Hypercore Decomposition for Non-Fragile Hyperedges

Fanchen Bu, Geon Lee, and Kijung Shin
{boqvezen97, geonlee0325, kijungs}@kaist.ac.kr
Code and Data: *bit.ly/hypercore_code (or scan the QR code)*

Summary

- **Novel concepts:** **Generalized k-cores in hypergraphs with non-fragile hyperedges**
- **Computational algorithms:** **The proposed concept can be computed efficiently**
- **Various observations:** **Insights into hypergraphs analysis**
  - **Within-domain similarity:** Real-world hypergraphs in the same domain (i.e., collected from similar scenarios) share similar patterns w.r.t the proposed concepts
  - **Universal statistical patterns:** Real-world hypergraphs across different domains exhibit specific statistical patterns w.r.t the proposed concepts
- **Extensive applications:** **Practical usefulness of the proposed concepts**
  - **Influential-node identification, dense substructure discovery, and hypergraph vulnerability detection**

Background: Group interactions and hypergraphs

- **Real-world group interactions are everywhere**
- **Ex 1) Co-authorship:** **Collaboration among researchers**

Hypercore decomposition for non-fragile hyperedges: concepts, algorithms, observations, and applications
Fanchen Bu[1] · Geon Lee[2] · Kijung Shin[1,2]

Microsoft Academic Graph

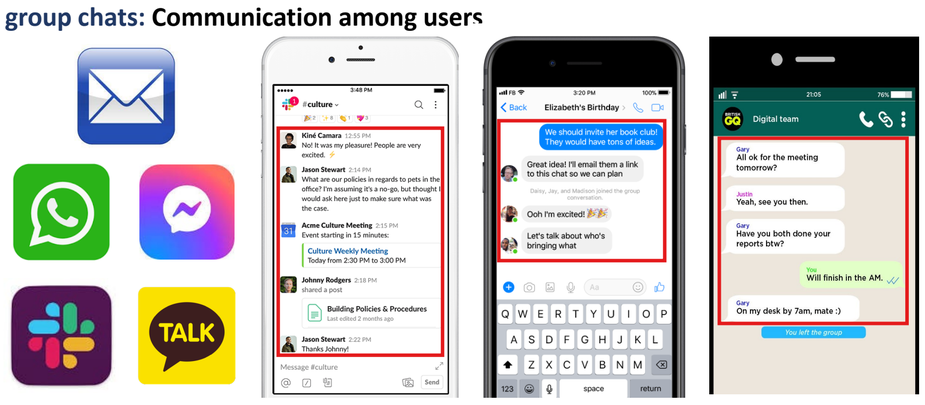
Improving the core resilience of real-world hypergraphs
Manh Tuan Do[1] · Kijung Shin[1,2]

Reciprocity in directed hypergraphs: measures, findings, and generators
Sunwoo Kim[1] · Minyoung Choe[1] · Jaemin Yoo[3] · Kijung Shin[1,2]

Datasets, tasks, and training methods for large-scale hypergraph learning
Sunwoo Kim[1] · Dongjin Lee[2] · Yul Kim[3] · Jungho Park[3] · Taeho Hwang[3] · Kijung Shin[1,2]

Interplay between topology and edge weights in real-world graphs: concepts, patterns, and an algorithm
Fanchen Bu[1] · Shinhwan Kang[2] · Kijung Shin[1,2]

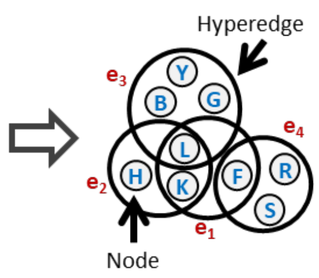
- **Ex 2) Online group chats:** **Communication among users**



- **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**

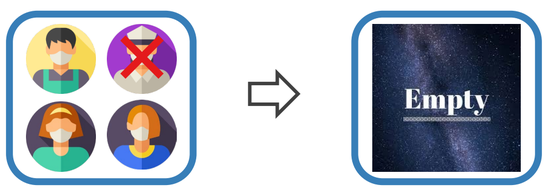
Authors (Nodes)	
Jure Leskovec (**L**)	Austin Benson (**B**)
Jon Kleinberg (**K**)	David Gleich (**G**)
Hao Yin (**Y**)	Timos Sellis (**S**)
Christos Faloutsos (**F**)	Nick Roussopoulos (**R**)
Daniel Huttenlocher (**H**)	

Publications (Hyperedges)
- e_1: (L, K, F) KDD'05
- e_2: (L, H, K) WWW'10
- e_3: (Y, B, G, L) KDD'17
- e_4: (S, R, F) VLDB'87



Generalizing k-cores in hypergraphs

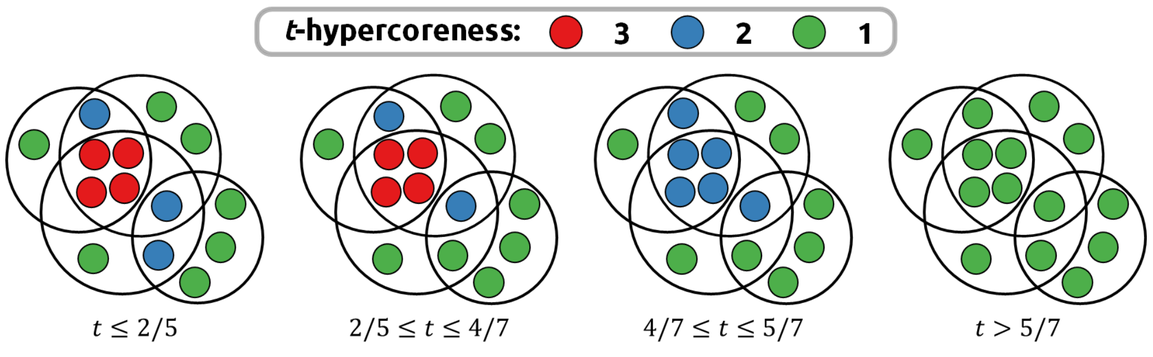
- The **k-core** $C_k(G)$ is the **maximum subgraph** of G such that each node in $C_k(G)$ is **incident** to $\geq k$ edges
  - Obtain a k-core by keeping **removing nodes violating the conditions**
  - Whenever a node is removed, **all its incident edges** are removed too
- **A naïve generalization:** The **k-hypercore** $C_k(H)$ of a hypergraph H is the **maximum subhypergraph** of G such that (1) each node in $C_k(H)$ is **incident** to $\geq k$ hyperedges, and (2) each remaining hyperedge contains all its original constituent nodes
  - Whenever a node is removed, **all its incident hyperedges** are removed too
  - Fragile hyperedges are assumed: **NOT realistic** and **less meaningful structures**



Every node has **the same hypercoreness!**

Proposed concepts: Hypercores with non-fragile hyperedges

- **Motivation:** **Group interactions can still be valid when some members leave**
  - **Group chats** are not dismissed when some users leave the group
  - **Online shopping carts** are not emptied when some items are deleted
- Given a hypergraph $H = (V, E)$, a positive integer k, and $\boldsymbol{t} \in [\mathbf{0}, \mathbf{1}]$
- 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
- The **t-hypercoreness** $c_t(v)$ of a node v is the maximum k^* such that $v \in C_{k^*,t}(H)$
- The **k-fraction** $f_k(v)$ of a node v is the maximum t^* such that $v \in C_{k,t^*}(H)$
- **Example:** Different (k,t)-hypercore structures with **different t values**



- **Computation:** Compute a (k,t)-hypercore similarly to how we compute a k-core
  - Keep **removing nodes and hyperedges violating the conditions**
  - Whenever a node is removed, we check **each of it incident hyperedge**: if the proportion of remaining constituent nodes falls below t, we remove the hyperedge
  - **Time complexity:** Linear to the total size of the input hypergraph ($O(\sum_{e \in \mathbf{E}} |e|)$)

Real-world hypergraph datasets

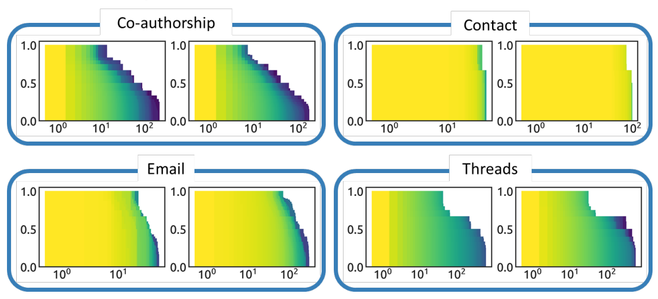
- **Datasets:** **14 real-world hypergraphs from 6 different domains**
  - **# nodes:** 143 – 2.3M
  - **# hyperedges:** 1,047 – 8.6M

Dataset	$\lvert V\rvert$	$\lvert E\rvert$	max./avg. $d(v)$	max./avg. $\lvert e\rvert$
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	520,468 / 427.77	5 / 3.87
threads-ubuntu	90,054	115,987	2,170 / 2.97	14 / 2.31
threads-math	153,806	535,323	11,358 / 9.08	21 / 2.61
threads-SO	2,321,751	8,589,420	34,925 / 9.75	25 / 2.64

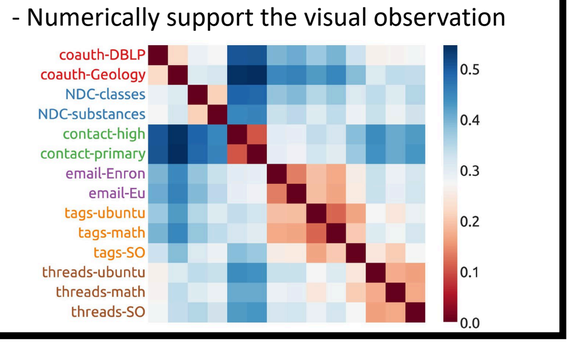
Domain	Nodes	Hyperedges
coauth	Researchers	Publications
NDC	Classes/Substances	Drugs
contact	People	Communications
email	Senders/Receivers	Emails
tags	Tags	Questions
threads	Users	Threads

Observation 1: Domain-based patterns of hypercore sizes

- **Observation:** Real-world hypergraphs in the **same domain** have **similar patterns** of hypercore sizes
- **Color of each pixel:** The (k, t)-hypercore sizes
- **x-Axis:** the value of k; **y-Axis:** the value of t
  - Visual similarity with each domain
  - Different patterns in different domains

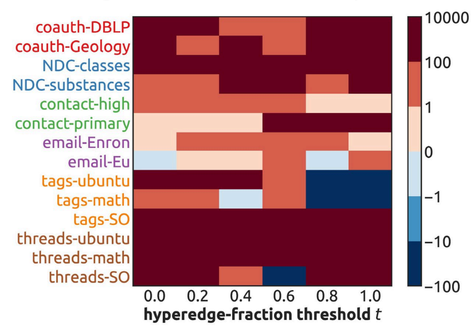


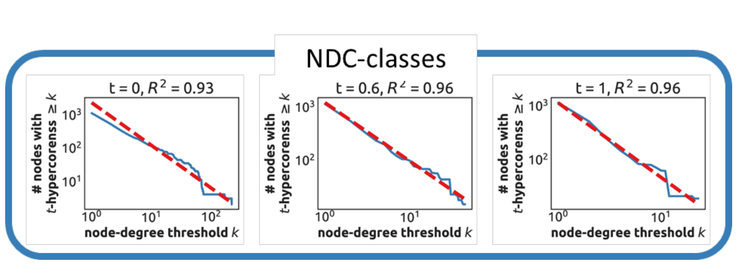
- **Color of each block:** The distance between a pair of datasets w.r.t hypercore sizes
  - The distance within the same domain is small
  - Numerically support the visual observation



Observation 2: Heavy-tailed hypercoreness distributions

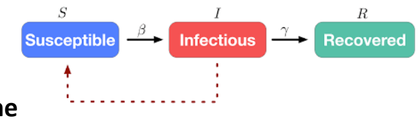
- **Observation:** In real-world hypergraphs, t-hypercoreness follows **heavy-tailed** distributions regardless of t
- Statistical tests on the t-hypercoreness distributions
- **Red:** High likelihood of a heavy-tailed distribution
- In some datasets, we observe a strong **power law**
- **Red:** Reference power-law fitting lines



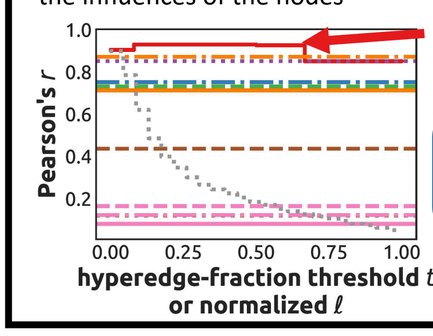


Application 1: Influential-node identification

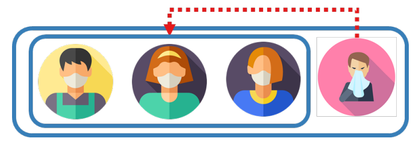
- **Summary:** t-Hypercoreness with a proper t value **identifies influential nodes** well
- **Set-up:** We use a widely-used epidemic model, the SIR model
  - A **single** initially infected node
  - Infected nodes can infect susceptible nodes **in the same hyperedge**
  - Infected nodes have some probability to **recover** and become **immune**

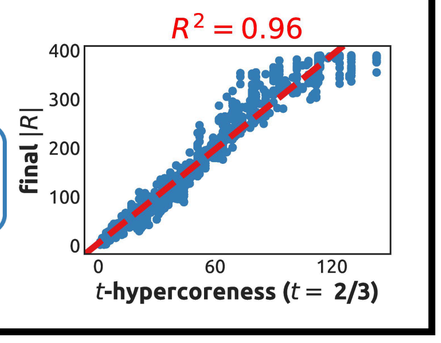


- **Influence:** The influence of a node v is the **# ever-infected nodes** when v is the initially infected node
- **Metric:** For each considered measure, we compute the **Pearson's r** between the values of the measure and the influences of the nodes



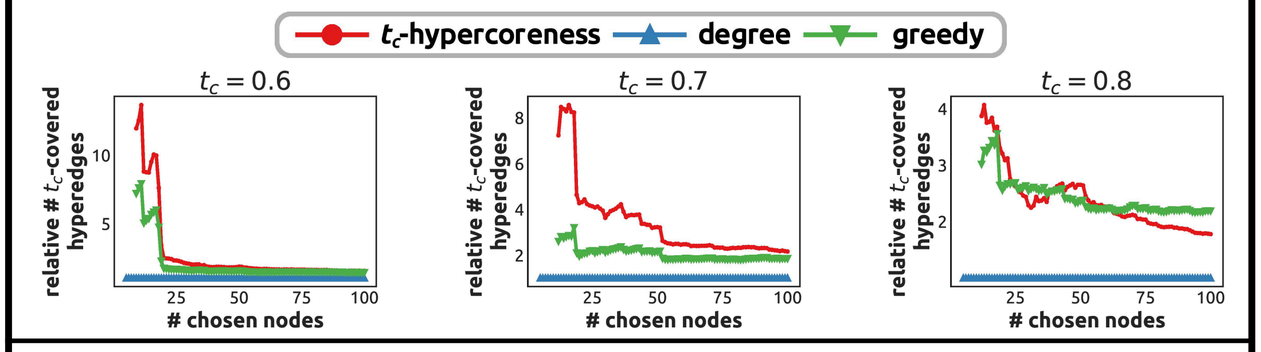
***t*-Hypercoreness** has the highest correlation with the influences





Application 2: Dense substructure discovery

- **Summary:** The proposed concepts can be used to **find dense substructures** in real-world hypergraphs
- **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
- **Results:** Overall, t_c-hypercoreness outperforms the other two methods, i.e., covers more hyperedges



- **Some observations and applications are not presented here due to space limit**
- **Observation:** t-Hypercoreness is statistically different **from other existing centrality measures**
  - t-Hypercoreness can provide unique insights of a hypergraph
- **Observation:** With **varying t values**, the t-hypercoreness can be statistically different **from each other**
  - Using different t values can provide us different insights
- **Application:** The concept of (k, t)-hypercores can be used to detect **vulnerability** in hypergraphs