This is a preliminary version of the slides that will be used for tutorials.

The slides will be revised to reflect recent studies and recommended improvements.

The final version may differ from this version.

Mining of Real-world Hypergraphs: Concepts, Patterns, and Generators

Part IV. Generative Models

Geon Lee

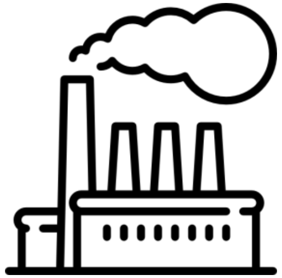
Jaemin Yoo

Kijung Shin

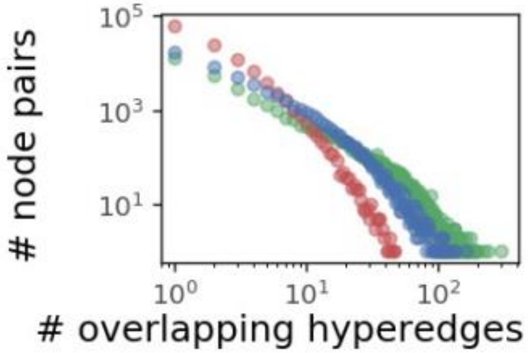
Part 3. Generative Models

*"How can we generate **realistic hypergraphs**?"*

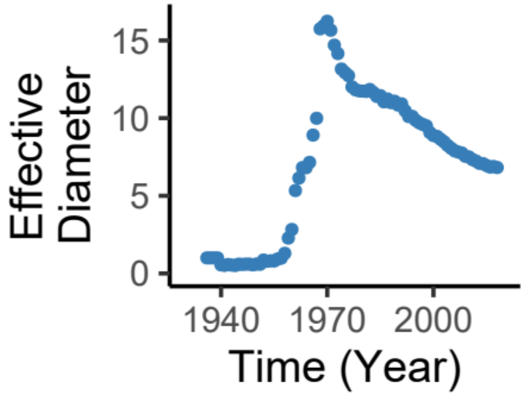
*"What are **underlying mechanisms** that lead to the observed patterns?"*



Generative models



Static Graphs (Part 3-1)



Dynamic Graphs (Part 3-2)

Roadmap

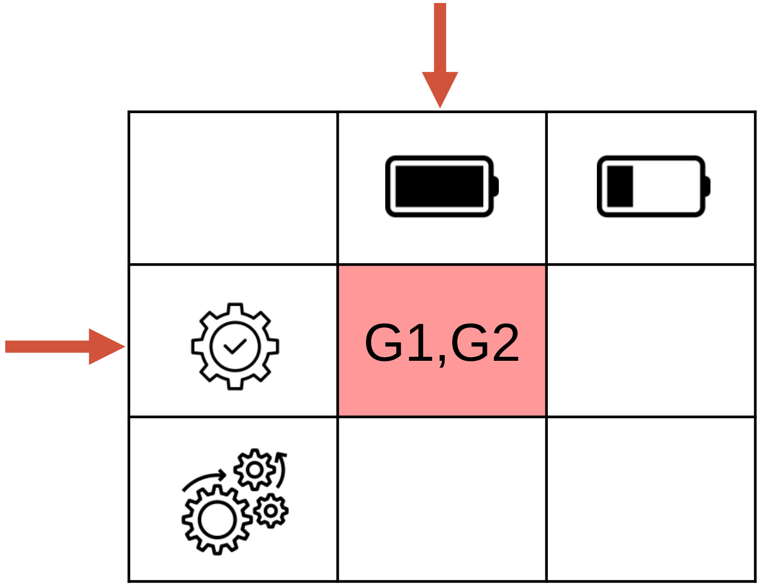
- **Part 1. Static Structural Patterns**
  - Basic Patterns
  - Advanced Patterns
- **Part 2. Dynamic Structural Patterns**
  - Basic Patterns
  - Advanced Patterns
- **Part 3. Generative Models**
  - Static hypergraph Generator **<<**
  - Dynamic hypergraph Generator

Part 3-1. Static Hypergraph Generative Models

		Part 3. Generative Models
Static Models	Full-Hypergraphs	C20, LCS21
	Sub-Hypergraphs	CYLBKS22
Dynamic Models	Full-Hypergraphs	DYHS20, KKS20
	Sub-Hypergraphs	BKT18, CK21

C20: Configuration Models

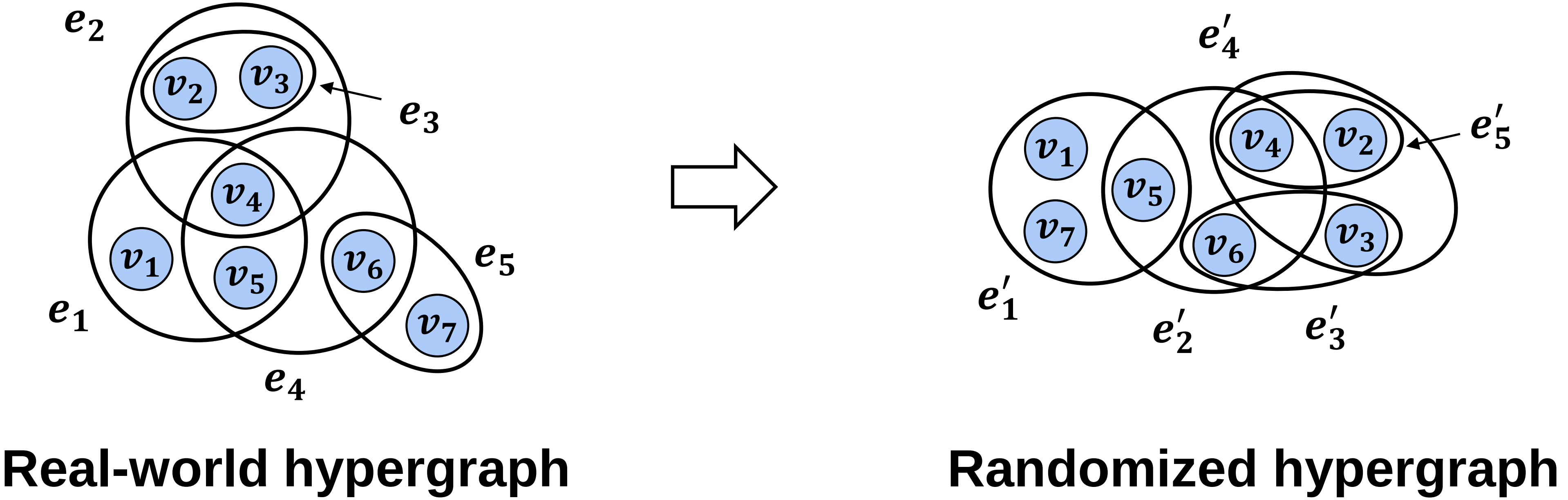
- **G1:** Hypergraph stub-matching
- **G2:** Markov Chain Monte Carlo



P. S. Chodrow, “Configuration Models of Random Hypergraphs”,
Journal of Complex Networks (2020)

Configuration Models

- **Configuration models** generate random hypergraphs that exactly preserve distributions of **node degrees** and **hyperedge sizes**.



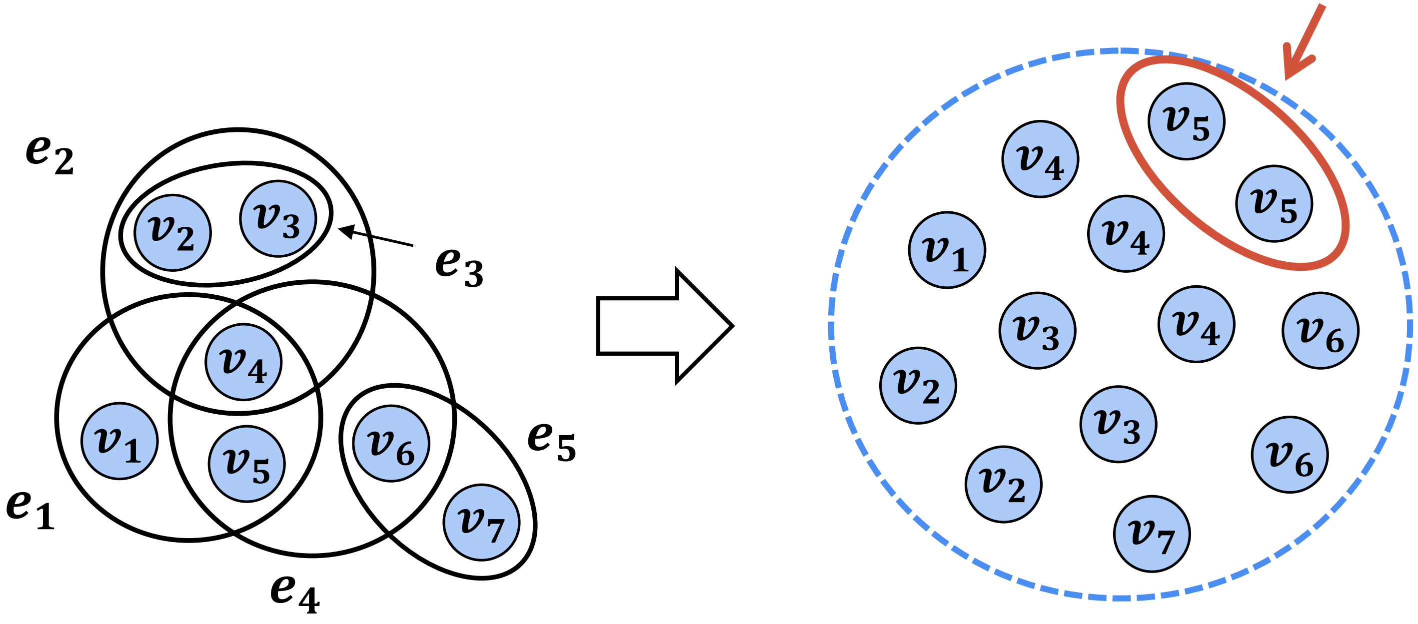
Configuration Models (cont.)

- Two configuration models of random hypergraphs:
  - **Stub-matching model**
  - **Pairwise reshuffling**

Hypergraph Stub-matching

Step 1. Multiset Generation

Degree number of the node



Multiset of nodes W

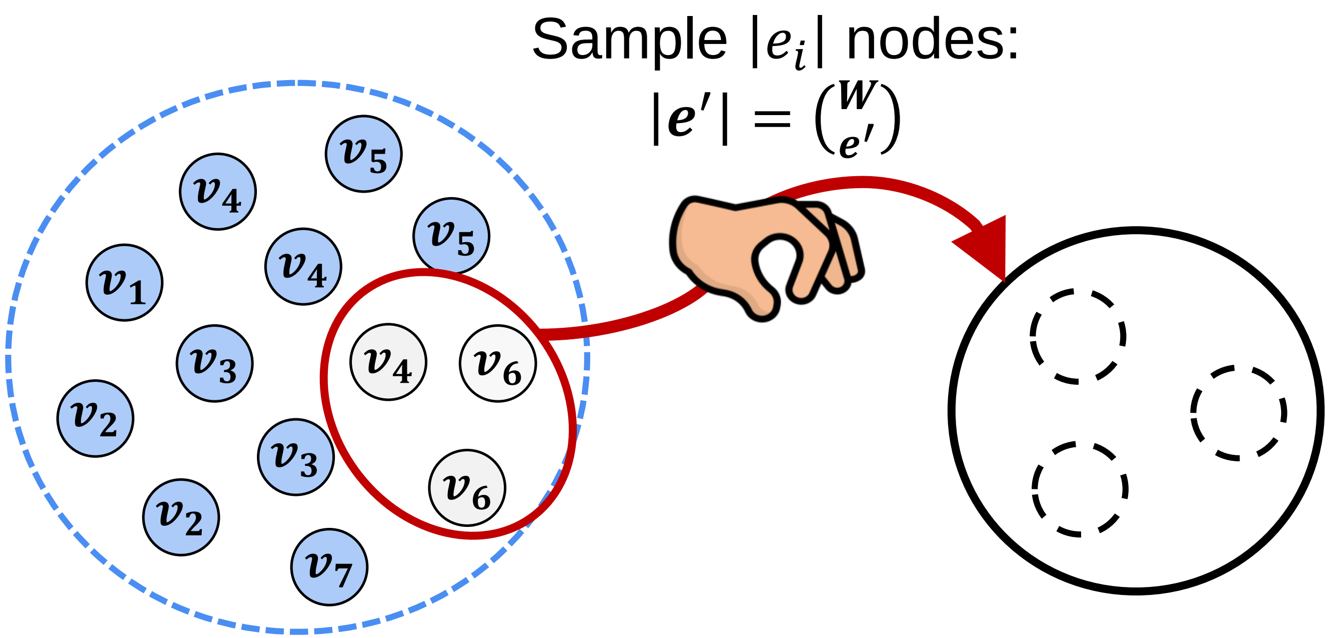
Step 1

Generate **a multiset *W*** of nodes such that the degree number of copies of each node is contained.

$$W = \biguplus_{v \in V} \{v_1, \ldots, v_d\}$$

Hypergraph Stub-matching (cont.)

Step 2. Hyperedge Sampling



Multiset of nodes $\boldsymbol{W}$

Generated hyperedge $\boldsymbol{e}_i'$

Step 2-1

To generate a hyperedge e_i', sample $|e_i|$ nodes from multiset W uniformly at random.

$$e_i' = \binom{\boldsymbol{W}}{|\boldsymbol{e}_i|}$$

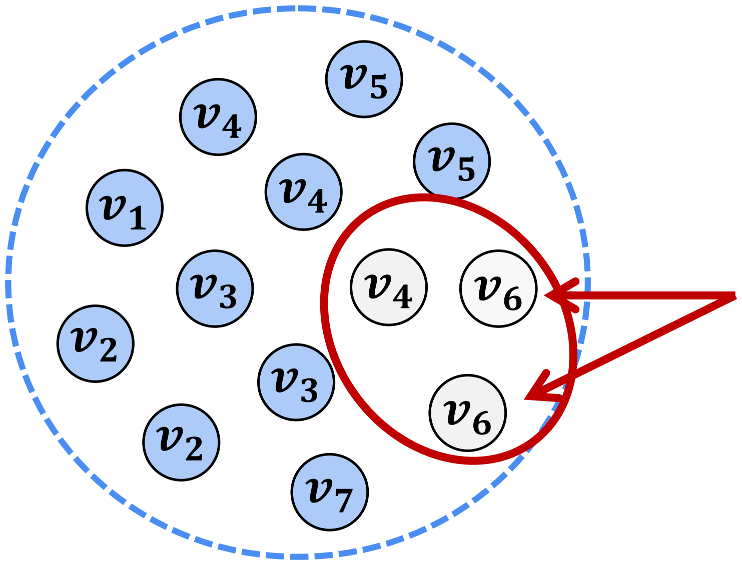
Step 2-2

Remove the items sampled from W.

$$W = W \setminus e_i'$$

Hypergraph Stub-matching (cont.)

Step 2. Hyperedge Sampling



Two copies of the same node

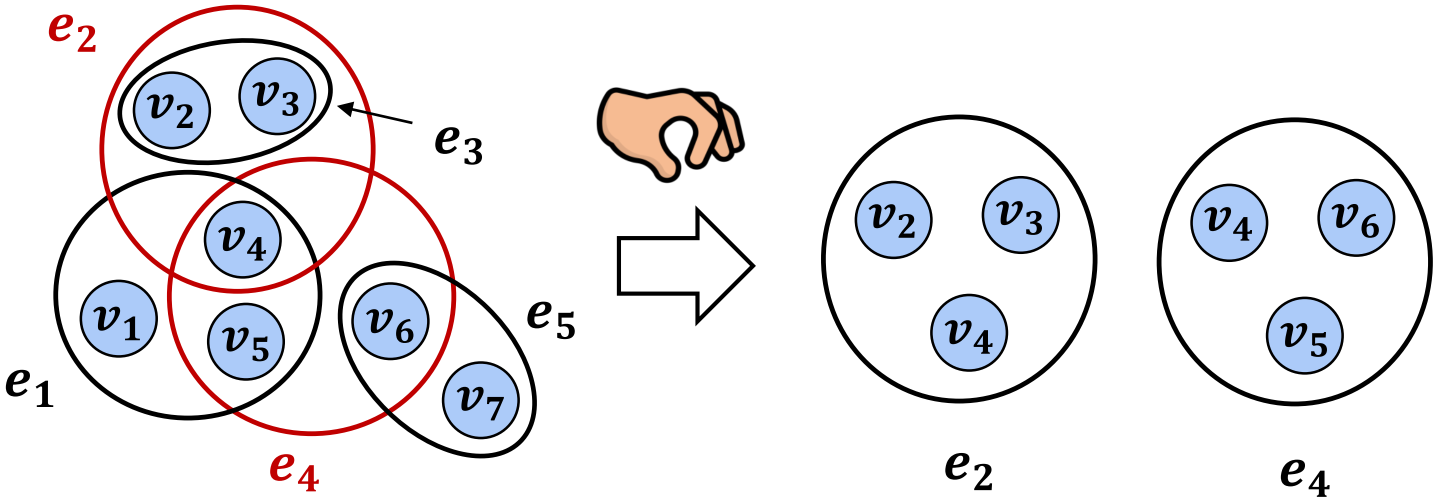
Problem

Two copies of the same node can be sampled, which yields a degenerate hyperedges.

Multiset of nodes W

Pairwise Reshuffling

Step 1. Hyperedge Pair Sampling



Step 1

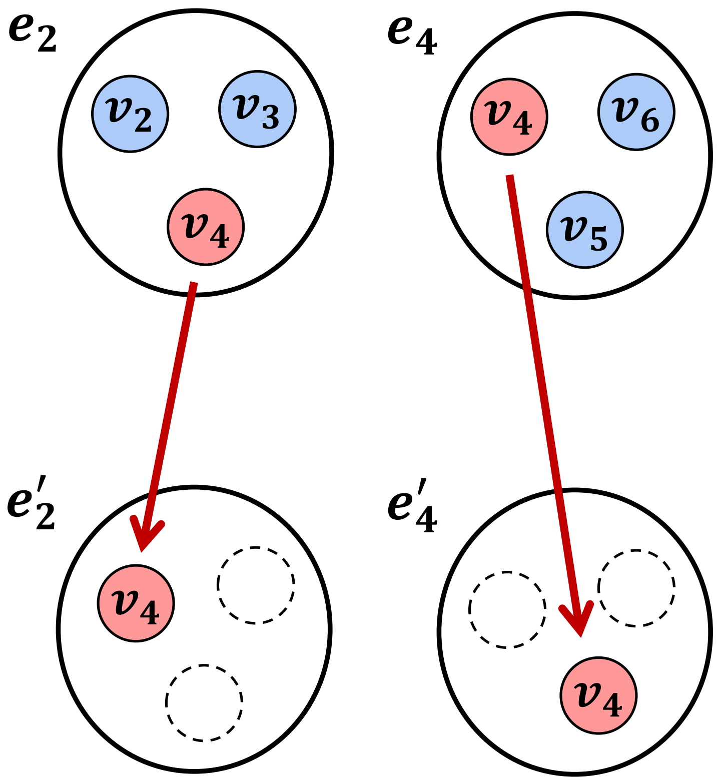
Sample a pair of hyperedges uniformly at random.

$$(e_i, e_j) \in \binom{\boldsymbol{E}}{\boldsymbol{2}}$$

P. S. Chodrow, "Configuration Models of Random Hypergraphs",
Journal of Complex Networks (2020)

Pairwise Reshuffling

Step 2. Shuffle Hyperedges

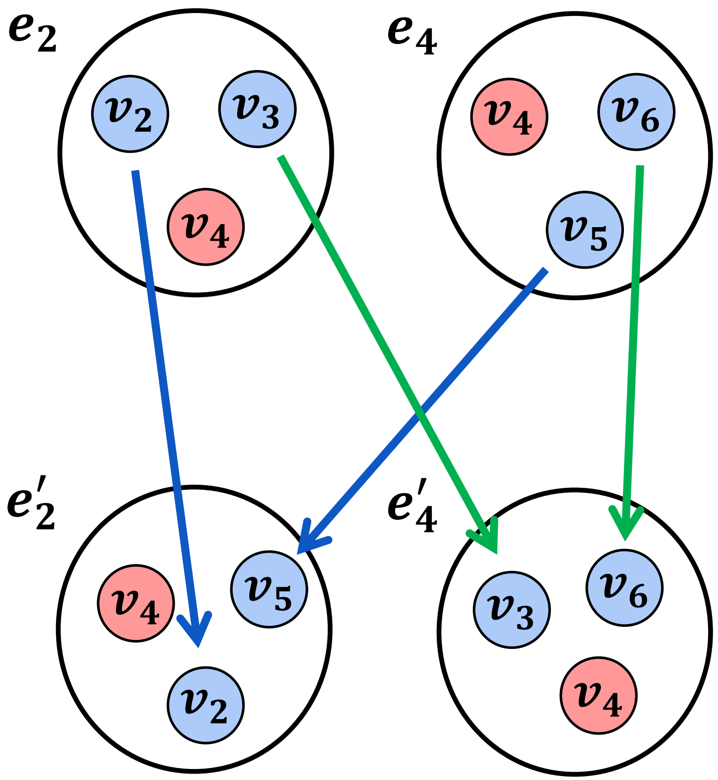


Step 2-1

For each node $v \in e_i \cap e_j$, add to both e'_i and e'_j.

Pairwise Reshuffling

Step 2. Shuffle Hyperedges



Step 2-2

From $(e_i \cup e_j) - (e_i \cap e_j)$, sample $|e_i - e_j|$ nodes and add to e'_i.

Step 2-3

Add remaining nodes to e'_j.

No degenerate hyperedges

P. S. Chodrow, "Configuration Models of Random Hypergraphs",
Journal of Complex Networks (2020)

Configuration Models: Evaluation

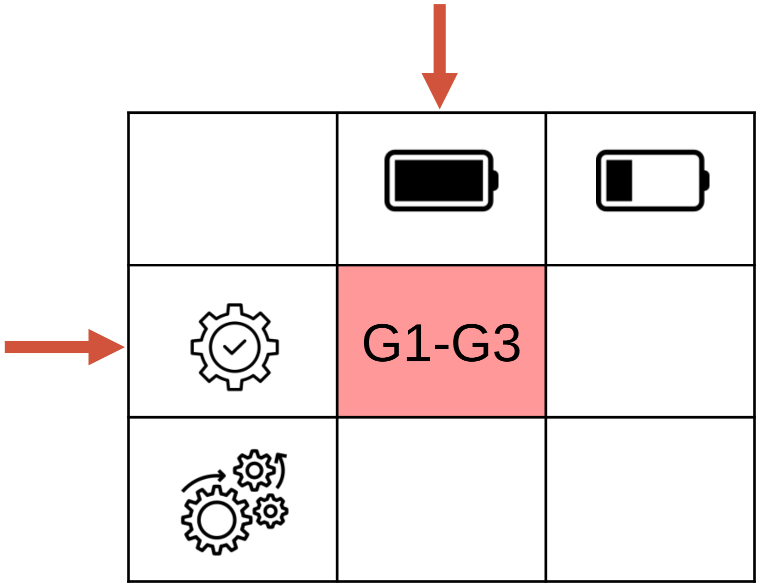
- Configuration models on hypergraphs accurately preserve the **average local clustering coefficient**.

	Real Hypergraph	Hypergraph		Projected graph	
		Stub	Reshuffle	Stub	Reshuffle
congress-bills	0.608	0.622	0.601	0.611	0.451
coauth-MAG-geology	0.820	0.818	0.819	0.000	0.000
email-Enron	0.658	0.808	0.825	0.797	0.638
email-Eu	0.540	0.601	0.569	0.598	0.398
tags-ask-ubuntu	0.571	0.631	0.609	0.499	0.183
threads-math-sx	0.293	0.426	0.435	0.093	0.041

P. S. Chodrow, “Configuration Models of Random Hypergraphs”,
Journal of Complex Networks (2020)

LCS21: Static Full-Hypergraph Generator

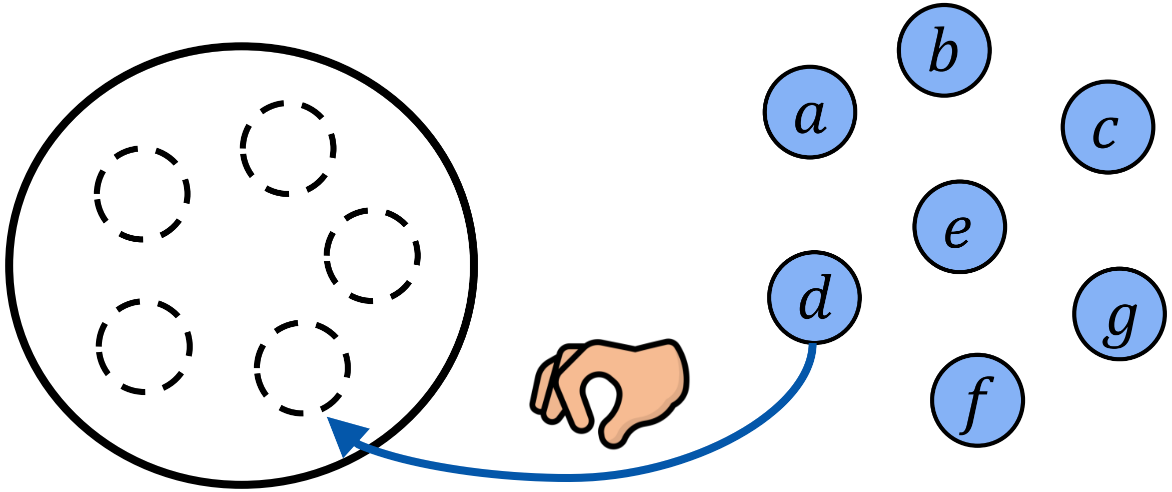
- **G1.** HyperCL: **Hyper**grpaph **C**hung-**L**u (Null model)
- **G2.** HyperLap: **Hyper**graph Over**Lap** (Multilevel HyperCL)
- **G3.** HyperLap^{+}: Parameter Selection



G. Lee, M. Choe, K. Shin, "How Do Hyperedges Overlap in Real-world Hypergraphs? – Patterns, Measures, and Generators", **WWW 2021**

HyperCL: Null Model

- **HyperCL** samples nodes with **probability** $\propto$ **degrees**.
- The **degree distribution** of nodes is empirically preserved.

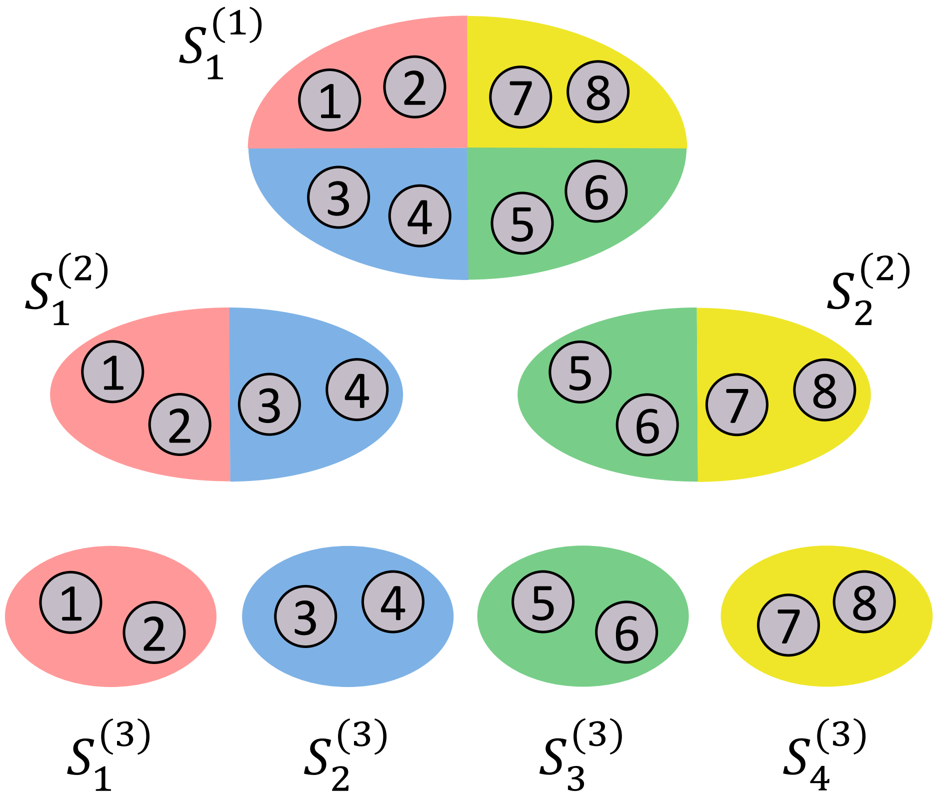


Random sampling
prob. $\propto$ $\boldsymbol{d}$**'s degree**

HyperLap: Multilevel HyperCL

Step 1. Hierarchical Node Partitioning

Given number of levels L (L=3 for example)



At the lowest **level 1**, group $\boldsymbol{S_1^{(1)}}$ is the entire node set:

$$\boldsymbol{S_1^{(1)} = V}$$

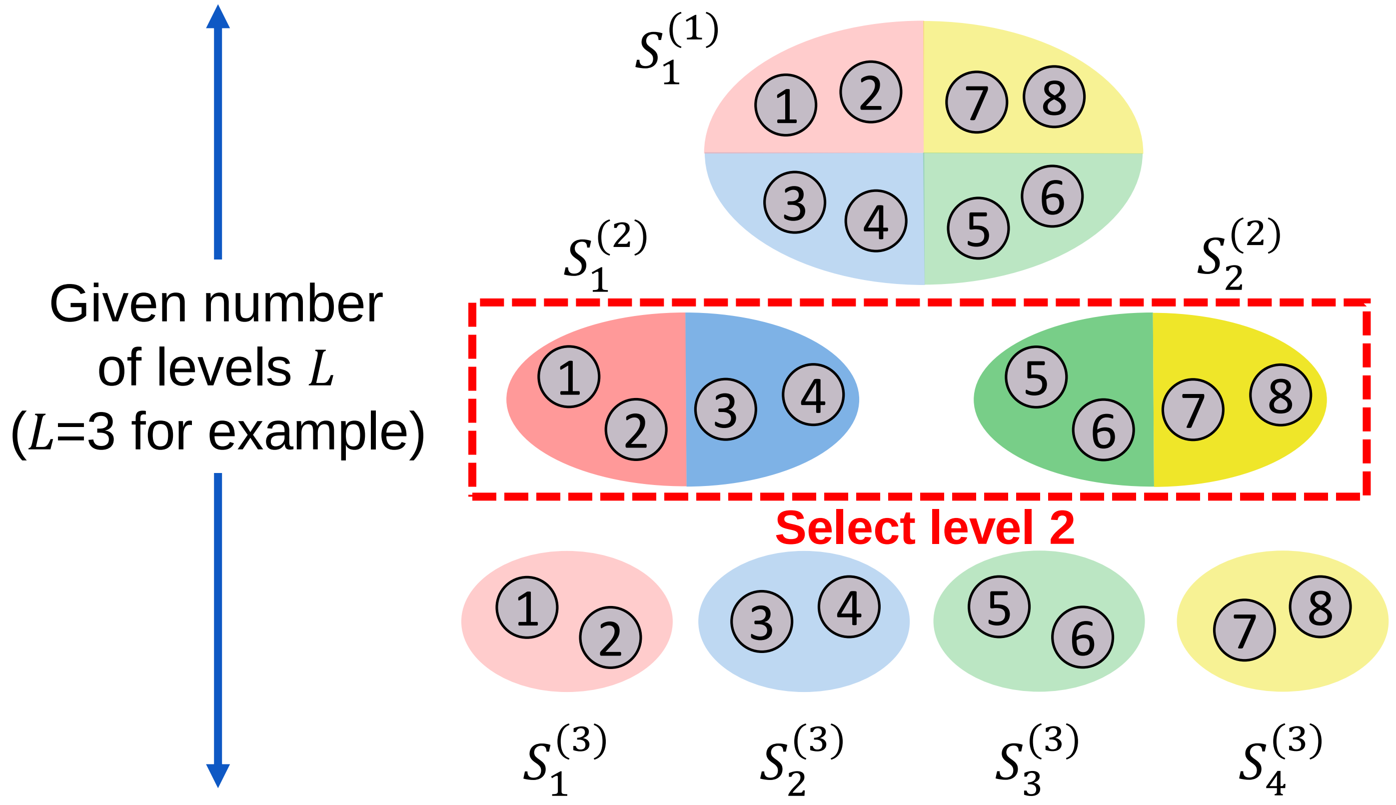
At **level ℓ**, group $\boldsymbol{S_i^{(\ell)}}$ is the union of the lower level's ones:

$$\boldsymbol{S_i^{(\ell)} = S_{2i-1}^{(\ell+1)} \cup S_{2i}^{(\ell+1)}}$$

Nodes are partitioned into $\boldsymbol{2^{L-1}}$ disjoint groups.

HyperLap: Multilevel HyperCL (cont.)

Step 2. Hyperedge Generation

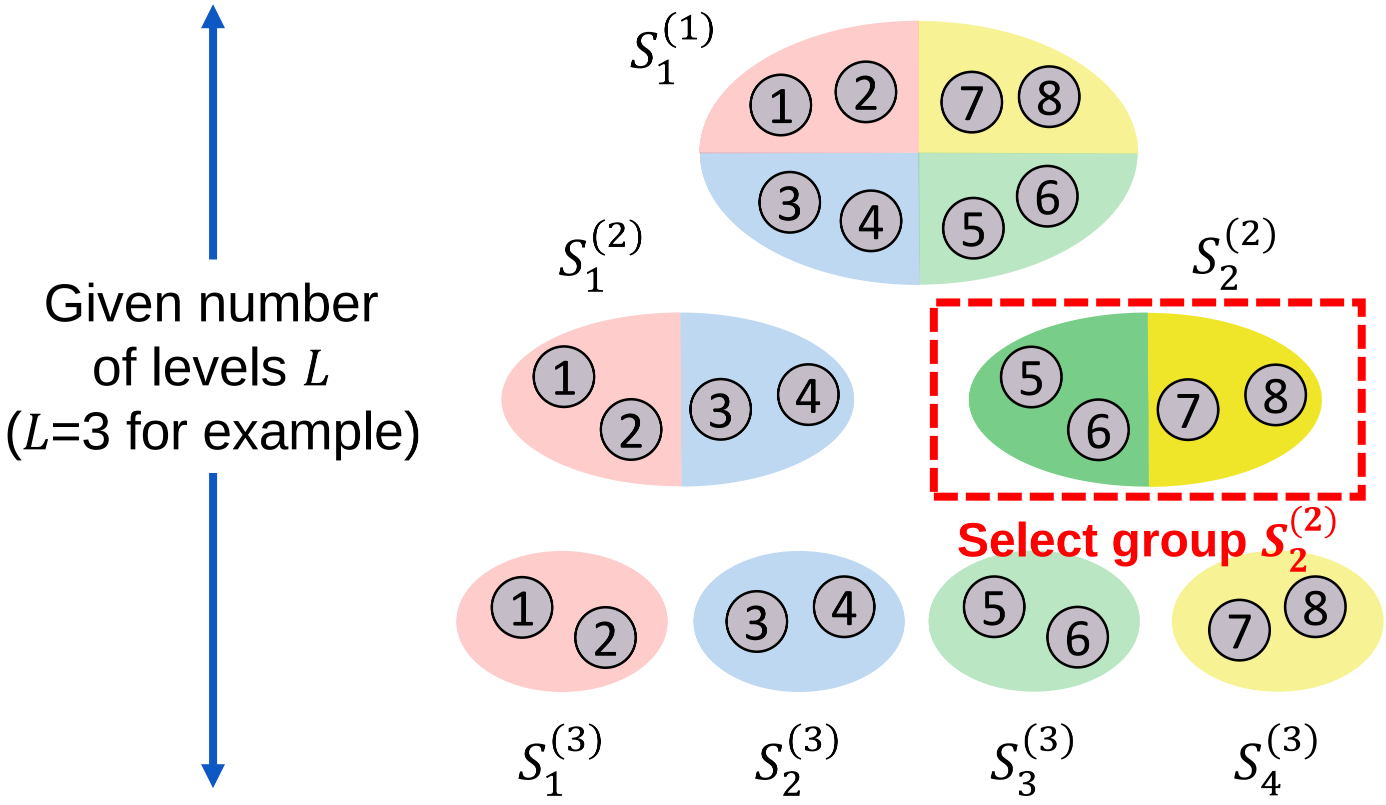


Step 2-1

Select a level with probability proportional to the given weight of each level $\{w_1, \ldots, w_L\}$.

HyperLap: Multilevel HyperCL (cont.)

Step 2. Hyperedge Generation

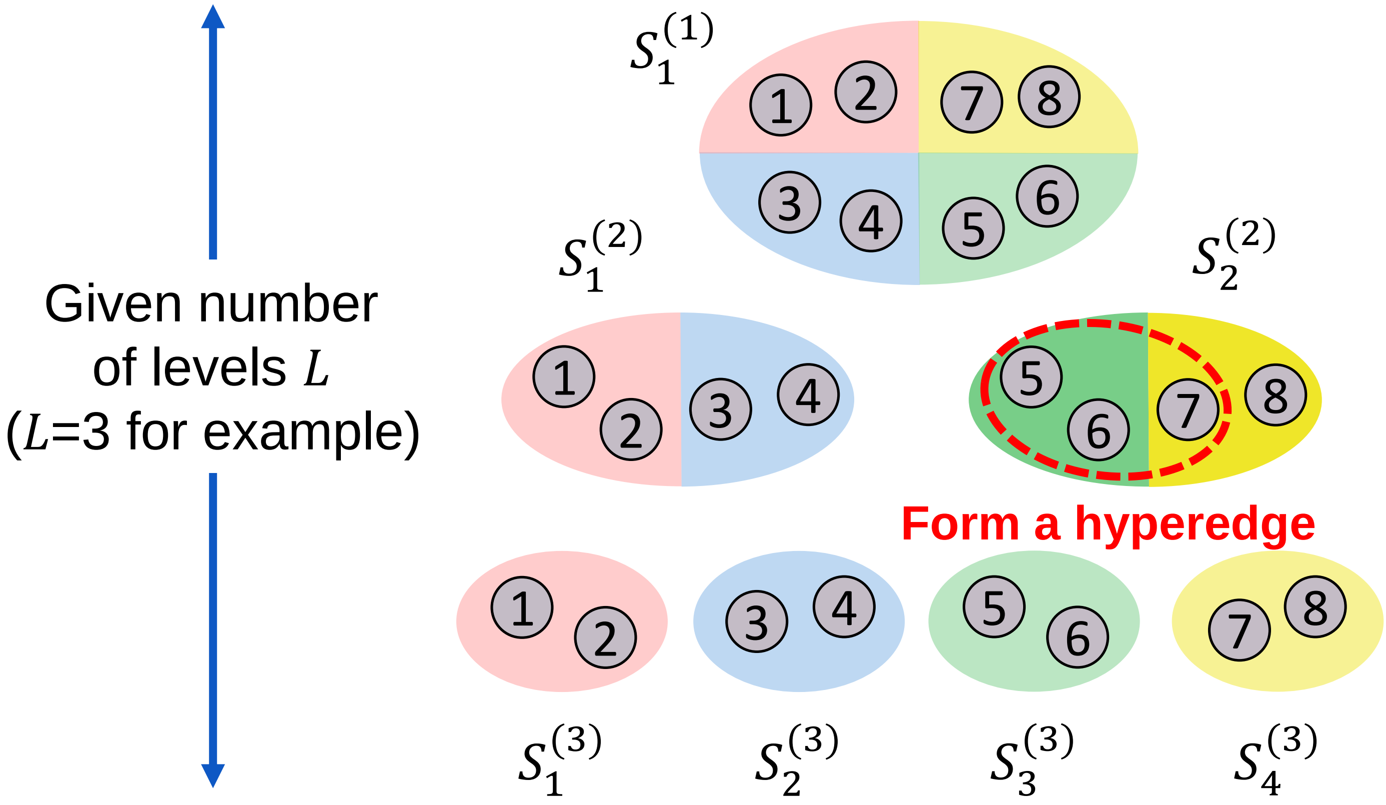


Step 2-2

Select a group uniformly at random.

HyperLap: Multilevel HyperCL (cont.)

Step 2. Hyperedge Generation



Step 2-3

Sample nodes independently with probability proportional to the degree of each node to form a hyperedge.

G. Lee, M. Choe, K. Shin, "How Do Hyperedges Overlap in Real-world Hypergraphs? – Patterns, Measures, and Generators", **WWW 2021**

Details

HyperLap: Design Principles

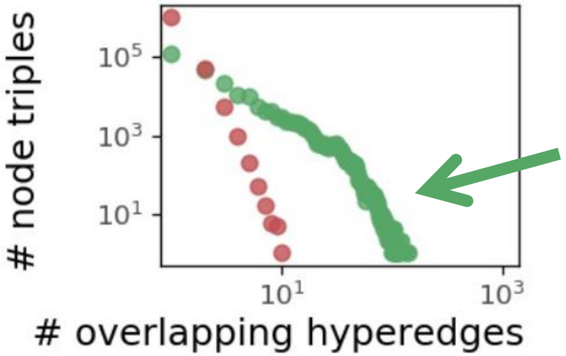
?

Question:

Why does **HyperLap** work?

Answer 1:

- Real hyperedges **highly overlap** within groups.
- **HyperLap** generate hyperedges from small groups.



The number of overlapping hyperedges at each pair or triple is **skewed**.

!

G. Lee, M. Choe, K. Shin, "How Do Hyperedges Overlap in Real-world Hypergraphs? – Patterns, Measures, and Generators", **WWW 2021**

Details

HyperLap: Design Principles (cont.)

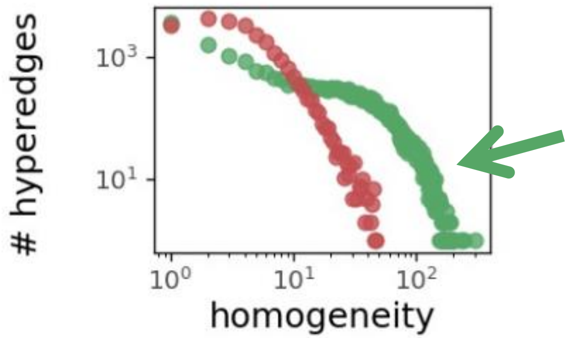
?

Question:

Why does **HyperLap** work?

Answer 2:

- Real hyperedges are **homogeneous**.
- **HyperLap** generate hyperedges with structurally similar nodes.



Hyperedges in real-world hypergraphs tend to have **high homogeneity**.

!

Details

HyperLap: Design Principles (cont.)

?

Question:

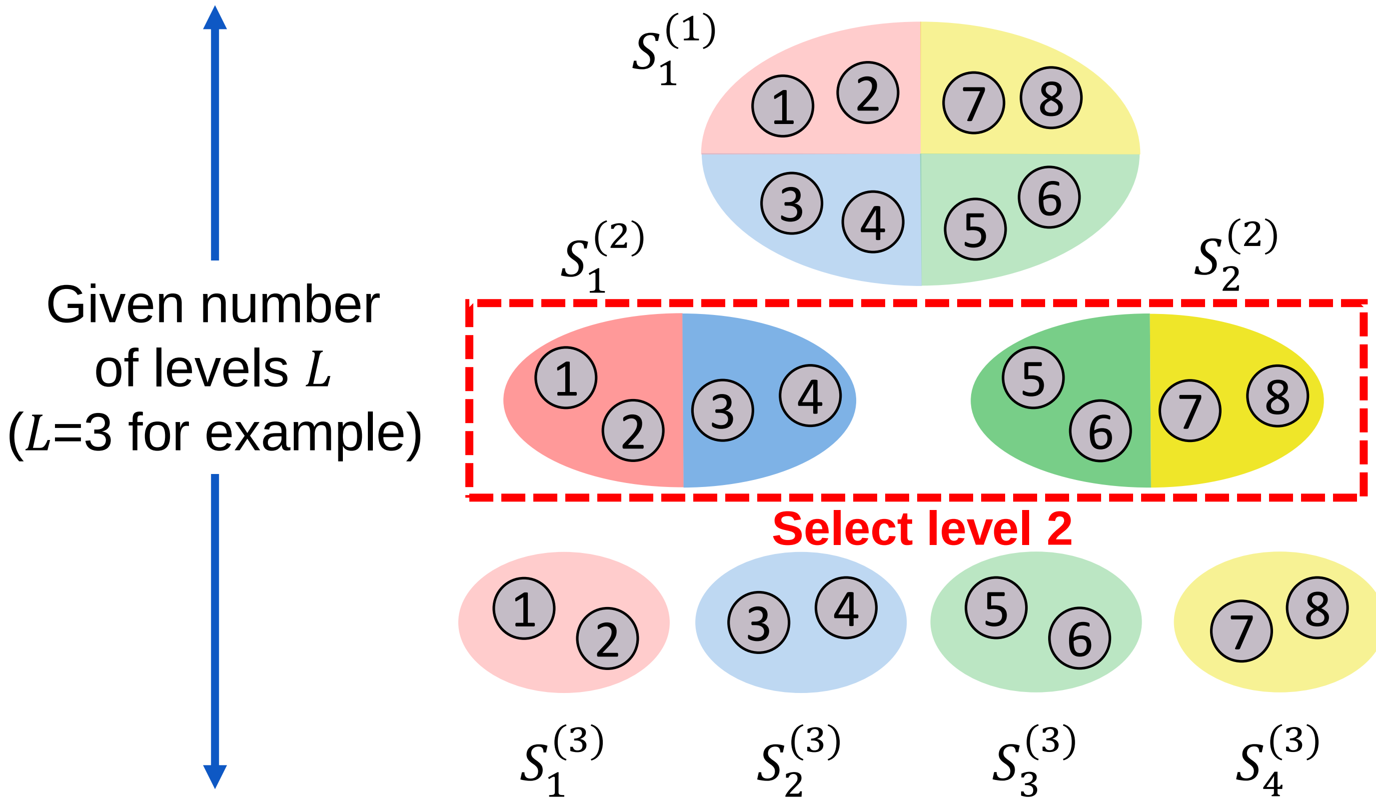
Why does **HyperLap** work?

Answer 3:

- Real egonets have **high density & overlapness**.
- In **HyperLap**, hyperedges in the egonet of each node are likely to contain nodes from the same group.

!

HyperLap^{+}: Motivation



Step 2-1

Select a level with probability proportional to the given weight of each level $\{\boldsymbol{w_1}, \ldots, \boldsymbol{w_L}\}$.

*Can we tune the parameters **automatically**?*

HyperLap^{+}: Objective

- Minimize the **hyperedge homogeneity distance** $HHD(\mathcal{G}, \hat{\mathcal{G}})$.

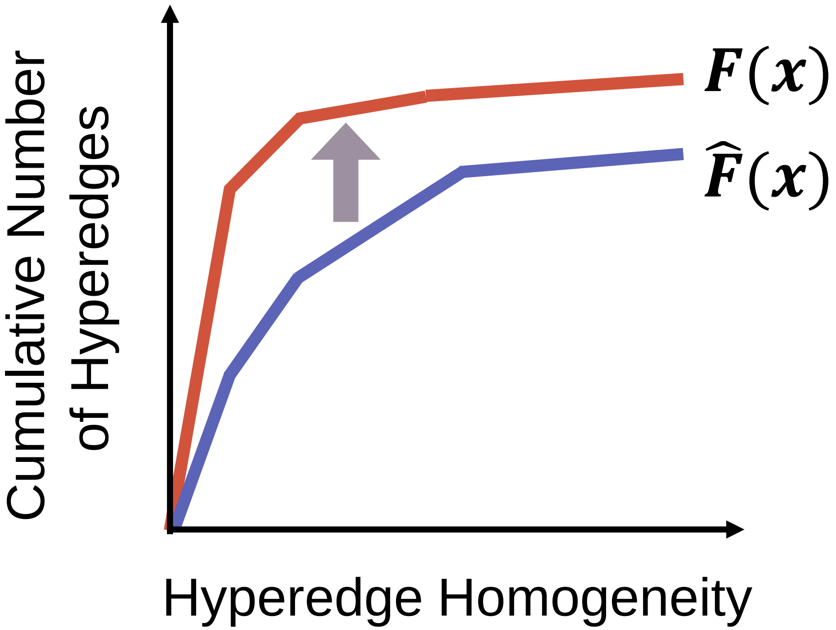
$$HHD(\mathcal{G}, \hat{\mathcal{G}}) = \max_{x}\{|F(x) - \hat{F}(x)|\}$$

Cumulative hyperedge homogeneity distribution of **input hypergraph $\mathcal{G}$**

Cumulative hyperedge homogeneity distribution of **generated hypergraph $\hat{\mathcal{G}}$**

HyperLap^{+}: Objective (cont.)

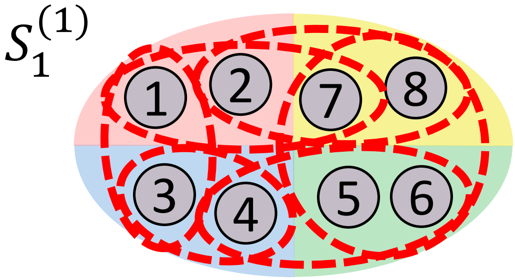
- Minimize the **hyperedge homogeneity distance** $\boldsymbol{HHD}(\mathcal{G}, \widehat{\mathcal{G}})$.



$$\min_{w_1, \dots, w_L} HHD(G, \hat{G}) \text{ where } w_1 + \cdots + w_L = 1$$

Learnable parameters

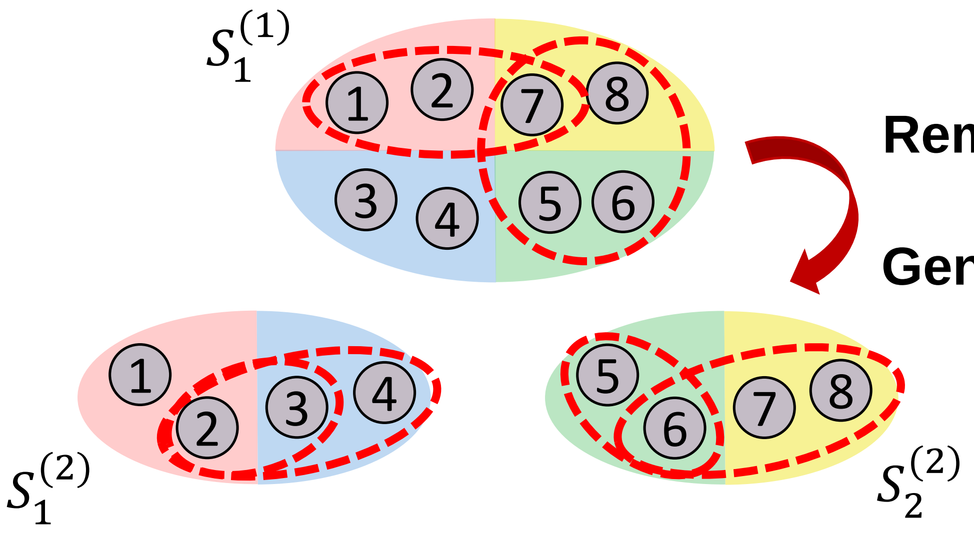
HyperLap^{+}: Automatic HyperLap



Step 1

Generate $|E|$ hyperedges at level 1.

HyperLap⁺: Automatic HyperLap (cont.)



Search for an optimal q_1.

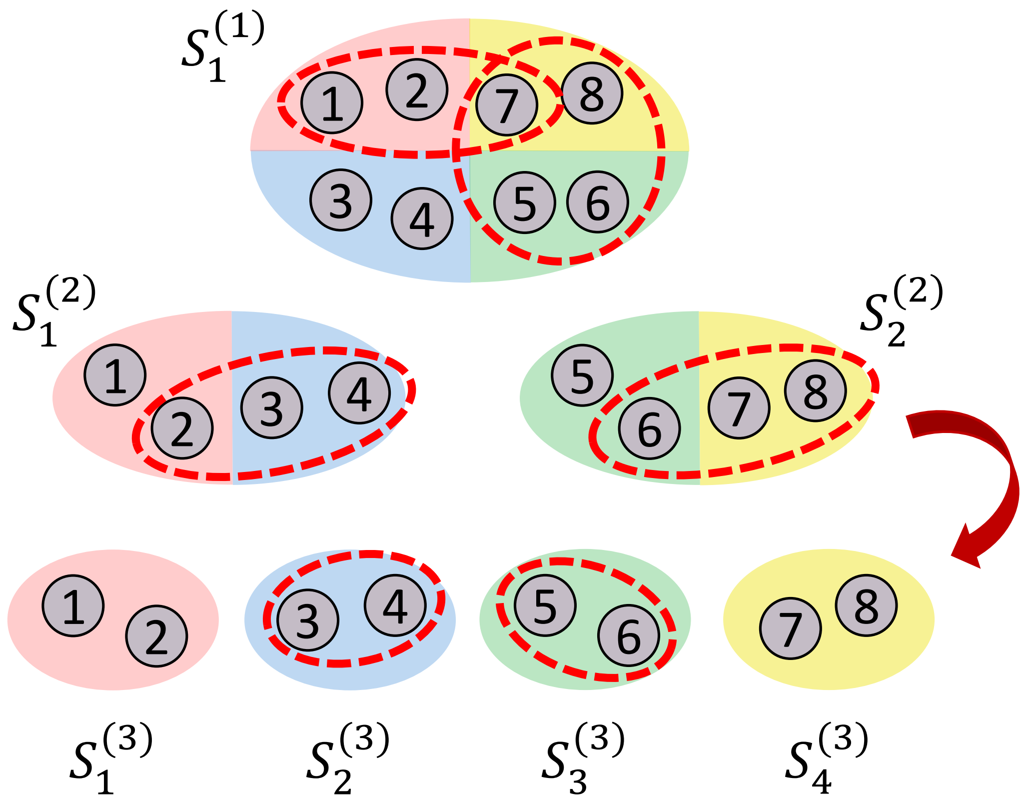
Remove ($\boldsymbol{q_1} \cdot 100$)% hyperedges at level 1.

Generate ($\boldsymbol{q_1} \cdot 100$)% hyperedges at level 2.

Step 2

Replace $(q_1 \cdot 100)$% hyperedges at level 1 to level 2.

HyperLap$^+$: Automatic HyperLap (cont.)



Search for an optimal q_2.

Remove ($q_2 \cdot 100$)% hyperedges at level 2.

Generate ($q_2 \cdot 100$)% hyperedges at level 3.

Step 3

Replace ($q_2 \cdot 100$)% hyperedges at level 2 to level 3.

Until Convergence

HyperLap^{+}: Evaluation

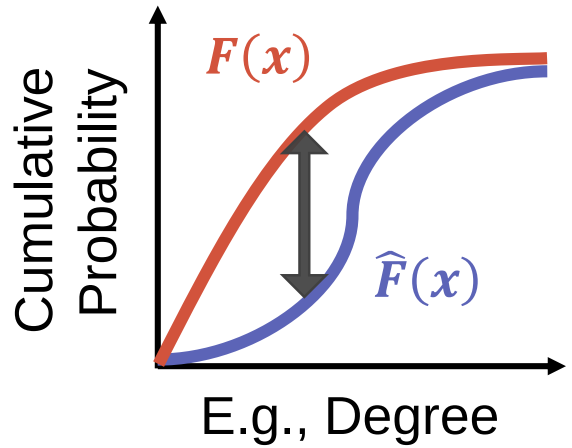
?

Question:

How to measure the similarity between $\mathcal{G}$ and $\tilde{\mathcal{G}}$?

Answer:

- We measure the **distance** between the distributions derived from $\mathcal{G}$ and $\tilde{\mathcal{G}}$ by **D-statistics**.



$$\max_{x}\{|F(x) - \hat{F}(x)|\}$$

!

HyperLap^{+}: Evaluation (cont.)

- **HyperLap^{+}** reproduces most accurately the distributions of:

  egonet density, **egonet overlapness**, and **hyperedge homogeneity**

Dataset	Density of Egonets (Obs. 1)					Overlapness of Egonets (Obs. 2)					Homogeneity of Hyperedges (Obs. 5)				
	H-CL	H-PA	H-FF	H-LAP	**H-LAP^{+}**	H-CL	H-PA	H-FF	H-LAP	**H-LAP^{+}**	H-CL	H-PA	H-FF	H-LAP	**H-LAP^{+}**
email-Enron	0.545	0.202	0.391	0.405	**0.125**	0.517	0.398	0.398	0.391	**0.111**	0.498	0.241	0.656	0.191	**0.136**
email-Eu	0.724	-	0.402	0.577	**0.310**	0.534	-	0.639	0.432	**0.197**	0.505	-	0.688	0.247	**0.168**
contact-primary	0.896	0.537	0.975	0.334	**0.128**	0.867	0.471	0.942	0.285	**0.095**	0.430	0.236	0.484	**0.142**	0.188
contact-high	0.948	0.529	0.880	0.522	**0.345**	0.874	0.431	0.703	0.486	**0.296**	0.423	0.196	0.336	**0.120**	0.178
NDC-classes	0.694	0.785	0.731	0.696	**0.635**	0.302	0.715	0.406	**0.231**	0.248	0.274	0.410	0.484	0.272	**0.225**
NDC-substances	0.451	-	0.801	0.426	**0.366**	0.321	-	0.338	0.243	**0.157**	0.377	-	0.740	0.262	**0.108**
tags-ubuntu	0.522	**0.162**	0.216	0.410	0.300	0.432	**0.117**	0.398	0.487	0.210	0.245	0.136	0.844	0.105	**0.011**
tags-math	0.496	0.350	0.561	**0.195**	0.227	0.460	0.325	0.709	**0.151**	0.186	0.337	0.217	0.921	0.086	**0.015**
threads-ubuntu	0.159	0.856	-	0.163	**0.159**	0.299	0.953	-	0.300	**0.297**	0.020	0.291	-	0.016	**0.011**
threads-math	0.137	0.492	-	**0.120**	0.135	0.232	0.714	-	0.235	**0.229**	0.060	0.368	-	0.102	**0.019**
coauth-DBLP	0.228	-	-	0.227	**0.132**	0.302	-	-	0.267	**0.244**	0.715	-	-	0.540	**0.026**
coauth-geology	0.200	-	-	0.202	**0.138**	**0.248**	-	-	0.252	0.266	0.624	-	-	0.481	**0.044**
coauth-history	**0.087**	-	-	0.090	0.089	**0.316**	-	-	0.321	0.324	0.154	-	-	0.125	**0.020**
Average	0.468	0.489	0.619	0.335	**0.237**	0.439	0.515	0.566	0.313	**0.219**	0.358	0.261	0.644	0.206	**0.088**

-: out of time (taking more than 10 hours) or out of memory

HyperLap$^+$: Evaluation (cont.)

- **HyperLap$^+$** reproduces most accurately the distributions of:

pair & triple degree distribution

Dataset	Pair of Nodes (Obs. 3)								Triple of Nodes (Obs. 4)							
	Distance from Real (D-statistics)					Heavy-tail Test			Distance from Real (D-statistics)					Heavy-tail Test		
	H-CL	H-PA	H-FF	H-LAP	**H-LAP$^+$**	pw	tpw	logn	H-CL	H-PA	H-FF	H-LAP	**H-LAP$^+$**	pw	twp	logn
email-Enron	0.143	**0.056**	0.217	0.075	0.139	-2.37	-0.29	-1.53	0.089	0.295	0.136	**0.061**	0.072	-0.22	**0.38**	**0.24**
email-Eu	0.225	-	0.352	0.162	**0.066**	**0.24**	**2.75**	**2.53**	0.480	-	0.516	0.337	**0.206**	**0.41**	**2.11**	**1.96**
contact-primary	0.196	0.062	0.223	0.070	**0.051**	**9.53**	**15.74**	**13.92**	0.137	0.061	0.110	0.053	**0.031**	-1.86	-1.27	**1.23**
contact-high	0.277	**0.062**	0.141	0.127	0.067	-3.09	-0.95	-0.06	0.210	**0.131**	0.182	0.182	0.193	-3.95	-	**0.50**
NDC-classes	0.273	0.197	0.196	0.246	**0.172**	**12.15**	**14.42**	**14.04**	0.376	**0.167**	0.405	0.349	0.286	**3.22**	**7.92**	**7.34**
NDC-substances	0.272	-	0.244	0.251	**0.202**	**33.69**	**40.13**	**39.66**	0.521	-	0.591	0.492	**0.453**	**45.30**	**55.38**	**54.99**
tags-ubuntu	0.091	**0.019**	0.182	0.034	0.033	**42.33**	**43.70**	**43.55**	0.148	0.067	0.191	**0.020**	0.074	**14.25**	**15.57**	**15.43**
tags-math	0.095	0.066	0.278	0.073	**0.011**	**42.75**	**45.60**	**45.41**	0.209	**0.053**	0.286	0.113	0.079	**21.38**	**23.12**	**22.99**
threads-ubuntu	0.011	0.137	-	**0.008**	0.009	**1.28**	**1.75**	**1.75**	**0.004**	0.130	-	**0.004**	**0.004**	-1,346	-1.72	-1.72
threads-math	0.041	0.163	-	**0.014**	0.033	**15.79**	**16.66**	**16.52**	0.006	0.138	-	**0.001**	0.005	-1.49	-0.98	**0.96**
coauth-DBLP	0.224	-	-	0.191	**0.032**	**55.86**	**74.95**	**73.45**	0.215	-	-	0.214	**0.192**	**2.87**	**6.73**	**6.46**
coauth-geology	0.178	-	-	0.157	**0.040**	**31.13**	**45.08**	**44.06**	0.086	-	-	0.085	**0.069**	-0.10	**1.10**	**0.84**
coauth-history	0.033	-	-	0.030	**0.009**	**1.74**	**1.77**	**1.63**	**0.001**	-	-	**0.001**	**0.001**	-0.86	-	**0.57**
Average	0.158	0.095	0.229	0.110	**0.066**				0.193	0.130	0.302	0.147	**0.128**			

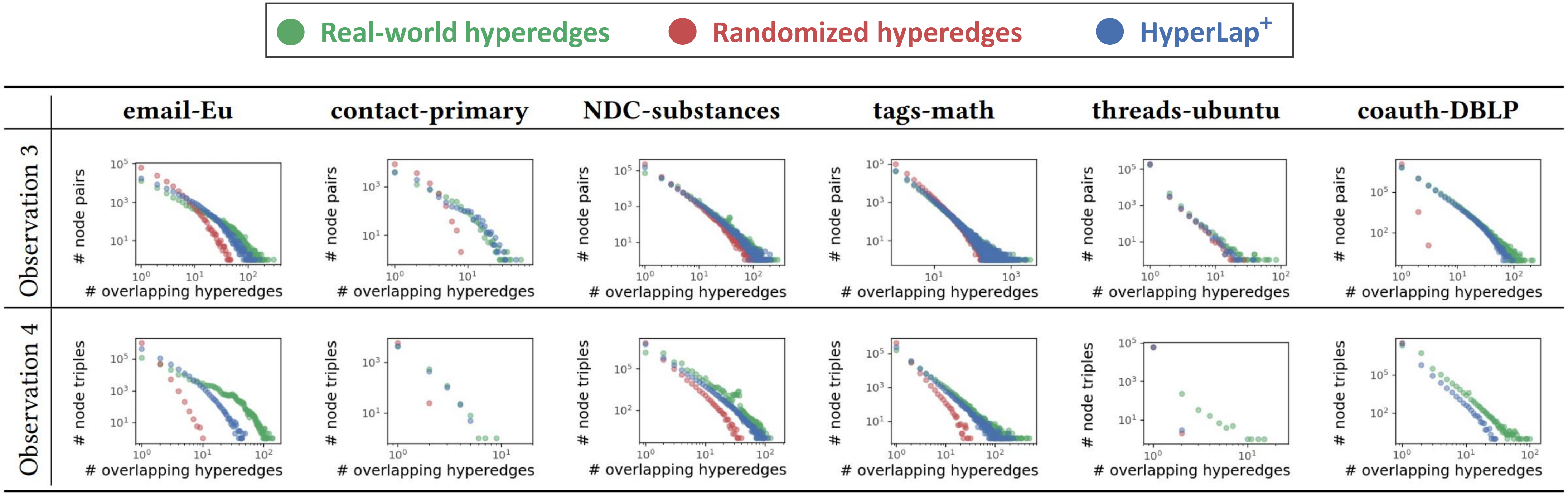
-: out of time (taking more than 10 hours) or out of memory

G. Lee, M. Choe, K. Shin, “How Do Hyperedges Overlap in Real-world Hypergraphs? – Patterns, Measures, and Generators”, **WWW 2021**

HyperLap$^+$: Evaluation (cont.)

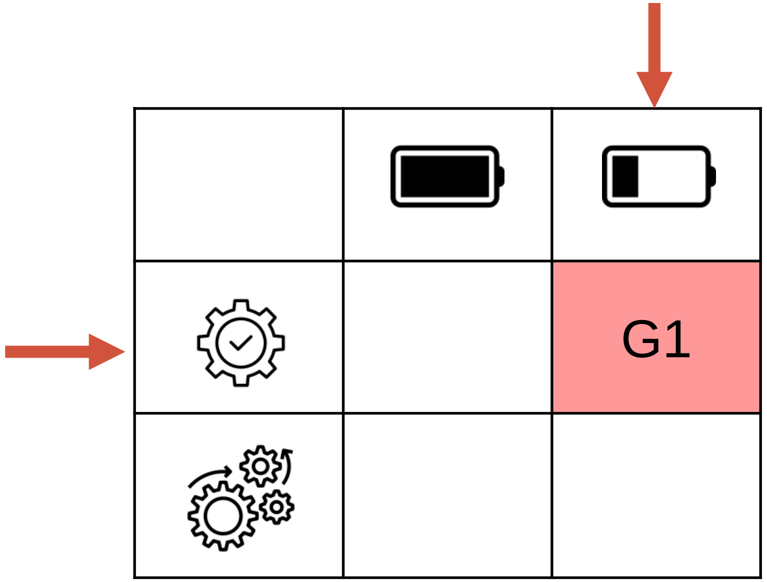
- **HyperLap$^+$** reproduces most accurately the distributions of:

pair & triple degree distribution



CYLBKS22: Static Sub-Hypergraph Generator

- **G1.** MiDaS: **Mi**nimum **D**egree Bi**a**sed **S**ampling of Hyperedges



M. Choe, J. Yoo, G. Lee, W. Baek, U Kang, K. Shin. "MiDaS: Representative Sampling from Real-world Hypergraphs", **WWW 2022**

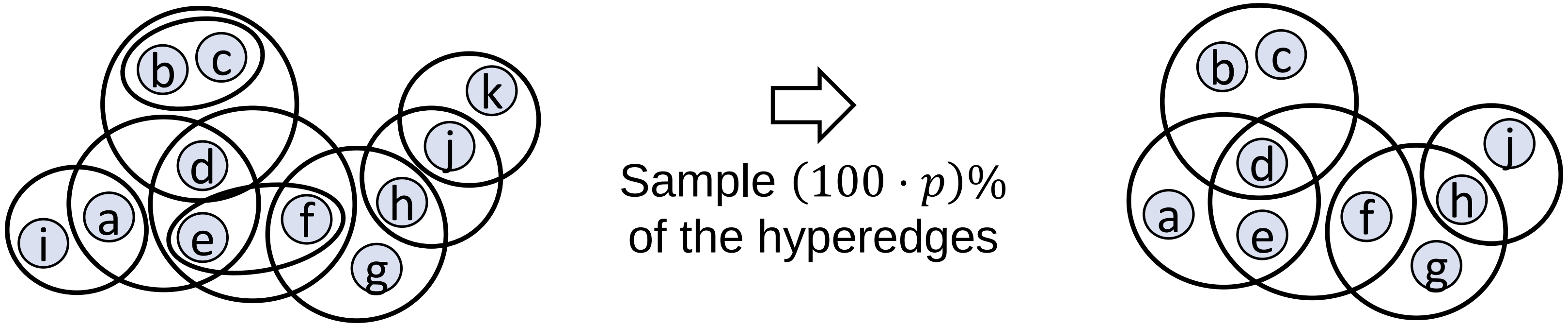
Hypergraph Representative Sampling

Question:

From a large hypergraph $\mathcal{G}$, how to generate a **small sub-hypergraph** $\widehat{\mathcal{G}}$ that preserves the structural properties?

Answer:

We **sample** *representative* hyperedges from $\mathcal{G}$ to generate $\widehat{\mathcal{G}}$ by the proposed method **MiDaS**.



Hypergraph Representative Sampling (cont.)

?

Question:

What is a **representative** sample?

Answer:

We compare sampled and entire hypergraphs using **10 structural properties**.

P1. Degree	P5. Singular Values	P8. Density
P2. Pair Degree	P6. Connected Component Size	P9. Overlapness
P3. Size	P7. Global Clustering Coefficient	P10. Effective Diameter
P4. Intersection Size		

Node-level, hyperedge-level, and hypergraph-level structural properties

!

Hypergraph Representative Sampling (cont.)

?

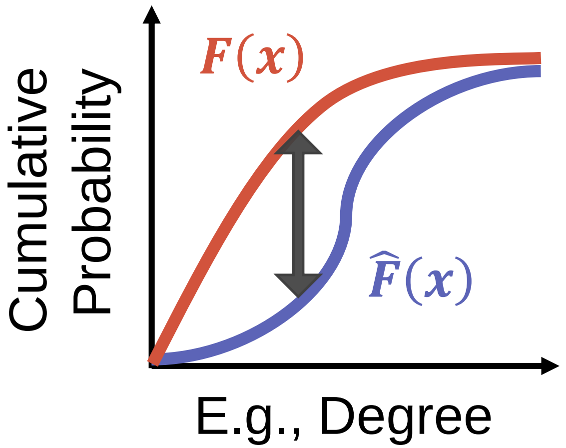
Question:

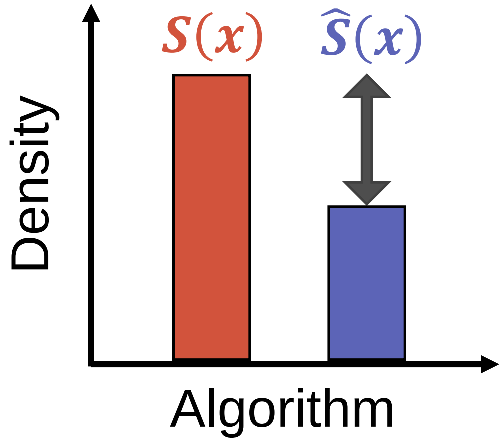
How to measure the similarity between $\mathcal{G}$ and $\hat{\mathcal{G}}$?

Answer 1:

- For probability functions (P1 – P6), we use **D-statistics**.
- For scalar values (P7 – P10), we use **relative difference**.

!





Hypergraph Representative Sampling (cont.)

?

Question:

How to measure the similarity between $\mathcal{G}$ and $\widehat{\mathcal{G}}$?

Answer 2:

To compare qualities of different scales, we average the ten distances by **rankings** and **Z-scores**.

	Size	Density
$\widehat{\mathcal{G}}_1$	0.2	7
$\widehat{\mathcal{G}}_2$	0.01	13
$\widehat{\mathcal{G}}_3$	0.02	1

Distances from $\widehat{\mathcal{G}}$

	Size	Density	Avg.
$\widehat{\mathcal{G}}_1$	3	2	2.5
$\widehat{\mathcal{G}}_2$	**1**	3	2
$\widehat{\mathcal{G}}_3$	2	**1**	**1.5**

Ranking

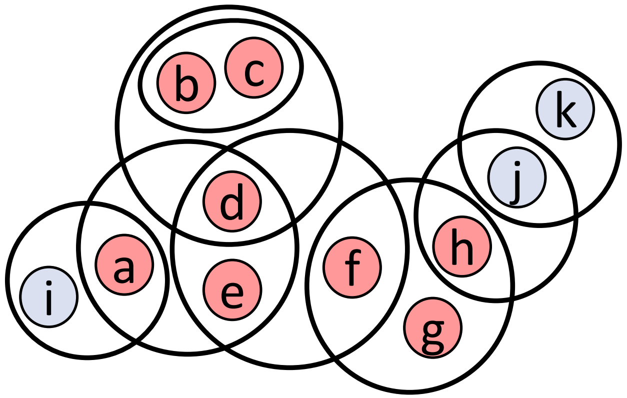
	Size	Density	Avg.
$\widehat{\mathcal{G}}_1$	1.6	0	0.8
$\widehat{\mathcal{G}}_2$	**-0.7**	1.2	0.25
$\widehat{\mathcal{G}}_3$	-0.6	**-1.2**	**-0.9**

Z-Score

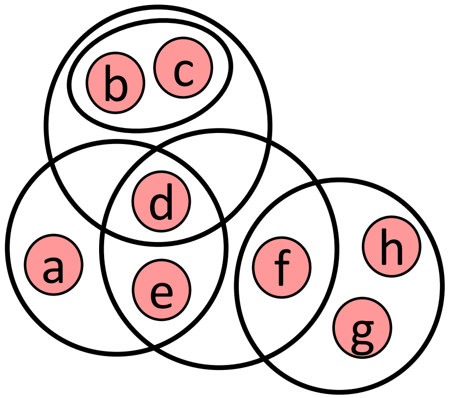
!

Simple and Intuitive Approaches

- **Node selection (NS)** chooses a subset of nodes and returns the induced sub-hypergraph.



Induced sub-hypergraph of {a,b,c,d,e,f,g,h}



- **Hyperedge selection (HS)** chooses a subset of hyperedges.

Simple and Intuitive Approaches (cont.)

- **Node selection (NS)** chooses a subset of nodes and returns the induced sub-hypergraph.

RNS	Random Node Sampling	draw a node **uniformly** at random
RDN	Random Degree Node	draw a node with probabilities proportional to node degrees
RW	Random Walk	random walk with restart on clique-expansion
FF	Forest Fire	forest fire in hypergraphs as in HyperFF

M. Choe, J. Yoo, G. Lee, W. Baek, U Kang, K. Shin. "MiDaS: Representative Sampling from Real-world Hypergraphs", **WWW 2022**

Simple and Intuitive Approaches (cont.)

- **Hyperedge selection (HS)** chooses a subset of hyperedges

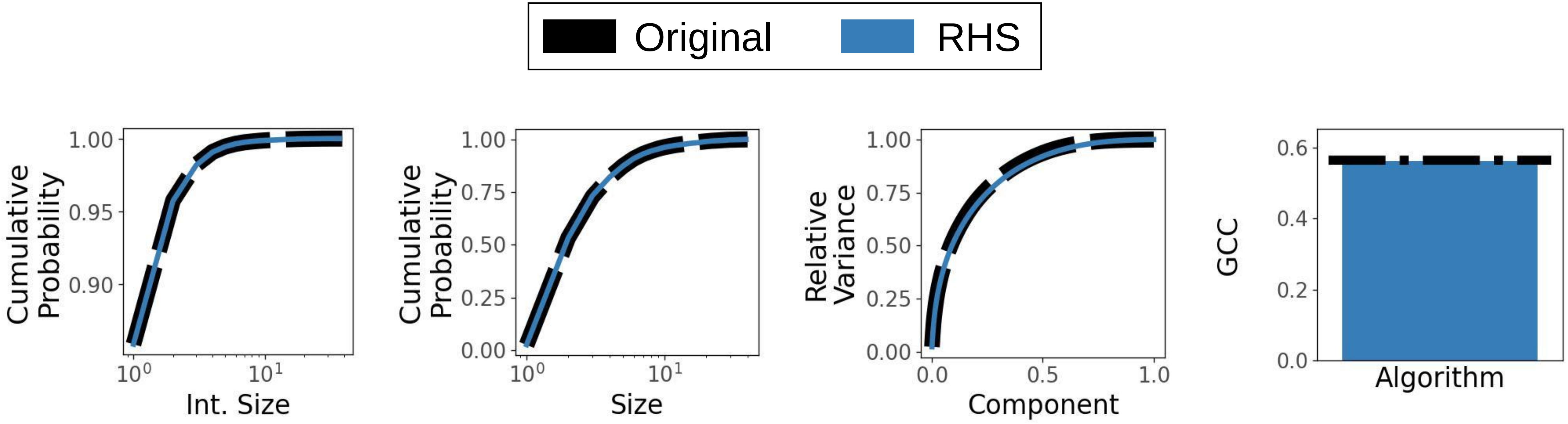
Focus!

RHS	Random Hyperedge Sampling	draw a target number of hyperedges **uniformly** at random
TIHS	Totally-Induced Hyperedge Sampling	extend totally-induced edge sampling to hypergraphs

M. Choe, J. Yoo, G. Lee, W. Baek, U Kang, K. Shin. "MiDaS: Representative Sampling from Real-world Hypergraphs", **WWW 2022**

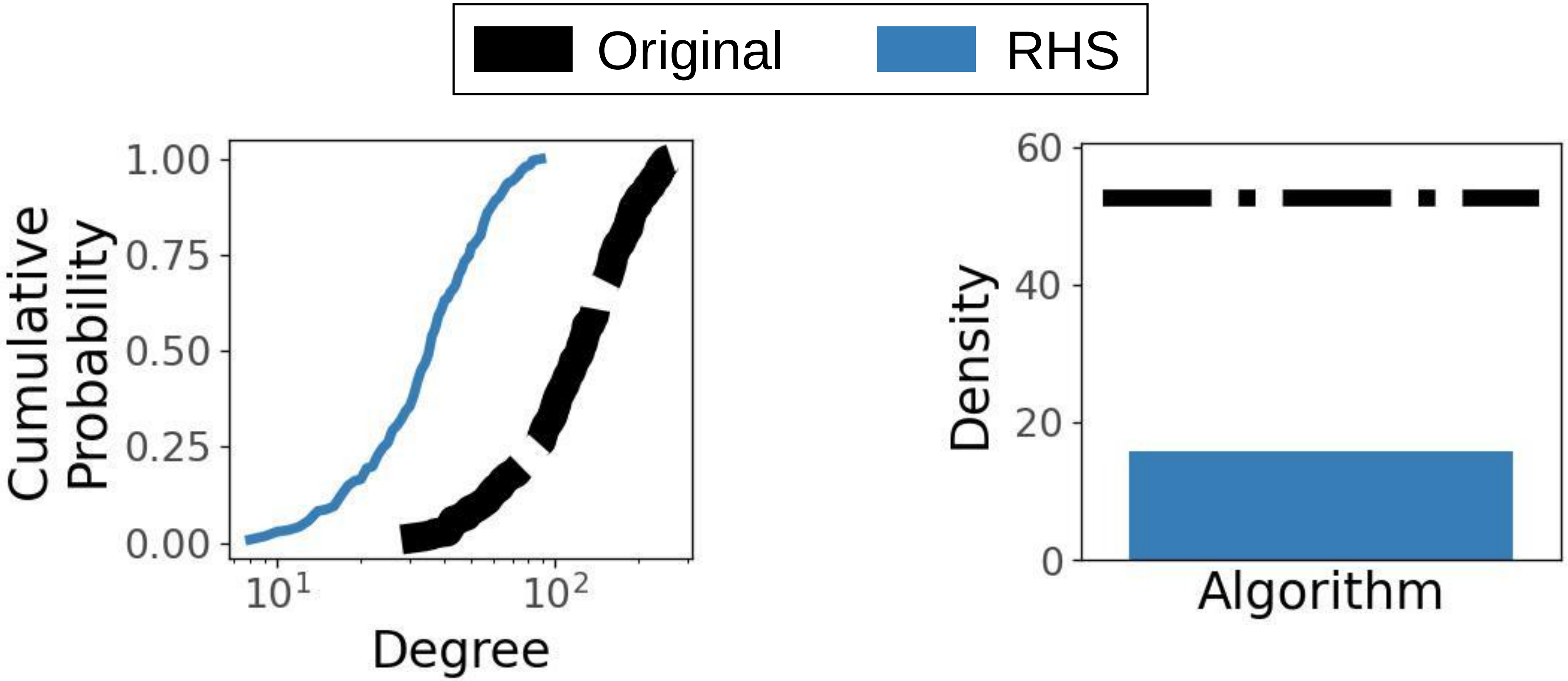
Random Hyperedge Sampling: Pros

- **RHS** well-preserves many structural properties.



Random Hyperedge Sampling: Cons

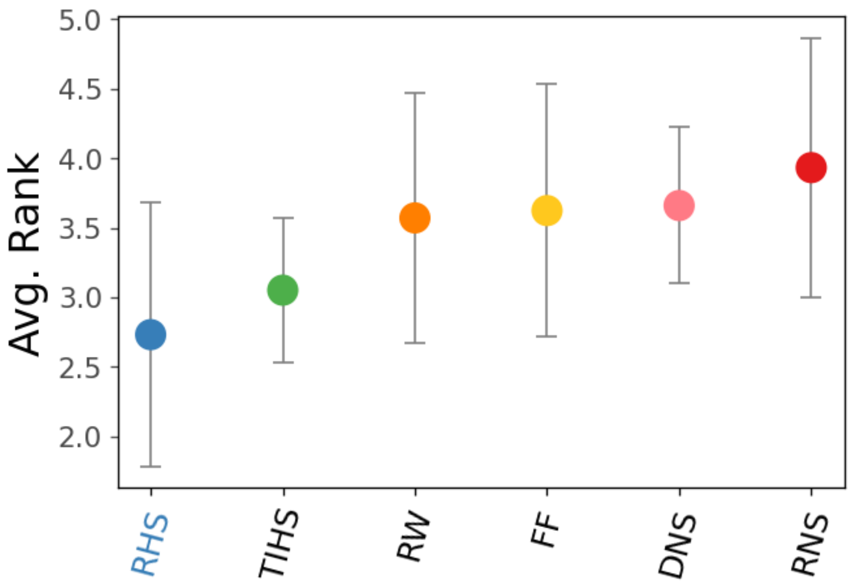
- **RHS** derives weakly connected sub-hypergraph.

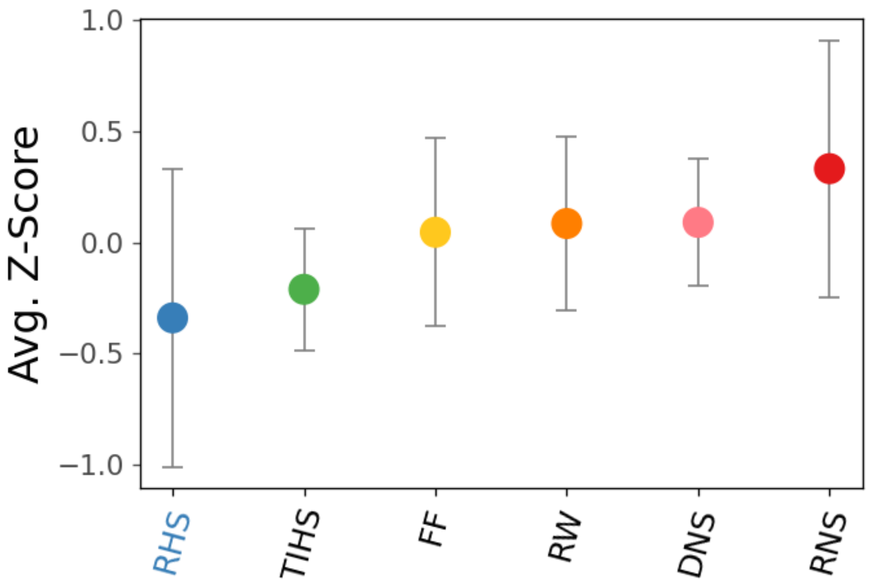


M. Choe, J. Yoo, G. Lee, W. Baek, U Kang, K. Shin. "MiDaS: Representative Sampling from Real-world Hypergraphs", **WWW 2022**

MiDaS: Intuition

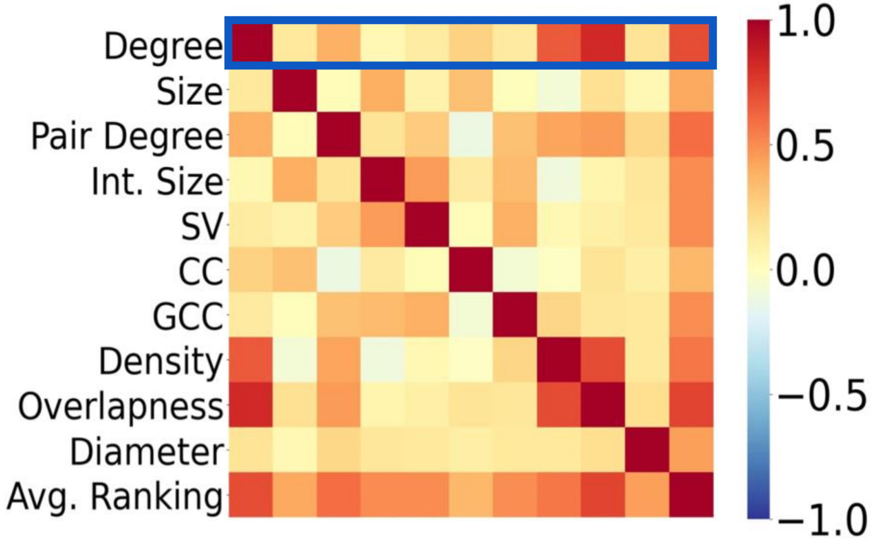
- **RHS** performs best overall, but its samples suffer from weak connectivity, including lack of high-degree nodes.



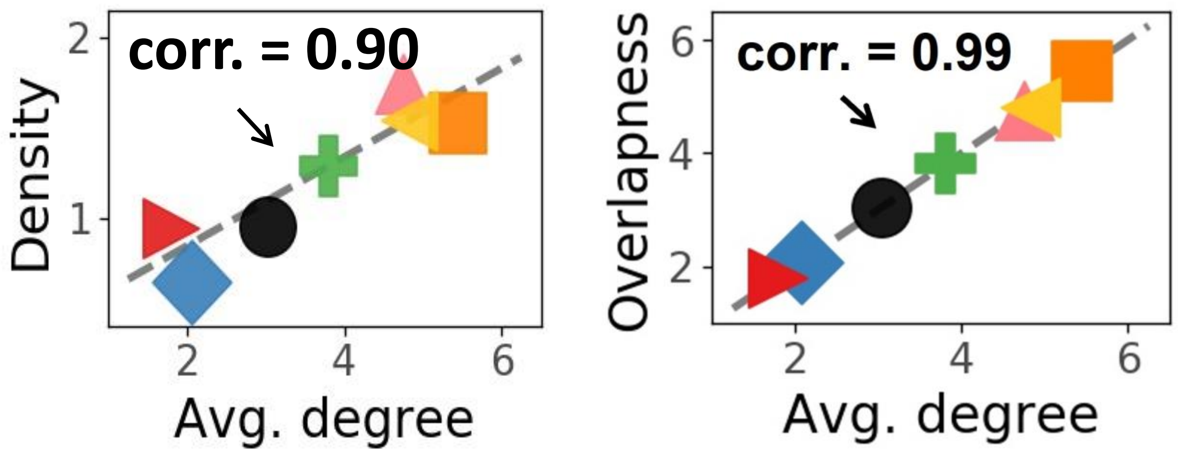


MiDaS: Intuition (cont.)

- **Degree preservation** is strongly correlated with the abilities to preserve other properties and thus the overall performance.



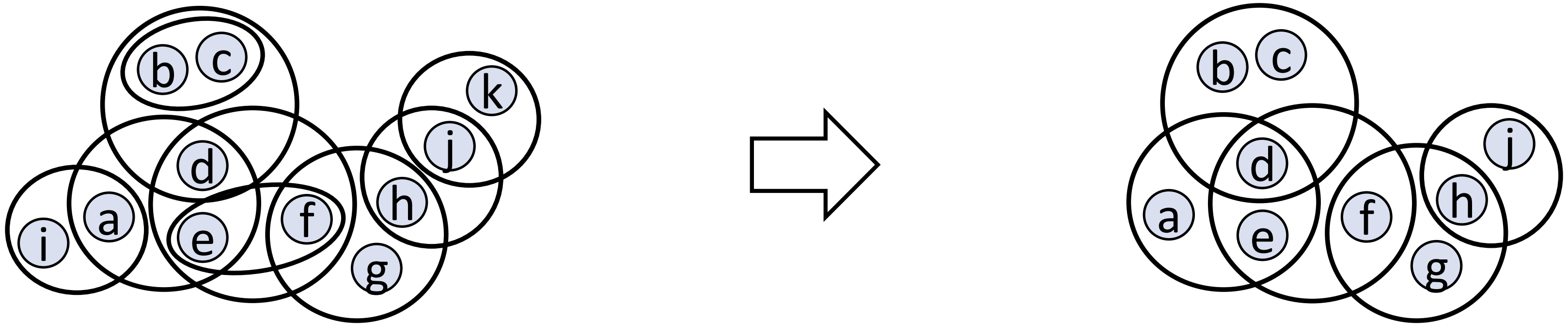
Pearson correlation coefficients between rankings w.r.t. P1 – P10



Pearson correlation (a) the average degree and (b) density and overlapness

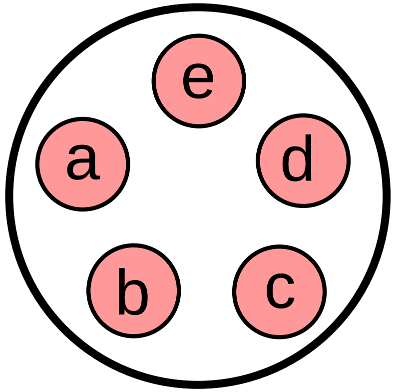
MiDaS: Intuition (cont.)

- Analyzing the simple approaches motivates to come up with **MiDaS**:
  - Aim to overcome the lack of high-degree nodes in RHS.
  - Focus on better preserving degree distribution.



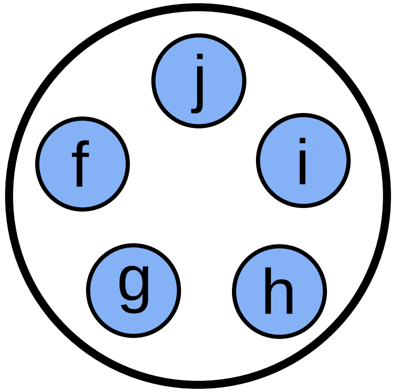
MiDaS-Basic: Preliminary Version

- To increase the fraction of high-degree nodes, **prioritize** hyperedges composed only of **high-degree nodes**.



Node (v)	Degree (d_v)
a	4
b	2
c	3
d	8
e	6

<

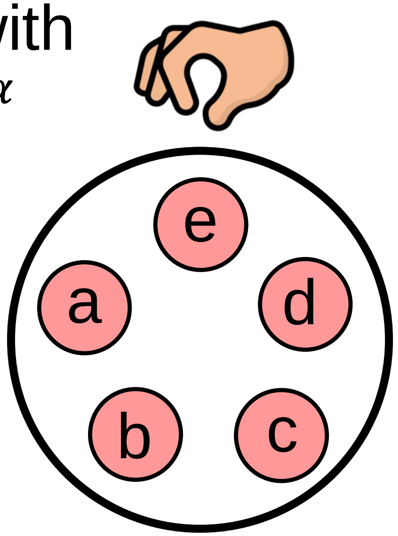


Node (v)	Degree (d_v)
a	4
b	2
c	3
d	8
e	6

MiDaS-Basic: Preliminary Version (cont.)

- Sampling a target number of hyperedges with probability proportional to the **minimum degree of nodes** in each hyperedge to the power of α

Sampling this hyperedge with probability $\propto (\min_{v \in e} d_v)^{\alpha}$



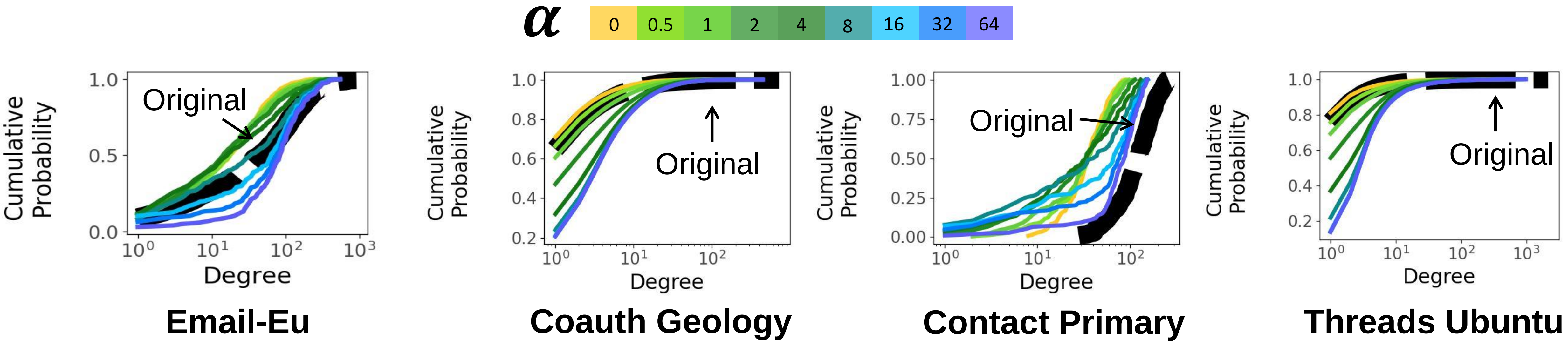
Node	Degree
a	4
b	2
c	3
d	8
e	6

Details

MiDaS-Basic: Empirical Properties

Observation 1

As α **increases**, the degree distributions in samples tend to be **more biased** toward high-degree nodes.
→ The bias in degree distributions can be controlled by α.

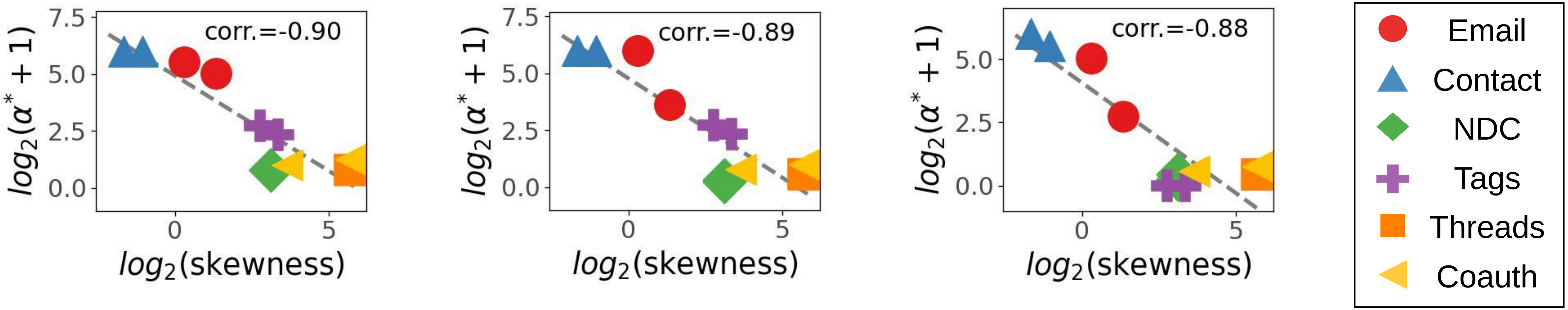


Details

MiDaS-Basic: Empirical Properties (cont.)

Observation 2

As degree distributions in original hypergraphs are **more skewed**, **larger α** values are required to preserve the distributions.



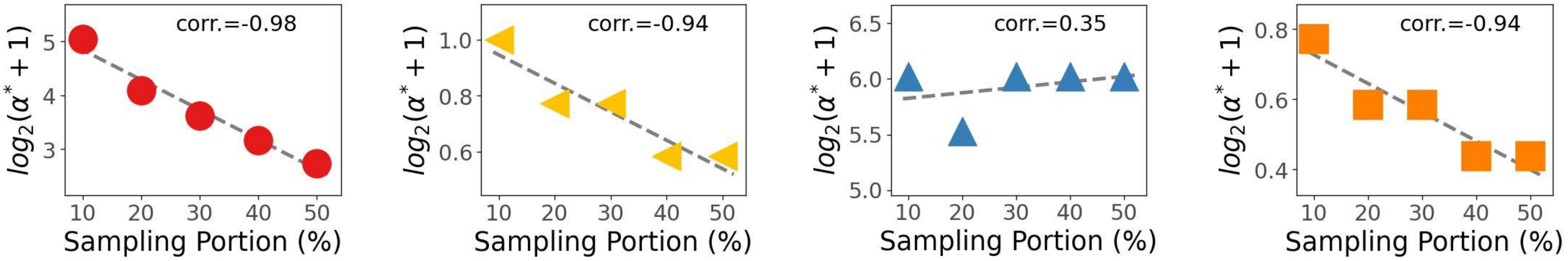
Sampling 10% **Sampling 30%** **Sampling 50%**

Details

MiDaS-Basic: Empirical Properties (cont.)

Observation 3

As we sample **fewer hyperedges**, **larger** α values are required to preserve degree distributions.



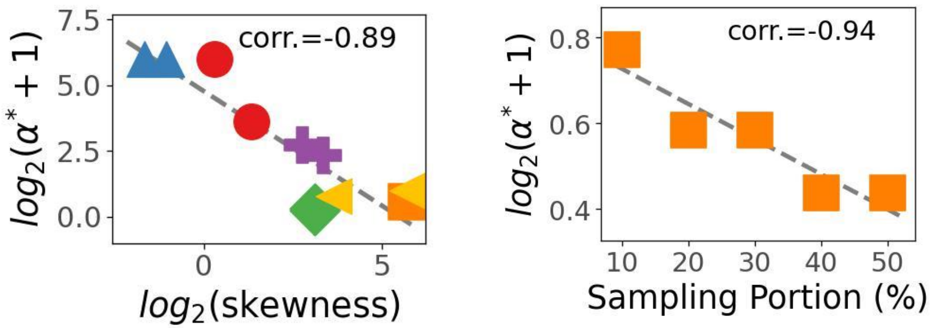
Email-Eu **Coauth Geology** **Contact Primary** **Threads Ubuntu**

MiDaS: Full-Fledged Version

- **MiDaS** is the final sampling algorithm that **automatically tunes** α.
- Based on the strong correlations in **Observations 2 & 3**, the best-performing α can be expected from skewness and sampling portions.

Skewness s

Sampling Portion p



Best-performing α
α^*

Given | **Correlations observed** | **Tuned parameter**

MiDaS: Full-Fledged Version (cont.)

- **MiDaS** fits **a linear regressor** $\boldsymbol{M}$ that fits (a) & (b) to (c):
  a. the skewness of the degree distribution of the input hypergraph
  b. the sampling portion
  c. a best-performing α value

Skewness s

Sampling Portion p

Given

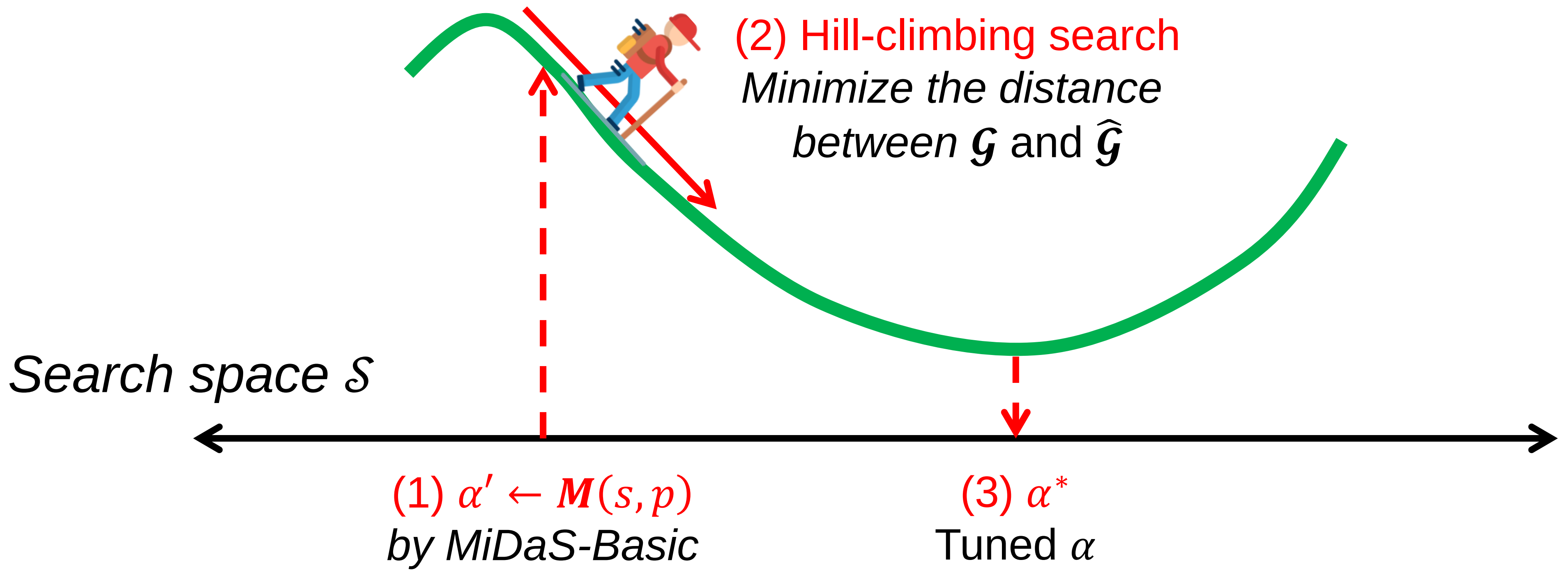
Linear Regression Model $\boldsymbol{M}$

Expected Best-performing α
α^*

Tuned parameter

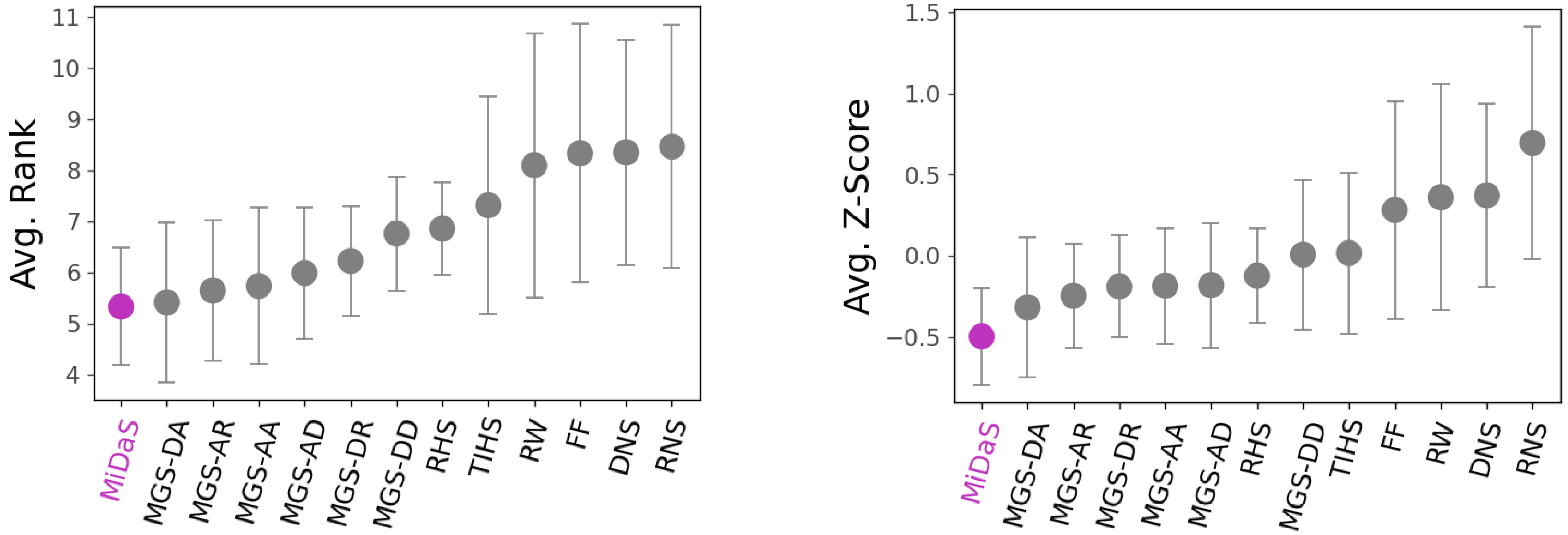
MiDaS: Full-Fledged Version (cont.)

- The $\boldsymbol{\alpha}$ value obtained by the linear regression model $\boldsymbol{M}$ is further tuned using **hill climbing**.



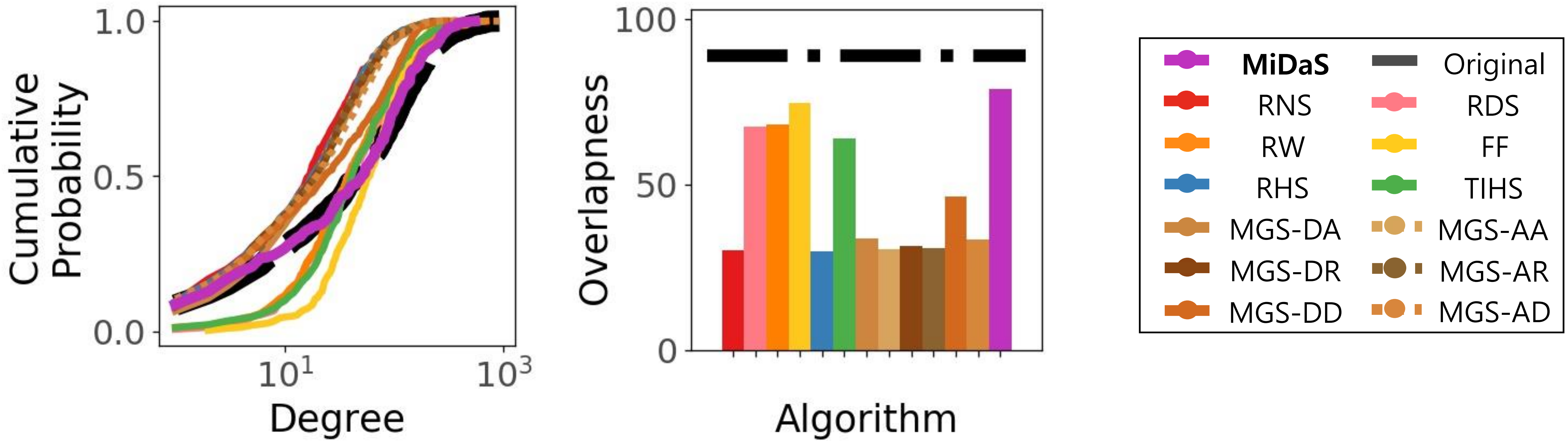
MiDaS: Evaluation

- **MiDaS** provides overall the most representative samples in terms of both average rankings and Z-scores.



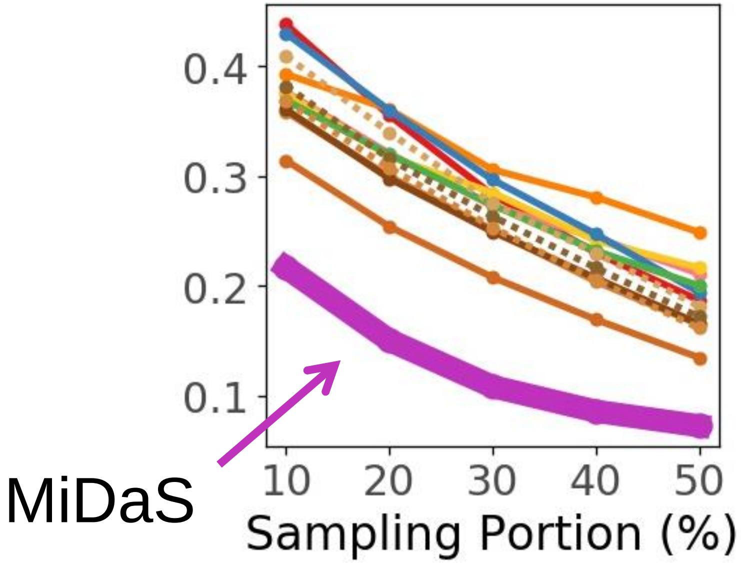
MiDaS: Evaluation (cont.)

- Especially, **MiDaS** best preserves node degrees, density, overlapness, and effective diameter.

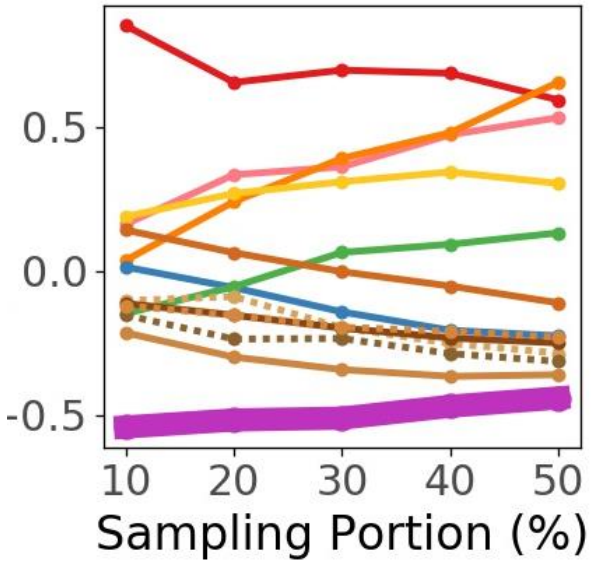


MiDaS: Evaluation (cont.)

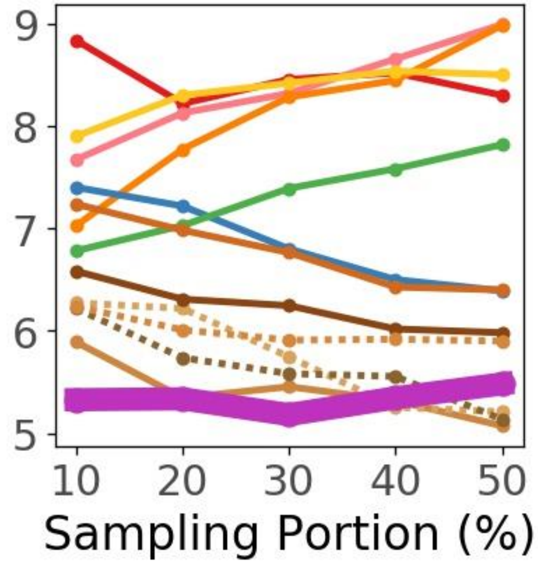
- **MiDaS** is consistently best regardless of sampling portions in degree distributions, average rankings and Z-scores.



Distance in degree distributions



Avg. Z-scores



Avg. rankings

Roadmap

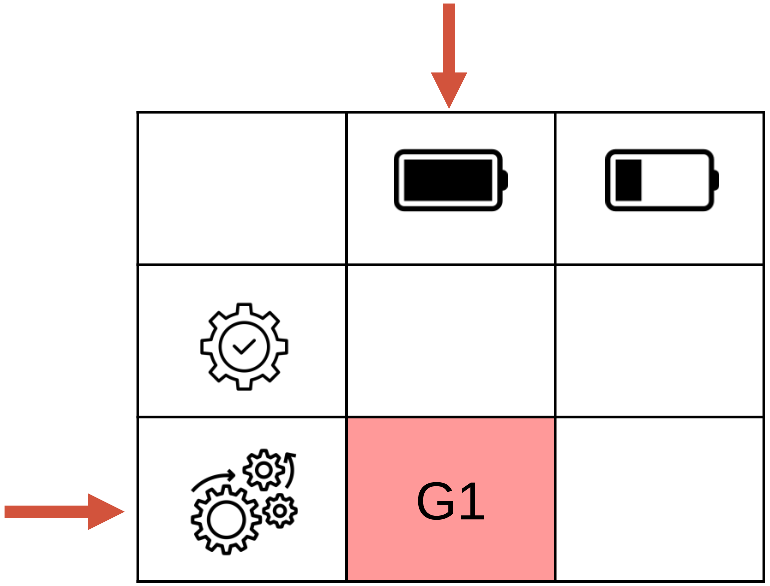
- **Part 1. Static Structural Patterns**
  - Basic Patterns
  - Advanced Patterns
- **Part 2. Dynamic Structural Patterns**
  - Basic Patterns
  - Advanced Patterns
- **Part 3. Generative Models**
  - Static hypergraph Generator
  - Dynamic hypergraph Generator **<<**

Part 3-2. Dynamic Hypergraph Generative Models

		Part 3. Generative Models
Static Models	Full-Hypergraphs	C20, LCS21
	Sub-Hypergraphs	CYLBKS22
Dynamic Models	Full-Hypergraphs	DYHS20, KKS20
	Sub-Hypergraphs	BKT18, CK21

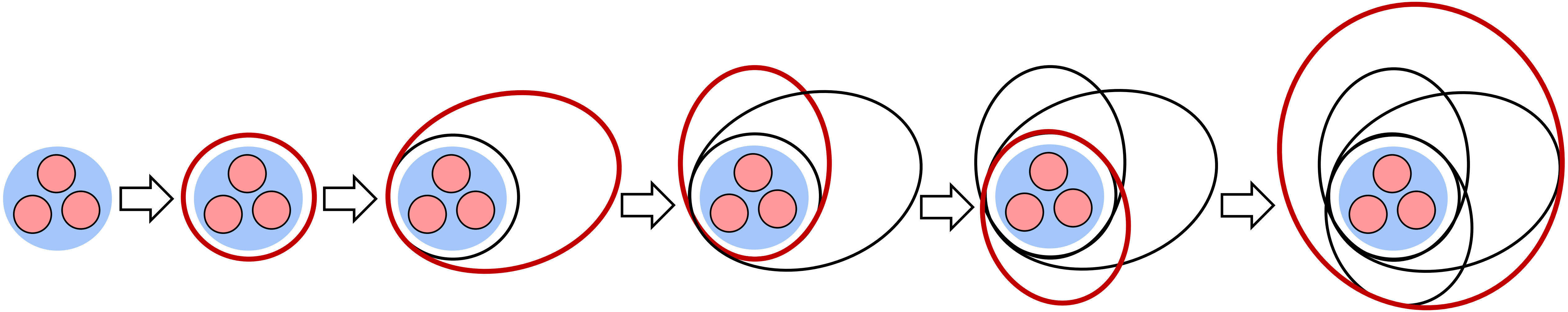
DYHS20: Dynamic Full-Hypergraph Generator

- **G1.** HyperPA: **Hyper**graph **P**referential **A**ttachment



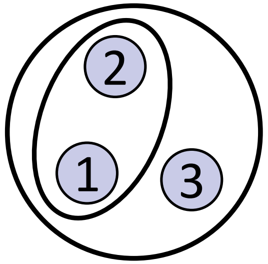
HyperPA: Preferential Attachment

- Main idea: **“Subsets get rich together”**
  - Groups of nodes appear with **probability** $\propto$ **“group degrees**.”



HyperPA: Preferential Attachment (cont.)

Step 1. Hyperedge Generation



{1}: 2 {2}: 2 {3}: 1
{1,2}: 2 {2,3}: 1 {3,1}: 1
{1,2,3}: 1

Group Degrees

Sample group prob. ∝ degree

A new **node 4** is added.

↓ *Number of hyperedges*

Add **2 hyperedges**.

↓ *Size of the* ***first*** *hyperedge*

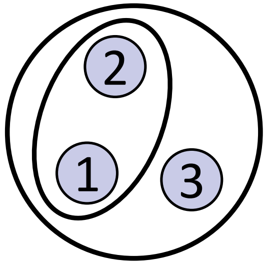
First hyperedge size: **2**

↓ *Sample groups*

Generate hyperedge: **{2,4}**

HyperPA: Preferential Attachment (cont.)

Step 1. Hyperedge Generation



{1}: 2 {2}: 2 {3}: 1

{1,2}: 2 {2,3}: 1 {3,1}: 1

{1,2,3}: 1

Group Degrees

Sample group prob. ∝ degree

A new **node 4** is added.

Number of hyperedges

Add **2 hyperedges**.

*Size of the **second** hyperedge*

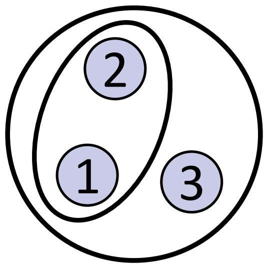
Second hyperedge size: **3**

Sample groups

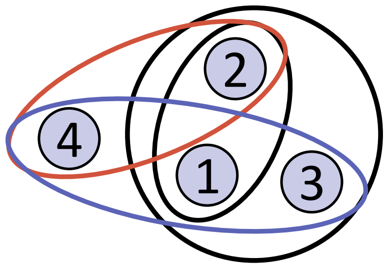
Generate hyperedge: **{1,3,4}**

HyperPA: Preferential Attachment (cont.)

Step 2. Hypergraph Update



Add 2 hyperedges:
{2,4}
{1,3,4}



{1}: 2 {2}: 2 {3}: 1
{1,2}: 2 {2,3}: 1 {3,1}: 1
{1,2,3}: 1

Group Degrees

Update **group degrees**

{1}: 3 {2}: 3 {3}: 2 {4}: 2
{1,2}: 2 {2,3}: 1 {3,1}: 2
{4,1}: 2 {4,2}: 1 {4,3}: 1
{1,2,3}: 1 {1,3,4}: 1

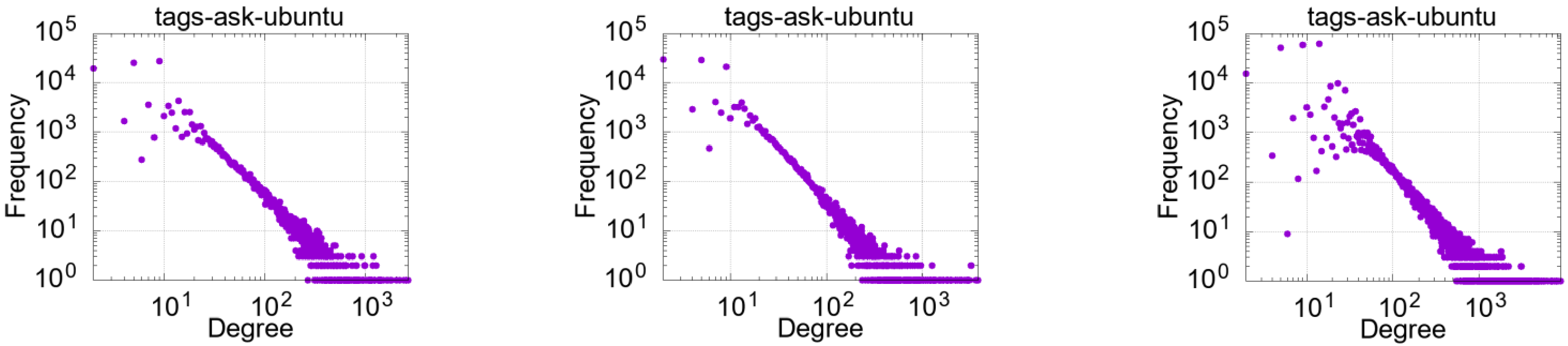
Group Degrees

For all nodes

…

HyperPA: Evaluation

- **HyperPA** generates realistic hypergraphs w.r.t. <u>edge-level</u>.
  - HyperPA considers "group degrees."
  - **NaivePA** (baseline) considers node degrees individually.

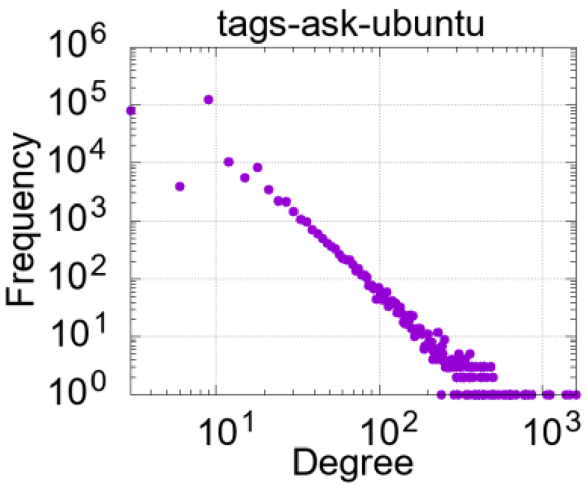


Real data **HyperPA** **NaivePA**

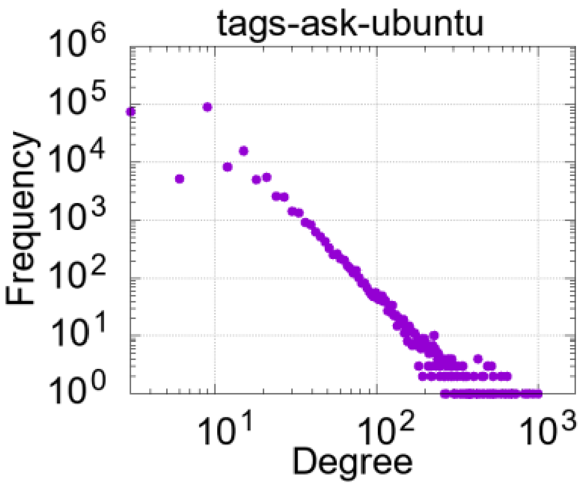
M. T. Do, S. Yoon, B. Hooi, K. Shin, "Structural Patterns and Generative Models of Real-world Hypergraphs", **KDD 2020**

HyperPA: Evaluation (cont.)

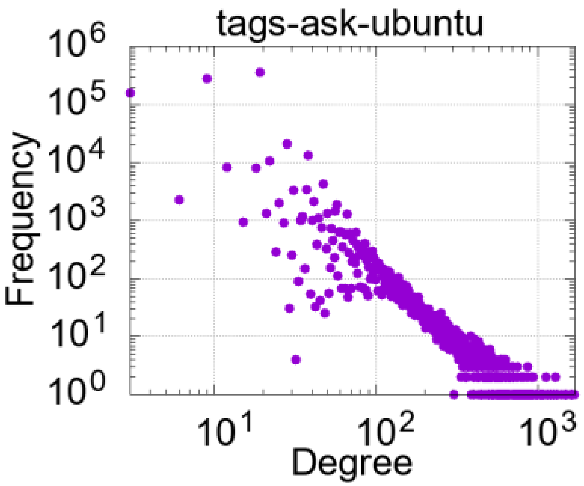
- **HyperPA** generates realistic hypergraphs w.r.t. triangle-level.
  - **HyperPA** considers "group degrees."
  - **NaivePA** (baseline) considers node degrees individually.



Real data



HyperPA

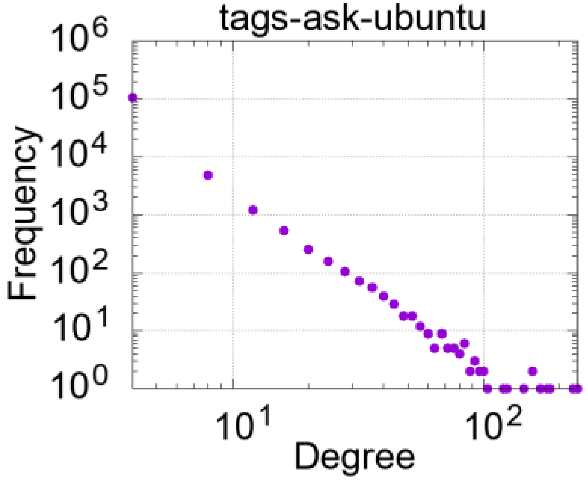


NaivePA

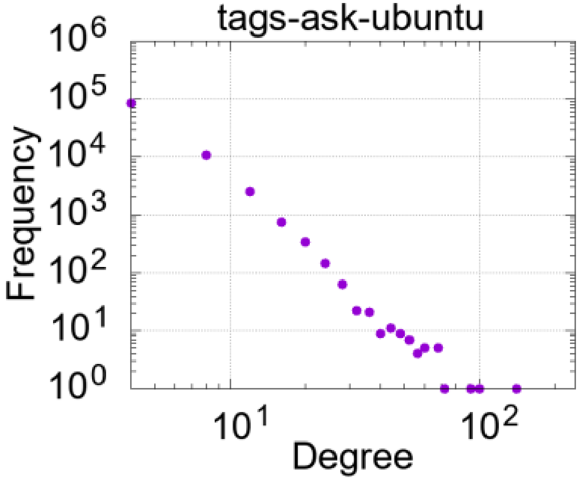
M. T. Do, S. Yoon, B. Hooi, K. Shin, "Structural Patterns and Generative Models of Real-world Hypergraphs", **KDD 2020**

HyperPA: Evaluation (cont.)

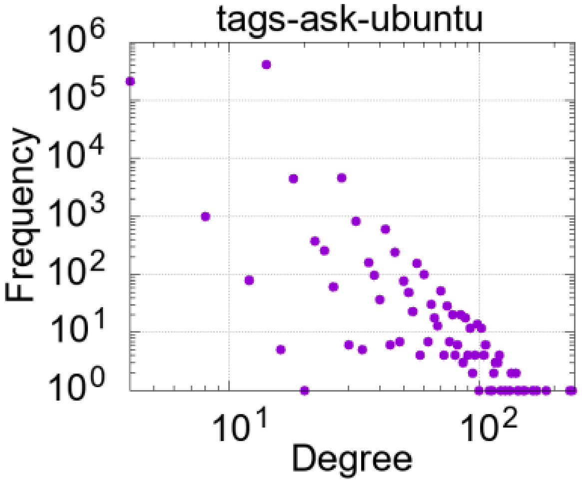
- **HyperPA** generates realistic hypergraphs w.r.t. 4-clique-level.
  - **HyperPA** considers "group degrees."
  - **NaivePA** (baseline) considers node degrees individually.



Real data



HyperPA

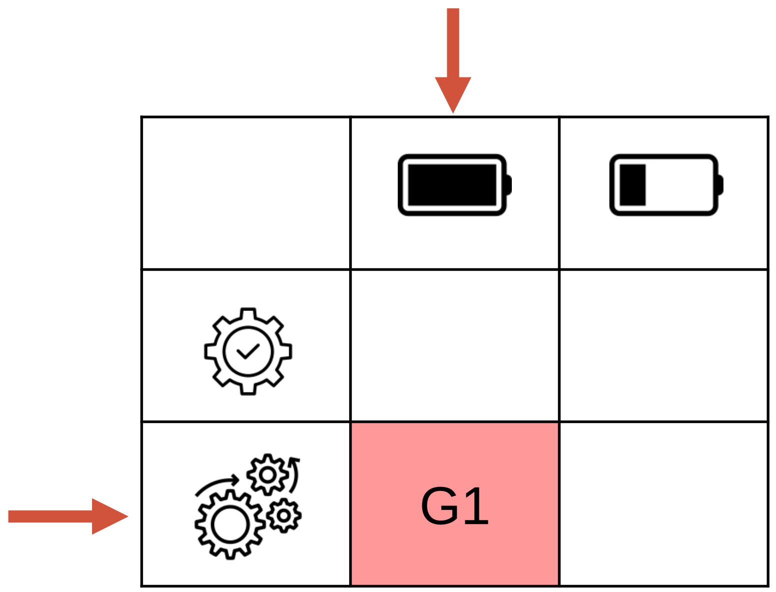


NaivePA

M. T. Do, S. Yoon, B. Hooi, K. Shin, "Structural Patterns and Generative Models of Real-world Hypergraphs", **KDD 2020**

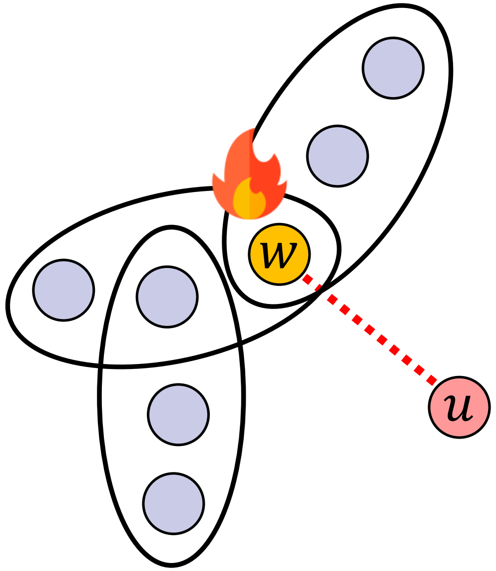
KKS20: Dynamic Full-Hypergraph Generator

- **G1.** HyperFF: **Hyper**graph **F**orest **F**ire



Y. Kook, J. Ko, K. Shin, "Evolution of Real-world Hypergraphs: Patterns and Models without Oracles", **ICDM 2020**

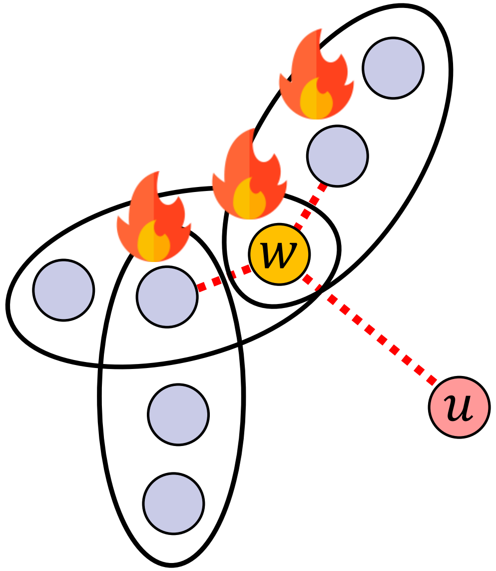
HyperFF: Forest Fire



Step 1-1
The new node $\boldsymbol{u}$ chooses a random **ambassador $\boldsymbol{w}$.**

Step 1-2
Burn the ambassador $\boldsymbol{w}$.

HyperFF: Forest Fire (cont.)



<u>Step 2-1</u>

$\boldsymbol{n}$ ← sample from the geometric distribution with mean $\frac{p}{1-p}$

Burning probability

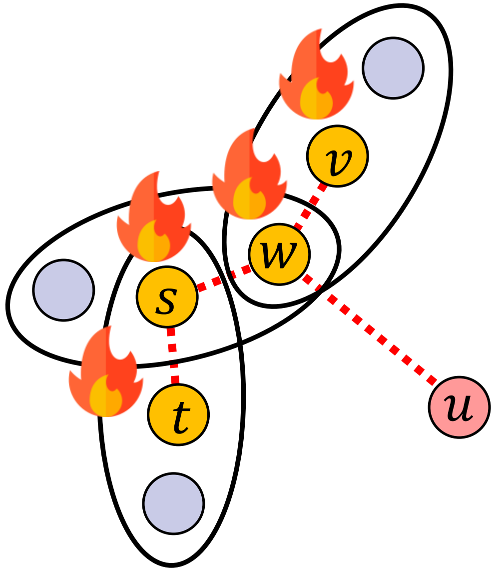
<u>Step 2-2</u>

Sample $\boldsymbol{n}$ neighbors of the ambassador $\boldsymbol{w}$ in the descending order of 'tie strength.'

<u>Step 2-3</u>

Burn the sampled $\boldsymbol{n}$ neighbors of the ambassador $\boldsymbol{w}$.

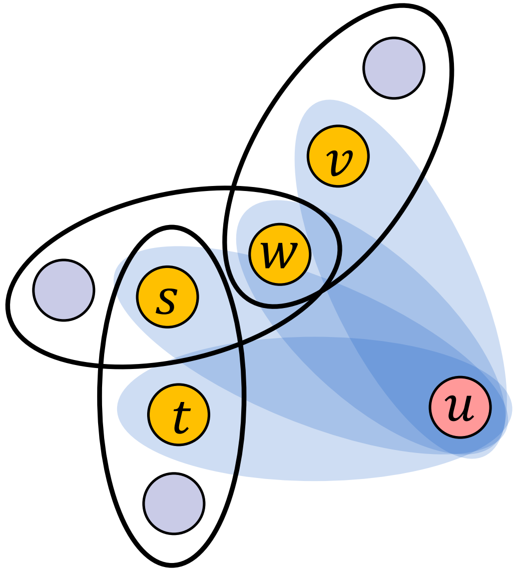
HyperFF: Forest Fire (cont.)



Step 3-1
View a burned neighbor as a **new ambassador**.

Step 3-2
Recursively apply Step 2 and burn neighbors of ambassadors.

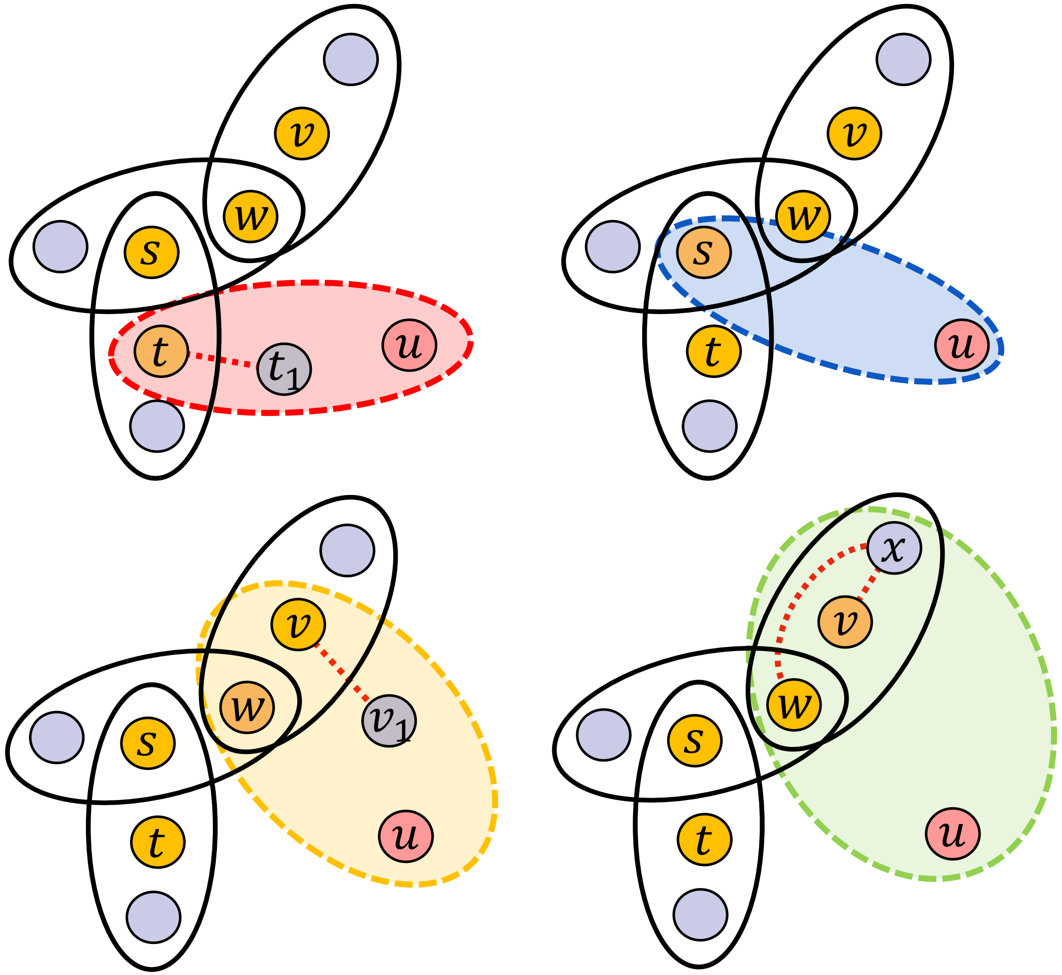
HyperFF: Forest Fire (cont.)



<u>Step 4-1</u>
Add size-2 hyperedges between node $\boldsymbol{u}$ and burned nodes.

<u>Step 4-2</u>
Increase 'tie strength' between node $\boldsymbol{u}$ and burned nodes by 1.

HyperFF: Forest Fire (cont.)



Step 5-1

Reset the burning history.

Step 5-2

For each burned node, start the burning process using the geometric distribution with mean $\frac{q}{1-q}$.

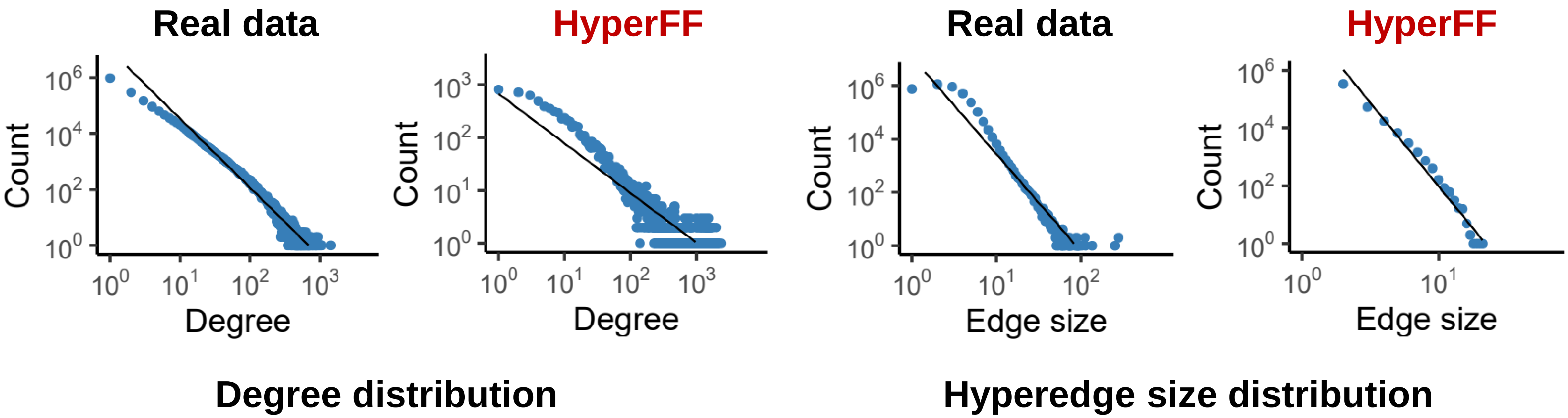
Expanding probability

Step 5-3

Expand the hyperedge until the process ends.

HyperFF: Evaluation

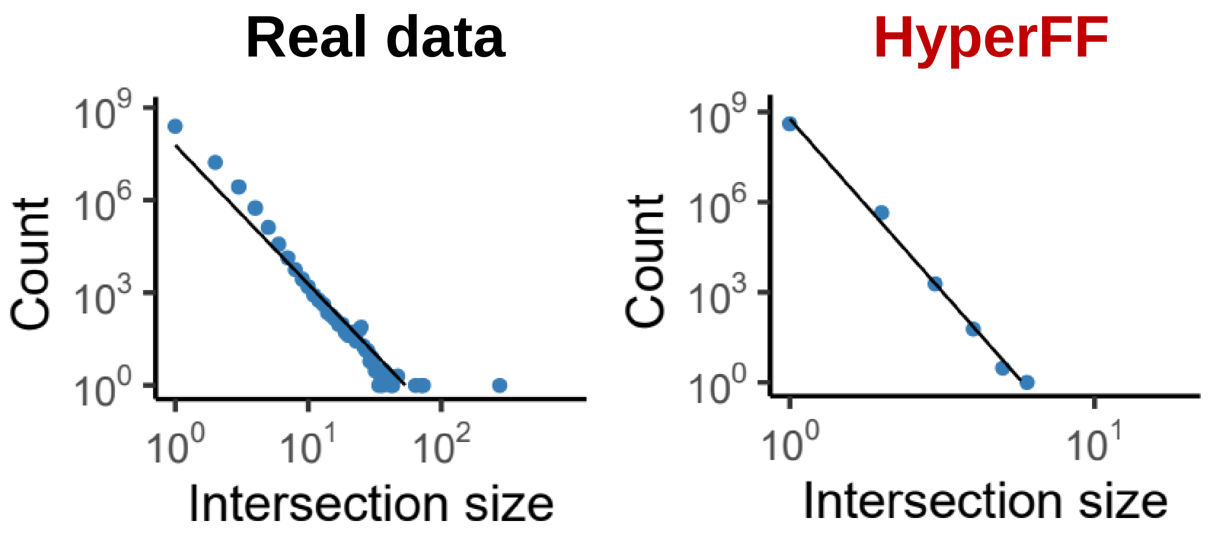
- **HyperFF** reproduces static structural patterns in real hypergraphs.



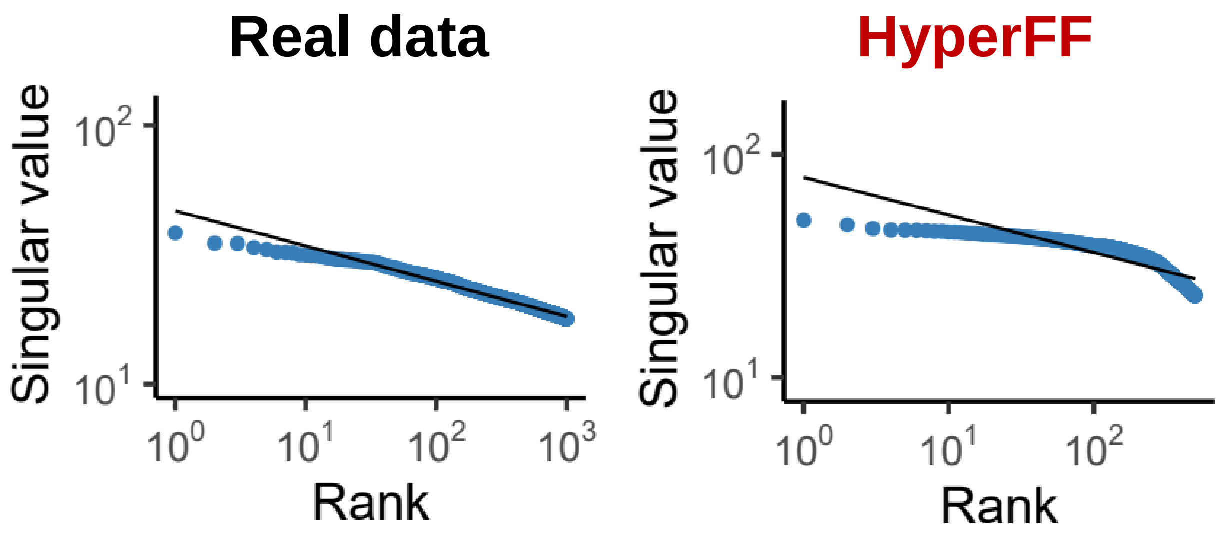
Y. Kook, J. Ko, K. Shin, “Evolution of Real-world Hypergraphs: Patterns and Models without Oracles”, **ICDM 2020**

HyperFF: Evaluation (cont.)

- **HyperFF** reproduces static structural patterns in real hypergraphs.



Intersection size distribution

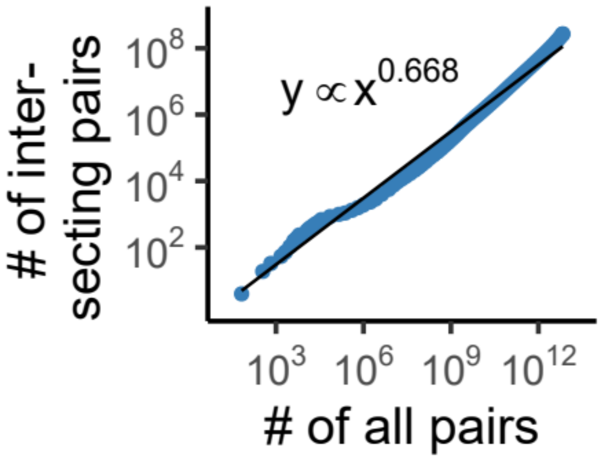


Singular value distribution

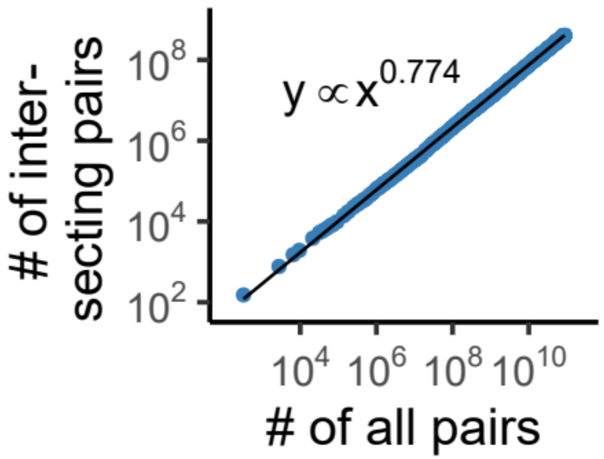
HyperFF: Evaluation (cont.)

- **HyperFF** reproduces dynamic structural patterns in real hypergraphs.

Real data

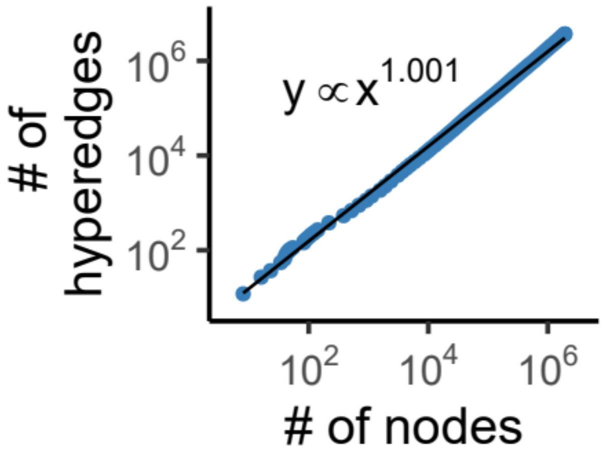


HyperFF

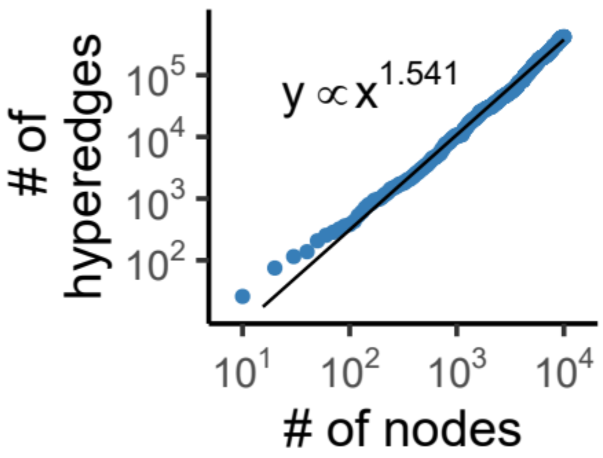


Diminishing overlaps

Real data



HyperFF

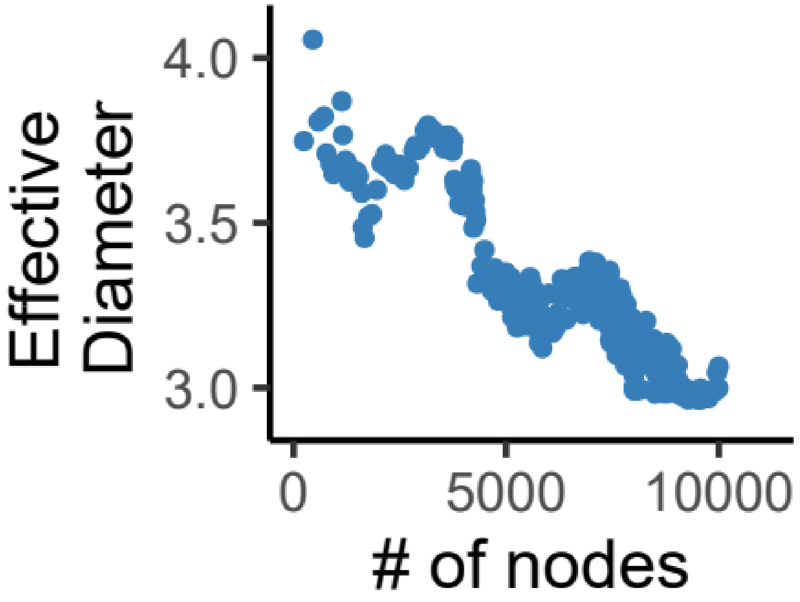


Increasing edge density

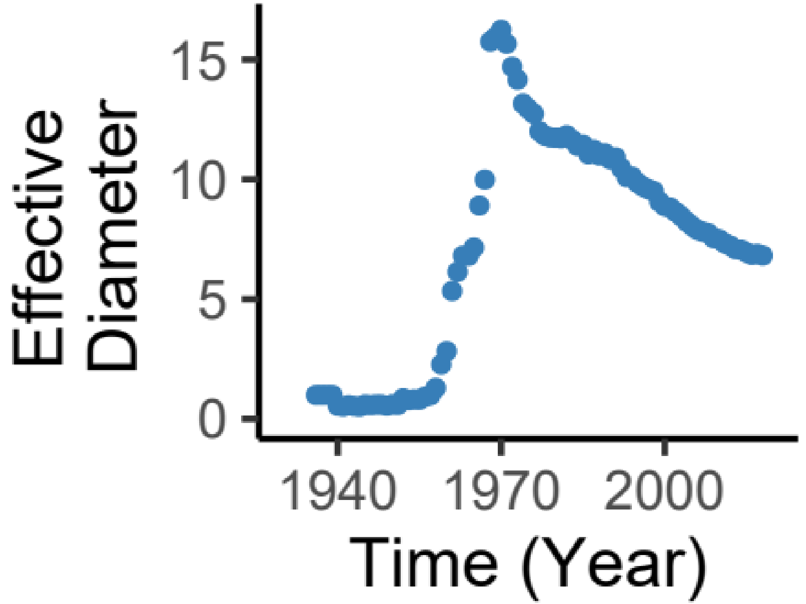
HyperFF: Evaluation (cont.)

- **HyperFF** reproduces dynamic structural patterns in real hypergraphs.

Real data



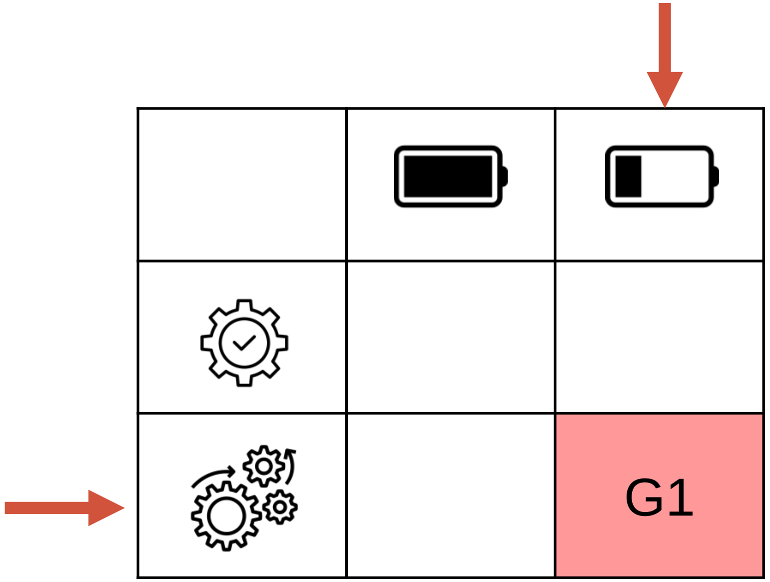
HyperFF



Shrinking diameter

BKT18: Dynamic Sub-Hypergraph Generator

- **G1.** Correlated Repeated Unions (CRU) model



Next Hyperedge Prediction

?

Question:

Given a sequence of temporal hyperedges (i.e., temporal hypergraph), how can we predict the **next hyperedge**?

Answer:

The CRU model predicts the next hyperedge based on three empirical observations: **(1) repeat behavior**, **(2) subset correlation**, and **(3) recency bias**.

!

A. R. Benson, R. Kumar, A. Tomkins, "Sequences of Sets", **KDD 2018**

Recap: Three Empirical Observations

Repeat Behavior

Temporal hyperedges tend to **repeat** previous ones.

Subset Correlation

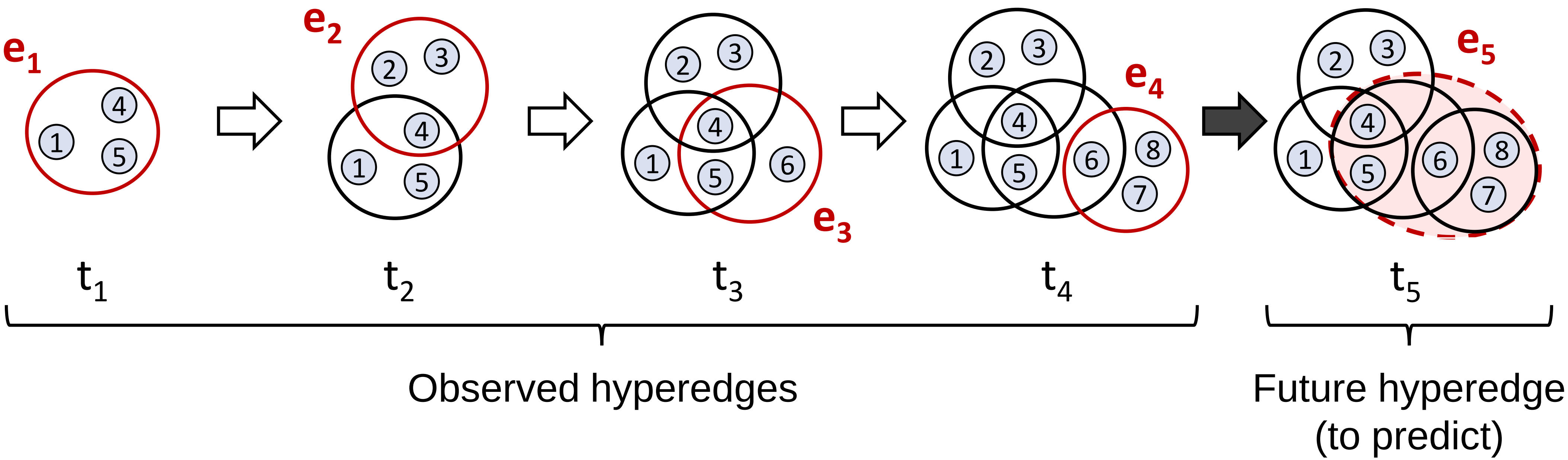
Subsets of nodes tend to be **correlated**.

Recency Bias

Temporal hyperedges tend to be similar to **recent** ones.

CRU: Correlated Repeated Unions

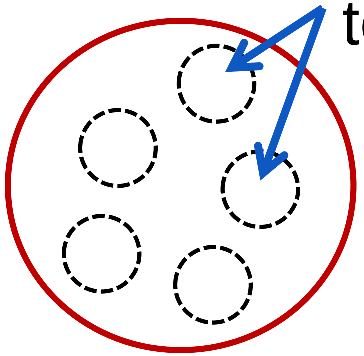
- To predict the next hyperedge e_{n+1}, **CRU** is given:
  - the size of the hyperedge $|e_{n+1}|$
  - the novel nodes in the hyperedge



CRU: Correlated Repeated Unions (cont.)

Step 0. Initialization

Number of nodes to be included: $\boldsymbol{r}$



The next hyperedge

$\boldsymbol{e_{n+1}} = \emptyset$

Recency weight vector $\boldsymbol{w}$

0.48	0.29	0.15	0.08

Correlation probability

$\boldsymbol{p} = 0.80$

Two parameters of CRU

CRU: Correlated Repeated Unions (cont.)

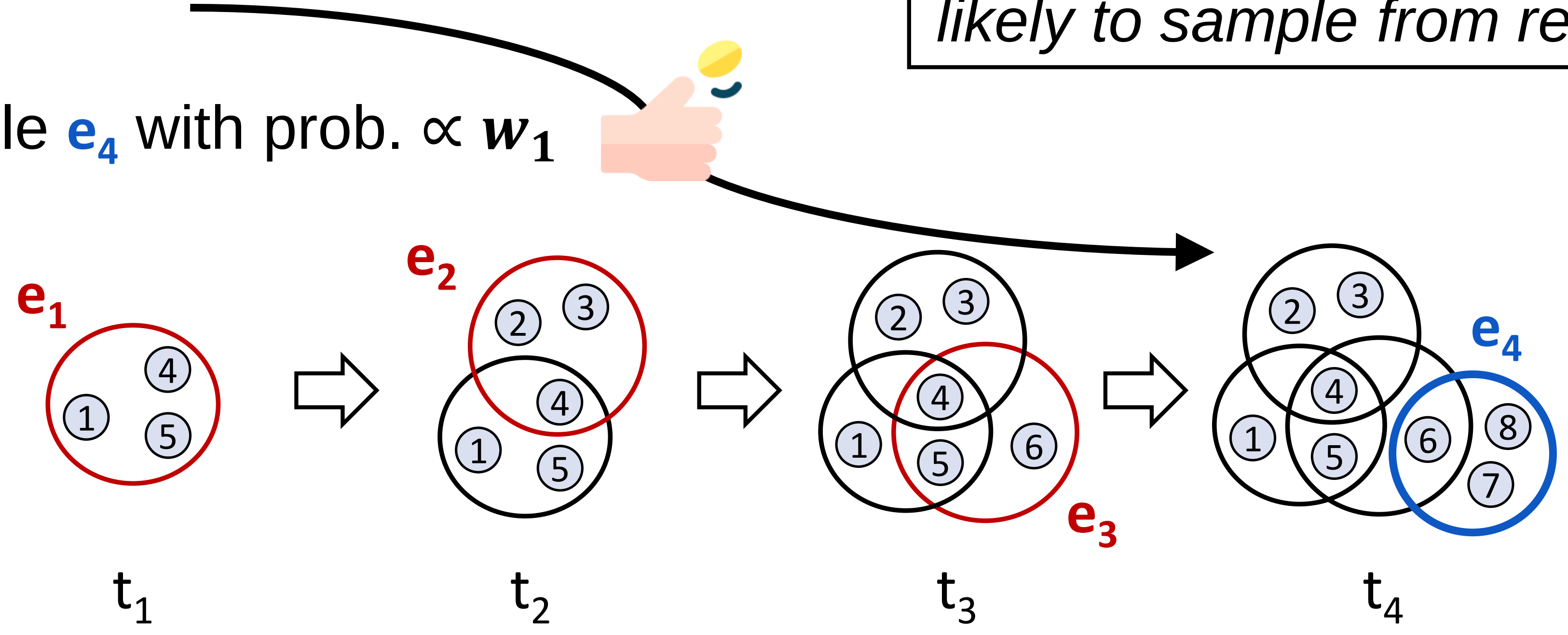
Step 1. Sample hyperedges

$\boldsymbol{w} =$

0.48	0.29	0.15	0.08

Intuition: $\boldsymbol{w}$ controls the **recency bias**.
Skewed $\boldsymbol{w}$ *toward smaller index → More likely to sample from recent hyperedges*

Sample $\mathbf{e_4}$ with prob. $\propto \boldsymbol{w_1}$

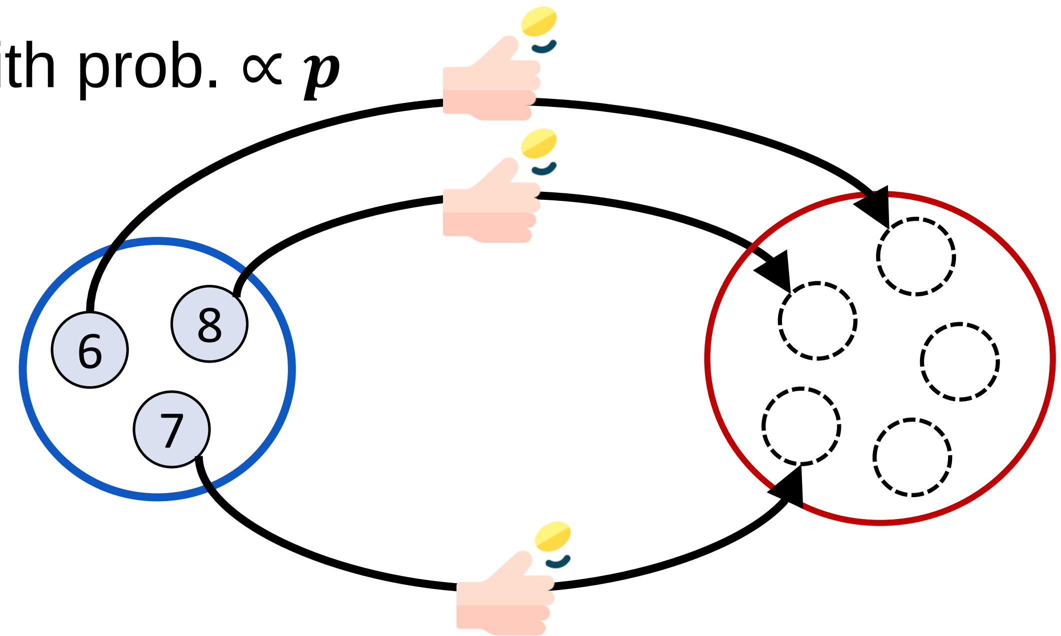


CRU: Correlated Repeated Unions (cont.)

Step 2. Sample nodes

Intuition: $\boldsymbol{p}$ controls the **subset correlation**.
A larger $\boldsymbol{p}$ → *More correlation in selecting items from the same hyperedge*

Sample node with prob. $\propto \boldsymbol{p}$

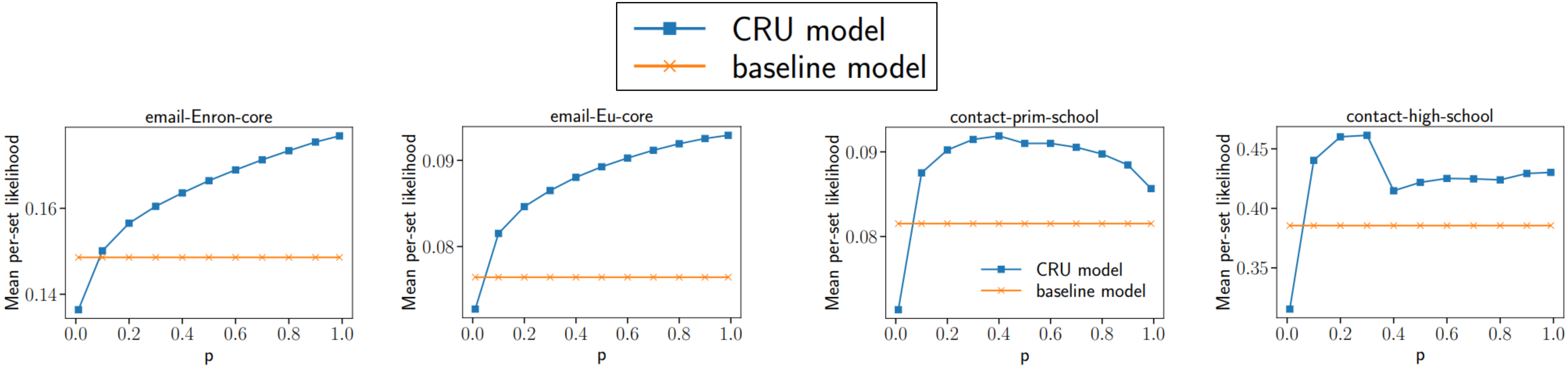


CRU: Learning Parameters

- Fix **correlation probability** $\boldsymbol{p}$ and learn **recency weight vector** $\boldsymbol{w}$.
  - $\boldsymbol{w}$ can be learned with **maximum likelihood estimation**.
  - Grid search over $\boldsymbol{p}$.

CRU: Evaluation

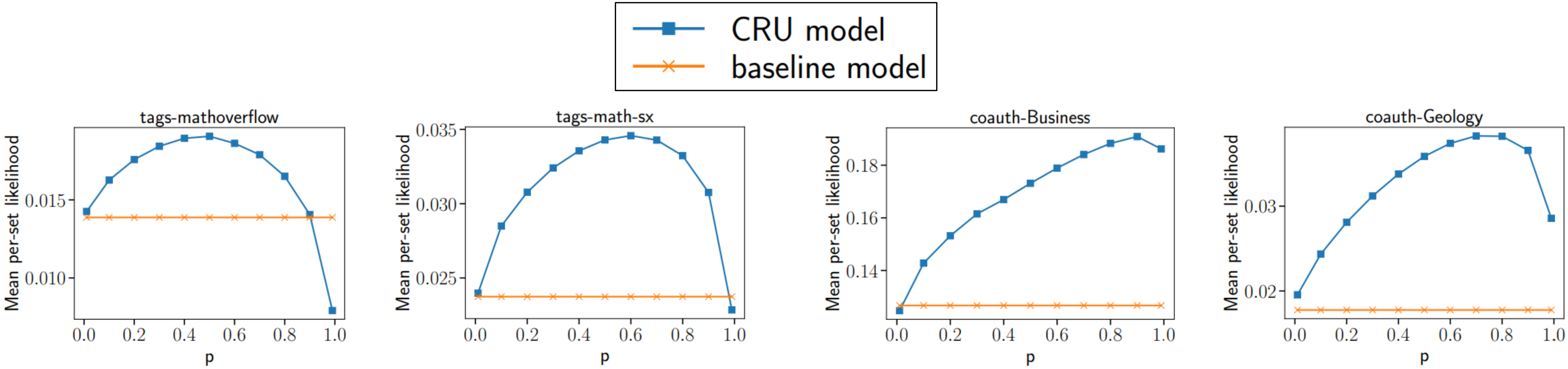
- The optimal **correlation probability** p is consistent within domain but differs between domains.



email domain **contact domain**

CRU: Evaluation (cont.)

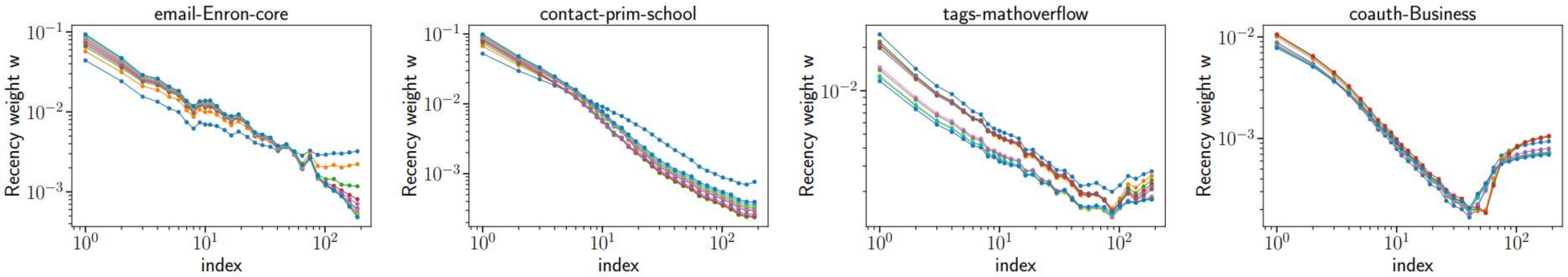
- The optimal **correlation probability** $\boldsymbol{p}$ is consistent within domain but differs between domains.



tags domain **coauth domain**

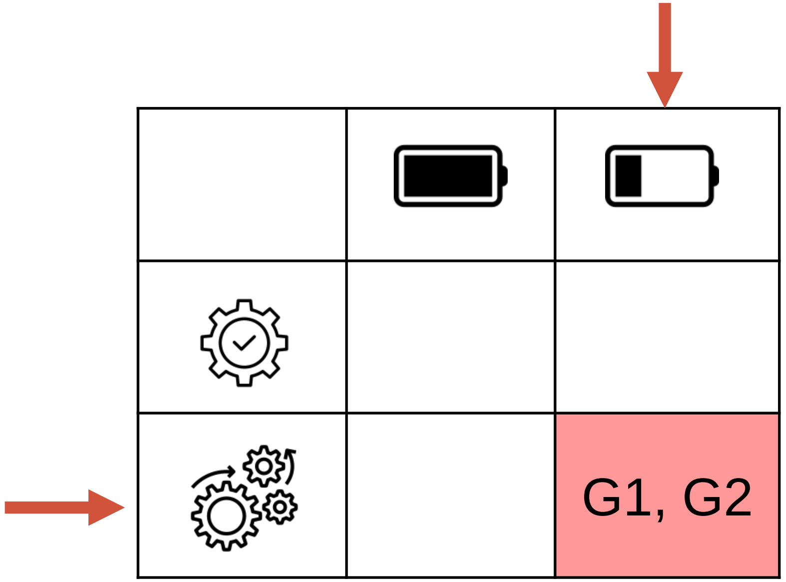
CRU: Evaluation (cont.)

- Learned **recency weights $\boldsymbol{w}$** tend to decrease monotonically, which agrees with recency bias observed in real hypergraphs.



CK21: Dynamic Sub-Hypergraph Generator

- **G1.** Temporal order prediction model
- **G2.** Temporal reconstruction model



Recap: Four Empirical Observations

Intersection Size of Ego-networks

Temporally adjacent hyperedges in ego-networks are similar.

Spread of Alter-networks

Spread of alter-networks are temporally local.

Anthropic Principle of Ego-networks

The arrival of ego-nodes occurs after pre-dated hyperedges.

Novelty of Ego-networks

Novelty decreases in ego-networks.

C. Comrie, J. Kleinberg, “Hypergraph Ego-networks and Their Temporal Evolution”, **ICDM 2021**

Temporal Order Prediction

?

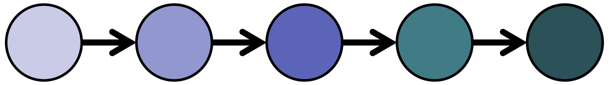
Question:

Has the given ego-network **evolved** reasonably?

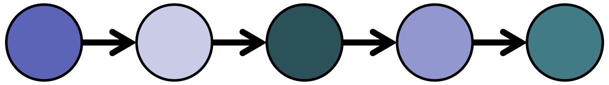
Answer:

- A supervised **binary classification** task is defined.
- Is the given ego-network **correctly** or **randomly** ordered?

!



Corrected ordered ego-network



Randomly ordered ego-network

Temporal Order Prediction (cont.)

- We train **a neural network classifier** using following features:
  - Length of the ego-network
  - Intersection density
  - Average alter-network spread
  - The number of future hyperedges the first hyperedge in the ego-network is a subset of
  - The number of prior hyperedges the last hyperedge is a superset of
  - Timestamp at which the ego-node entered the ego-network

For radial & contracted ego-networks

Temporal Order Prediction (cont.)

- Compared to **random guessing (baseline)**, the **trained classifier** significantly outperforms on all datasets and ego-network types.

	Star ego-network		Radial ego-network		Contracted ego-network	
	Random	**Proposed**	Random	**Proposed**	Random	**Proposed**
coauth-DBLP	0.50	**0.93 ± 0.01**	0.50	**0.91 ± 0.01**	0.50	**0.85 ± 0.01**
email-Avocado	0.50	**0.84 ± 0.09**	-	-	-	-
threads-ask-ubuntu	0.50	**0.72 ± 0.05**	-	-	-	-

Omitted due to their sizes

Classification accuracy

C. Comrie, J. Kleinberg, “Hypergraph Ego-networks and Their Temporal Evolution”, **ICDM 2021**

Temporal Order Prediction (cont.)

- Specifically, the **alter-network spread** is the most important feature.
  - **Proposed (Full):** Trained using all considered features
  - **Proposed (Single):** Trained using a single feature (i.e., alter-network spread)

	Star ego-network	Radial ego-network	Contracted ego-network
Random	0.50	0.50	0.50
Proposed (Full)	**0.93**	**0.91**	**0.85**
Proposed (Single)	0.89	0.84	0.75

Classification accuracy in coauth-DBLP

C. Comrie, J. Kleinberg, “Hypergraph Ego-networks and Their Temporal Evolution”, **ICDM 2021**

Temporal Reconstruction

?

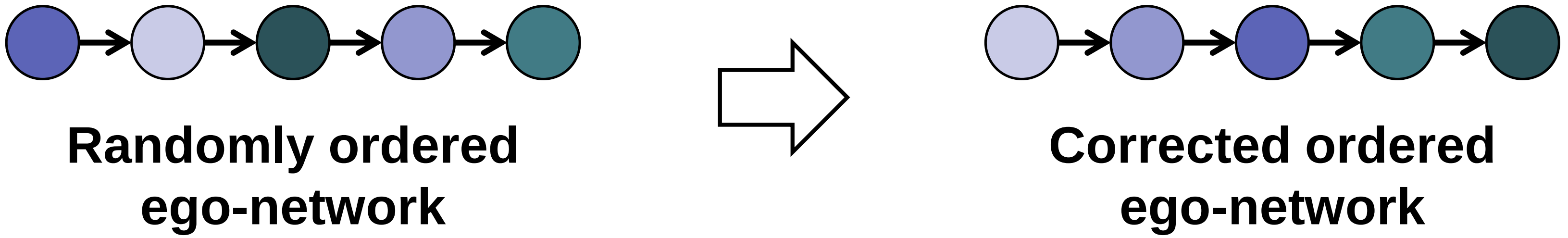
Question:

How can we properly **reconstruct** the temporal order of the given ego-network?

Answer:

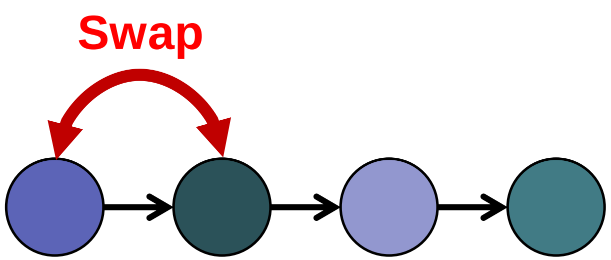
A local search algorithm is used to iteratively sort a randomly shuffled ego-network.

!

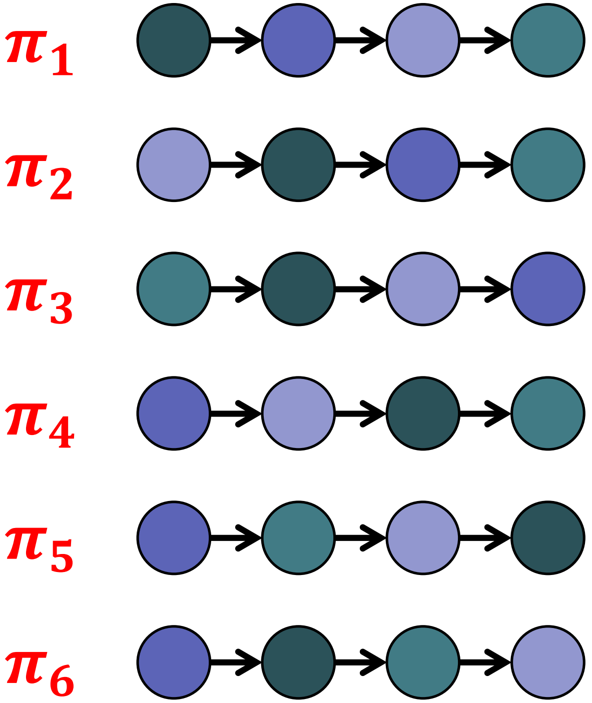


Temporal Reconstruction (cont.)

Step 1. Swap pairs



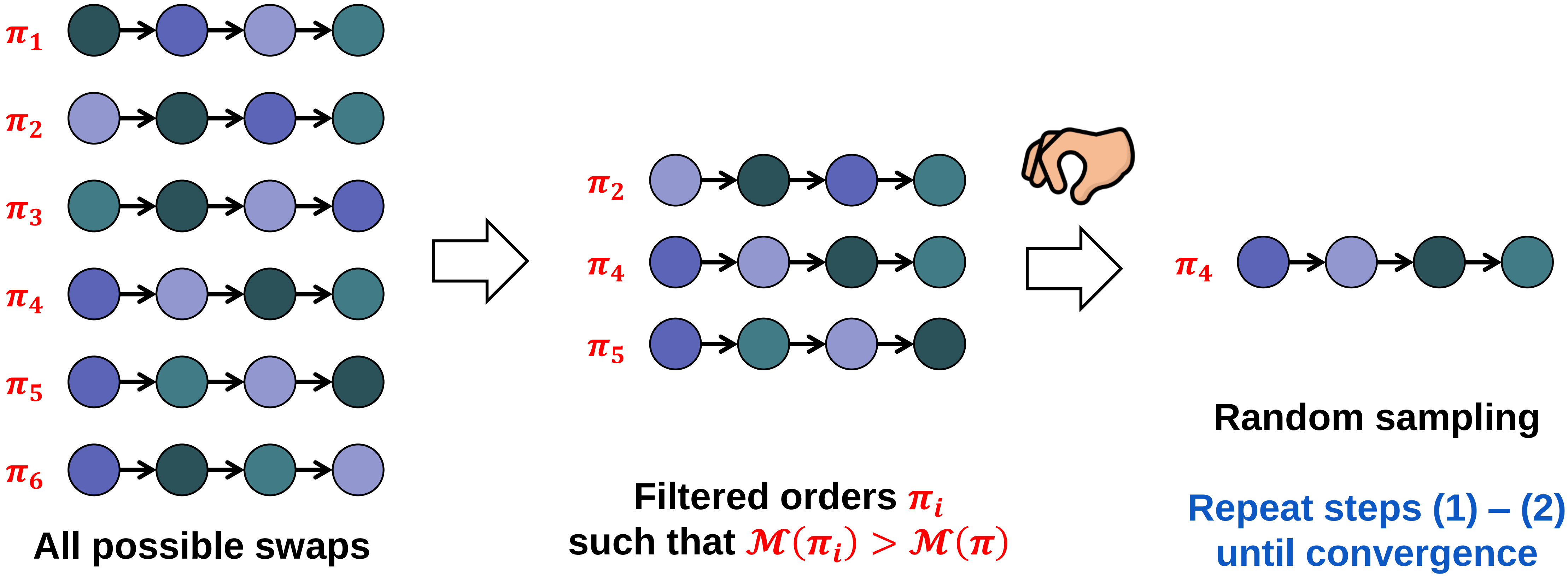
Randomly ordered
ego-network π



All possible swaps

Temporal Reconstruction (cont.)

Step 2. Predict the order

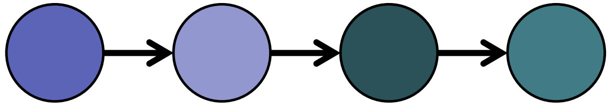


Repeat steps (1) – (2) until convergence

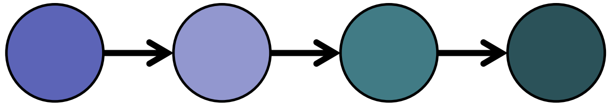
Temporal Reconstruction (cont.)

Step 3. Multiple trials

Trial 1: $\pi^{(1)}$

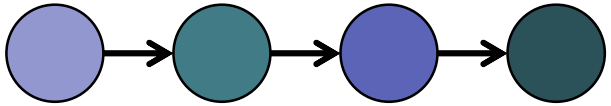


Trial 2: $\pi^{(2)}$



⋮

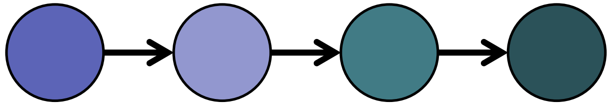
Trial T : $\pi^{(T)}$



Best orders from each trial

Select the best one:

$\max_{1\le t\le T}\mathcal{M}\left(\pi^{(t)}\right)$



Final output

Temporal Reconstruction (cont.)

- The proposed algorithm shows a **non-trivial improvement** over random guessing (baseline).

	Star ego-network		Radial ego-network		Contracted ego-network	
	Random	**Proposed**	Random	**Proposed**	Random	**Proposed**
coauth-DBLP	0.50	**0.65 $\pm$ 0.08**	0.50	**0.56 $\pm$ 0.05**	0.50	**0.65 $\pm$ 0.08**
email-Avocado	0.50	**0.63 $\pm$ 0.11**	-	-	-	-
threads-ask-ubuntu	0.50	**0.70 $\pm$ 0.07**	-	-	-	-

Omitted due to their sizes

Reconstruction accuracy, ***i.e., the ratio of corrected predicted pairs of hyperedges***

References

1. [BKT18] Benson, Austin R., Ravi Kumar, and Andrew Tomkins, “Sequences of Sets.” KDD 2018.
2. [CK21] Comrie, Cazamere, and Jon Kleinberg. “Hypergraph Ego-networks and Their Temporal Evolution.” ICDM 2021.
3. [CYLBKS22] Choe, Minyoung, et al. “MiDaS: Representative Sampling from Real-world Hypergraphs.” WWW 2022.
4. [DYHS20] Do, Manh Tuan, et al. “Structural Patterns and Generative Models of Real-world Hypergraphs.” KDD 2020.
5. [KKS20] Kook, Yunbum, Jihoon Ko, and Kijung Shin. “Evolution of Real-world Hypergraphs: Patterns and Models without Oracles.” ICDM 2020.
6. [LCS21] Lee, Geon, Minyoung Choe, and Kijung Shin. “How Do Hyperedges Overlap in Real-world Hypergraphs? – Patterns, Measures, and Generators." WWW 2021.