Seeded Graph Matching: The Power of Multi-hops

Jiaming Xu

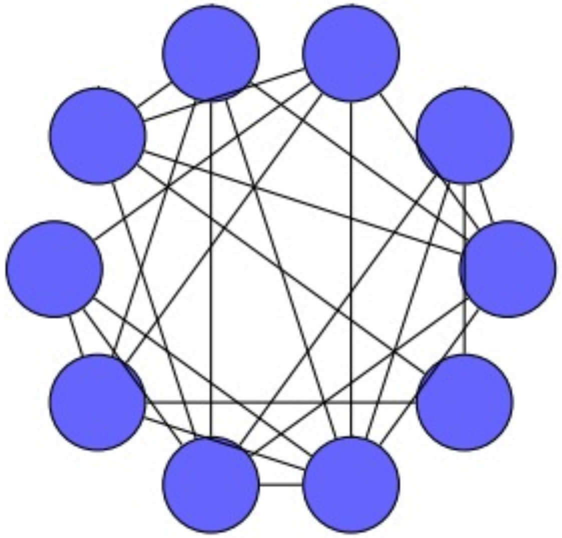
The Fuqua School of Business

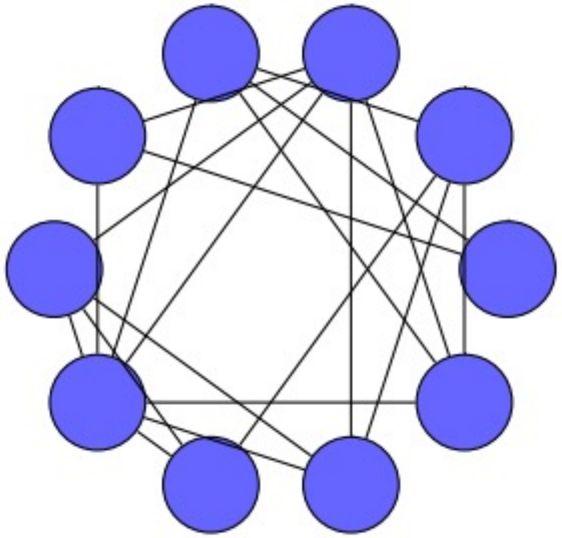
Duke University

Joint work with Elchanan Mossel (MIT)

Xiaojun Lin and Liren Yu (Purdue)

Graph matching (network alignment)

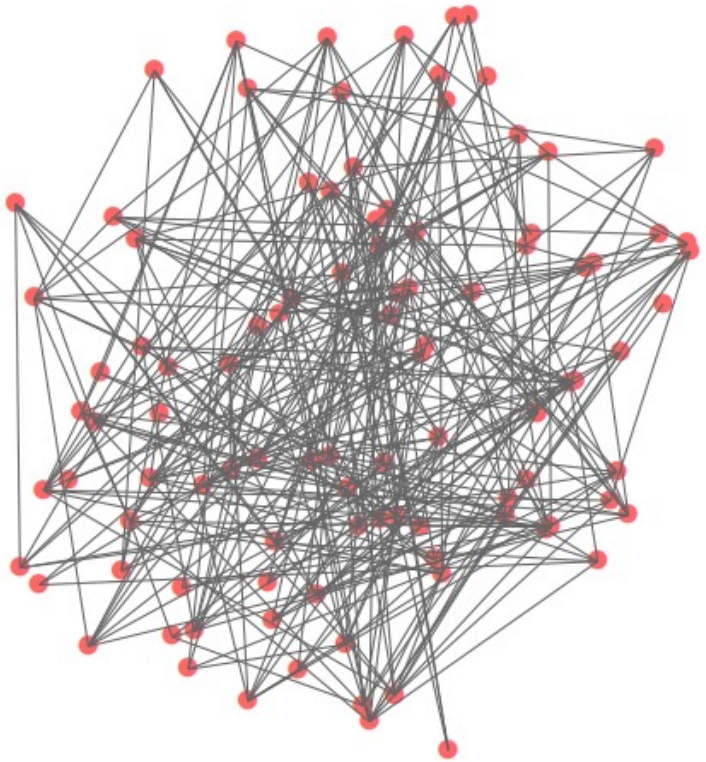


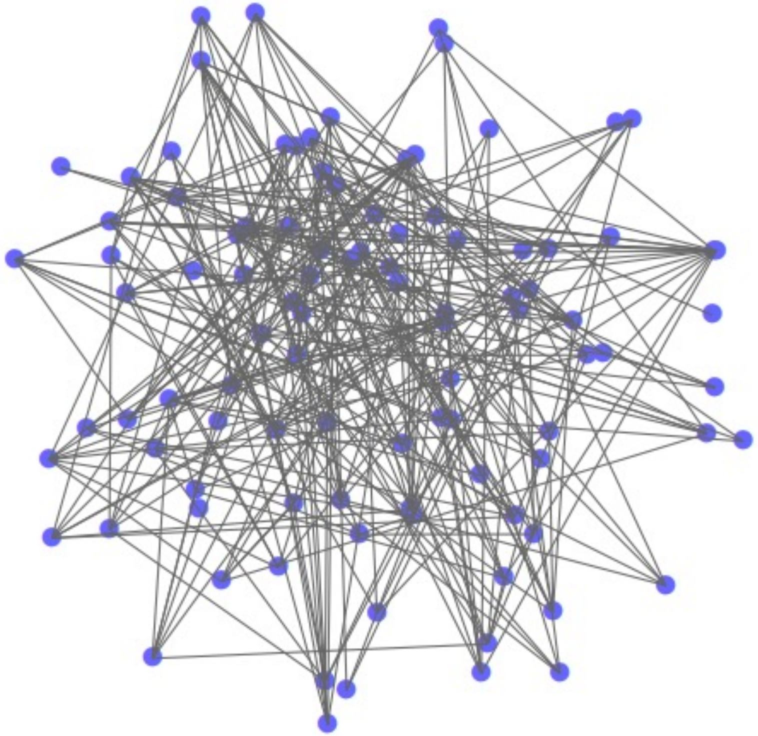


- Goal: find the node correspondence between two graphs that minimizes # of adjacency disagreements
- Noiseless case: reduce to graph isomorphism

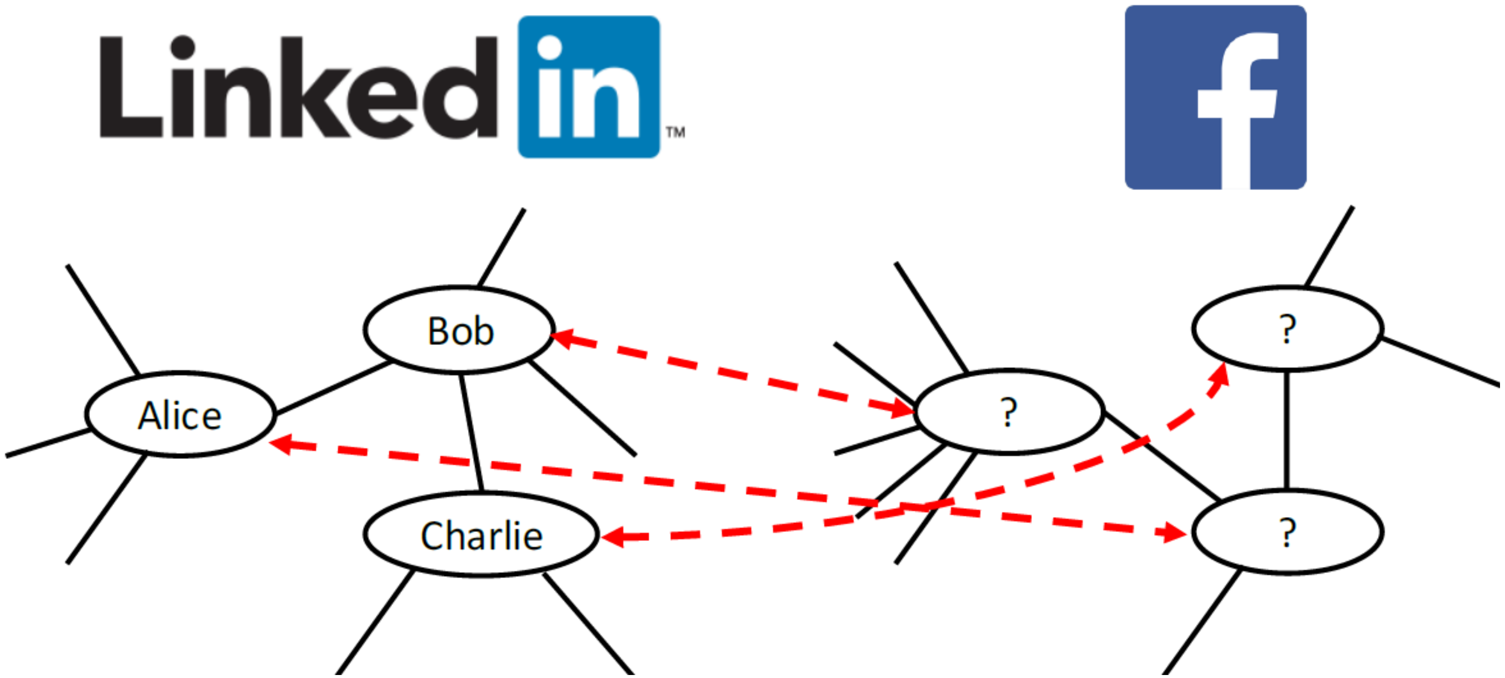
Two key challenges

- Statistical: two graphs are not exactly isomorphic
- Computational: # of possible node mapping is $n!$



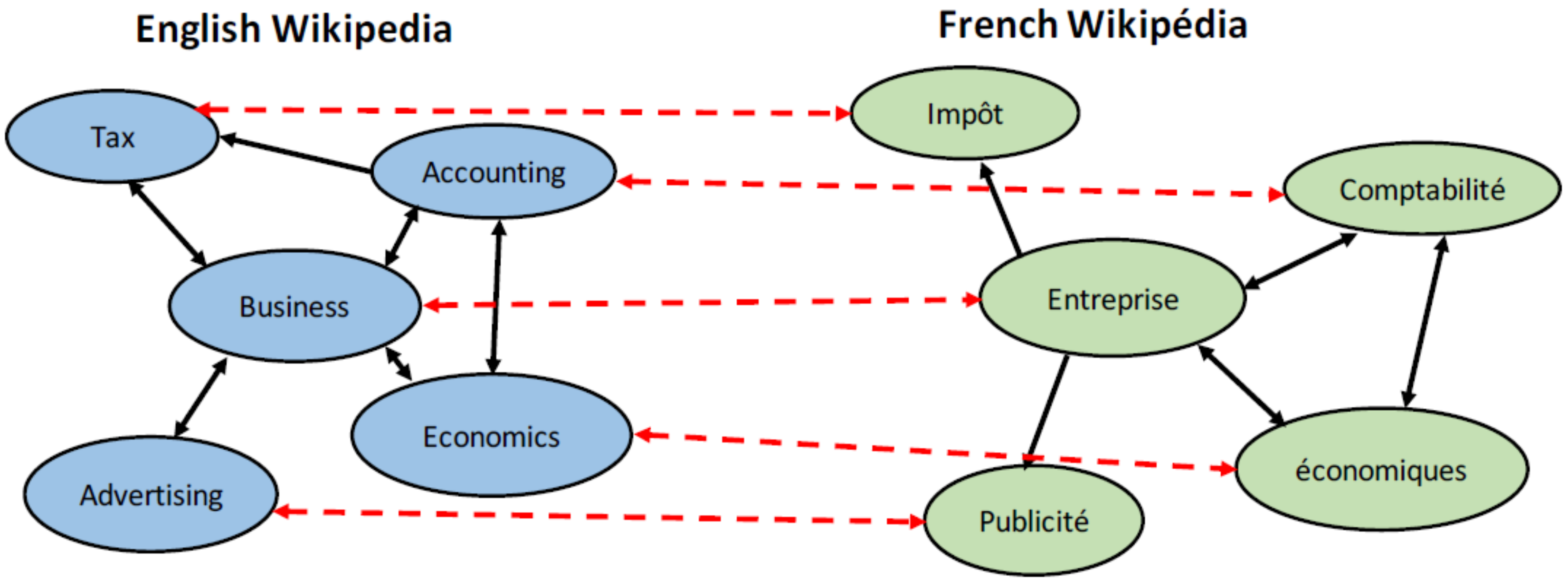


Application 1: Network de-anonymization



- Successfully de-anonymize Netflix by matching it to IMDB [Narayanan-Shmatikov '08]
- Correctly identified 30.8% of node mappings between Twitter and Flickr [Narayanan-Shmatikov '09]

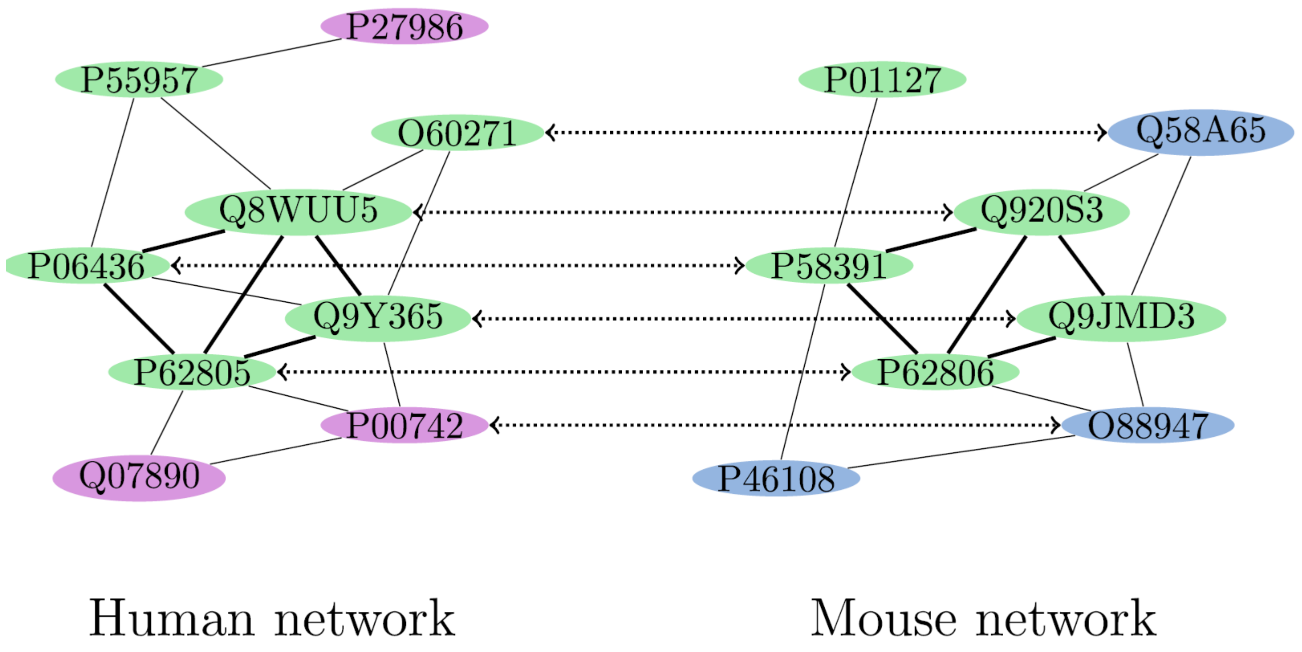
Application 2: Machine translation



Automatically find/correct corresp. wiki articles in different languages

[Fishkind-Adali-Patsolic-Meng-Lyzinski-Priebe ’12]

Application 3: Protein interaction network

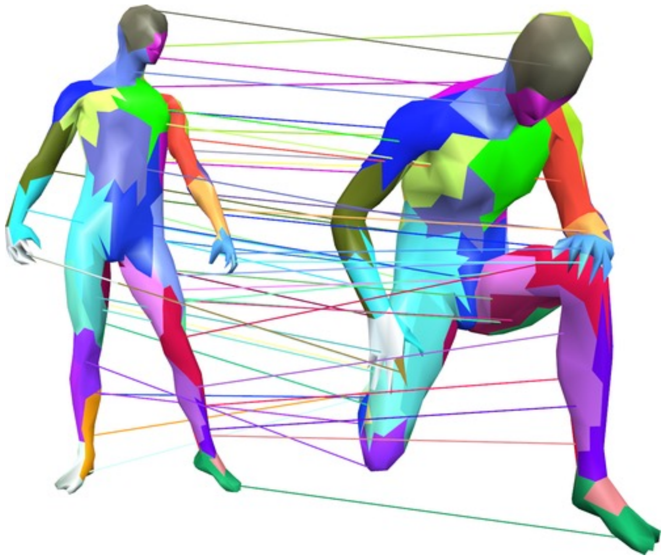


[Kazemi-Hassani-Grossglauser-Modarres `16]

Aligning PPI networks between different species, to identify conserved components and genes with common function [Singh-Xu-Berger' 08]

Application 4: Computer vision

A fundamental problem in computer vision: Detect and match similar objects that undergo different deformations



Shape Retrieval Contest (SHREC) dataset [Lahner et al '16]

3-D shapes -> geometric graphs (features -> nodes, distance -> edges)

Beyond worst-case intractability

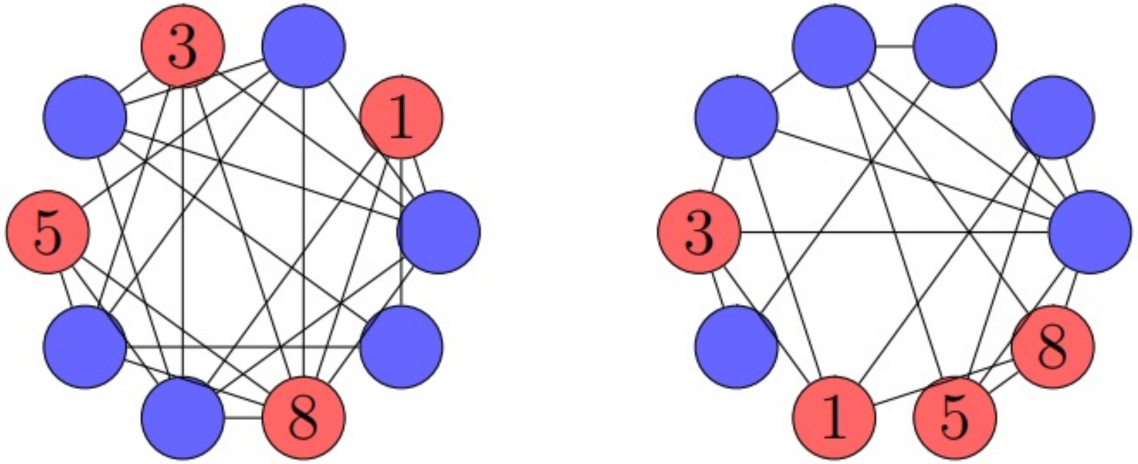
- Cast as quadratic assignment problem (QAP)

$$\min ||A_1 - \Pi A_2 \Pi^\top||_F$$

- NP-hard to solve or approximate in the worst case
- However, real networks are not designed by adversary!
- Recent surge of interest on average-case analysis of matching correlated random graphs [Cullina-Kiyavash '16, 17, Ding-Ma-Wu-X. '18, Barak-Chou-Lei-Schramm-Sheng '19, Fan-Mao-Wu-X. '19a, 19b, Ganassali-Massoulie '20, Mao-Rudelson-Tikhomirov '21,...]

Focus of this talk: Seeded graph matching

- An initial seed set of true pairs is revealed



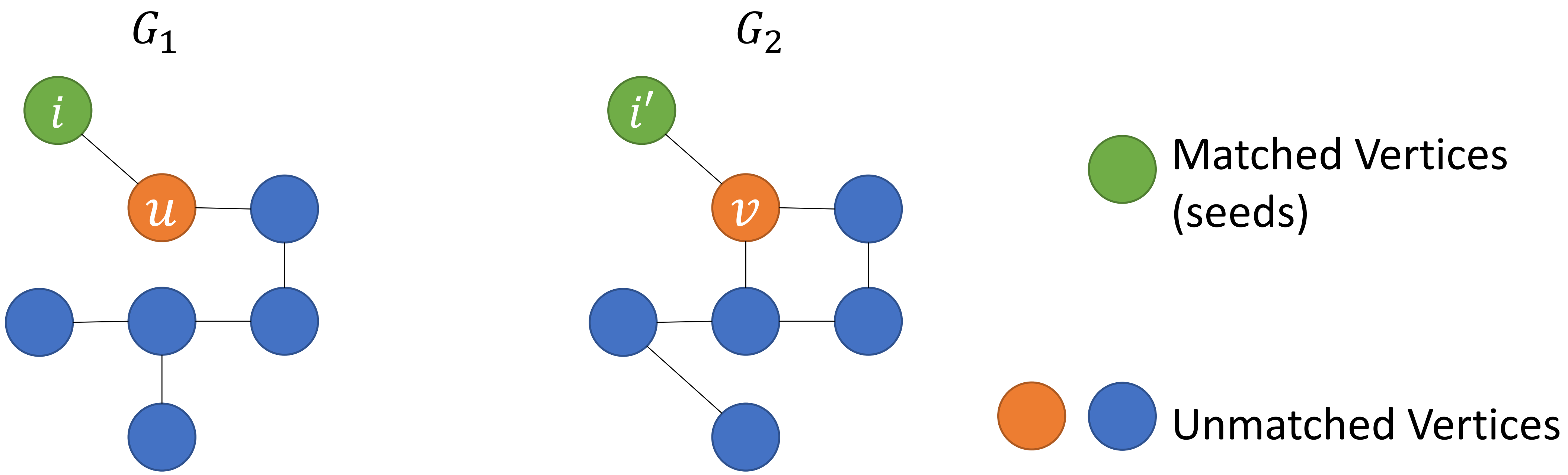
- Goal: Match the remaining vertices based on seeds and graph structures

Seeded graph matching

- The seeds can be obtained by prior knowledge or manual labeling
  - Example: Some users provide identifiable information across different social media [Narayanan-Shmatikov `08]

- However, we often only have very few seeds

Previous ideas: 1-hop witnesses



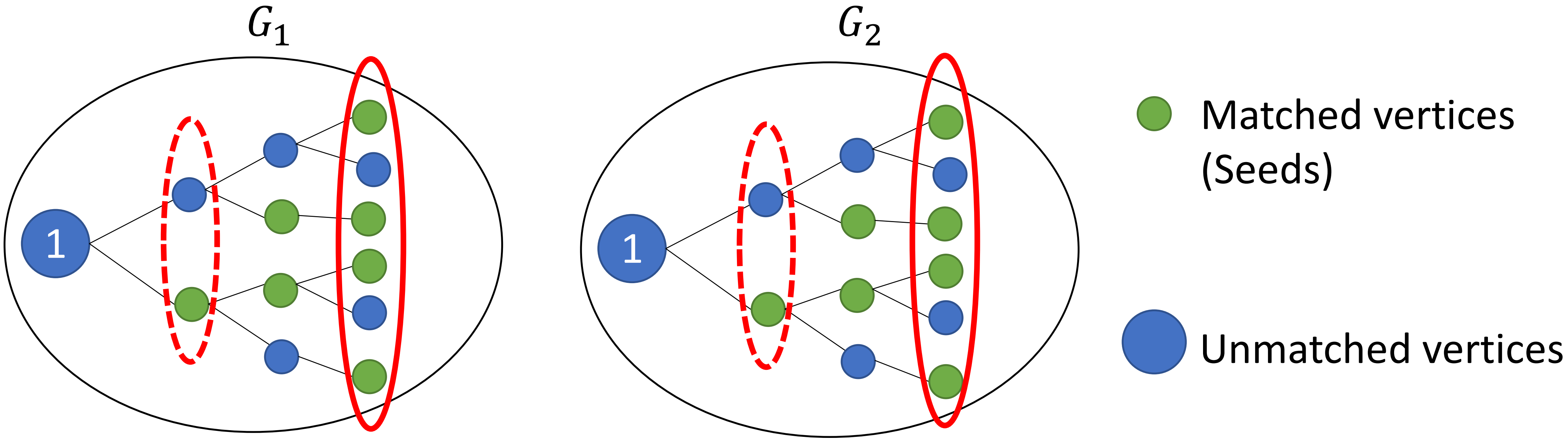
The seed (i, i') is a **1-hop witness** for (u, v)

- # of 1-hop witnesses => similarity measure
- Most existing seeded matching algorithms use only 1-hop witnesses [Yartseva-Grossglauser '13; Korula-Lattanzi '13; Kazemi-Hassani-Grossglauser '15].

Our ideas: multi-hop witnesses

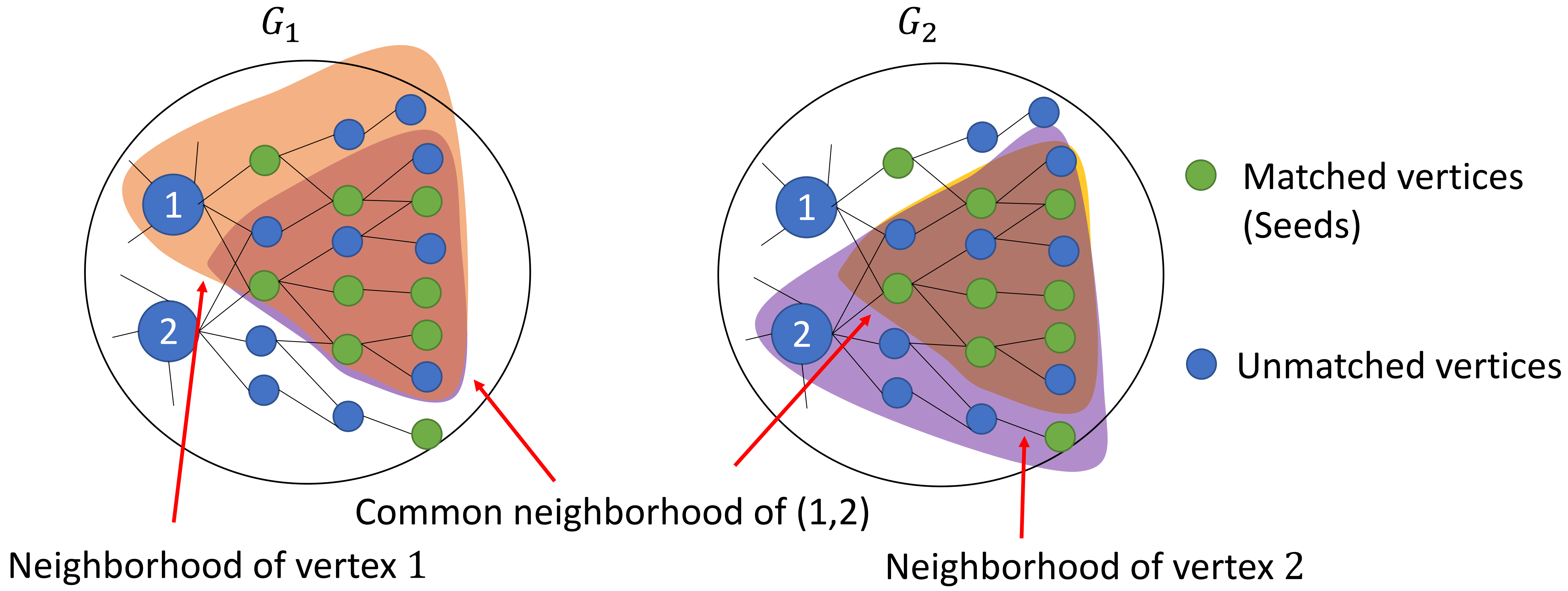
- Using 1-hop witness is insufficient
  - The size of the 1-hop neighborhood can be too small => too few witnesses even for true pairs

- Explore much larger neighborhoods => more multi-hop witnesses



A central challenge in using multi-hop witnesses

Fake pairs may have too many multi-hop witnesses

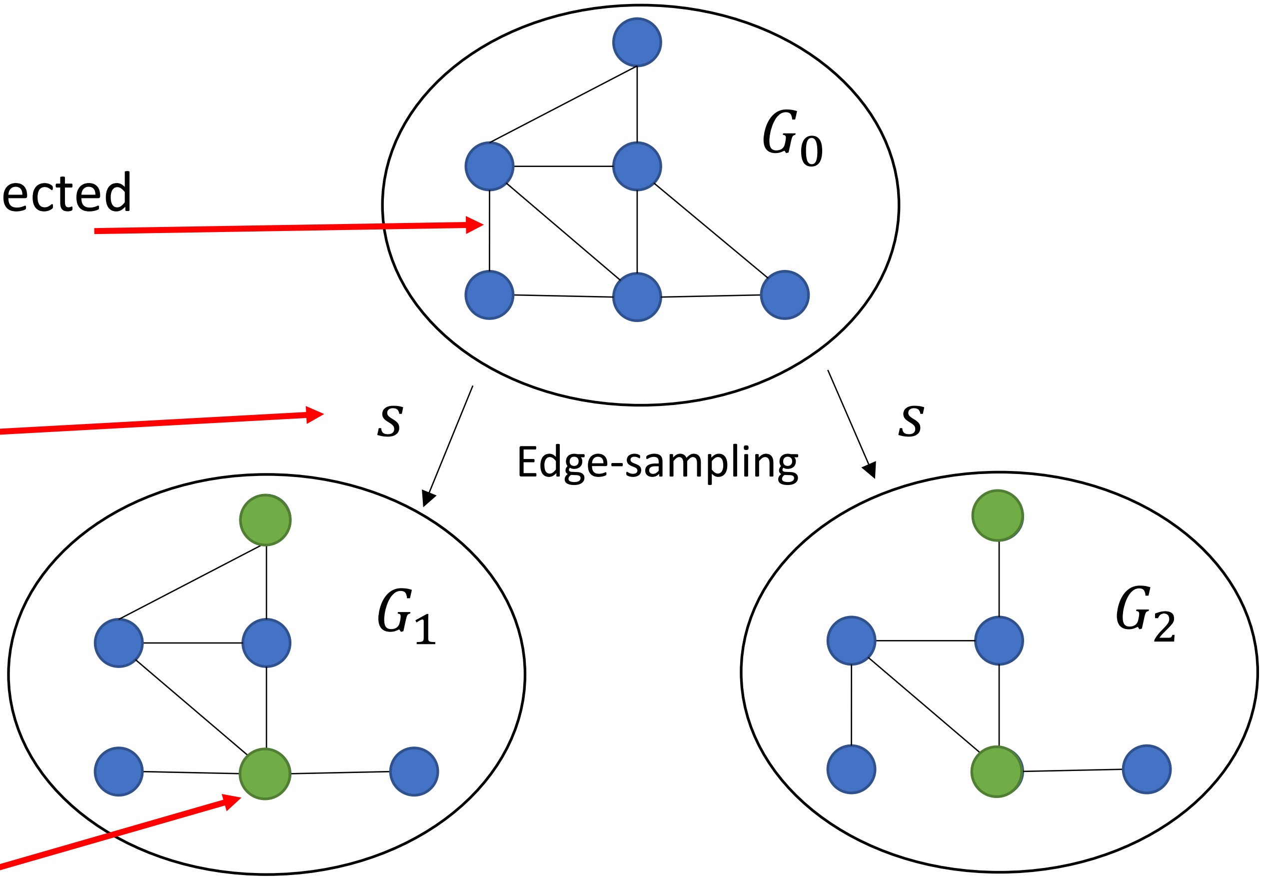


Outline of the remainder

1. Matching correlated ER random graphs
2. Matching power-law graphs
3. Seeded graph neural network
4. Conclusion

Correlated Erdős-Rényi Random Graph Model

- The parent graph G_0 has n vertices
- For any two vertices i and j, they are connected independently with probability p
- Sample edges in G_0 to construct G_1 and G_2 with probability s
- A fraction α of true pairs are chosen as seeds



Relabel nodes according to random permutation π^*

Seeds

Performance guarantee

Theorem [Mossel- X. '20]

Suppose $s = \Theta(1)$. All vertices can be correctly matched in polynomial-time with high probability, if

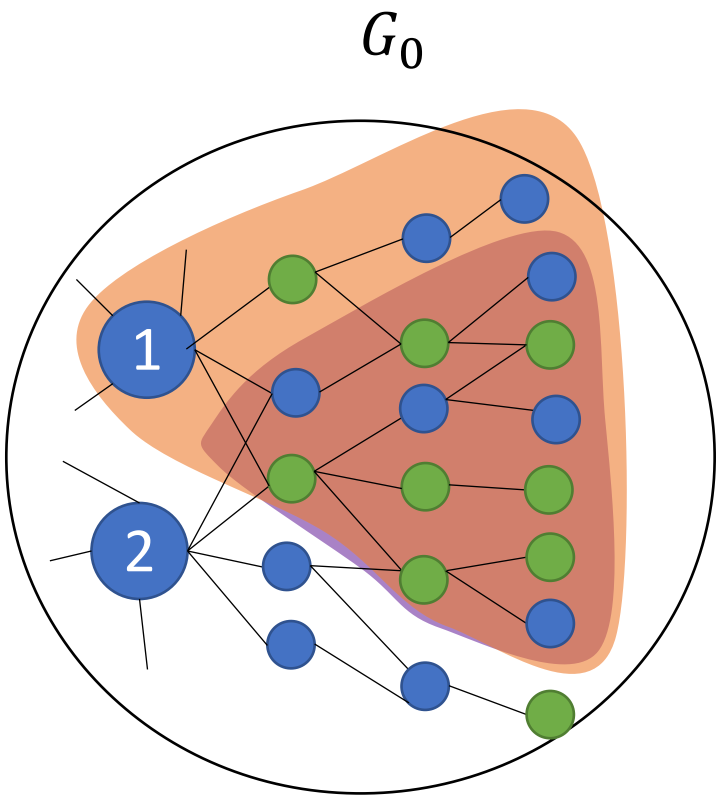
$$\alpha n \geq \begin{cases} n^{\varepsilon}, & \log n \ll np \leq n^{\varepsilon} \\ \Omega(\log n), & np = \Theta(n^{1/k}) \end{cases},$$

(Sparse regime)

(Dense regime)

- Previous work on 1-hop witnesses need $\alpha n \gtrsim \frac{1}{p}$ [Korula-Lattanzi '14]
- Our results can achieve exponential reduction in seed size requirement

Intuition behind



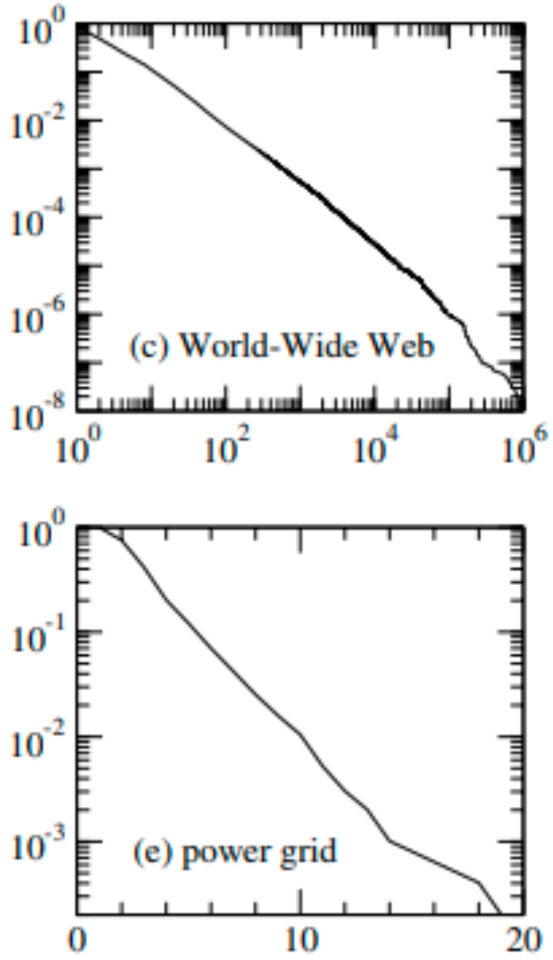
- The size of D-hop neighborhood $\approx (np)^D$
- The size of intersection of two D-hop neighborhoods $\approx (np)^D \frac{(np)^D}{n}$
- So we need

$$\underbrace{(np)^D \lesssim n}_{\text{Fewer witnesses for fake pairs}} \qquad \underbrace{\alpha(nps^2)^D \gtrsim \log n}_{\text{Sufficient witnesses for true pairs}}$$

Outline of the remainder

1. Matching correlated ER random graphs
2. Matching power-law graphs
3. Seeded graph neural network
4. Conclusion

Real-world networks have power-law degree distribution



Fraction of vertices with degree $\geq$ k versus the threshold k

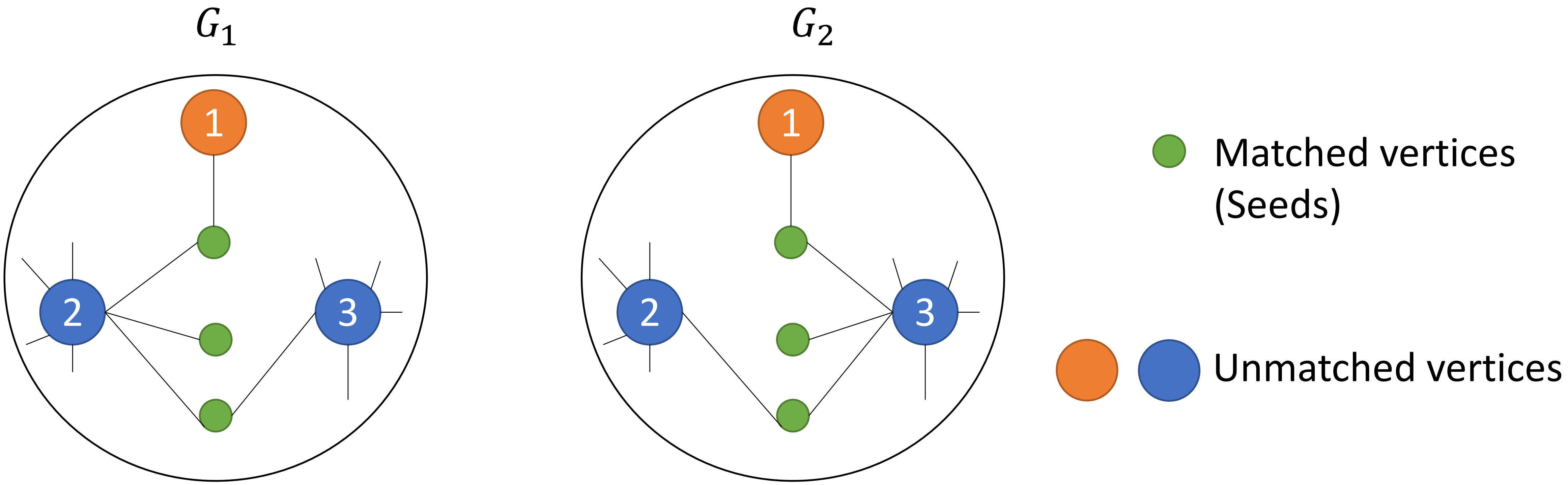
Many real-world networks have power-law degree distribution:

$$P(\text{degree} \geq k) \sim k^{1-\eta}$$

ER graphs do not match this property

Difficulty in matching power-law graphs

Due to the degree fluctuations, a fake pair with high degrees may have many more witnesses than a true pair with low degrees.



The true pair $(1,1)$ only has 1 witness, but the fake pair $(2,3)$ has 3 witnesses.

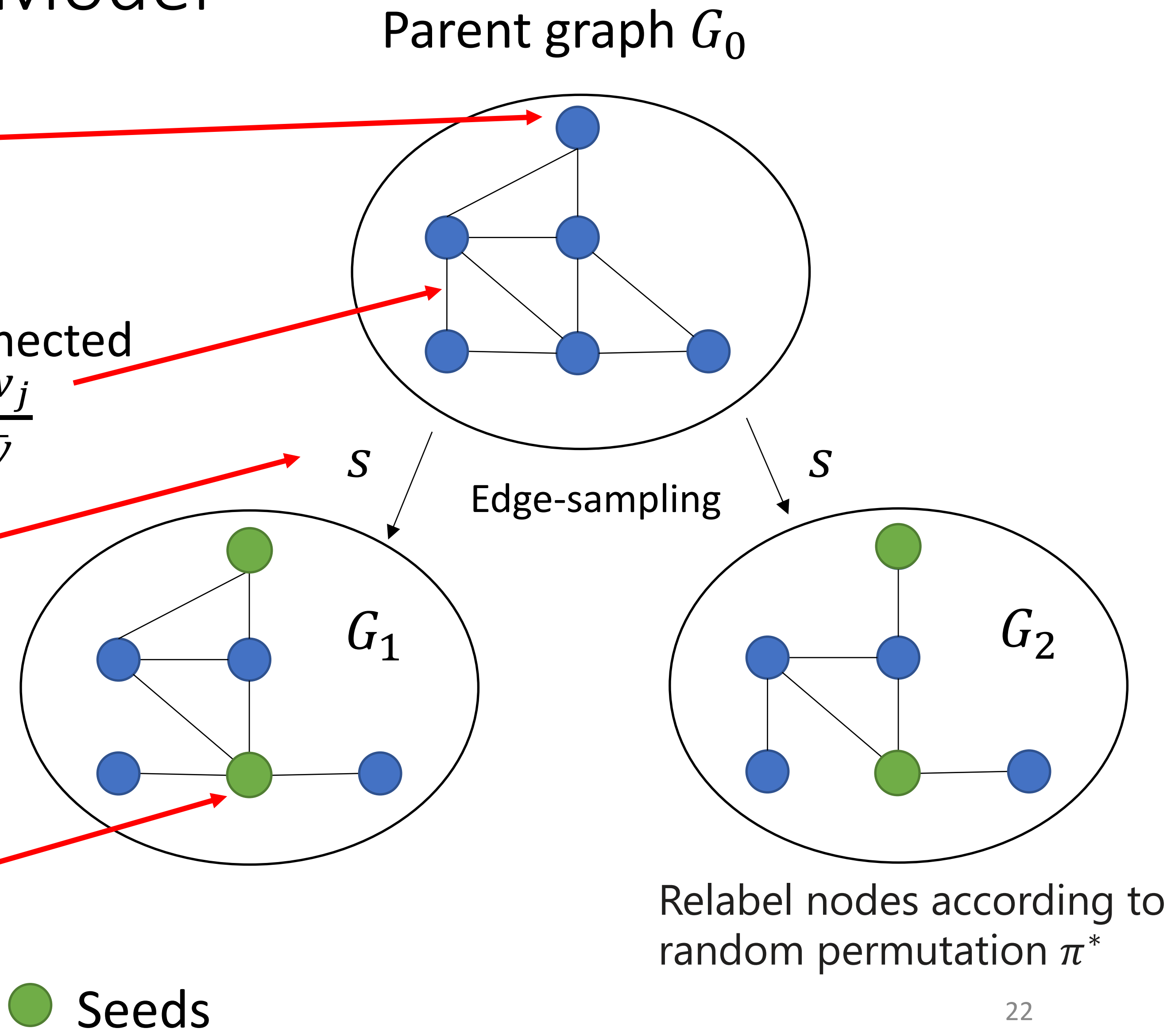
Difficulty in matching power-law graphs

	User Matching [Korula-Lattanzi '14]	**DDM** [Chiasserini-Garetto-Leonardi '16]	**Y-test** [Bringmann-Friedrich-Krohmer '14]	**Power-law D-hop (PLD)** (ours)
Number of seeds required to match a constant fraction of n vertices	$\Omega(n/\log(n))$	$\Omega(n^{1/2+\epsilon})$	$\Omega(n^{1/2+\epsilon})$	$\Omega((\log n)^{4-\eta})$ η: the constant exponent of power-law degree distribution

Our Contribution: PLD only needs $\Omega(\text{polylog}\, n)$ seeds!

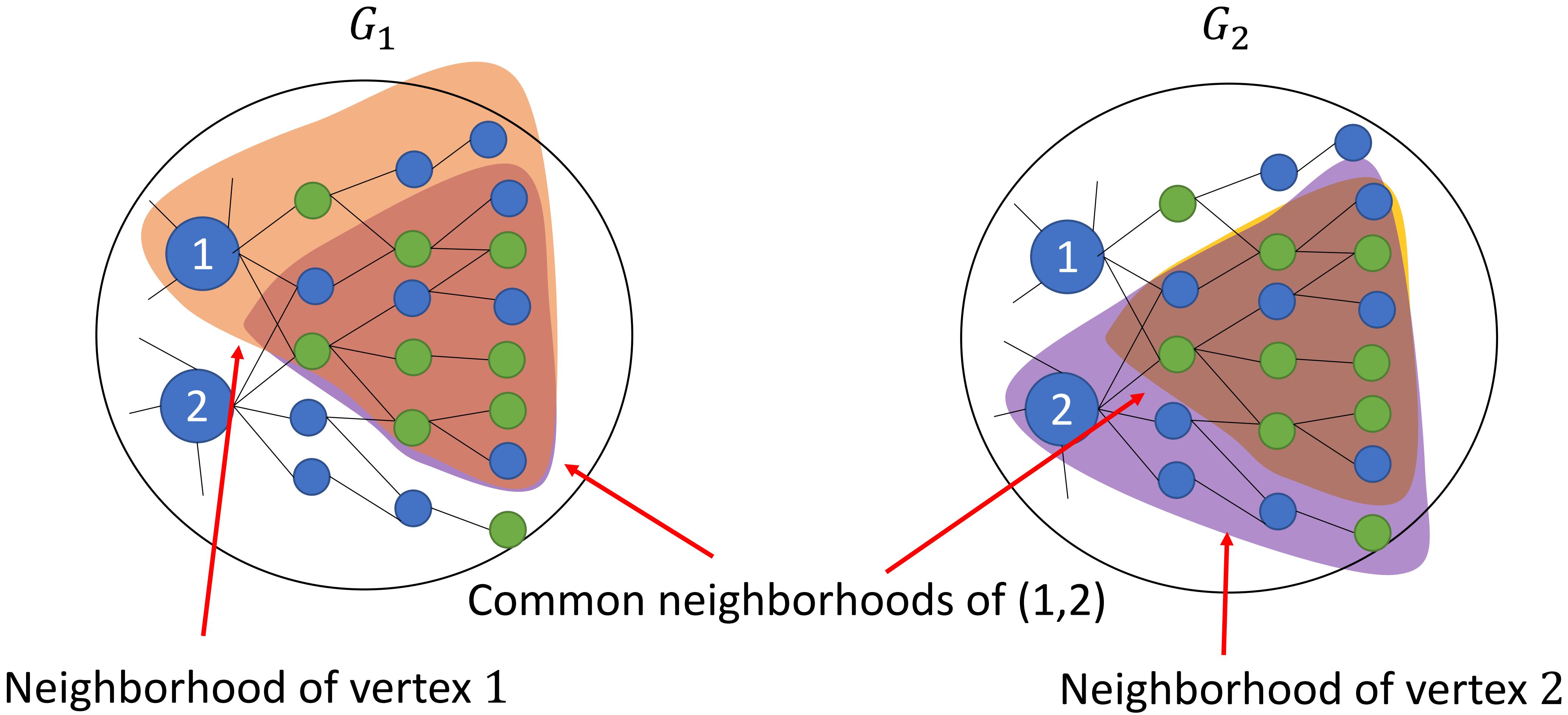
Chung-Lu Random Graph Model

- Weight of vertex i: $w_i = c(n/i)^{\frac{1}{\eta-1}}$
  - $P(\text{weight} \geq k) \approx k^{1-\eta}$
  - $2 < \eta < 3$
- For any two vertices i and j, they are connected independently with probability $p_{ij} = \frac{w_i w_j}{n\bar{w}}$
  - $\bar{w} = \frac{1}{n}\sum_i w_i$
- Sample edges in G_0 to construct G_1 and G_2 with probability s
- Each true pair is added into the seed set with probability α independently.

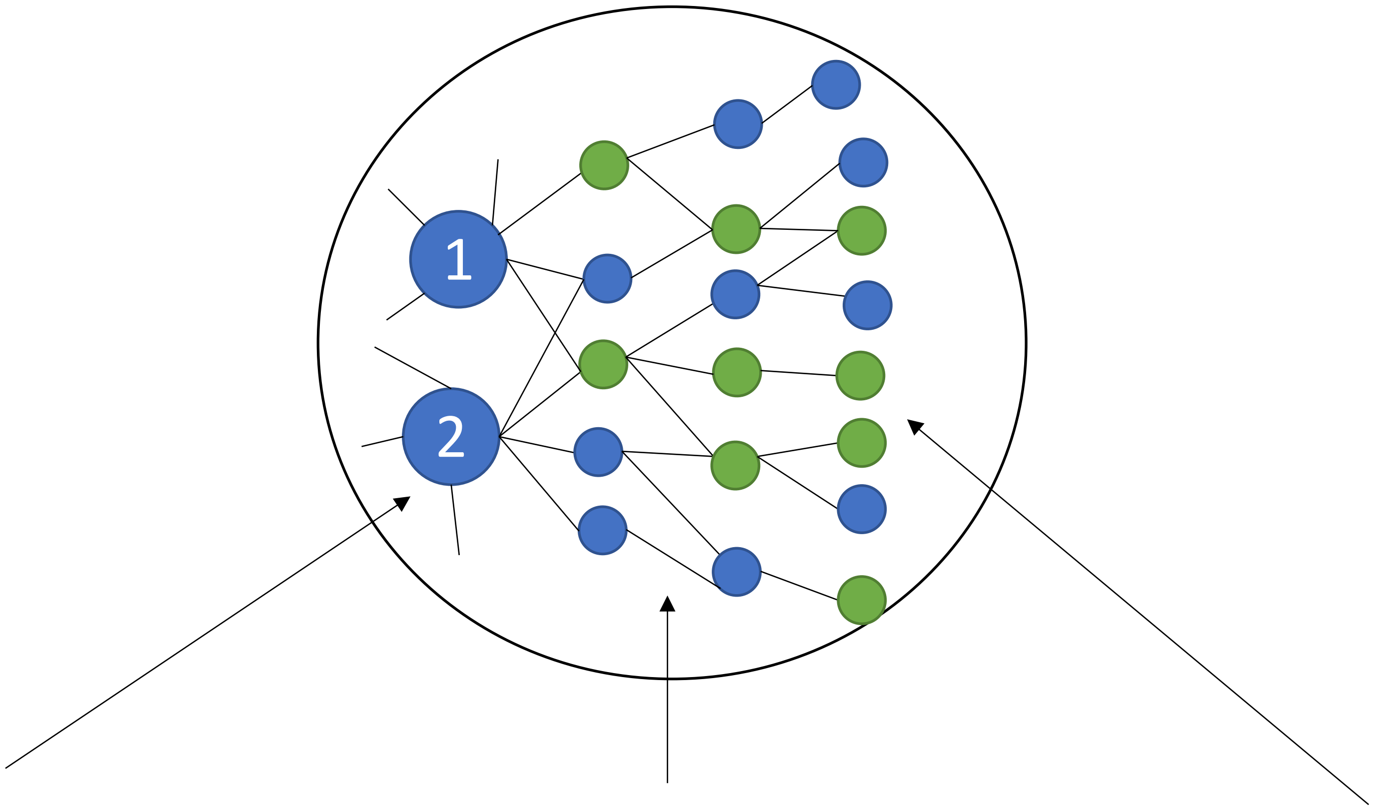


Key challenge: how to apply D-hop witnesses

Fake pairs with high weight may have too many D-hop witnesses



Key new idea: control the D-hop neighborhood sizes



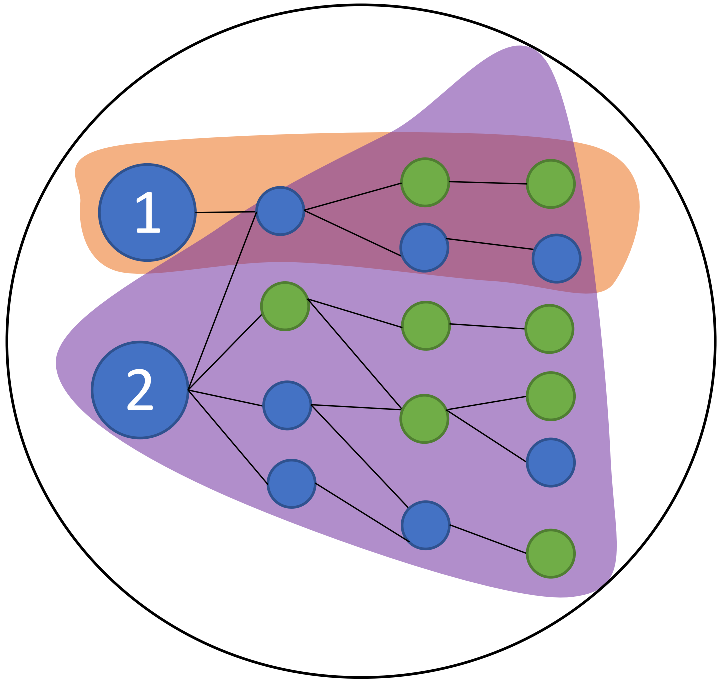
candidate vertex-pairs intermediate vertices seeds

Key choice #1: the candidate vertex-pairs

- Carefully choose the candidate vertex-pairs to be matched using the D-hop witnesses

  - Weight is too small
    $\Longrightarrow$ True pairs have too few D-hop witnesses

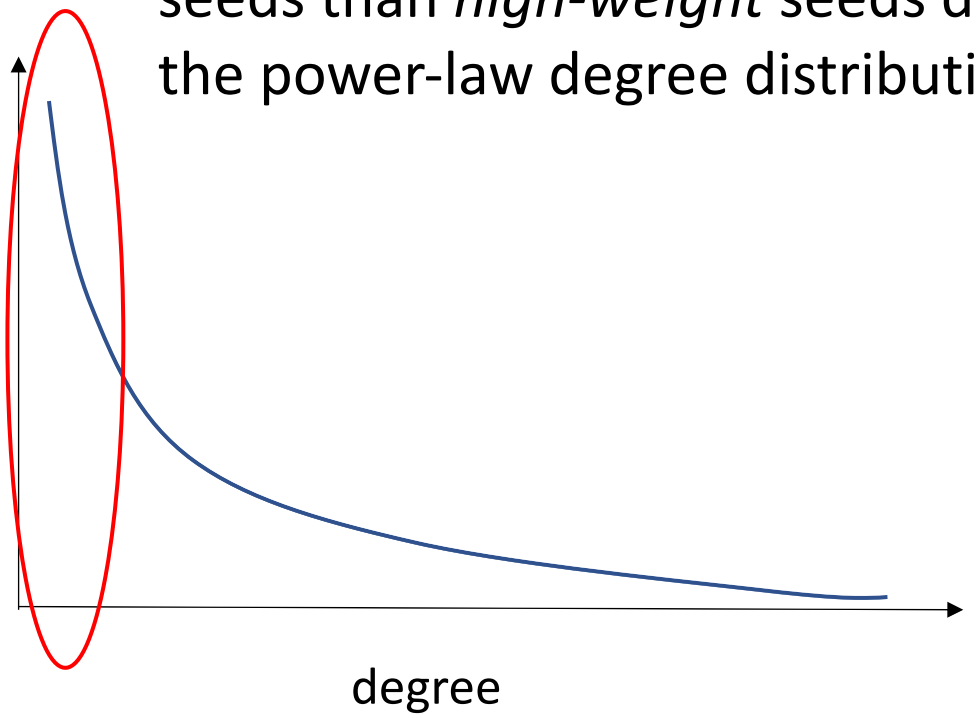
  - Weight is too large
    $\Longrightarrow$ Fake pairs have too many D-hop witnesses

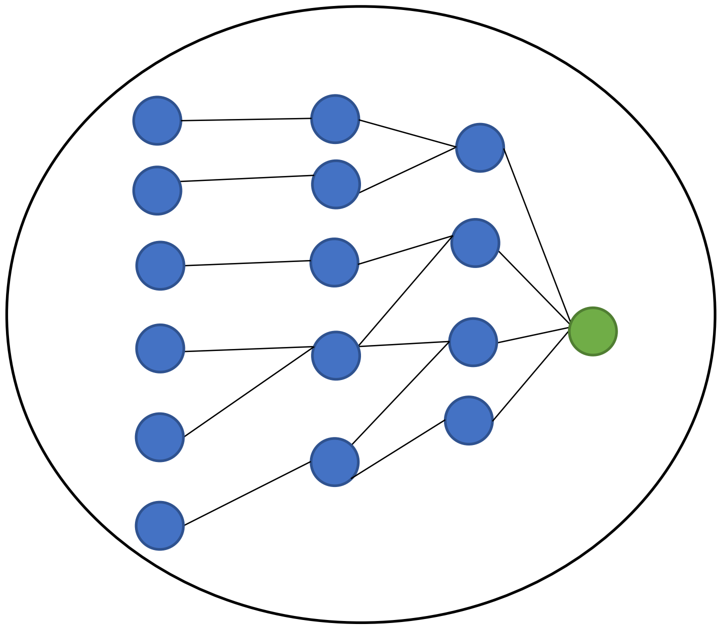


  - Our choice: weight of candidate vertex-pairs $\approx n^{\gamma}$

Key choice #2: the seeds

- Utilize low-weight seeds while avoiding high-weight seeds.
  - There are many more *low-weight* seeds than *high-weight* seeds due to the power-law degree distribution
  - Too many vertex-pairs will have *high-weight* seeds as witnesses.

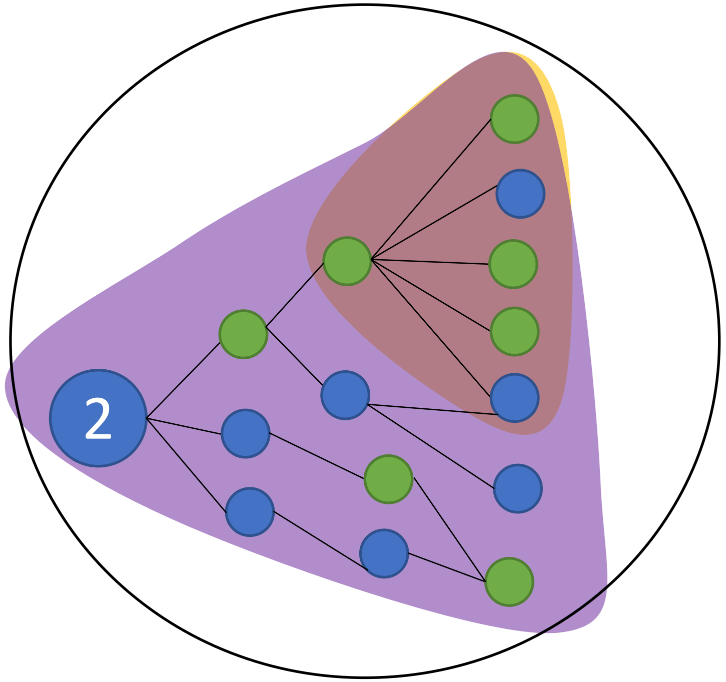




- Our choice: weight of seeds $= \Theta(1)$

Key choice #3: the intermediate vertices

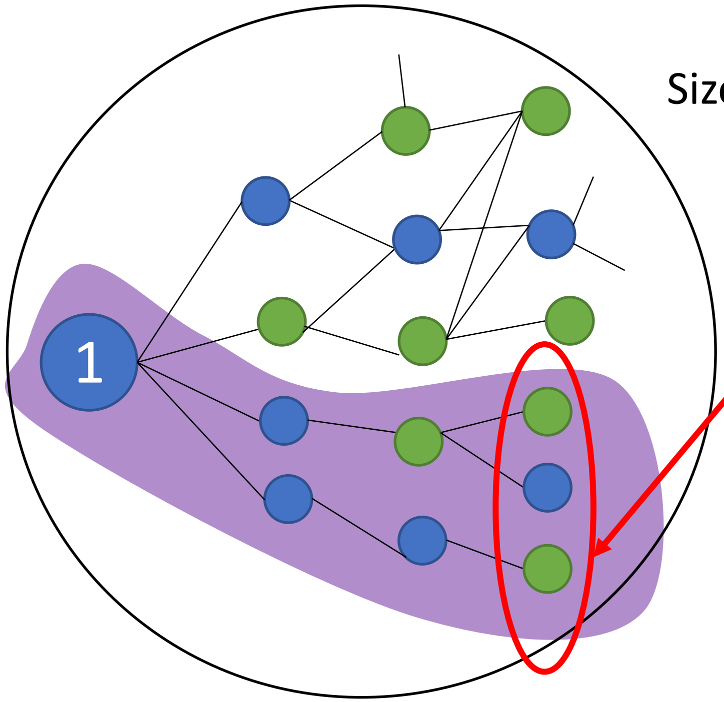
- The high-weight vertices are not suitable to be the intermediate vertices when constructing the D-hop neighborhoods
  - When its weight is too large, an intermediate vertex leads to a very large neighborhood



  - Our choice: weight of intermediate vertices $\leq n^{\gamma}$

Estimating the size of the controlled D-hop neighborhoods

- The weight of candidate vertex-pairs $\approx n^{\gamma}$
- The weight of intermediate vertices $\leq n^{\gamma}$
- The weight of seeds ≈ 1



Size of the D-hop neighborhood

$N_D = ?$

Recursion:

$N_1 \approx n^{\gamma}$

$N_2 \approx N_1 \cdot n^{\gamma(3-\eta)}$

$N_3 \approx N_2 \cdot n^{\gamma(3-\eta)}$

$\vdots$

$N_D \approx N_{D-1} \cdot n^{\gamma(3-\eta)} \approx n^{\gamma} \cdot n^{(D-1)\gamma(3-\eta)}$

Choice of γ

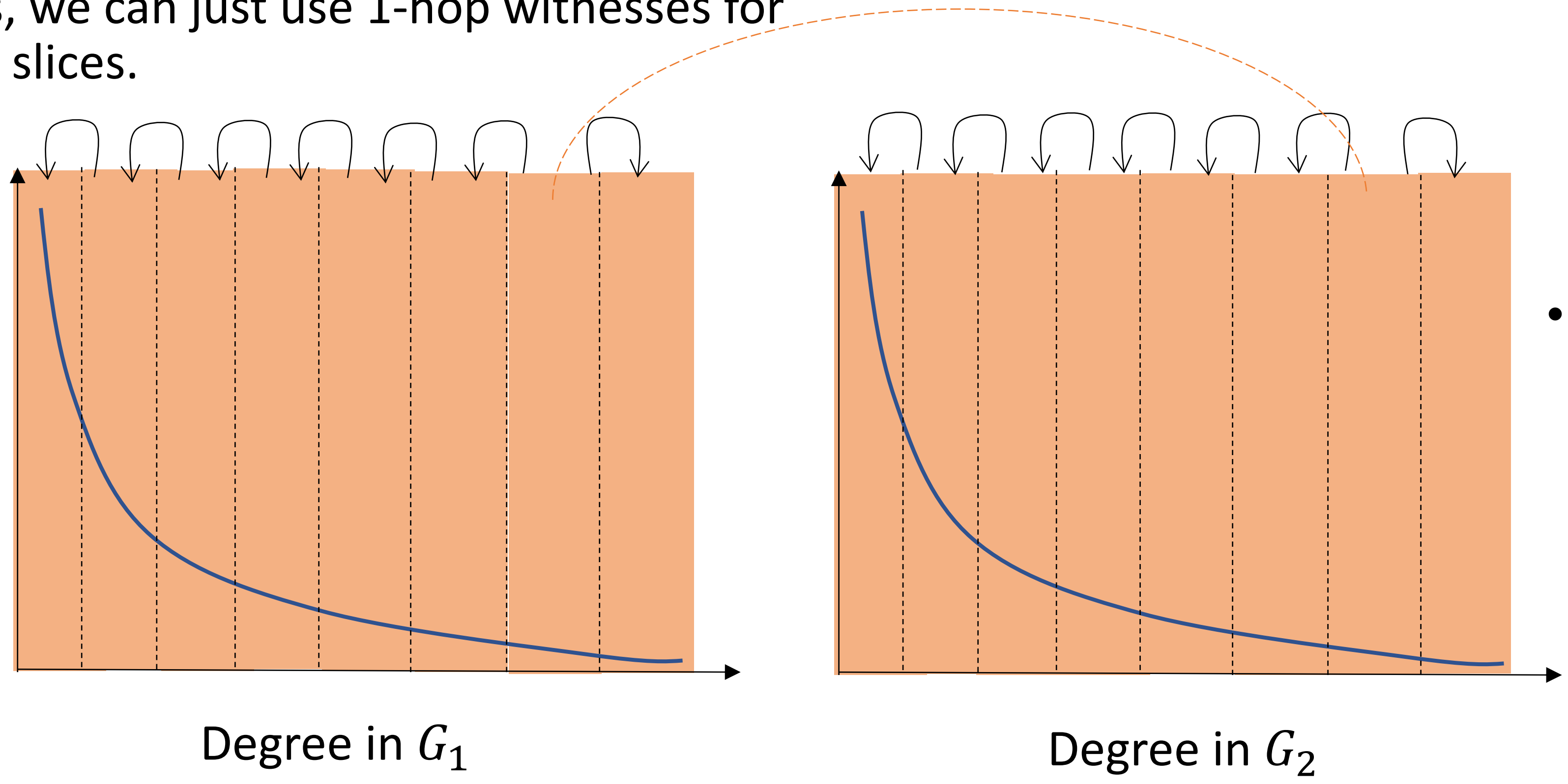
- The size of the controlled D-hop neighborhood $N_D \approx n^{\gamma((3-\eta)(D-1)+1)}$
- So we need

$$\underbrace{N_D \lesssim \frac{n}{(\log n)^{3-\eta}}}_{\text{Fewer witnesses for fake pairs}} \qquad \underbrace{\alpha N_D \gtrsim \log n}_{\text{Sufficient witnesses for true pairs}}$$

- Together, the seed requirement can be dramatically reduced to $\Omega(\,(\log n)^{4-\eta}\,)$

Sketch of the whole PLD algorithm

- Using matched pairs as new seeds to trigger a cascading process. Due to sufficient new seeds, we can just use 1-hop witnesses for other slices.
- Match the first slice (degree $\approx n^{\gamma}$) with D-hop witnesses



- Partition two graphs into slices based on the vertex degree

Further complication: 1. A true pair may have different degrees. We instead partition graphs by overlapped “imperfect slices”.
2. For low-degree vertices with insufficient 1-hop witnesses, we apply the PGM algorithm in [Yartseva-Grossglauser ‘13] to match them.

Theoretical performance guarantee

Theorem [Yu-X.-Lin `21]

Suppose $D > \frac{4-\eta}{3-\eta}$*. Choose*

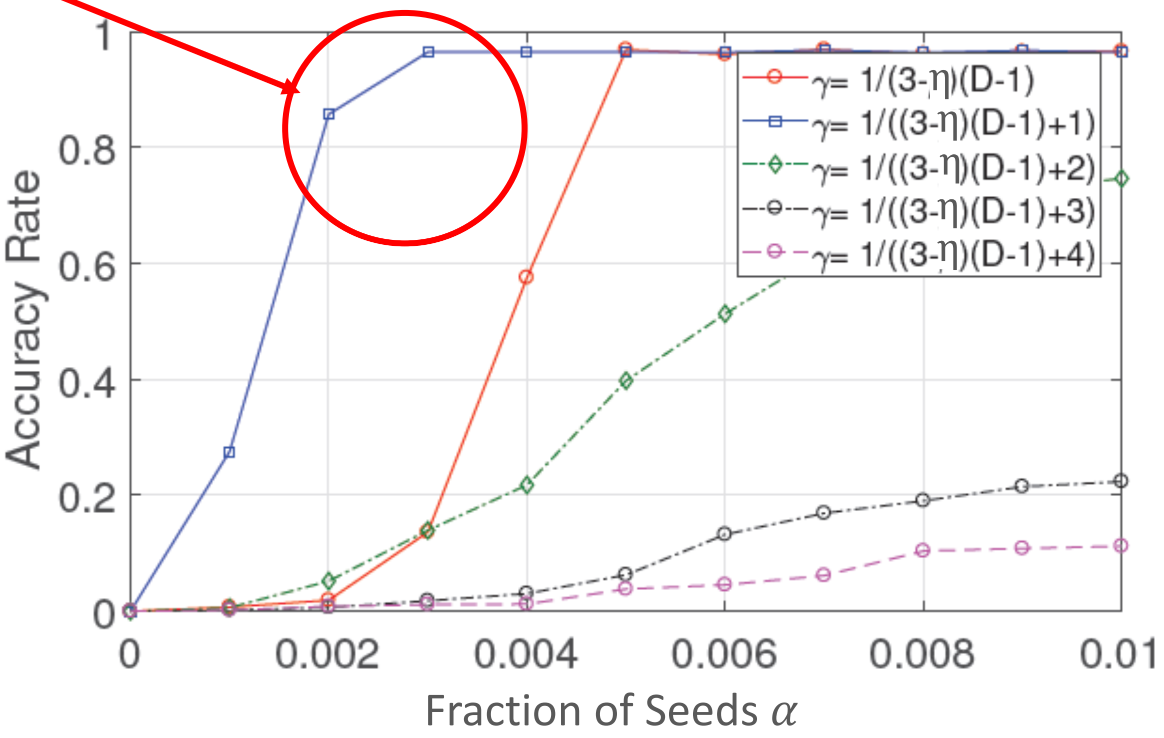
$$n^{\gamma((3-\eta)(D-1)+1)} = \frac{cn}{(\log n)^{3-\eta}},$$

for a sufficiently small constant c*. If there are* $\Omega(\,(\log n)^{4-\eta}\,)$ *initial seeds chosen independently at random, with high probability our Power-Law D-hop (PLD) algorithm correctly matches* $\Omega(n)$ *vertex-pairs without any error.*

- Time Complexity: $O\left(n^{3-2\gamma(\eta-1)}\right)$

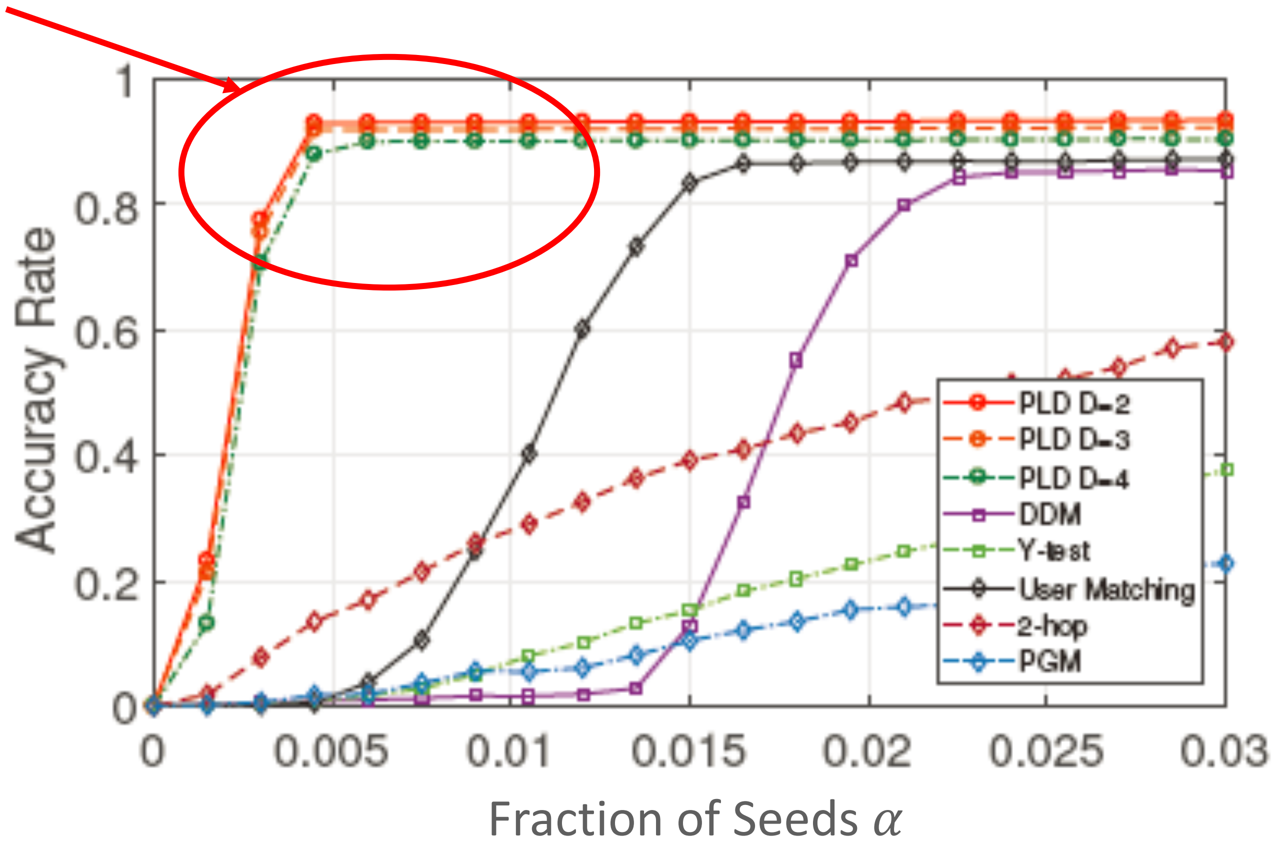
Experimental results: choice of γ

- Chung-Lu model with 10000 vertices, η = 2.5
- Edge-sampling probability s = 0.8
- $D = 3$ (use 3-hop witnesses)
- When $\gamma = 1/[(3-\eta)\ (D-1)+1]$, PLD achieves the best matching accuracy



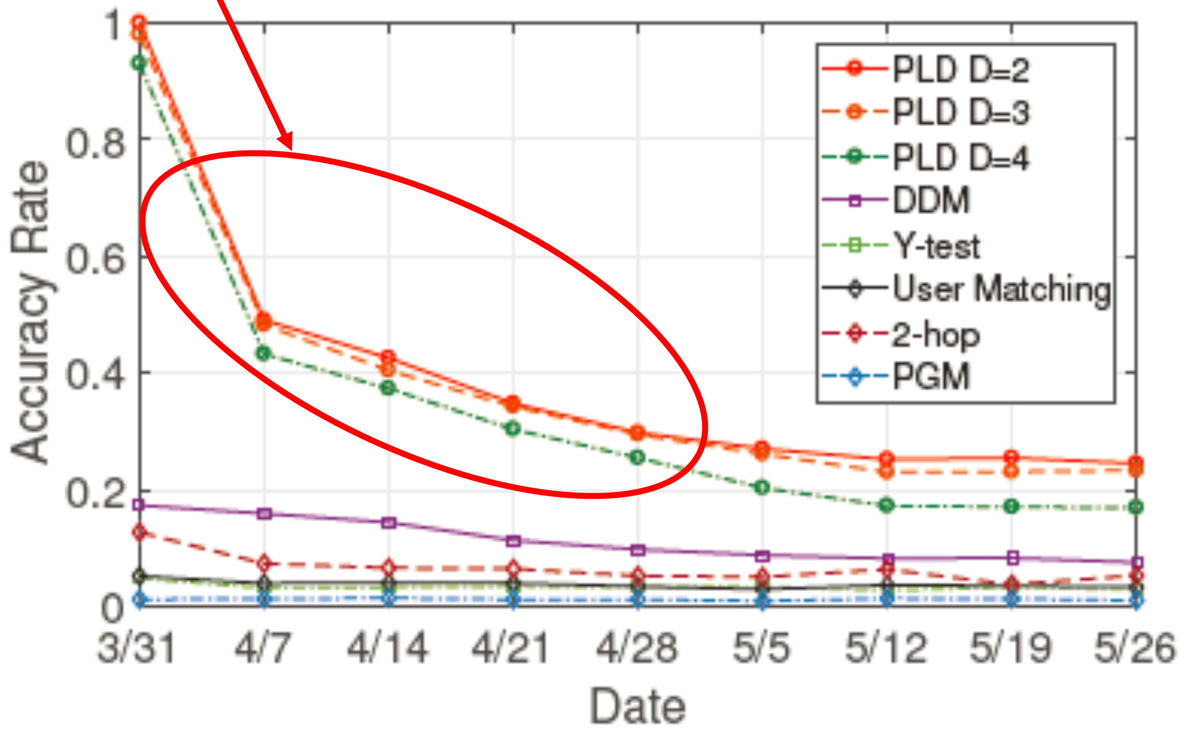
Experimental results: simulated data

- Chung-Lu model with 10000 vertices, η= 2.5
- Edge-sampling probability s = 0.8
- PLD (with optimal $\gamma = 1/[(3-\eta)\ (D-1)+1]$) achieves the best matching accuracy



Experimental results: real data

- An Internet router network observed on 9 days (10K nodes, 22K-23K edges)
- Fraction of seeds $\alpha = 0.01$
- PLD achieves the best matching accuracy



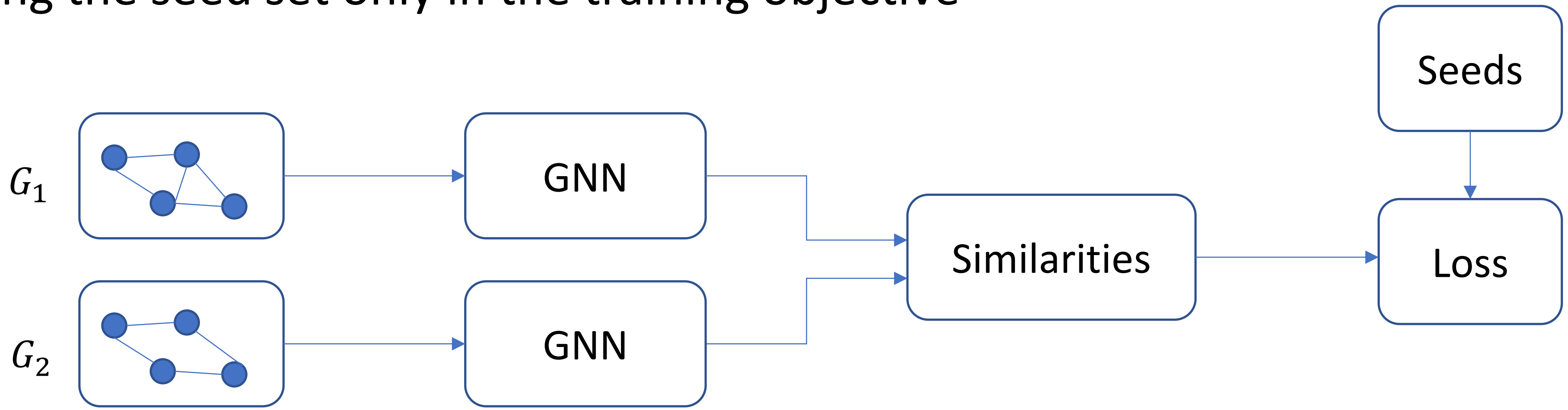
Outline of the remainder

1. Matching correlated ER random graphs
2. Matching power-law graphs
3. Seeded graph neural network
4. Conclusion

Prior work

Limited to semi-supervised learning:

- Learn node embedding using a common GNN on each graph
- Using the seed set only in the training objective

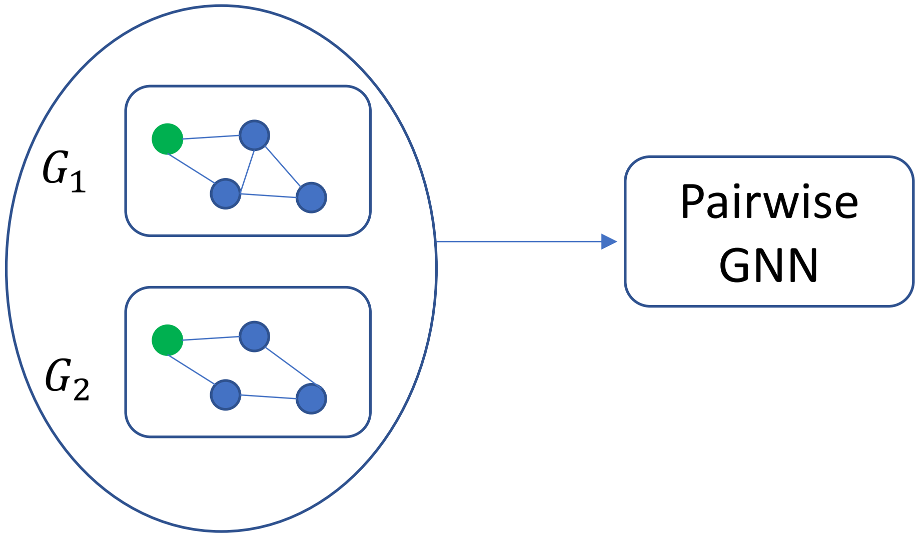


Downsides:

- Require a very large seed set
- Require additional informative node features
- Only learn within a given pair of graphs and do not generalize

Our method: SeedGNN

- Apply the GNN jointly over two graphs:
  - Encode seeds as input
  - learn the node-pair similarities directly



- Only require topological information
- Supervised learning from matched graph pairs and generalizing to unseen graph pairs with only a few seeds

Encode seed information as input

- If the node-pair (i,j) is a seed, then $S_1(i,j) = 1$, and 0 otherwise.

$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 3 & 0 & 0 & 0 \end{array}$$

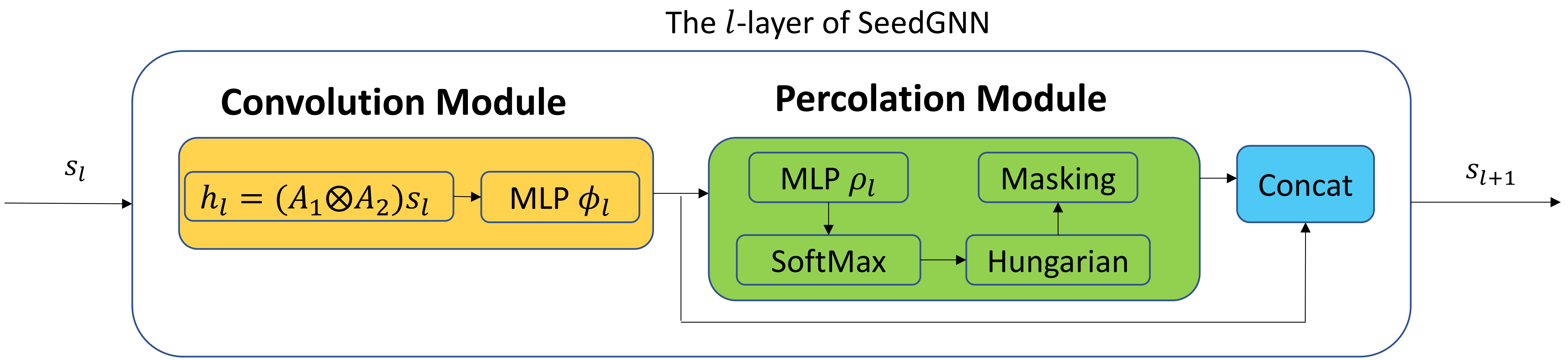
S_1

Matched nodes (Seeds)

Unmatched nodes

- Vectoralization input: $s_1 = \mathrm{vec}(S_1) \in \{0,1\}^{n_1 n_2 \times 1}$

Architecture overview



- Convolution (local): Computing multi-hop witness information
- Percolation (global): Use highly-confident matched pairs as new seeds

Convolution Module

Convolution Module

$h_l = (A_1 \otimes A_2) s_l$ → MLP ϕ_l

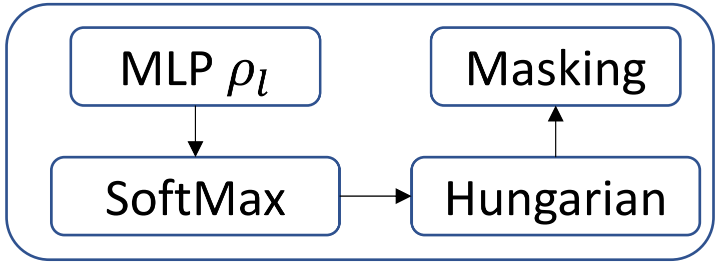
- Count 1-hop witnesses: $h_1 = (A_1 \otimes A_2) s_1$
  - A_i is the adjacent matrix of G_i, $i = 1,2$
  - $h_1[(i-1)n_2 + j, :] = \sum_{(u,v):\, A_1(u,i)=1, A_2(v,j)=1} s_1[(u-1)n_2 + v, :]$ (Neighborhood aggregation)
- Compute l-hop witnesses: $h_l = (A_1 \otimes A_2) s_l$
  - s_l contains witness information within $(l-1)$-hops and new seeds from percolation
- Apply K-layer neural network to combine different types of witness information

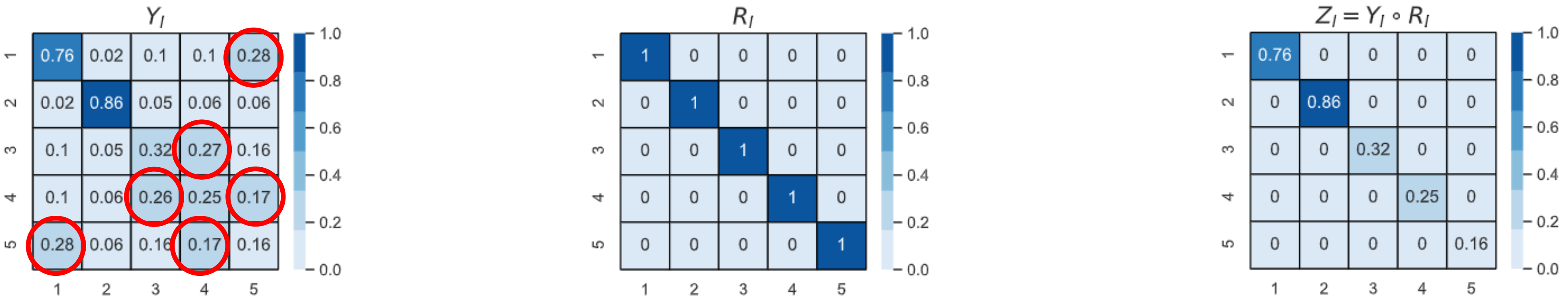
$$m_l = \phi_l(h_l)$$

Percolation Module

- Map vector representations to scalar similarities: $x_l = \rho_l(m_l)$
- Normalization: $Y_l = \mathrm{softmax}(X_l)$
- Similarity matrix contains a lot of "noisy" information:
  - Many fake pairs have comparable similarity with true pairs.

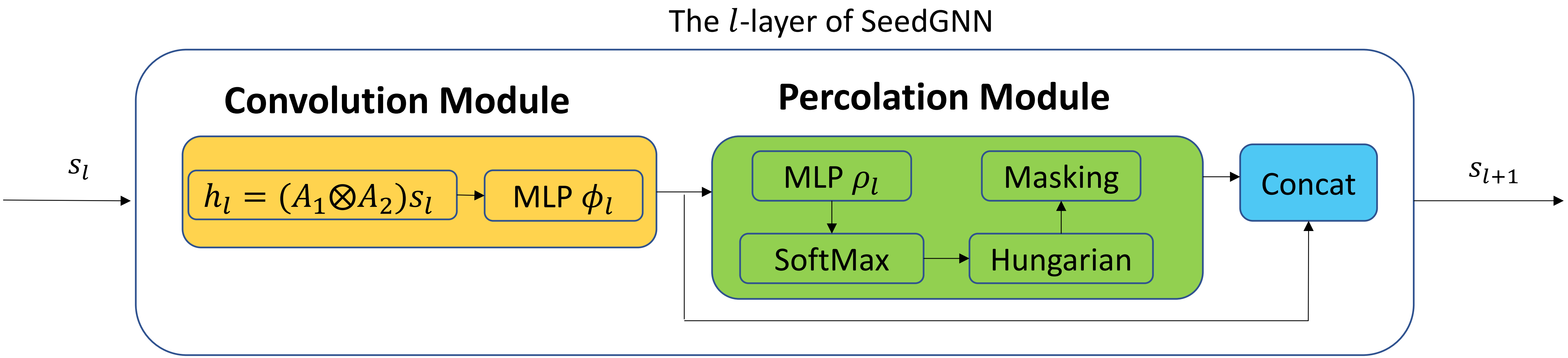
Percolation Module





- Use "Masking" to clean up the "noisy" information:
  - Use the Hungarian algorithm to find highly-confident node-pairs.
  - Discard potential noisy node-pairs

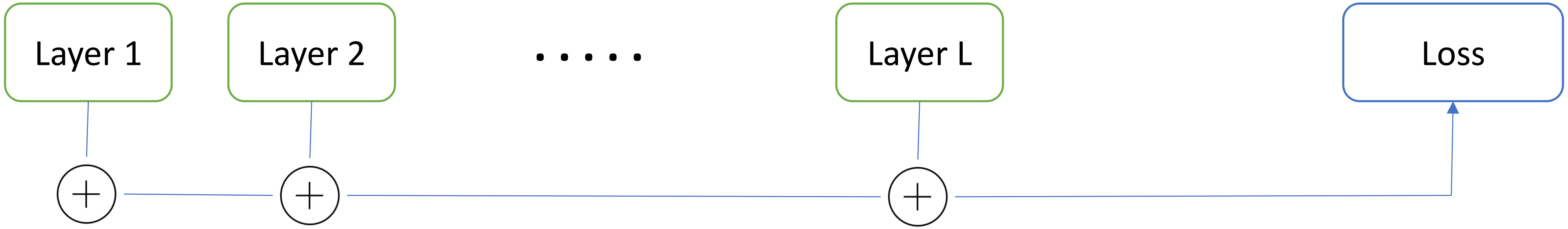
Architecture overview



Convolution + Percolation adaptively decide on using which hops of witness information

- Time complexity: $O(n_1 n_2^2)$
- Space complexity: $O(n_1 n_2)$

Loss function



- For each pair of graphs $\wp$, add up the cross-entropy loss of every layer:

$$Loss_{\wp}(\vartheta) = -\sum_{l=1}^{L}\left(\sum_{(i,j),j=\pi(i)} \log(Y_l(i,j)) + \sum_{(i,j),j\neq\pi(i)} \log(1-Y_l(i,j))\right)$$

- The total loss function is:

$$Loss(\vartheta) = \sum_{\wp\in\, training\ set} Loss_{\wp}(\vartheta)$$

Experimental setting

Training set:

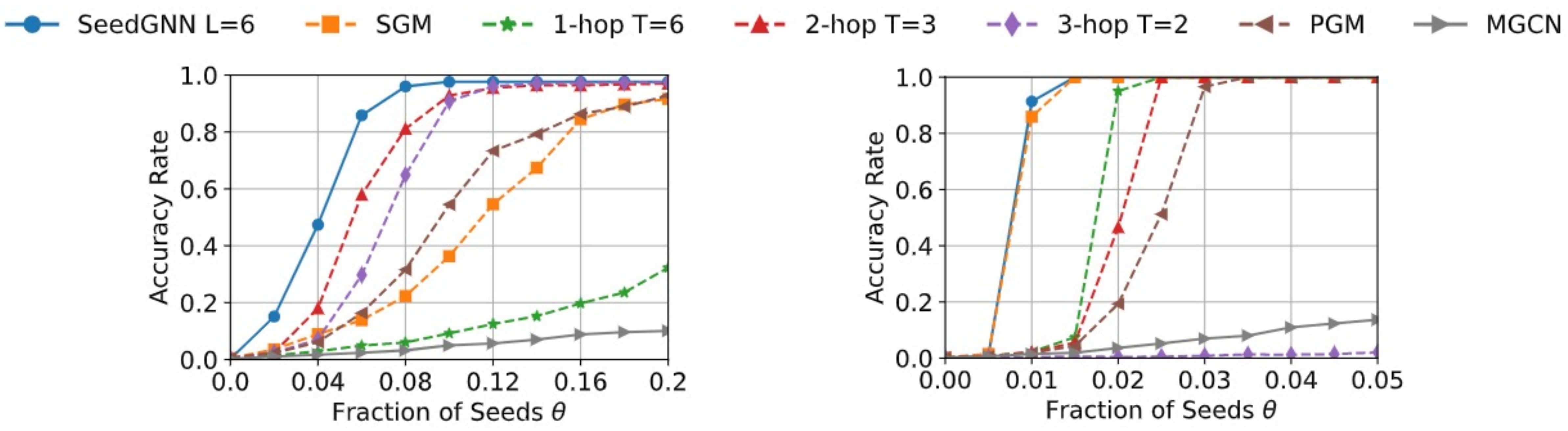
- The correlated Erdős-Rényi graph model:
  - 100 pairs of graphs, $n = 100, p \in \{0.1, 0.3, 0.5\}, s \in \{0.6, 0.8, 1\}$
- Subsampled facebook networks [Traud et al., 2012]: size range from 962 to 32361

Baselines for comparison:

- **D-hop algorithm**: Use D-hop witnesses, iterate T times
- **PGM**: Iteratively match node-pairs with ≥ 2 witnesses as new seeds
- **SGM**: Convex relaxation algorithm using the Frank–Wolfe method
- **PLD**: Designed for power-law graphs
- **MGCN:** Semi-supervised seeded graph matching

Experimental results: correlated Erdős-Rényi graphs

- **Test graph pairs: $n = 500$**, $s = 0.8$, $p = 0.01$ or 0.2.

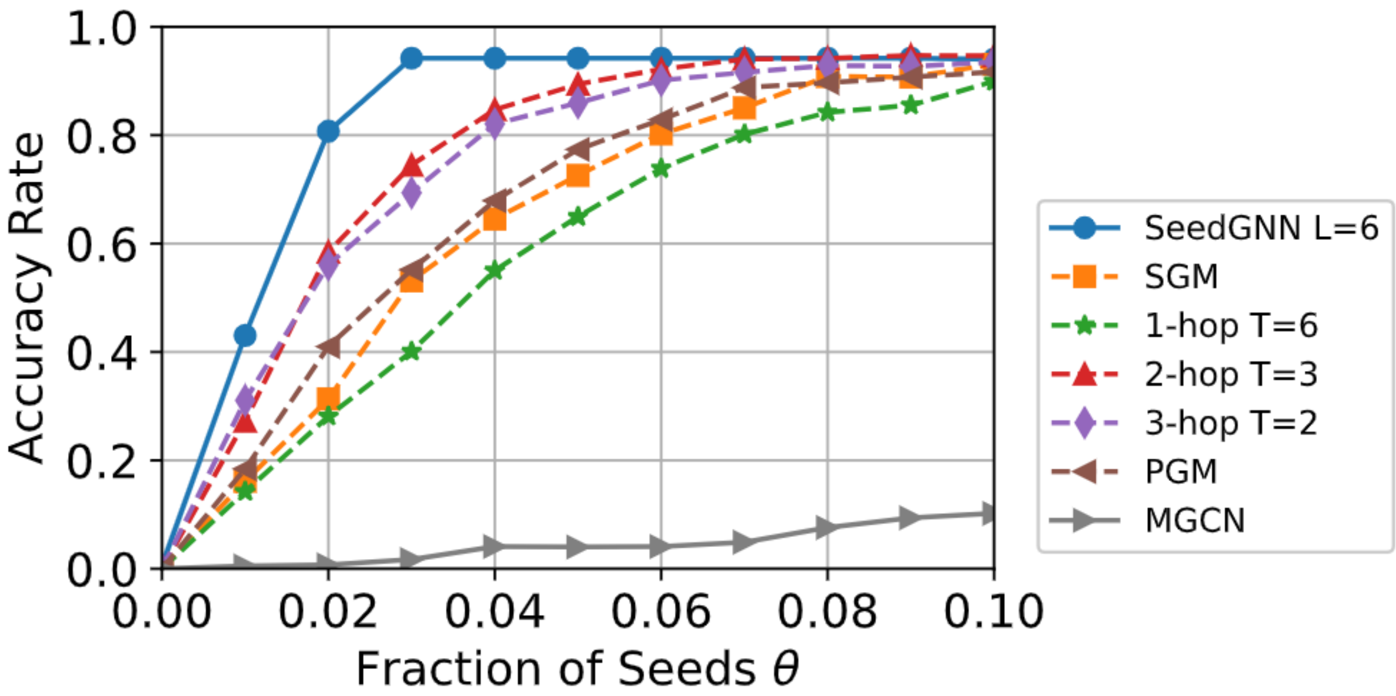


(a) $p = 0.01$

(b) $p = 0.2$

Experimental results: computer vision data

- Matching 3D deformable shapes: each shape is represented by a triangulated mesh graph (8K–11K vertices, vertex degrees highly concentrate on 6)
- The SHREC’16 Dataset is not in the training set



Conclusion

- Develop a new notion of "multi-hop witness" for seeded graph matching
- # of seeds needed for poly-time recovery can be as low as $\Omega(\mathrm{polylog}\, n)$ for matching both ER and power-law graphs
- Design a new graph neural network that learns to compute "multi-hop" witnesses and to match unseen graphs of various types and sizes.