BalanSiNG: Fast and Scalable Generation of Realistic Signed Networks

EDBT/ICDT 2020

Jinhong Jung
Seoul National University
Seoul, South Korea

Ha-Myung Park
Kookmin University
Seoul, South Korea

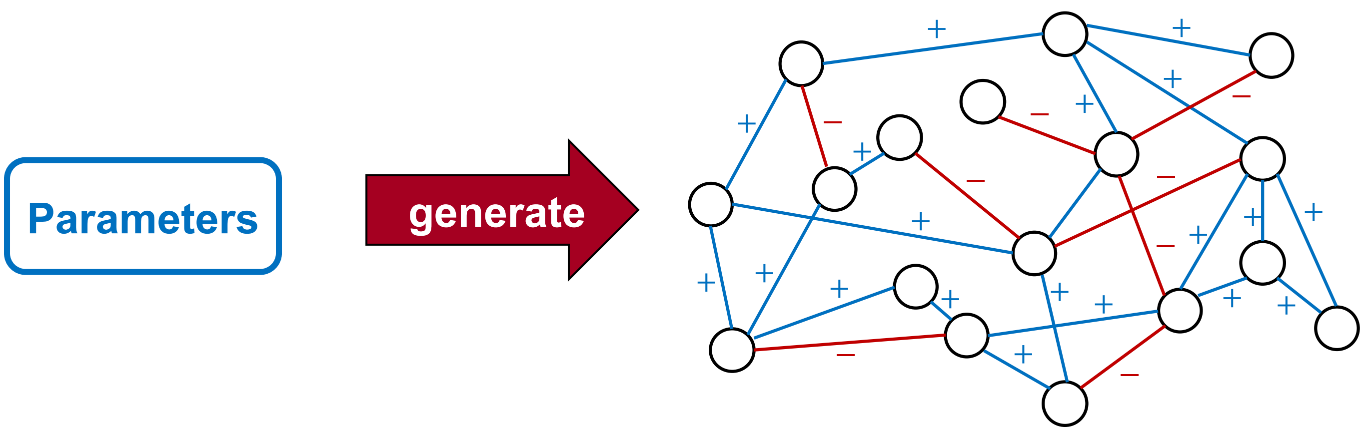
U Kang
Seoul National University
Seoul, South Korea

Outline

- **Introduction**
- Proposed Method
- Experiments
- Conclusion

Research Question

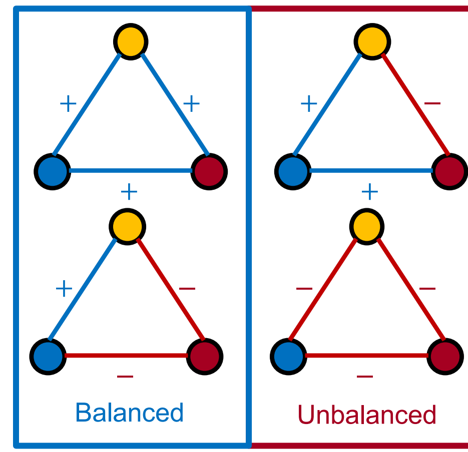
- **How can we efficiently generate realistic signed networks?**
  - Graphs with signed edges (+⇒trust, −⇒distrust)
  - Online social services (e.g., *Epinions/Slashdot*)
  - Sign prediction, ranking, anomaly detection, etc.

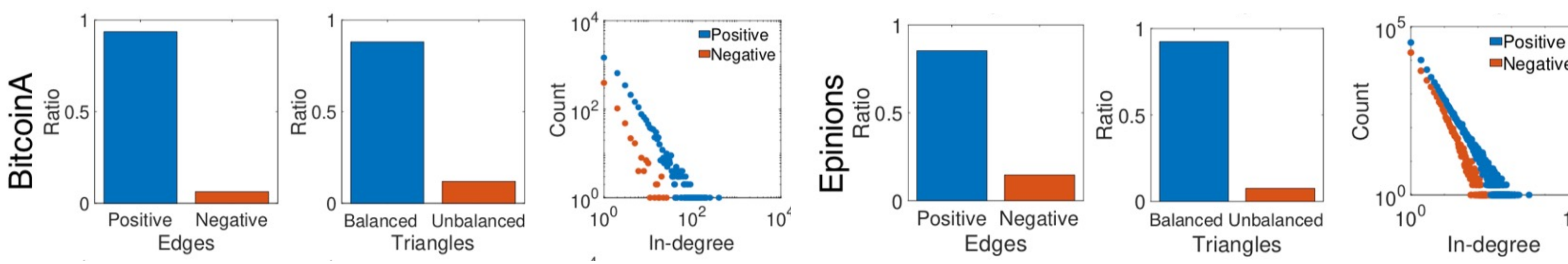


Given some parameters, generate synthetic signed networks following various realistic properties

Desired Properties

- **Real-world signed networks share common tendencies on various properties**
  - Properties w.r.t. edge sign
    - **D1)** Positively skewed sign ratio
    - **D2)** Highly balanced triangle ratio
    - **D3)** Power-law degree distribution for only +/– edges





D1 **D2** **D3** **D1** **D2** **D3**

- *See the paper for other properties!*

Problem Definition & Importance

- **Signed network generation problem**
  - Given n and m (target # of nodes & edges, resp.)
  - To synthetically generate a signed network
    - Having $n = 2^L$ nodes and m signed edges
    - Showing the desired properties of real-world signed net.
- **Why important?**
  - To understand the formation of real-world networks
    - Historically profound in network science
  - Extremely useful for researches on signed networks
    - Scalability evaluation, network simulation, anonymization

Related Work & Challenge

- **Models for unsigned networks**
  - Stochastic Kronecker Graph (SKG)
    - Simulate a self-similarity using Kronecker product for modeling realistic properties of unsigned networks
    - Scalable, but no consideration on forming signed edges!
- **Models for signed networks**
  - Balanced Signed Chung-Lu (BSCL)
    - Extended version of Chung-Lu model considering the formation of balanced triangles
    - Realistic, but not scalable for generating large networks! $O(d_{max}^2 m + n)$

How to efficiently generate large-scale & realistic signed networks?

Outline

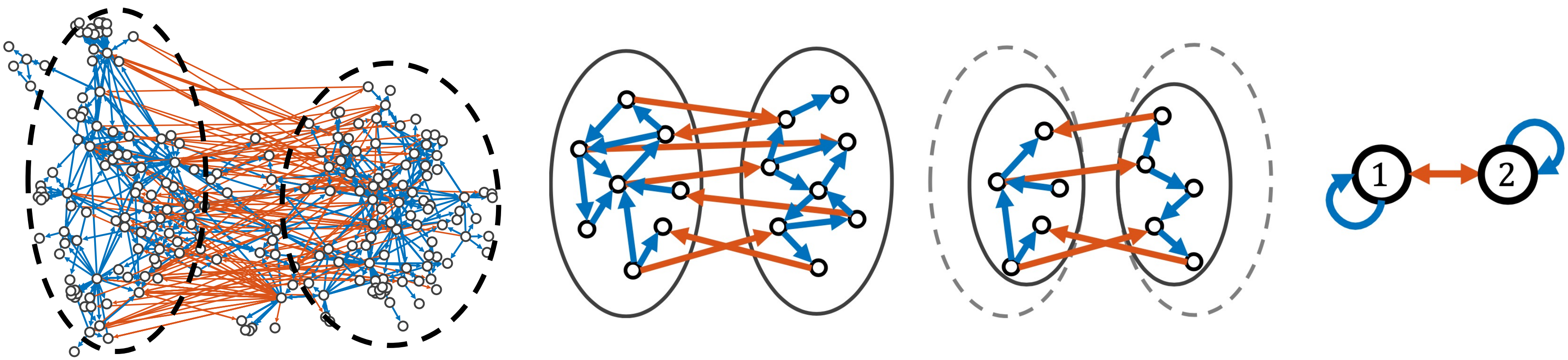
- Introduction
- **Proposed Method**
- Experiments
- Conclusion

Proposed Method

- **BalanSiNG** **(Balanced Signed Network Generator)**
  - Novel method for generating realistic signed networks showing the desired properties
  - **Main Approaches**
    - 1) ***Self-similar Balanced Structure for Signed Net.***
    - 2) ***Basic Stochastic Kronecker Signed Graph*** ***(SKSG-B)***
      - For simulating self-similar balanced structure
    - 3) ***Stochastic Kronecker Signed Graph*** ***(SKSG)***
      - For more realistic signed networks
    - 4) ***BalanSiNG***, an efficient and parallel method
      - While supporting SKSG

Self-similar Balanced Structure

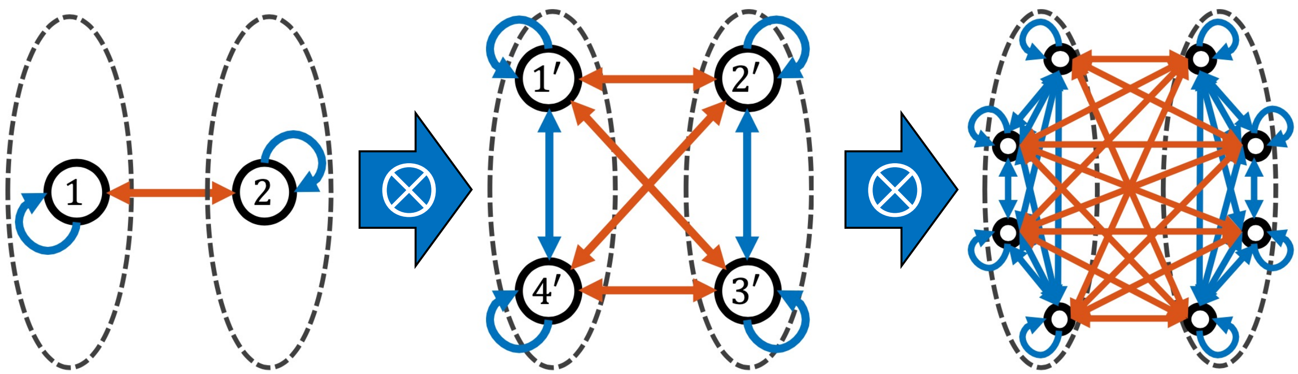
- **Real-world signed networks have self-similar balanced structure!**
  - A self-similar object is (approx.) similar to a part of itself
  - Two clusters in signed network ⇒ *balanced structure*
  - Internally, two smaller clusters appear ⇒ *self-similar*



Real-world signed network (Congress dataset) | **Global balanced structure** | **Zoomed-in balanced structure** | **Self-similar balanced structure**

SKSG-B Model (1)

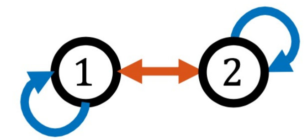
- **Simulate the self-similarity with Kronecker product $\otimes$**
  - Given the initial graph, double itself using $\otimes$ at each iteration $\Rightarrow$ $2^l \times 2^l$ adjacency matrix



  - Kronecker product: $\underset{2\times 2}{\boldsymbol{A}} \otimes \underset{2\times 2}{\boldsymbol{B}} = \underset{2^2\times 2^2}{\begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} \end{bmatrix}}$

SKSG-B Model (2)

- **How to simulate the self-similarity?**
  - 1) Represent the self-similarity pattern



$$\mathbf{T}^{(1)} = \{+\mathcal{P}, -\mathcal{M}\} = \left\{+\begin{bmatrix} p_{11} & 0 \\ 0 & p_{22} \end{bmatrix}, -\begin{bmatrix} 0 & m_{12} \\ m_{21} & 0 \end{bmatrix}\right\}$$

    - $p_{11} = P(1, 1, +)$: prob. that edge (1, 1) is positive
  - 2) Apply Kronecker product

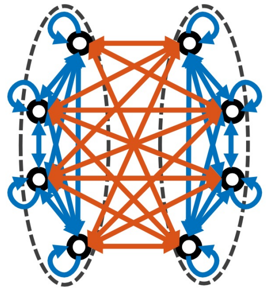
$$\mathbf{T}^{(2)} = \mathbf{T}^{(1)} \otimes \mathbf{T}^{(1)} = \{+\mathcal{P} \otimes \mathcal{P}, -\mathcal{P} \otimes \mathcal{M}, -\mathcal{M} \otimes \mathcal{P}, +\mathcal{M} \otimes \mathcal{M}\}$$

  - 3) Aggregate them according to their sign

$$\widetilde{\mathbf{T}}^{(2)} = f_b\left(\mathbf{T}^{(1)} \otimes \mathbf{T}^{(1)}\right) \leftarrow \text{Balanced sign aggregator}$$

$$= \{+(\mathcal{P} \otimes \mathcal{P} + \mathcal{M} \otimes \mathcal{M}), -(\mathcal{P} \otimes \mathcal{M} + \mathcal{M} \otimes \mathcal{P})\}$$

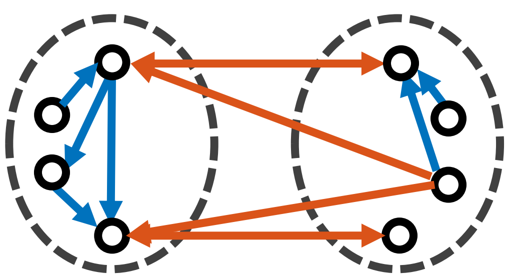
SKSG-B Model (3)


(c) $\tilde{\mathbf{T}}^{(3)}$

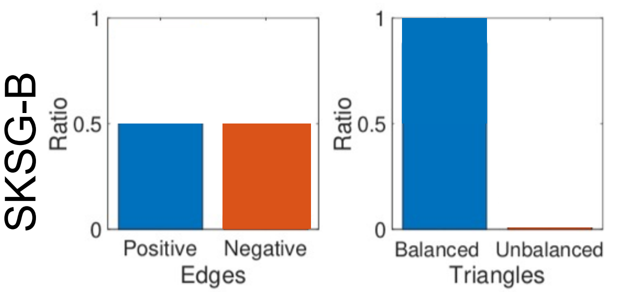
- **How to simulate the self-similarity?**
  - 4) Repeat steps 2 and 3: $\tilde{\mathbf{T}}^{(l)} = f_b\left(\mathbf{T}^{(1)} \otimes \tilde{\mathbf{T}}^{(l-1)}\right)$
    - $\tilde{\mathbf{T}}^{(l)} = \left\{+\boldsymbol{\mathcal{P}}^{(l)}, -\boldsymbol{\mathcal{M}}^{(l)}\right\}$ has two stochastic matrices in $2^l \times 2^l$
      - $\boldsymbol{\mathcal{P}}_{uv}^{(l)} = P(u, v, +)$ and $\boldsymbol{\mathcal{M}}_{\boldsymbol{uv}}^{(l)} = P(u, v, -)$
  - How to build a signed network from $\tilde{\mathbf{T}}^{(l)}$?
    - For each $(u, v) \in V$, randomly determine the edge
      - Edge: toss a coin with $P(u, v) = P(u, v, +) + P(u, v, -)$
      - Sign: positive if $P(+|u, v) > P(-|u, v)$; otherwise, negative
        - $P(+|u, v) = P(u, v, +)/P(u, v)$
  - **Lemma**: two clusters are preserved as l increases!
    - *See the paper for the proof for the lemma*

SKSG-B Model (4)

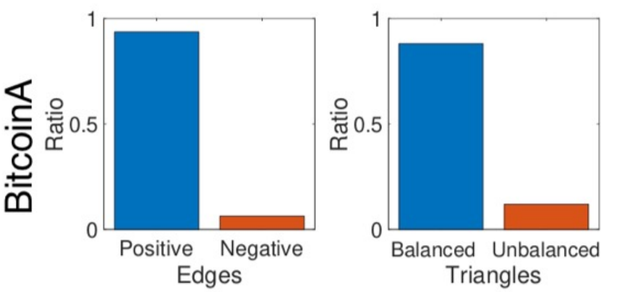
- **SKSG-B returns fully balanced signed net.!**
  - 1) + edges in each group & – edges b.t.w. groups
  - 2) Δ_{+++} in each group & Δ_{+--} between groups
  - **Problems of SKSG-B**
    - **P1)** Uniform sign ratio ⇒ *much more positive edges*
    - **P2)** Only balanced Δ ⇒ *a few unbalanced Δ (e.g., Δ_{++-})*



Example signed netw
ork of SKSG-B



Properties of signed
networks of SKSG-B



Properties of
real-world signed networks

SKSG Model (1)

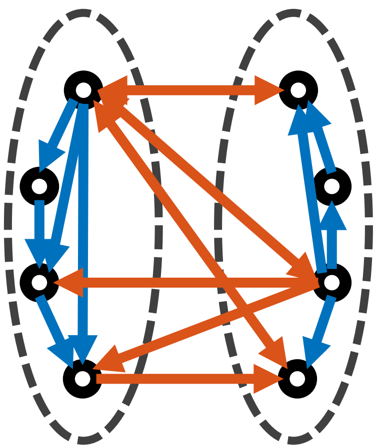
- **Main ideas of SKSG**
  - **Weight splitting:** to increase probabilities on generating positive signs with $0 \leq \alpha \leq 1$

$$f_\alpha(\mathbf{T}) = f_\alpha(\{+\boldsymbol{\mathcal{P}}, -\boldsymbol{\mathcal{M}}\}) = \{+(\boldsymbol{\mathcal{P}} + \alpha\boldsymbol{\mathcal{M}}), -(1-\alpha)\boldsymbol{\mathcal{M}}\}$$

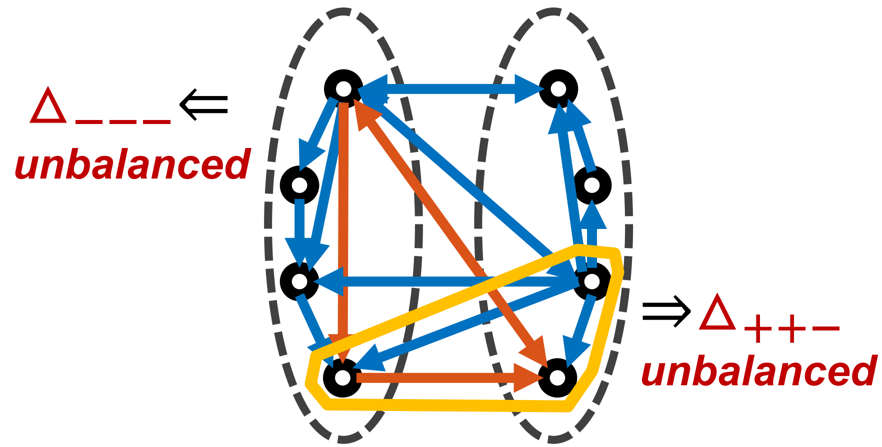
  - **Stochastic sign determination:** to make the chance of forming negative edges inside each group
    - SKSG-B: + if $P(+|u,v) > P(-|u,v)$; otherwise, −
    - SKSG: toss a coin with $P(+|u,v)$; $\text{Head} \rightarrow +$ or $\text{Tail} \rightarrow -$

SKSG Model (2)

- **Effects of SKSG**
  - 1) Increase # of positive edges
    - Some − edges in SKSG-B become + ones in SKSG
      - Higher $\alpha \Rightarrow$ larger # of positive edges
  - 2) Make the chance of forming unbalanced triangles



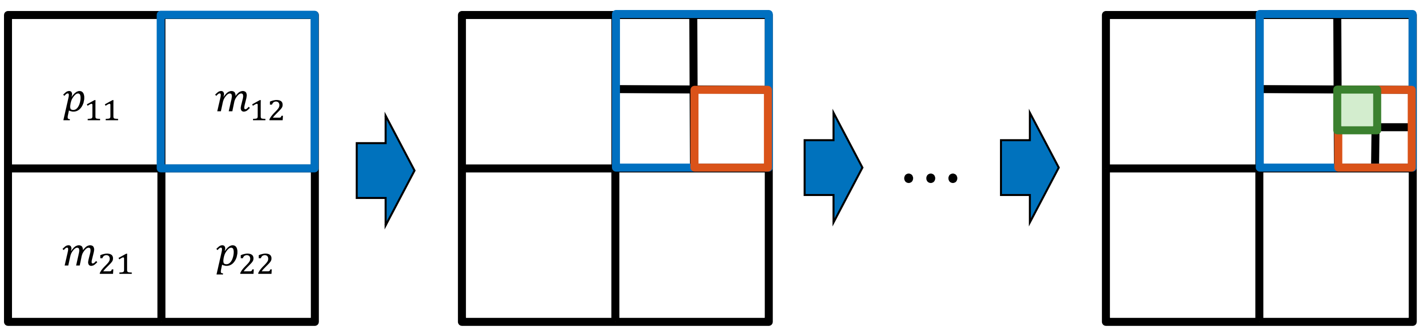
SKSG-B



SKSG

BalanSiNG

- **Efficient & scalable method following SKSG**
  - SKSG requires $O(n^2)$ time for constructing $\widetilde{\mathbf{T}}^{(l)}$
  - Instead of building $\widetilde{\mathbf{T}}^{(l)}$ explicitly, directly determine an edge $P(u, v)$ and track its sign probabilities $P(u, v, \pm)$
  - **Main intuition**: recursively select regions of adj. mat.



- Each edge is determined **in $O(L) = O(\log n)$** (if $n = 2^L$)
- Each edge can be determined **in parallel**!
- *See the paper for details and proofs!*

Total: $O(m \log n)$

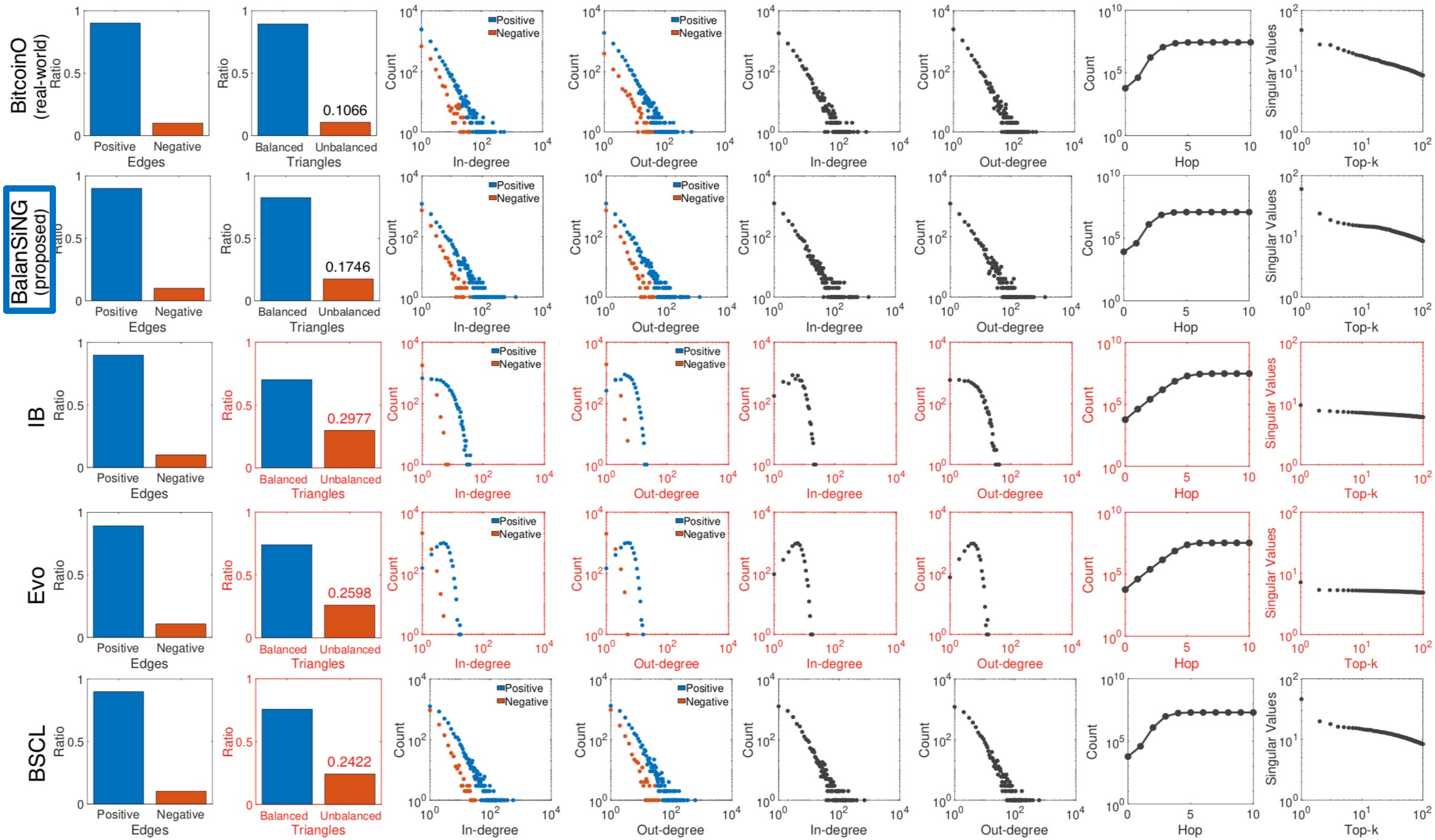
Outline

- Introduction
- Proposed Method
- ➡ **Experiments**
- Conclusion

Experimental Setting

- **Main experimental questions**
  - **Q1.** Is BalanSiNG able to generate signed networks showing the **desired properties**?
  - **Q2.** How **efficient** is BalanSiNG for generating large-scale signed networks?
- **Datasets:** BitcoinA, BitcoinO, Epinions
- **Competitors:** IB, EBO, BSCL
- **Implementation of BalanSiNG**
  - Single machine: c++
  - Distributed machines: Apache Spark (17 machines)

Q1. Desired Properties

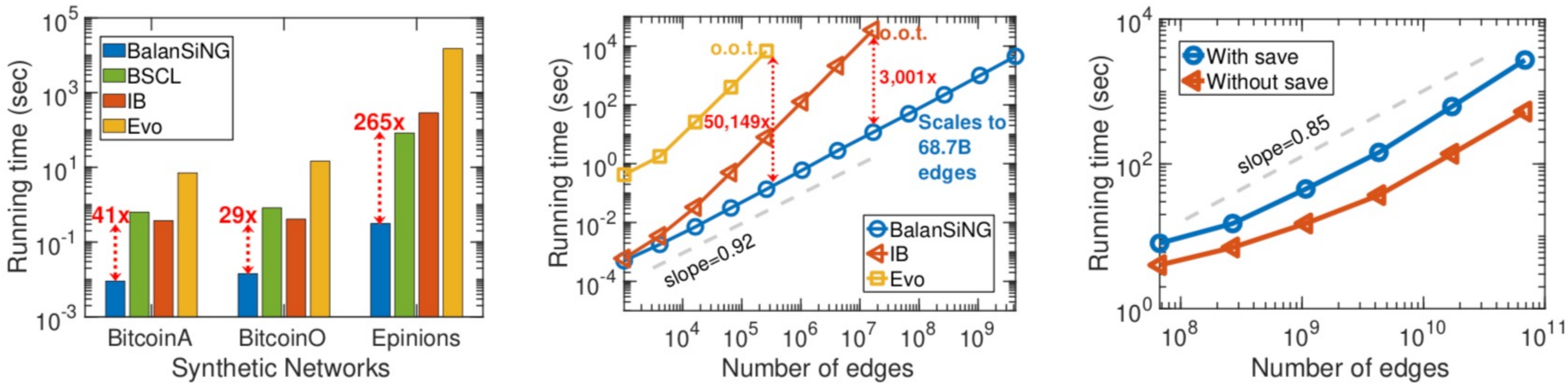


(a) Global sign distribution (b) Balanced triangle distribution (c) Signed in-degree distribution (d) Signed out-degree distribution (e) In-degree distribution (f) Out-degree distribution (g) Hop plot (h) Singular value distribution

BalanSiNG outputs the most similar network to the real-world network!

Q2. Computational Efficiency

- **On both single and distributed machines**
  - 1) up to 265x faster for imitating input signed network on single machine
  - 2) near-linear scalability w.r.t. # of edges on both single and distributed machines (scale to 68.7B edges)



(a) Generation time on a single machine (b) Data scalability on a single machine (c) Data scalability on distributed machines

BalanSiNG is efficient and scalable for generating signed networks!

Outline

- Introduction
- Proposed Method
- Experiments
- **Conclusion**

Conclusion

- **BalanSiNG** **(Balanced Signed Network Generator)**
  - Efficient and parallel method for generating realistic signed networks
  - Simulating self-similar balanced structure in real-world signed networks
- **Main Results**
  - Generate **the most realistic signed networks**
  - Up to **265x faster** for imitating input signed network
  - **Near-linear scalability** w.r.t. # of edges on both single and distributed machines
    - Successfully scale to **68.7B edges**!

Thank You!

Q & A

Codes & datasets

https://datalab.snu.ac.kr/balansing