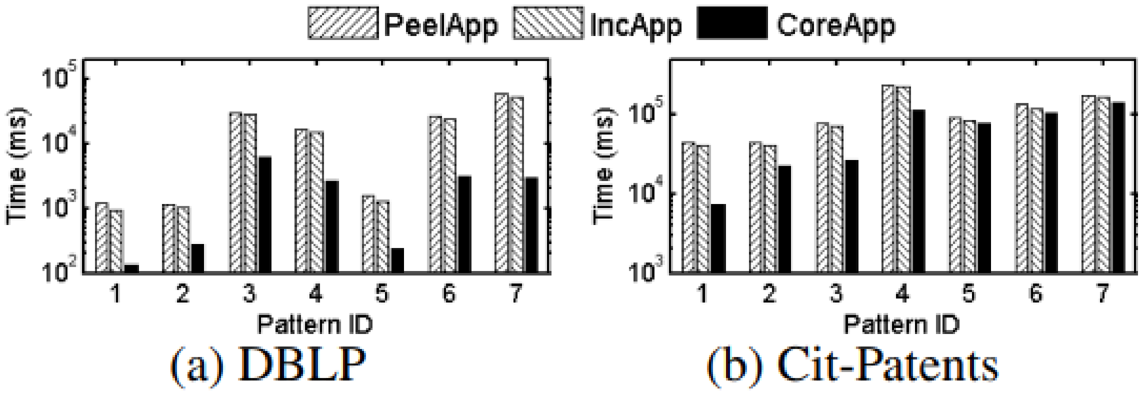
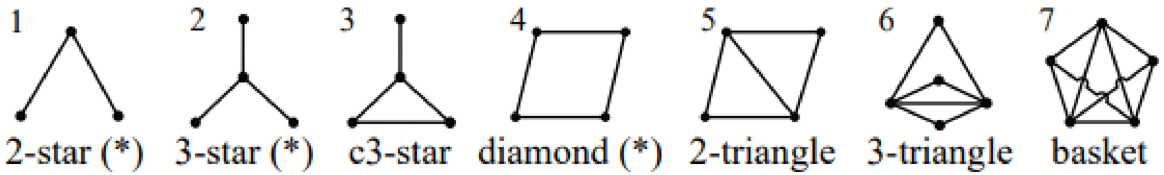


Figure 16: Efficiency of approx. PDS algorithms.



Evaluation

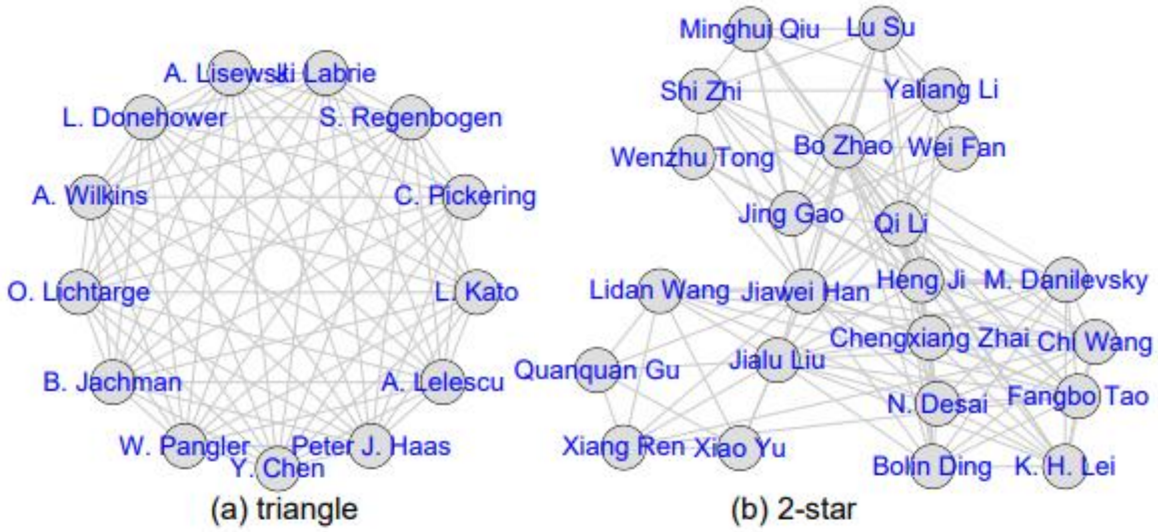


Figure 17: The densest subgraphs found in DBLP network, based on triangle and 2-star patterns.

Conclusion

- Densest subgraphs can be derived efficiently from k-cores. They extend k-core to $(k,\Psi)-core$ by incorporating an $h-clique$ Ψ.
- Based on $(k,\Psi)-cores$, they develop core-based exact and approximation solutions to the DSD problem.
- Moreover, they generalize the edge and $h-clique-density$ to $pattern-density$ and show that our solutions can be easily adapted for finding pattern-density-based densest subgraphs.
- Extensive experiments show that their exact (approximation) core-based solutions outperform existing algorithms by up to four orders (two orders) of magnitude.

THANK YOU

Xiaojia Xu