Detecting Cash-out Users via Dense Subgraphs

Yingsheng Ji[1], Zheng Zhang[1], Xinlei Tang[1], Jiachen Shen[1], Xi Zhang[2], Guangwen Yang[3]

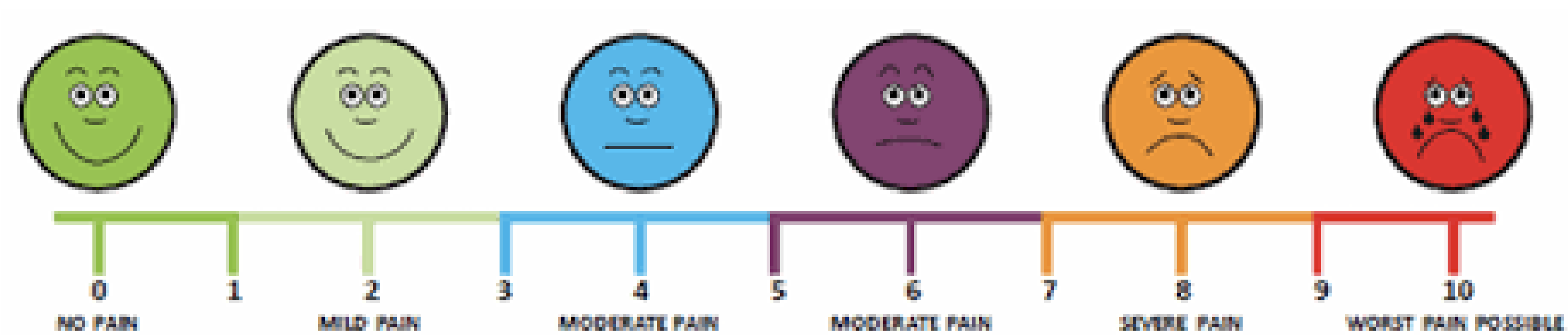
[1]China Etek Service & Technology

[2]Beijing University of Posts and Telecommunications

[3]Tsinghua University

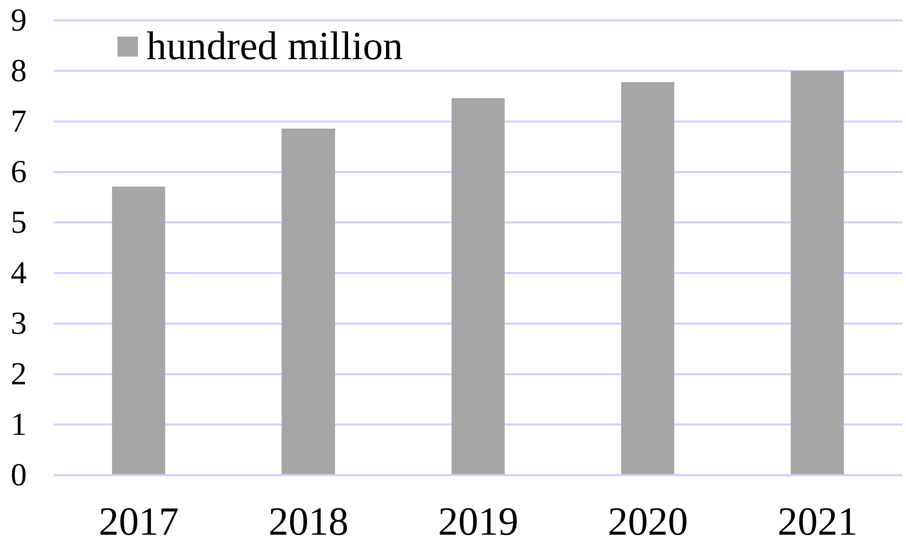
What is cash-out fraud?

- Cash-out is a fraud, where
  - Users get money from credit cards
  - Merchants offer fake purchase services
- For credit card services in bank, cash-out accounts are gray or black



Why should we care?

The number of credit cards issued in China

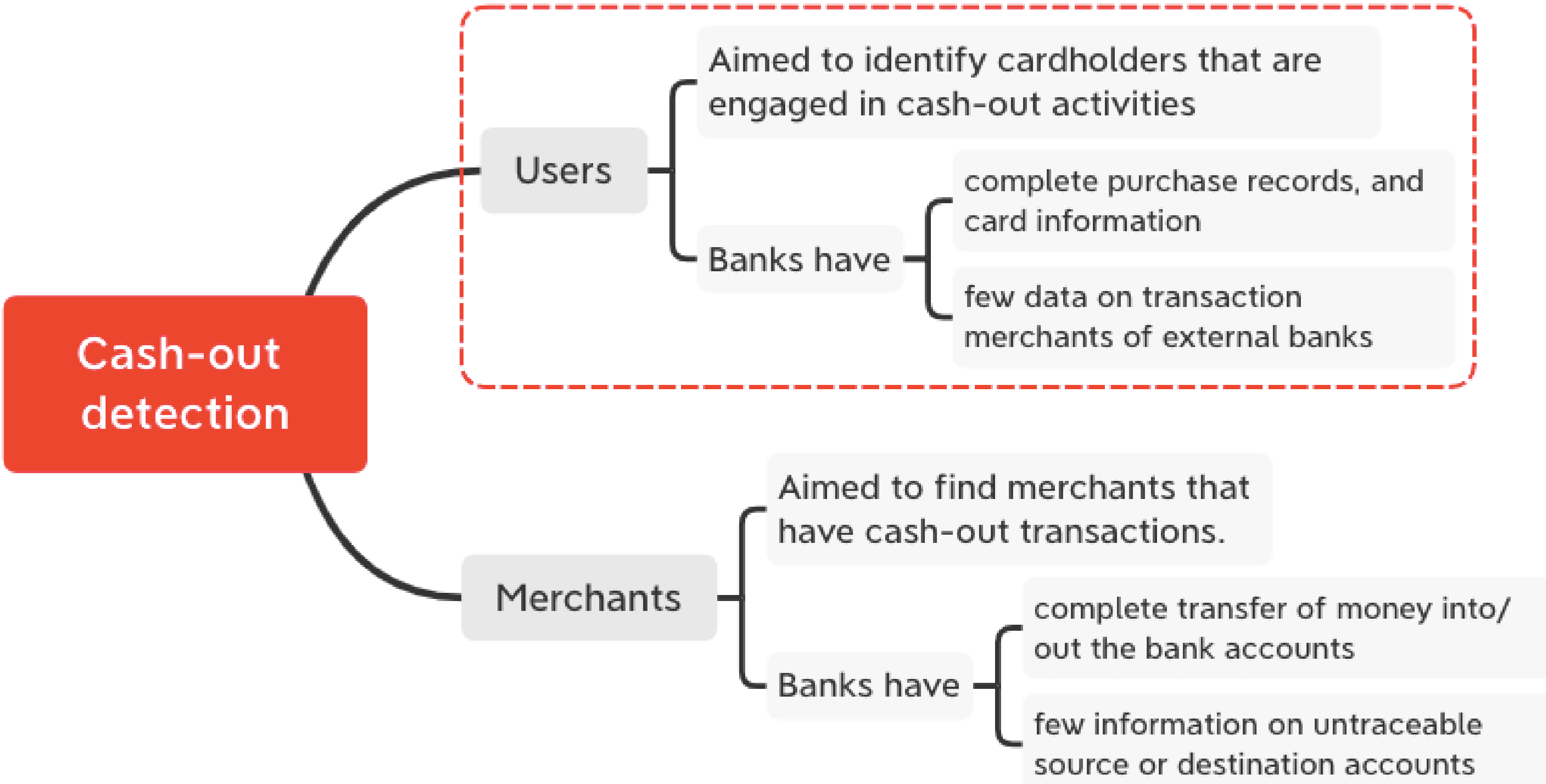


The charge-off and delinquency rate of major banks in China

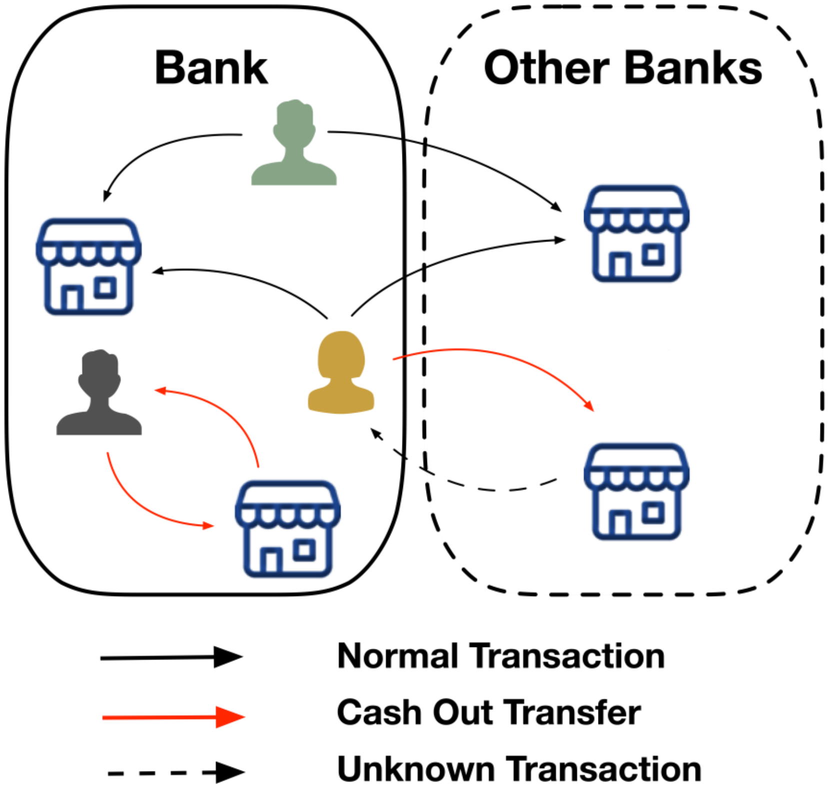
Bank	2017	2018	2019	2020	2021
CCBC	1.17%	1.09%	1.21%	1.40%	1.33%
BCM	1.98%	1.84%	2.49%	2.27%	2.20%
CMB	1.26%	1.14%	1.30%	1.66%	1.65%
CNCB	1.30%	0.98%	1.74%	1.65%	1.83%
PAB	1.20%	1.19%	1.66%	2.16%	2.11%

Data sources: Banking industry report

Two separate tasks



Challenges: broken money transfer chain



- The essential way is to find capital recycling into connected accounts
- However,
  - Fraudsters manipulate inter-bank transfers
  - Banks handle no chain transactions

Challenges: deceptive pattern

Date	Amount	For
2022/6/2 10:01	$38.00	Coffee
2021/6/2 10:34	$150.00	Shopping
2021/6/2 10:50	$169.00	Shopping
2021/6/2 11:00	$45.00	Lunch
2021/6/2 12:20	$270.00	Groceries
2021/6/2 13:30	$75.00	Yoga Class
2021/6/2 14:45	$175.00	Shopping
2021/6/2 16:50	$100.00	Gas Station
2021/6/2 19:00	$650.00	Steak House
2021/6/2 21:37	$300.00	Spa
2021/6/2 23:09	$199.00	Club
TOTAL	$2,171.00	

- Cash-out is a thriving industry
  - Much less than interest
  - Third-part agencies offer diverse cash-out ways
- Perfect-bill services can offer highly imitation daily expenses

Other challenges on data

Lacking labels

Limited knowledge cash-out behaviors, and deficient sample accumulation

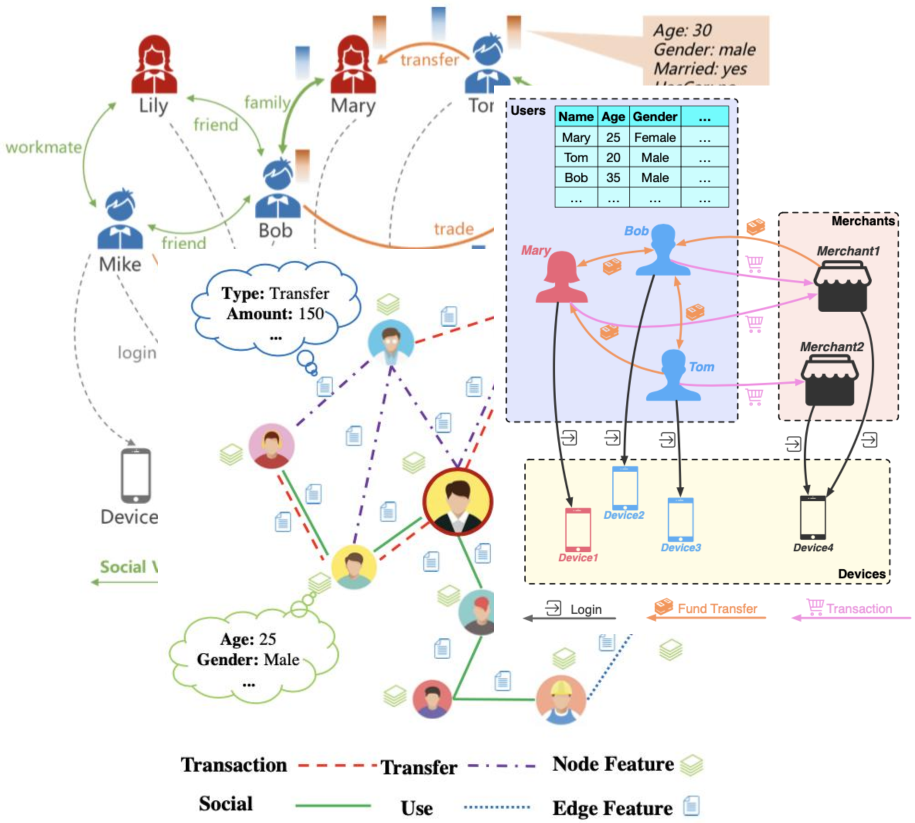
Offline scenarios

Traditional credit card services have POS transactions, while online services have more data for modeling

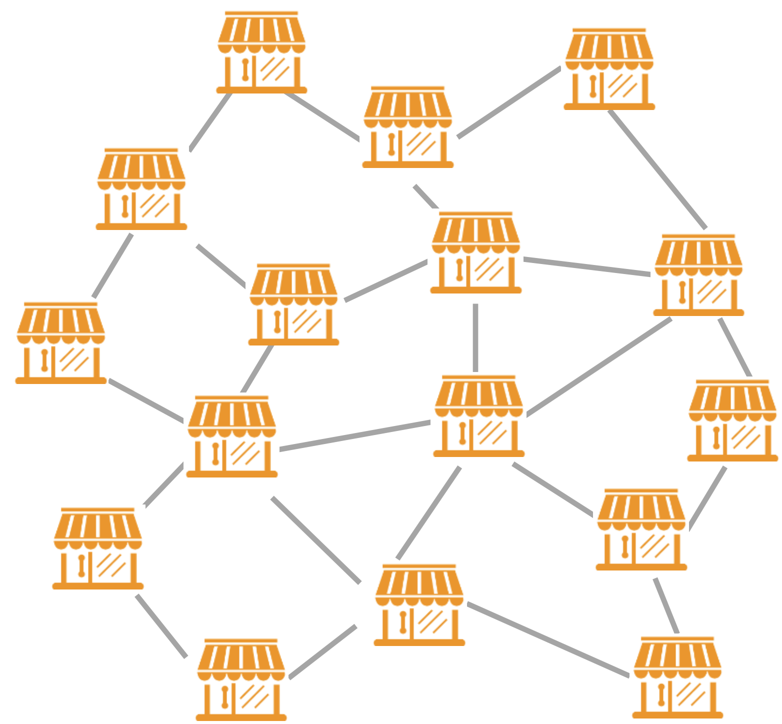
Motivations

- Graph mining:
  - To find cash-out users
  - To leverage relationships in transactions
- Problem#1: How do graphs look like?
- Problem#2: How do cash-out users behave?
- Problem#3: How to spot cash-out users?

Problem #1 - How do graphs look like?

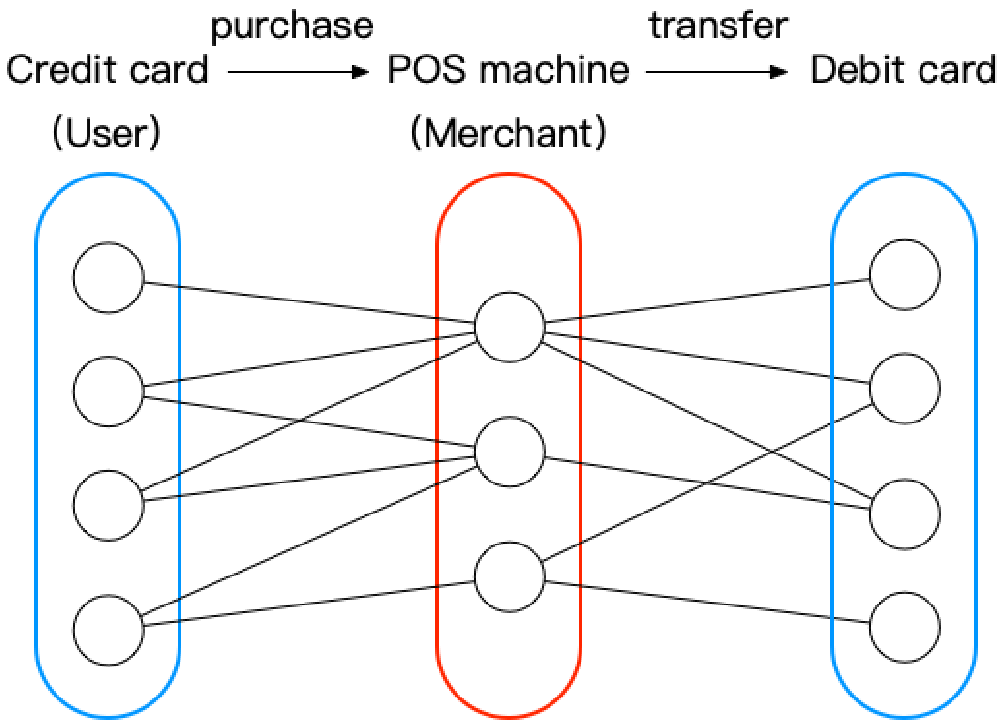


HACUD (AAAI 2019), MAHINDER (WWW 2020), AMG-DP (CIKM 2020)

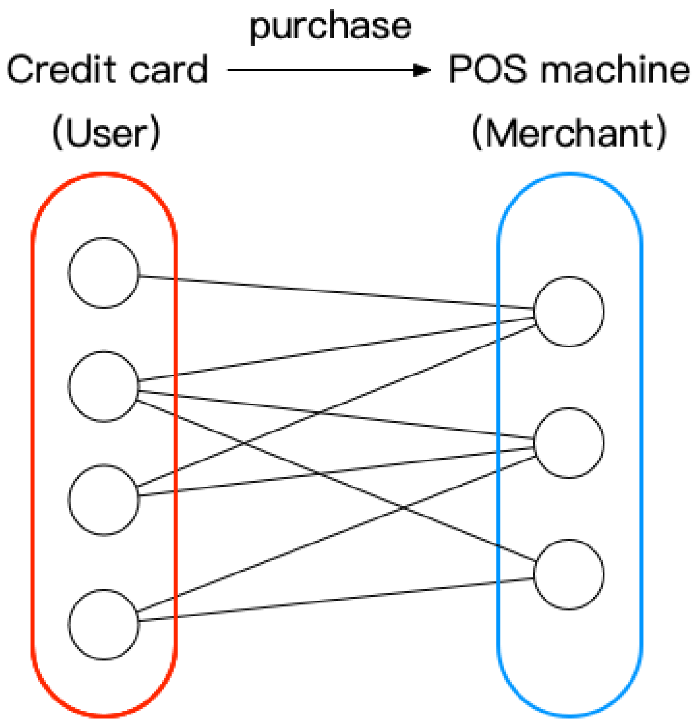


Shared-Card-Based Network, e.g.Capital One, Unionpay

We use k-partite graph



FLOWSCOPE (AAAI 2020)



ANTICO

Problem #2 - How do cash-out users behave?

Fraudulent schemes are for financial gain.
Fraudsters share and reuse resources (e.g. accounts, POS machine)
to maximize benefits.

Under such economic law, financial fraudulent activities usually give rise to the temporal-spatial aggregation.

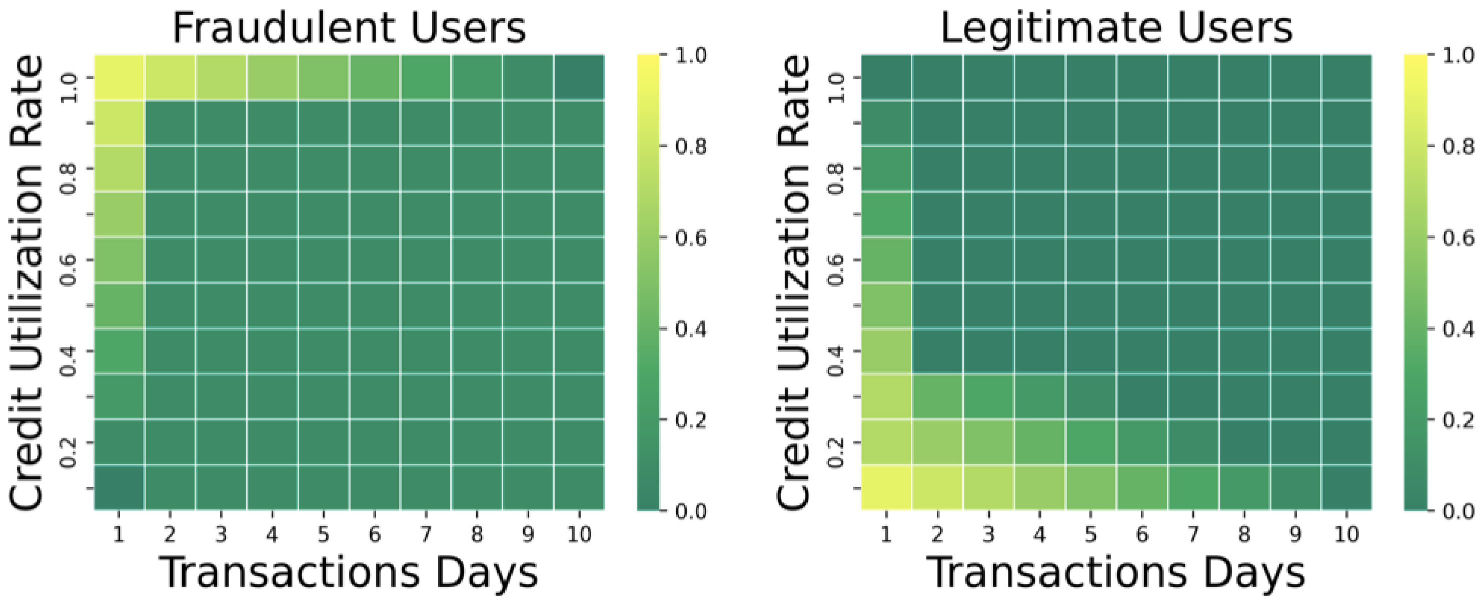
Cash-out behaviors involve

Intensiveness

Cash-out users tend to draw cash from credit cards within a short time.

Mass

Cash-out exhaust most credit limits. Meanwhile, users may have general payments.



Cash-out behaviors involve

Cyclicity

Cash-out activities behave periodically.

Shared merchants

Cash-out transactions are concentrated on limited merchants.

Problem #3 - How to spot cash-out users?

Crowds of cash-out behaviors differ from normal ones with respect to their unavoidable transaction connectivity, i.e. edges in graph.

These differences can be detected by identifying structural anomalies with density signals.

Formal problem

- Formalization

Given a card-merchant transaction graph G,

-**find** dense subgraphs in G, consisting of cash-out users and merchants,

-**optimize** a class of suspiciousness metrics under the traits in terms of temporal, capital and topological behaviors.

- Requirements

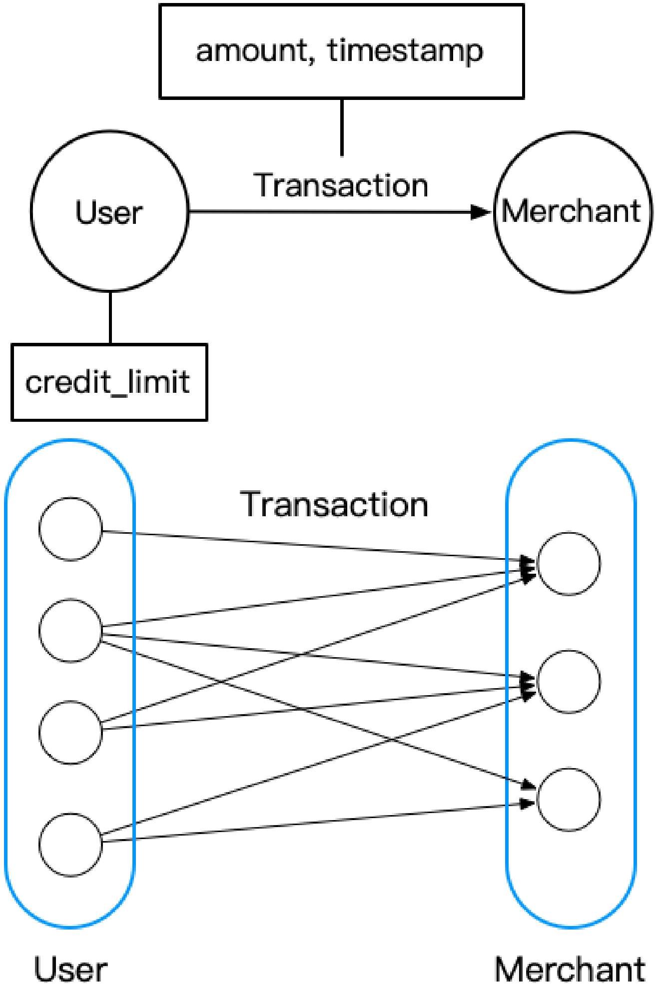
The model should have

- ☐ **Effectiveness:** works in practice
- ☐ **Accuracy:** provides an accuracy guarantee
- ☐ **Runtime:** runs in near-linear time

Define a graph

Modeling

Focus on single-step transactions and define a user-merchant bipartite graph

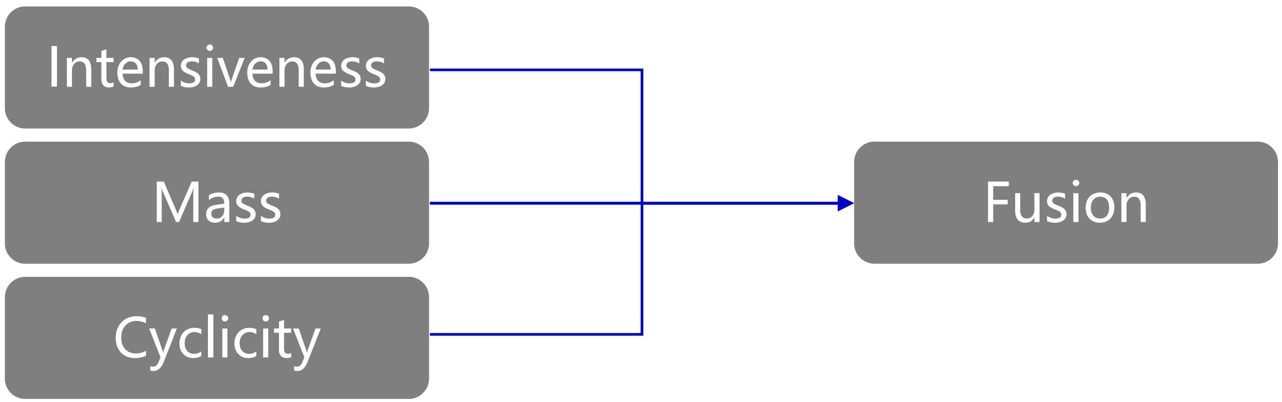


Objective

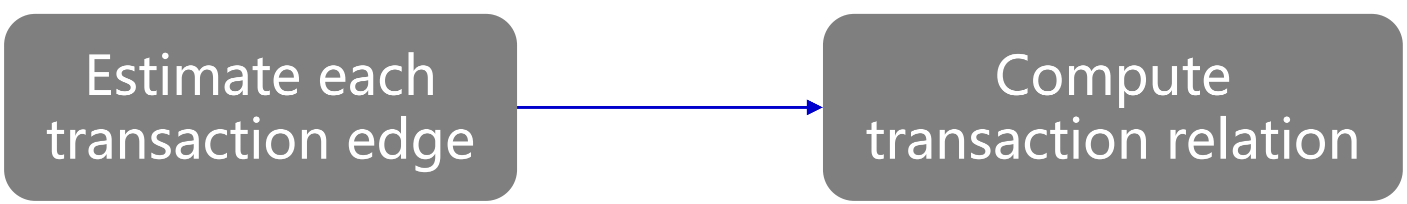
Detect dense subgraphs containing cash-out users and merchants

Assess cash-out suspiciousness

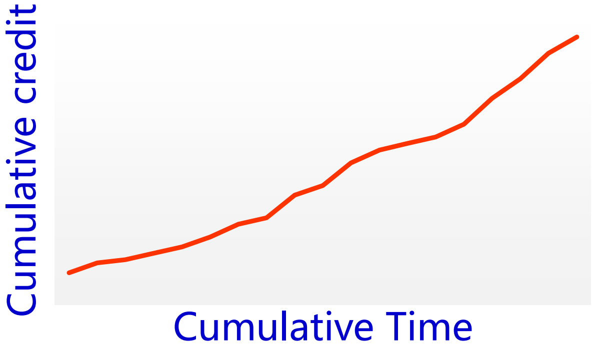
- Evaluate the transactions made by a user within the given time



- Evaluate the transaction relation between user-merchant node pairs

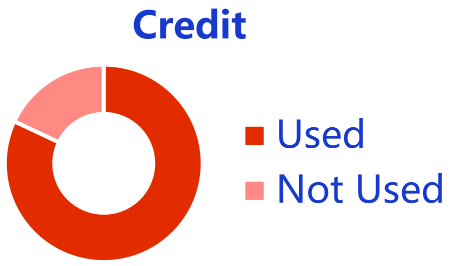


A class of metrics



Intensiveness

Gini Index is adopted to measure temporal uneven patterns.



Mass

A natural choice is to adopt credit utilization rate.

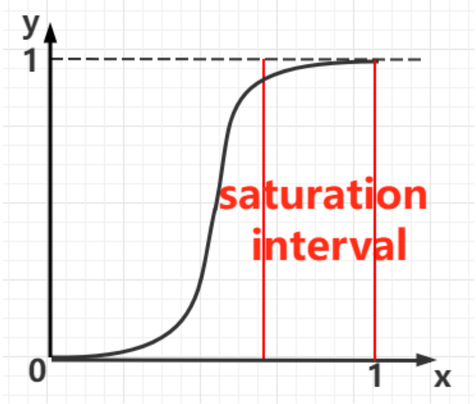
Cyclicity

The average of multiple billing cycles are calculated.

A class of metrics

Fusion

A sigmoid function is employed for probability estimate.

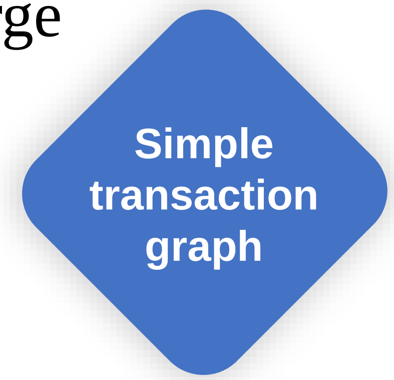


Edge suspiciousness

A higher weight is assigned to any transaction within the time slice when the volume is large

Aggregation

Transaction relations are weighted by scaling amounts.



Edge suspiciousness depends on the temporal distribution

- Can the suspiciousness be further allocated to transactions within a time slice?
  - Fraudsters can reduce suspiciousness by faking a batch of petty trades.
  - Banks charge by the amount, not the number of transactions. The latter wouldn’t raise cost.
- What if normal transactions occur in the time slice with a high weight?
  - The noises can be handled by exploring topology in graph.

Overall framework

Density metrics

$$g(A \cup B) = \frac{1}{|A| + |B|} \sum_{i \in A} \sum_{j \in B} a_{ij} \cdot e'_{ij} \cdot c_{ij}$$

Transaction relation

$$c_{ij} = \sum_{e_{ij}(t) \in E} \frac{e_{ij}(t)}{\sum_{\tau} e_{ij}(\tau)} \cdot b_{ij}(t)$$

Transaction suspiciousness

$$b_{ij}(t) \coloneqq P(i \in A, j \in B | i \in A, t) \cdot P(i \in A)$$

$$P(i \in A, j \in B | i \in A, t) \propto \frac{\sum_{\tau \in T_s^{(k)}} e_{i*}(\tau)}{\sum_{t \in T} \sum_{\tau \in T^{(k)}} e_{i*}(\tau)}$$

Fusion

$$P(i \in A) \propto \frac{1}{1 + b^{(1 - 2\phi(\mathcal{H}_i))}}$$

$$where\ \phi(\mathcal{H}_i) = \sum_{h_{ij} \in \mathcal{H}_i} \frac{1}{|\mathcal{H}|} \cdot h_{ij}$$

Gini $\mathcal{T}_i(T^{(k)}) = \frac{2 \sum_{j=1}^{d} j \cdot x_j}{d \sum_{j=1}^{d} x_j} - \frac{d+1}{d}$

Mass $\mathcal{R}_i(T^{(k)}) = \frac{\sum_{t \in T^{(k)}, j \in M} e_{ij}(t)}{X_{ik}}$

$$\mathcal{H} = \{\mathcal{T}, \mathcal{R}\}$$

Greedy algorithm

Step 1

Initailize the node score

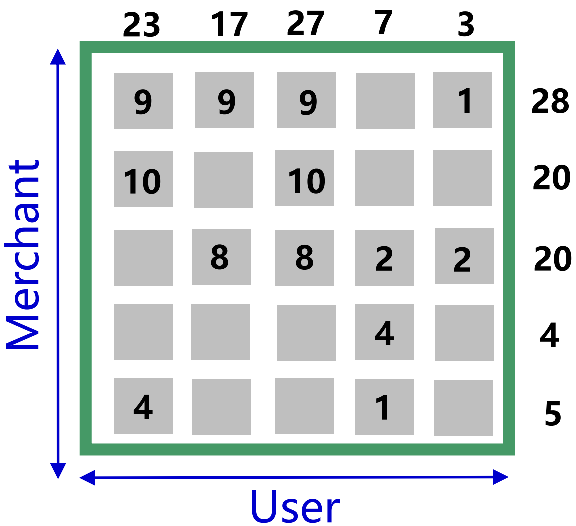
$$w_i(S) = \begin{cases} \sum_{j \in B} a_{ij} \cdot e'_{ij} \cdot c_{ij} & i \in A \\ \sum_{j \in A} a_{ji} \cdot e'_{ji} \cdot c_{ji} & i \in B \end{cases}$$

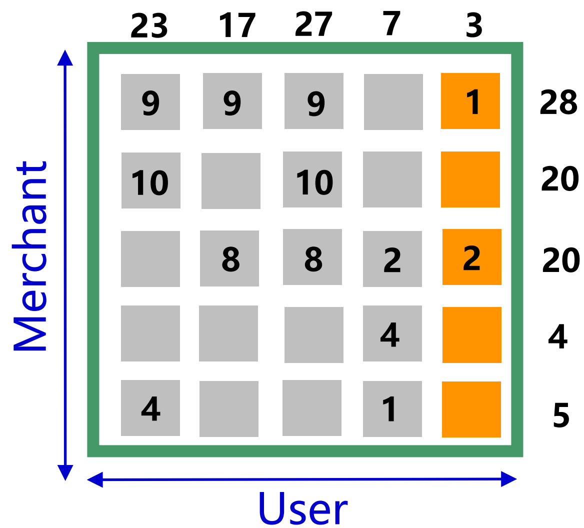
Step 2

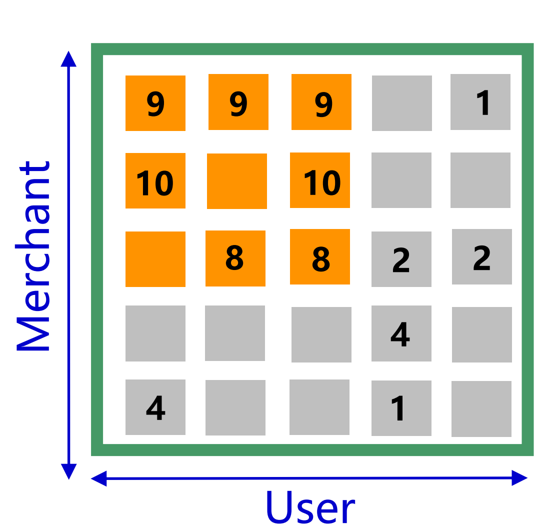
Greedily peeling nodes from the whole graph until empty sets

Step 3

Return the result that maximizes $g(A \cup B)$



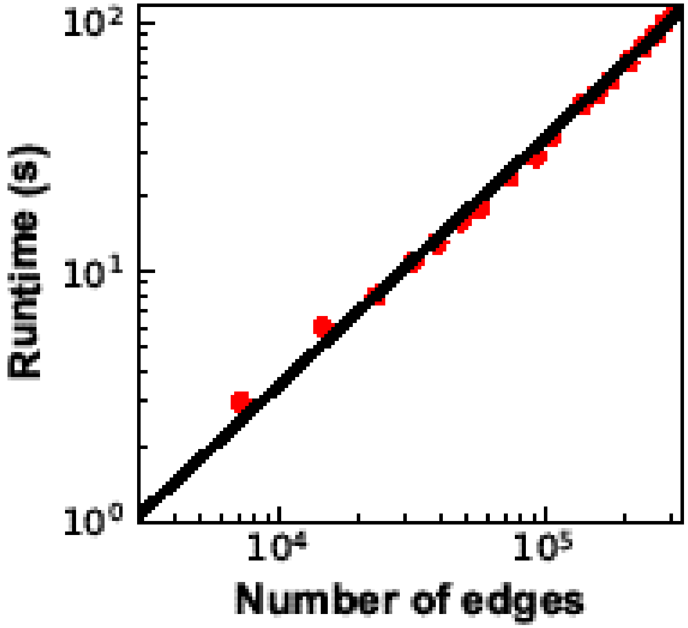




Near-linear time

Time complexity (using priority tree):

$$O|E'|\log|V|$$



Theoretical guarantee

Theorem: The subgraph (A,B) returned by ANTICO satisfies：

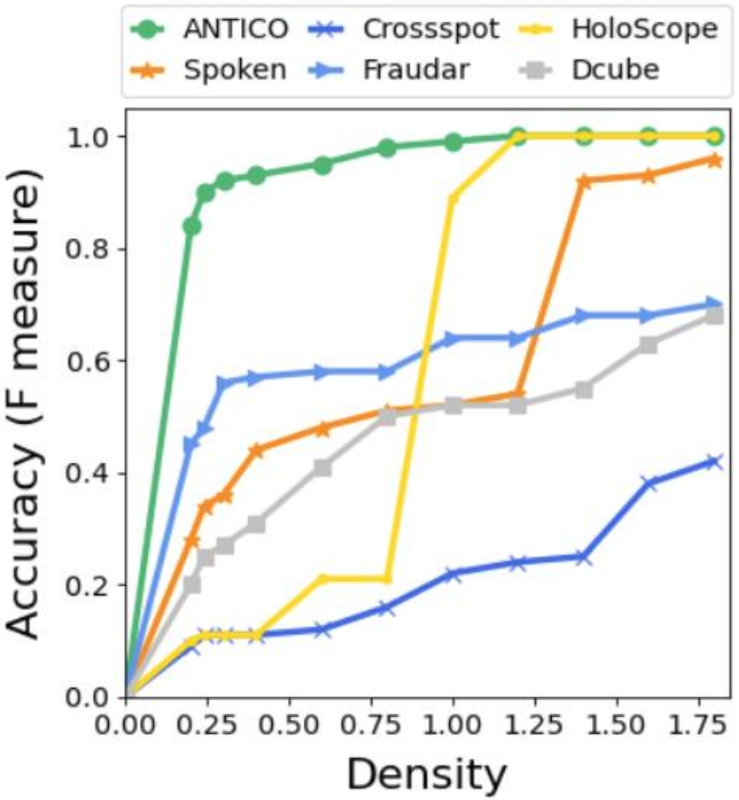
$$g(A \cup B) \geq \frac{1}{2} g(A_{opt} \cup B_{opt})$$

ANTICO subgraph

Optimal subgraph

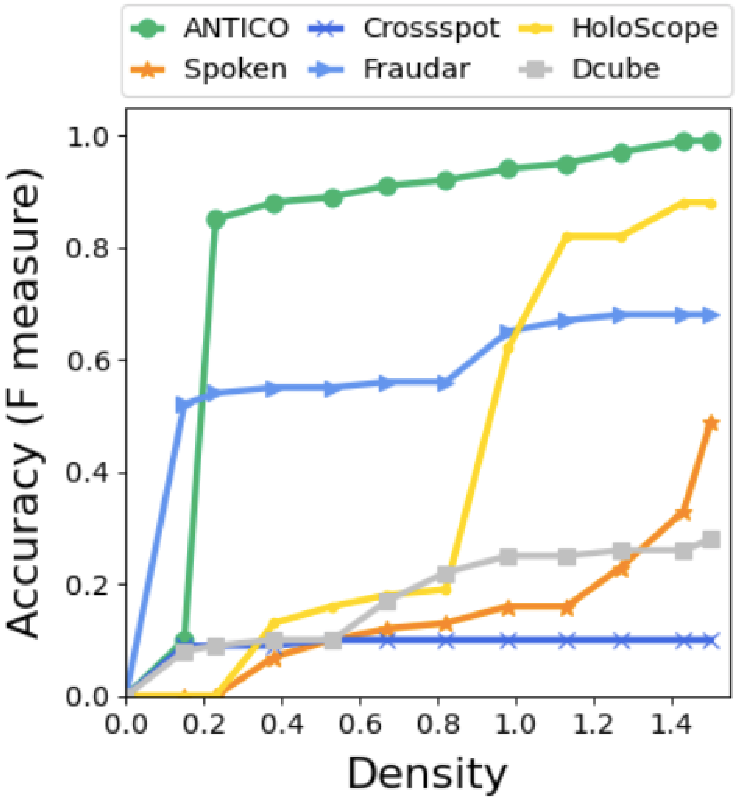
Evaluation: effectiveness study

- Synthetic settings
  - 20K users, 20K merchants
  - Same credit limit
  - Randomly add general transaction edges
- Other conditions
  - Cash-out connections
  - Credit utilization
  - Camouflage attacks



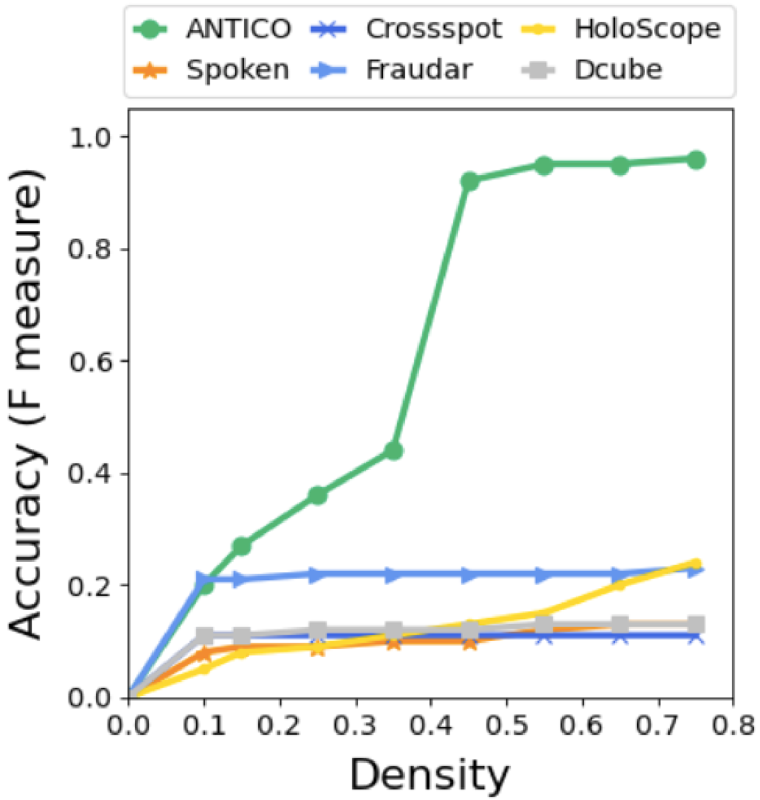
Evaluation: effectiveness study

- Synthetic settings
  - 20K users, 20K merchants
  - Same credit limit
  - Randomly add general transaction edges
- Other conditions
  - Cash-out connections
  - Credit utilization
  - Camouflage attacks.



Evaluation: effectiveness study

- Synthetic settings
  - 20K users, 20K merchants
  - Same credit limit
  - Randomly add general transaction edges
- Other conditions
  - Cash-out connections
  - Credit utilization
  - Camouflage attacks.



Evaluation: work in practice

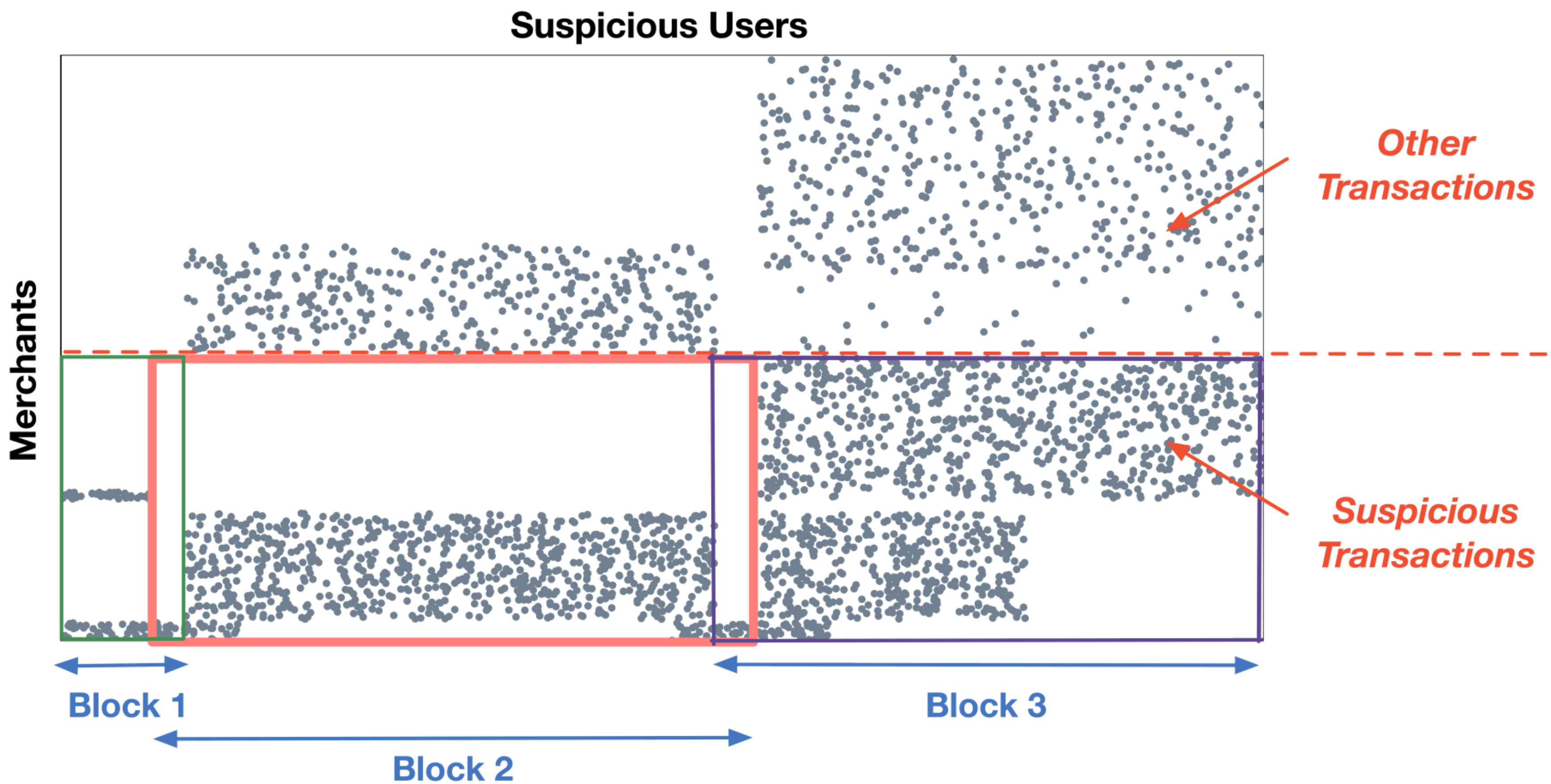
- Real-world graphs
  - 2M users, 1.1M merchants, and 45.8M transactions
  - Derived from top commercial bank
  - Separate regions and time spans

Datasets	#Nodes	#Edges	Density	Time Span
nx21co	(233K,155K)	4.29M	0.41	May.01,2021-Jun.30,2021
jl21co	(446K,243K)	8.89M	0.34	Jun.01,2021-Jul.31,2021
gz21co	(360K,265K)	7.34M	0.30	Jul.01,2021-Aug.31,2021
sx21co	(854K,302K)	19.62M	0.25	Jul.01,2021-Aug.31,2021
dl21co	(151K,134K)	5.69M	0.26	Jun.01,2021-Aug.31,2021

Evaluation: work in practice

Methods	nx21co		jl21co		gz21co		sx21co		ndl21co	
	AUC	K-S	AUC	K-S	AUC	K-S	AUC	K-S	AUC	K-S
Spoken	0.6426	0.2861	0.5778	0.1563	0.6313	0.2684	0.6715	0.3403	0.6359	0.2755
CrossSpot	0.5392	0.2303	0.5173	0.2464	0.6073	0.3033	0.5125	0.1964	0.5458	0.2532
Dcube	0.5542	0.2312	0.5655	0.2881	0.6684	0.4002	0.4389	0.2692	0.5556	0.3377
Fraudar	0.7324	0.4648	0.6606	0.3236	0.6899	0.3602	0.6399	0.3323	0.7185	0.3643
HoloScope	0.5876	0.1735	0.5558	0.1658	0.5955	0.1899	0.5711	0.1918	0.5643	0.2011
ANTICO	**0.7472**	**0.4944**	**0.7051**	**0.4099**	**0.7325**	**0.4652**	**0.7534**	**0.5068**	**0.7563**	**0.5127**

Posteriori analysis: multiple dense subgraphs



Important related works

- Application

Flowscope (AAAI 2020) shows the potential of graphed based anomaly detection on banking capital data.

- Framework

Fraudar (KDD 2016) provides a framework to extract dense subgraphs, integrated with weighting scheme, Charikar’s greedy strategy, and theoretical guarantees.

- Metrics

HoloScope (CIKM 2017) combines suspicious signals for fraud detection.

Summary

- Problem
  - Cash-out user detection
  - Traditional credit card services in bank
- Approach
  - Detect cash-out users in bipartite graph by exploiting suspicious signals
  - Theoretical bound and near-linear time
  - Well done in practice

Thank you

Yingsheng Ji
China Etek Service

Zheng Zhang
China Etek Service

Xinlei Tang
China Etek Service

Jiachen shen
China Etek Service

Xi Zhang
BUPT

Guangwen Yang
Tsinghua University

Special Thanks
TigerGraph

Project link:

https://github.com/transcope/antico