Ph.D. Dissertation Defense

Random Walk-based Large Graph Mining Exploiting Real-world Graph Properties

실세계 그래프 특징을 활용한 랜덤 워크 기반 대규모 그래프 마이닝

Jinhong Jung

Ph.D. Candidate

Dept. of Computer Science & Engineering

Seoul National University

Thesis Committee

문봉기 교수님

서울대학교 컴퓨터공학부 (심사위원장)

강　　유 교수님

서울대학교 컴퓨터공학부 (부심사위원장)

김형주 교수님

서울대학교 컴퓨터공학부

이영기 교수님

서울대학교 컴퓨터공학부

김상욱 교수님

한양대학교 컴퓨터공학부

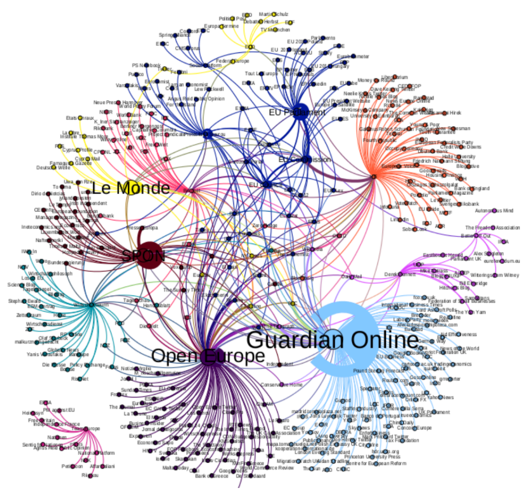
Outline

- **Overview**
- Proposed Methods
- Future Works
- Conclusion

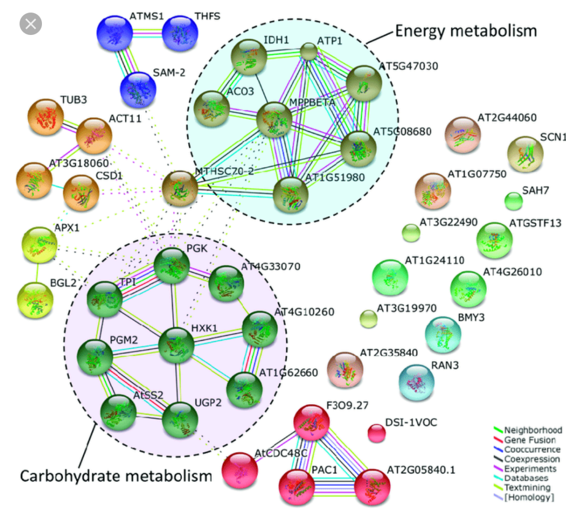
Graphs are Everywhere!

- **Numerous real-world phenomena are represented as graphs!**

Social Network



Hyperlink Network

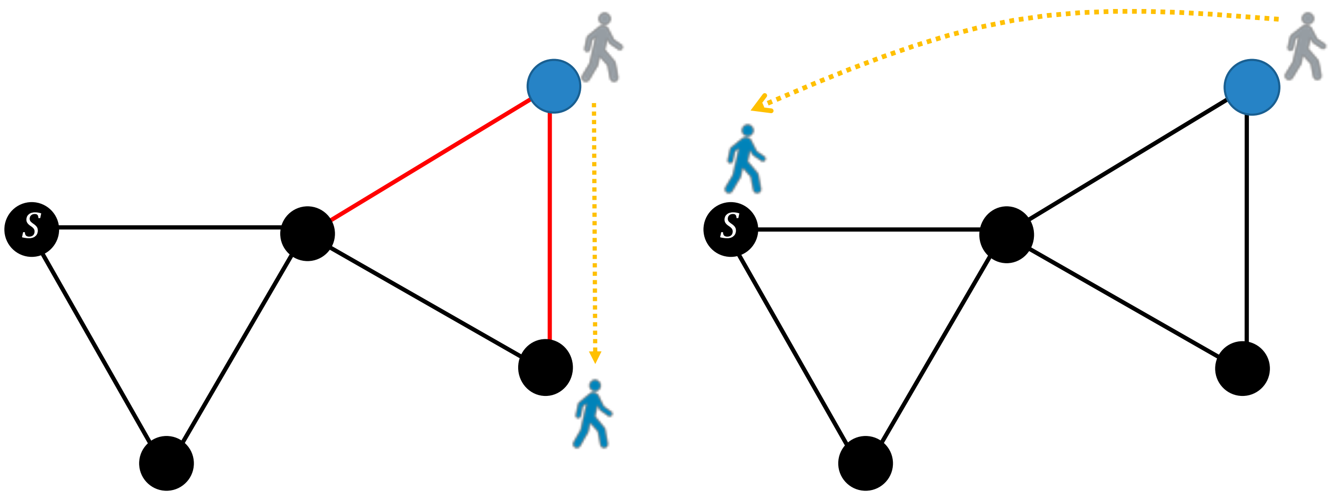


Protein Interaction Network

- Important to analyze such graphs
  - 1) Gain a **better understanding** of real-world events
  - 2) Develop **beneficial applications** on top of the insight

Random Walk in Graphs

- **Random walk has been extensively utilized to analyze real-world graph data**
  - Random Walk with Restart (RWR)
    - **Random walk**: moves to one of neighbors
    - **Restart**: jumps back to query node s



Random walk (with prob. $1 - c$)

Restart (with prob. c)

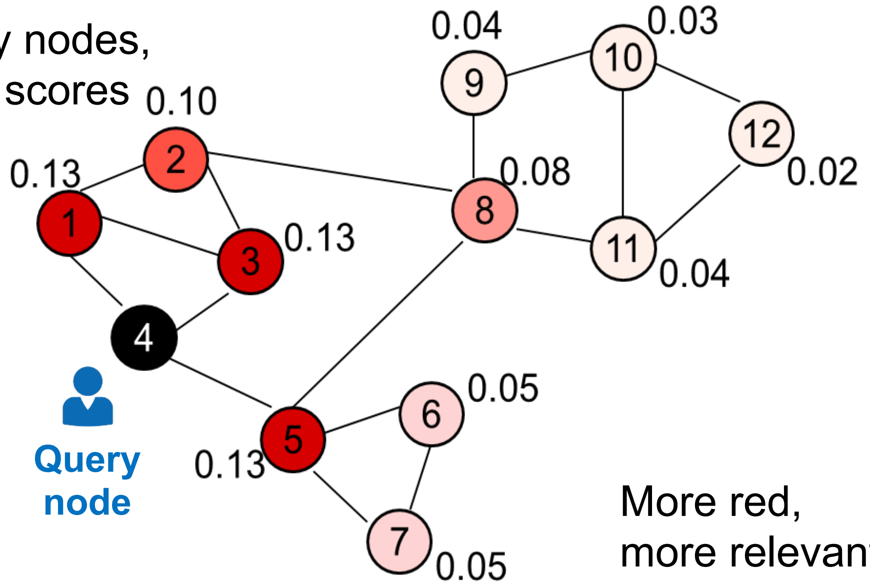
Restart probability

Random Walk with Restart (1)

■ Input and Output of RWR

[Tong et al., ICDM'06]

Nearby nodes, higher scores



More red, more relevant

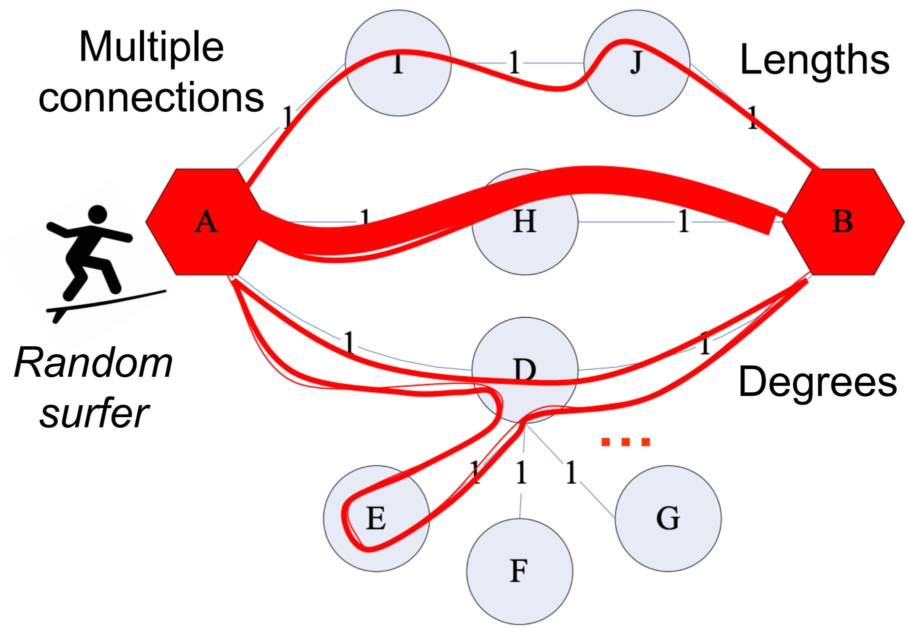
	Node 4
Node 1	0.13
Node 2	0.10
Node 3	0.13
Node 4	0.22
Node 5	0.13
Node 6	0.05
Node 7	0.05
Node 8	0.08
Node 9	0.04
Node 10	0.03
Node 11	0.04
Node 12	0.02

Input: an adjacency matrix A & query node s **Output**: a ranking vector $\boldsymbol{r}$ w.r.t. s

- ***Single-source Random Walk with Restart***
- Provides a **personalized node ranking**

Random Walk with Restart (2)

- **RWR is a fundamental building block on various graph mining applications**



Well reflect **multi-facet relationships** with considering **global network topology**

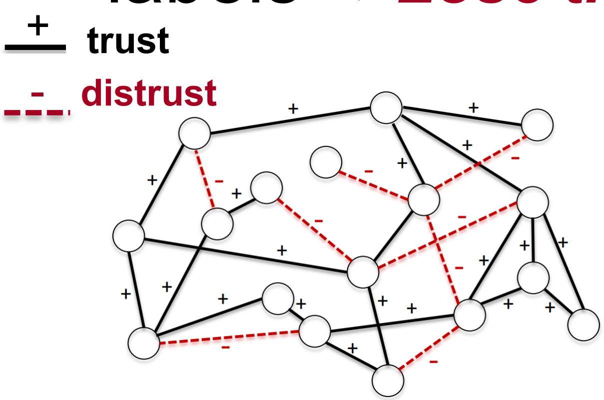
- **Applications**
  - Node Ranking
  - Node embedding
  - Link Prediction
  - Recommendation
  - Anomaly detection
  - Community detection
  - Subgraph mining
  - Image segmentation

Technical Challenges (1)

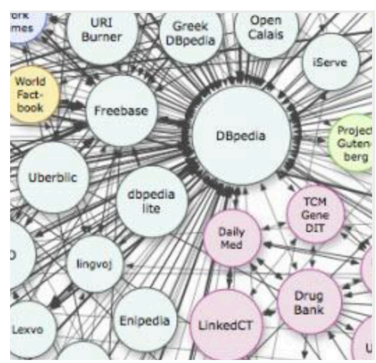
- **Real-world graphs are massive!**
  - e.g., Wikipedia has **40 million** articles, and
    Facebook has **2.41 billion** users
  - Limitations of previous methods for RWR
    - *Exact methods ⇒* ***suffer from speed & scalability***
    - *Approximate methods ⇒* ***too degraded quality***
    - *Top-k methods ⇒* ***limited applications***
- **Extremely challenging to satisfy all of speed, scalability, and exactness**
  - For computing single-source RWR scores ***in such large-scale graphs***

Technical Challenges (2)

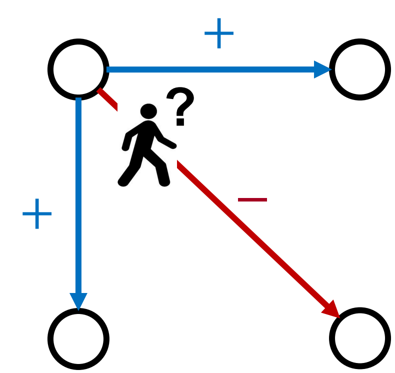
- **Real-world graphs are rich in information!**
  - Various labels to represent complicated relationships between nodes
  - Traditional random surfer does not consider such labels ⇒ ***Lose the identity of a labeled graph***



Signed Networks



Knowledge Bases



Traditional Random Walk

- **How to reflect such labels into random walk?**
  - What do the labels mean for random walk?

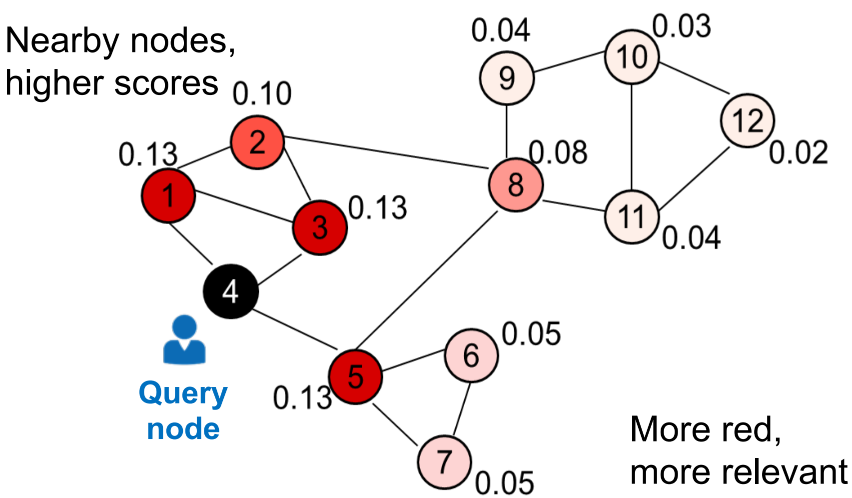
Research Goals and Importance

- **Research Goals**
  - **G1.** To devise **fast**, **scalable**, and **exact** methods for **random walk** in billion-scale graphs
  - **G2.** To design effective **random walk** models utilizing **label** data in labeled graphs
- **Research Importance**
  - **I1.** Advance our understanding of handling ***large graphs*** & ***random walk on labeled graphs***
  - **I2.** Enable us to ***analyze large-scale graphs***
  - **I3.** Lead to ***novel & high-quality*** applications based on random walk in labeled graphs

Research Problems (1)

- **P1.** **Fast**, **scalable** & **exact** RWR computation in **large-scale** graphs
  - To develop a *novel* & *in-memory* algorithm working on a *single* machine
    - *Input graph and intermediate data are stored in memory*

[Tong et al., ICDM'06]

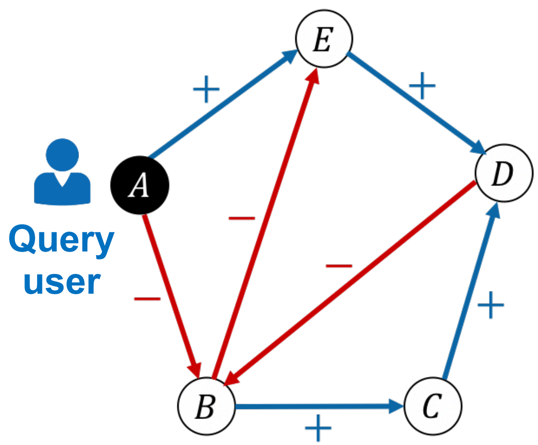


	Node 4
Node 1	0.13
Node 2	0.10
Node 3	0.13
Node 4	0.22
Node 5	0.13
Node 6	0.05
Node 7	0.05
Node 8	0.08
Node 9	0.04
Node 10	0.03
Node 11	0.04
Node 12	0.02

Input: an adjacency matrix $\boldsymbol{A}$ & query node s **Output**: a ranking vector $\boldsymbol{r}$ w.r.t. s

Research Problems (2)

- **P2. Random walk in signed networks** (+/− sign)
  - Effective for *personalized node ranking*
    - **Input:** Signed network G (each edge has + or − sign) having n nodes & Query (or seed) node s
    - **Output:** Trustworthiness (ranking) scores $\boldsymbol{r} \in \mathbb{R}^n$ of all nodes w.r.t. seed node s



Input: a signed network & seed node A

Rank	Node	$\boldsymbol{r}$: Trust-worthiness	$\boldsymbol{r}^+$: Positive score	$\boldsymbol{r}^-$: Negative score
1st	A	0.2500	0.2500	0.0000
2nd	E	0.1487	0.1687	0.0200
3rd	D	0.0703	0.1416	0.0713
4th	C	-0.0549	0.0200	0.0750
5th	B	-0.1465	0.0534	0.1999

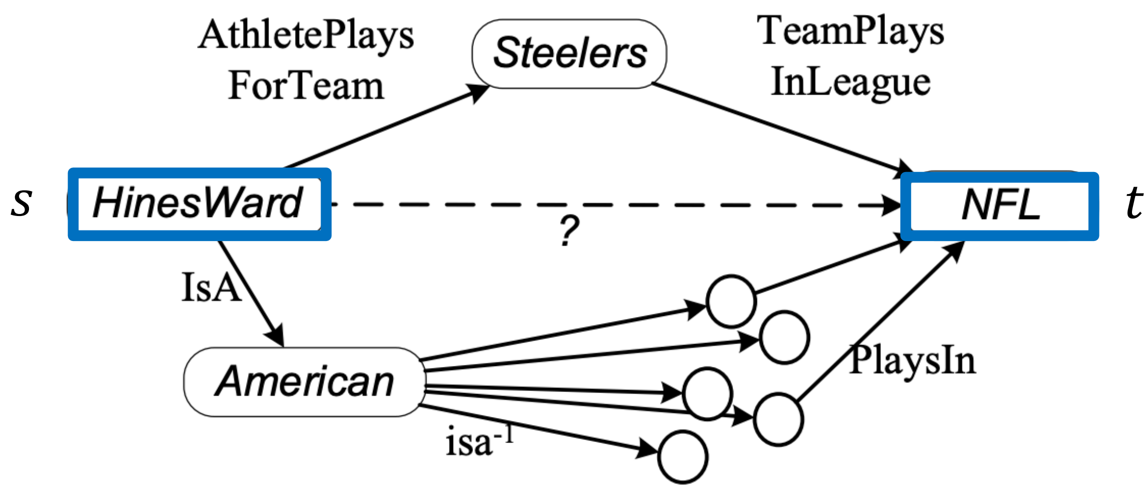
trustful

distrustful

Output: the trustworthiness score vector $\boldsymbol{r}$ w.r.t. the seed node

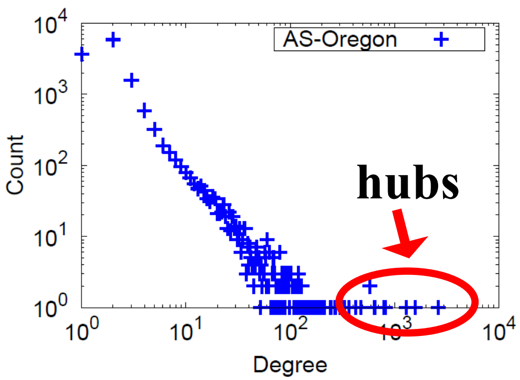
Research Problems (3)

- **P3. Random walk** in **edge-labeled** graphs
  - Each edge has one of K categorical labels
  - Effective for *relational reasoning* b.t.w. two nodes
    - **Input:** Edge-labeled graph G (each edge has one of K categorical labels) & Two nodes s and t
    - **Output:** K relevance scores on t w.r.t s

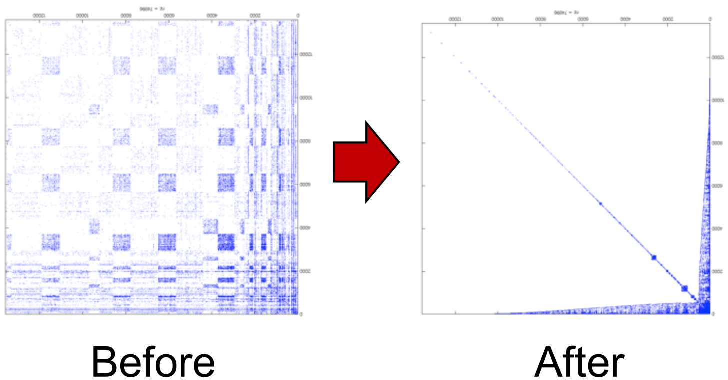


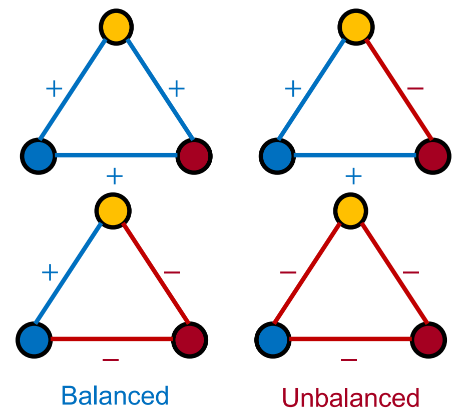
Main Approaches

- **A1. Real-world Graph Properties**
  - e.g., Power-law degree distribution / balance theory



[Kang et al., ICDM'11]





- **A2. Numerical Computing Methods**
  - To boost the computational speed on adjacency matrices
- **A3. Linear Algebra & Stochastic Process**
  - To design new random walk models in labeled graphs

Outline

- Overview
- **Proposed Methods**
- Future Works
- Conclusion

Proposed Methods

- **Random Walk-based Large Graph Mining Exploiting Real-world Graph Properties**

Current Works *(Ph.D. Course)*		
Plain Graphs (No edge labels)	**Signed Graphs (Two edge labels)**	**Edge-labeled Graphs (K edge labels)**
Fast Scalable & Exact RWR in Billion-scale Graphs **BePI** [SIGMOD'17]	Random Walk in Signed Graphs: Personalized Ranking **SRWR** [ICDM'16] [KAIS'19]	Random Walk in Edge-labeled Graphs: Relational Reasoning **MuRWR** [WWWJ'20]

Proposed Methods

- **Random Walk-based Large Graph Mining Exploiting Real-world Graph Properties**

Current Works ***(Ph.D. Course)***

Plain Graphs (No edge labels)	Signed Graphs (Two edge labels)	Edge-labeled Graphs (K edge labels)
Fast Scalable & Exact RWR in Billion-scale Graphs	Random Walk in Signed Graphs: Personalized Ranking	Random Walk in Edge-labeled Graphs: Relational Reasoning
BePI	**SRWR**	**MuRWR**
[SIGMOD'17]	[ICDM'16] [KAIS'19]	[WWWJ'20]

Introduction

- **Problem:** **Random Walk with Restart**
  - **Input:** Adjacency matrix $\mathbf{A}$ of a graph having n nodes & Query (or seed) node s
  - **Output:** Relevance (ranking) scores $\boldsymbol{r} \in \mathbb{R}^n$ of all nodes w.r.t. seed node s
  - *In-memory computation on a single machine*
  - **Recursive Equation**
    - $$\mathbf{r} = \underbrace{(1-c)\widetilde{\mathbf{A}}^{\mathbf{T}}\mathbf{r}}_{\text{Random Walk}} + \underbrace{c\mathbf{q}_s}_{\text{Restart}}$$ ← *Query vector (s-th unit vector)*
      - c is called restart probability
  - **Linear System**
    - $$\left(\mathbf{I} - (1-c)\widetilde{\mathbf{A}}^{\mathbf{T}}\right)\mathbf{r} = c\mathbf{q}_s$$
      $$\Leftrightarrow \mathbf{H}\mathbf{r} = c\mathbf{q}_s$$

Challenges

- **Q. How to compute exact RWR scores quickly on very large graphs?**
  - ***Iterative Methods*** iteratively update RWR scores until convergence
    - e.g., power iteration: $\mathbf{r}^{(t)} \leftarrow (1-c)\tilde{\mathbf{A}}^{\mathbf{T}}\mathbf{r}^{(t-1)} + c\mathbf{q}_s$
    - **Pros:** scale to very large-graphs $\Leftarrow O(m)$ space
    - **Cons:** slow query speed $\Leftarrow O(Tm)$ query time

T: # of iterations
m: # of edges
n: # of nodes

  - ***Preprocessing Methods*** compute RWR scores directly from precomputed data
    - e.g., matrix inversion: $\mathbf{r} = c\mathbf{H}^{-1}\mathbf{q}_s$ where $\mathbf{H} = (\mathbf{I} - (1-c)\tilde{\mathbf{A}}^{\mathbf{T}})$
    - **Pros:** fast query speed $\Leftarrow O(n)$ query time
    - **Cons:** cannot handle very large graphs $\Leftarrow O(n^3)$ prep. time
      $O(n^2)$ space

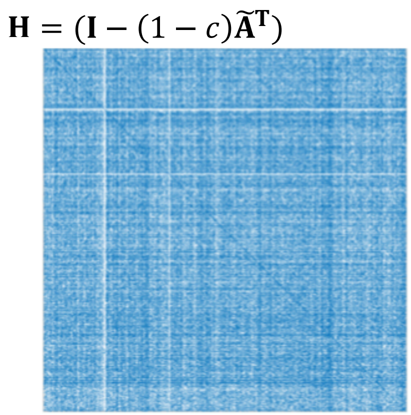
Why Important?

- **I1) Why Fast & Scalable RWR computation?**
  - Improve computational performance of various applications based on RWR in large graphs
- **I2) Why exact RWR computation?**
  - Existing approximate methods dramatically degrade the quality of applications using RWR
- **I3) Why all nodes' scores w.r.t. seed?**
  - Previous top-k approaches focus on getting top-k nodes, not their scores
  - Lots of applications still rely on the scores of all nodes $\Rightarrow$ e.g., anomaly detection, local clustering, subgraph mining

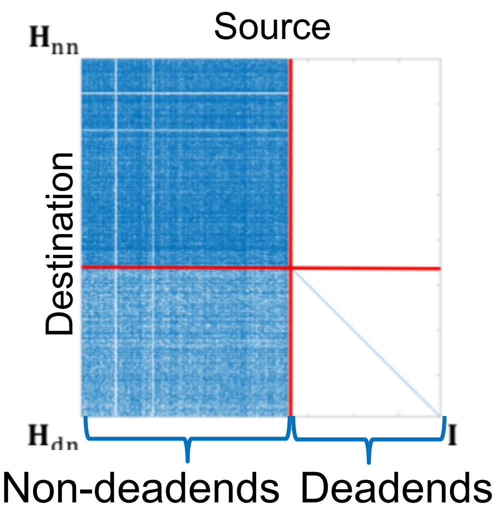
Proposed Method: BePI (1)

- **BePI** (**Be**st of **P**reprocessing and **I**terative approaches)
  - A fast and scalable method by taking the advantages of both preprocessing and iterative approaches
- Key Ideas
  - **Idea 1) Exploit real-world graph structures** to make it easy-to-preprocess
  - **Idea 2) Incorporate an iterative method** to increase the scalability
  - **Idea 3) Optimize the performance of the iterative method** to accelerate RWR computation speed

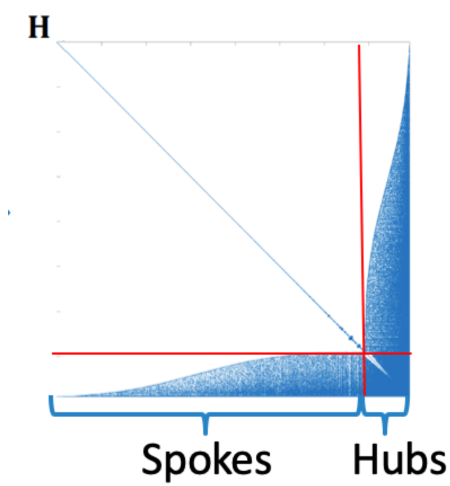
Real-world Graph Properties



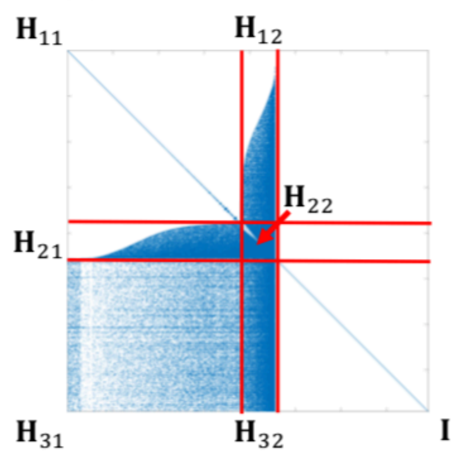
(a) Original matrix $\mathbf{H}$



(b) Deadend reordering



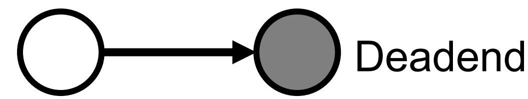
(c) Hub-and-spoke reordering



(d) Deadend and hub-and-spoke reordering

Deadend

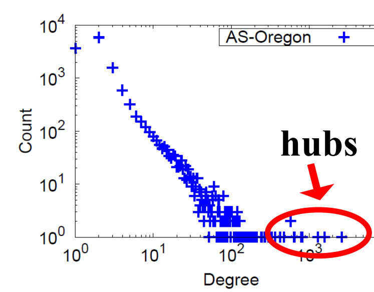
- Deadend is a node having no out-going edges, e.g., an image in a web-document graph [Langville et al., JSC'06]



Ratio of deadend: 5~40%

Hub-and-spoke

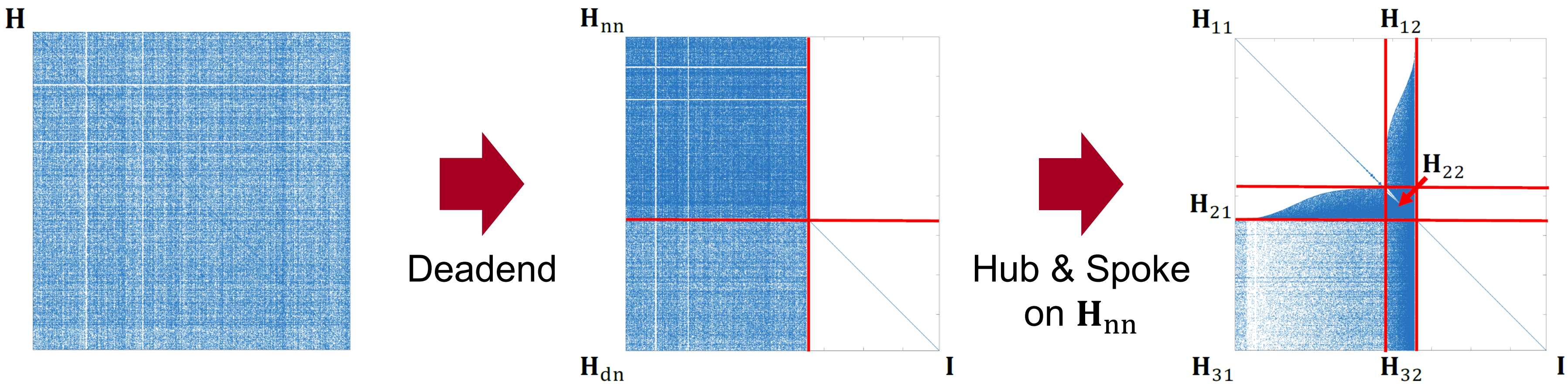
- **Hubs** are high degree nodes, **spokes** are low degree nodes
- Few hubs, and a majority of spokes in real-world graphs [Kang et al., ICDM'11]



Ratio of hub: 5~20%

Proposed Method: BePI (2)

- **Idea 1) Exploit real-world graph structures** to make it easy-to-preprocess

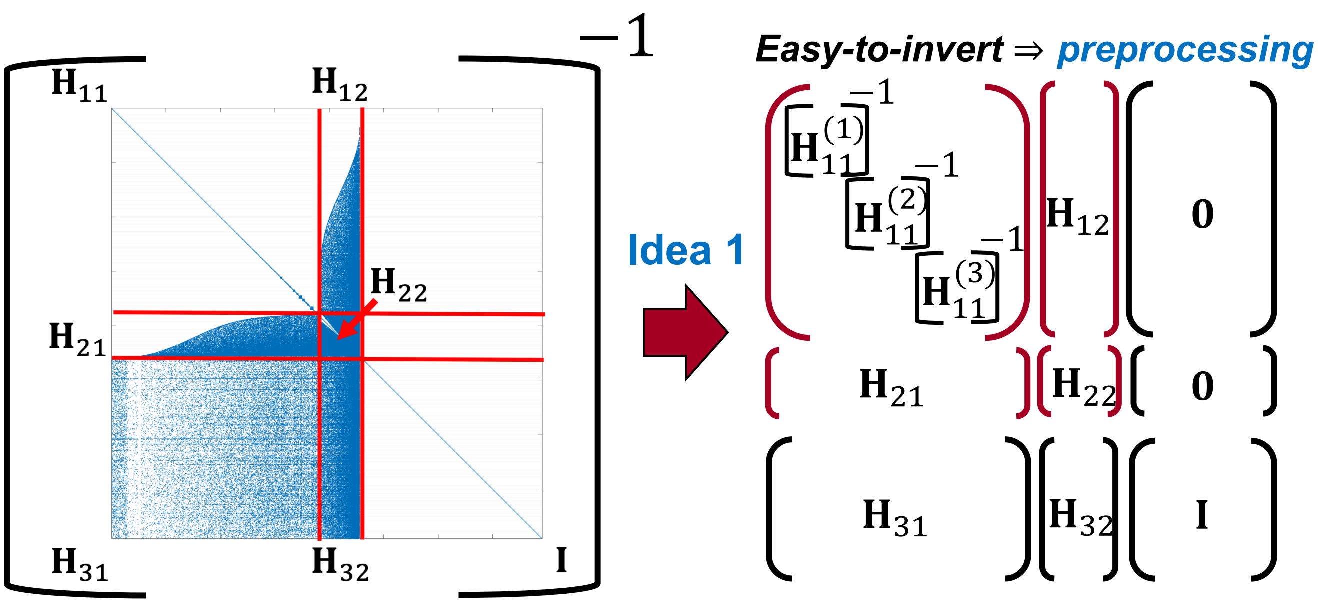


$\mathbf{H}_{11}$ is a block diagonal matrix!

$$\mathbf{H}\mathbf{r} = c\mathbf{q}_s \Leftrightarrow \begin{bmatrix} \mathbf{H}_{11} & \mathbf{H}_{12} & \mathbf{0} \\ \mathbf{H}_{21} & \mathbf{H}_{22} & \mathbf{0} \\ \mathbf{H}_{31} & \mathbf{H}_{32} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix} = c \begin{bmatrix} \mathbf{q}_1 \\ \mathbf{q}_2 \\ \mathbf{q}_3 \end{bmatrix}$$

Proposed Method: BePI (3)

- RWR is obtained by solving a linear system on the reordered matrix ($\mathbf{r} = c\mathbf{H}^{-1}\mathbf{q}_s$)
  - **Efficiently solved by handling smaller blocks**



(Block Elimination)

[Boyd et al., 2009] [Shin et al., 2015]

Proposed Method: BePI (4)

- Apply **block elimination** as a preprocessing approach

Details

$$\mathbf{H}\mathbf{r}_s = c\mathbf{q}_s \Leftrightarrow \begin{bmatrix} \mathbf{H}_{11} & \mathbf{H}_{12} & \mathbf{0} \\ \mathbf{H}_{21} & \mathbf{H}_{22} & \mathbf{0} \\ \mathbf{H}_{31} & \mathbf{H}_{32} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix} = c \begin{bmatrix} \mathbf{q}_1 \\ \mathbf{q}_2 \\ \mathbf{q}_3 \end{bmatrix}$$

Block elimination ➡

$$\begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{11}^{-1}(c\mathbf{q}_1 - \mathbf{H}_{12}\mathbf{r}_2) \\ \mathbf{S}^{-1}(c\mathbf{q}_2 - c\mathbf{H}_{21}\mathbf{H}_{11}^{-1}\mathbf{q}_1) \\ c\mathbf{q}_3 - \mathbf{H}_{31}\mathbf{r}_1 - \mathbf{H}_{32}\mathbf{r}_2 \end{bmatrix}$$

$\mathbf{S} = \mathbf{H}_{22} - \mathbf{H}_{21}\mathbf{H}_{11}^{-1}\mathbf{H}_{12}$, the Schur complement of $\mathbf{H}_{11}$

Precompute the blue-colored matrices to make RWR computation fast!

Proposed Method: BePI (5)

- **Idea 2) Incorporate an iterative method** to increase the scalability
  - Hard to invert $\boldsymbol{S}$ in large graphs ($\dim(\boldsymbol{S})$ = # of hubs $\simeq 10^6$)
  - $\Rightarrow$ Solve the system on $\boldsymbol{S}$ iteratively (GMRES) [Saad et al., 1986]

Details

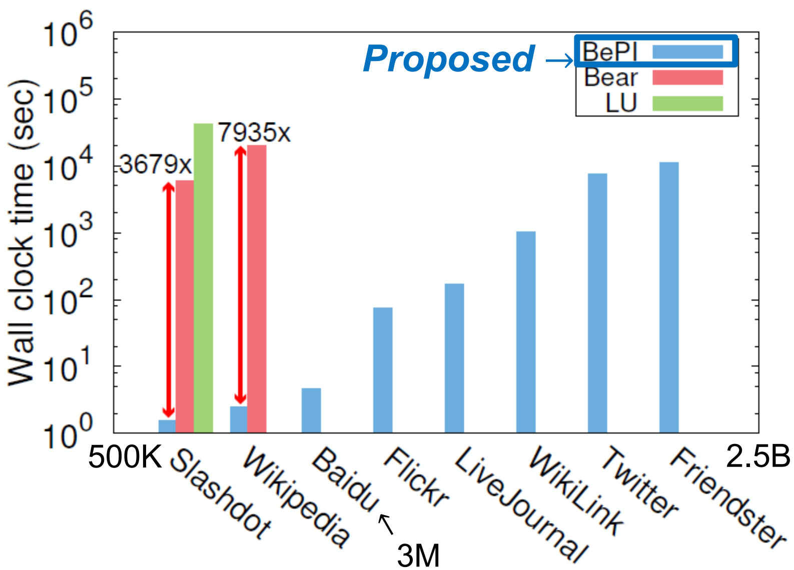
$$\mathbf{r}_2 = \mathbf{S}^{-1}\underbrace{(c\mathbf{q}_2 - c\mathbf{H}_{21}\mathbf{H}_{11}^{-1}\mathbf{q}_1)}_{\triangleq \tilde{\mathbf{q}}_2} \Leftrightarrow \mathbf{S}\mathbf{r}_2 = \tilde{\mathbf{q}}_2$$

- **Idea 3) Optimize the performance of the iterative method** to accelerate RWR speed
  - e.g., Preconditioning for faster convergence

The sophisticated combination of these techniques leads to fast & scalable RWR with the guarantee of exactness

Experimental Results (1)

- Experimental settings
  - Machine: single machine with 500GB memory
  - Data: real-world large-scale graphs (up to billion-scale)
  - Competitors: Bear & LU (Prep.), Power & GMRES (Iter.)
- **Preprocessing time**

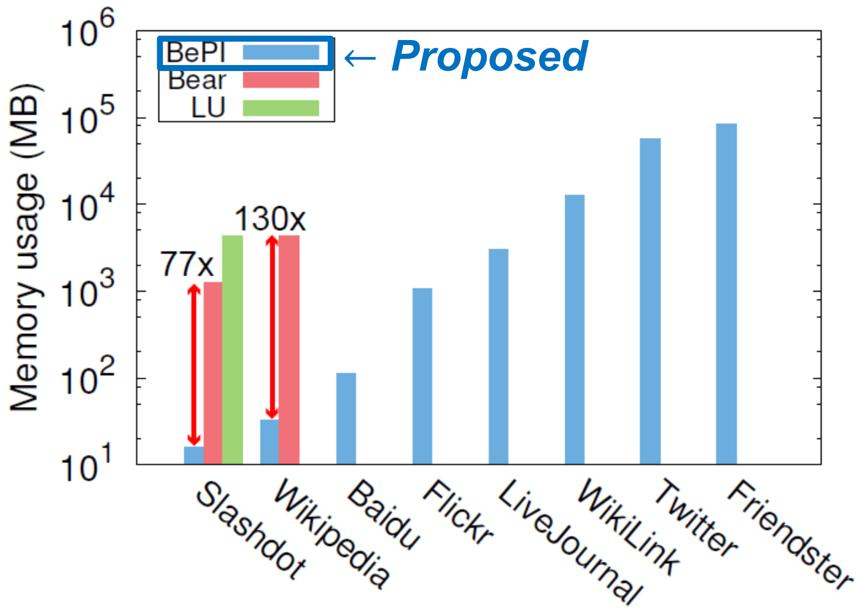


(a) Preprocessing time

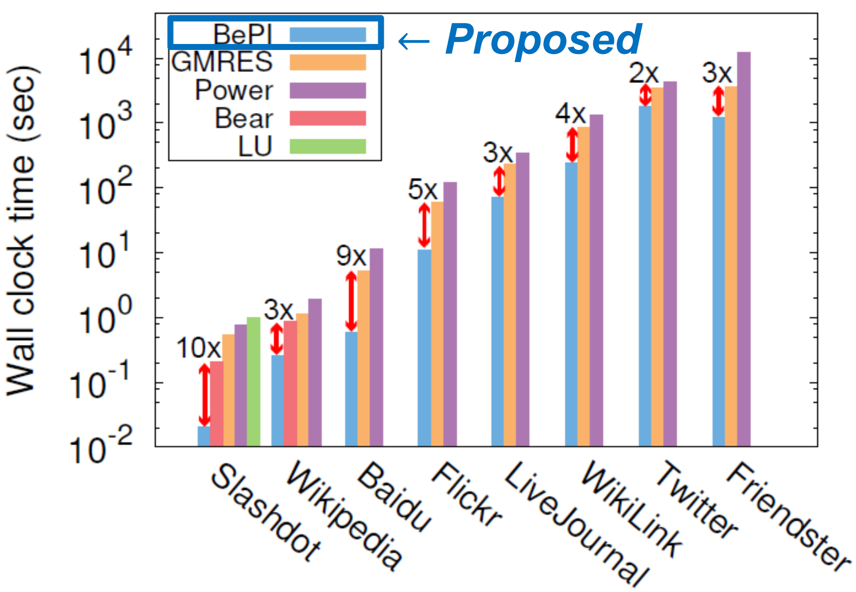
- **BePI is significantly faster** than other preprocessing methods
- **Only BePI successfully scales to** the largest graph (Friendster, 2.5B edges)

Experimental Results (2)

- **Memory requirement and query time**
  - Competitors: **Bear** & **LU** (Prep.), **Power** & **GMRES** (Iter.)



(b) Memory space for preprocessed data



(c) Query time

BePI requires 130× less memory space & computes RWR 9× faster!

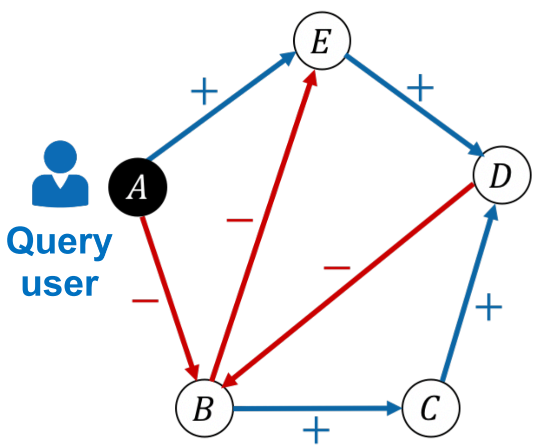
Proposed Methods

- **Random Walk-based Large Graph Mining Exploiting Real-world Graph Properties**

Current Works *(Ph.D. Course)*		
Plain Graphs (No edge labels)	**Signed Graphs (Two edge labels)**	**Edge-labeled Graphs (K edge labels)**
Fast Scalable & Exact RWR in Billion-scale Graphs	Random Walk in Signed Graphs: Personalized Ranking	Random Walk in Edge-labeled Graphs: Relational Reasoning
BePI	**SRWR**	**MuRWR**
[SIGMOD'17]	[ICDM'16] [KAIS'19]	[WWWJ'20]

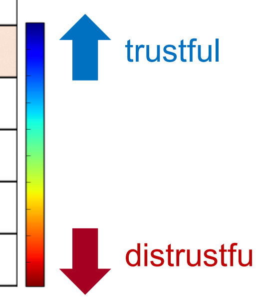
Introduction

- **Problem: Personalized Ranking in Signed Networks**
  - **Input:** Signed network G (each edge has $+$ or $-$ sign) having n nodes & Query (or seed) node s
  - **Output:** Trustworthiness (ranking) scores $\boldsymbol{r} \in \mathbb{R}^n$ of all nodes w.r.t. seed node s



Input: a signed network & seed node A

Rank	Node	$\boldsymbol{r}$: Trust-worthiness	$\boldsymbol{r}^+$: Positive score	$\boldsymbol{r}^-$: Negative score
1st	A	0.2500	0.2500	0.0000
2nd	E	0.1487	0.1687	0.0200
3rd	D	0.0703	0.1416	0.0713
4th	C	-0.0549	0.0200	0.0750
5th	B	-0.1465	0.0534	0.1999

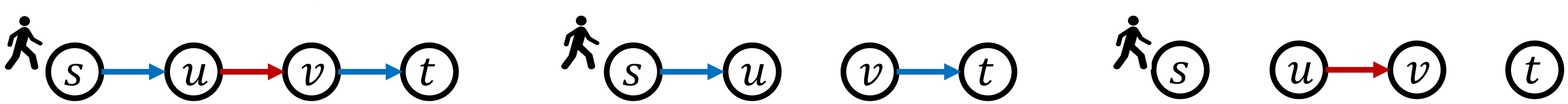


Output: the trustworthiness score vector $\boldsymbol{r}$ w.r.t. the seed node

Limitations

- **Naïve approaches fail to provide proper personalized ranking in signed network G**
  - RWR after removing signs from G
    - $\Rightarrow$ No consideration on distrustful relationships
  - Modified RWR (M-RWR)
    - **Step 1.** Split G into G^+ and G^- (i.e., $G = G^+ \cup G^-$)
    - **Step 2.** Positive RWR scores $\boldsymbol{r}^+$ on G^+ & Negative RWR scores $\boldsymbol{r}^-$ on G^-
    - **Step 3.** Trustworthiness scores $\boldsymbol{r} = \boldsymbol{r}^+ - \boldsymbol{r}^-$
    - $\Rightarrow$ Many connections are broken

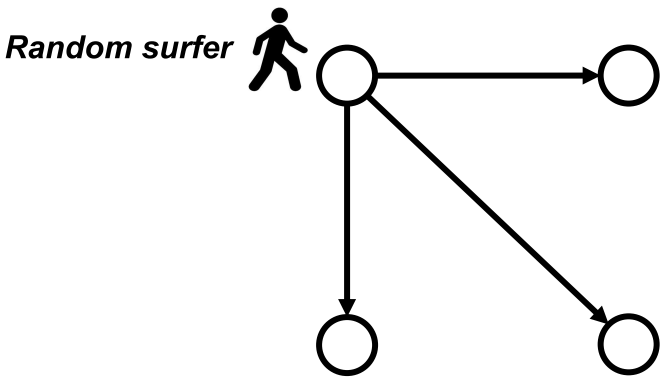
It cannot reach t on both G^+ & G^-



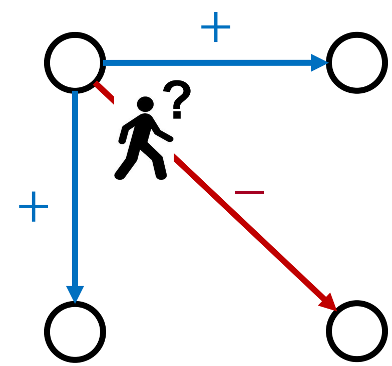
Signed Graph G | Positive Graph G^+ | Negative Graph G^-

Challenges

- **Q. How to deal with signed edges for random walks?**



Traditional random surfer
only consider unsigned edges



No rules for handling
signed edges

- **Importance**
  - Lead to proper personalized node ranking scores in signed network *(More trustful ⇒ Higher ranking)*
  - Enable us to effectively analyze signed networks based on random walk (link prediction, anomaly detection, etc.)

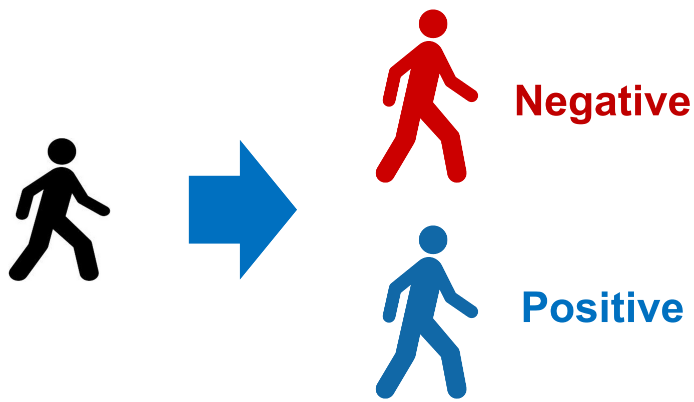
Proposed Method: SRWR (1)

- **SRWR** (**S**igned **R**andom **W**alk with **R**estart)
  - Personalized node ranking in signed networks
    - **Idea 1) Introduce sign into random surfer**
    - **Idea 2) Adopt balance theory to signed surfer**
      - The theory describes ***signed triangle pattern***, a distinct structure in real-world signed networks
  - Two methods for SRWR
    - **SRWR-Iter**: Iteratively computes SRWR scores
    - **SRWR-Pre**: Efficiently computes SRWR scores in a preprocessing manner
      - **Idea 3) Exploit real-world graph structures**

Proposed Method: SRWR (2)

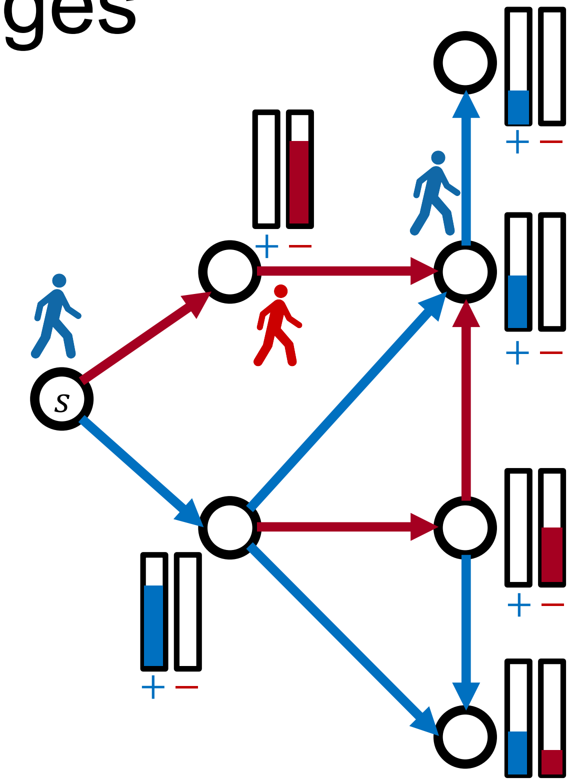
- **Idea 1) Introduce a sign into a random surfer** to handle signed edges

Signed Random Surfer



Traditional random surfer

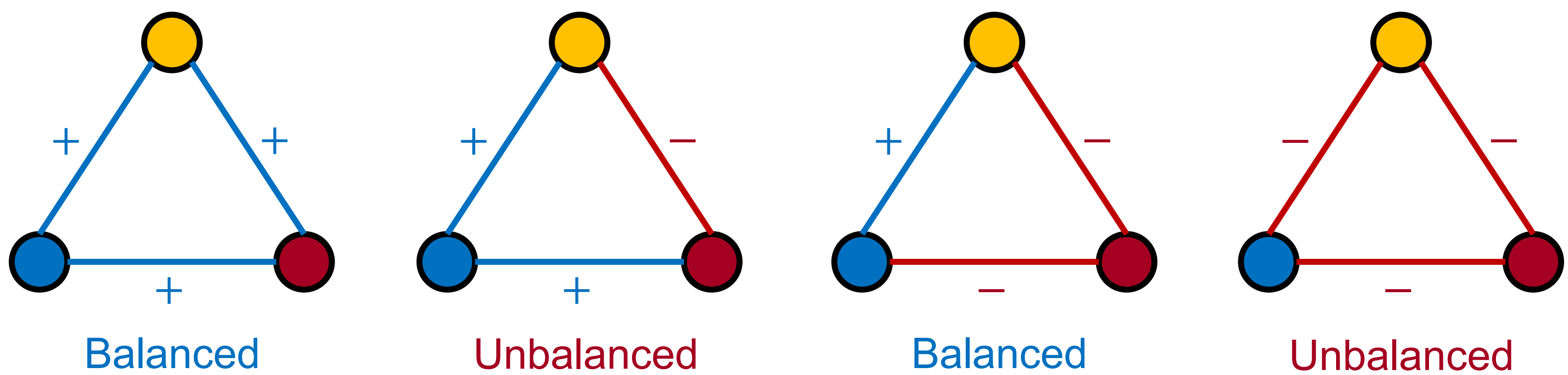
Signed random surfer (proposed)



How to change the surfer's sign?
⇒ Balance Theory

Real-world Graph Properties

- **Balance Theory: Real-world Signed Networks are Balanced!**
  - There are **88~92%** balanced triangles

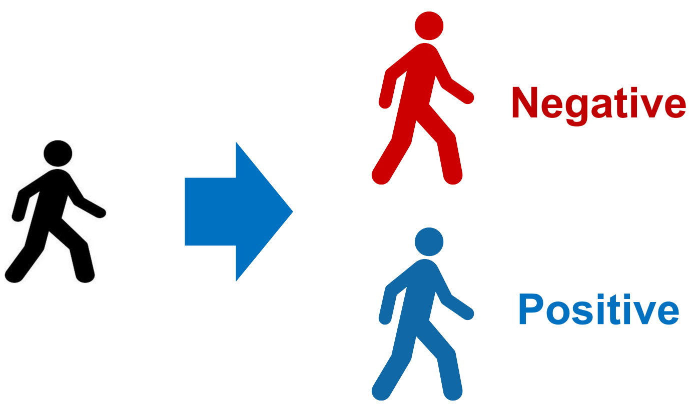


  - Examples
    - a) Friend of **my** **friend** is **my friend**! ⇒ balanced
    - b) Enemy of **my** **friend** is **my friend**? ⇒ unbalanced
    - c) Enemy of **my** **friend** is **my enemy**! ⇒ balanced
    - d) Enemy of **my** **enemy** is **my enemy**? ⇒ unbalanced

Proposed Method: SRWR (3)

- **Idea 2) Adopt balance theory to the signed surfer**

Signed Random Surfer



Traditional random surfer

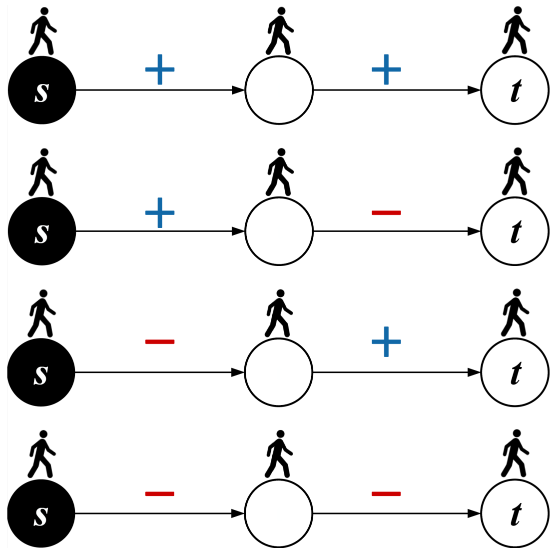
Signed random surfer (proposed)

Rules from Balance Theory

1) **Friend** of my **friend** is my **friend**
2) **Enemy** of my **friend** is my **enemy**
3) **Friend** of my **enemy** is my **enemy**
4) **Enemy** of my **enemy** is my **friend**

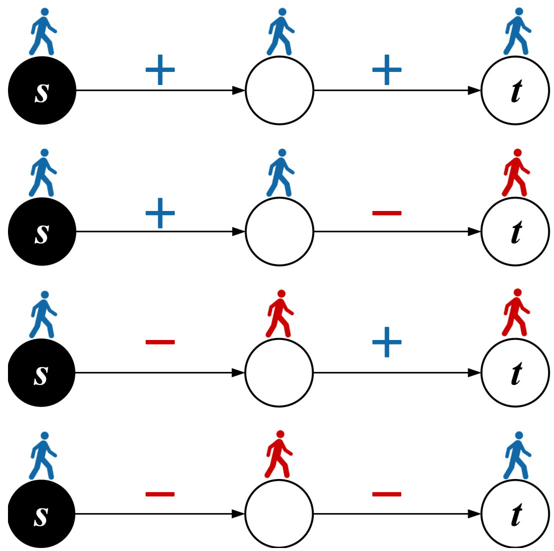
Proposed Method: SRWR (4)

- **Idea 2) Adopt balance theory to the signed surfer**
  - Flip surfer's sign if she encounters negative edges



Traditional random walk

Cannot identify node t



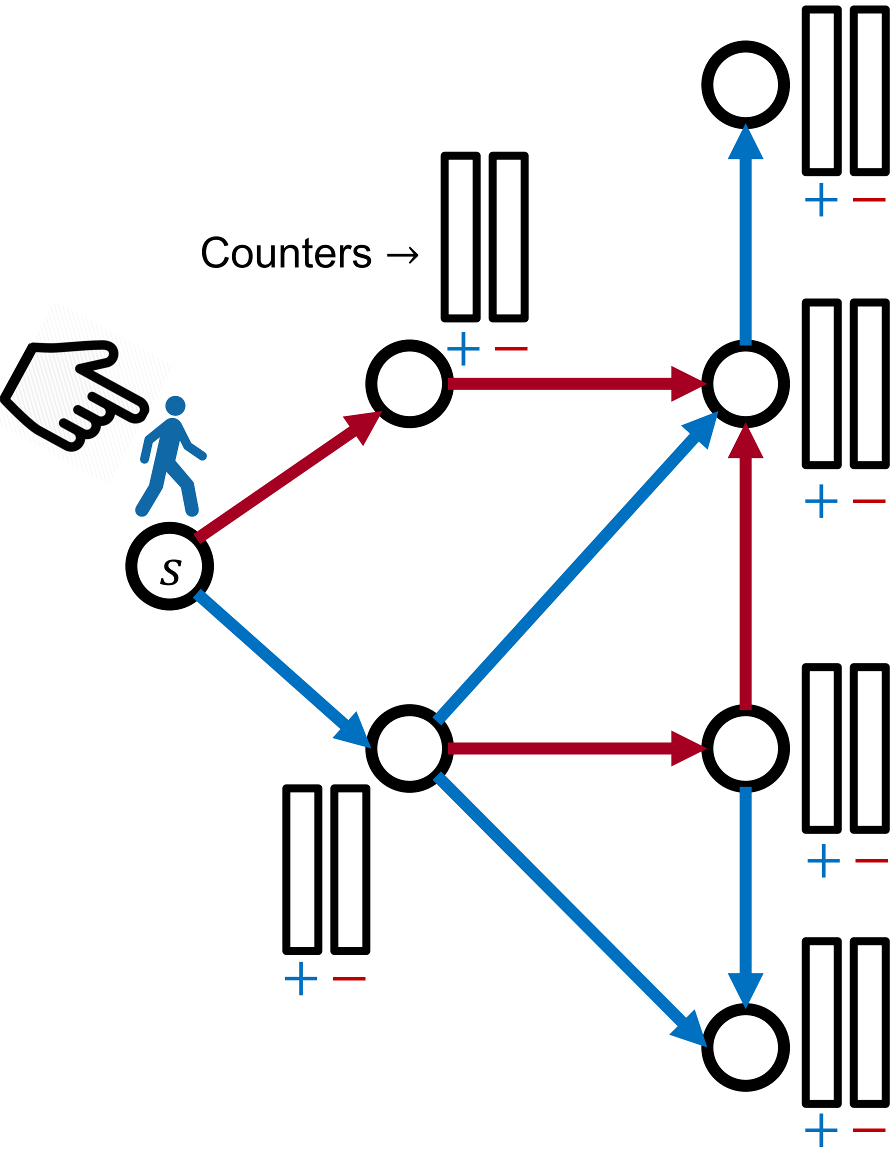
Signed random walk

Consistent with balance theory

Proposed Methods: SRWR (5)

- **Signed Random Walk with Restart Model**
  - **Action 1: Signed Random Walk**
    - The surfer randomly moves to one of neighbors from node u with prob. $1 - c$
    - *She flips her sign if she encounters a **negative** edge*
  - **Action 2: Restart**
    - The surfer goes back to the query node s with prob. c
    - *Her sign should become **positive** at the query node*

Example of SRWR (1)



Start from query node s

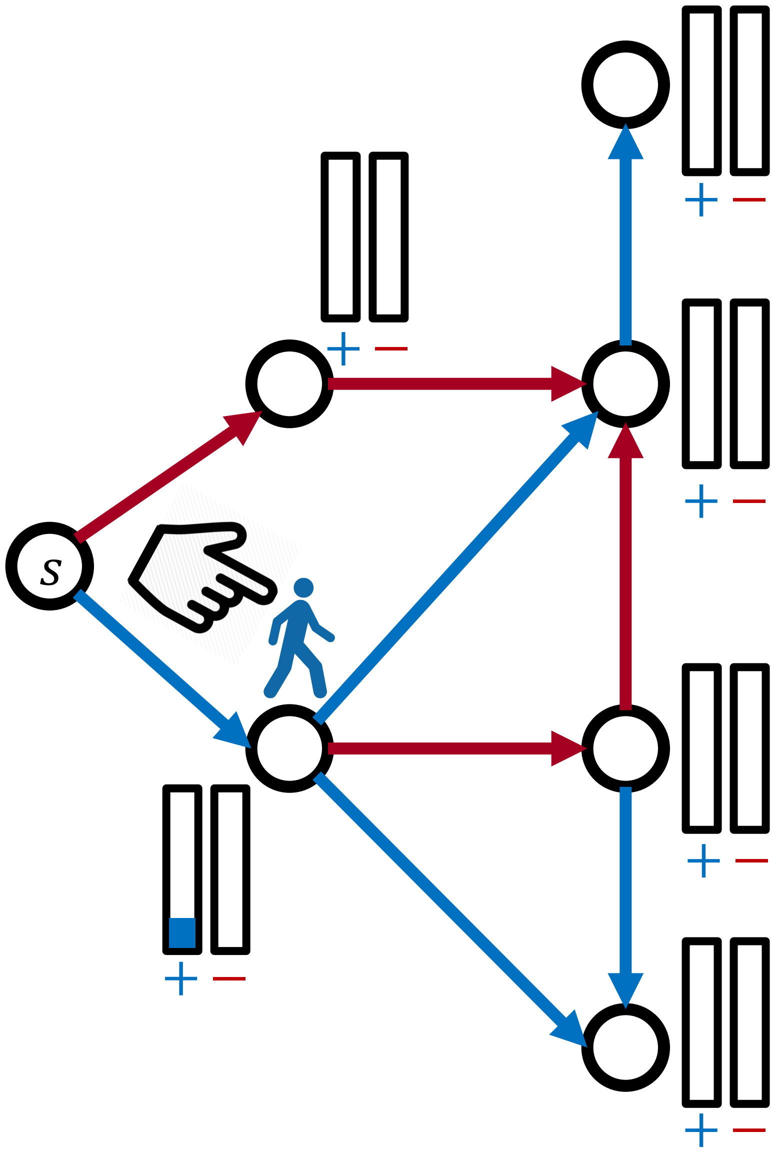
Toss a biased coin

H → Signed random walk

T → Restart

Suppose H appears

Example of SRWR (2)



Do signed random walk

Count it as positive visit

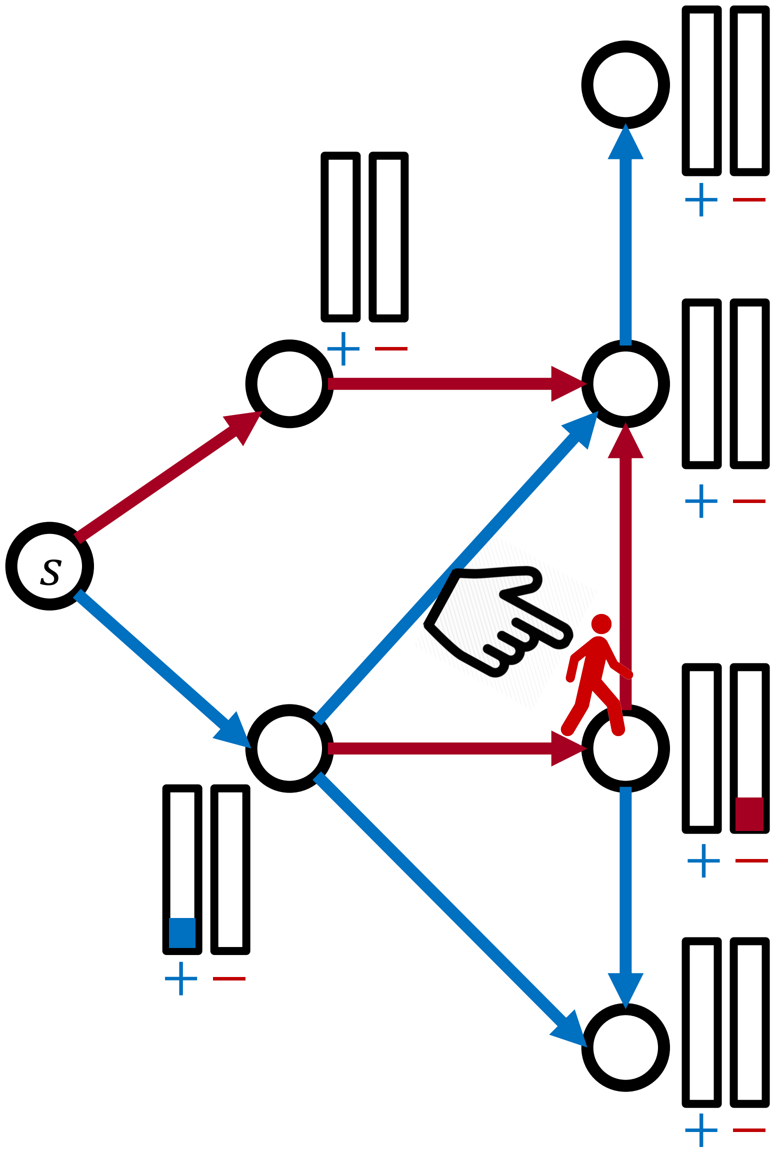
Toss a biased coin again

H → Signed random walk

T → Restart

Suppose H appears

Example of SRWR (3)



Do signed random walk

Flip her sign due to negative edge

Count it as negative visit

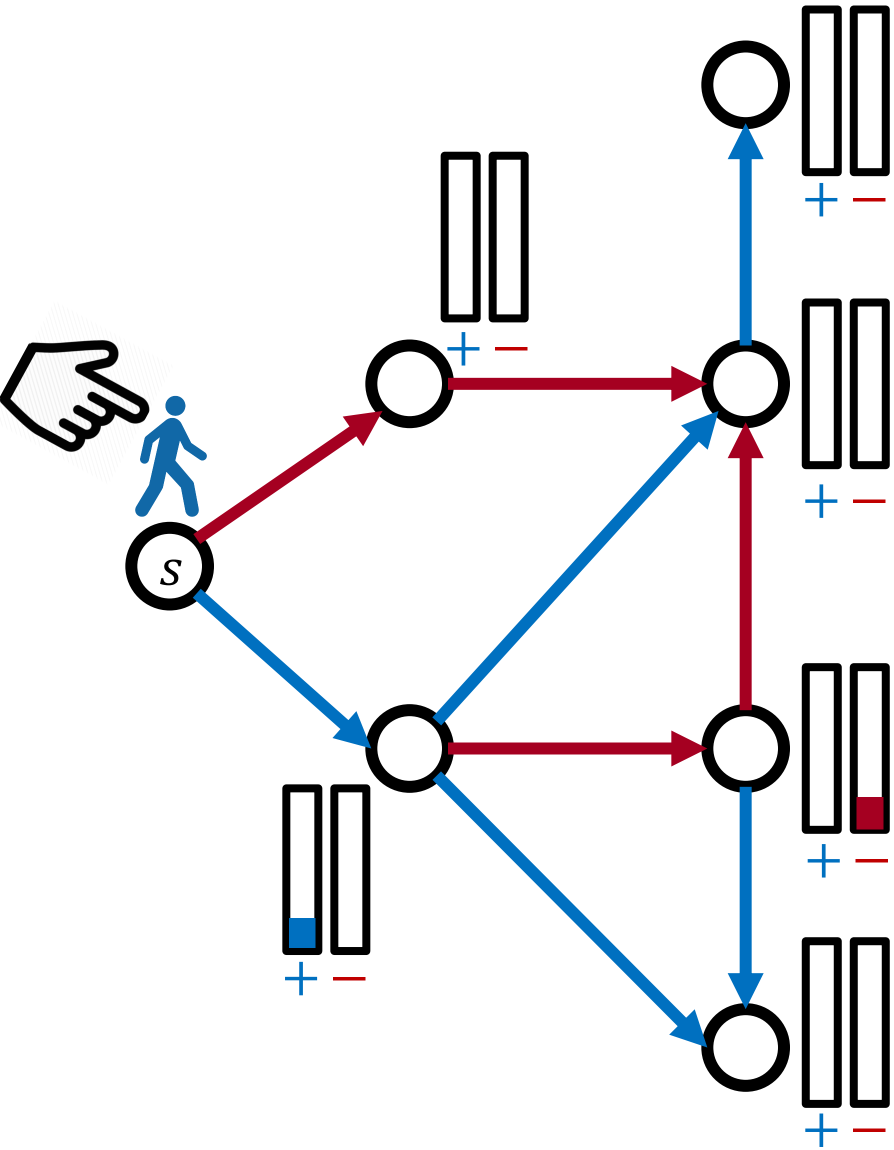
Toss a biased coin again

H → Signed random walk

T → Restart

Suppose T appears

Example of SRWR (4)



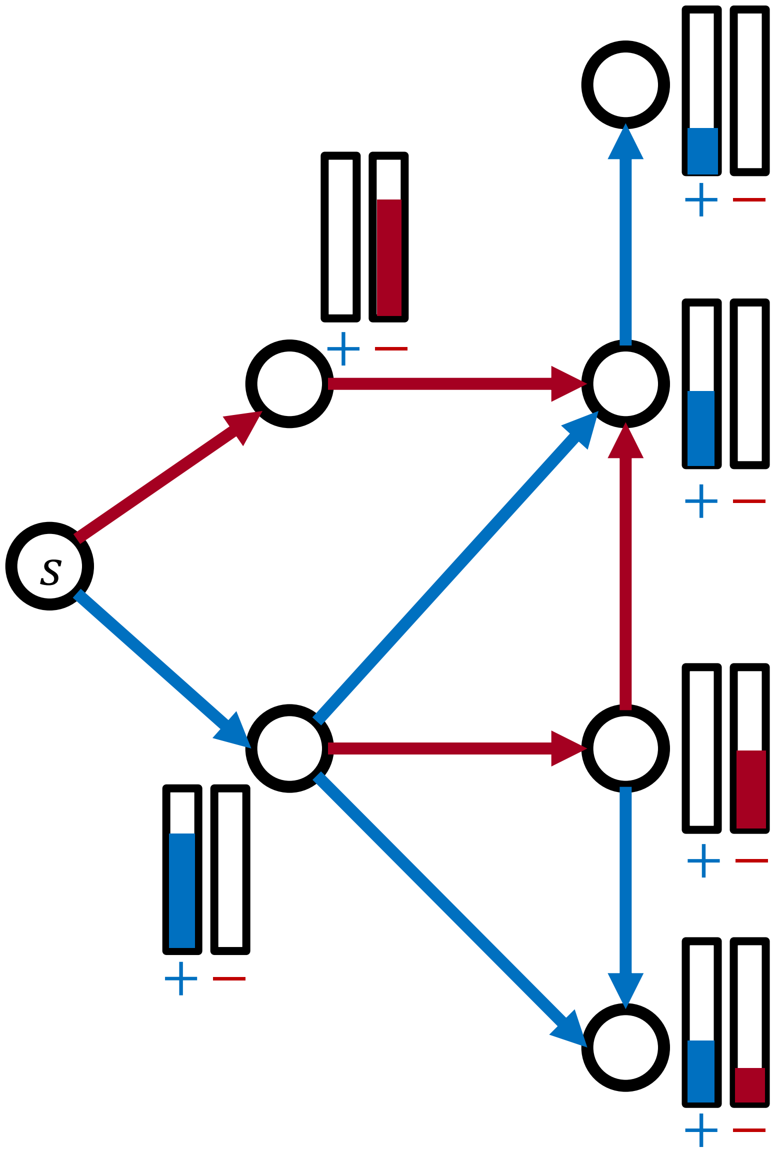
Do restart

Her sign becomes positive

Repeat SRWR

so many times

Example of SRWR (5)



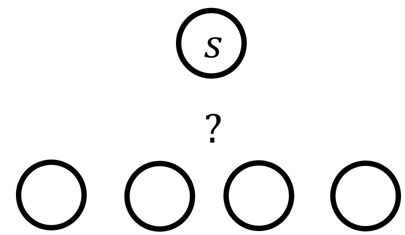
Measure visit probabilities
: = visit count/total # of trials

Probabilities on a node
are used as ranking scores

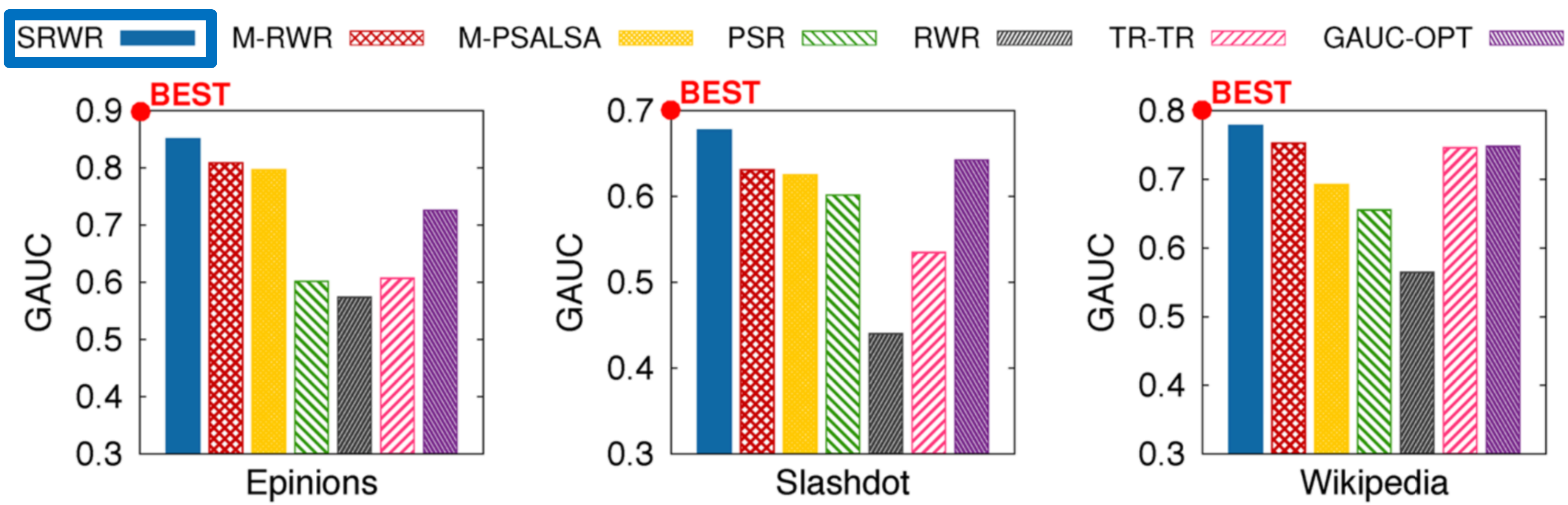
Experimental Results (1)

- **Experimental settings**
  - Data: real-world signed networks
- **Signed Link Prediction**

Which nodes will be connected positively or negatively?



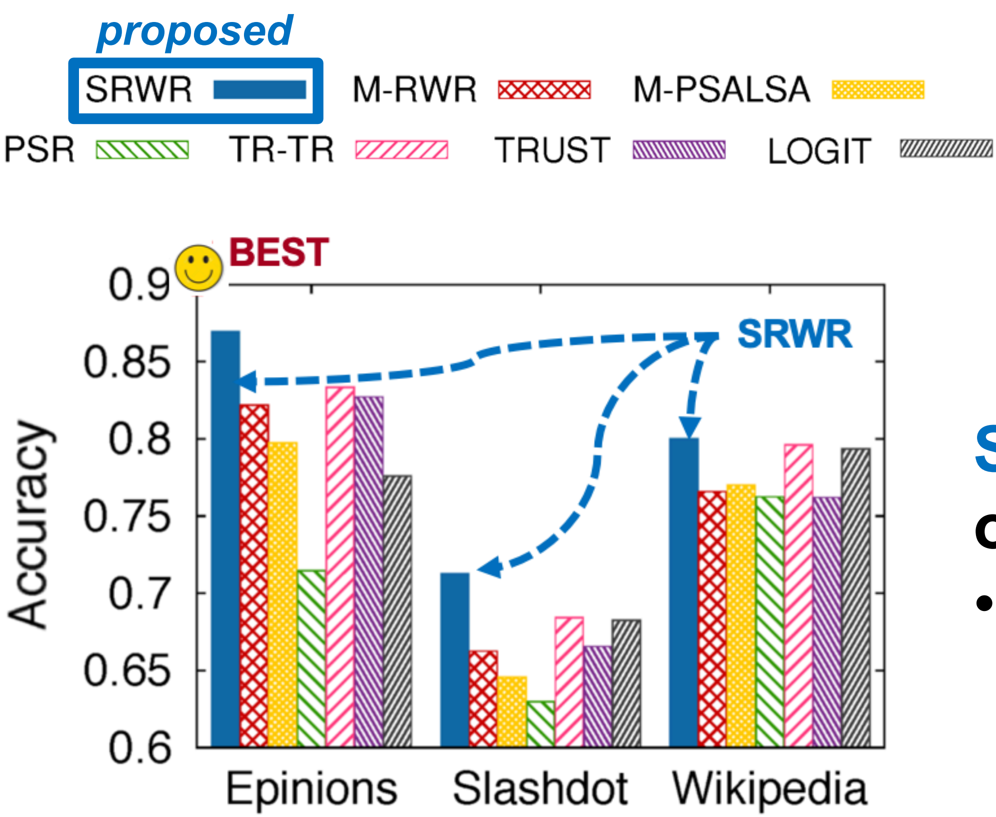
Proposed



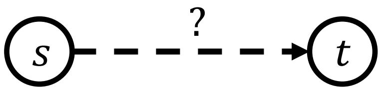
SRWR shows the best link prediction performance for all the datasets

Experimental Results (2)

- **Edge Sign Prediction**



What is the sign of the connection from s to t?



SRWR outperforms other ranking models

- **Achieve best accuracy**

Experimental Results (3)

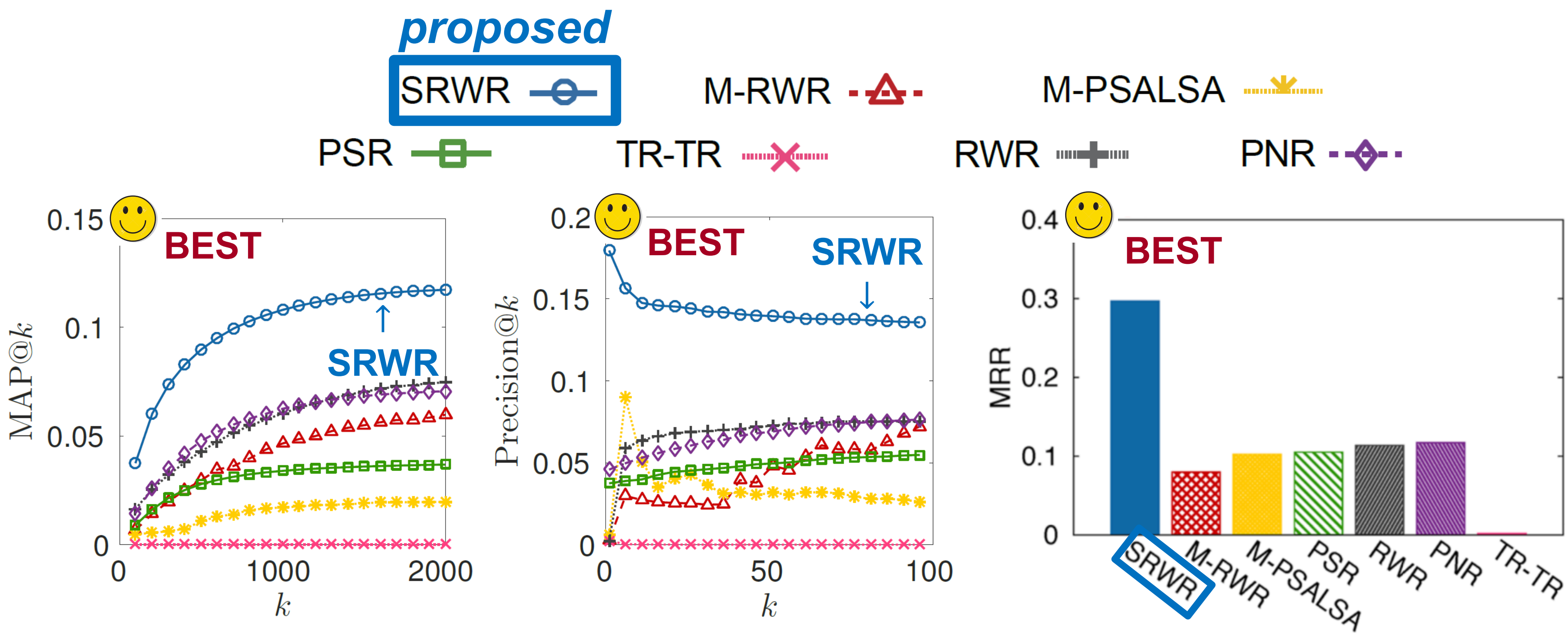
- **Troll Identification in the Slashdot dataset**
  - ***Blue***: *query user (yagu)* & ***Red***: *trolls*

	SRWR (proposed)		M-RWR		M-PSALSA		PSR		TR-TR	
Rank	Trust Ranking	Distrust Ranking	Trust Ranking	Distrust Ranking	Trust Ranking	Distrust Ranking	Trust Ranking	Distrust Ranking	Trust Ranking	Distrust Ranking
1	**yagu***	**Klerck**†	**yagu***	dubba-d	Work+Ac	HanzoSa	**yagu***	SmurfBu	**yagu***	Jack+B.
2	Photon+	**Adolf+H**†	Bruce+P	derago	Unknown	**Jerk+Ci**†	Uruk	Dr.Seus	dexterp	inTheLo
3	Uruk	GISGEOL	CmdrTac	msfodde	afidel	NineNin	Photon+	Doctor_	Jamie+Z	Mactrop
4	stukton	Nimrang	CleverN	cramus	heirony	Rogerbo	clump	artoo	ryanr	DiceMe
5	TTMuskr	Kafka_C	Uruk	lakerdo	bokmann	**SexyKel**†	TTMuskr	Juggle	KshGodd	Einstei
6	clump	Thinkit	Photon+	p414din	ezeri	ScottKi	stukton	FreakyG	TheIndi	FinchWo
7	Bruce+P	**CmderTa**†	stukton	an+unor	As+Seen	qurob	RxScram	RunFatB	daoine	Penus+T
8	RxScram	SteakNS	clump	exfuga	KillerD	bendodg	charlie	jmpoast	Berylli	r_glen
9	CmdrTac	JonKatz	TTMuskr	kryptok	potaz	ArnoldY	ssbg	El_Muer	danhara	Roland+
10	aphor	Henry+V	RxScram	toomz	byolinu	jcr	Idarubi	Ghost+H	Degrees	sting3r

- The **query user** is ranked 1st in our trust ranking
- **Many trolls** are ranked high in our distrust ranking

Experimental Results (4)

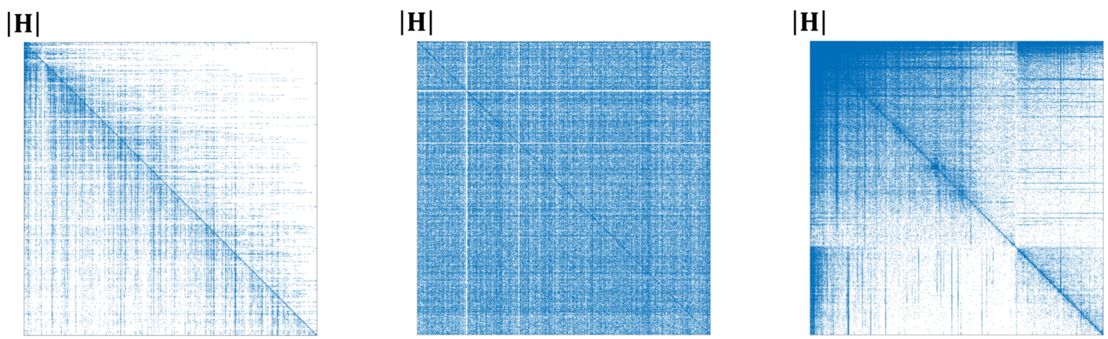
- **Troll Identification in the Slashdot dataset**



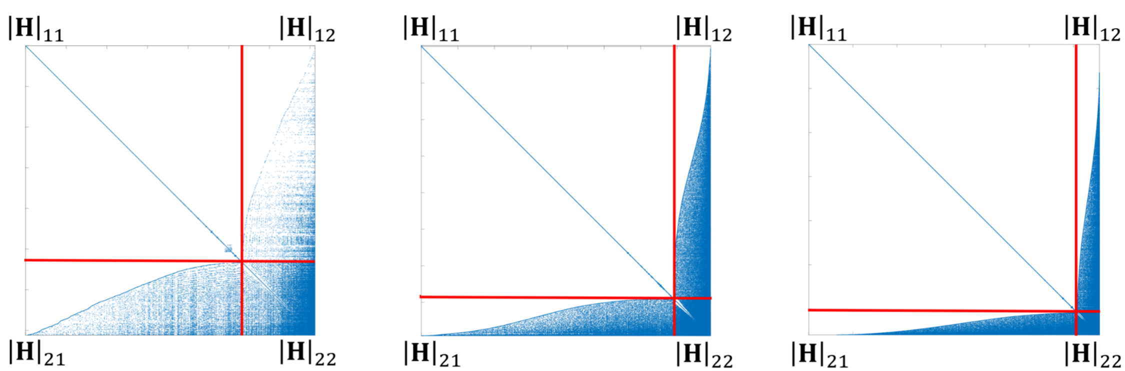
SRWR captures trolls better than other ranking models!

Proposed Method: SRWR (6)

- **Idea 3) Exploit real-world graph structures for SRWR-Pre (Prep. Method for SRWR)**



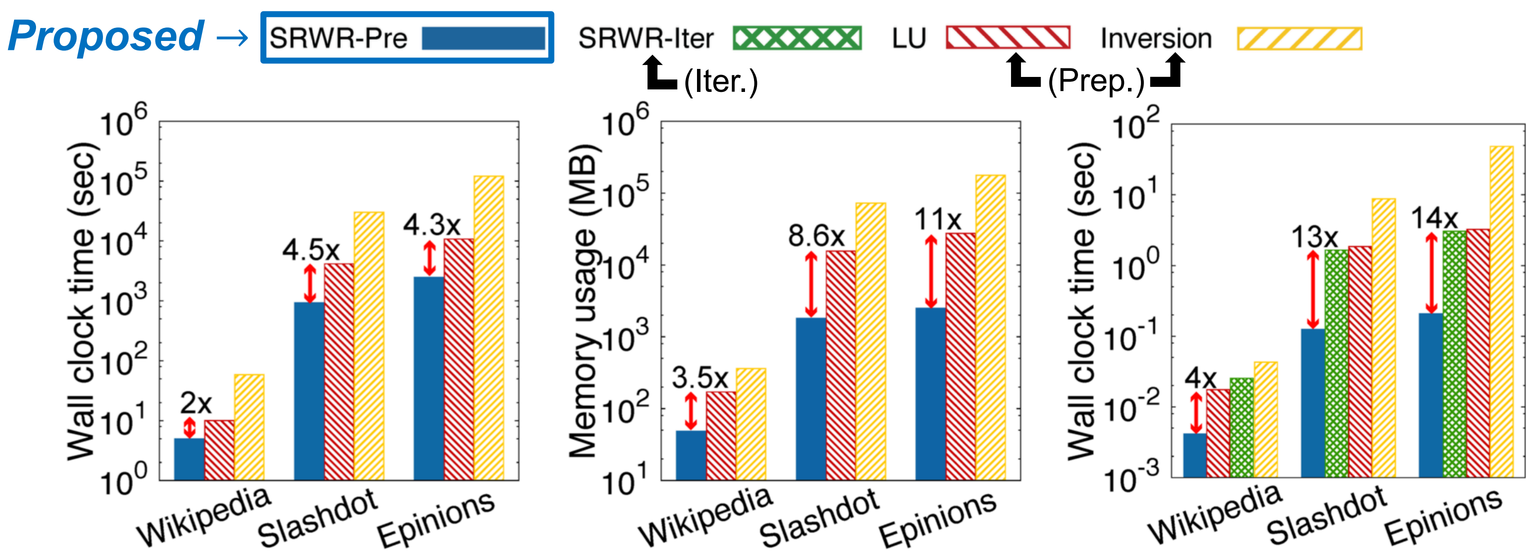
(a) Original matrix $|\mathbf{H}|$ in the Wikipedia dataset (b) Original matrix $|\mathbf{H}|$ in the Slashdot dataset (c) Original matrix $|\mathbf{H}|$ in the Epinions dataset



(d) Reordered matrix $|\mathbf{H}|$ in the Wikipedia dataset (e) Reordered matrix $|\mathbf{H}|$ in the Slashdot dataset (f) Reordered matrix $|\mathbf{H}|$ in the Epinions dataset

Experimental Results

- **Experimental settings**
  - Machine: single machine with 500GB memory
  - Data: real-world signed networks



(a) Preprocessing time

(b) Memory space for preprocessed data

(c) Query time

SRWR-Pre requires 11× less memory space & computes SRWR 14× faster!

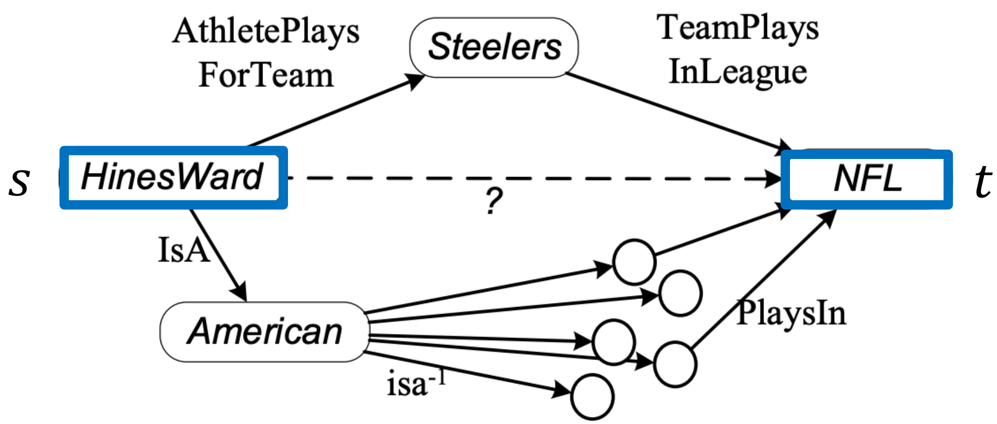
Proposed Methods

- **Random Walk-based Large Graph Mining Exploiting Real-world Graph Properties**

Current Works *(Ph.D. Course)*		
Plain Graphs (No edge labels)	**Signed Graphs (Two edge labels)**	**Edge-labeled Graphs (K edge labels)**
Fast Scalable & Exact RWR in Billion-scale Graphs	Random Walk in Signed Graphs: Personalized Ranking	Random Walk in Edge-labeled Graphs: Relational Reasoning
BePI	**SRWR**	**MuRWR**
[SIGMOD'17]	[ICDM'16] [KAIS'19]	[WWWJ'20]

Introduction

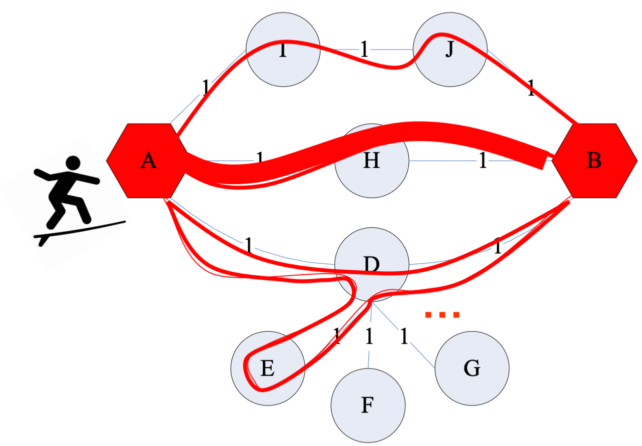
- **Problem: Relational Reasoning in Edge-labeled Graphs**
  - **Input:** Edge-labeled graph G (each edge has one of K categorical labels) & Two nodes s and t
  - **Output:** K relevance scores on t w.r.t s



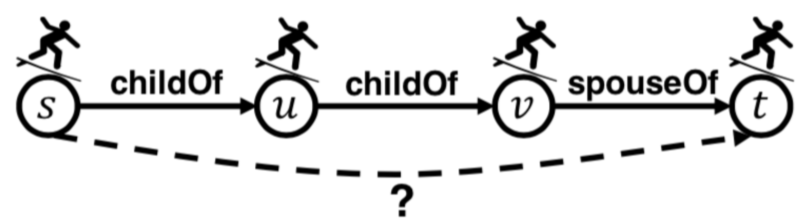
  - **Importance:** increase KB's quality via knowledge completion ⇒ helpful for applications based on KB

Limitation & Challenge

- **RWR can capture diverse relationship between two nodes**
  - Multiple connections considering quality
    - Multi-hops/degree/weight…
  - **But it cannot consider edge labels!**
- **How to reflect such labels into random walk?**



Trajectory of the random surfer



The surfer in RWR cannot identify the relation between the nodes!

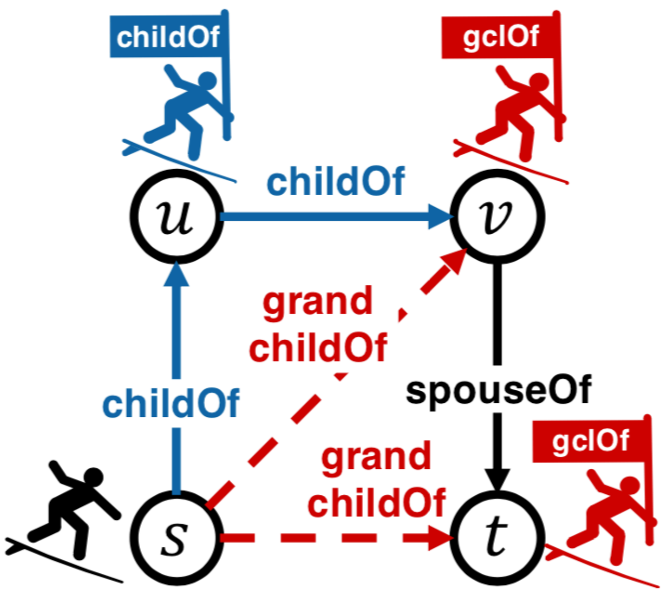
Proposed Method: MuRWR (1)

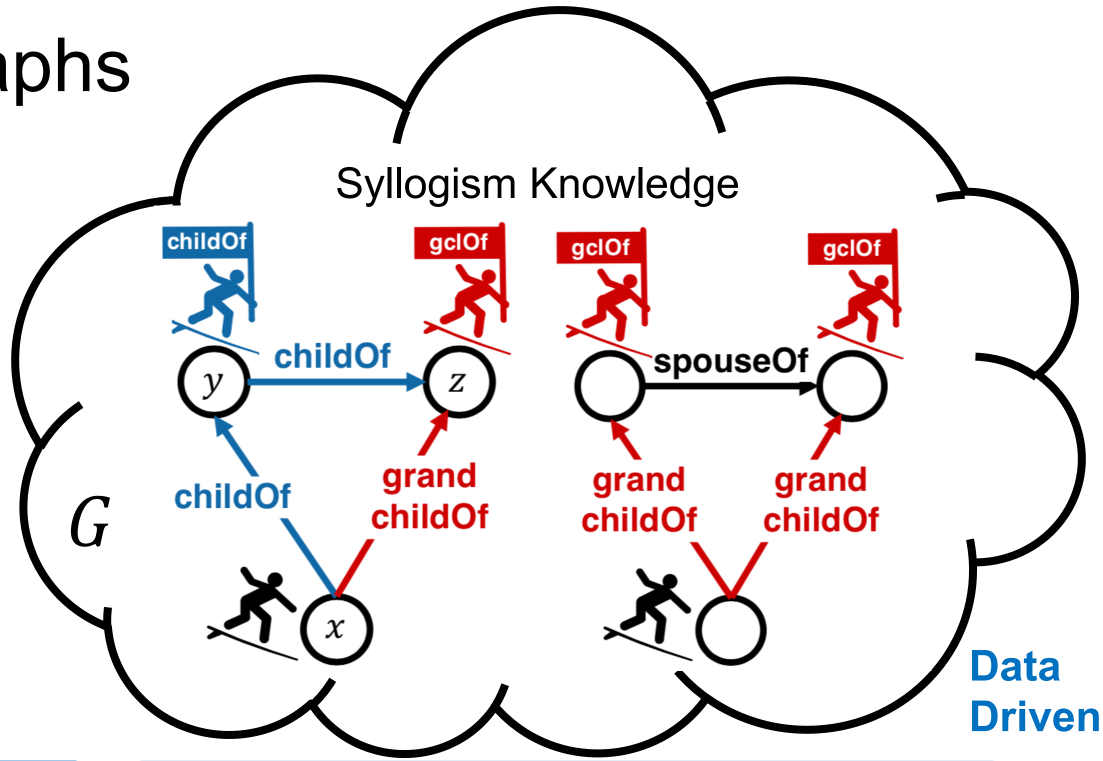
- **MuRWR (Multi-Labeled Random Walk with Restart)**
  - Random walk-based model for relevance scores in edge-labeled graphs
- **Key Ideas**
  - **Idea 1)** Introduce a **labeled random surfer**
    - Whose label at a node indicates the inferred relation
  - **Idea 2)** Allow **the surfer to change her label** during random walk with some rules
  - **Idea 3)** Exploit a **data-driven approach** to extract knowledge from a graph so that the surfer learns the rules
  - To sum up, **MuRWR is the generalization of SRWR!**

K labels on edges **2 labels on edges**

Proposed Method: MuRWR (2)

- **MuRWR (Multi-Labeled Random Walk with Restart)**
  - Random walk-based model for relevance scores in edge-labeled graphs





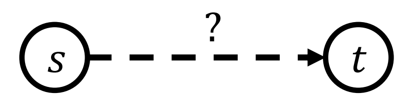
How the labeled surfer walks along the path from s to t

Labeled triangles used for the rules on how to change the surfers' label

Experimental Results

- **Experimental settings**
  - Data: real-world edge-labeled graphs
  - Applications: relational reasoning

What is the relation of
the connection from s to t?



Methods	$K = 2$		Accuracy			$K = 18$
	WikiVote	Slashdot	Epinions	Advogato	WN11	WN18
Random	0.497	0.500	0.493	0.340	0.090	0.078
LINE [34]	0.781	0.771	0.903	0.552	0.489	0.404
node2vec [8]	0.779	0.765	0.905	0.586	0.426	0.401
MRWR [30]	0.805	0.769	0.890	0.550	0.194	0.342
SRWR [11]	0.825	0.790	0.906	-	-	-
PRA [15]	0.813	0.804	0.913	0.683	0.580	0.556
TransE [4]	0.793	0.802	0.902	0.644	0.617	0.653
TransR [22]	0.800	0.757	0.874	0.672	0.609	0.530
Proposed → MuRWR†	**0.830**	**0.820**	**0.929**	**0.727**	**0.641**	**0.689**
Improvement	1%	2%	2%	6%	4%	5%

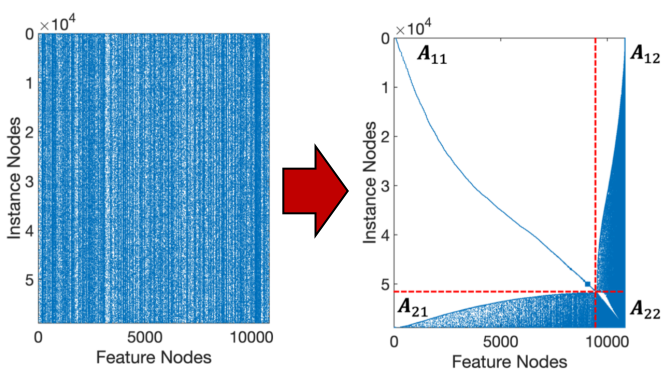
MuRWR shows the best accuracy among all tested methods!

Outline

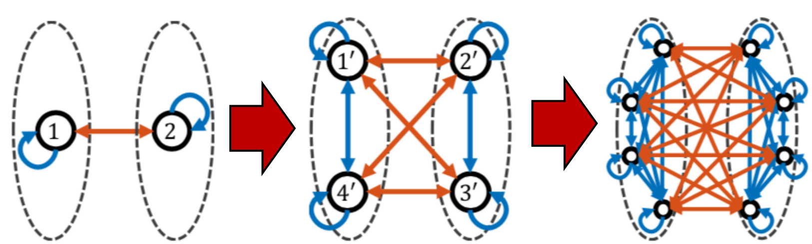
- Overview
- Proposed Methods
- **Future Works**
- Conclusion

Future Works

- **Further extend our approach exploiting distinct properties in real-world data**
  - **1)** To develop a method for fast & accurate **SVD based pseudoinverse computation**
  - **2)** To design a method for fast & scalable **signed network generation** following real-world properties
  - **3)** To make our methods working on **graph databases** or **distributed systems**



F1) Reordering for Rectangular Matrix



F2) Simulation of Balanced Structure

F3) Graph DB & Distributed processing

Outline

- Overview
- Proposed Methods
- Future Works
- **Conclusion**

Conclusion

- **Random Walk-based Large Graph Mining Exploiting Real-world Graph Properties**
  - **G1.** To devise **fast**, **scalable**, and **exact** methods for **random walk** in large-scale graphs
  - **G2.** To design effective **random walk** models utilizing **label** data in labeled graphs
- **Approach:** to exploit real-world graph properties

Current Works *(Ph.D. Course)*		
Plain Graphs (No edge labels)	**Signed Graphs (Two edge labels)**	**Edge-labeled Graphs (K edge labels)**
Fast Scalable & Exact RWR in Billion-scale Graphs **BePI** [SIGMOD'17]	Random Walk in Signed Graphs: Personalized Ranking **SRWR** [ICDM'16] & [KAIS'19]	Random Walk in Edge-labeled Graphs: Relational Reasoning **MuRWR** [WWWJ'20]
Deadend Structure Hub-and-Spoke Structure	Signed Triangle Patterns Hub-and-Spoke Structure	Labeled Triangle Patterns (Syllogism Knowledge)

Thank You!

Q & A