Research Summary

Xinyu Wu

xinyuwu1@mit.edu

Laboratory for Information and Decision Systems,
Massachusetts Institute of Technology

Contents

M.S./Ph.D. Stage

- Finite-Buffer Communication Networks Analysis
- Power System Failure Cascade Analysis

B.S. Stage

- Social Network De-anonymization
- Wireless Fingerprints Prediction for Positioning

Contents

M.S./Ph.D. Stage

- Finite-Buffer Communication Networks Analysis
- Power System Failure Cascade Analysis

B.S. Stage

- Social Network De-anonymization
- Wireless Fingerprints Prediction for Positioning

Contents

M.S./Ph.D. Stage

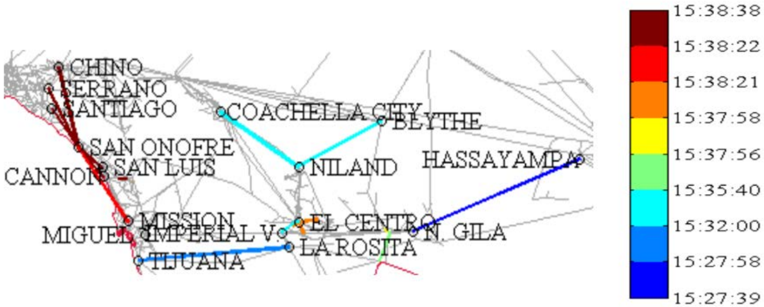
- Finite-Buffer Communication Networks Analysis
- Power System Failure Cascade Analysis

B.S. Stage

- Social Network De-anonymization
- Wireless Fingerprints Prediction for Positioning

Motivation

Large power blackouts



San Diego Power Blackout on Sept. 8, 2011[1]

A 500 kV line tripped off by a mistake.
Wrongly cut off 2 generators in Mexico.
Finally separated into three islands which collapsed afterwards.

Manhatten City Blackout on Jul. 13, 2019[2]

A transformer fired at West 64th Street and West End Avenue

- Starting from extrinsic effects (weather, man-made error, etc.) on few nodes or links.
- After some point, the blackout propagates broadly within minutes.
- The failure of a component may affect another remote component.

[1] G.Zussman et.al., "Power Grid Vulnerability to Geographically Correlated Failures: Analysis and Control Implications", IEEE INFOCOM 2014
[2] https://en.wikipedia.org/wiki/Manhattan_blackout_of_July_2019

Motivation

Question: Given the initially failed components, can we predict the cascade process?

- **How many** components will fail?
- **Which** component will fail?
- If it will fail, **when** will it fail once initial failure occurs?
- ...

To solve the problem
- a model is needed to capture such cascade process
- historical cascade records may help us to refine the model

Contributions

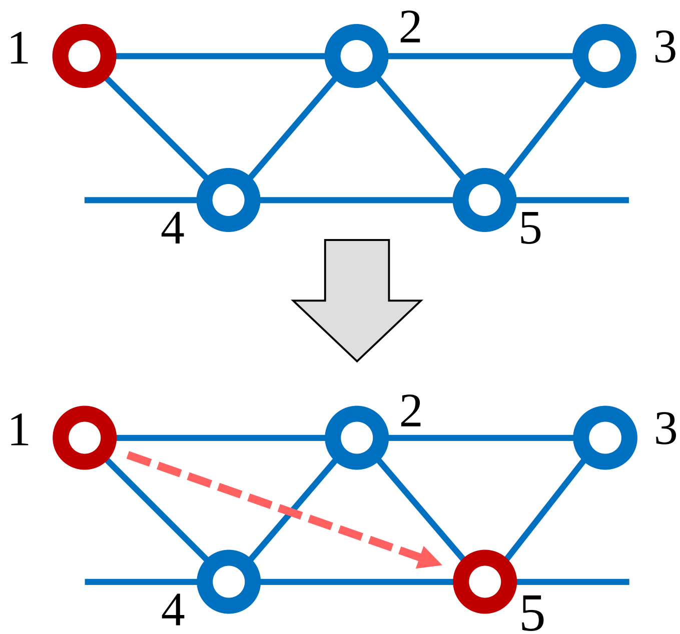
We applied influence model to model failure cascade.

- proposed a **hybrid learning framework** that can efficiently train the influence model for very large systems.

- applied the influence model to a few large scale power systems to predict their failure cascade sequences, whose performance is thoroughly evaluated at **different levels of granularity**.

- the influence model can predict the cascade sequence two orders of magnitude faster than simulation based on power flow calculation, with small compromise in accuracy

Influence Model



Failure of node-1 induces failure of node-5.

We can estimate a 2-by-2 influence matrix $\mathbf{A}_{15}$ for the pair of node-1 and node-5.

Prob of node-5 on if node-1 on

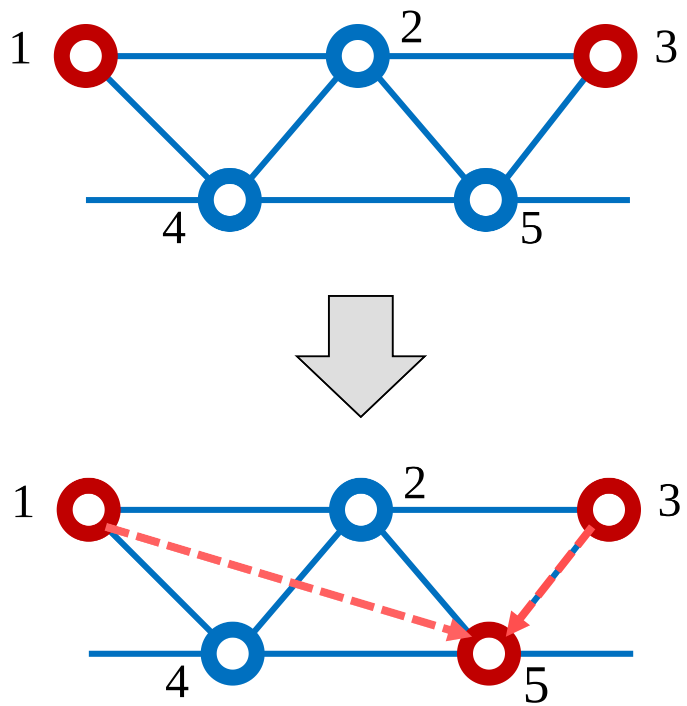
Prob of node-5 off if node-1 on

$$A_{15} = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$$

Prob of node-5 on if node-1 off

Prob of node-5 off if node-1 off

Influence Model



Node 3 may cause the failure of Node 5 as well.

The state of Node 5 depends on both Node 1 and Node 3. ($\mathbf{A}_{15}$, $\mathbf{A}_{35}$)

The total influence on Node 5 should be a weighted combination of $\mathbf{A}_{15}$, $\mathbf{A}_{35}$ (maybe involving other nodes)

The weights: $\boldsymbol{d}_{15}, \boldsymbol{d}_{35}$

If only Node 1 and 3 will influence Node 5, then we have

$$\boldsymbol{d}_{15} + \boldsymbol{d}_{35} = 1, \quad \boldsymbol{d}_{25} = \boldsymbol{d}_{45} = \boldsymbol{d}_{55} = 0$$

$$A_{15} = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$$

Influence Model Formulation

In our setting we consider the **link** failure cascade

Given a system with *N* buses and *M* transmission links.

The state of link i at time t: $s_i[t] \in \{0,1\}$ 1 normal; 0 failed

Each pair of link (i, j) induces an influence matrix $\mathbf{A}_{ij}$

$$\mathbf{A}_{ij} \in [0,1]^{2\times 2} \quad \text{and} \quad \begin{aligned} \mathbf{A}_{ij}(1,1) + \mathbf{A}_{ij}(1,2) = 1 \\ \mathbf{A}_{ij}(2,1) + \mathbf{A}_{ij}(2,2) = 1 \end{aligned}$$

Hence for $\mathbf{A}_{ij}$ only 2 independent components exist

- only consider $\mathbf{A}_{ij}(1,1)$ and $\mathbf{A}_{ij}(2,1)$ for each link pair (i,j)

$$\mathbf{A}_{ij} \in [0,1]^{2\times 2}$$

Influence Model Formulation

Build **pairwise influence matrices**: $\mathbf{A}^{11}, \mathbf{A}^{01} \in [0,1]^{M\times M}$

where $\mathbf{A}^{11}_{ji} = \mathbf{A}_{ji}(1,1) = P(s_i[t+1]=1 \mid s_j[t]=1)$

$$\mathbf{A}^{01}_{ji} = \mathbf{A}_{ji}(2,1) = P(s_i[t+1]=1 \mid s_j[t]=0)$$

Weight of the pairwise influence

$$d_{ij} \geq 0 \qquad \sum_{j=1}^{M} d_{ij} = 1$$

Therefore we have the following iteration

$$\tilde{s}_i[t+1] = \sum_{j=1}^{M} d_{ij}(\mathbf{A}^{11}_{ji} s_j[t] + \mathbf{A}^{01}_{ji}(1 - s_j[t]))$$

an estimated value of $\mathbf{s_i[t+1]}$

Massachusetts Institute of Technology

Influence Model Formulation

$$\tilde{s}_i[t+1] = \sum_{j=1}^{M} d_{ij}(\mathbf{A}_{ji}^{11} s_j[t] + \mathbf{A}_{ji}^{01}(1 - s_j[t]))$$

$\tilde{s}_i[t+1]$ can be viewed as probability that node i takes '1' at t+1

Hence an intuitive way to predict $s_i[t+1]$

$$\hat{s}_i[t+1] = \begin{cases} 1, & \text{w.p. } \tilde{s}_i[t+1] \\ 0, & \text{w.p. } 1 - \tilde{s}_i[t+1] \end{cases}$$

However in our prediction
- given an initial failure, the cascade process should be deterministic under fixed power injection
- we need to predict the whole process rather than only 1 generation.

Influence Model Formulation

We use the **threshold-**based prediction mechanism

$$\hat{s}_i[t+1] = \begin{cases} 1, & \tilde{s}_i[t+1] \geq \varepsilon_i \\ 0, & \tilde{s}_i[t+1] < \varepsilon_i \end{cases}$$

Hence the iteration process is

$$\tilde{s}_i[t+1] = \sum_{k=1}^{M} d_{ik} (\mathbf{A}_{ji}^{11} s_j[t] + \mathbf{A}_{ji}^{01} (1 - s_j[t]))$$

$$\hat{s}_i[t+1] = \begin{cases} 1, & \tilde{s}_i[t+1] \geq \varepsilon_i \\ 0, & \tilde{s}_i[t+1] < \varepsilon_i \end{cases}$$

$$t \leftarrow t+1$$

Given K historical sequences $\{s^k[t]\}_{t=0}^{T_k},\ k = 1,2,\ldots,K$

We need to learn $\{\mathbf{A}^{11}, \mathbf{A}^{01}, d, \varepsilon\}$

Hybrid Learning Framework

1. Learning Pairwise Influence Matrices $\{\mathbf{A}^{11}, \mathbf{A}^{01}\}$

   Monte-Carlo based method

2. Learning weights $\boldsymbol{d}$

   Convex quadratic programming

$$\min_{d} \frac{1}{K} \sum_{k=1}^{K} \sum_{t=1}^{T_k} \sum_{i=1}^{M} \left(s_i^k[t+1] - \tilde{s}_i^k[t+1] \right)^2$$

$$\text{s.t.} \quad \tilde{s}_i^k[t+1] = \sum_{j=1}^{M} d_{ij} (\mathbf{A}_{ji}^{11} s_j^k[t] + \mathbf{A}_{ji}^{01}(1 - s_j^k[t]))$$

$$d_{ij} \geq 0 \qquad \sum_{k=1}^{M} d_{ik} = 1$$

$$\boldsymbol{s}_i[\boldsymbol{t}+1]=\begin{cases}1, & \tilde{\boldsymbol{s}}_i[\boldsymbol{t}+1]\geq \varepsilon_i\\ 0, & \tilde{\boldsymbol{s}}_i[\boldsymbol{t}+1]< \varepsilon_i\end{cases}$$

Hybrid Learning Framework

3. Threshold ε

Example: one cascade sequence with 3 links

Link 1	0	0	0	0	0	0	No Way to Estimate!
$\tilde{s}_1[t]$	0	0.45	0.36	0.29	0.28	0.26	
Link 2	1	1	1	1	0	0	(0.67+0.63)/2=0.65
$\tilde{s}_2[t]$	1	0.78	0.71	0.67	0.63	0.62	
Link 3	1	1	1	1	1	1	0.8*0.76=0.608
$\tilde{s}_3[t]$	1	0.91	0.85	0.80	0.77	0.76	
t	0	1	2	3	4	5	

For the k-th sequence, we can estimate a threshold for each link i $\hat{\varepsilon}_i^k$

Idea:

- Forming the threshold pool of each link from all sample sequences
- To estimate the threshold values for each link under a new initial contingency, we use the threshold values of the training cascade sequence with initial state closest to the new one.

$$\boldsymbol{k}^* = \arg\min_{\boldsymbol{k}\in\{1,2,\ldots,\boldsymbol{K}\}} \left\| \boldsymbol{s}^{\text{new}}[0]-\boldsymbol{s}^{\boldsymbol{k}}[0]\right\|_1 \qquad \varepsilon^{\text{new}} = \varepsilon^{\boldsymbol{k}^*}$$

Performance Evaluation

Dataset:

TABLE I: Default Cascade Sample Information

System	1354-Bus	2383-Bus	3012-Bus
#Generators	260	327	297
#Links	1710	2886	3566
#Eff. Links	762	2088	2083
Eff. Rate	44.6%	72.4%	58.4%
Avg. Fail Size	179	598	263
Max Fail Size	314	862	792
Min Fail Size	2	110	11

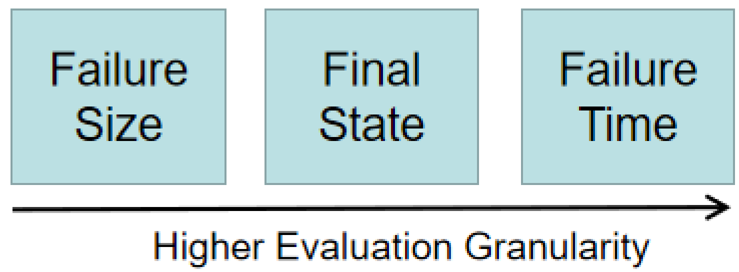
- consider M-2 initial contingencies
- training set size: at most 50,000
- training set accounts for very small portion $\frac{50{,}000}{\binom{1710}{2}} = 3.4\%$

Generate synthetic cascade sequence
- DC simulation with load shedding
- Fixing the power injection values

Performance Evaluation

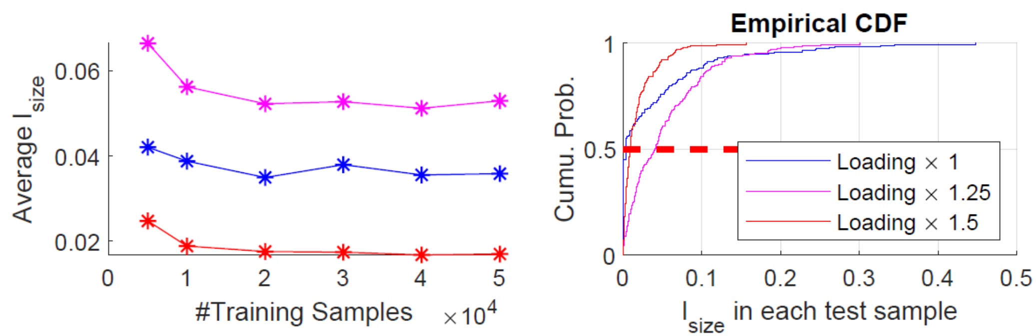
Metrics:

- I_{size}: Avg. Failure size error rate
- I_f: Avg. Final state error rate
- I_t: Avg. Failure time error



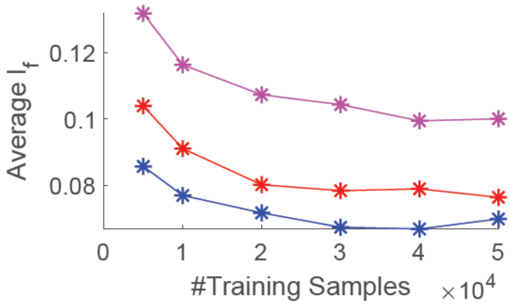
We consider 3 different values of loading:
- default loading & 1.5 times & 2 times

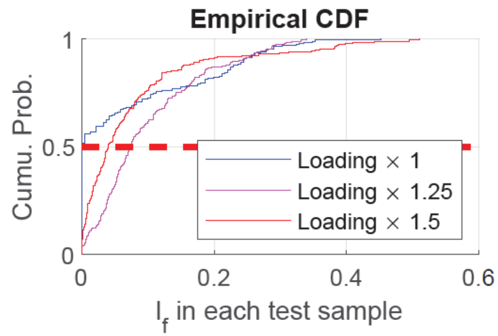
I_{size} (3012-node system)



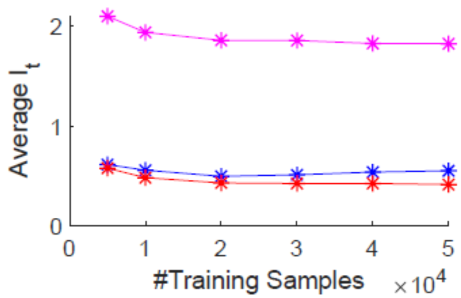
Performance Evaluation

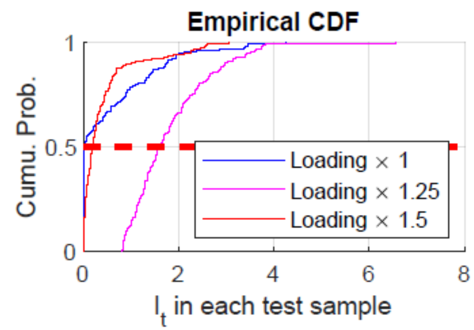
l_f (3012-node system)





l_t (3012-node system)





Performance Evaluation

Prediction time cost:

- compare time cost of our prediction and simulation by DC power flow calculation
- a | b | **c**, where 'a' is the time cost of the DC simulation, 'b' is that of our prediction, 'c' is b/a, the gain.

TABLE II: Prediction Time Cost on 1,000 Samples

	Low Load	Medium Load	High Load
1354-bus	808 | 21.3 | **38**	1930 | 19.8 | **97**	1740 | 19.6 | **89**
2383-bus	2597 | 43.3 | **60**	3490 | 37.8 | **92**	3603 | 34.7 | **104**
3012-bus	3891 | 59.4 | **66**	8020 | 58.9 | **136**	5864 | 46.9 | **125**

Contents

M.S./Ph.D. Stage

- Finite-Buffer Communication Networks Analysis
- Power System Failure Cascade Analysis

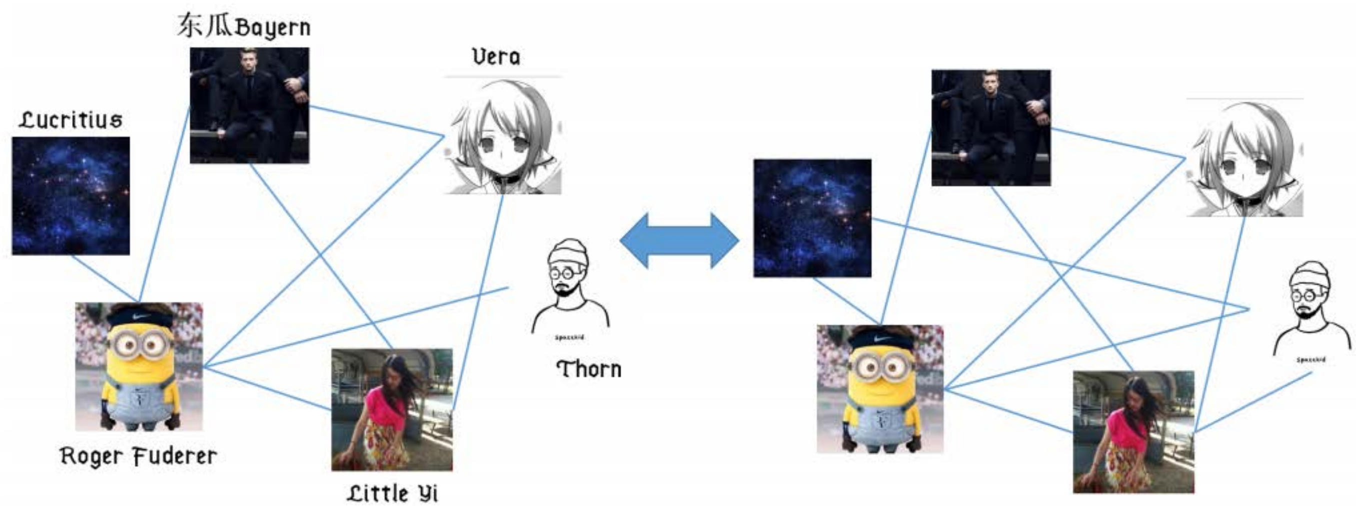
B.S. Stage

- Social Network De-anonymization
- Wireless Fingerprints Prediction for Positioning

Social Network De-Anonymization

Motivation:

Whether anonymity can protect network users from privacy leakage?

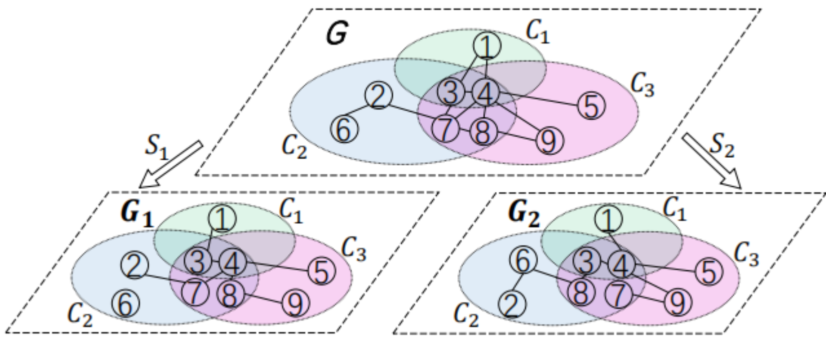


Un-anonymized LinkedIn **Anonymized Wechat**

Social Network De-Anonymization

Problem Modeling:

Can We De-anonymize an anonymized network?



Model:
Overlapping Stochastic Block Model

G: Underlying Relationship Network
G_1: Anonymized Network
G_2: Unanonymized Network

Goal:
Finding the **correct user correspondence** between networks G_1 and G_2.

Social Network De-Anonymization

Contributions:

1. Theoretical Aspect:

- **Derived a cost function** to evaluate the average de-anonymization error based on **Minimum Mean Square Error** criterion.

$$\hat{\mathbf{\Pi}} = \arg\min_{\mathbf{\Pi}\in\mathbf{\Pi^n}} \mathbf{E}_{\mathbf{\Pi_0}}\{d(\mathbf{\Pi}, \mathbf{\Pi_0})\}$$
$$= \arg\min_{\mathbf{\Pi}\in\mathbf{\Pi^n}} \sum_{\mathbf{\Pi_0}\in\mathbf{\Pi^n}} ||\mathbf{\Pi} - \mathbf{\Pi_0}||_F^2 Pr(\mathbf{\Pi_0}|G_1, G_2, \boldsymbol{\theta}),$$

- **Showed the NP-hardness** of minimizing the cost function.
- Transformed the problem into a **polynomially solvable** one by restriction of **Sequence Inequality**.

$$\tilde{\mathbf{\Pi}} = \arg\min_{\mathbf{\Pi}\in\mathbf{\Pi^n}} ||\mathbf{\Pi}\hat{\mathbf{A}} - \hat{\mathbf{B}}\mathbf{\Pi}||_F^2.$$

Social Network De-Anonymization

Transformed Cost Function

$$\tilde{\Pi} = \arg\min_{\Pi \in \Pi^n} ||\Pi\hat{A} - \hat{B}\Pi||_F^2.$$

2. Algorithmical Aspect:

- **Optimality:**
  - Showed that minimizing the transformed cost function can **asymptotically vanish the de-anonymization error** under **mild conditions**.

Theorem 2. *Given* $G_1(V_1, E_1, \mathbf{A})$, $G_2(V_2, E_2, \mathbf{B})$, $\boldsymbol{\theta}$ *and* $\mathbf{W}$. *Set* $\tilde{p}_{\mathbf{C}_i\mathbf{C}_j} = w_{ij}p_{\mathbf{C}_i\mathbf{C}_j}$ *and*

$$K = \min_{s,t,j}\{(\tilde{p}_{\mathbf{C}_s\mathbf{C}_j} + \tilde{p}_{\mathbf{C}_t\mathbf{C}_j})\min\{s_1, s_2\}\},$$
$$L = \max_{s,t,j}\{[(\tilde{p}_{\mathbf{C}_s\mathbf{C}_j} + \tilde{p}_{\mathbf{C}_t\mathbf{C}_j})\max\{s_1, s_2\}]^2\}. \qquad (6)$$

If

- *(i)* $\frac{L}{K} = o(1)$;
- *(ii)* $\frac{||\hat{\mathbf{A}} - \mathbf{\Pi_0}\hat{\mathbf{B}}\mathbf{\Pi_0^T}||_F^2}{||\hat{\mathbf{A}} - \tilde{\mathbf{\Pi}}\hat{\mathbf{B}}\tilde{\mathbf{\Pi}}^\mathbf{T}||_F^2} = \Omega(1)$;
- *(iii)* $||\hat{\mathbf{A}} - \mathbf{\Pi_0}\hat{\mathbf{B}}\mathbf{\Pi_0^T}||_F^2 = o(Kn^2)$;
- *(iv)* $\mathbf{\Pi_0}$ *and* $\tilde{\mathbf{\Pi}}$ *keep invariant of community representations,*

then as $n \to \infty$, $\frac{||\tilde{\mathbf{\Pi}} - \mathbf{\Pi_0}||_F^2}{||\mathbf{\Pi_0}||_F^2} \to 0$.

Corollaries:

- Overlapping Stochastic Block Model meets such conditions.
- More overlapping, better de-anonymization.

Proof Technique: Matrix Theory

Social Network De-Anonymization

2. Algorithmical Aspect:

- **Solvability**:
  - Designed a **Convex-concave** Based De-anonymization Algorithm to solve de-anonymization.

Transformed Cost Function

$$\tilde{\mathbf{\Pi}} = \arg\min_{\mathbf{\Pi}\in\mathbf{\Pi^n}} ||\mathbf{\Pi}\hat{\mathbf{A}} - \hat{\mathbf{B}}\mathbf{\Pi}||_F^2.$$

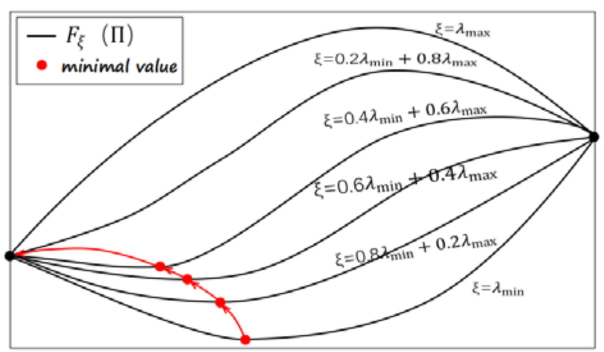
$F_0(\mathbf{\Pi})$	Cost function with communities
$F_1(\mathbf{\Pi})$	Convex Relaxation of $F_0(\mathbf{\Pi})$
$F_2(\mathbf{\Pi})$	Concave Relaxation of $F_0(\mathbf{\Pi})$
$F(\mathbf{\Pi})$	Convex-concave based cost function

$$F_0(\mathbf{\Pi}) = ||\hat{\mathbf{A}} - \mathbf{\Pi}\hat{\mathbf{B}}\mathbf{\Pi}^{\mathbf{T}}||_{\mathrm{F}}^2 + \mu||\mathbf{\Pi}\mathbf{M} - \mathbf{M}||_F^2$$

$$F_1(\mathbf{\Pi}) = F_0(\mathbf{\Pi}) + \frac{\lambda_{min}}{2}(n - ||\mathbf{\Pi}||_F^2);$$

$$F_2(\mathbf{\Pi}) = F_0(\mathbf{\Pi}) + \frac{\lambda_{max}}{2}(n - ||\mathbf{\Pi}||_F^2)$$

$$F(\mathbf{\Pi}) = (1-\alpha)F_1(\mathbf{\Pi}) + \alpha F_2(\mathbf{\Pi})$$



Algorithm 1: Convex-concave Based De-anonymization Algorithm (CBDA)

Input: Adjacent matrices $\mathbf{A}$ and $\mathbf{B}$; Community assignment matrix $\mathbf{M}$; Weight controlling parameter μ; Adjustable parameters δ, $\Delta\xi$.

Output: Estimated permutation matrix $\tilde{\mathbf{\Pi}}$.

```
Form the objective function F_0(Π) and F(Π).
ξ ← 0, k ← 1, Π_1 ← 1_{n×n}./n. Set ξ_m, the upper limit of ξ.
while ξ < ξ_m and Π_k ∉ Ω_0 do
    while k = 1 or |F(Π_{k+1}) − F(Π_k)| ≥ δ do
        X^⊥ ← arg min_{X^⊥} tr(∇_{Π_k} F(Π_k)^T X^⊥), where X^⊥ ∈ Ω.
        γ_k ← arg min_γ F(Π_k + γ(X^⊥ − Π_k)), where γ_k ∈ [0, 1].
        Π_{k+1} ← Π_k + γ_k(X^⊥ − Π_k), k ← k + 1.
    end while
    ξ ← ξ + Δξ.
end while
```

Social Network De-Anonymization

3. Experimental Aspect:

Datasets: (1) **SNAP, 500~2000** nodes; (2) **Microsoft Academic Map, ~3000** nodes

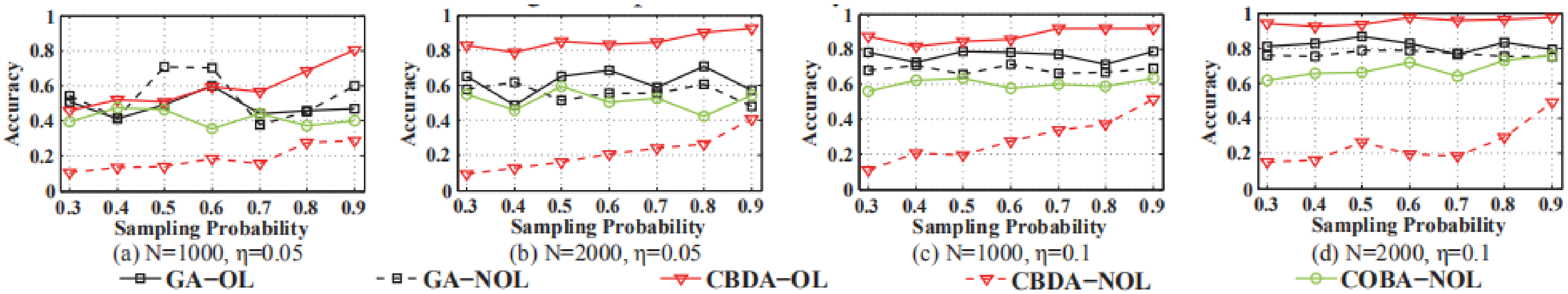


Fig. 3: Experiments on Sampled Real Social Networks.

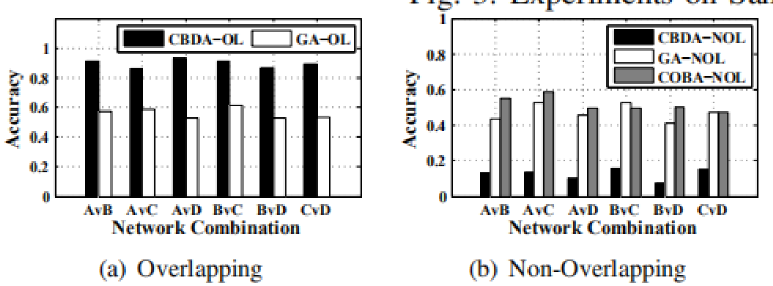


Fig. 4: Experiments on Cross-Domain Co-author Networks

Conclusions:

- Overlapping Communities **benefits** higher de-anonymization accuracy.
- Our algorithm suits better in **large social networks**.

Contents

M.S./Ph.D. Stage

- Finite-Buffer Communication Networks Analysis
- Power System Failure Cascade Analysis

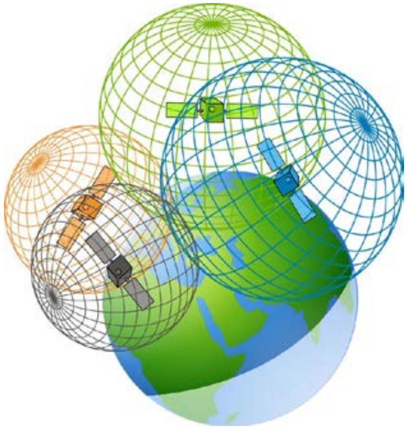
B.S. Stage

- Social Network De-anonymization
- Wireless Fingerprints Prediction for Positioning

Wireless Fingerprints Prediction for Positioning

Motivation:

GPS performs poorly in **Urban Canyon Areas**.



At least 3 satellites for GPS positioning

- Some locations are unable to have 1 single satellite visible;
- A notable of locations have less than 3 satellites visible.

How to overcome the weakness of GPS positioning?

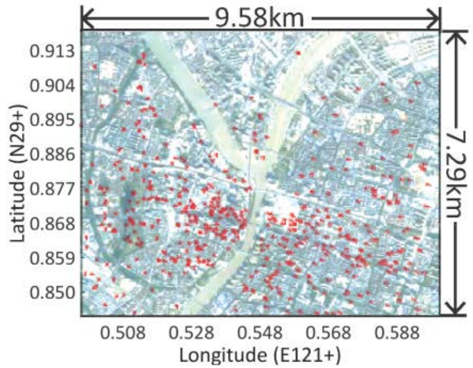
[1] X. Wu, X. Tian, X. Wang, "Large-scale Wireless Fingerprints Prediction for Cellular Network Positioning'', IEEE International Conference on Computer Communications (INFOCOM), 2018.

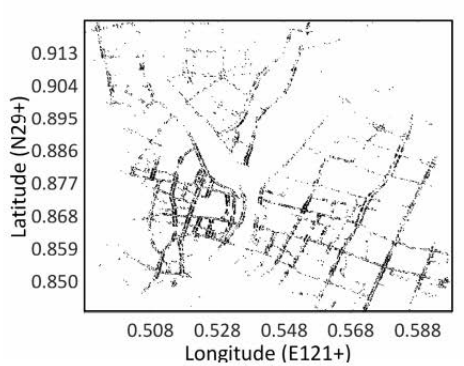
Wireless Fingerprints Prediction for Positioning

Can We Utilize Fingerprinting Localization to Enhance the performance?

Requirement: A Whole Fingerprinting Database

Challenge: Vast areas needed to be surveyed in outdoor situation.





Idea: Sample a part of easily available fingerprints, and predict other parts.

Goal:

Propose **Fingerprinting Prediction Mechanism** & Conduct **Localization**

Wireless Fingerprints Prediction for Positioning

Problem Modeling:

$$\text{Fingerprint Prediction} \rightarrow \text{Matrix Completion}$$

$$\min_{\Omega,\hat{A}} \| P_\Omega(A) - P_\Omega(\hat{A}) \|, \ \text{s.t.} |\Omega| \le |\Omega_m|$$

A : The Groundtruth Radio Map

$\hat{A}$: An Estimation of A

$P_\Omega(A)$: The Sampled Fingerprints in A

$|\Omega|$: The Number of Samples

$|\Omega_m|$: The Upper Limit of $|\Omega|$

A :

-51	-45	-66
-74	-70	-57
-68	-63	-59

$P_\Omega(A)$:

-51	?	?
?	-70	-57
-68	?	-59

How to solve this optimization problem?

Wireless Fingerprints Prediction for Positioning

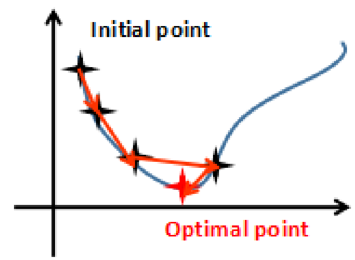
Contributions:

1. Fingerprinting Prediction:

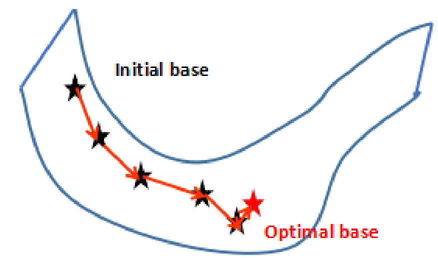
- Proposed a **Streamlined Stiefel-manifold Optimization Algorithm (SSOA)** for fingerprinting prediction.
  - ➢ Inspired by Gradient Descent Algorithm -> Developed **Gradient Descent on Stiefel Manifolds**

Stiefel manifold:

-- The set of all d-dimensional orthonormal bases of a subspace.



Gradient Descent on Real Value



Gradient Descent on Stiefel Manifold

Theoretical Challenges:

- **Finding the descent direction**
- **Obtaining the iteration equation**
- **Determining the step size**

Wireless Fingerprints Prediction for Positioning

1. Fingerprinting Prediction:

- Proposed a **Streamlined Stiefel-manifold Optimization Algorithm (SSOA)** for fingerprinting prediction.
  - Developed **Gradient Descent on Stiefel Manifolds**

Descent direction ∇F

$$F = \sum_{i=1}^{n} ||[U_d w_j]_\Omega - [a_j]_\Omega||_2^2$$

Iteration: $$U_{t+1} = U_t + 2\eta_t \frac{r_t w_t^T}{||r_t|| ||w_t||}$$

Step Size: $$\eta_t = \frac{1}{2} \frac{||r_t||}{||w_t||}$$

Algorithm 1: Streamlined Stiefel-manifold optimization algorithm (SSOA)

Input:
An initial column-orthonormal $m \times d$ matrix U_0;
sample set Ω, $m \times n$ sample matrix $P_\Omega(A)$;
maximum number of iteration T.

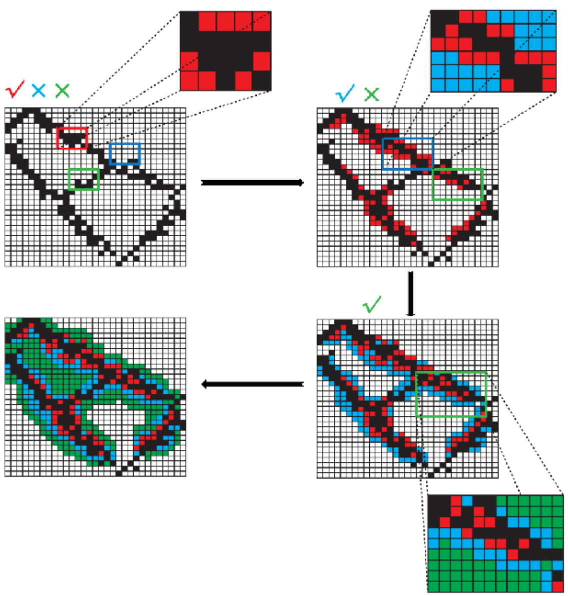
Output:
Estimated matrix A_d.

```
t = 0;
while t < T do
    Randomly choose a column index q ∈ {1, 2, ..., n}, get [a_q]_Ω;
    w_t = ([U_t]_Ω^T [U_t]_Ω)^{-1} [U_t]_Ω [a_q]_Ω;
    p_t = U_t w_t;
    r_t = P_Ω(v_t − p_t);
    U_{t+1} = U_t + r_t w_t^T / ||w_t||^2;
    t = t + 1;
end while
U = U_t;
for each i ∈ {1, 2, ..., n} do
    â_i = U([U]_Ω^T [U]_Ω)^{-1} [U]_Ω [a_i]_Ω;
end for
A_d = [â_1, â_2, ..., â_n].
```

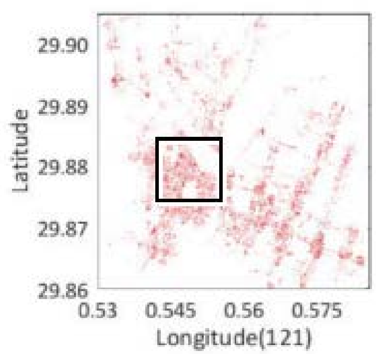
Wireless Fingerprints Prediction for Positioning

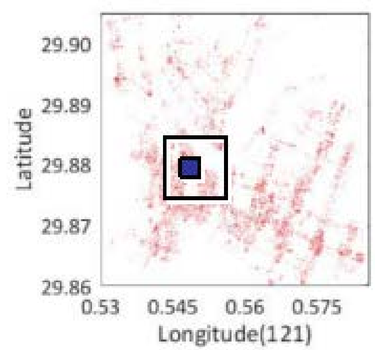
1. Fingerprinting Prediction:

- Designed a **sliding-window mechanism** to build up the fingerprint database for **the whole region**.
- Provided a reasonable way to **estimate the dimension of the subspace** when applying SSOA in a window



Theorem 2. *We use l_d and $\tilde{l}_d$ to denote the remained information of the incomplete $p \times q$ window matrix A and a complete $s \times t$ sub-matrix $\tilde{A}$ within, where we set $p \leq q$ and $s \leq t$, and d is the dimension of the subspace obtained by performing SVD to the sub-matrix; if the linear correlation of fingerprints is strong enough in the window matrix A, then $|l_d - \tilde{l}_d| \to 0$.*





Wireless Fingerprints Prediction for Positioning

2. Fingerprinting Localization:

- Conducted Localization on a **69.8km²** region, with **8,820,000** records. Showed that our **SSOA** triumphed over **Cell-ID** and **Gaussian-Mixture-Model** approaches.

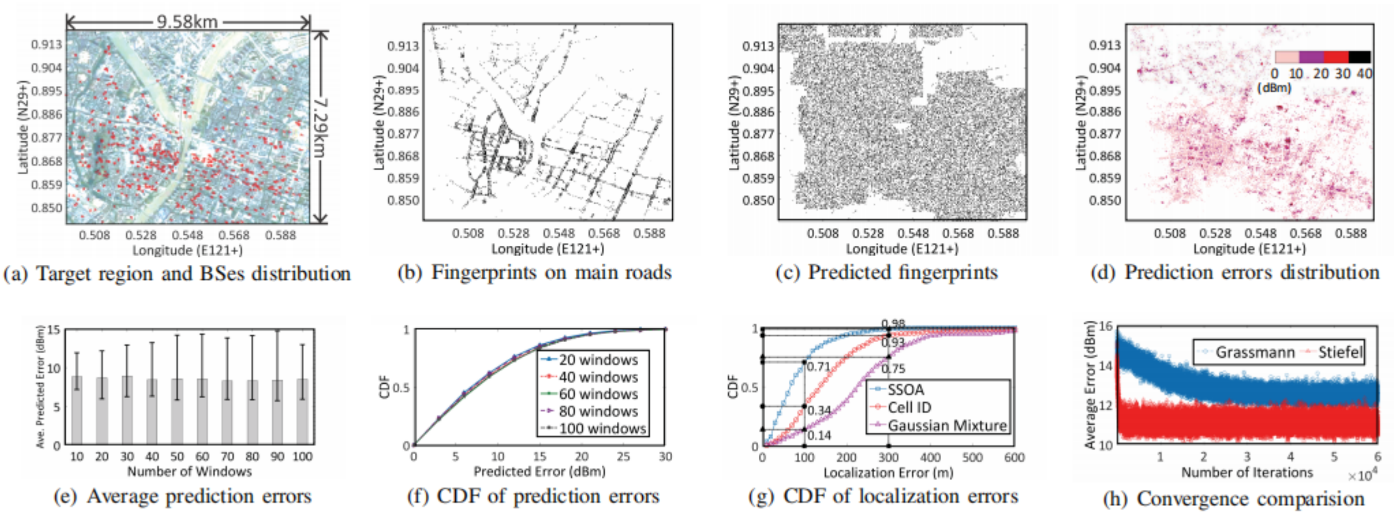


Fig. 2. Experimental Results on $69.8km^2$ Data Set