大数据分析

Big Graph Mining II

刘盛华

Outline

- Patterns in big graphs
- Dense block detection

多属性图

- homogeneous graphs
- bipartite graphs
- multi-attribute graphs
- streaming graphs
- more topics

Motivation: Why tensors?

- Tensor: N-D generalization of matrix:

KDD'09

	data	mining	classif.	tree	...
John	13	11	22	55	...
Peter	5	4	6	7	...
Mary	...	...	...	...	...
Nick	...	...	...	...	...
...	...	...	...	...	...

Motivation: Why tensors?

- Tensor: N-D generalization of matrix:

KDD'07

KDD'08

KDD'09

	data	mining	classif.	tree	...
John	13	11	22	55	...
Peter	5	4	6	7	...
Mary	...	...	...	...	...
Nick	...	...	...	...	...
...	...	...	...	...	...

Motivation: Why tensors?

- A: N-D generalization of matrix:

KDD'07

KDD'08

KDD'09

	data	mining	classif.	tree	...
John	13	11	22	55	...
Peter	5	4	6	7	...
Mary	...	...	...	...	...
Nick	...	...	...	...	...
...	...	...	...	...	...

Tensors are useful for 3 or more modes

Terminology: 'mode' (or 'aspect'):

术语

Mode#3

data	mining	classif.	tree	...
13	11	22	55	...
5	4	6	7	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

Mode#2

Mode (== aspect) #1

Notice

- 3rd mode does not need to be time
- we can have more than 3 modes

Dest. port

125

80

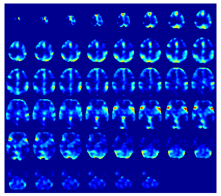
13	11	22	55	...
5	4	6	7	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

IP source

IP destination

Notice

- 3rd mode does not need to be time
- we can have more than 3 modes
  - Eg, fFMRI: x,y,z, time, person-id, task-id



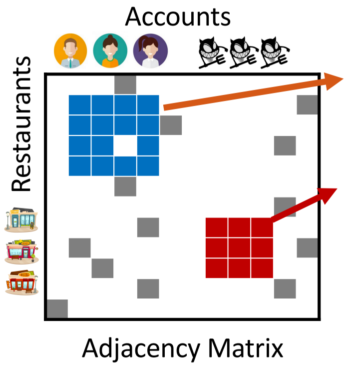
http://denlab.temple.edu/bidms/cgi-bin/browse.cgi

More tensors

- **Tensors are useful**
  - **web mining (TOPHITS)**
  - **environmental sensors**
  - **Intrusion detection (src, dst, time, dest-port)** 入侵检测
  - **Money laundering (src, mid, dest, time, money)**洗钱
  - **Social networks (src, dst, time, type-of-contact)**社交网络
  - **etc …**

Problem: Natural Dense Blocks

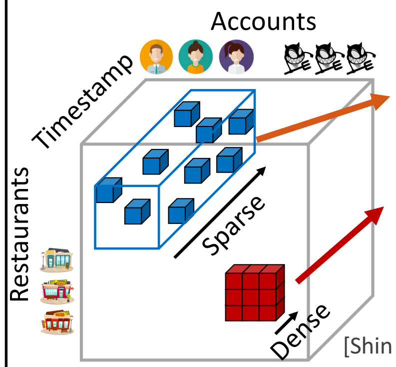
自然密集块



natural dense blocks
(core, community, etc.)

suspicious dense blocks
formed by fraudsters

Solution: Tensor Modeling



natural dense blocks: sparse on the time axis

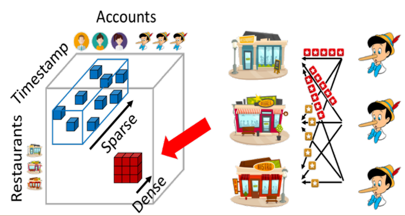
suspicious dense blocks: dense on the time axis (synchronous behavior)

[Shin et al. 2016, Jiang et al. 2015]

D-Cube : Dense-Block Detection in Terabyte-Scale Tensors [K. Shin+, wsdm2017]

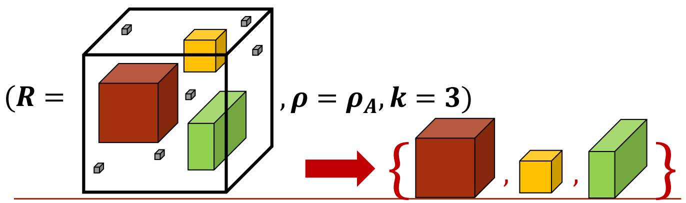
- **Tensor-based methods for fraud detection**
  - **M-Zoom, D-Cube, CrossSpot**

- Kijung Shin, Bryan Hooi, Jisu Kim, Christos Faloutsos
  - *Carnegie Mellon University*



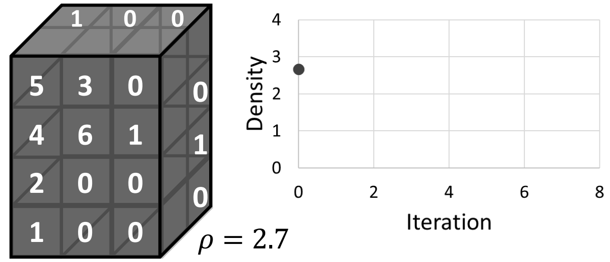
Problem Definition

- Given: (1) R: a tensor not fitting in memory,
  (2) ρ: a density measure
  (three measures are supported)
  (3) k: the number of blocks
- Find: k distinct dense blocks maximizing ρ



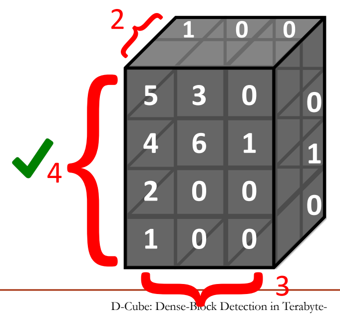
Single Dense Block Detection

- Search among $2^{(\#\ slices)}$ possible sub-tensors
- Step 1. search starts from the entire tensor



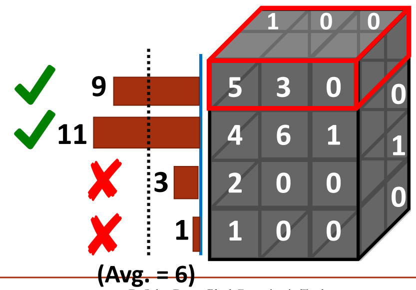
Single Dense Block Detection (cont.)

- Step 2. repeat until we reach the empty tensor
  - choose a mode with the most remaining slices
  - remove slices with mass at most the average mass



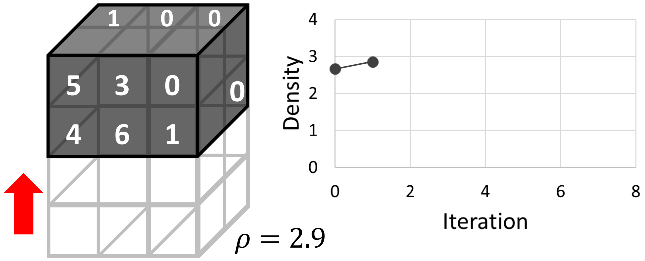
Single Dense Block Detection (cont.)

- Step 2. repeat until we reach the empty tensor
  - choose a mode with the most remaining slices
  - remove slices with mass at most the average mass



Single Dense Block Detection (cont.)

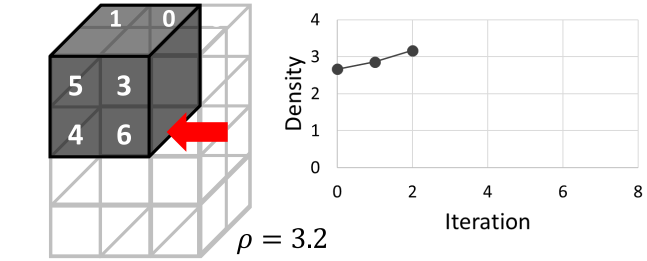
- Step 2. repeat until we reach the empty tensor
  - choose a mode with the most remaining slices
  - remove slices with mass at most the average mass



$\rho = 2.9$

Single Dense Block Detection (cont.)

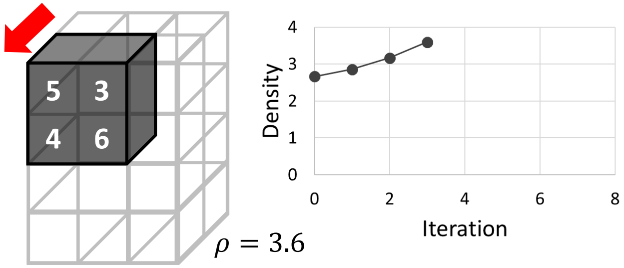
- Step 2. repeat until we reach the empty tensor
  - choose a mode with the most remaining slices
  - remove slices with mass at most the average mass



$\rho = 3.2$

Single Dense Block Detection (cont.)

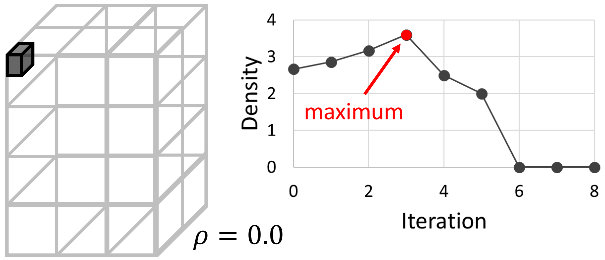
- Step 2. repeat until we reach the empty tensor
  - choose a mode with the most remaining slices
  - remove slices with mass at most the average mass



$\rho = 3.6$

Single Dense Block Detection (cont.)

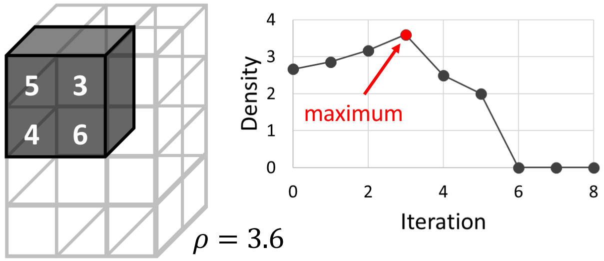
- Step 2. repeat until an empty tensor is left
  - choose a mode with the most remaining slices
  - remove slices with mass at most the average mass



$\rho = 0.0$

Single Dense Block Detection (cont.)

- Step 3. return the densest block so far



Accuracy Guarantee

- Theorem [Approximation Guarantee]

$$\boldsymbol{\rho_A(B,R) \geq \frac{1}{N}\rho_A(B^*,R)}$$

D-Cube Result　Input Tensor　Order　Densest Block

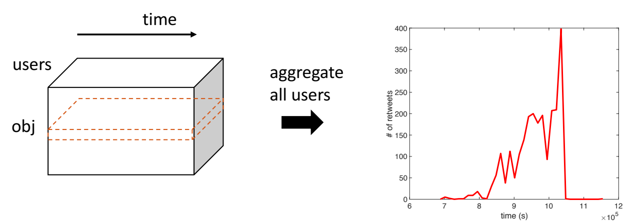
Any problem?

Temporal dim in Tensors

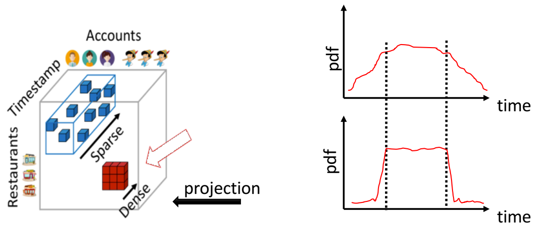
张量中的时间维

Graphs with temporal information

- view of time series



Graphs with temporal information

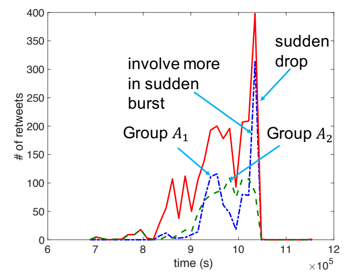


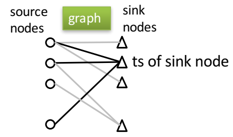
Tensor-based methods (M-Zoom, D-Cube, CrossSpot) detect the two cases as the same density level in temporal dim.

M-Zoom [K Shin, PKDD2017], D-Cube [K Shin, WSDM2017], CrossSpot [M Jiang ICDM2015]

Temporal spike: burst and drop are suspicious 时间尖峰:爆发和下降是可疑的

- The histogram (time series) of a sink node
  - users retweet a message in Sina Weibo data.

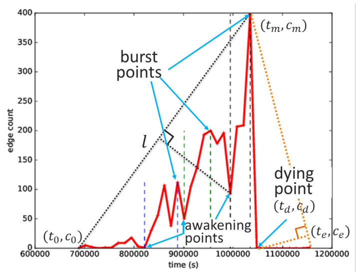




Group A_1 and A_2 has the same total # of retweets
A_1 is more suspicious than A_2

Detect spikes in time series of a sink node

- SB (Sleeping Beauty) defines burst and awakening point
- drop and dying point



awakening point: the point has the largest distance to l

$$t_a = \arg\max_{(c,t)\in\mathcal{T}, t<t_m} \frac{|(c_m - c_0)t - (t_m - t_0)c + t_m c_0 - c_m t_0|}{\sqrt{(c_m - c_0)^2 + (t_m - t_0)^2}}$$

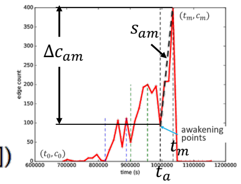
Detecting and identifying Sleeping Beauties in science [Ke et al, PNAS'15]

HoloScope considers time spikes

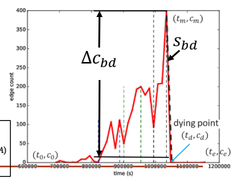
- multibust
  - $P(v_i|A) = q(\varphi_i),\ \varphi_i = \frac{\Phi(T_A)}{\Phi(T_U)}$

(t_a,t_m)

$$\Phi(T) = \sum \Delta c_{am} \cdot s_{am} \sum_{t\in T} 1(t \in [t_a, t_m])$$



- sudden drop
  - $f_A(v_i) = \sum_j \sigma_{ji} e_{ji}$
  - $\sigma_{ji} = \Delta c_{bd} \cdot s_{bd}$



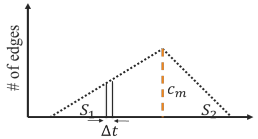
$$\max_A HS(A) := \mathbb{E}[D(A,B)] = \frac{1}{|A| + \sum_{k\in V} P(v_k|A)} \sum_{i\in V} f_A(v_i) P(v_i|A)$$

S. Liu, B. Hooi, C. Faloutsos, CIKM 2017; S. Liu, B. Hooi, C. Faloutsos, TKDE 2019

Time obstruction for fraudsters

骗子的时间障碍

- **Theorem 1**



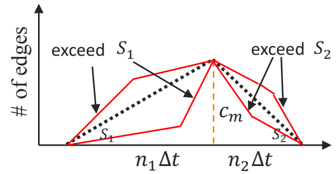
Let N be the number of edges that fraudsters want to create for an object.
If the fraudsters use time less than

$$\tau \geq \sqrt{\frac{2N\Delta t \cdot (S_1 + S_2)}{S_1 \cdot S_2}}$$

then they will be tracked by a suspicious burst or drop.

- Δt is the size of time bins,
- S_1 and S_2 are the slopes of normal rise and decline respectively 分别是上升下降的斜率

Proof



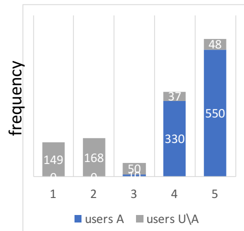
$$\frac{c_m}{n_1\Delta t} = S_1, \quad \frac{c_m}{n_2\Delta t} = S_2, \quad (n_1 + n_2)\cdot c_m = 2N'.$$

- n1 and n2 are # of time bins before and after the burst.
- N' is the total # of rating edges, and $N' \geq N$

$$\tau = (n_1 + n_2)\Delta t = \sqrt{\frac{2N'\Delta t(S_1 + S_2)}{S_1 \cdot S_2}} \geq \sqrt{\frac{2N\Delta t(S_1 + S_2)}{S_1 \cdot S_2}}$$

HoloScope: make holistic use of signals 使用整体信号

- **Topology awareness:** $\alpha_i = \frac{f_A(v_i)}{f_U(v_i)}$
- **Temporal-spike awareness:** $\varphi_i = \frac{\Phi(T_A)}{\Phi(T_U)}$
- **Rating deviation:** κ_i 评级偏差
  - κ_i=KL-divergence$(A, U\backslash A)$
  - $\kappa_i \leftarrow \kappa_i \cdot \min\{\frac{f_A(v_i)}{f_{U\backslash A}(v_i)}, \frac{f_{U\backslash A}(v_i)}{f_A(v_i)}\}$
- **Contrast susp of HS**
  - $P(v_i|A) = \boldsymbol{q}(\alpha_i)\boldsymbol{q}(\varphi_i)\boldsymbol{q}(\kappa_i) = b^{\alpha_i+\varphi_i+\kappa_i-3}$
  - **"joint probability"**



# of stars

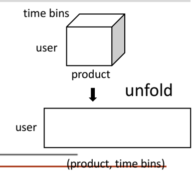
HoloScope: scalable algorithm

- **Main idea: feed small groups of users $\tilde{U}$ into *GreedyShaving* Procedure (previous HS alg.)**

Algorithm 4 *FastGreedy* Algorithm for Fraud detection.

Given bipartite multigraph $\mathcal{G}(U, V, E)$.
$\mathbb{L}$ = get first several left singular vectors
for all $L^{(k)} \in \mathbb{L}$ **do**
Rank source nodes U decreasingly on $L^{(k)}$
$\tilde{U}^{(k)}$ = truncate $u \in U$ when $L_u^{(k)} \leq \frac{1}{\sqrt{|U|}}$
GreedyShaving with initial $\tilde{U}^{(k)}$.
end for
return the best A^* with maximized objective $HS(A^*)$, and the rank of $v \in V$ by $f_{A^*}(v) \cdot P(v|A^*)$.

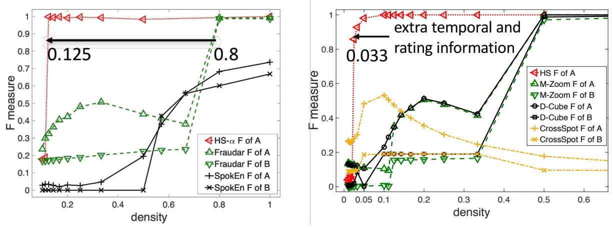
To consider temporal and #star information, we matricize tenor into a matrix



HoloScope: detect fraudsters with less density

- Mimic fraudsters to inject edges, time stamps and #stars, with different fraudulent density

BeerAdvocate Data



HS-α consider only topology (density)   HS consider all signals

2019/9/25   33

Outline

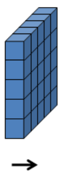
- Patterns in big graphs
- Dense block detection

- homogeneous graphs
- bipartite graphs
- multi-attribute graphs
- streaming graphs
- more topics

2019/9/25   34

Streaming graph

- graphs usually expand with time.



time

2019/9/25   35

How do we detect dense subgraphs in streaming graphs?

How do we even characterize these anomalies?

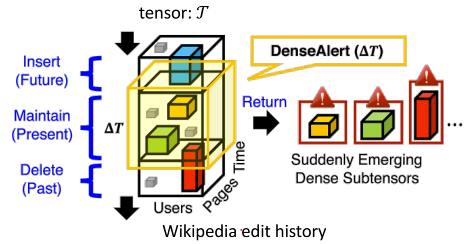
我们如何描述这些异常现象呢

2019/9/25   36

Problem

- **Given:**
  - a stream of triplets (***user*, *object*, *timestamp***).
- **Find:**
  - at ***each time step***, a group of users and objects who have ***densest*** edges
  - ***detect*** suspicious surges of density
    检测可疑的密度波动

DenseAlert [Shin, K+, KDD 2017]



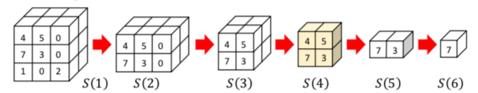
Given: a **stream** of changes, e.g. adding/removing edges in tensor $\mathcal{T}$.

- **Maintain:** a subtensor $\mathcal{T}(S)$, where S is set of slice indices.
- **to maximize:** density $\rho(\mathcal{T}(S))$

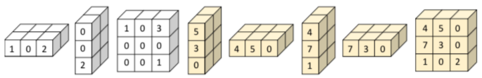
$$\rho(\mathcal{T}(S)) = \frac{sum(\mathcal{T}(S))}{|S|}$$

DenseAlert [Shin, K+, KDD 2017]

- **Greedy shaving and D-ordering**
  - **Greedy Shaving:** repeatedly removing a slice with the minimum mass in the remaining tensor

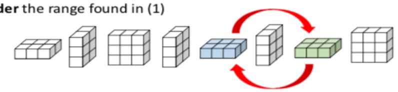


  - **D-ordering**: the order by which Greedy Shaving removes the slices



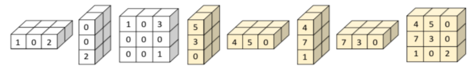
when tensor update:

(2) **Reorder** the range found in (1)

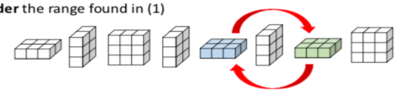


DenseAlert [Shin, K+, KDD 2017]

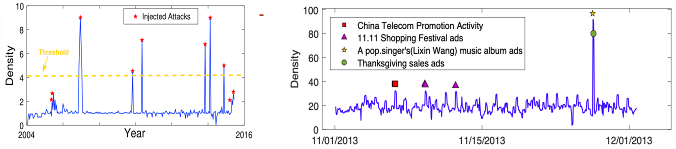
- **Greedy shaving and D-ordering**
  - **D-ordering**: the order by which Greedy Shaving removes the slices



(2) **Reorder** the range found in (1)

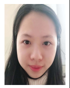


- design an updating operation for ***every single adding or removing*** edge.
- Running a bit ***slow***.



EigenPulse: Detecting Surges in Large Streaming Graph with Row Augmentation

Jiabao Zhang+, Shenghua Liu+, Wenjian Yu*, Wenjie Feng+ and Xueqi Cheng+

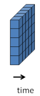


Model: Row-Augmented Matrix

行增广矩阵

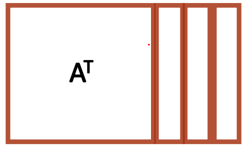
- Row-Augmented Matrix (RAM)
  - rows are augmented with same cols
  - *concatenate*

连接的



row: concat(object, ts)

col: user



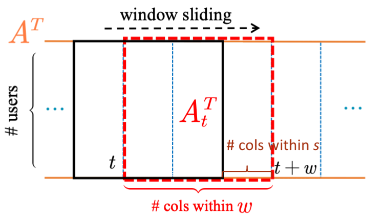
row augmented with timestamp growing!

  - equivalently, unfolding dynamic tensors (removing empty rows)
  - *e.g. same restaurant changes overtime, upgrading of a product, etc.*

同一家餐厅加班、升级产品等。

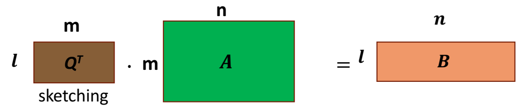
Model: Row-Augmented Matrix

- RAM with sliding window



stride window size

QB approximation



(1) $\mathbf{\Omega} = \mathrm{randn}(n,\ k+s)$
(2) $\boldsymbol{Q} = \mathrm{orth}(\boldsymbol{A\Omega})$
(3) $\boldsymbol{B} = \boldsymbol{Q}^T \boldsymbol{A}$

$A: m\times n, l = k+s$
$\Omega: n\times l$
$Q: \mathrm{m}\times l$
$B: l\times n$

$$\left.\begin{array}{l} B = \tilde{U}\tilde{\Sigma}\tilde{V}^T. \\ A \approx QB = Q\tilde{U}\tilde{\Sigma}\tilde{V}^T. \end{array}\right\} \Rightarrow U = Q\tilde{U}, \Sigma = \tilde{\Sigma}, V = \tilde{V}$$

胖高矩阵通过乘一个矮胖的矩阵变成了一个矮胖矩阵，再通过SVD分解，U =QU~

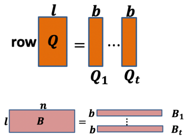
AugSVD: incrementally building Q

- generate matrices Q, B by G, H (streaming)

G: $m \times l$
H: $n \times l$

repeat
  Read rows $\mathbf{a}$ for next stride s in augmented A
  $\mathbf{g} = \mathbf{a}\mathbf{\Omega}$; $\mathbf{h} = \mathbf{a}^T\mathbf{g}$
  $glist.enqueue(\mathbf{g})$; $hlist.enqueue(\mathbf{h})$
until the elements in $glist$ corresponds to a window w
for all $\mathbf{g}$ in $glist$, $\mathbf{h}$ in $hlist$ **do**
  $\mathbf{G} = [\mathbf{G}, \mathbf{g}]$; $\mathbf{H} = \mathbf{H} + \mathbf{h}$
end for
 $\mathbf{Q} = [\,]$; $\mathbf{B} = [\,]$
for $i = 1, 2, \cdots, t$ **do**
  $\mathbf{\Omega}_i = \mathbf{\Omega}(:, (i-1)b+1 : ib)$; $\mathbf{Y}_i = \mathbf{G}(:, (i-1)b+1 : ib) - \mathbf{Q}(\mathbf{B}\mathbf{\Omega}_i)$
  $[\mathbf{Q}_i, \mathbf{R}_i] = qr(\mathbf{Y}_i)$
  $[\mathbf{Q}_i, \widetilde{\mathbf{R}}_i] = qr(\mathbf{Q}_i - \mathbf{Q}(\mathbf{Q}^T\mathbf{Q}_i))$
  $\mathbf{R}_i = \widetilde{\mathbf{R}}_i\mathbf{R}_i$
  $\mathbf{B}_i = \mathbf{R}_i^{-T}(\mathbf{H}(:, (i-1)b+1 : ib)^T - \mathbf{Y}_i^T\mathbf{Q}\mathbf{B} - \mathbf{\Omega}_i^T\mathbf{B}^T\mathbf{B})$
  $\mathbf{Q} = [\mathbf{Q}, \mathbf{Q}_i]$; $\mathbf{B} = [\mathbf{B}^T, \mathbf{B}_i^T]^T$
end for

Y_i: $m \times b$, Q_i: $m \times b$



AugSVD: incrementally building Q

- generate matrices Q, B by G, H (streaming)

G: $m \times l$
H: $n \times l$

repeat
  Read rows $\mathbf{a}$ for next stride s in augmented A
  $\mathbf{g} = \mathbf{a}\mathbf{\Omega}$; $\mathbf{h} = \mathbf{a}^T\mathbf{g}$
  $glist.enqueue(\mathbf{g})$; $hlist.enqueue(\mathbf{h})$
until the elements in $glist$ corresponds to a window w
for all $\mathbf{g}$ in $glist$, $\mathbf{h}$ in $hlist$ **do**
  $\mathbf{G} = [\mathbf{G}, \mathbf{g}]$; $\mathbf{H} = \mathbf{H} + \mathbf{h}$
end for

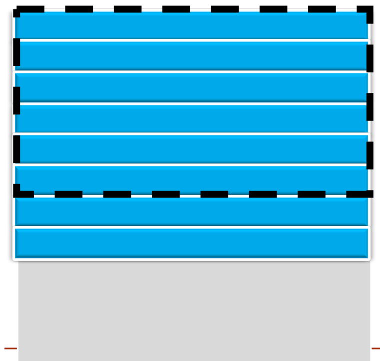
了解

- with Q and B

$[\widetilde{\mathbf{U}}, \mathbf{S}, \mathbf{V}] = svd(\mathbf{B})$
$\mathbf{U} = \mathbf{Q}\widetilde{\mathbf{U}}$
$\mathbf{U} = \mathbf{U}(:, 1:k)$; $\mathbf{V} = \mathbf{V}(:, 1:k)$; $\mathbf{S} = \mathbf{S}(1:k, 1:k)$

AugSVD

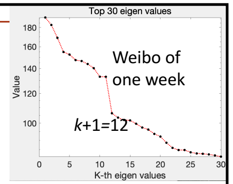
Combine *Sliding Window*, Change matrices G, H generation.



(g0,h0)
(g1,h1)
(g2,h2)
(g3,h3)
(g4,h4)
(g5,h5)
(g6,h6)
(g7,h7)

(glist,hlist) => G, H, Q, B

Theoretical analysis



- Theorem:
  - Let approx error rate be $\varepsilon_i = (\sigma_i - \hat{\sigma}_i)/\sigma_i$, then

$$|\varepsilon_i| \lesssim 2\frac{\sigma_{k+1}}{\sigma_i} + \frac{e\sqrt{2k+1}}{k}\left(\sum_{j=k+1}^{\min(m,n)}\left(\frac{\sigma_j}{\sigma_i}\right)^2\right)^{1/2}, \quad i = 1, \ldots, k$$

less than approximately.
$(k+1)$-th original singular value
k is the truncated length

  - Error is small when σ is ***highly skewed***

Theoretical analysis: proof

skip details

AugSVD algorithm inherits the *SinglePassPCA* theoretical error bound :

$$\mathbb{E}\|\mathbf{A}-\mathbf{Q}\mathbf{Q}^T\mathbf{A}\| \le \left(1+\sqrt{\frac{k}{s-1}}\right)\sigma_{k+1} + \frac{e\sqrt{k+s}}{s}\left(\sum_{j=k+1}^{\min(m,n)}\sigma_j^2\right)^{1/2} \quad (1)$$

where $\mathbb{E}$ denotes expectation, $s = l - k$. If choosing $s = k+1$, we have

$$\mathbb{E}\|\mathbf{A}-\hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^T\| \le 2\sigma_{k+1} + \frac{e\sqrt{2k+1}}{k}\left(\sum_{j=k+1}^{\min(m,n)}\sigma_j^2\right)^{1/2} \quad (2)$$

2019/9/25 49

Theoretical analysis: proof

skip details

Applying a rough analysis, we have:

$$\mathbb{E}\max_{i=1,\dots,k}|\sigma_i-\hat{\sigma_i}| = \mathbb{E}\|\mathbf{\Sigma}-\hat{\mathbf{\Sigma}}\| \le 2\sigma_{k+1} + \frac{e\sqrt{2k+1}}{k}\left(\sum_{j=k+1}^{\min(m,n)}\sigma_j^2\right)^{1/2} \quad (3)$$

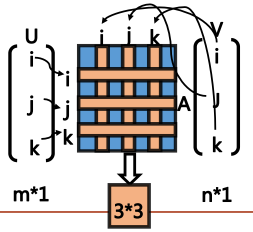
$$|\sigma_i-\hat{\sigma_i}| \lesssim 2\sigma_{k+1} + \frac{e\sqrt{2k+1}}{k}\left(\sum_{j=k+1}^{\min(m,n)}\sigma_j^2\right)^{1/2}, \ i=1,\dots,k \quad (4)$$

2019/9/25 50

EigenPulse

- At every time stride,
  - Choose dense blocks based on the first several singular vectors.
    - above average

$$\tau_u = \frac{1}{\sqrt{m_t}}; \quad \tau_v = \frac{1}{\sqrt{n}}$$



2019/9/25 51

EigenPulse

- At every time stride,
  - Choose dense blocks based on the first several singular vectors.
  - [Optional] dense block detection in small selected blocks
    - use Fraudar and HoloScope (HS-α)
  - Calculate density

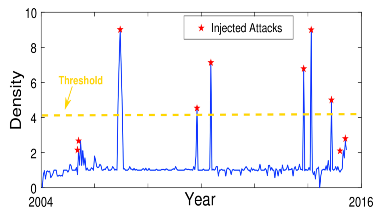
$$D_t(rowset, colset) = \frac{\sum_{i\in rowset}\sum_{j\in colset}\mathbf{A}_t(i,j)}{|rowset|+|colset|}$$

- plotting EigenPulse, and detecting anomalies.

2019/9/25 52

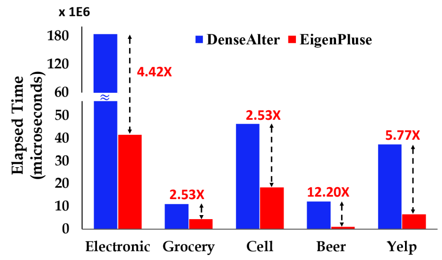
EigenPluse: detect injection accurately and instantly

- Injected 10 dense blocks for *Yelp* dataset
  - 686k x 85.3k, 2.68 M



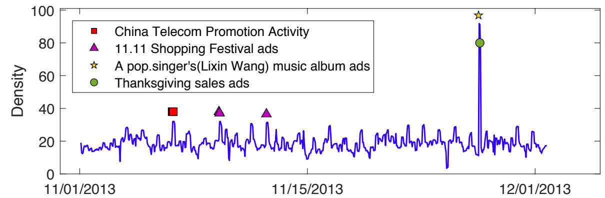
$threshold = 99.7$ percentile

EigenPluse: run faster than state-of-art methods



EigenPulse achieves more than 2.53× speed up.

EigenPluse: detect anomalous surges on Microblog data



set $w = 2h$ and $s = 1h$ on Sina Weibo data

Detected Blocks in Sina Weibo

Message Topic	Size	Time range	#Edges
China Telecom Promotion Activity	39 × 57	6:00~ 8:00, Nov 7	2,004
	78 × 58	7:00~9:00, Nov 7	4,051
	151 × 119	8:00~10:00, Nov 7	8,295
11.11 Shopping Festival ads	201 × 139	6:00~8:00, Nov 10	7,012
	196 × 111	7:00~9:00, Nov 10	9,668
	126 × 93	8:00~10:00, Nov 13	638
A pop. singer's (Lixin Wang) music album ads.	**7 × 8**	22:00~24:00, Nov 26	**953**
Thanksgiving sale ads	26 × 36	23:00, Nov 26~1:00, Nov 27	629
	43 × 34	1:00~3:00, Nov 27	263

$\approx 100\ rt/user$

$\approx 140\ rt/user$

7 users × 8 messages, 953 edges in 2 hours means every user retweeted more than once per minute.

More detailed stories and messages

11.11 Shopping Festival ads

#1111淘宝病# menscolor梵可-源自荷兰天然有机男士化妆品品牌！抢双11天猫旗舰店半价体验机会→ http://t.cn/zj3XsvO 关注 @menscolor梵可

【有奖转发】双11来你，您想告别单身吗？您想领取万元大奖吗？只要关注@绅度表 转发本条微博，并且@三个好友就有机会把奖品带回去。

#小易服饰 鞋服要大牌#闹开了，要狠的，玩疯的，搞大的，小易服饰超级0元购火爆开启。活动时间：11.8—11.11。详情点击：http://t.cn/z

A pop singer’s music album ads

曾经让王栎鑫骄傲的集体啊，他还是以你们为骄傲，请回来吧，我们一起再让他骄傲下去，只是那个叫做王栎鑫的少年！ 他是我们的信仰，而我们，却是他的依靠啊

#王栎鑫新专降落伞# @王栎鑫 新唱片《降落伞》中豪华打造的12款月历封面大发布~片中，糊糊造型百变，多款有型有格调的年轻硬汉造型，充满质感和时尚感。[心]12款

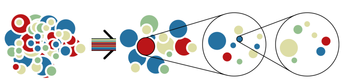
Outline

- Patterns in big graphs
- **Dense block detection**

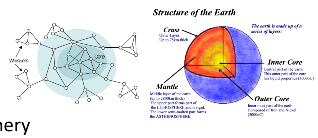
- homogeneous graphs
- bipartite graphs
- multi-attribute graphs
- streaming graphs
- more topics

Hierarchical Structure

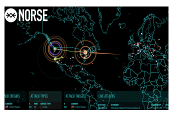
- **Static and Dynamic scenario**



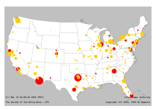
Community & Core–Periphery
[Daniel Edler et al, 2017], [jure etal 2008]



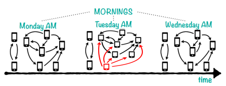
Earth structure



Cyberthreat
Real-Time Map



Witty Worm spread

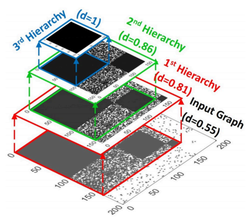


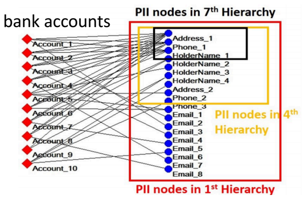
Evolving patterns

HiDDen: hierarchical dense subgraph

隐藏:层次密集子图

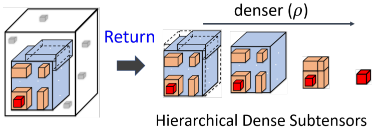
- **Dense measure in the subgraph, simultaneously**
  - **(1) maximize the number of edges**
  - **(2) minimize the number of the missing edges**





PII: personal identity information

S Zhang+, HiDDen: hierarchical dense subgraph detection with application to financial fraud detection. In SDM’17

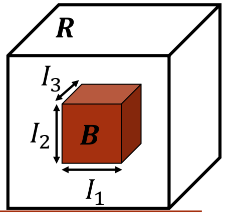


CatchCore: Hierarchical Dense Subtensor Detection

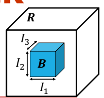
Wenjie Feng Shenghua Liu Xueqi Cheng

Terminologies

- **Assume a block (subtensor) B in a 3-way tensor $\boldsymbol{R}$**
  - ***Inclusion*** **:** $\boldsymbol{B} \preccurlyeq \boldsymbol{R}$
  - $\boldsymbol{M_B} = Mass(\boldsymbol{B})$**: (sum of entries)**
  - $\boldsymbol{V_B} = Volume(\boldsymbol{B})$**:** $I_1 \times I_2 \times I_3$
  - $\boldsymbol{D_B} = cadinalties(\boldsymbol{B})$**:** $I_1 + I_2 + I_3$



A general metric for dense block detection



- **Arithmetic Avg. Degree**
  - $f_B = M_B / (\frac{1}{N}\sum_N |B_n|)$ $\Rightarrow \ln f_B = \ln M_B - \ln(\frac{1}{N}\sum_N |B_n|)$
- **Geometric Avg. Degree**
  - $f_B = M_B / (\prod_N |B_n|)^{1/N}$ $\Rightarrow \ln f_B = \ln M_B - \frac{1}{N}\ln\prod_N |B_n|$
- **Volume Density**
  - $f_B = M_B / \prod_N |B_n|$ $\Rightarrow \ln f_B = \ln M_B - \ln\prod_N |B_n|$
- **A general metric**
  - $f_B = g(M_B) - \phi \cdot h(S_B)$
  - where S_B can be average # of nodes or volume

Hierarchical Dense Subtensors

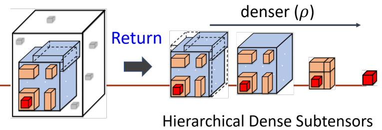
分层密度子tensor

- **Given: (1).** $\boldsymbol{R}$**: N-way tensor**
  (2). η**: density ratio**
  (3). K**: the maximum number of hierarchies**
- **Find:** $\boldsymbol{r}$ **(<** $\boldsymbol{K}$**) significant hierarchical dense block** $\{X^1, X^2, \ldots, X^r\}$

$$\max_{X^1,\cdots,X^K} \sum_{h=1}^{K} f_{X^h}$$

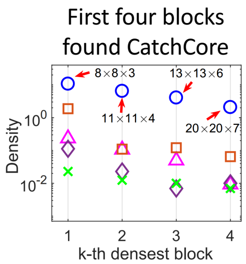
s.t. volume density $\rho_{h+1} \geq \eta \cdot \rho_h, \quad h = 1, \cdots, K$

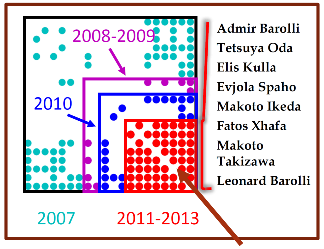
$X^1 \geq \cdots \geq X^K$



CatchCore detects evolving co-author research community

- **DBLP co-authorship network**
  - **co-author research community (20 users)**





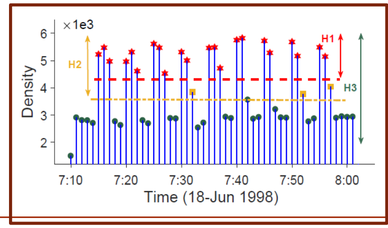
Avg.=10.74

CatchCore D-Cube M-Zoom CrossSpot CPD

CatchCore detects seasonal Neptune attacks

- **Darpa TCP Dumps**
  - **Neptune attack: a forged src to a network host**

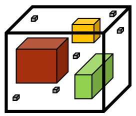
First three blocks found by CatchCore: H1, H2 and H3

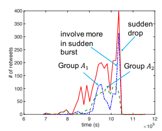


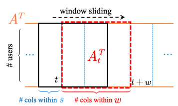
periodicity: ~5 minutes.

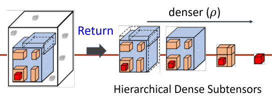
Take away

- **Multi-attributed graphs ==> Tensors**
- **Dense block detection in Tensor**
  - **D-cube**
- **Graph with temporal information**
  - **HoloScope**
- **Streaming Graph**
  - **DenseStream, EigenPulse**
- **Hierarchical dense blocks**









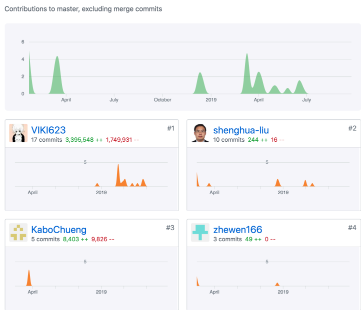
SparTAn2——recruitment

```
import spartan as st

# set the computing engine
st.config(st.engine.SINGLEMACHINE)

# load graph data, data stores as edgelist in database
data = st.loadTensor(name="yelp", path="~/Data/", col_ids = ["uid", "oid", "rating"], col_types = [int, int, int])

# create a anomaly detection model
hsmodel = st.anomaly_detection.create(data, st.ad_policy.HOLOSCOPE)

 # run the model
hsmodel.run(k=3)

# show the results
hsmodel.showResults()

A, B = hsmodel.nodes(n=0)
g = st.subgraph(data, A, B)
```

Eigen decomposition
Dense block detection
Anomaly detection
… …

SparTAn2——recruitment

- Github
  https://github.com/shenghua-liu/spartan2
- email
  - liushenghua@ict.ac.cn
- Your credits
  - job
  - career



Questions?