Hypercore Decomposition for Non-Fragile Hyperedges: Concepts, Algorithms, Observations, and Applications

Fanchen Bu  Geon Lee  Kijung Shin
Group Interactions are Everywhere

- **Ex 1: Collaboration** among researchers, e.g., co-authorship

---

**Hypercore decomposition for non-fragile hyperedges:**
- Concepts, algorithms, observations, and applications

**Authors:**
- Fanchen Bu
- Geon Lee
- Kijung Shin

---

**Improving the core resilience of real-world hypergraphs**

**Authors:**
- Manh Tuan Do
- Kijung Shin

---

**Datasets, tasks, and training methods for large-scale hypergraph learning**

**Authors:**
- Sunwoo Kim
- Dongjin Lee
- Yul Kim
- Jungho Park
- Taeho Hwang
- Kijung Shin

---

**Reciprocity in directed hypergraphs:**
- Measures, findings, and generators

**Authors:**
- Sunwoo Kim
- Minyoung Choe
- Jaemin Yoo
- Kijung Shin

---

**Interplay between topology and edge weights in real-world graphs:**
- Concepts, patterns, and an algorithm

**Authors:**
- Fanchen Bu
- Shinhwan Kang
- Kijung Shin

---
Group Interactions are Everywhere (cont.)

• **Ex 2:** Communication among a group, e.g., emails, online group chats
Group Interactions are Everywhere (cont.)

• **Ex 3:** Co-purchases of items on Ecommerce platforms
Hypergraphs: A Good Model for Group Interactions

**Q:** How can we represent real-world group interactions?

- **Hypergraphs** model group interactions among individuals or objects
- Each **hyperedge** is a subset of any number of nodes
- Each hyperedge indicates a **group interaction** among its members
Hypergraphs: A Good Model for Group Interactions

- **Q:** How can we represent real-world group interactions?
- **Hypergraphs** model group interactions among individuals or objects
- Each **hyperedge** is a subset of any number of nodes
- Each hyperedge indicates a **group interaction** among its members

### Authors (Nodes)
- Jure Leskovec (L)
- Jon Kleinberg (K)
- Hao Yin (Y)
- Christos Faloutsos (F)
- Daniel Huttenlocher (H)

### Publications (Hyperedges)
- $e_1$: (L, K, F) KDD’05
- $e_2$: (L, H, K) WWW’10
- $e_3$: (Y, B, G, L) KDD’17
- $e_4$: (S, R, F) VLDB’87
Hypergraph: A Good Model for Group Interactions

- **Q**: How can we represent real-world group interactions?
- **Hypergraphs** model group interactions among individuals or objects
- Each **hyperedge** is a subset of any number of nodes
- Each hyperedge indicates a group interaction among its members

<table>
<thead>
<tr>
<th>Authors (Nodes)</th>
<th>Publications (Hyperedges)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jure Leskovec (L)</td>
<td>e₁: (L, K, F) KDD’05</td>
</tr>
<tr>
<td>Jon Kleinberg (K)</td>
<td>e₂: (L, H, K) WWW’10</td>
</tr>
<tr>
<td>Hao Yin (Y)</td>
<td>e₃: (Y, B, G, L) KDD’17</td>
</tr>
<tr>
<td>Christos Faloutsos (F)</td>
<td>e₄: (S, R, F) VLDB’87</td>
</tr>
<tr>
<td>Daniel Huttenlocher (H)</td>
<td></td>
</tr>
</tbody>
</table>

F. Bu, G. Lee, and K. Shin

[ECML PKDD’23 (Journal Track)] Hypercore decomposition for non-fragile hyperedges
Hypergraphs: A Good Model for Group Interactions

• **Q:** How can we represent real-world group interactions?
• **Hypergraphs** model group interactions among individuals or objects
• Each **hyperedge** is a subset of any number of nodes
• Each hyperedge indicates a **group interaction** among its members

<table>
<thead>
<tr>
<th>Authors (Nodes)</th>
<th>Publications (Hyperedges)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jure Leskovec (L)</td>
<td>( e_1 : (L, K, F) \text{ KDD’05} )</td>
</tr>
<tr>
<td>Jon Kleinberg (K)</td>
<td>( e_2 : (L, H, K) \text{ WWW’10} )</td>
</tr>
<tr>
<td>Hao Yin (Y)</td>
<td>( e_3 : (Y, B, G, L) \text{ KDD’17} )</td>
</tr>
<tr>
<td>Christos Faloutsos (F)</td>
<td>( e_4 : (S, R, F) \text{ VLDB’87} )</td>
</tr>
<tr>
<td>Daniel Huttenlocher (H)</td>
<td></td>
</tr>
</tbody>
</table>

- \( e_1 \): Jure Leskovec, Jon Kleinberg, Christos Faloutsos
- \( e_2 \): Jure Leskovec, Hao Yin, Jon Kleinberg
- \( e_3 \): Jure Leskovec, Hao Yin, Christos Faloutsos, Daniel Huttenlocher
- \( e_4 \): Christos Faloutsos, Hao Yin, Christos Faloutsos, Daniel Huttenlocher
Hypergraphs: A Good Model for Group Interactions

• **Q:** How can we represent real-world group interactions?
• **Hypergraphs** model group interactions among individuals or objects
• Each **hyperedge** is a subset of any number of nodes
• Each hyperedge indicates a **group interaction** among its members

<table>
<thead>
<tr>
<th>Authors (Nodes)</th>
<th>Publications (Hyperedges)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jure Leskovec (L)</td>
<td>(e_1): (L, K, F) KDD’05</td>
</tr>
<tr>
<td>Jon Kleinberg (K)</td>
<td>(e_2): (L, H, K) WWW’10</td>
</tr>
<tr>
<td>Hao Yin (Y)</td>
<td>(e_3): (Y, B, G, L) KDD’17</td>
</tr>
<tr>
<td>Christos Faloutsos (F)</td>
<td>(e_4): (S, R, F) VLDB’87</td>
</tr>
<tr>
<td>Daniel Huttenlocher (H)</td>
<td></td>
</tr>
</tbody>
</table>

Publications (Hyperedges)

\[e_1\]: (L, K, F) KDD’05
\[e_2\]: (L, H, K) WWW’10
\[e_3\]: (Y, B, G, L) KDD’17
\[e_4\]: (S, R, F) VLDB’87
Hypergraphs: A Good Model for Group Interactions

Q: How can we represent real-world group interactions?

Hypergraphs model group interactions among individuals or objects
Each hyperedge is a subset of any number of nodes
Each hyperedge indicates a group interaction among its members

Authors (Nodes)
Jure Leskovec (L)  Austin Benson (B)
Jon Kleinberg (K)  David Gleich (G)
Hao Yin (Y)       Timos Sellis (S)
Christos Faloutsos (F) Nick Roussopoulos (R)
Daniel Huttenlocher (H)

Publications (Hyperedges)
\[ e_1: (L, K, F) \text{ KDD’05} \]
\[ e_2: (L, H, K) \text{ WWW’10} \]
\[ e_3: (Y, B, G, L) \text{ KDD’17} \]
\[ e_4: (S, R, F) \text{ VLDB’87} \]
Roadmap

- Concepts & Algorithms
- Observations
- Applications
- Conclusions
Important and useful $k$-cores

- The $k$-core $C_k(G)$ of a graph $G$ is the **maximum subgraph** of $G$ such that each node in $C_k(G)$ is **incident** to $\geq k$ edges
  - In (pairwise) graphs, each edge $e = (v_1, v_2)$ consists of a pair of nodes, $v_1$ and $v_2$, and we say both $v_1$ and $v_2$ are incident to this edge $e$
  - The number of edges that a node $v$ is incident to is also called the **degree** of $v$
  - It is maximum in the sense that no node or edge can be added into the $k$-core while still satisfying the conditions
Important and useful $k$-cores

- The **$k$-core** $C_k(G)$ of a graph $G$ is the **maximum subgraph** of $G$ such that each node in $C_k(G)$ is **incident** to $\geq k$ edges
  - In (pairwise) graphs, each edge $e = (v_1, v_2)$ consists of a pair of nodes, $v_1$ and $v_2$, and we say both $v_1$ and $v_2$ are incident to this edge $e$
  - The number of edges that a node $v$ is incident to is also called the **degree** of $v$
  - It is maximum in the sense that no node or edge can be added into the $k$-core while still satisfying the conditions
Important and useful $k$-cores

- The $k$-core $C_k(G)$ of a graph $G$ is the **maximum subgraph** of $G$ such that each node in $C_k(G)$ is **incident** to $\geq k$ edges
  - In (pairwise) graphs, each edge $e = (v_1, v_2)$ consists of a pair of nodes, $v_1$ and $v_2$, and we say both $v_1$ and $v_2$ are incident to this edge $e$
  - The number of edges that a node $v$ is incident to is also called the **degree** of $v$
  - It is maximum in the sense that **no node or edge can be added** into the $k$-core while still satisfying the conditions
Important and useful $k$-cores

The $k$-core $C_k(G)$ of a graph $G$ is the maximum subgraph of $G$ such that each node in $C_k(G)$ is incident to $\geq k$ edges.

- In (pairwise) graphs, each edge $e = (v_1, v_2)$ consists of a pair of nodes, $v_1$ and $v_2$, and we say both $v_1$ and $v_2$ are incident to this edge $e$.
- The number of edges that a node $v$ is incident to is also called the degree of $v$.
- It is maximum in the sense that no node or edge can be added into the $k$-core while still satisfying the conditions.
Important and useful $k$-cores

- The $k$-core $C_k(G)$ of a graph $G$ is the maximum subgraph of $G$ such that each node in $C_k(G)$ is incident to $\geq k$ edges.
  - We can obtain the $k$-core by keeping removing nodes violating the conditions.
  - Whenever a node is removed, all its incident edges are removed too.
Important and useful $k$-cores

- The $k$-core $C_k(G)$ of a graph $G$ is the **maximum subgraph** of $G$ such that each node in $C_k(G)$ is **incident** to $\geq k$ edges.

- **Applications:** community detection, anomaly detection, network visualization, important-node identification, ...
Hypercores: $k$-cores in hypergraphs

• **Q:** How can we generalize $k$-cores in hypergraphs?

• The **$k$-core** $C_k(G)$ of a graph $G$ is the maximum subgraph of $G$ such that each node in $C_k(G)$ is incident to $\geq k$ edges.

• The **$k$-hypercore** $C_k(H)$ of a hypergraph $H$ is the max. subhypergraph of $H$ such that each node in $C_k(H)$ is incident to $\geq k$ hyperedges.

• **Q:** What kind of subhypergraphs should we allow?
Hypercores: $k$-cores in hypergraphs

- The $k$-hypercore $C_k(H)$ of a hypergraph $H$ is the max. subhypergraph of $H$ such that each node in $C_k(H)$ is incident to $\geq k$ hyperedges.

- Q: What kind of subhypergraphs should be allowed in $k$-hypercores?
  - Simply removing some of the hyperedges?
Hypercores: $k$-cores in hypergraphs

• The $k$-hypercore $C_k(H)$ of a hypergraph $H$ is the max. subhypergraph of $H$ such that each node in $C_k(H)$ is incident to $\geq k$ hyperedges

• Q: What kind of subhypergraphs should be allowed in $k$-hypercores?
  • What if we remove some of the nodes from a hyperedge?
Hypercores: \( k \)-cores in hypergraphs

- The \( k \)-hypercore \( C_k(H) \) of a hypergraph \( H \) is the max. subhypergraph of \( H \) such that each node in \( C_k(H) \) is incident to \( \geq k \) hyperedges.

- **Q:** What kind of subhypergraphs should be allowed in \( k \)-hypercores?
  - **How many** nodes are allowed to leave a hyperedge?

---

F. Bu, G. Lee, and K. Shin  [ECML PKDD’23 (Journal Track)] Hypercore decomposition for non-fragile hyperedges
A Naïve Definition with Fragile Hyperedges

• Hyperedges are **fragile**, with **every** member **indispensable**
• A hyperedge is kept only if **all** the constituent nodes remain
• But this is **NOT** realistic!
Real-World Groups are Not Fragile

• Many real-world groups still remain valid even if some members leave
Definition: \((k, t)\)-Hypercores

- Given a hypergraph \(H = (V, E)\), a positive integer \(k\), and \(t \in [0, 1]\)
- The \((k, t)\)-hypercore \(C_{k,t}(H)\) of \(H\), is the maximum subhypergraph of \(H\) such that
  - Each node in \(C_{k,t}(H)\) is incident to \(\geq k\) hyperedges
  - Each hyperedge in \(C_{k,t}(H)\) contains \(\geq t\) proportion of the original constituent nodes (and at least two nodes)

Example: \(t = \frac{3}{4}\)
Related Concepts: \( t \)-Hypercoreness & \( k \)-Fraction

- The **\( t \)-hypercoreness** \( c_t(v) \) of a node \( v \) is the maximum \( k^* \) such that \( v \) is in the \( (k^*, t) \)-hypercore.

- The **\( k \)-fraction** \( f_k(v) \) of a node \( v \) is the maximum \( t^* \) such that \( v \) is in the \( (k, t^*) \)-hypercore.

- **Example:** Different \((k, t)\)-hypercore structures with different \( t \) values.

\[
\begin{align*}
\text{\( t \)-hypercoreness:} & \quad 3 & 2 & 1 \\
\text{\( t \leq 2/5 \)} & \quad & \quad & \\
\text{\( 2/5 \leq t \leq 4/7 \)} & \quad & \quad & \\
\text{\( 4/7 \leq t \leq 5/7 \)} & \quad & \quad & \\
\text{\( t > 5/7 \)} & \quad & \quad & 
\end{align*}
\]
Related Concepts: $t$-Hypercoreness & $k$-Fraction

- The $t$-hypercoreness $c_t(v)$ of a node $v$ is the maximum $k^*$ such that $v$ is in the $(k^*, t)$-hypercore.
- The $k$-fraction $f_k(v)$ of a node $v$ is the maximum $t^*$ such that $v$ is in the $(k, t^*)$-hypercore.
- Example: Different $(k, t)$-hypercore structures with different $t$ values.

$t$-hypercoreness: 3 2 1

$t \leq 2/5$

$2/5 \leq t \leq 4/7$

$4/7 \leq t \leq 5/7$

$t > 5/7$
Computation Algorithms: Find the Core

• **Peeling:** We repeatedly remove nodes and hyperedges violating the conditions, until the conditions are satisfied

• **Time complexity:** Linear to the total size of the input hypergraph
  - Specifically, $O(\sum_{e \in E} |e|)$
Computation Algorithms: Example \((t = 1)\)

- **Example:** how we obtain a \((k, t)\)-hypercore

\[(k = 1, t = 1)\)-hypercore
Computation Algorithms: Example ($t = 1$)

**Example:** how we obtain a ($k, t$)-hypercore

- Each node should be incident to $\geq k$ hyperedges
- Each hyperedge should contain $\geq t$ proportion of the original constituent nodes (and at least two nodes)
Computation Algorithms: Example ($t = 1$)

- **Example:** how we obtain a $(k, t)$-hypercore
  - Each **node** should be incident to $\geq k$ hyperedges
  - Each **hyperedge** should contain $\geq t$ proportion of the original constituent nodes (and at least two nodes)

![Diagram of hypercore](image)
Computation Algorithms: Example ($t = 1/2$)

- **Example:** how we obtain a $(k, t)$-hypercore

$(k = 1, t = 1/2)$-hypercore
Computation Algorithms: Example \((t = 1/2)\)

- **Example:** how we obtain a \((k, t)\)-hypercore
  - Each **node** should be incident to \(\geq k\) hyperedges
  - Each **hyperedge** should contain \(\geq t\) proportion of the original constituent nodes (and at least two nodes)

\[ k = 2 \]
Computation Algorithms: Example ($t = 1/2$)

- **Example:** how we obtain a $(k, t)$-hypercore
  - Each **node** should be incident to $\geq k$ hyperedges
  - Each **hyperedge** should contain $\geq t$ proportion of the original constituent nodes (and at least two nodes)
Computation Algorithms: Example ($t = 1/2$)

- **Example:** how we obtain a $(k, t)$-hypercore
  - Each **node** should be incident to $\geq k$ hyperedges
  - Each **hyperedge** should contain $\geq t$ proportion of the original constituent nodes (and at least two nodes)

$$\frac{6}{8} > \frac{1}{2}$$
Computation Algorithms: Example \((t = 1/2)\)

- **Example:** how we obtain a \((k, t)\)-hypercore
  - Each **node** should be incident to \(\geq k\) hyperedges
  - Each **hyperedge** should contain \(\geq t\) proportion of the original constituent nodes (and at least two nodes)

\[
\begin{align*}
\text{e}_1 & \quad \text{e}_2 \\
6 & > \frac{1}{2} \\
7 & > \frac{1}{2} \\
\text{e}_3 & \quad \text{e}_4
\end{align*}
\]
Computation Algorithms: Example \((t = 1/2)\)

**Example:** how we obtain a \((k, t)\)-hypercore

- Each **node** should be incident to \(\geq k\) hyperedges
- Each **hyperedge** should contain \(\geq t\) proportion of the original constituent nodes (and at least two nodes)
**Example:** how we obtain a \((k, t)\)-hypercore

- Each **node** should be incident to \(\geq k\) hyperedges
- Each **hyperedge** should contain \(\geq t\) proportion of the original constituent nodes (and at least two nodes)
Computation Algorithms: Example ($t = 1/2$)

• **Example:** how we obtain a $(k, t)$-hypercore
  
  • Each **node** should be incident to $\geq k$ hyperedges
  
  • Each **hyperedge** should contain $\geq t$ proportion of the original constituent nodes (and at least two nodes)
Computation Algorithms: Example \((t = 1/2)\)

- **Example:** how we obtain a \((k, t)\)-hypercore
  - Each **node** should be incident to \(\geq k\) hyperedges
  - Each **hyperedge** should contain \(\geq t\) proportion of the original constituent nodes (and at least two nodes)
**Computation Algorithms: Example \((t = 1/2)\)**

- **Example:** how we obtain a \((k, t)\)-hypercore
  - Each **node** should be incident to \(\geq k\) hyperedges
  - Each **hyperedge** should contain \(\geq t\) proportion of the original constituent nodes (and at least two nodes)
Roadmap

- Concepts & Algorithms
- Observations
- Applications
- Conclusions
Real-World Hypergraph Datasets

- **Datasets:** 14 real-world hypergraphs from 6 different domains
  - # nodes: 143 – 2.3M
  - # hyperedges: 1,047 – 8.6M
- **Source:** [https://www.cs.cornell.edu/~arb/data/](https://www.cs.cornell.edu/~arb/data/)

| Dataset          | |V| | |E| | max./avg. d(v) | max./avg. | |e|
|------------------|---|---|---|---|---|---|---|
| coauth-DBLP      | 1,831,126 | 2,169,663 | 846 / 4.06 | 25 / 3.42 |
| coauth-Geology   | 1,087,111 | 908,516   | 716 / 3.21  | 25 / 3.84 |
| NDC-classes      | 1,149     | 1,047     | 221 / 5.57  | 24 / 6.11 |
| NDC-substances   | 3,438     | 6,264     | 578 / 14.51 | 25 / 7.96 |
| contact-high     | 327       | 7,818     | 148 / 55.63 | 5 / 2.33 |
| contact-primary  | 242       | 12,704    | 261 / 126.98| 5 / 2.42 |
| email-Enron      | 143       | 1,457     | 116 / 31.43 | 18 / 3.09 |
| email-Eu         | 979       | 24,399    | 910 / 86.93 | 25 / 3.49 |
| tags-ubuntu      | 3,021     | 145,053   | 12,930 / 164.56| 5 / 3.43 |
| tags-math        | 1,627     | 169,259   | 13,949 / 363.80| 5 / 3.50 |
| tags-SO          | 49,945    | 5,517,054 | 520,488 / 427.77| 5 / 3.87 |
| threads-ubuntu   | 99,054    | 115,987   | 2,170 / 2.97 | 14 / 2.31 |
| threads-math     | 153,806   | 555,323   | 11,358 / 9.08 | 21 / 2.61 |
| threads-SO       | 2,321,751 | 8,589,420 | 34,925 / 9.75 | 25 / 2.64 |

- **Coauth:** Co-authorship
- **NDC:** National Drug Code
- **Tags and threads** are collected from different sub-sites on an online Q&A platform [https://stackexchange.com/](https://stackexchange.com/)

F. Bu, G. Lee, and K. Shin [ECML PKDD’23 (Journal Track)] Hypercore decomposition for non-fragile hyperedges
Real-World Hypergraph Datasets

- **Datasets**: 14 real-world hypergraphs from 6 different domains
  - # nodes: 143 – 2.3M
  - # hyperedges: 1,047 – 8.6M

- **Source**: [https://www.cs.cornell.edu/~arb/data/](https://www.cs.cornell.edu/~arb/data/)

<table>
<thead>
<tr>
<th>Domain</th>
<th>Nodes</th>
<th>Hyperedges</th>
</tr>
</thead>
<tbody>
<tr>
<td>coauth</td>
<td>Researchers</td>
<td>Publications</td>
</tr>
<tr>
<td>NDC</td>
<td>Classes/Substances</td>
<td>Drugs</td>
</tr>
<tr>
<td>contact</td>
<td>People</td>
<td>Communications</td>
</tr>
<tr>
<td>email</td>
<td>Senders/Receivers</td>
<td>Emails</td>
</tr>
<tr>
<td>tags</td>
<td>Tags</td>
<td>Questions</td>
</tr>
<tr>
<td>threads</td>
<td>Users</td>
<td>Threads</td>
</tr>
</tbody>
</table>

- **Coauth**: Co-authorship
- **NDC**: National Drug Code
- **Tags and threads** are collected from different sub-sites on an online Q&A platform [https://stackexchange.com/](https://stackexchange.com/)
Observation 1: Domain-Based Patterns of Hypercore Sizes

- **Observation**: Real-world hypergraphs in the same domain have similar patterns of hypercore sizes with different $k$ and $t$ values.

- **Color of each pixel**: The normalized size of the $(k, t)$-hypercore.

---

Co-authorship

Contact

Email

Threads
Observation 1: Pairwise Distance

- We define a **distance metric** to compare the patterns of hypercore sizes between different hypergraphs, which can be interpreted as the average **pixel-level difference** between the plots in the previous page.
Observation 1: Pairwise Distance

- We define a **distance metric** to compare the patterns of hypercore sizes between different hypergraphs, which can be interpreted as the average **pixel-level difference** between the plots in the previous page.
Observation 2: Heavy-Tailed Hypercoreness Distributions

- **Observation:** In real-world hypergraphs, \( t \)-hypercoreness follows heavy-tailed distributions regardless of \( t \). In some datasets, the \( t \)-hypercoreness strongly follows a **power law**.

- **Left:** **Red colors** indicates heavy-tailed distributions.
Observation 2: Heavy-Tailed Hypercoreness Distributions

• **Observation:** In real-world hypergraphs, \( t \)-hypercoreness follows heavy-tailed distributions regardless of \( t \). In some datasets, the \( t \)-hypercoreness strongly follows a power law.

• **Left:** Red colors indicates heavy-tailed distributions

![Hypercoreness Distribution Graphs](image-url)
Observation 2: Heavy-Tailed Hypercoreness Distributions

- **Observation:** In real-world hypergraphs, $t$-hypercoreness follows heavy-tailed distributions regardless of $t$. In some datasets, the $t$-hypercoreness strongly follows a **power law**.

- **Right:** Red dashed lines are reference power-law fitting lines.
Observation 2: Heavy-Tailed Hypercoreness Distributions

- **Observation:** In real-world hypergraphs, $t$-hypercoreness follows heavy-tailed distributions regardless of $t$. In some datasets, the $t$-hypercoreness strongly follows a power law.

- **Right:** Red dashed lines are reference power-law fitting lines.
Observation 3: $t$-Hypercoreness is Different

- **Observation**: $t$-Hypercoreness is statistically different from other existing centrality measures
  - **Meaning**: $t$-Hypercoreness can provide unique insights of a hypergraph
- **Observation**: Even for the same hypergraph, with varying $t$ values, the $t$-hypercoreness can be statistically different from each other
  - **Meaning**: Using different $t$ values can provide us different insights
Roadmap

- Concepts & Algorithms
- Observations
- Applications
- Conclusions
Application 1: Influential-Node Identification

- **Spoiler:** In real-world hypergraphs, $t$-hypercoreness with a proper $t$ value identifies influential nodes well.

- **Set-up:** We use a widely-used epidemic model, the SIR model.
  - A single initially infected node
  - Infected nodes can infect susceptible nodes in the same hyperedge
  - Infected nodes have some probability to recover (and become immune)
  - The process terminates with only susceptible and recovered nodes

![SIR Model Diagram]
Application 1: Influential-Node Identification

- **Spoiler:** In real-world hypergraphs, $t$-hypercoreness with a proper $t$ value identifies influential nodes well.

- **Influence:** We measure the influence of each node $v$ as the number of ever-infected nodes when $v$ is the initially infected node.
  - **Ever-infected nodes:** the nodes in the recovered state in the end.
  - The influence of each node is measured by simulations.

In real-world hypergraphs, $t$-hypercoreness with a proper $t$ value identifies influential nodes well. Influence: We measure the influence of each node $v$ as the number of ever-infected nodes when $v$ is the initially infected node. Ever-infected nodes: the nodes in the recovered state in the end. The influence of each node is measured by simulations.
Application 1: Results

• **Metric:** For each considered measure, we compute the Pearson’s $r$ between the values of the **measure** and the **influences** of the nodes
  - The higher Pearson’s $r$, the better

• **Result summary:** $t$-Hypercoreness with a proper $t$ usually has **high correlation** with the influences of nodes

• **Dataset:** coauth-DBLP
Application 1: Results

• **Dataset:** email-Eu

\[ R^2 = 0.96 \]

\[ R^2 = 0.69 \]

\[ R^2 = 0.73 \]

\[ R^2 = 0.79 \]

\[ R^2 = 0.67 \]

\[ R^2 = 0.88 \]

F. Bu, G. Lee, and K. Shin  [ECML PKDD’23 (Journal Track)] Hypercore decomposition for non-fragile hyperedges
Application 2: Dense Substructure Discovery

• **Spoiler:** $(k, t)$-Hypercores can be used to efficiently and effectively find dense substructures

• **Set-up:** We consider a vertex cover problem
  • **Given:** A hypergraph $H$, # nodes to choose $k_c$, and the cover threshold $t_c$
  • **Aim to:** Choose $k_c$ nodes to maximize # covered hyperedges
  • A hyperedge is **covered** if $\geq t_c$ proportion of its constituent nodes are chosen

• **Considered methods:**
  • $t_c$-Hypercoreness: The $k_c$ nodes with the highest $t_c$-hypercoreness
  • Degree: The $k_c$ nodes with the highest degree
  • Greedy: Greedily increases the number of covered hyperedges
Application 2: Results

• We vary $k_c$ from 10 to 100 and vary $t_c \in \{0.6, 0.7, 0.8\}$
• The performance of the degree method is used as the reference one
• **Result summary:** Overall, $t_c$-hypercoreness outperforms the other two methods, i.e., covers more hyperedges
• Below are the results averaged over all the datasets
Application 3: Hypergraph Vulnerability Detection

- **TL;DR:** The concept of \((k, t)\)-hypercores can be used to detect vulnerability in hypergraphs.
- We consider an **optimization problem** on hypergraphs using the concept of \((k, t)\)-hypercores.
- We aim to remove a small number of nodes from a given hypergraph so that the size of a \((k, t)\)-hypercore is minimized.
  - Such nodes are **vulnerable nodes** in the hypergraph.
  - We can pay attention to and **protect** such nodes in real-world applications.
Roadmap

- Concepts & Algorithms
- Observations
- Applications
- Conclusions
Conclusions

• Our contributions are summarized as follows:

✓ Novel concepts generalizing $k$-cores to hypergraphs, with

✓ Theoretical properties and practical computational algorithms

✓ Various observations utilizing the proposed concepts

✓ Extensive applications showing the usefulness of the proposed concepts

Code: bit.ly/hypercore_code