KDD2018 ODD v5.0

EagleMine:

vision-guided large graph mining

Wenjie Feng[+] Shenghua Liu[+] Christos Faloutsos[*]

Bryan Hooi[*] Huawei Shen[+] Xueqi Cheng[+]

[+]*Institute of Computing Technology ICT, CAS*

[*]*Computer Science Dept., Carnegie Mellon University*

Vision-Knowledge based Diagnosis

Vision-Knowledge based Diagnosis

Human Healthcare

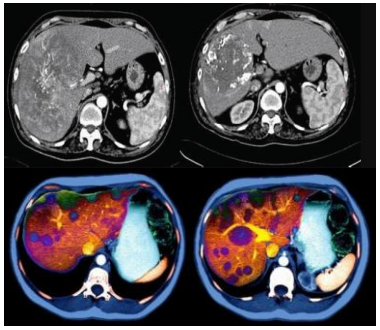
Vision-Knowledge based Diagnosis

Human Healthcare

Vision-Knowledge based Diagnosis

Human Healthcare

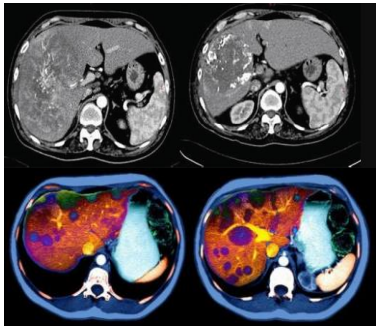
Chest CT scan

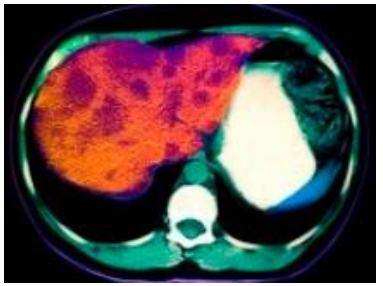


Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



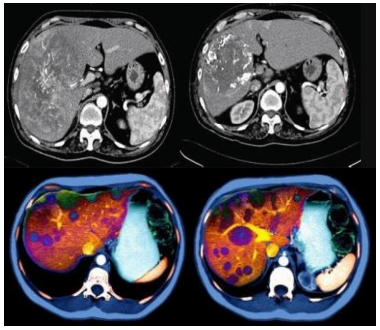


[Liver Cancer]

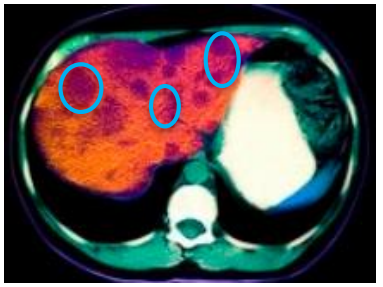
Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



Cancer / Tumor

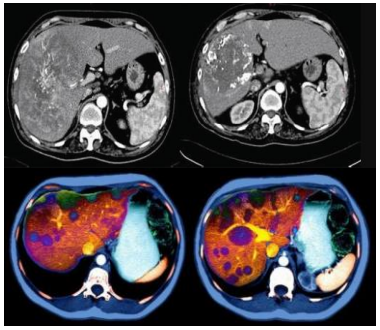


[Liver Cancer]

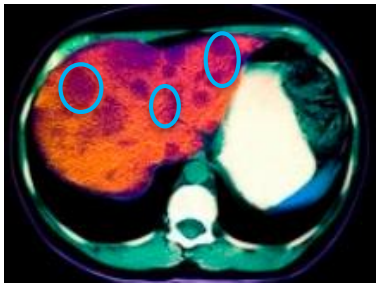
Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



Cancer / Tumor



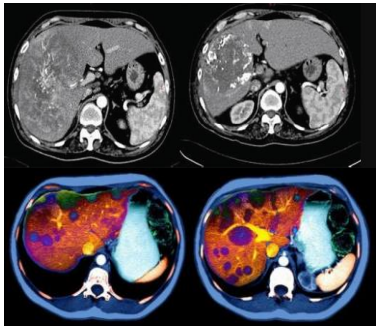
[Liver Cancer]

How about very graph diagnosis?

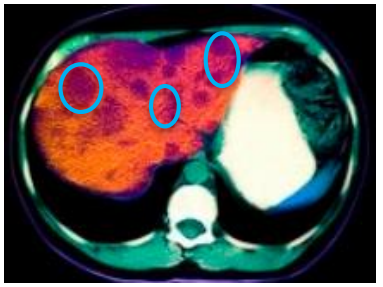
Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



Cancer / Tumor



[Liver Cancer]

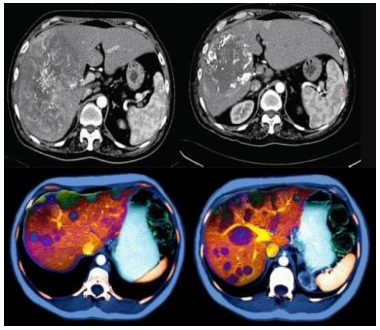
How about very graph diagnosis?

Social Network [Facebook]

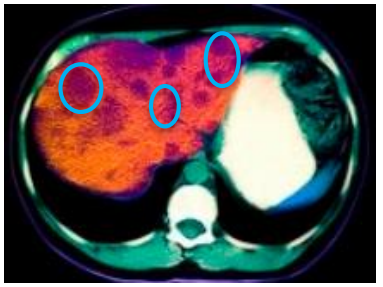
Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



Cancer / Tumor

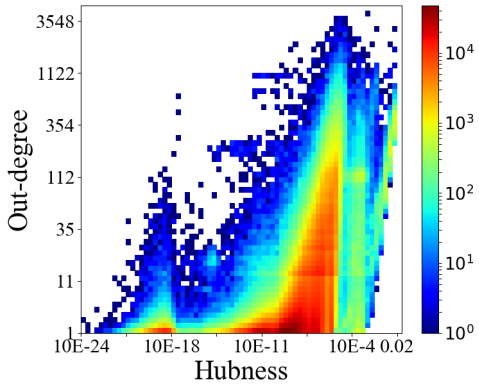


[Liver Cancer]

How about very graph diagnosis?

Social Network [Facebook]

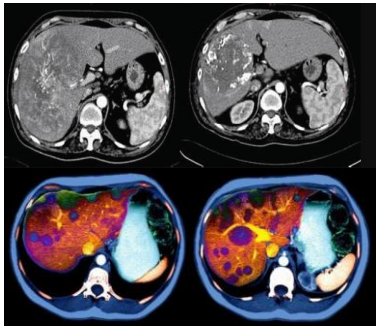
View spaces



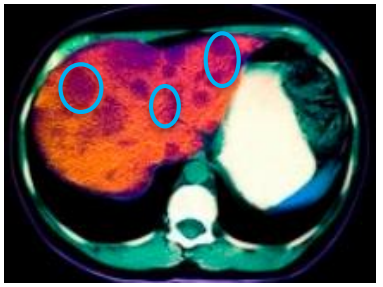
Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



Cancer / Tumor

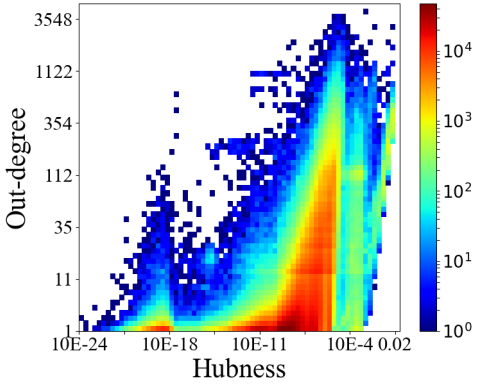


[Liver Cancer]

How about very graph diagnosis?

Social Network [Facebook]

View spaces

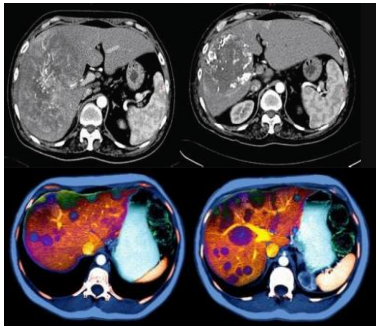


Anomaly Patterns

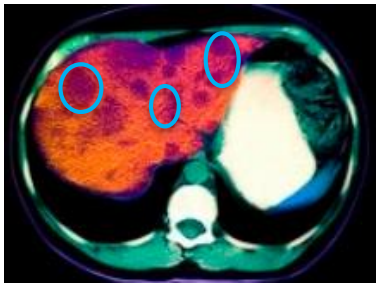
Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



Cancer / Tumor

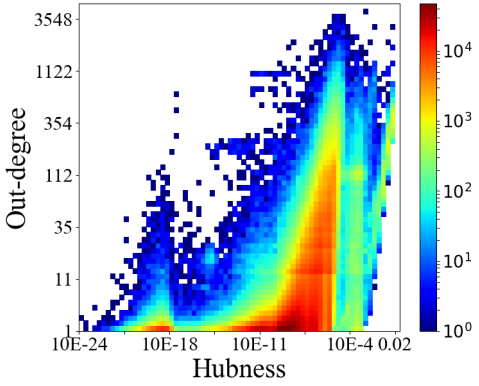


[Liver Cancer]

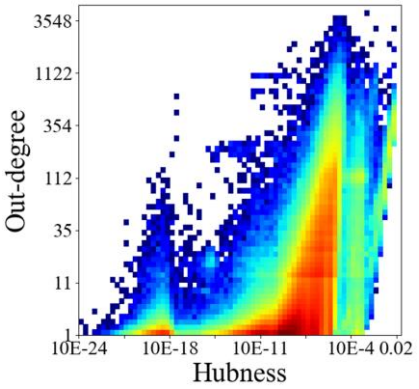
How about very graph diagnosis?

Social Network [Facebook]

View spaces



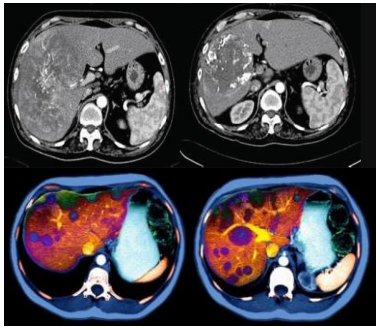
Anomaly Patterns



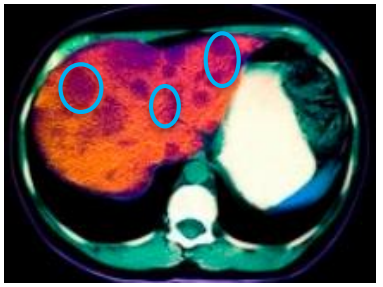
Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



Cancer / Tumor

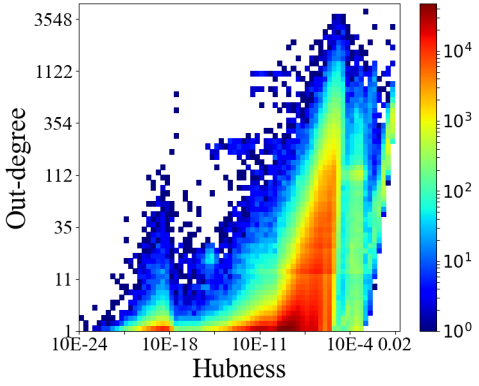


[Liver Cancer]

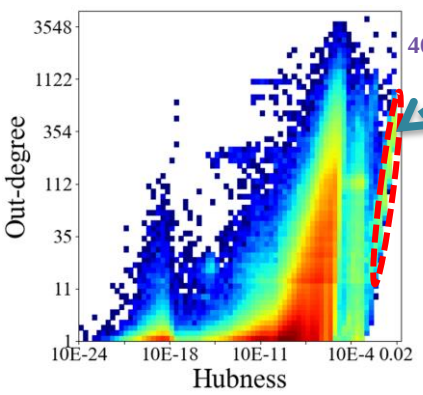
How about very graph diagnosis?

Social Network [Facebook]

View spaces



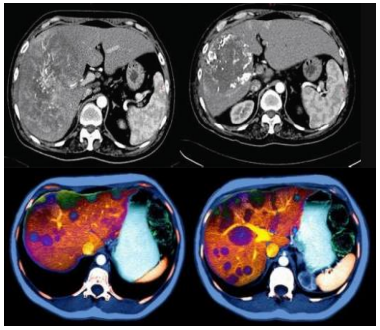
Anomaly Patterns



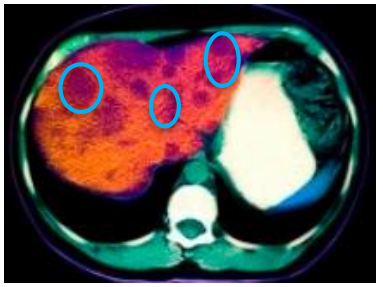
Vision-Knowledge based Diagnosis

Human Healthcare

Chest CT scan



Cancer / Tumor

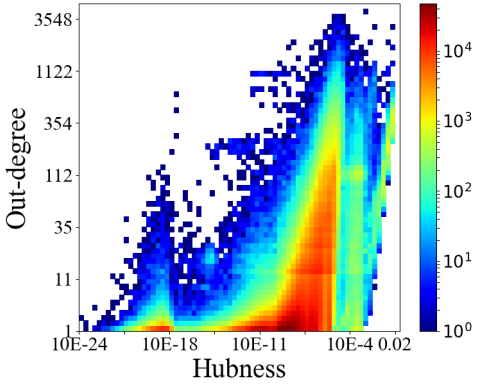


[Liver Cancer]

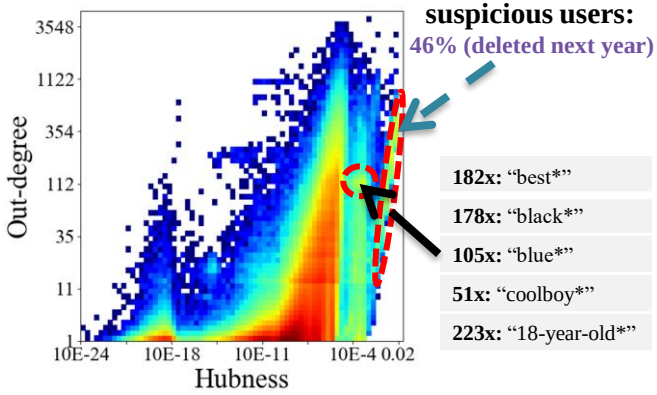
How about very graph diagnosis?

Social Network [Facebook]

View spaces



Anomaly Patterns

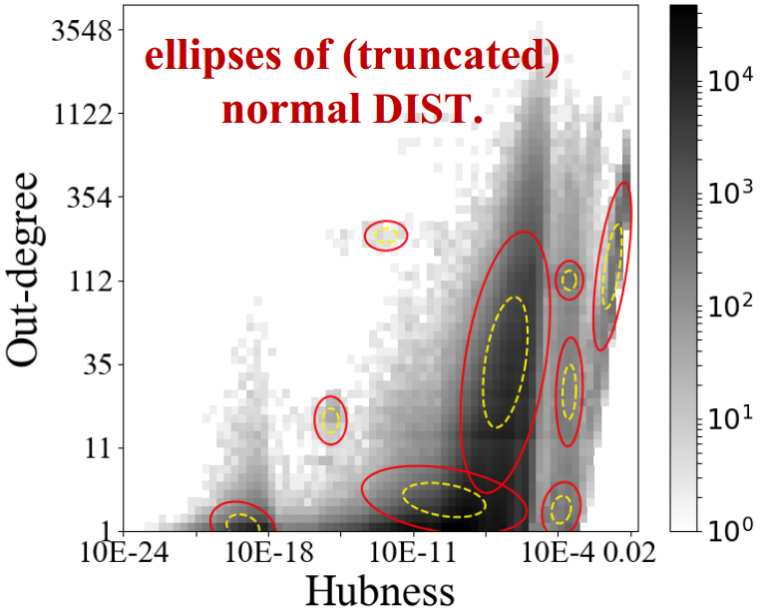


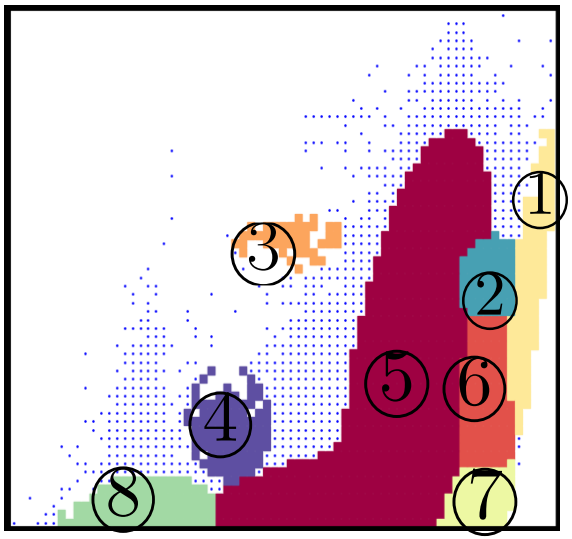
Outline

- Introduction
- Overview <<
- Proposed Method: **EagleMine**
- Experiments
- Conclusion

EagleMine: vision-guided large graph mining

- Our **EagleMine**: Automatically summarize the node distribution in graph correlated feature spaces.

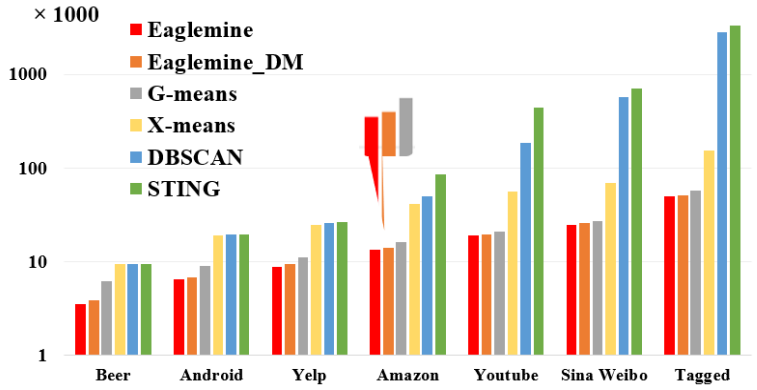




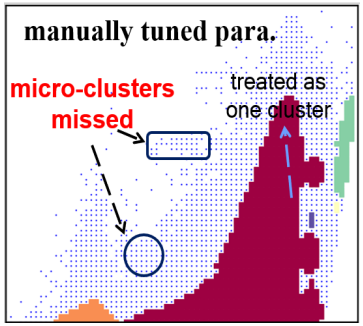
Detected micro-clusters

EagleMine: vision-guided large graph mining

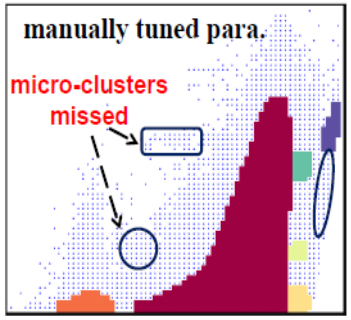
- Our **EagleMine**: Effectively, concisely describe and detect node groups, and get visually sense-making results.



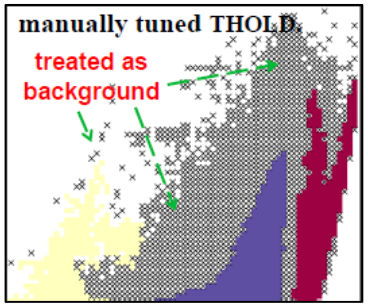
EagleMine achieves shortest MDL.



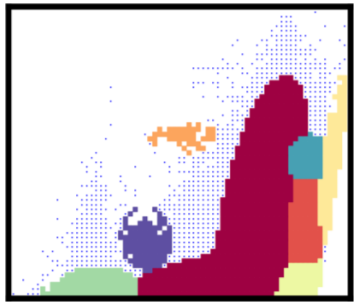
STING



DBSCAN



Watershed



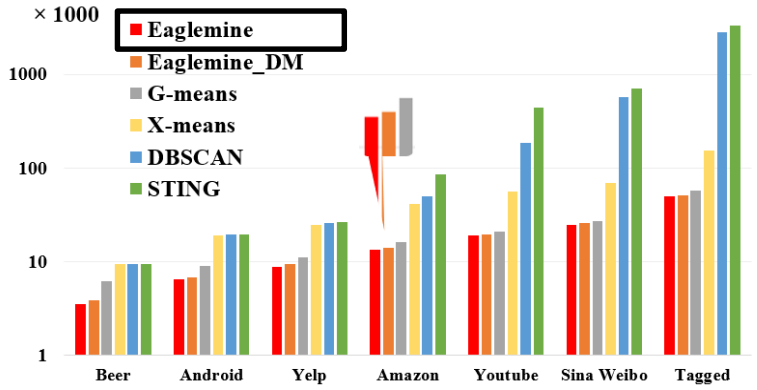
EagleMine

Quantitative

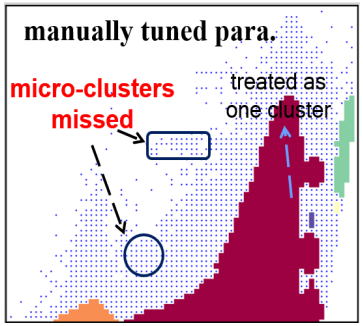
Qualitative

EagleMine: vision-guided large graph mining

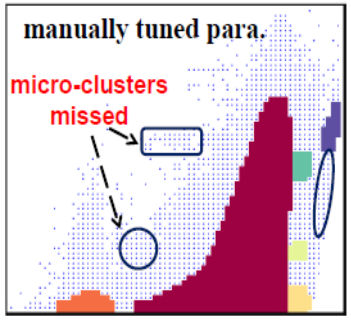
- Our **EagleMine**: Effectively, concisely describe and detect node groups, and get visually sense-making results.



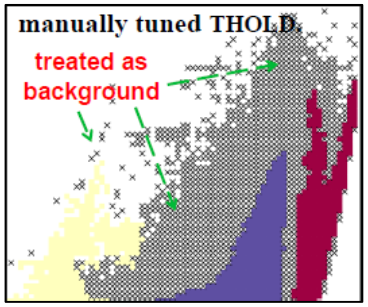
EagleMine achieves shortest MDL.



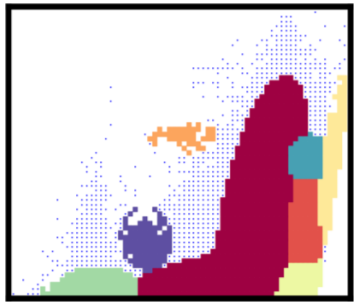
STING



DBSCAN



Watershed



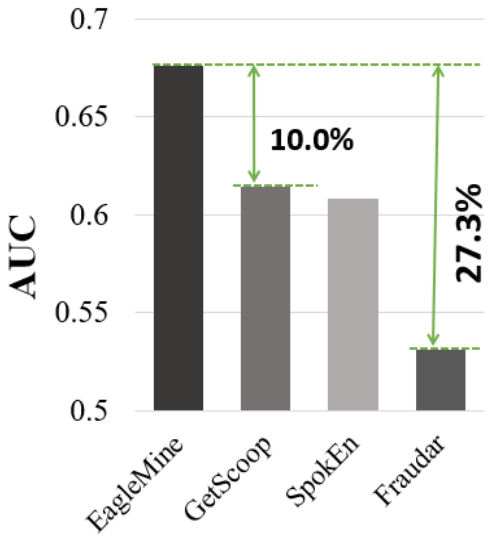
EagleMine

Quantitative

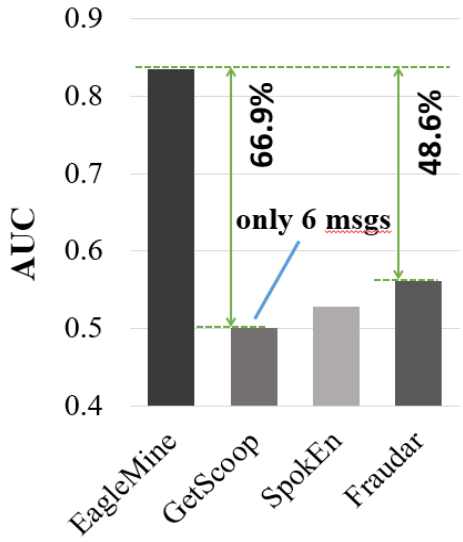
Qualitative

EagleMine: vision-guided large graph mining

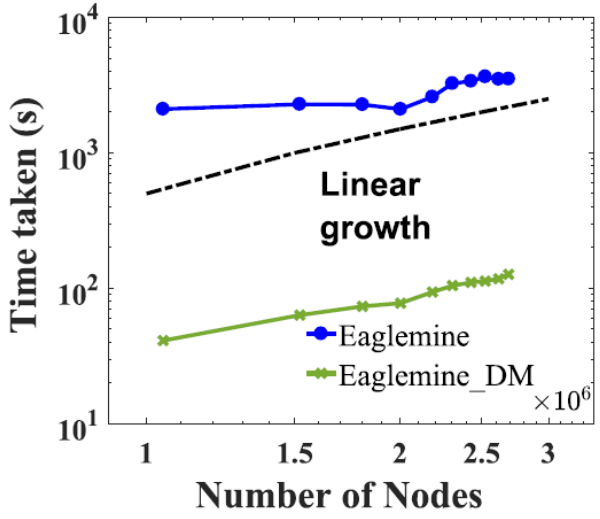
- Our **EagleMine**: Detect explainable anomaly pattern more effective and scalable for running linearly in # of nodes .



Suspicious Users



Suspicious Msgs



Time vs. # node of graph

Anomaly detection

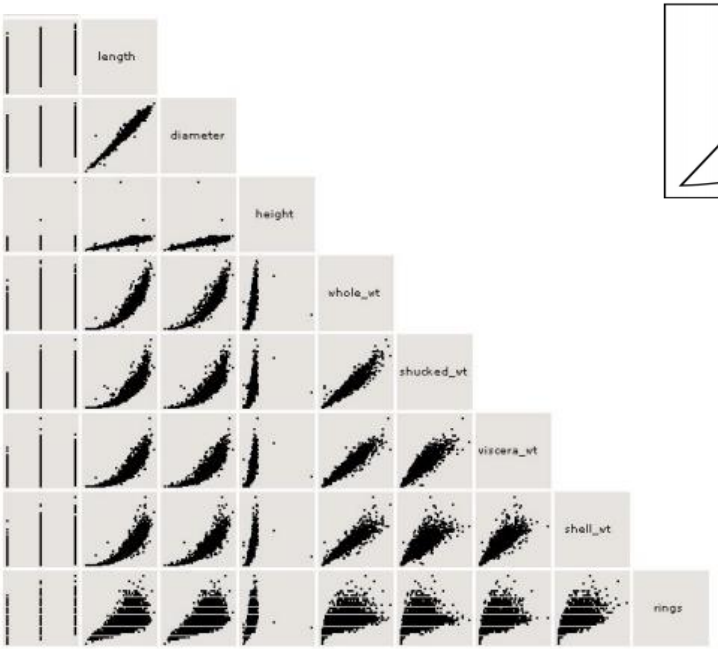
Scalability

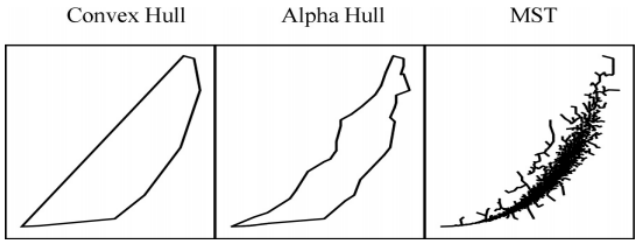
Outline

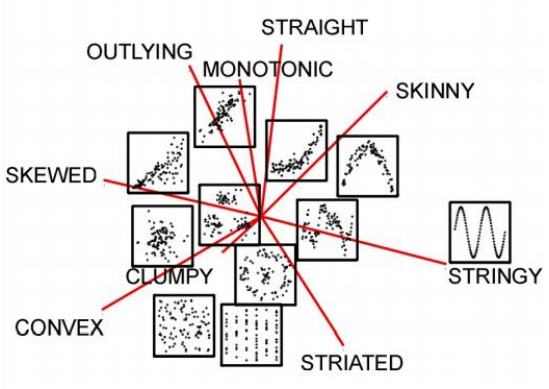
- Introduction
- Overview
- Proposed Method: **EagleMine <<**
  - Problem definition **<<**
  - Algorithm
- Experiments
- Conclusion

Graph-Theoretic Scagnostics

- graph-theoretic summaries of high-dimensional scattered point data.

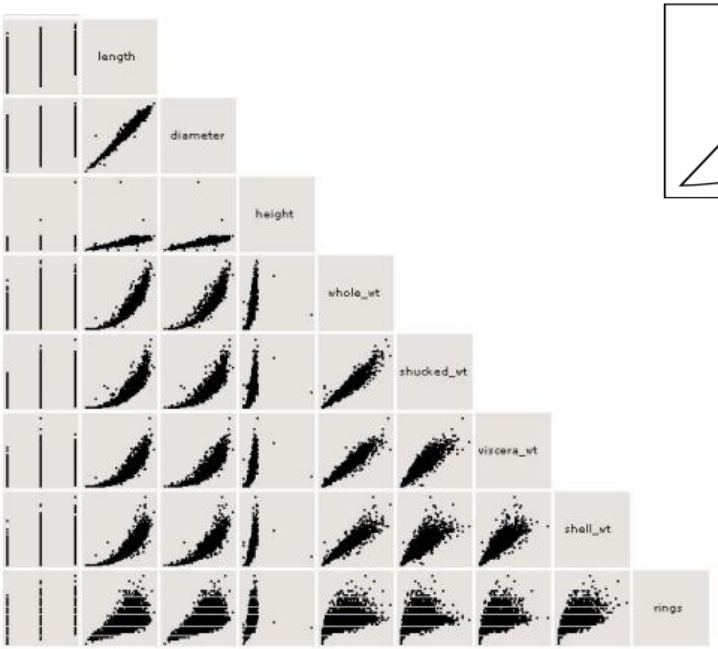


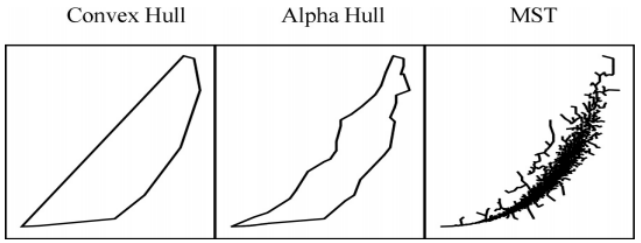


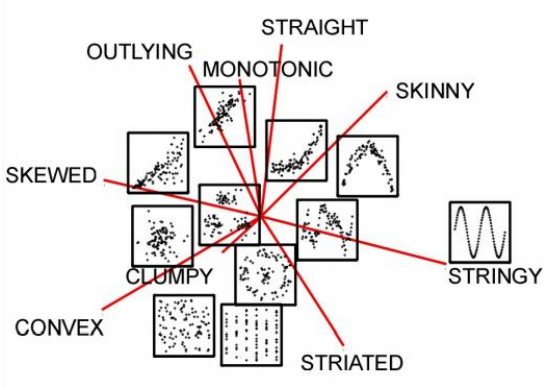


Graph-Theoretic Scagnostics

- graph-theoretic summaries of high-dimensional scattered point data.







scagnostics. [L. Wikinson et al, INFOVIS'05], [J. W. Tukey et al NCGA'85]

Large Graph Diagnosis

Large Graph Diagnosis

- Graph diagnosis (‘CT’ scans)

  Graph nodes distribution in correlated feature spaces.

Large Graph Diagnosis

- Graph diagnosis ('CT' scans)

  Graph nodes distribution in correlated feature spaces.

  E.g. Degree (in- / out- degree); # Triangle; Coreness;

  PageRank: (undirected homogeneous graph $\mathbf{Ax} = \lambda \mathbf{x}$)

  spectral vectors: $\mathbf{A} \approx \sigma \cdot \mathbf{u} \cdot \mathbf{v}^T$ ($\mathbf{u}$: hubness / $\mathbf{v}$: authority), etc.

Large Graph Diagnosis

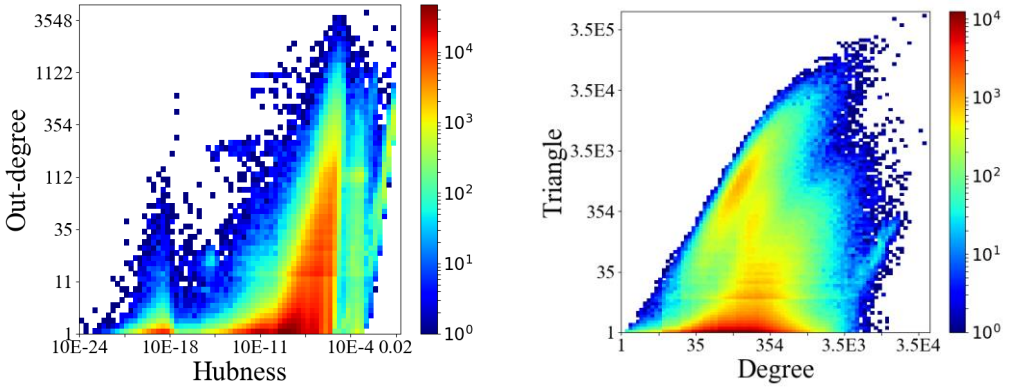
- Graph diagnosis ('CT' scans)

Graph nodes distribution in correlated feature spaces.

E.g. Degree (in- / out- degree); # Triangle; Coreness;

PageRank: (undirected homogeneous graph $\mathbf{Ax} = \lambda \mathbf{x}$)

spectral vectors: $\mathbf{A} \approx \sigma \cdot \mathbf{u} \cdot \mathbf{v}^T$ ($\mathbf{u}$: hubness / $\mathbf{v}$: authority), etc.



Two-dimension spaces

Large Graph Diagnosis

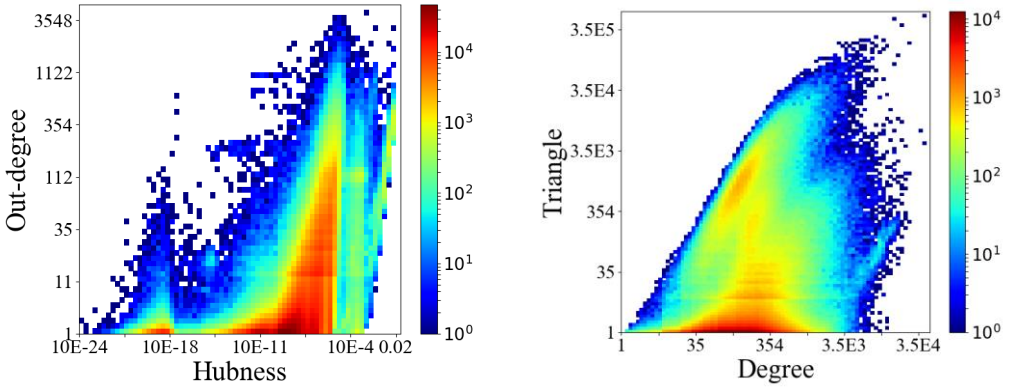
- Graph diagnosis ('CT' scans)

Graph nodes distribution in correlated feature spaces.

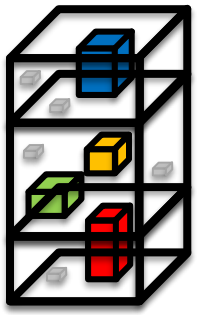
E.g. Degree (in- / out- degree); # Triangle; Coreness;

PageRank: (undirected homogeneous graph $\mathbf{Ax} = \lambda \mathbf{x}$)

spectral vectors: $\mathbf{A} \approx \sigma \cdot \mathbf{u} \cdot \mathbf{v}^T$ ($\mathbf{u}$: hubness / $\mathbf{v}$: authority), etc.



Two-dimension spaces



High-dimension spaces

Large Graph Diagnosis

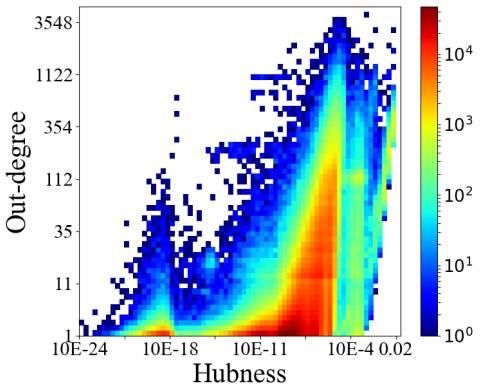
- Graph diagnosis ('CT' scans)

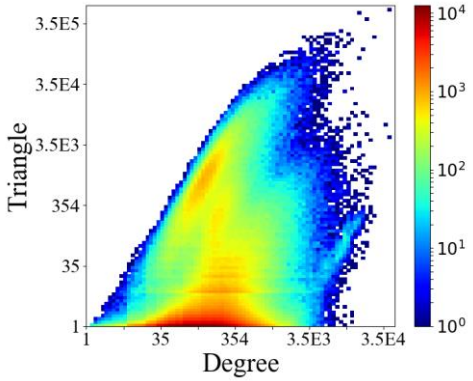
Graph nodes distribution in correlated feature spaces.

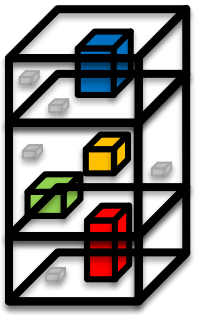
E.g. Degree (in- / out- degree); # Triangle; Coreness;

PageRank: (undirected homogeneous graph $\mathbf{Ax} = \lambda \mathbf{x}$)

spectral vectors: $\mathbf{A} \approx \sigma \cdot \mathbf{u} \cdot \mathbf{v}^T$ ($\mathbf{u}$: hubness / $\mathbf{v}$: authority), etc.







Two-dimension spaces

High-dimension spaces

Problem definition

- **Vision-guided** diagnosis & mining for large graph:

Problem definition

- **Vision-guided** diagnosis & mining for large graph:

Given:

Problem definition

- **Vision-guided** diagnosis & mining for large graph:

Given:

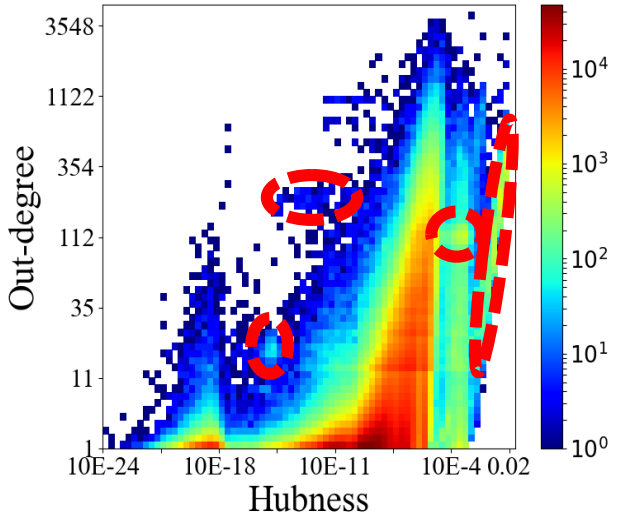
- Graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, **A** is the adjacency matrix
- Node features histogram $\mathcal{H}$

Problem definition

- **Vision-guided** diagnosis & mining for large graph:

Given:

- Graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, **A** is the adjacency matrix
- Node features histogram $\mathcal{H}$



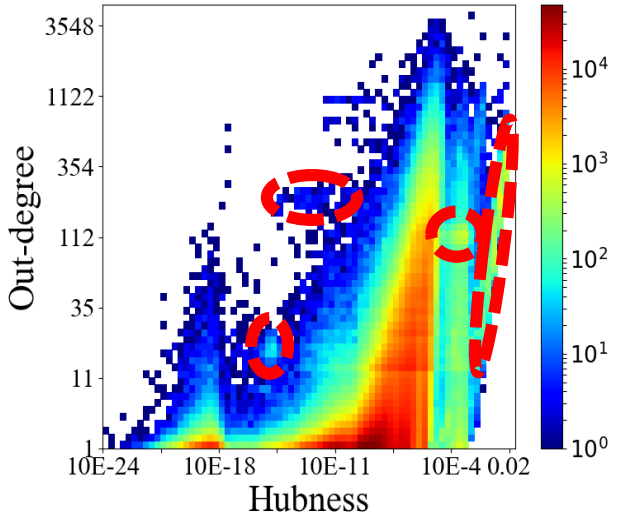
Problem definition

- **Vision-guided** diagnosis & mining for large graph:

Given:

- Graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, **A** is the adjacency matrix
- Node features histogram $\mathcal{H}$

Goal (automatically):



Problem definition

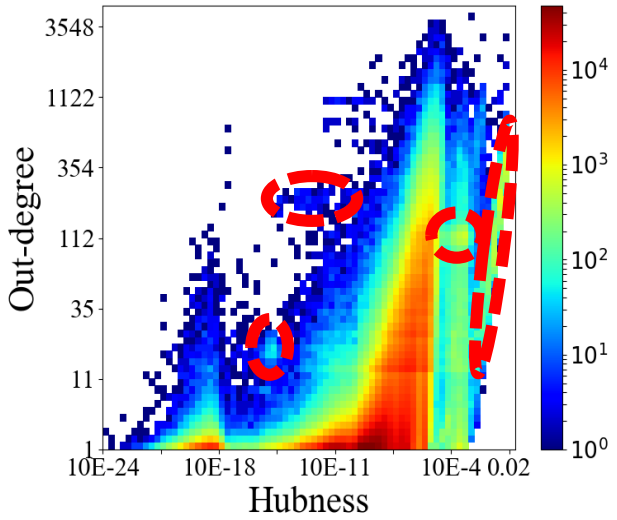
- **Vision-guided** diagnosis & mining for large graph:

Given:

- Graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, **A** is the adjacency matrix
- Node features histogram $\mathcal{H}$

Goal (automatically):

- 1. *recognize* and monitor node groups;
- 2. *summary* graph nodes in feature space & *identify* suspicious micro-clusters



Problem definition

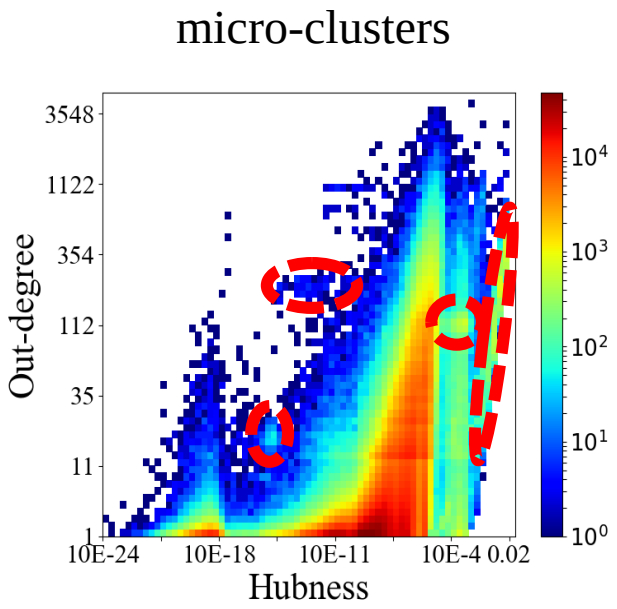
- **Vision-guided** diagnosis & mining for large graph:

Given:

- Graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, **A** is the adjacency matrix
- Node features histogram $\mathcal{H}$

Goal (automatically):

- 1. *recognize* and monitor node groups;
- 2. *summary* graph nodes in feature space & *identify* suspicious micro-clusters



Problem definition

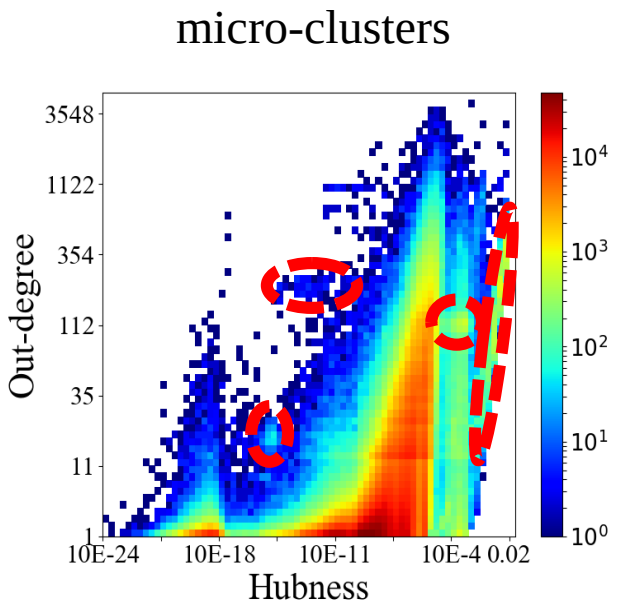
- **Vision-guided** diagnosis & mining for large graph:

Given:

- Graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, **A** is the adjacency matrix
- Node features histogram $\mathcal{H}$

Goal (automatically):

- 1. *recognize* and monitor node groups;
- 2. *summary* graph nodes in feature space & *identify* suspicious micro-clusters



Solution: **EagleMine** algorithm

Outline

- Introduction
- Overview
- Proposed Method: **EagleMine <<**
  - Problem definition
  - Algorithm **<<**
- Experiments
- Conclusion

EagleMine algorithm

- **Vision-guided** diagnosis & mining for large graph:

EagleMine algorithm

- **Vision-guided** diagnosis & mining for large graph:
  - 1. Recognize node groups with Hierarchical tree structure for $\mathcal{H}$

    WATERLEVELTREE algorithm

EagleMine algorithm

- **Vision-guided** diagnosis & mining for large graph:
  - 1. Recognize node groups with Hierarchical tree structure for $\mathcal{H}$

    WATERLEVELTREE algorithm
  - 2. Search the tree and get summarization of the histogram:

    TREEEXPLORE algorithm

EagleMine algorithm

- **Vision-guided** diagnosis & mining for large graph:
  - 1. Recognize node groups with Hierarchical tree structure for $\mathcal{H}$

    WATERLEVELTREE algorithm
  - 2. Search the tree and get summarization of the histogram:

    TREEEXPLORE algorithm

    Vocabulary-based summarization model

    Hypothesis test selection

EagleMine algorithm

- **Vision-guided** diagnosis & mining for large graph:
  - 1. Recognize node groups with Hierarchical tree structure for $\mathcal{H}$

    WATERLEVELTREE algorithm
  - 2. Search the tree and get summarization of the histogram:

    TREEEXPLORE algorithm

    Vocabulary-based summarization model

    Hypothesis test selection

Brain recognize object [J. DiCarlo et al Neuron'12]

EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

EagleMine algorithm part I

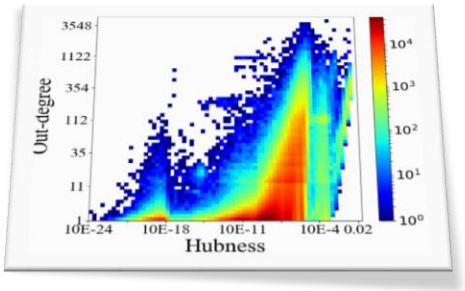
1. Recognize micro-clusters **WATERLEVELTREE**
    I. **Build** waterlevel tree;

EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree;

①



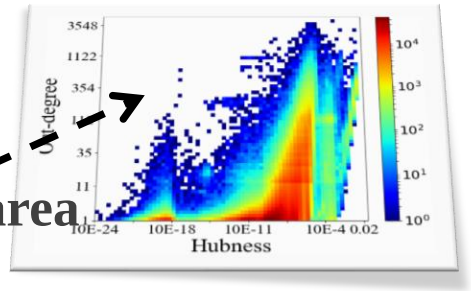
EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree;

①

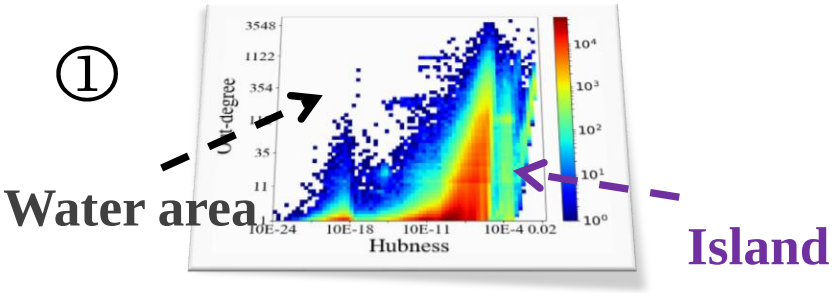
Water area



EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

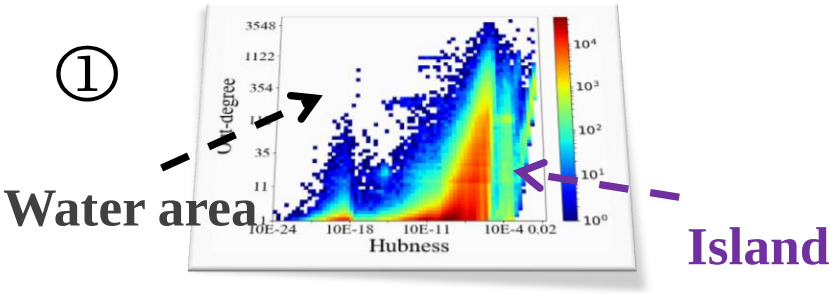
   I. **Build** waterlevel tree;



EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

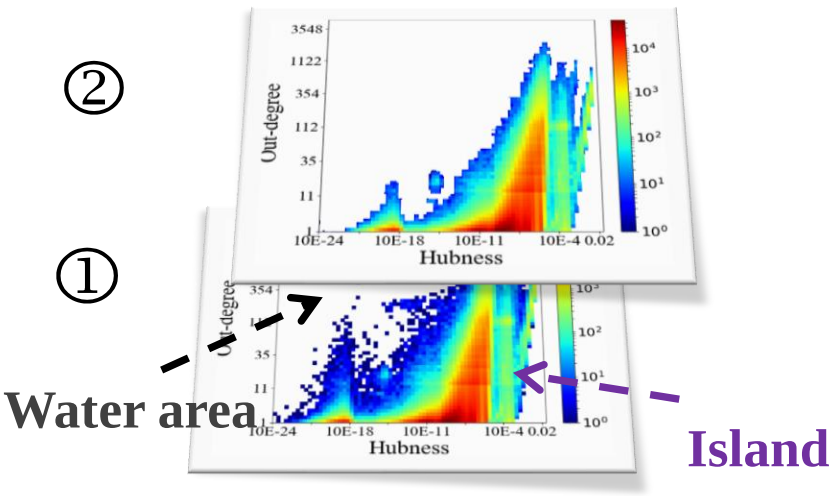
   I. **Build** waterlevel tree;



EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

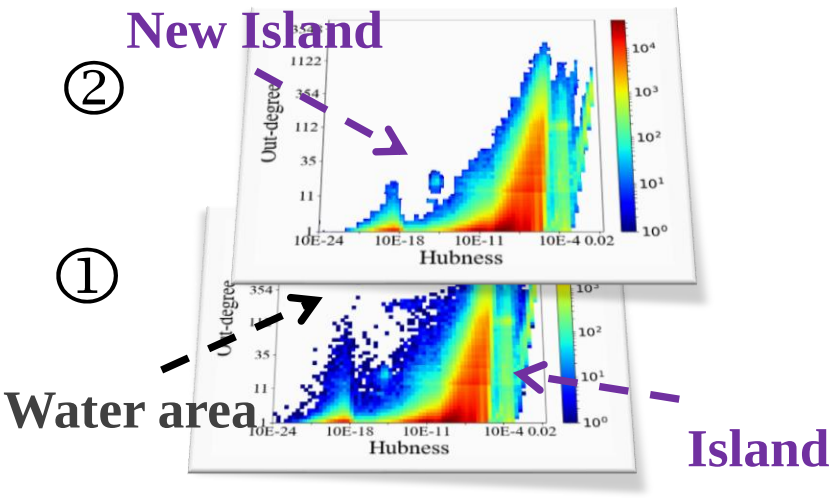
   I. **Build** waterlevel tree;



EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

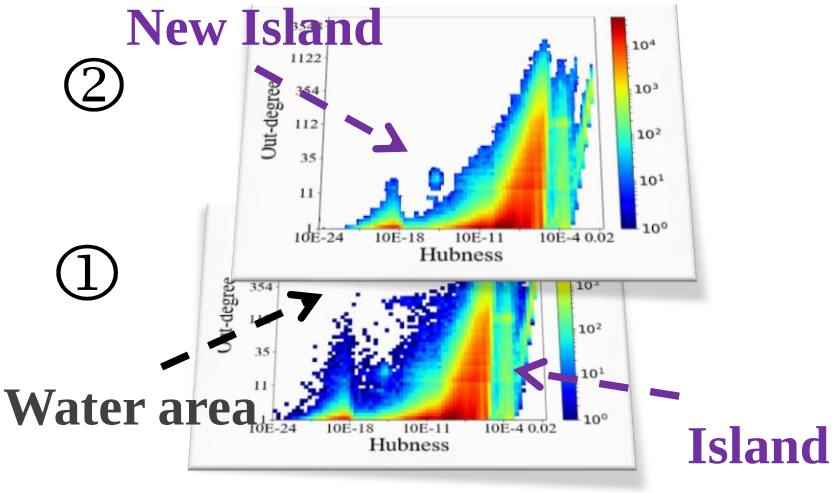
   I. **Build** waterlevel tree;

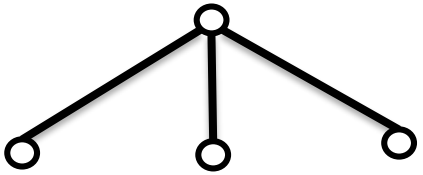


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree;

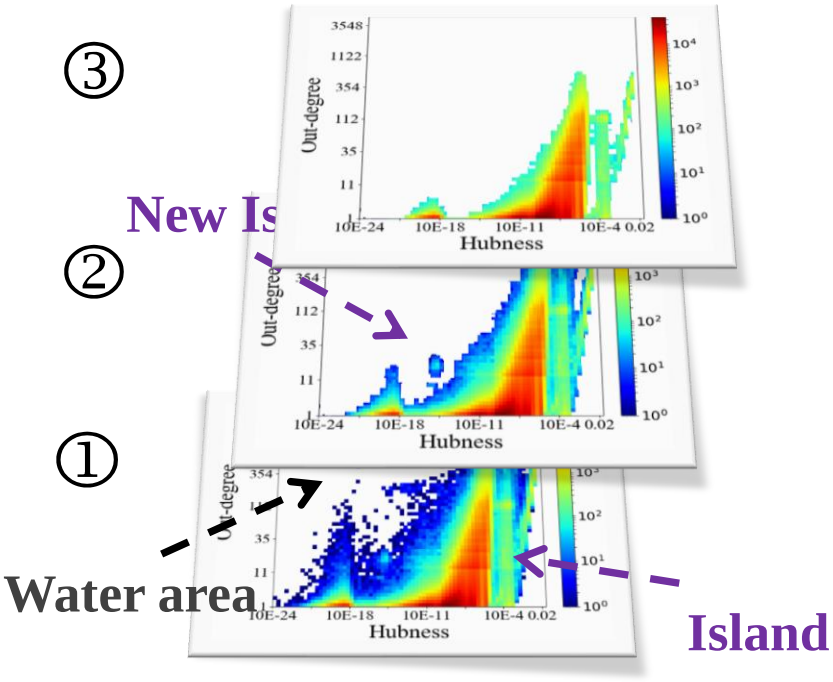


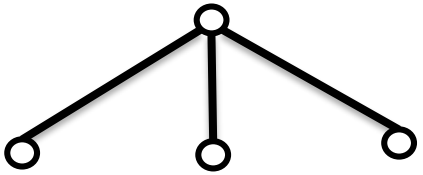


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree;

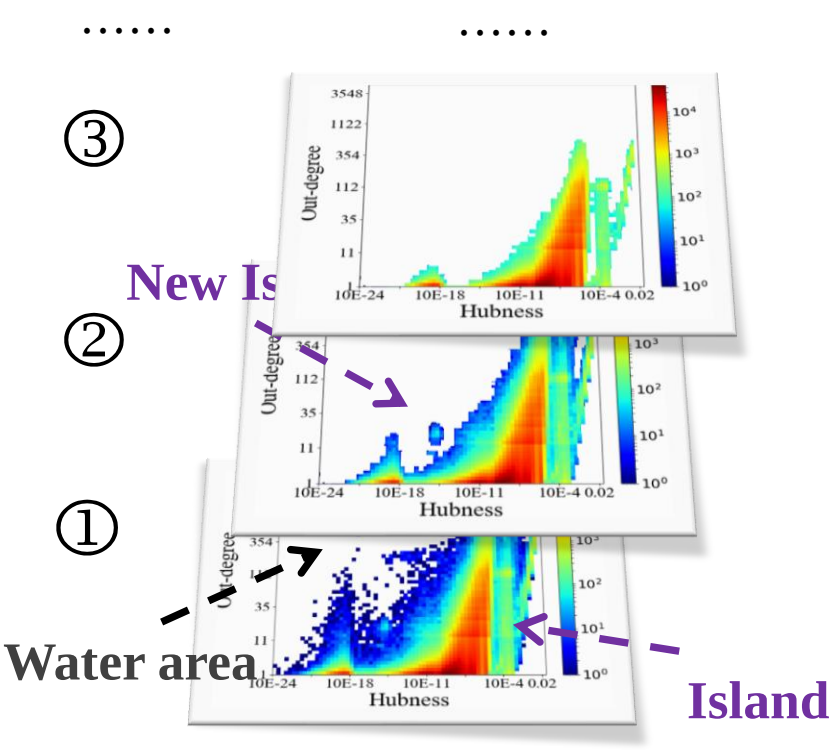


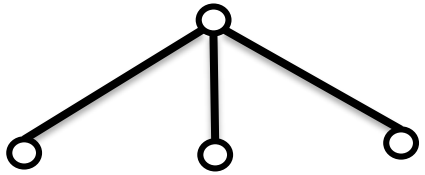


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree;

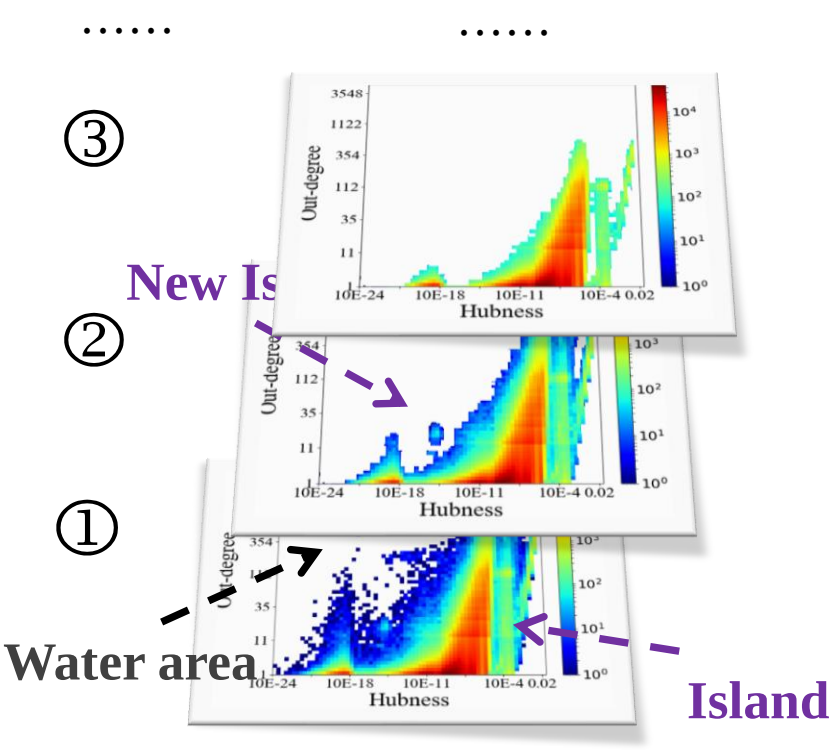


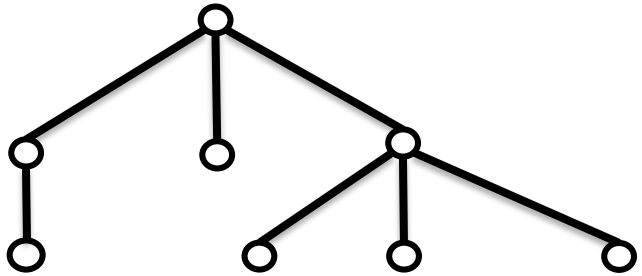


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree;

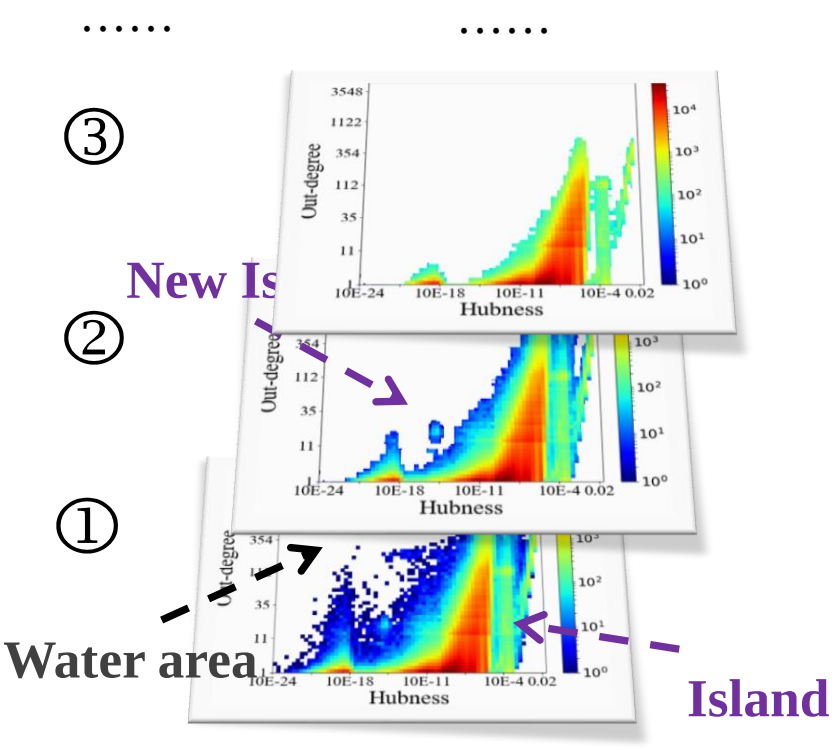


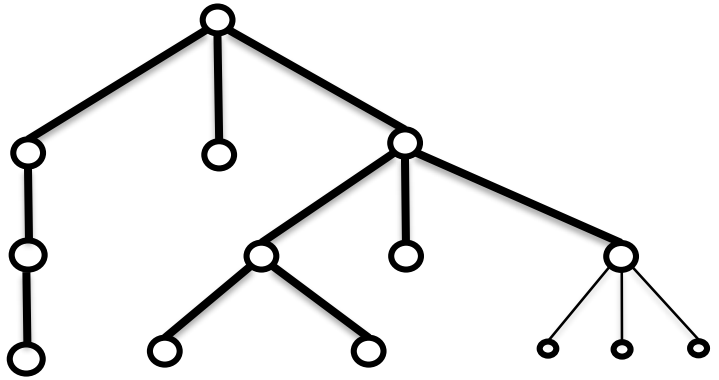


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree;

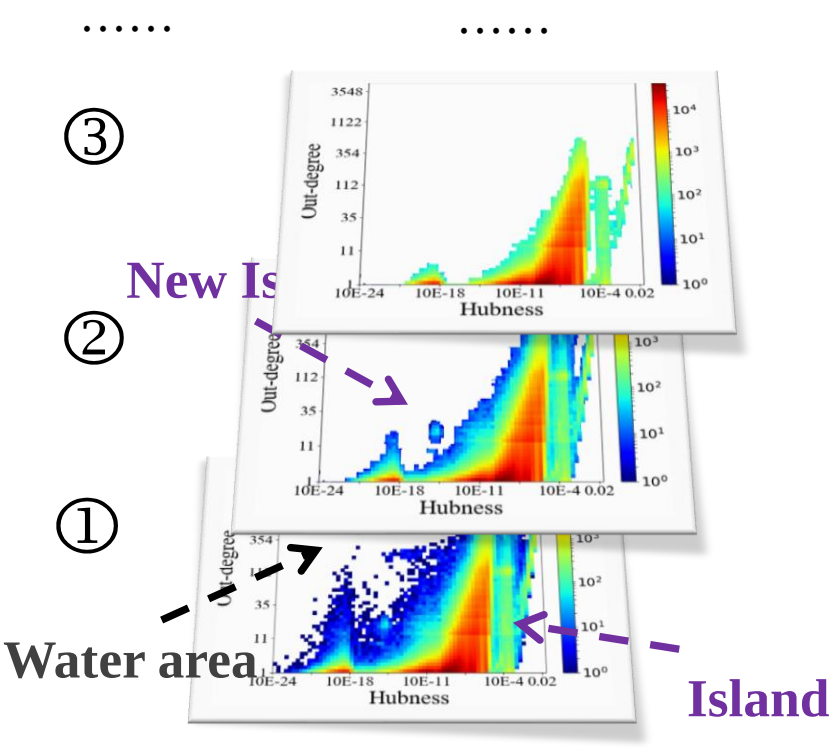


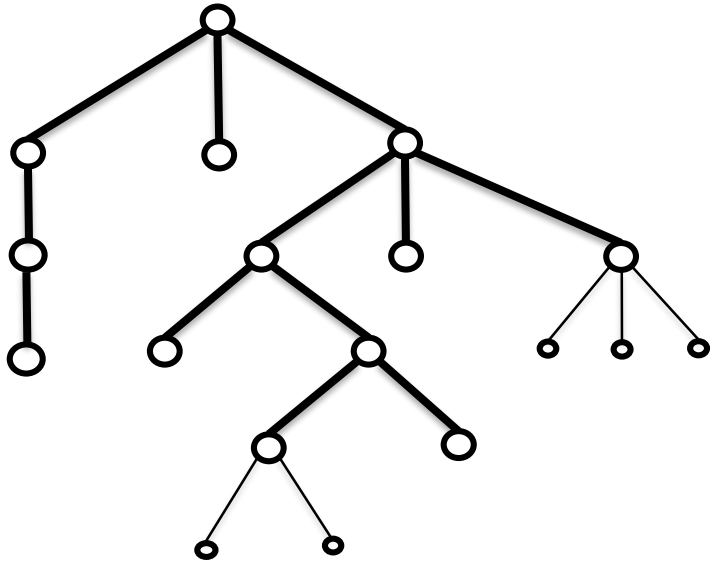


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree;

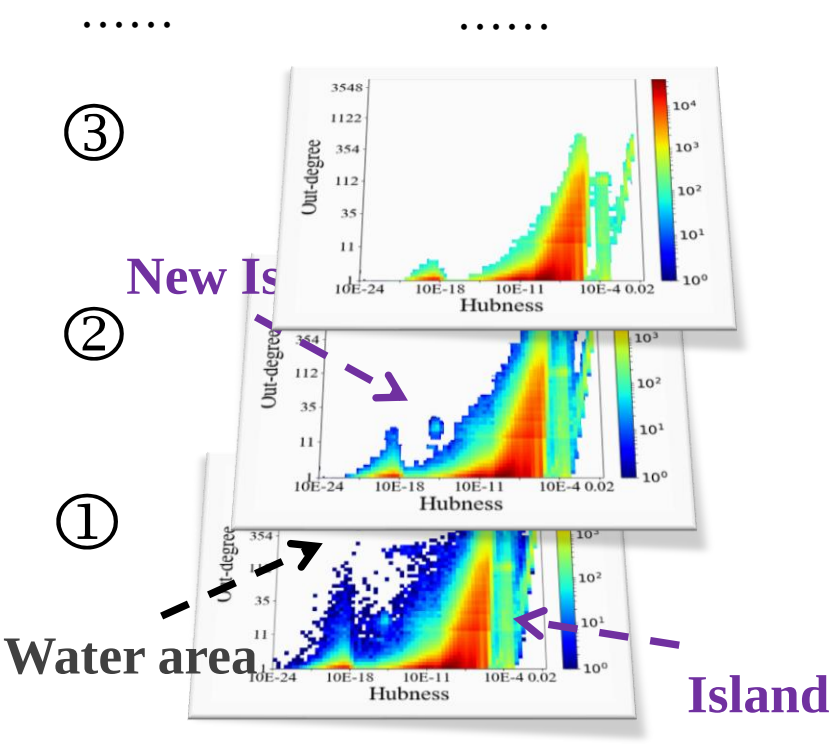


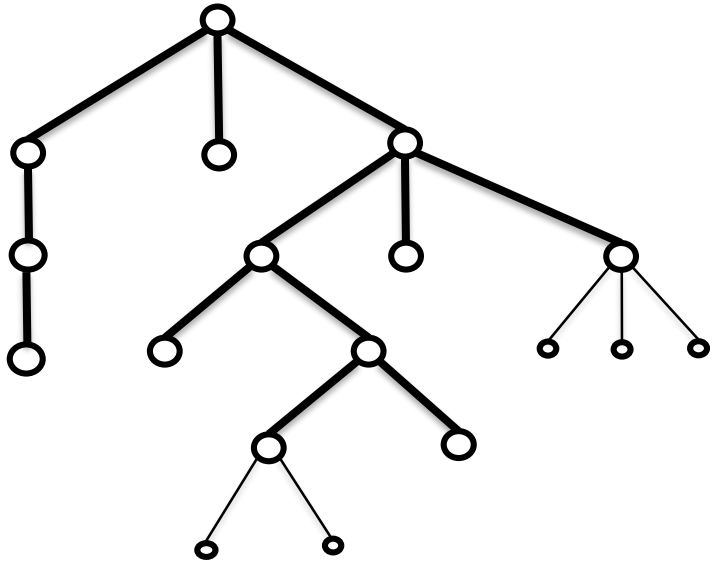


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

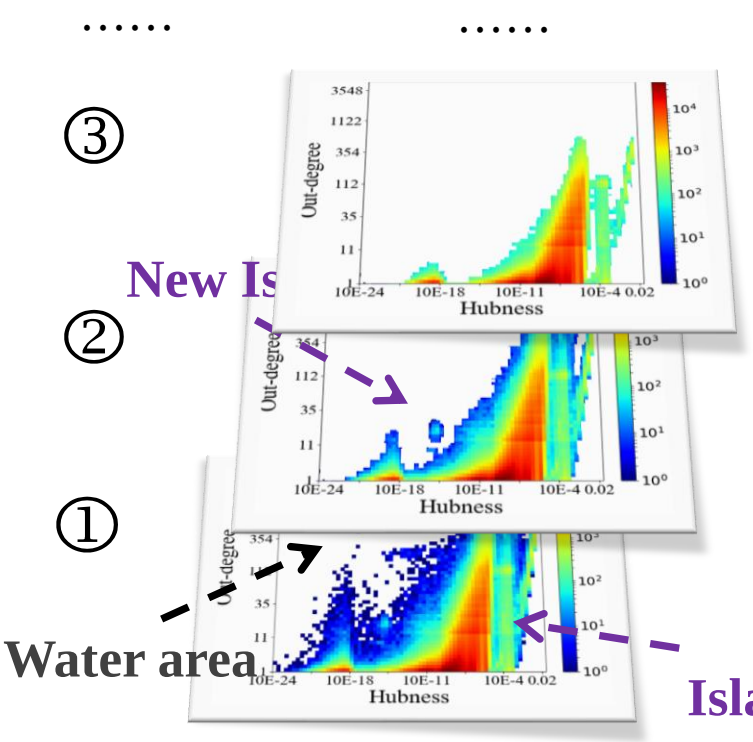


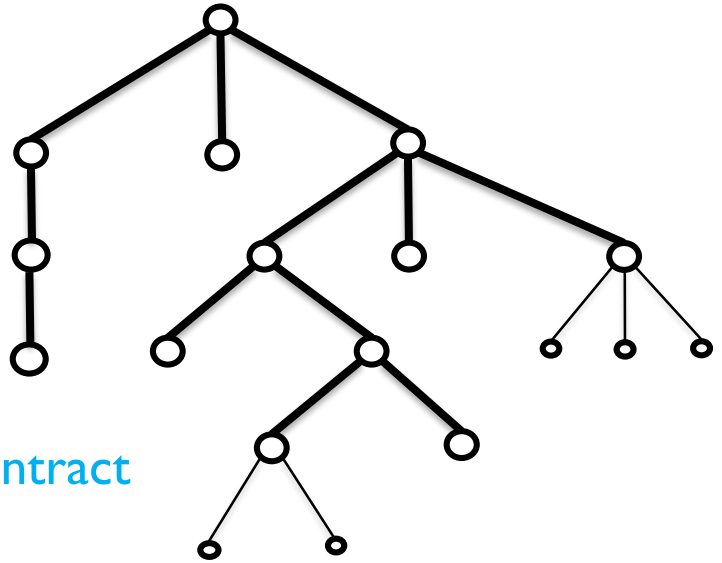


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

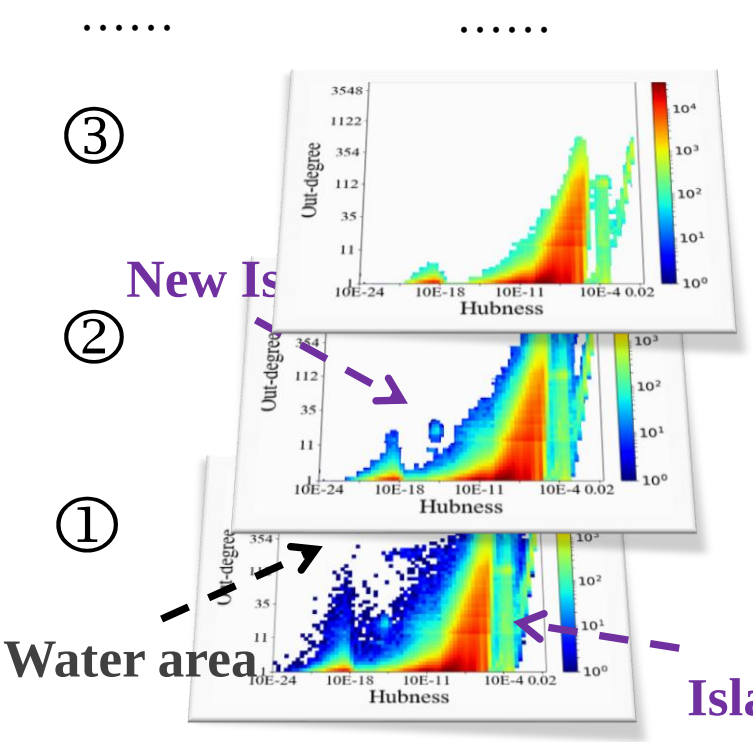


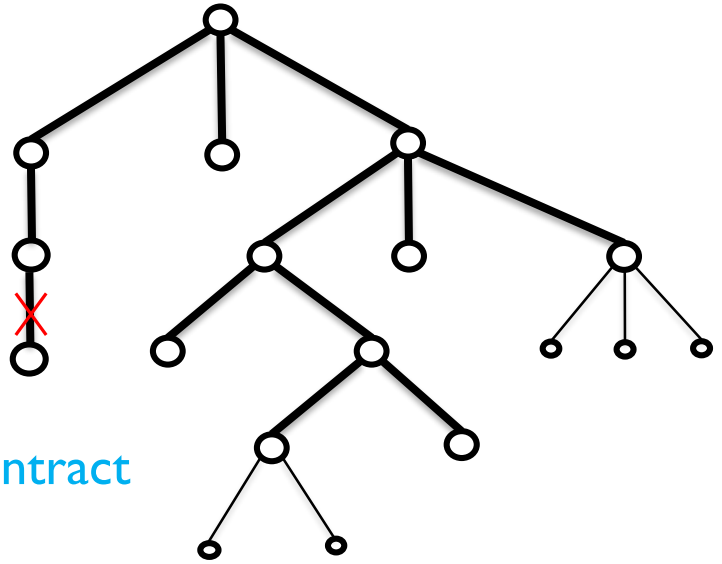


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

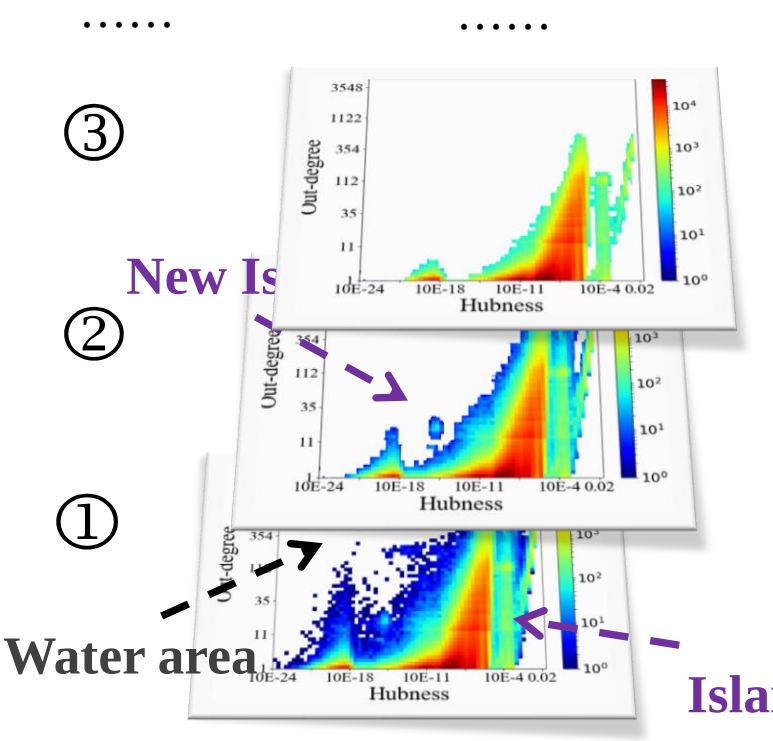


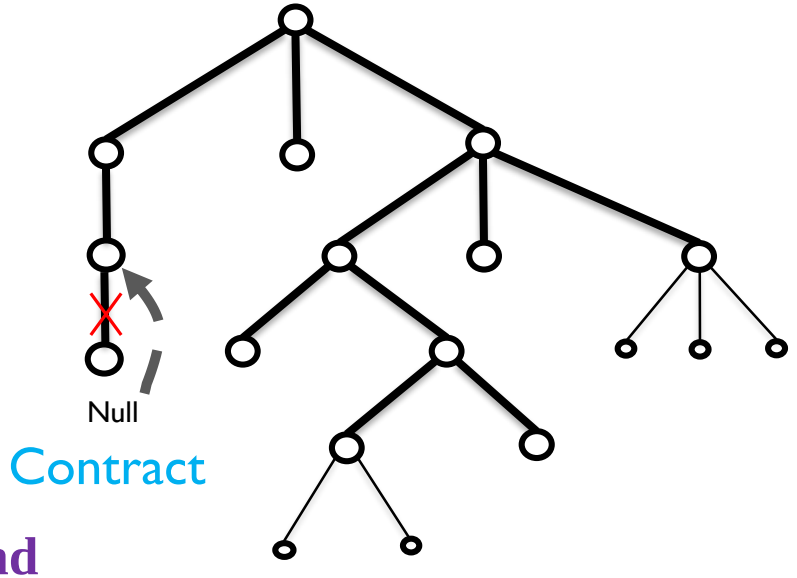


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

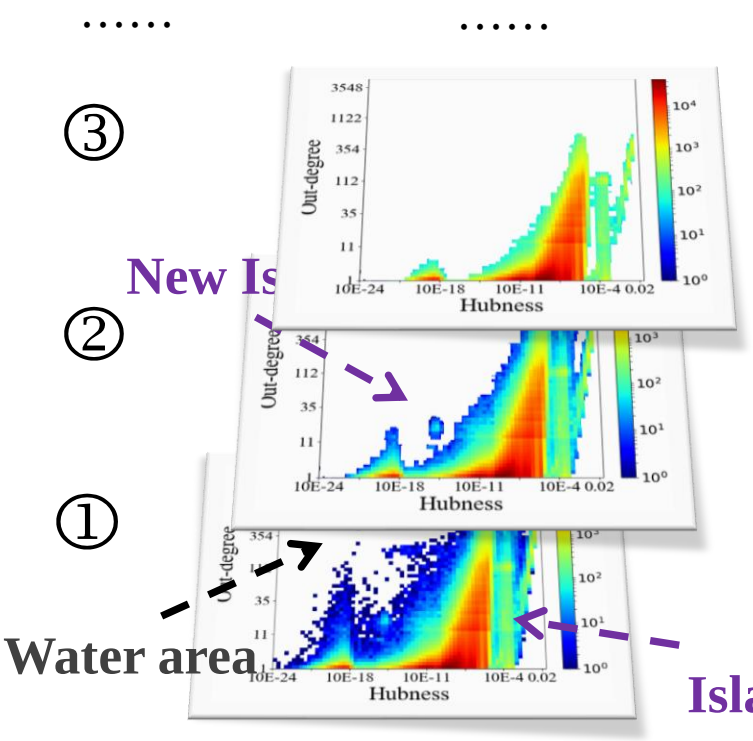


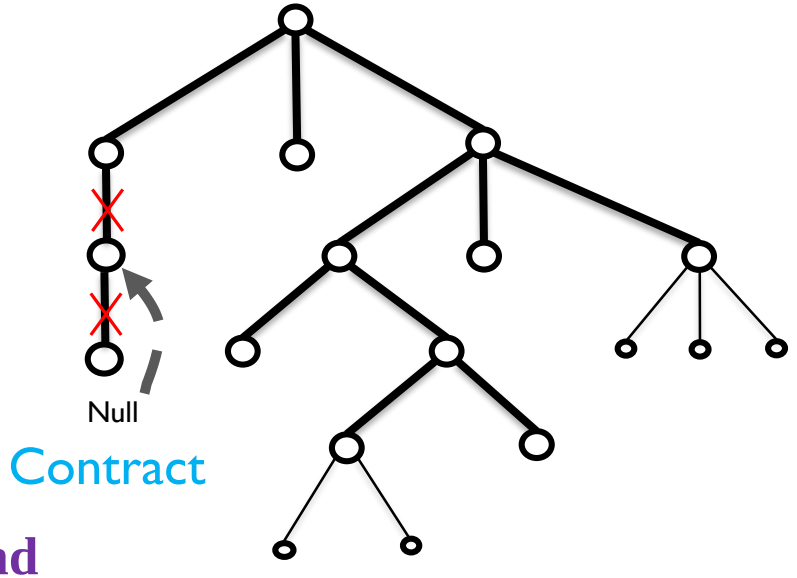


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

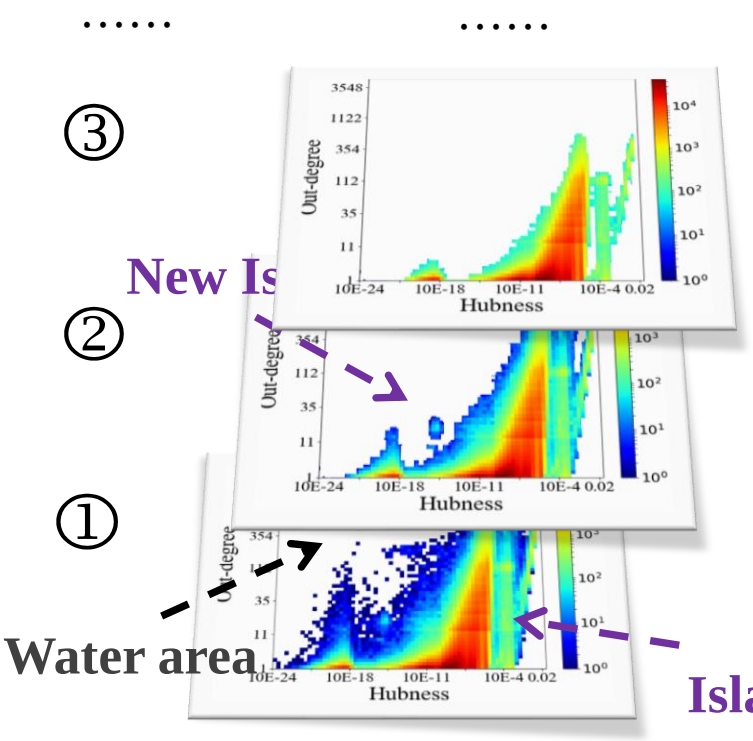


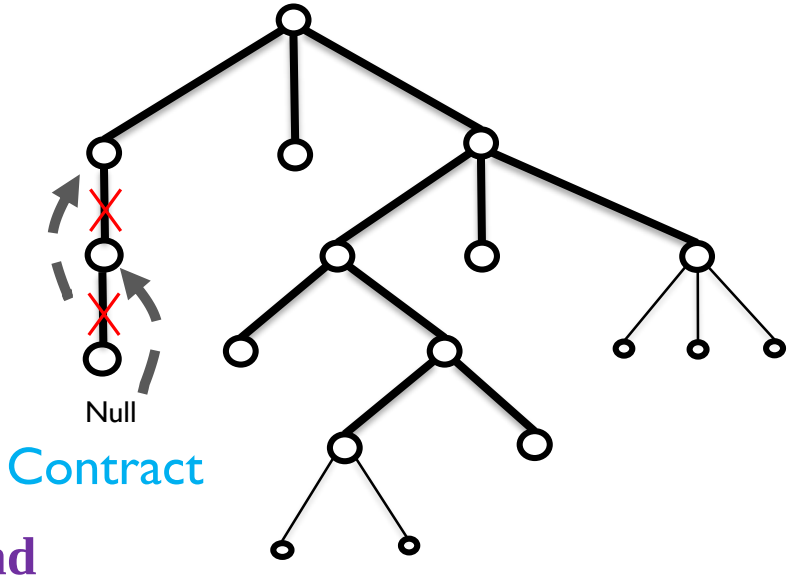


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

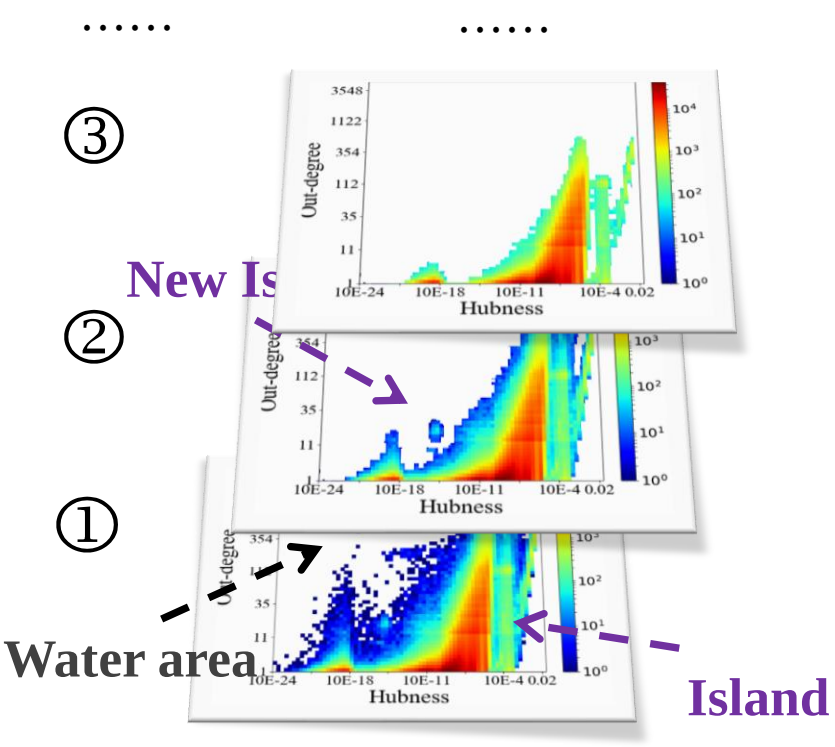


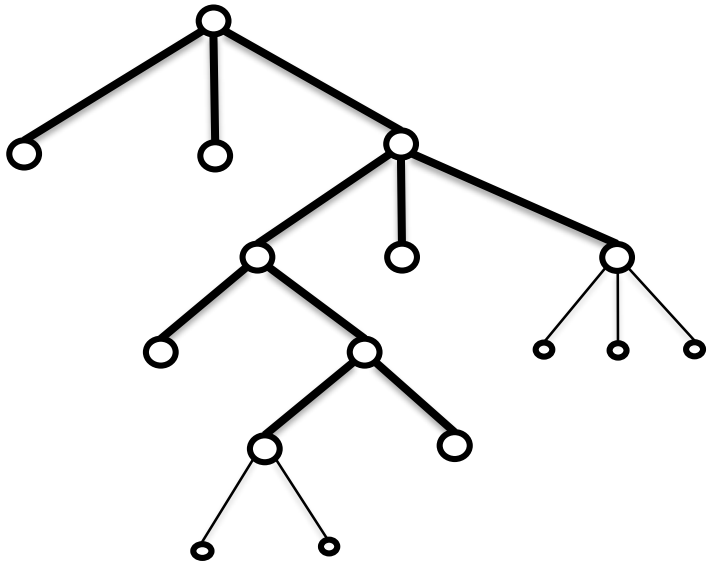


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

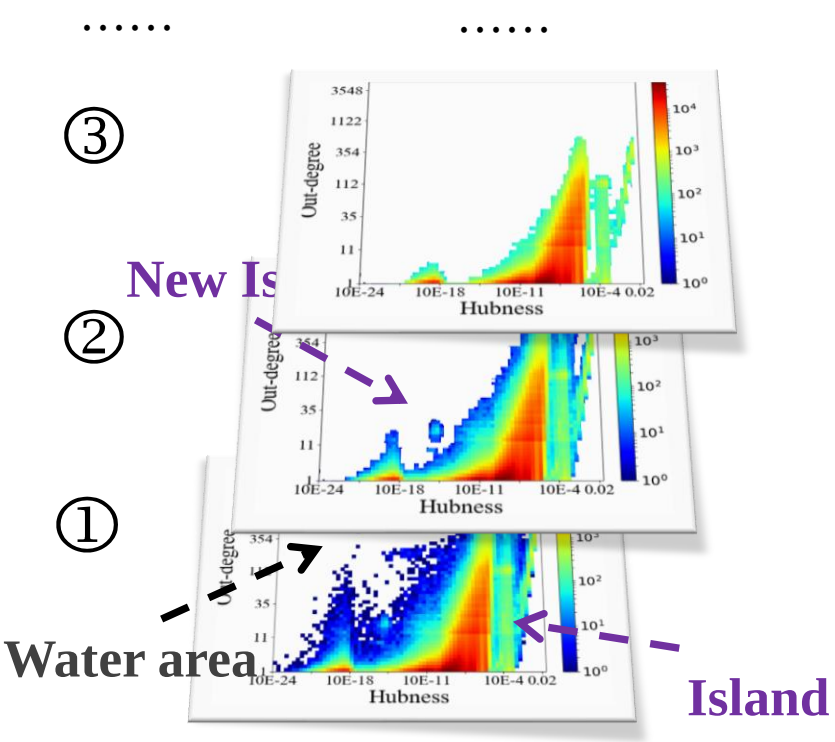


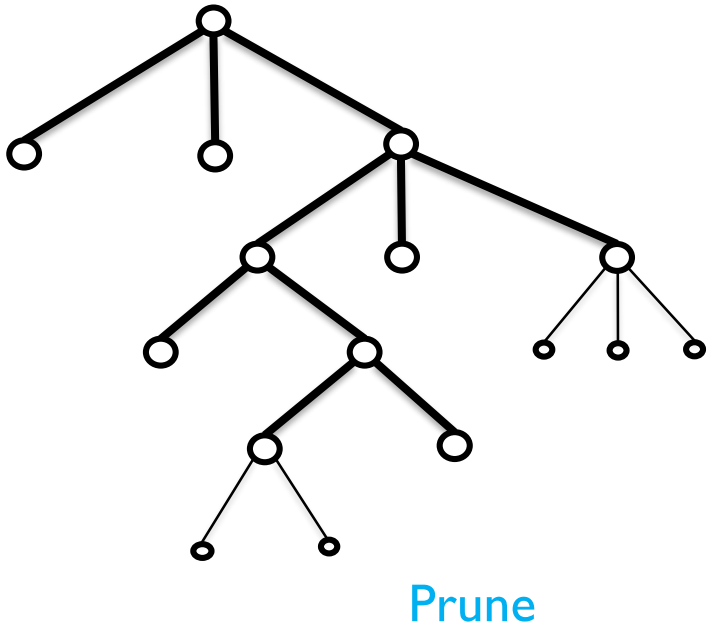


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

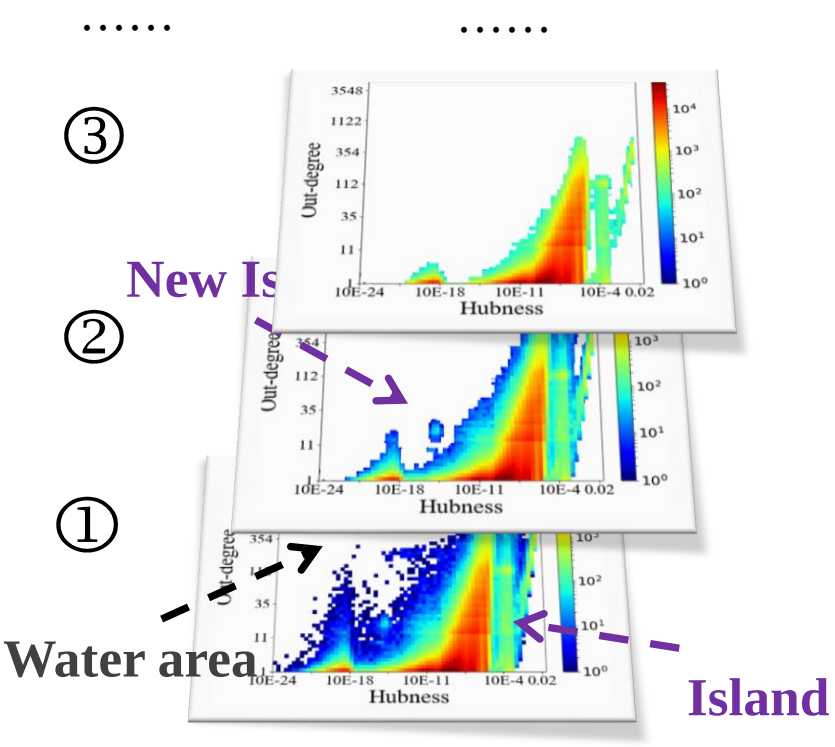


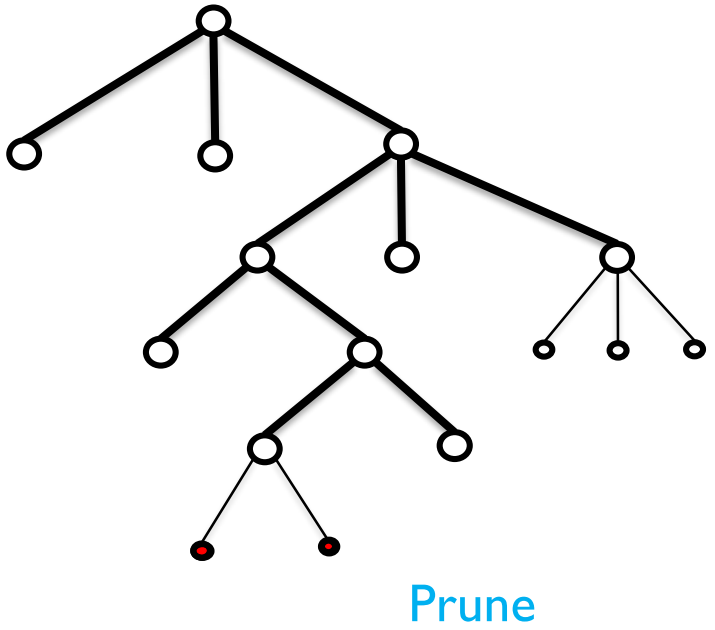


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;



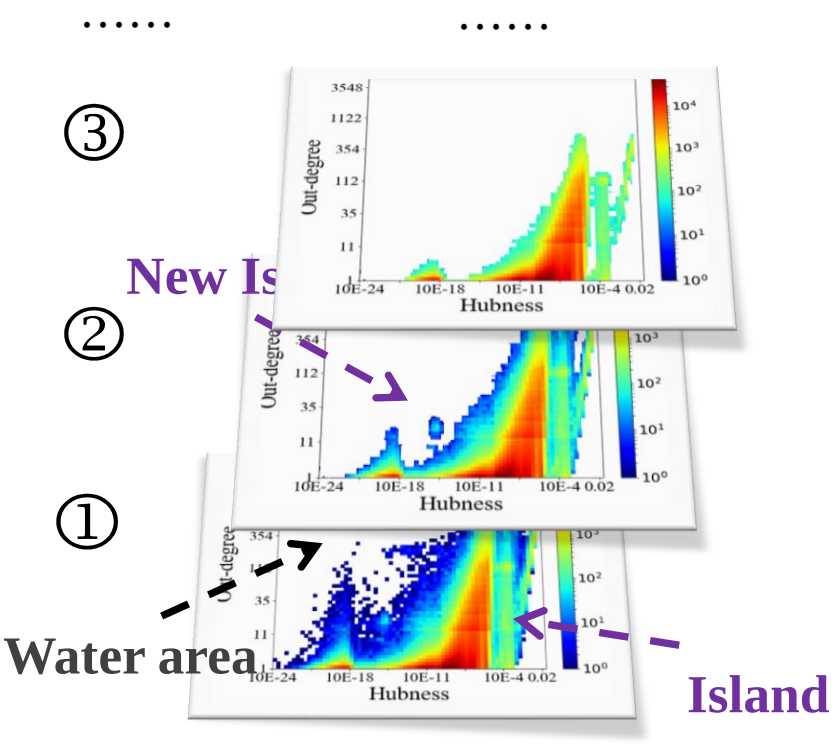


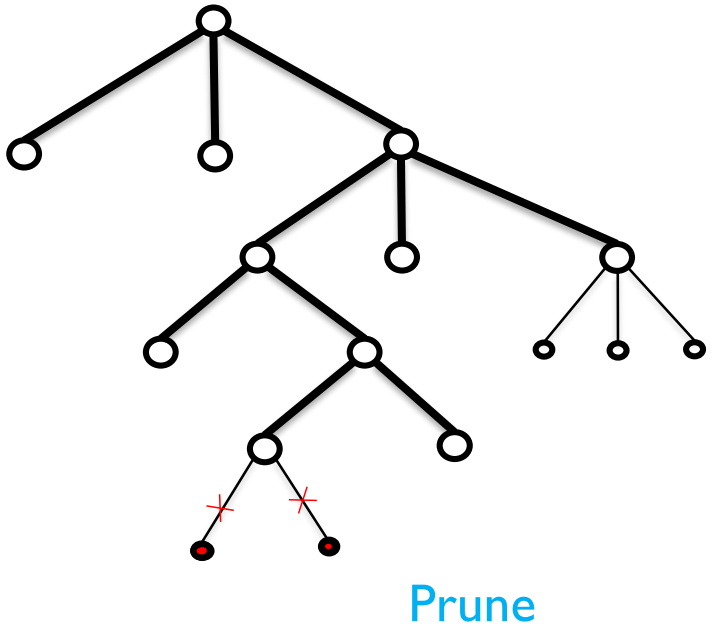
Prune

EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

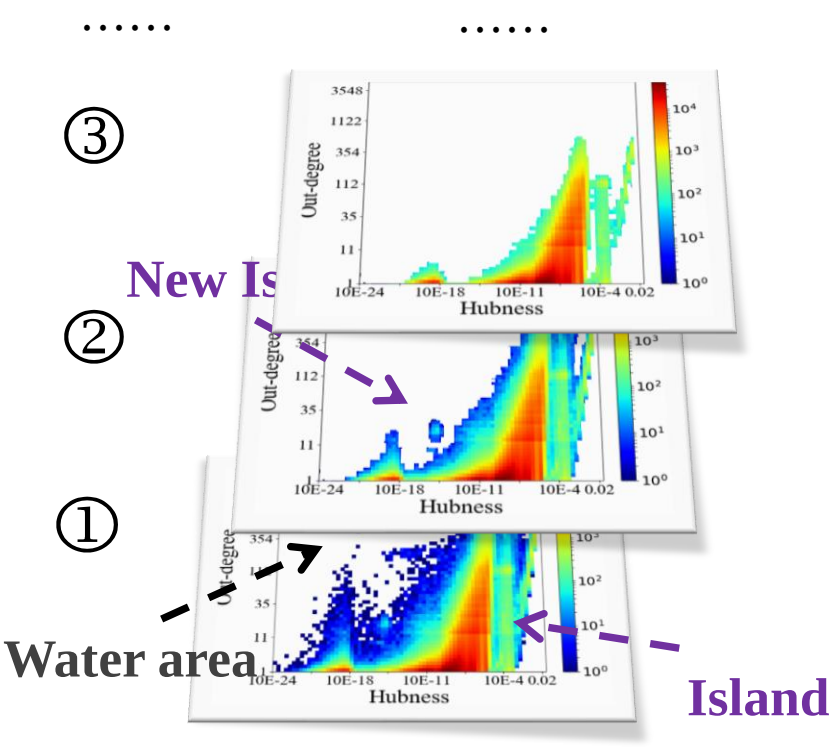


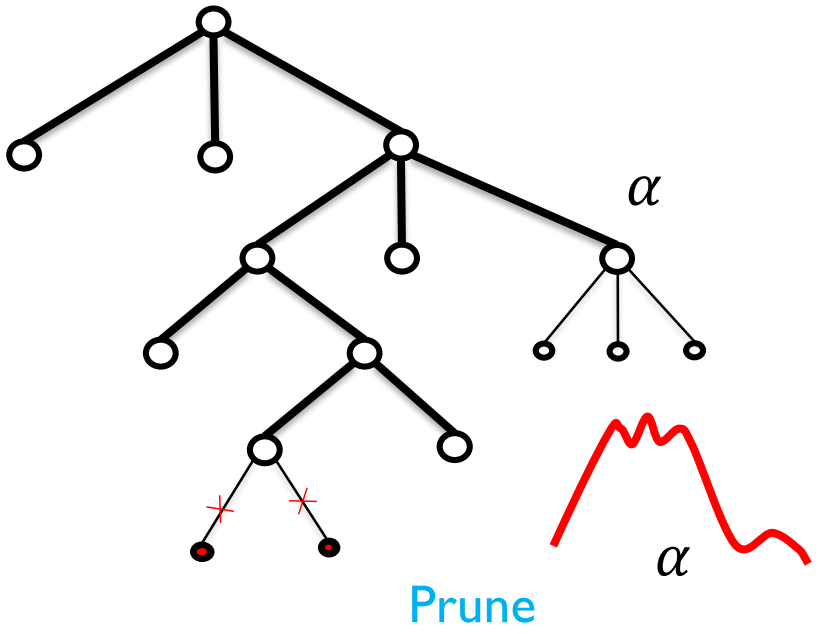


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

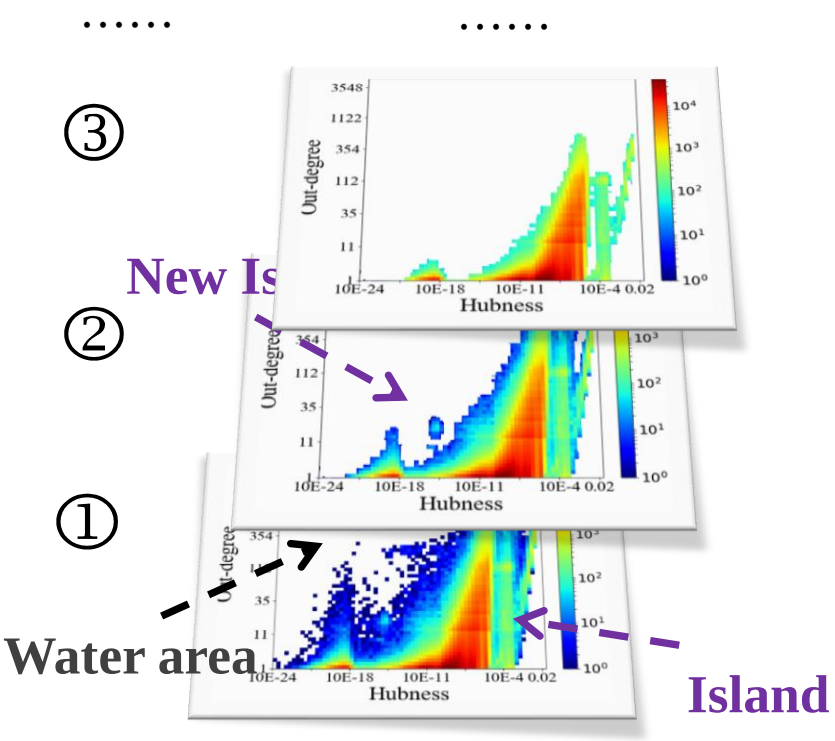


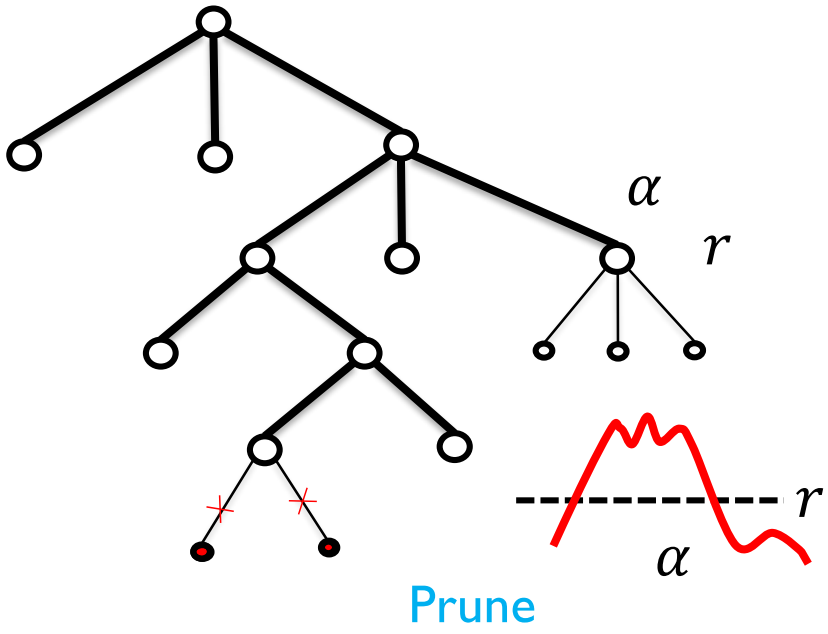


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

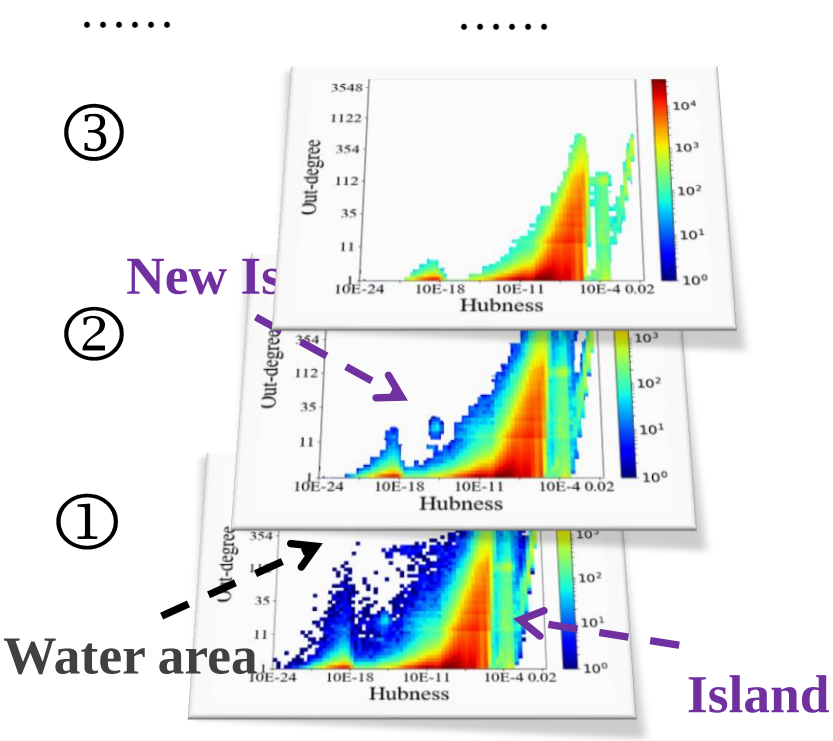


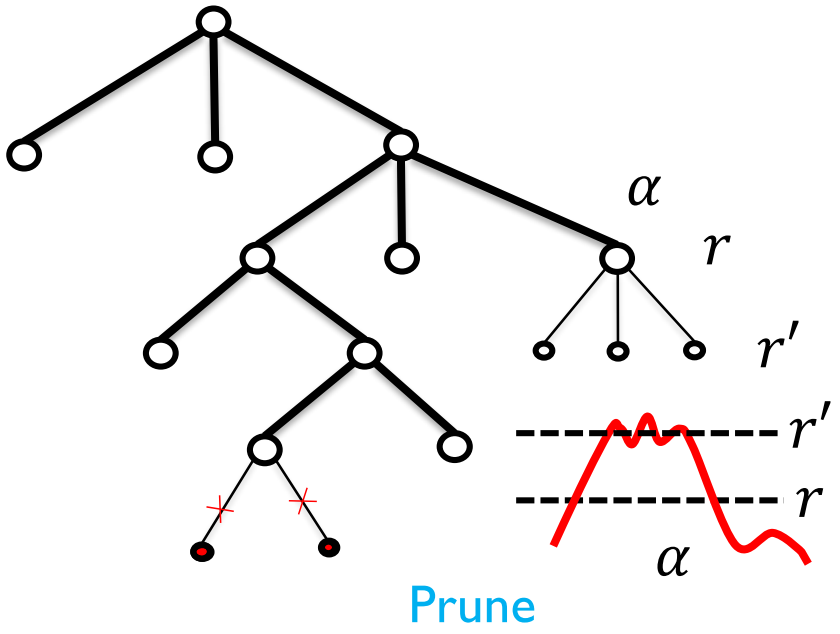


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

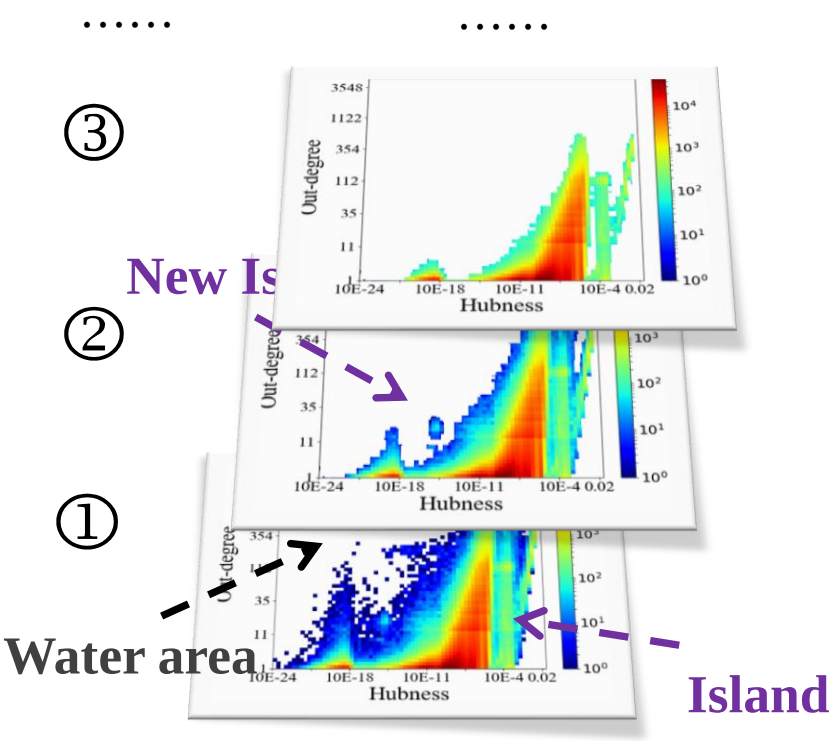


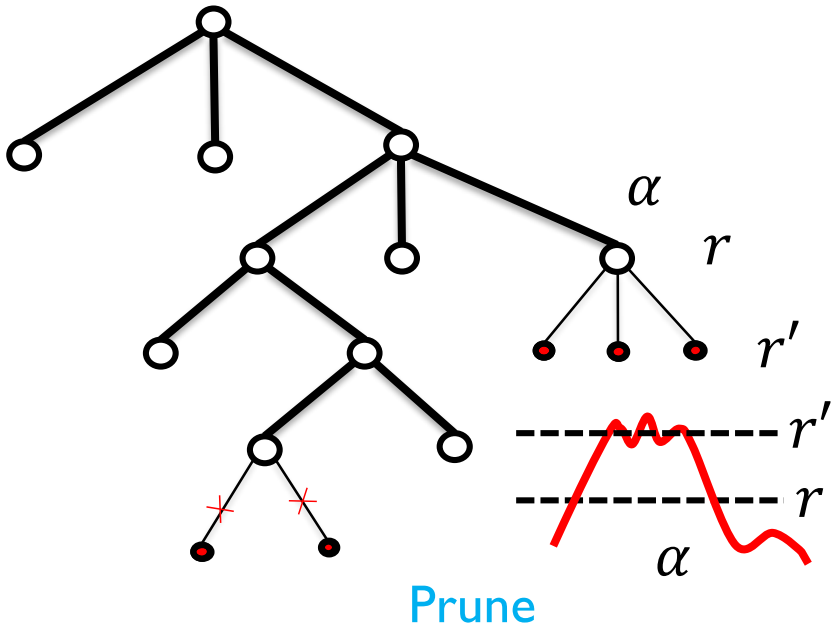


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

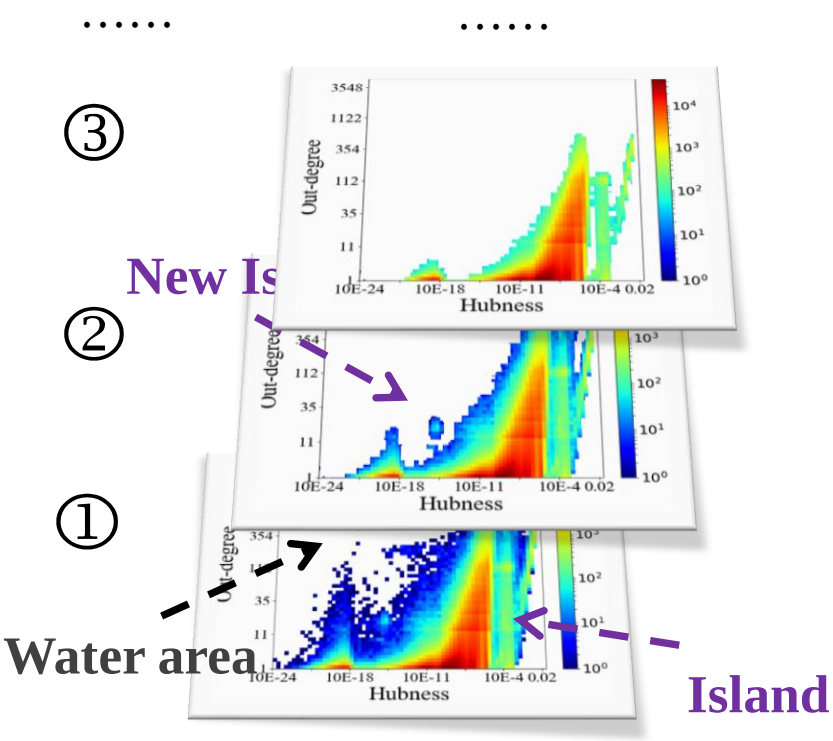


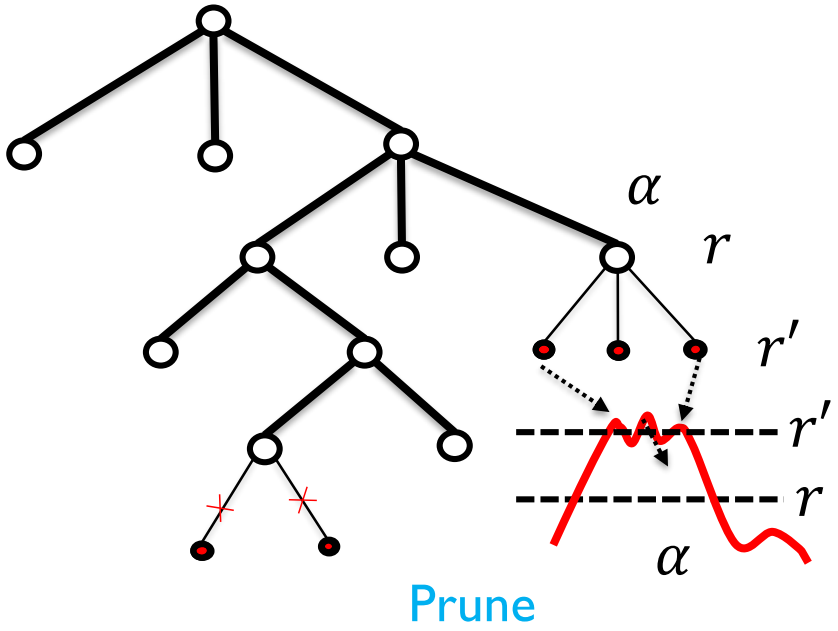


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

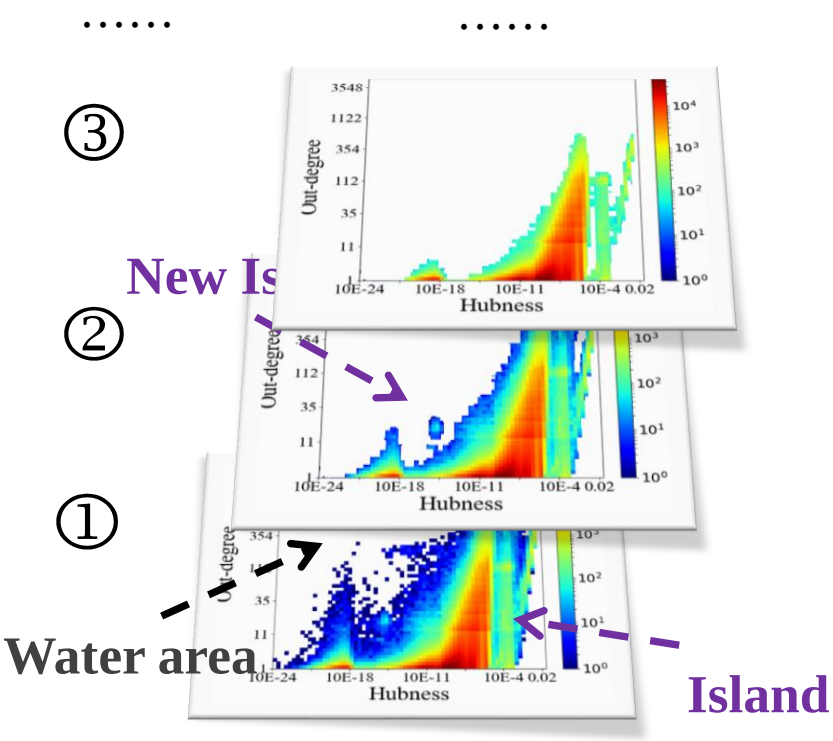


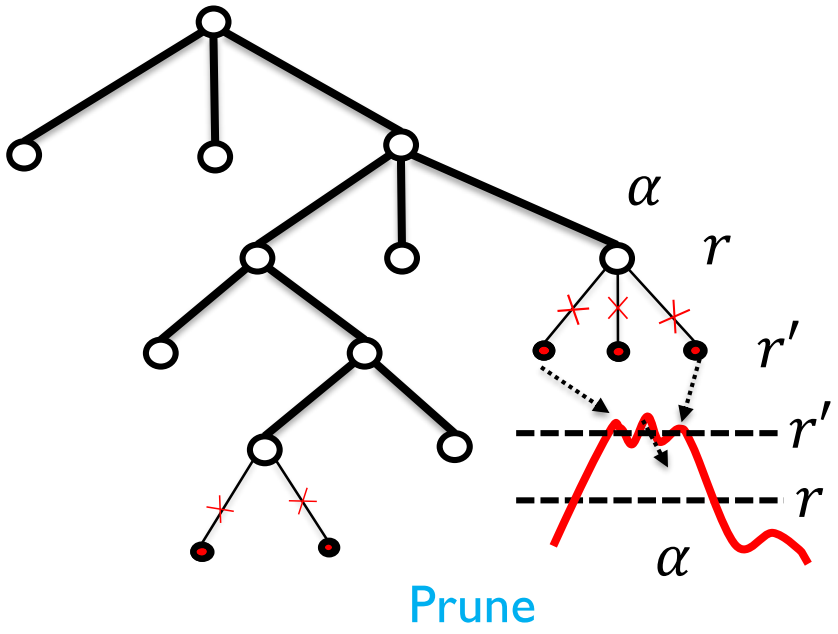


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

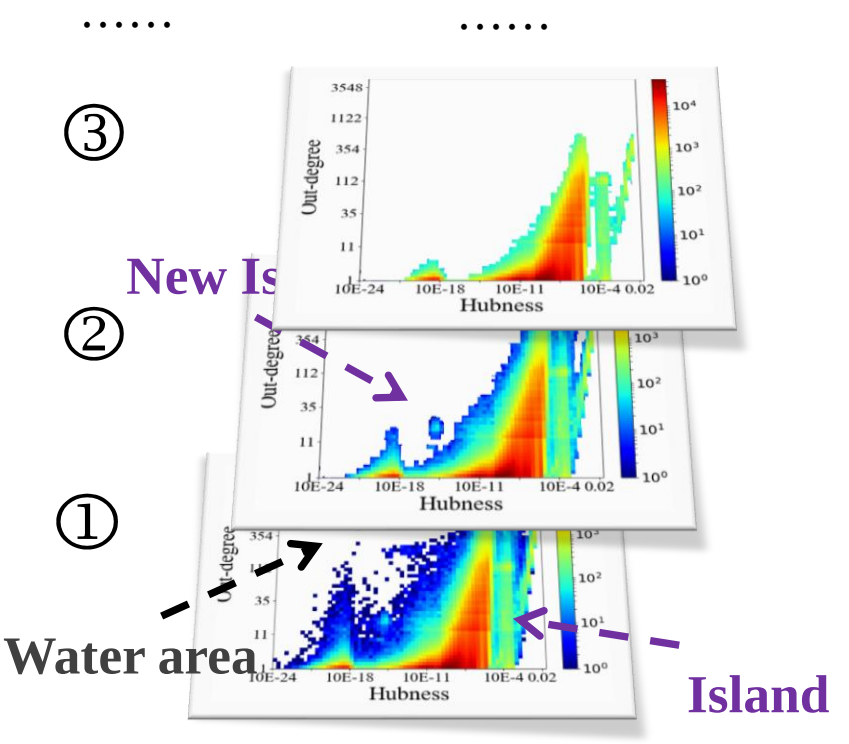


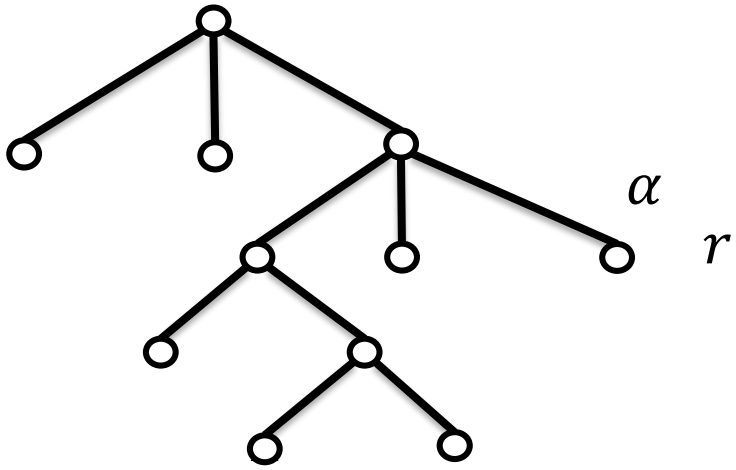


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

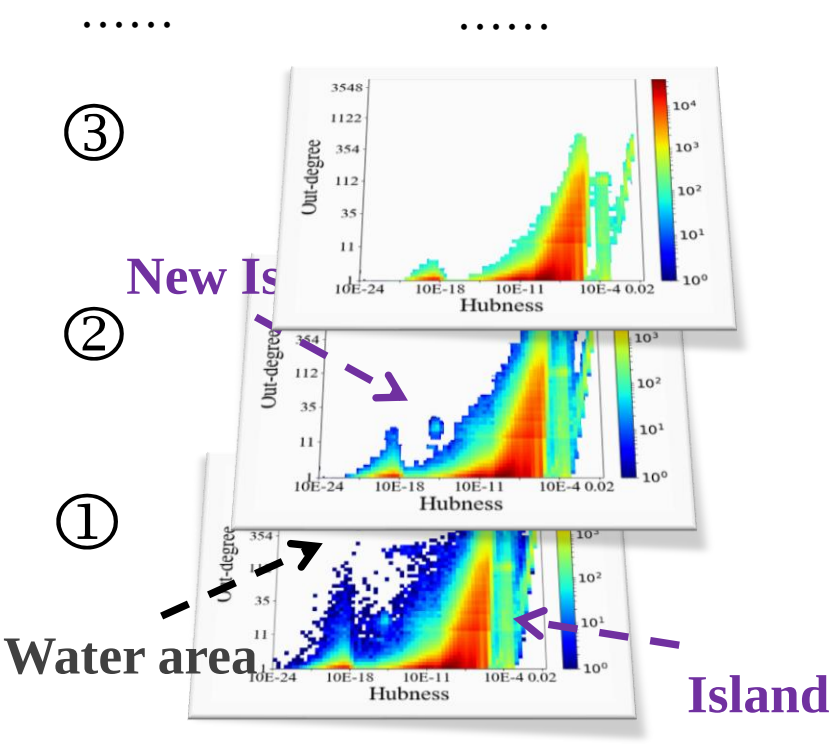


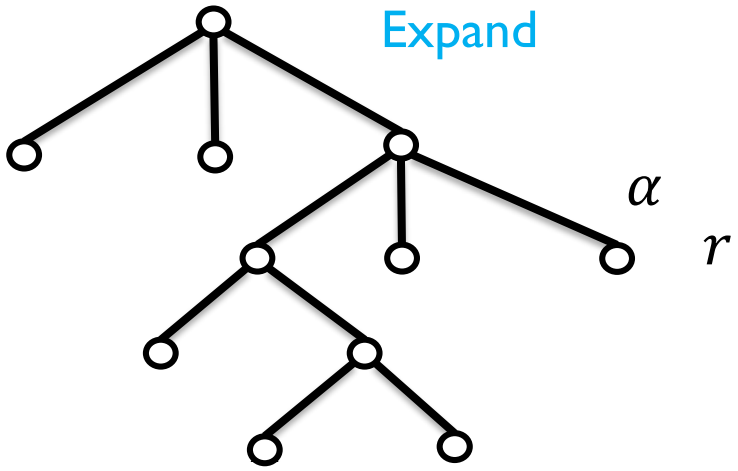


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

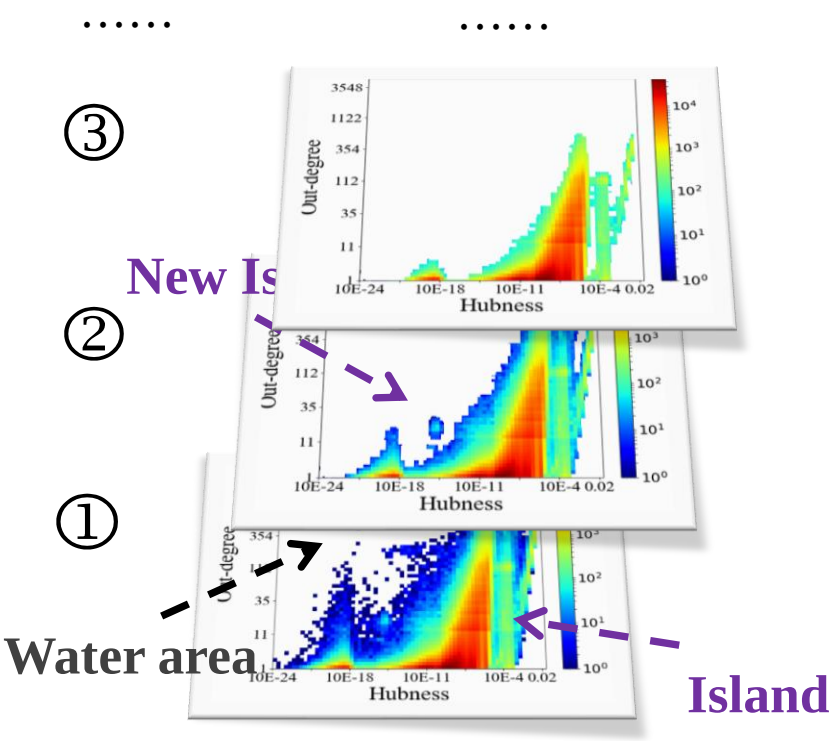


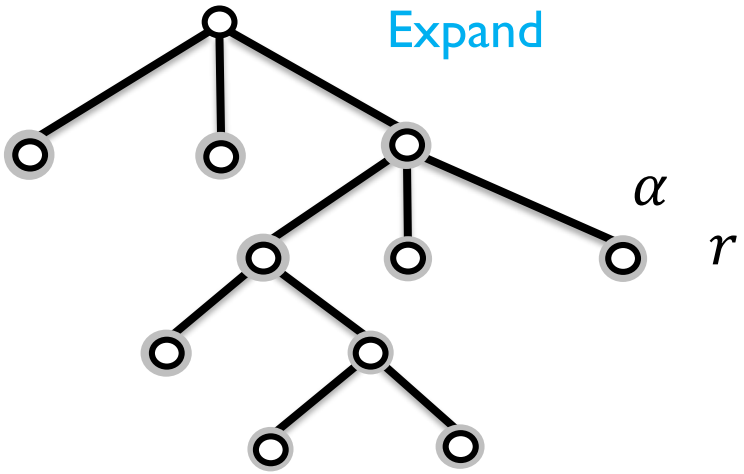


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

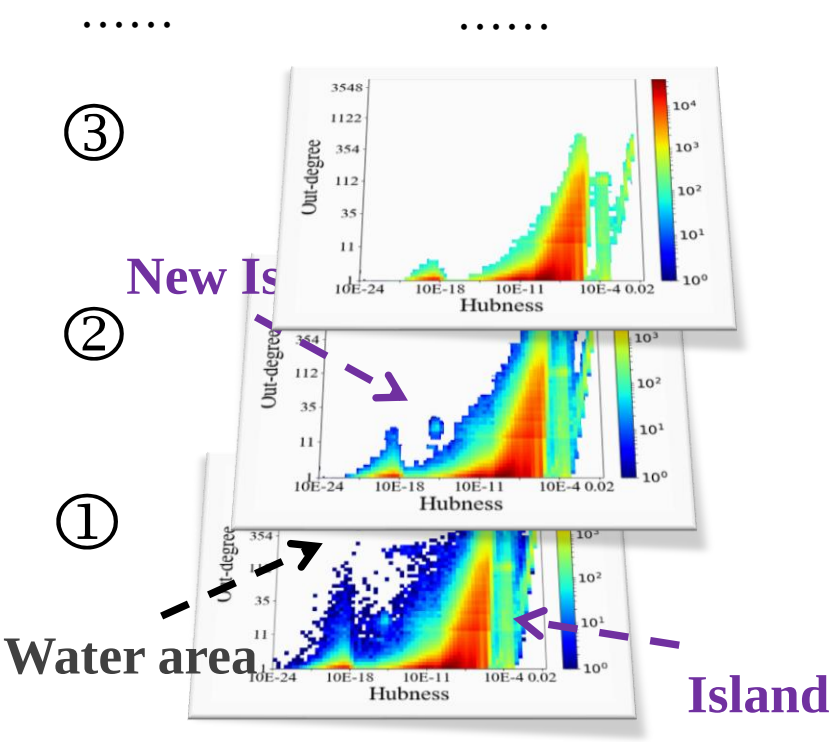


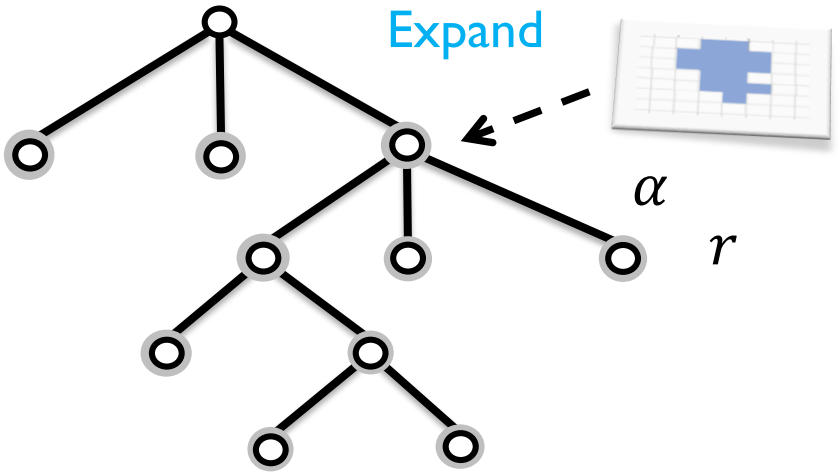


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

   I. **Build** waterlevel tree; II. Tree **refine**;

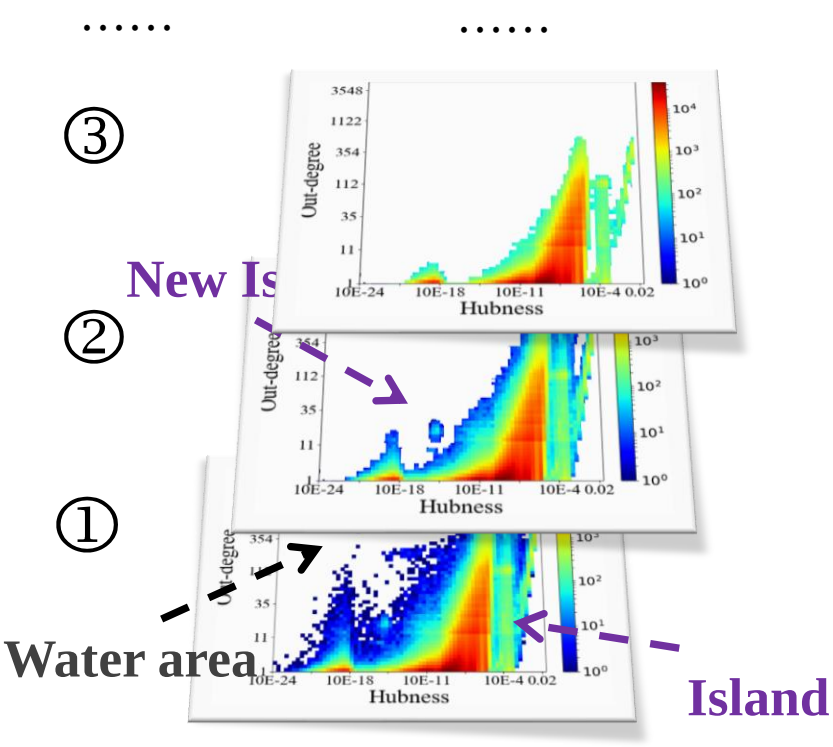


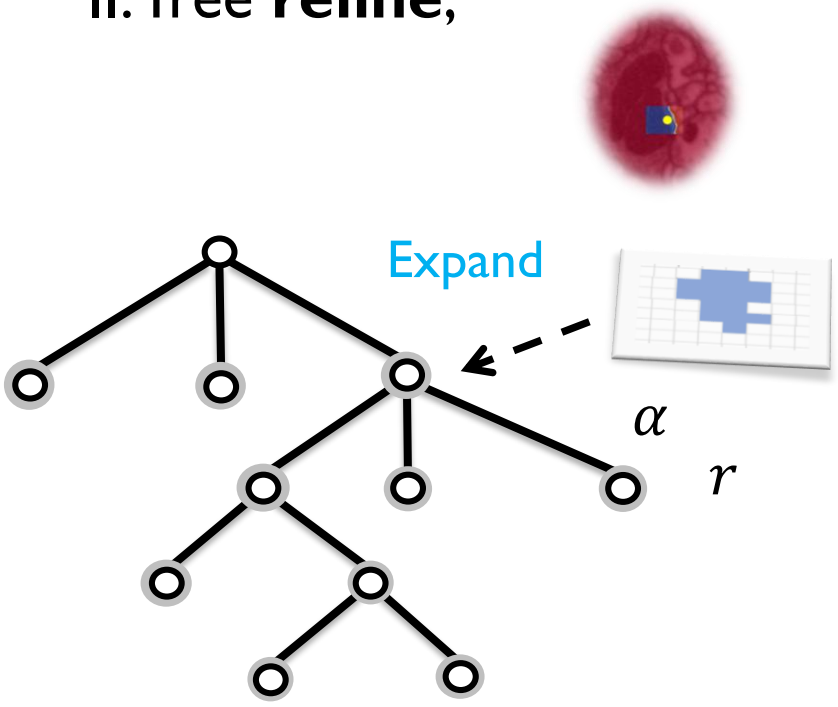


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

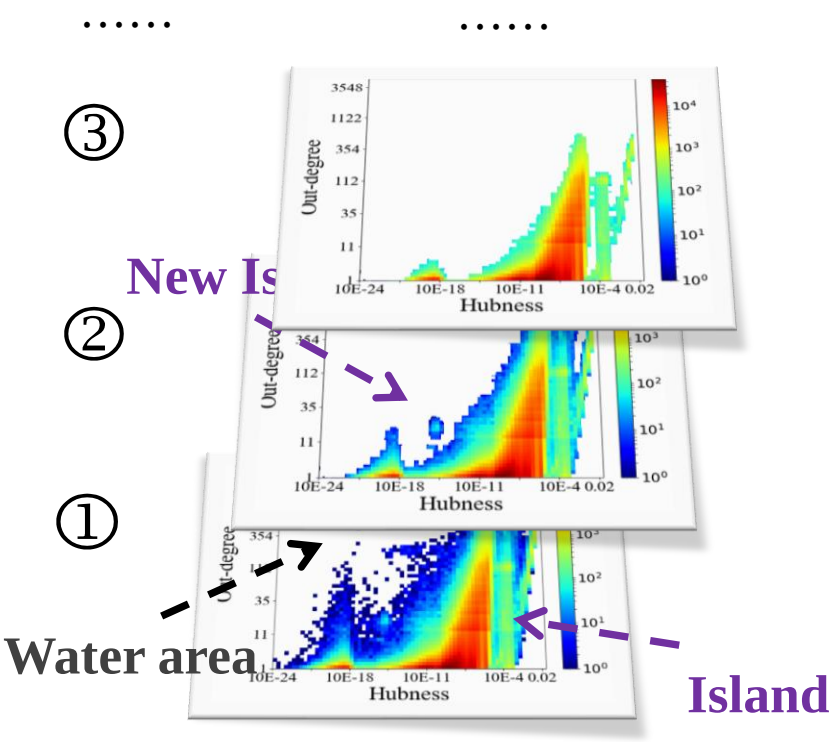


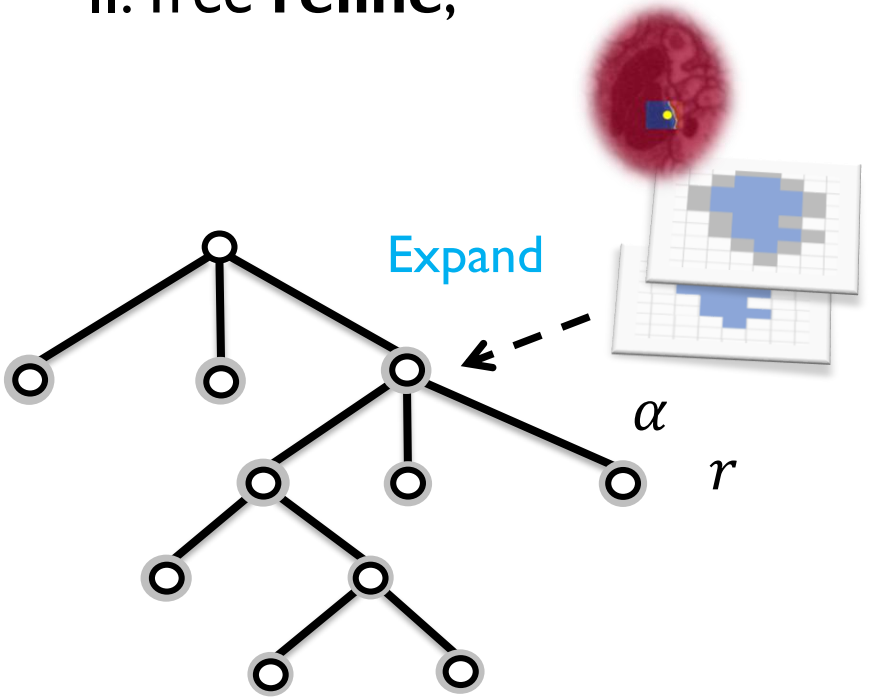


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;

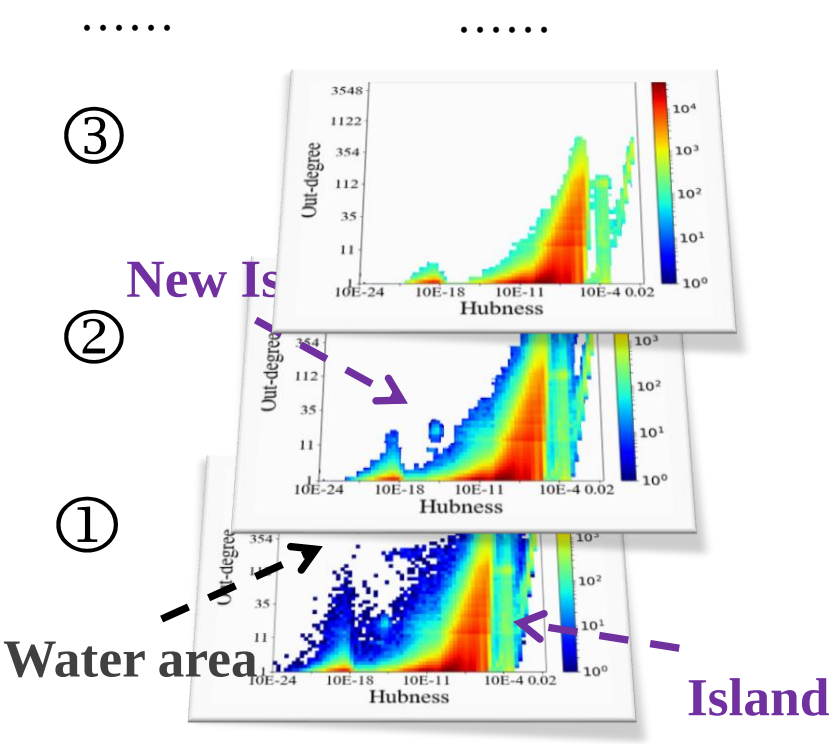


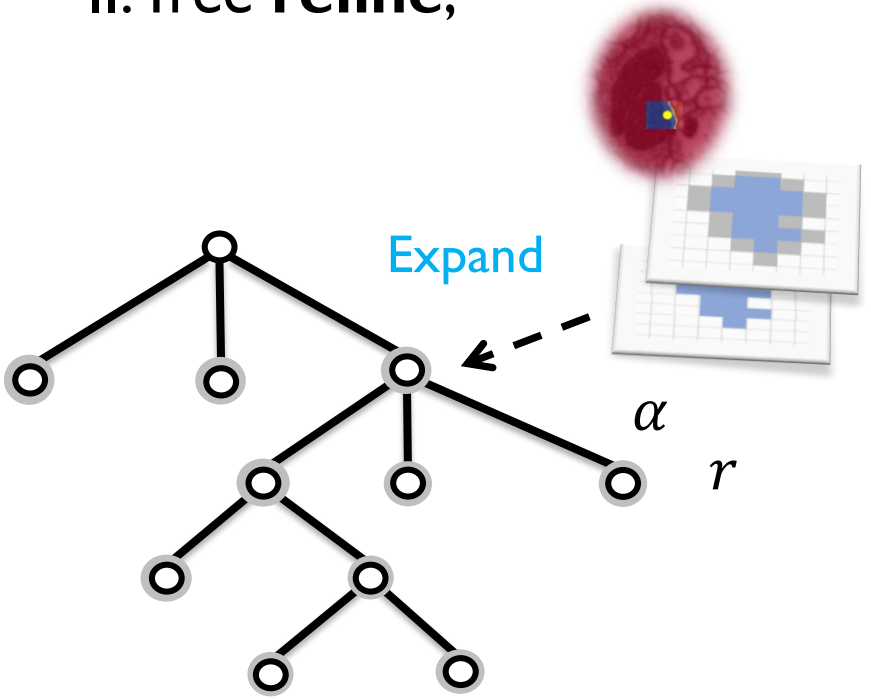


EagleMine algorithm part I

1. Recognize micro-clusters **WATERLEVELTREE**

I. **Build** waterlevel tree; II. Tree **refine**;





Flood fill, morphology operations [Rafael C Gonzalez et al 2007]

EagleMine algorithm part II

2. Summary node group **model**.

EagleMine algorithm part II

2. Summary node group **model**.

   1. **Vocabulary**-based summarization model;

EagleMine algorithm part II

2. Summary node group **model**.

   1. **Vocabulary**-based summarization model;

      use statistical distribution describe node in island.

EagleMine algorithm part II

2. Summary node group **model**.

   1. **Vocabulary**-based summarization model;

      use statistical distribution describe node in island.

      - Vocabulary: distributions y

EagleMine algorithm part II

2. Summary node group **model**.

I. **Vocabulary**-based summarization model;

use statistical distribution describe node in island.

- Vocabulary: distributions $\mathcal{Y}$

E.g. Gaussian, Uniform, Laplacian, Exp. etc.

EagleMine algorithm part II

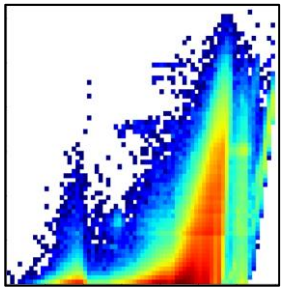
2. Summary node group **model**.

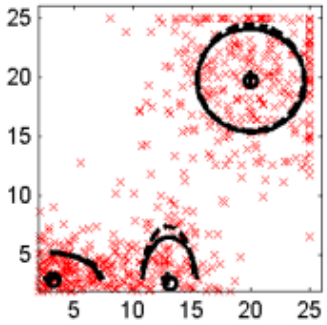
I. **Vocabulary**-based summarization model;

use statistical distribution describe node in island.

- Vocabulary: distributions y

E.g. Gaussian, Uniform, Laplacian, Exp. etc.





EagleMine algorithm part II

2. Summary node group **model**.

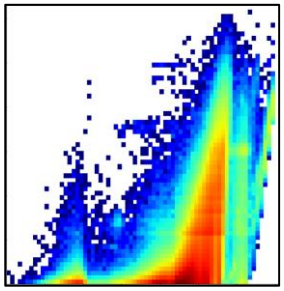
I. **Vocabulary**-based summarization model;

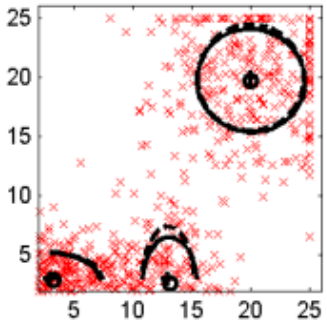
use statistical distribution describe node in island.

- Vocabulary: distributions $\mathcal{Y}$

E.g. Gaussian, Uniform, Laplacian, Exp. etc.

Discretized, Truncated, Multivariate **DTM gaussian**





EagleMine algorithm part II

2. Summary node group **model**.

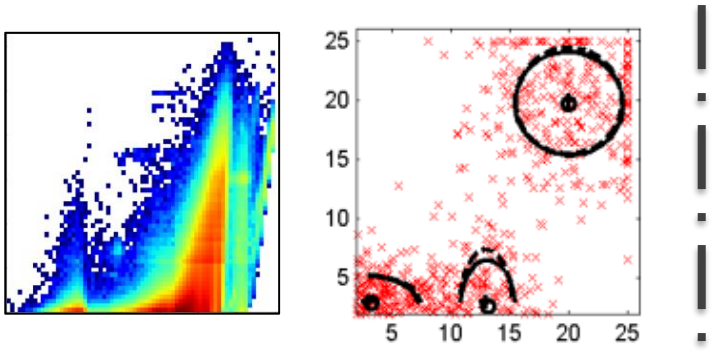
I. **Vocabulary**-based summarization model;

use statistical distribution describe node in island.

- Vocabulary: distributions $\mathcal{Y}$

E.g. Gaussian, Uniform, Laplacian, Exp. etc.

Discretized, Truncated, Multivariate **DTM gaussian**



EagleMine algorithm part II

2. Summary node group **model**.

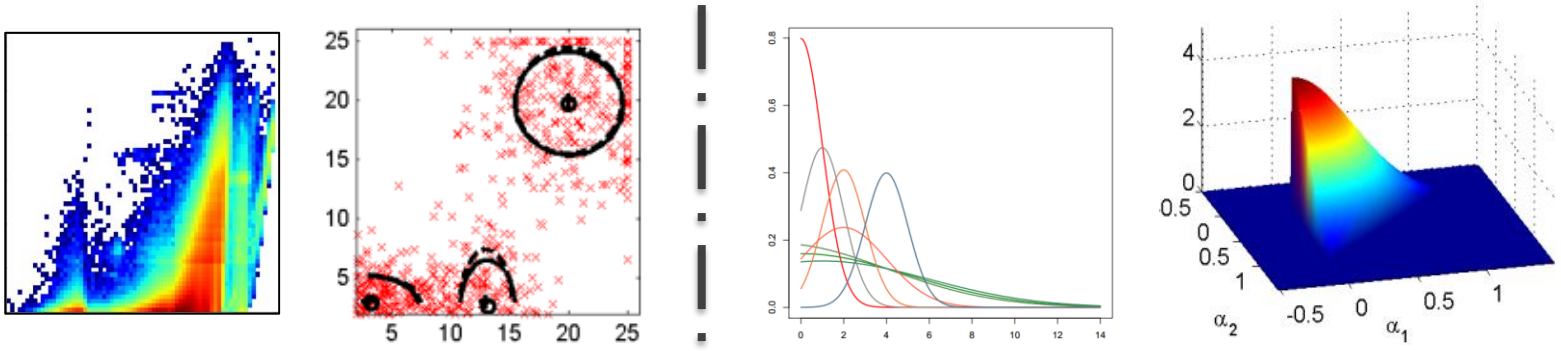
I. **Vocabulary**-based summarization model;

use statistical distribution describe node in island.

- Vocabulary: distributions $\mathcal{Y}$

E.g. Gaussian, Uniform, Laplacian, Exp. etc.

Discretized, Truncated, Multivariate **DTM gaussian**



EagleMine algorithm part II

2. Summary node group **model**.

   1. **Vocabulary**-based summarization model;

      use statistical distribution describe node in island.

      - Vocabulary: distributions y

EagleMine algorithm part II

2. Summary node group **model**.

   1. **Vocabulary**-based summarization model;

      use statistical distribution describe node in island.

      - Vocabulary: distributions y

   Description: For C node groups:

EagleMine algorithm part II

2. Summary node group **model**.

1. **Vocabulary**-based summarization model;

   use statistical distribution describe node in island.

   - Vocabulary: distributions $\mathcal{Y}$

Description: For C node groups:

- Assignments: $\mathcal{S} = \{s_1, \cdots, s_C\}$
- Model parameter: $\Theta = \{\theta_1, \cdots, \theta_C\}$
- Outliers: unassigned bins $\mathcal{O}$

EagleMine algorithm part II

2. Summary node group **model**.

EagleMine algorithm part II

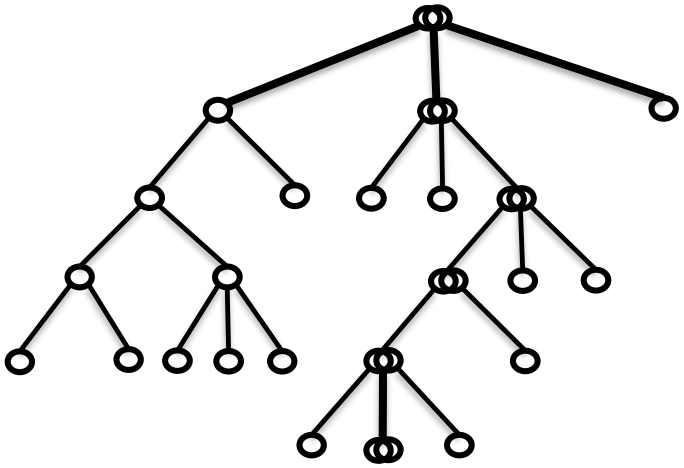
2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

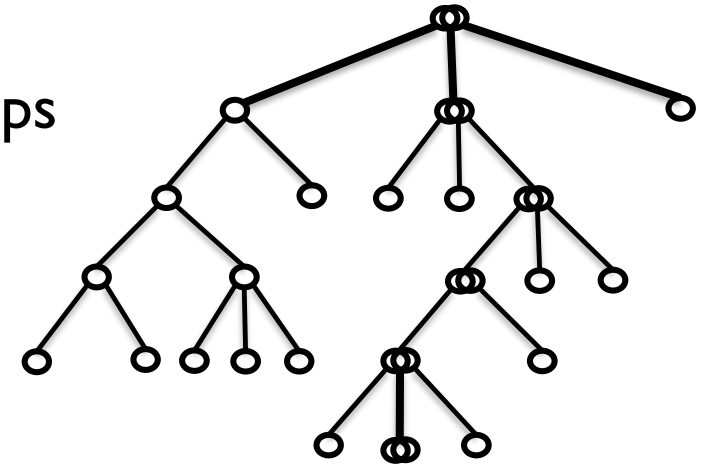


EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



EagleMine algorithm part II

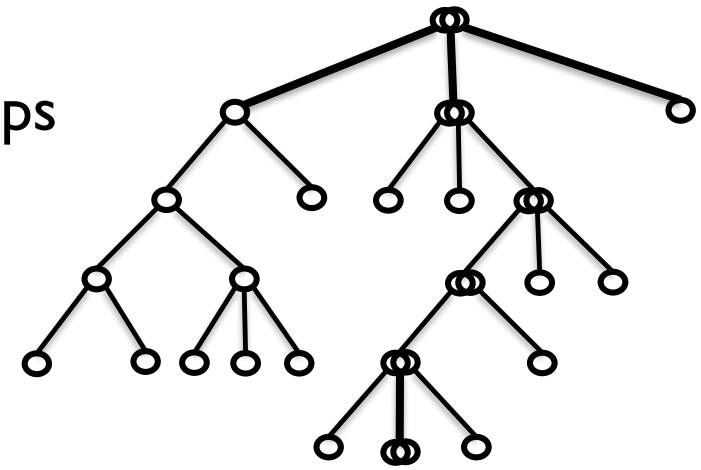
2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test

Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

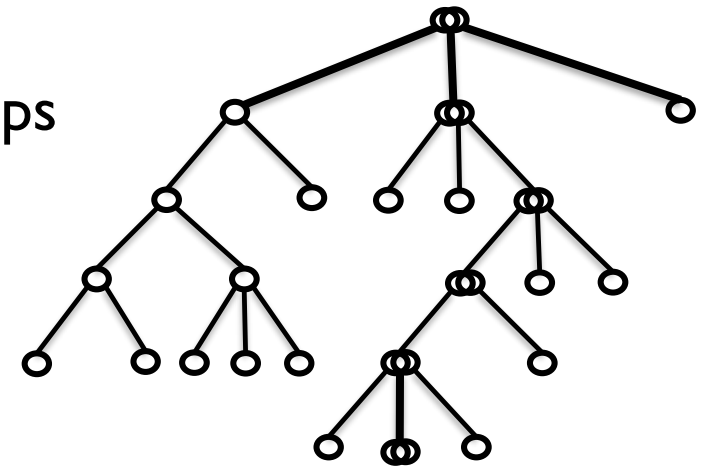


EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

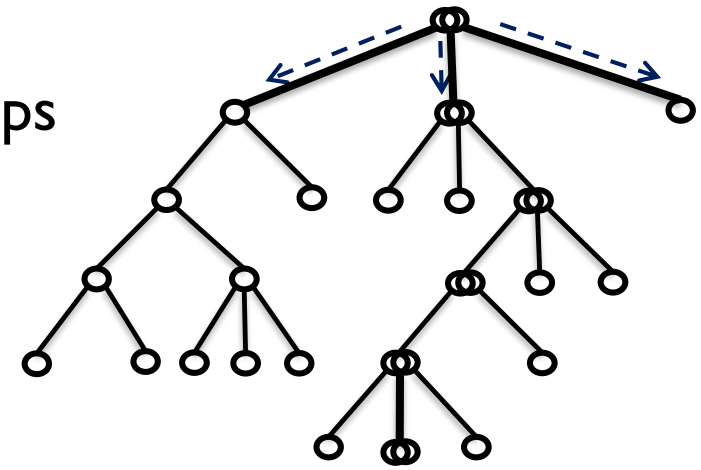
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

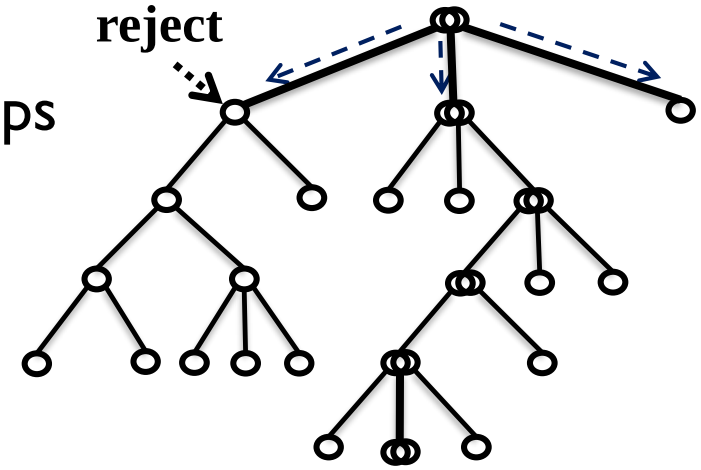
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

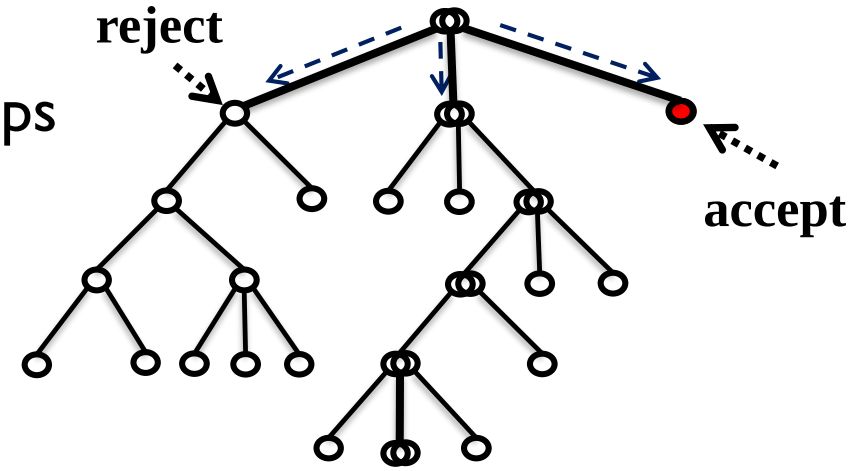
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

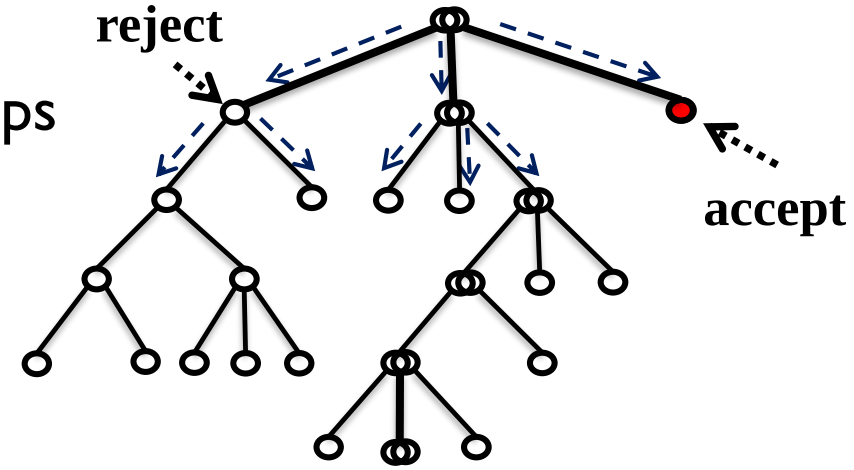
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

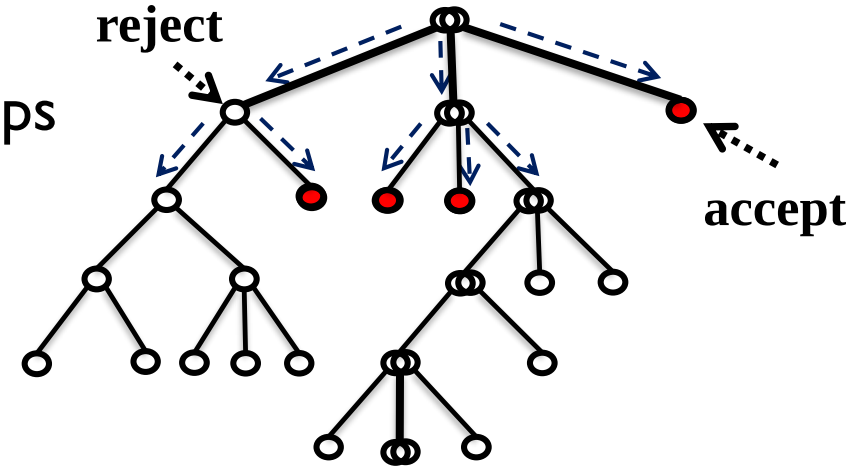
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

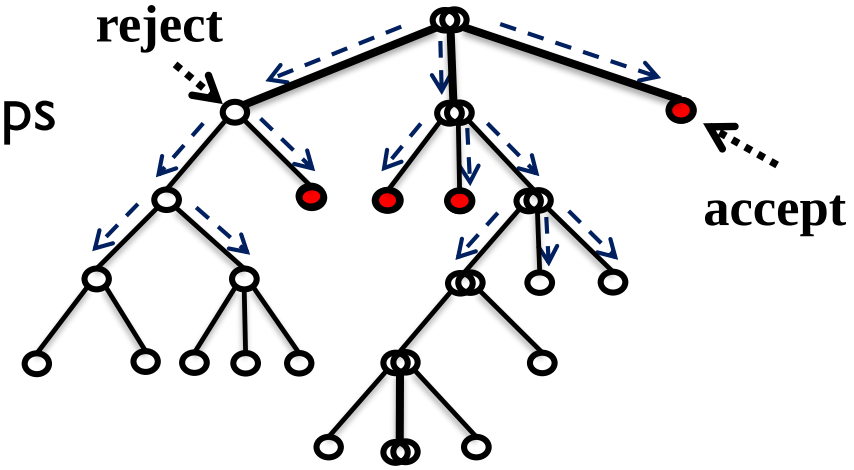
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

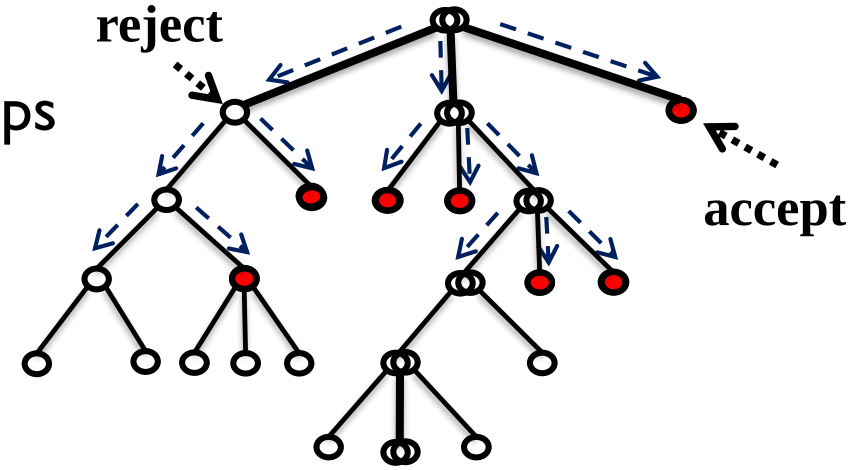
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

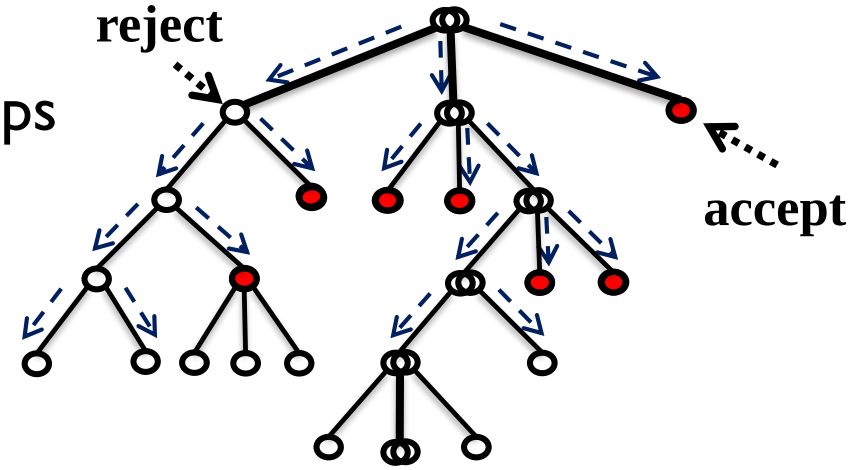
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

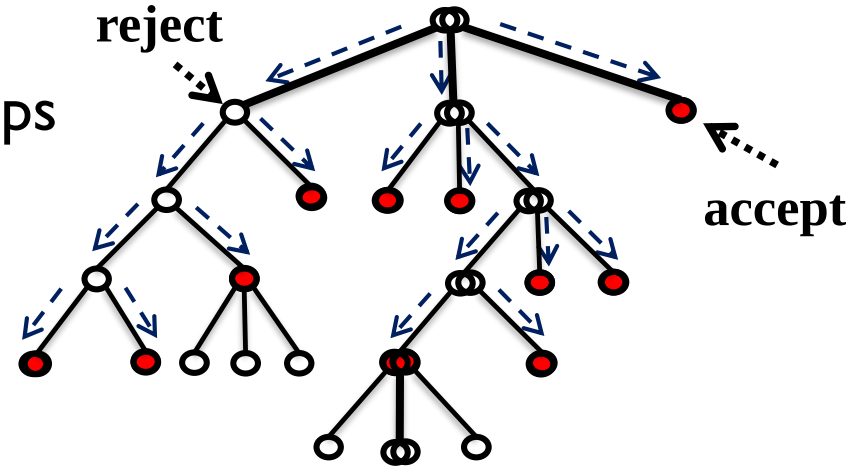
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

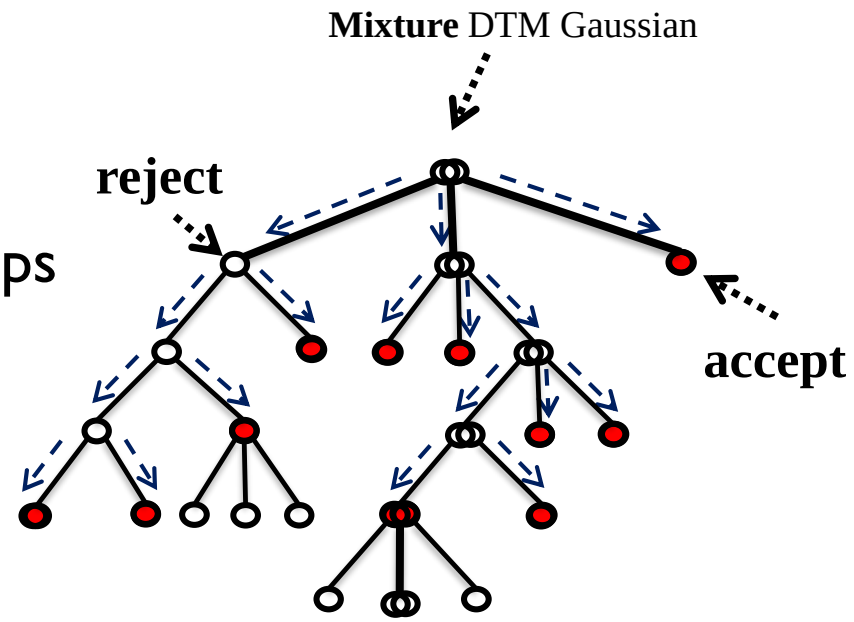
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test



Vocabulary Assignment: s_α

Heuristic rule / Pearson's χ^2

DIST. Parameter: θ_α

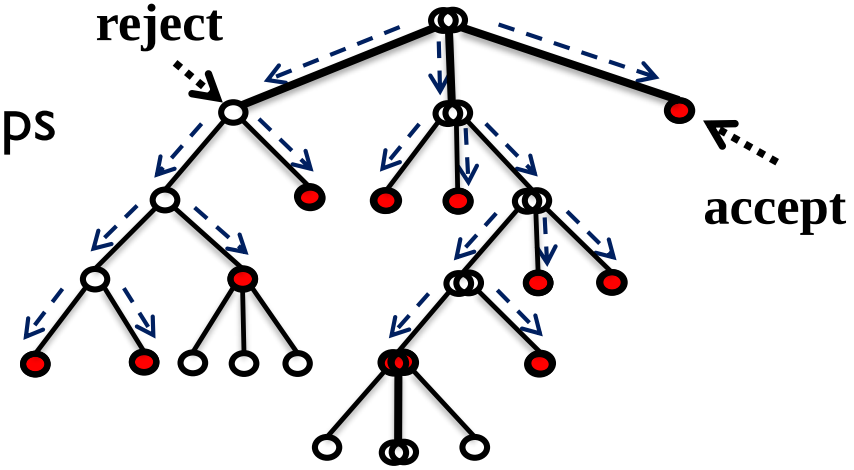
Gradient descent fitting

EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test

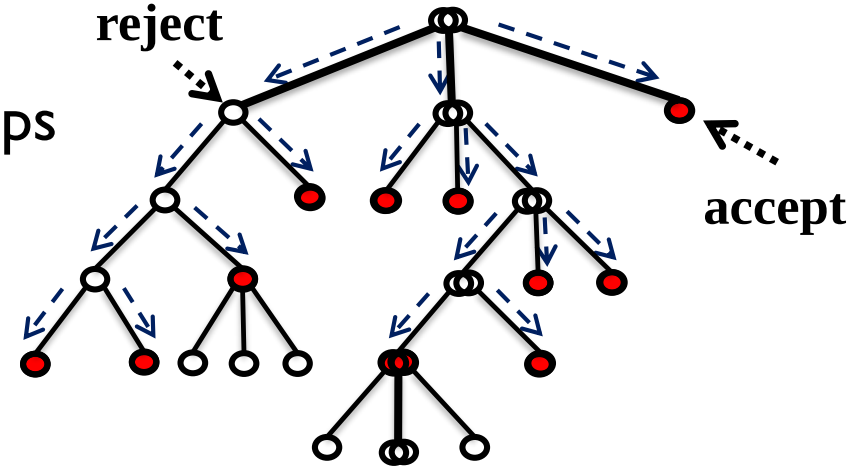


EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test
- Stitch for enhancing

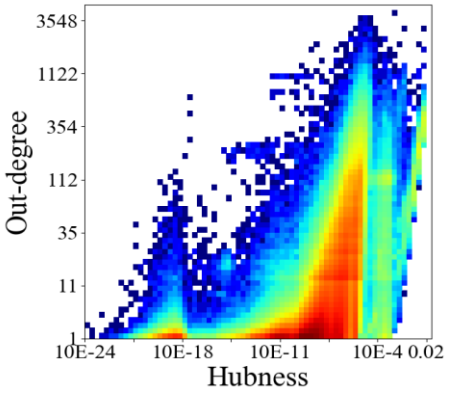


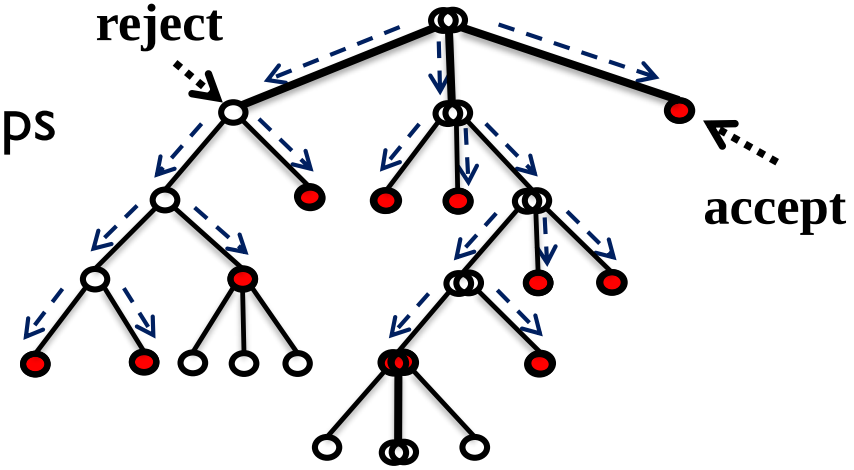
EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test
- Stitch for enhancing



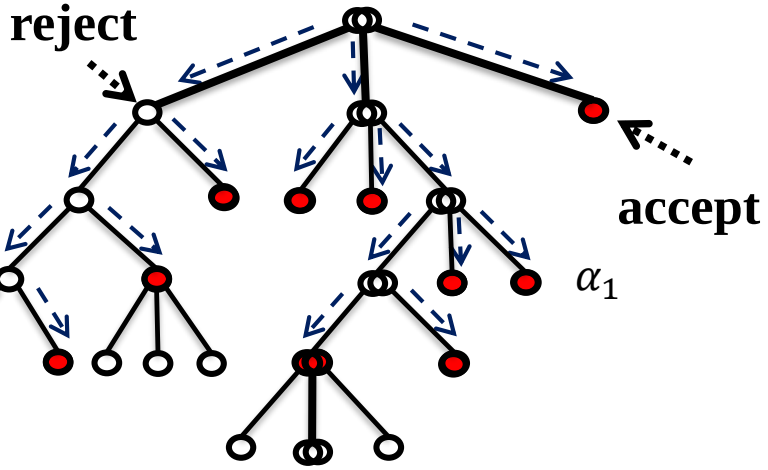


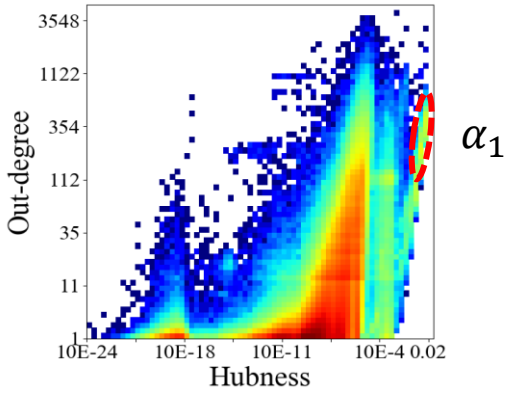
EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test
- Stitch for enhancing



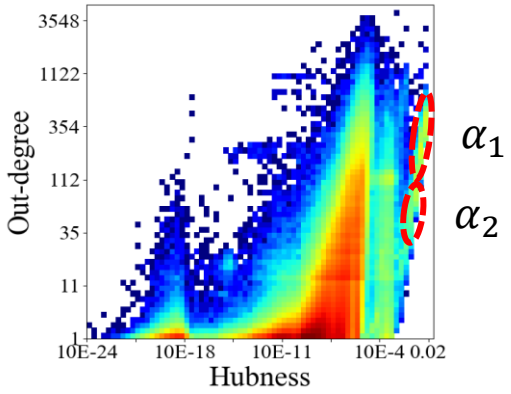


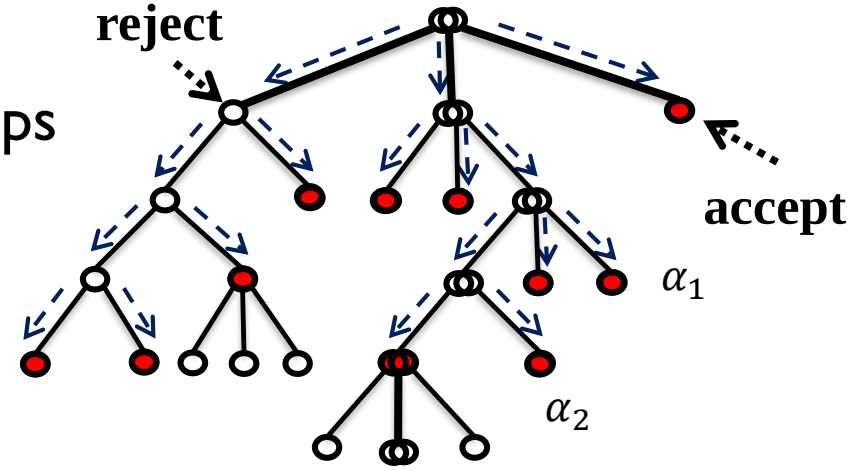
EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test
- Stitch for enhancing



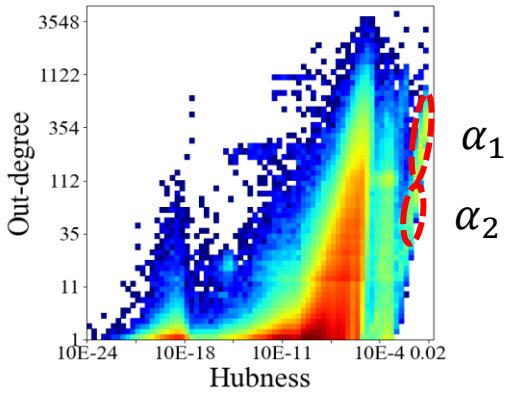


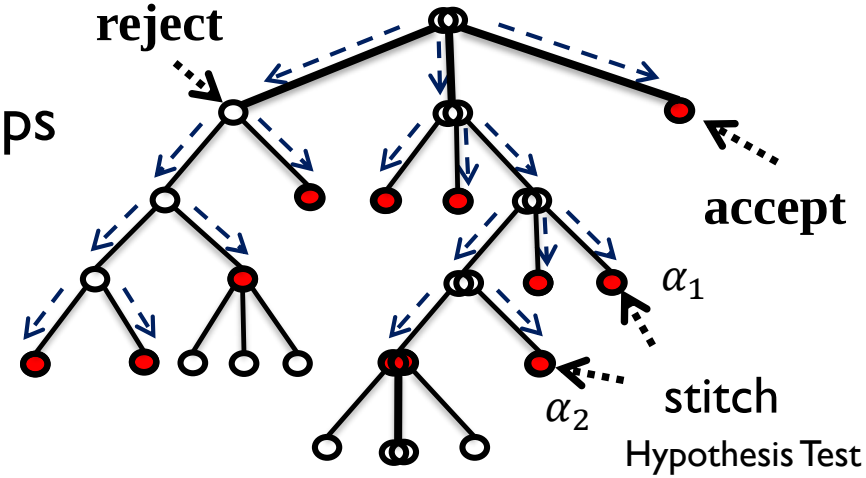
EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

- BFS search
- Determine optimal node groups with Hypothesis Test
- Stitch for enhancing



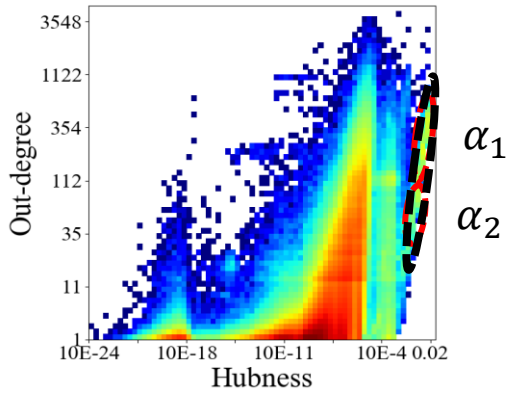


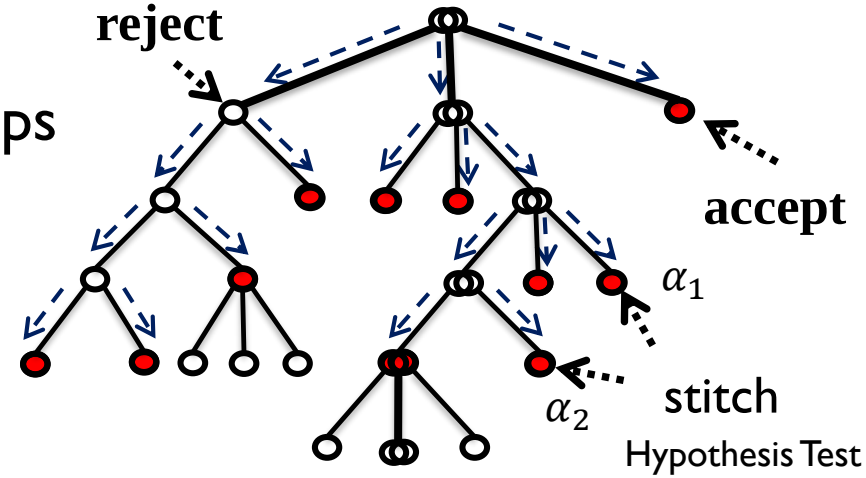
EagleMine algorithm part II

2. Summary node group **model**.

II. WaterLevel Tree **Explore**;

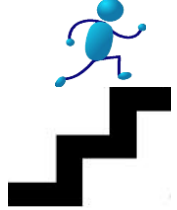
- BFS search
- Determine optimal node groups with Hypothesis Test
- Stitch for enhancing





EagleMine algorithm part III

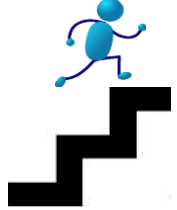
3. Identify suspicious micro-clusters.



EagleMine algorithm part III

3. Identify suspicious micro-clusters.

   Node group **suspiciousness score**:

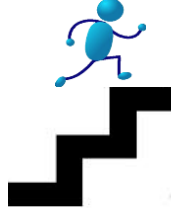


EagleMine algorithm part III

3. Identify suspicious micro-clusters.

   Node group **suspiciousness score**:

      Weighted probability ***KL distance*** with the majority island.



EagleMine algorithm part III

3. Identify suspicious micro-clusters.

Node group **suspiciousness score**:

Weighted probability ***KL distance*** with the majority island.

$$\kappa(\theta_i) = \log \bar{d}_i \cdot \sum_{\boldsymbol{b}} N_i \cdot KL(P_{\theta_i}(\boldsymbol{b}) \,||\, P_{\theta_m}(\boldsymbol{b}))$$

EagleMine algorithm part III

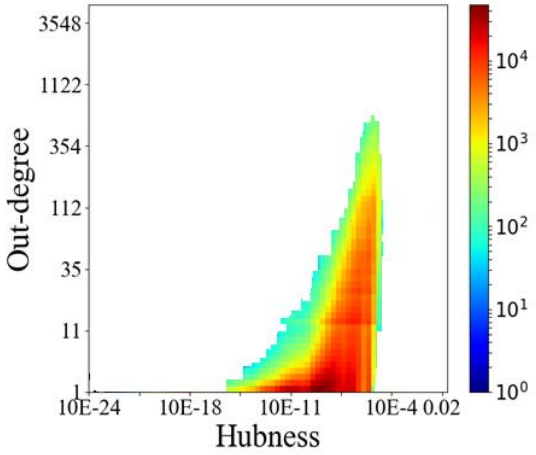
3. Identify suspicious micro-clusters.

Node group **suspiciousness score**:

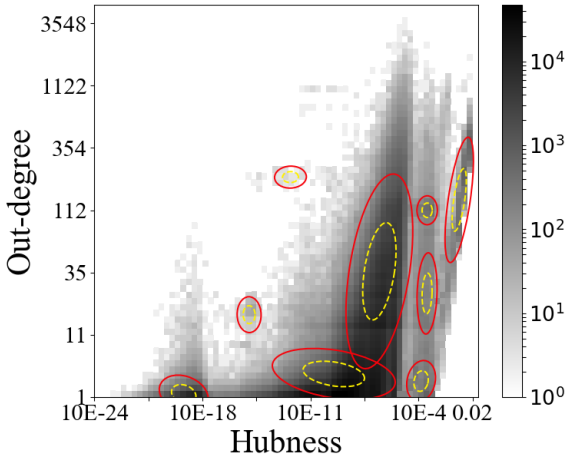
Weighted probability ***KL distance*** with the majority island.

$$\kappa(\theta_i) = \log \bar{d}_i \cdot \sum_{\boldsymbol{b}} N_i \cdot KL(P_{\theta_i}(\boldsymbol{b}) \,||\, P_{\theta_m}(\boldsymbol{b}))$$

The majority island – Normal nodes



DTM Gaussian Description



Outline

- Introduction
- Overview
- Proposed Method: **EagleMine**
- Experiments <<
- Conclusion

Data sets

Table I. Dataset statistic information

	# of nodes	# of edges	Content
Amazon rating	(2.14M, 1.23M)	5.84M	Rate
Android	(1.32M, 61.27K)	2.64M	Rate
BeerAdvocate	(33.37K, 65.91K)	1.57M	Rate
Yelp	(686K, 85.54K)	2.68M	Rate
Tagged	(2.73M, 4.65M)	150.8M	Anonymized Links
Youtube	(3.22M, 3.22M)	9.37M	Who-follow-who
Sina weibo	(2.75M, 8.08M)	50.1M	User-retweet-msg

Data sets

Table I. Dataset statistic information

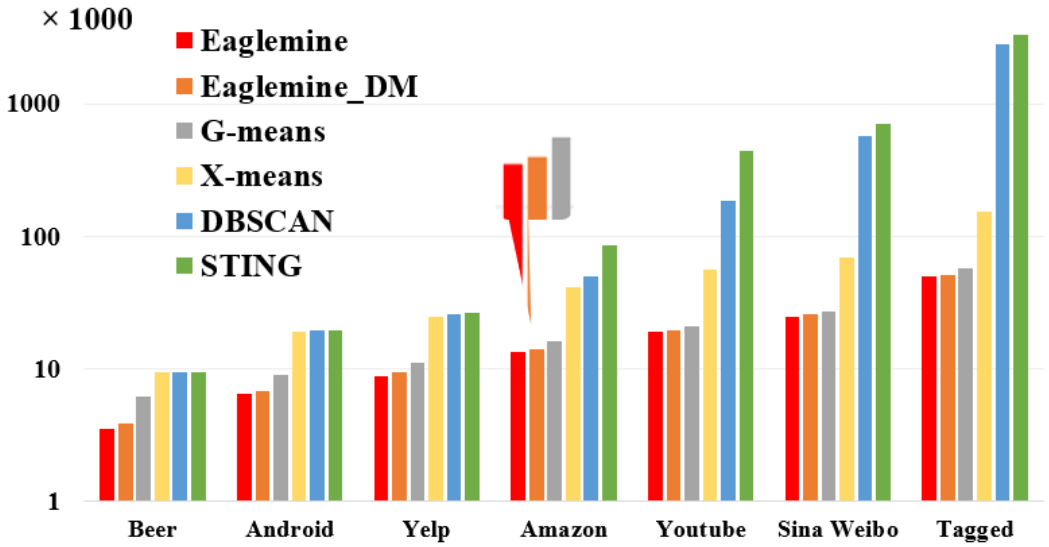
	# of nodes	# of edges	Content
Amazon rating	(2.14M, 1.23M)	5.84M	Rate
Android	(1.32M, 61.27K)	2.64M	Rate
BeerAdvocate	(33.37K, 65.91K)	1.57M	Rate
Yelp	(686K, 85.54K)	2.68M	Rate
Tagged	(2.73M, 4.65M)	150.8M	Anonymized Links
Youtube	(3.22M, 3.22M)	9.37M	Who-follow-who
Sina weibo	(2.75M, 8.08M)	50.1M	User-retweet-msg

Related publications [J. McAuley, R. Pandey & J. Leskovec, KDD'15], [J. McAuley & J. Leskovc, WWW'13], [A. Mislove, M. Marcon et al, SIGCOM'07], [S. Fakraei et al KDD'15]
Most of datasets are public available at https://snap.stanford.edu/data/index.html

Exp 1. Quantitative evaluation of summarization

Measure: MDL (minimum Description Length)

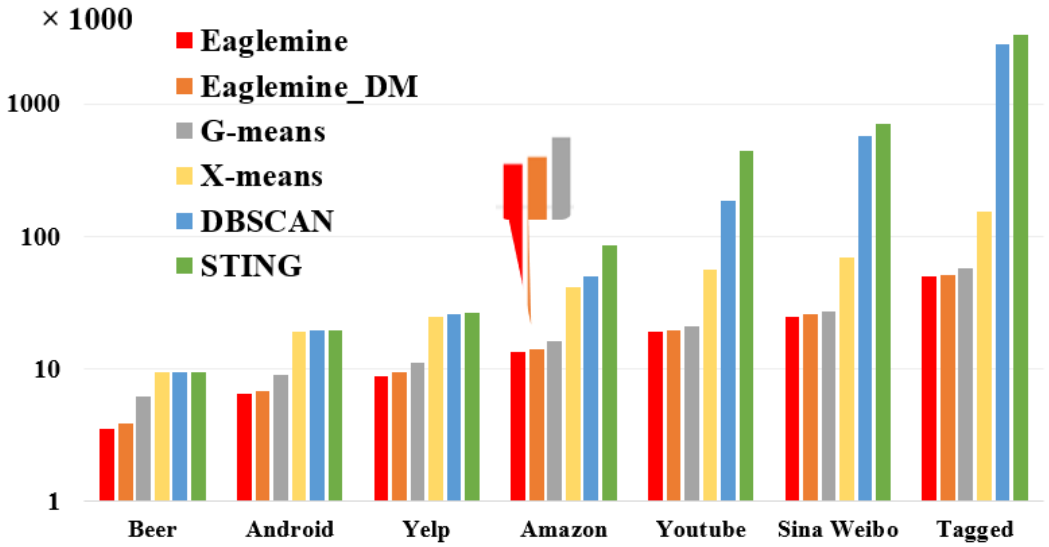
The best model has shortest MDL.



Exp 1. Quantitative evaluation of summarization

Measure: MDL (minimum Description Length)

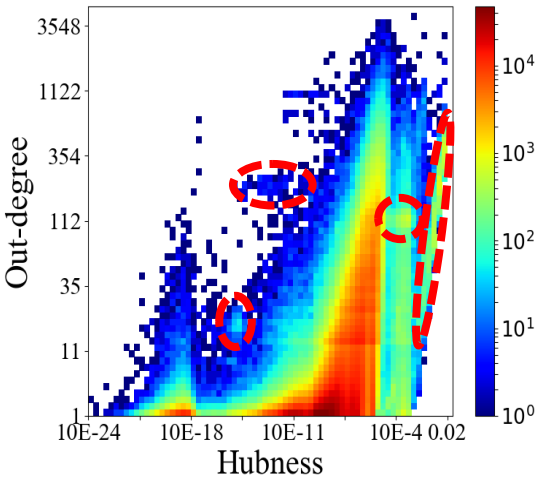
The best model has shortest MDL.

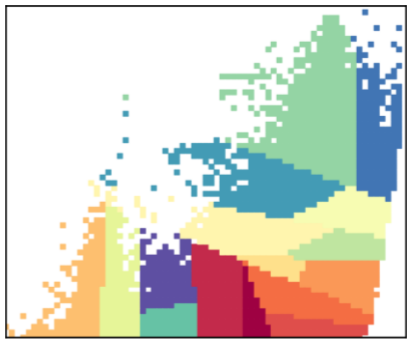


EagleMine achieves concise summarization of graph.

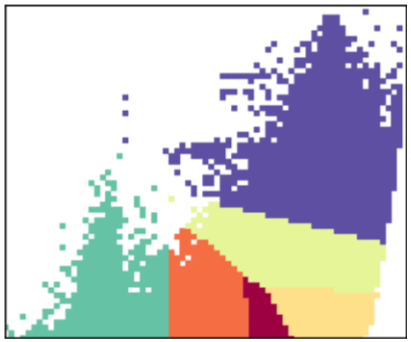
Exp 2. Qualitative of identifying micro-clusters

Sina_weibo user: out-degree vs. hubness

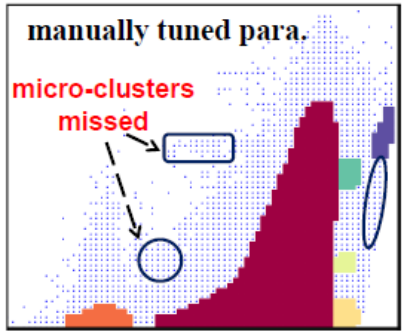




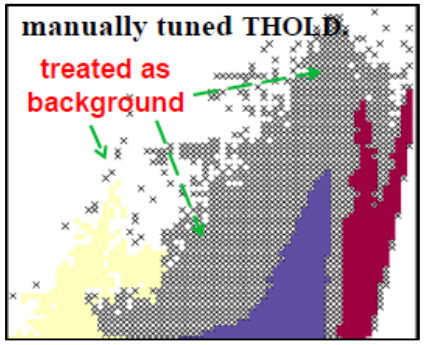
Gmeans



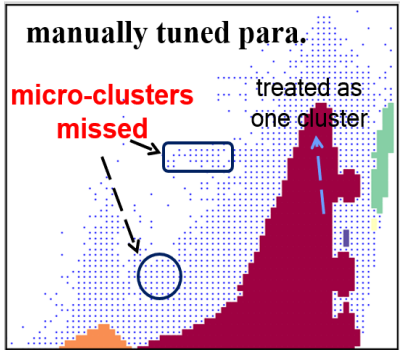
Xmeans



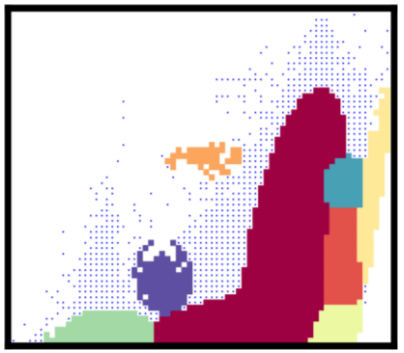
DBSCAN



Watershed



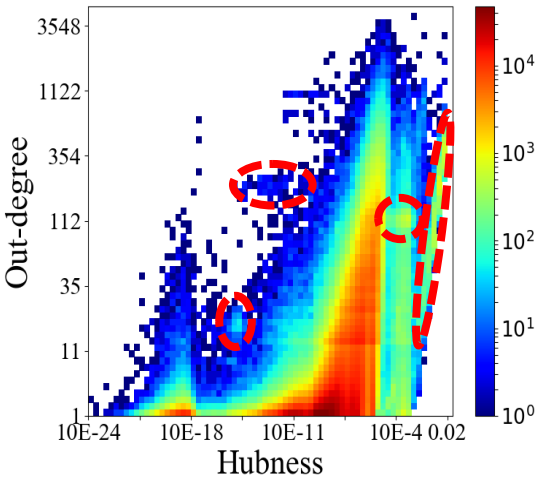
STING

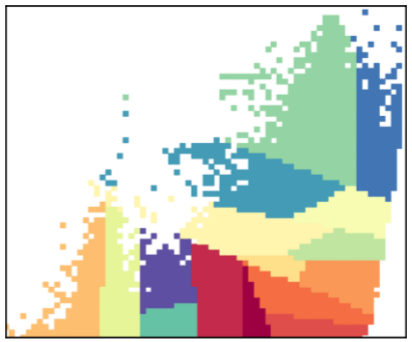


EagleMine

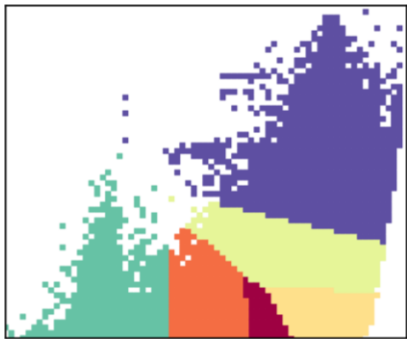
Exp 2. Qualitative of identifying micro-clusters

Sina_weibo user: out-degree vs. hubness

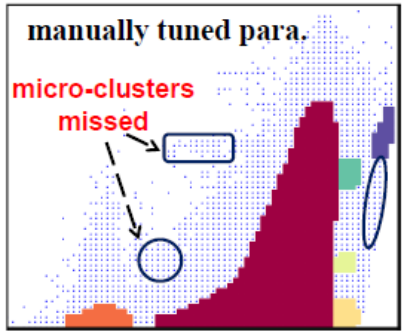




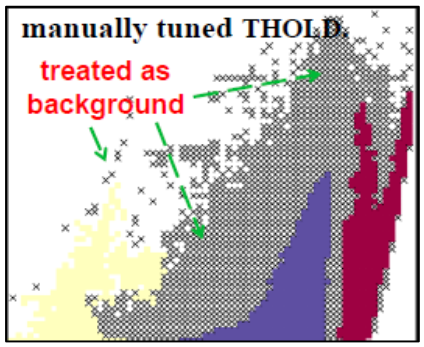
Gmeans



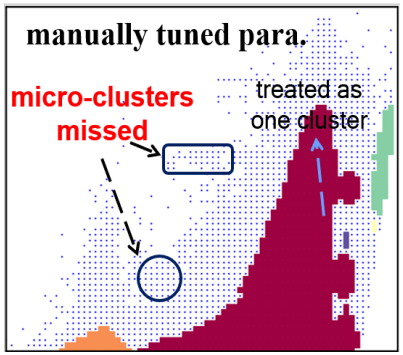
Xmeans



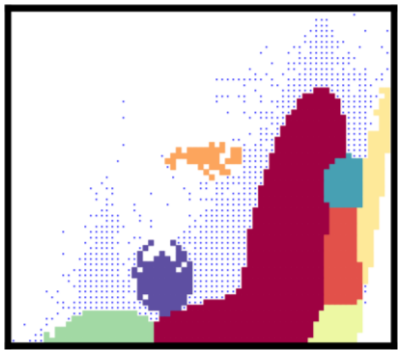
DBSCAN



Watershed



STING

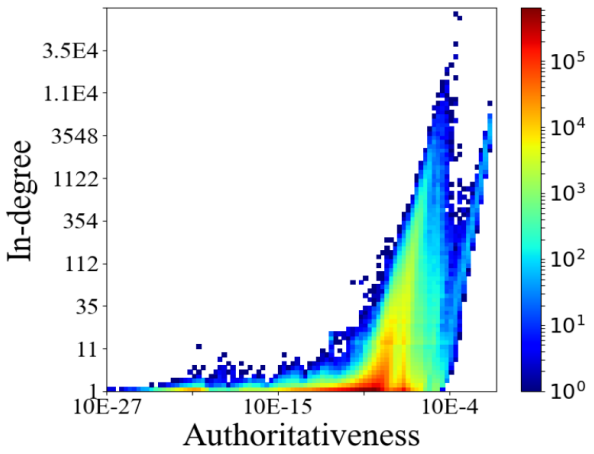


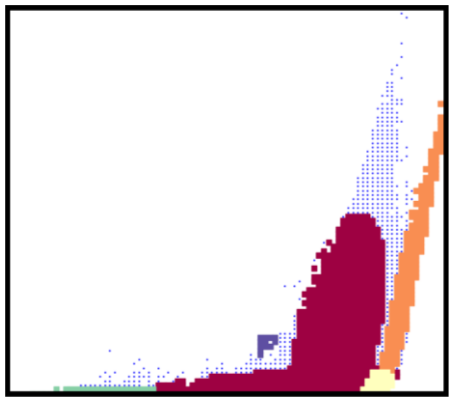
EagleMine

Watershed [L.Vincent PAMI'91], DBSCAN [M. Ester et al KDD'96], STING [W.Wang et al VLDB'97]

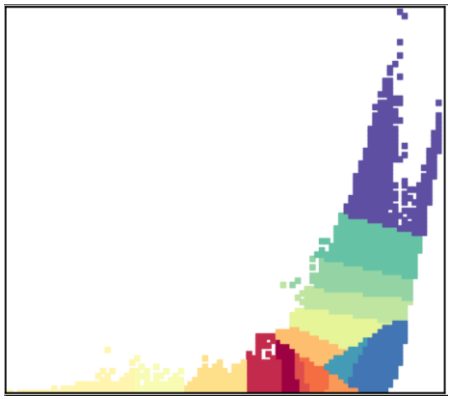
Exp 2. Qualitative of identifying micro-clusters

Sina_weibo msg: in-degree vs. authority

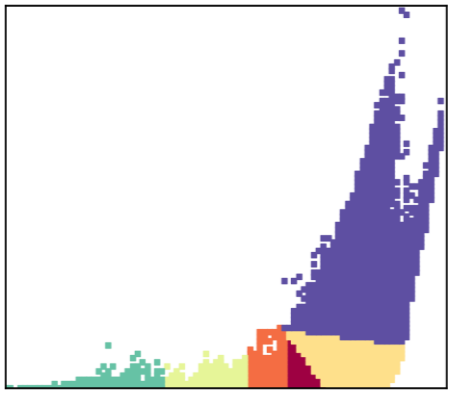




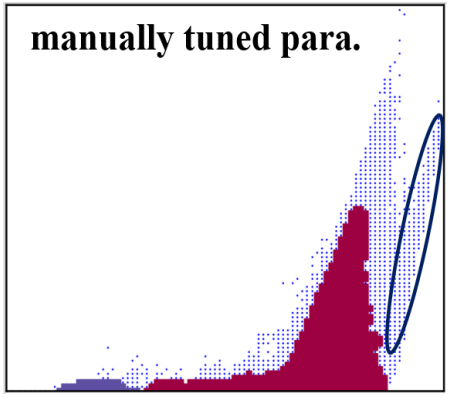
EagleMine



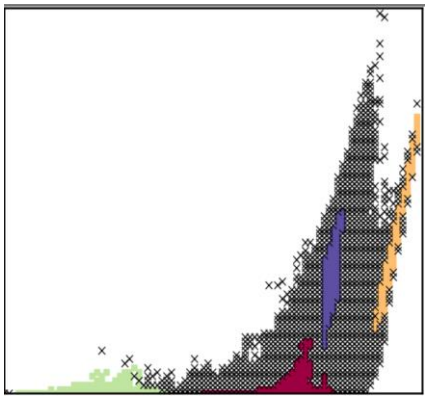
Gmeans



Xmeans



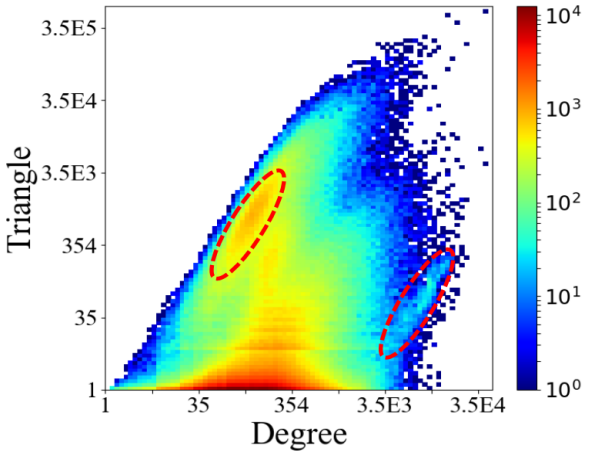
STING

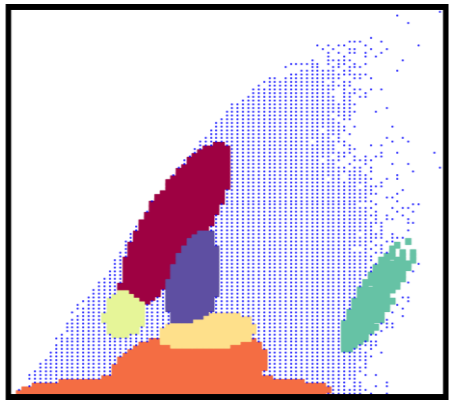


Watershed

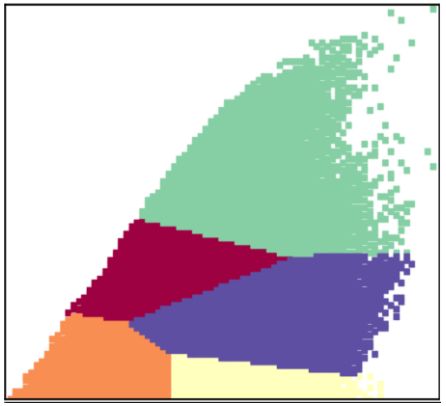
Exp 2. Qualitative of identifying micro-clusters

Tagged friendship: #triangle vs. degree

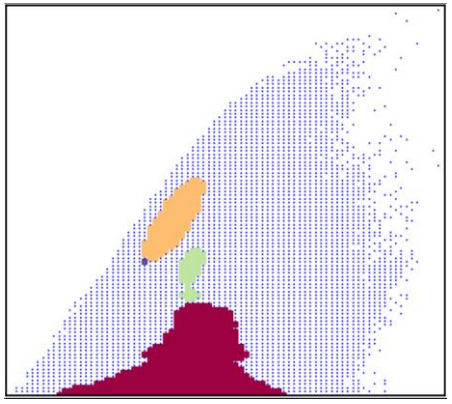




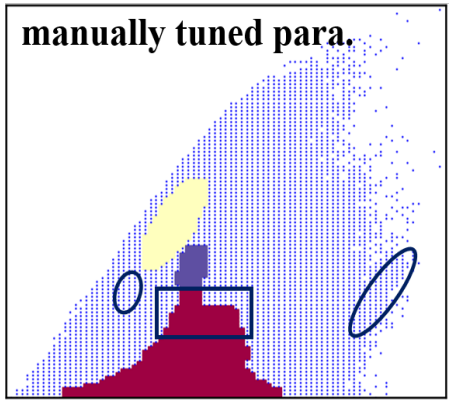
EagleMine



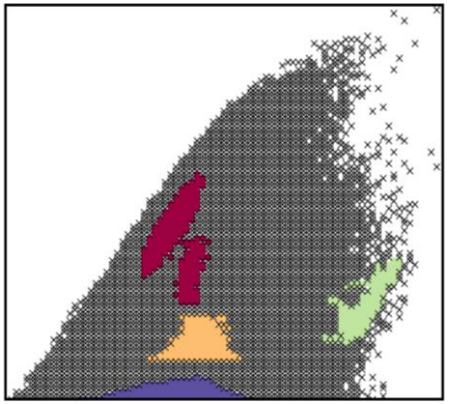
Xmeans



STING



DBSCAN



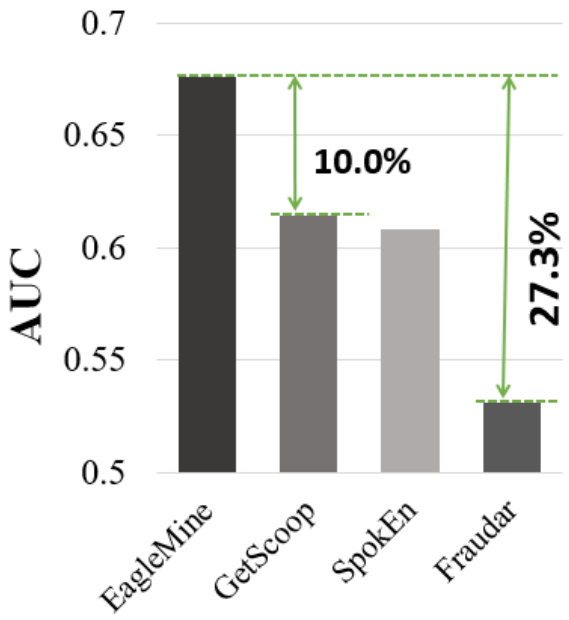
Watershed

Exp 3. Anomaly detection

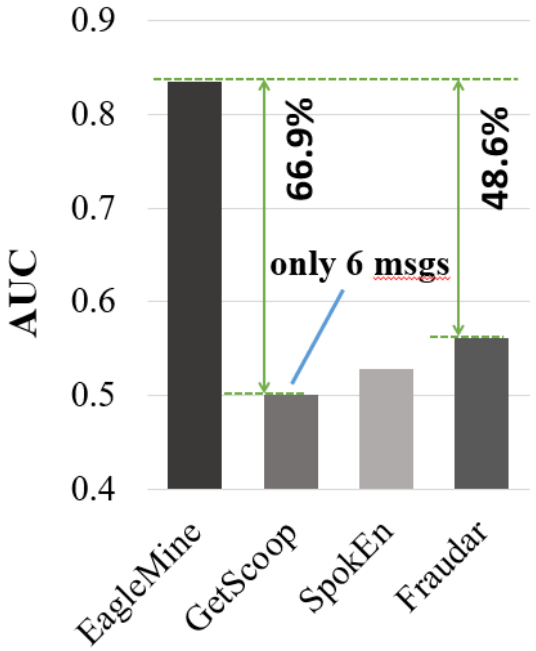
Sina Weibo dataset: (user-retweet-msg)

user: 2.75M
msg: 8.08M
edge: 50.1M

Time:
Nov. 1-30, 2013



Suspicious Users



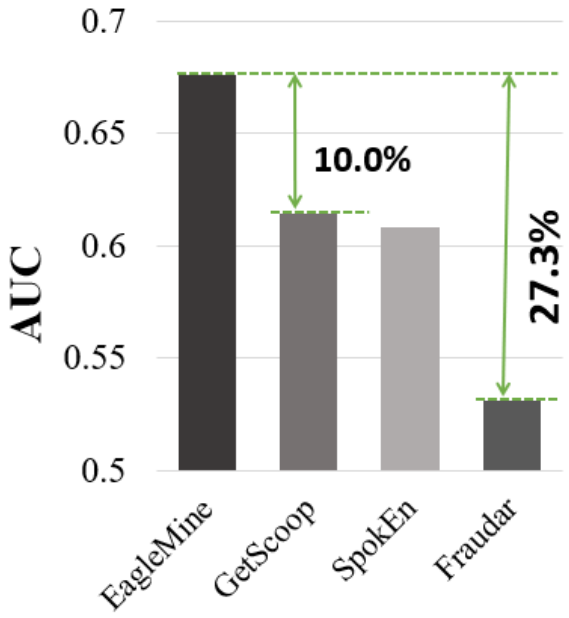
Suspicious Msgs

Exp 3. Anomaly detection

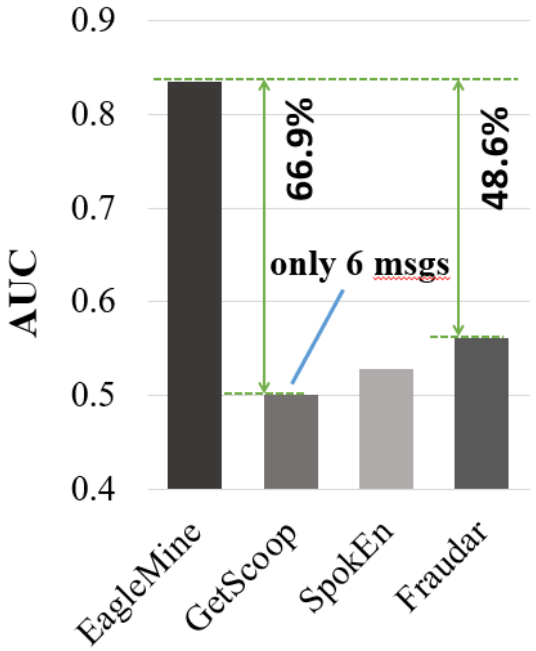
Sina Weibo dataset: (user-retweet-msg)

user: 2.75M
msg: 8.08M
edge: 50.1M

Time:
Nov. 1-30, 2013



Suspicious Users



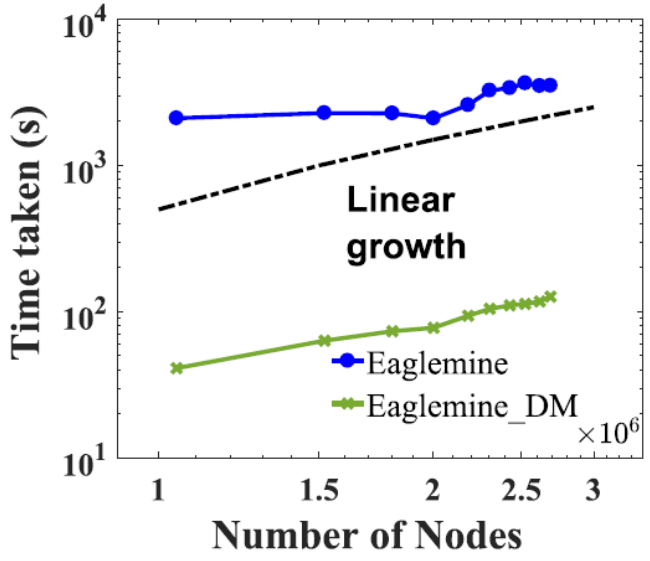
Suspicious Msgs

GetScoop [M. Jiang et al PAKDD'14], SpokEn [B.Prakash PAKDD'10], Fraudar [B. Hooi et al KDD'16]

Exp 4. Scalability test

EagleMine time complexity:

$$O(\frac{\log h_{max}}{\rho} \cdot nnz(\mathcal{H}) + C \cdot T \cdot nnz(\mathcal{H}))$$



Time vs. # node of graph

h_{max}: maximum of H;

ρ: water-level step size;

C: # of node group;

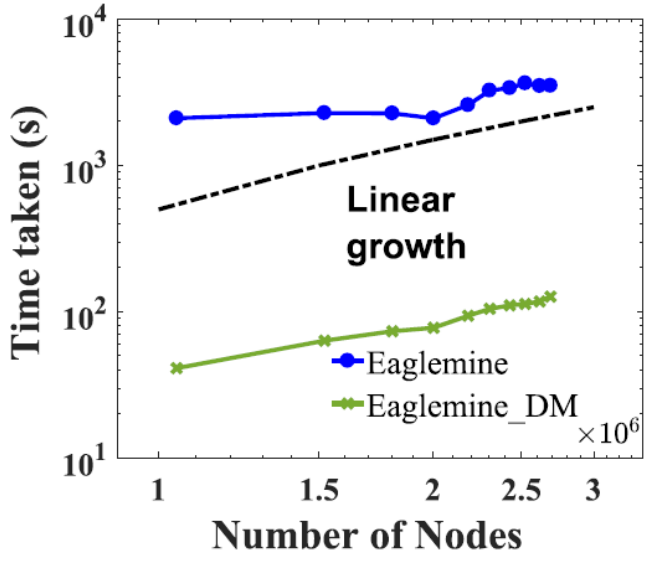
nnz(H): # of non-zero values of H;

T: # of iteration of learning;

Exp 4. Scalability test

EagleMine time complexity:

$$O\left(\frac{\log h_{max}}{\rho} \cdot \boxed{nnz(\mathcal{H})} + C \cdot T \cdot \boxed{nnz(\mathcal{H})}\right)$$



Time vs. # node of graph

h_{max}: maximum of H;

ρ: water-level step size;

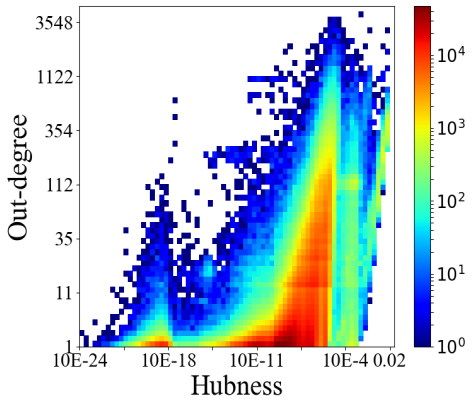
C: # of node group;

nnz(H): # of non-zero values of H;

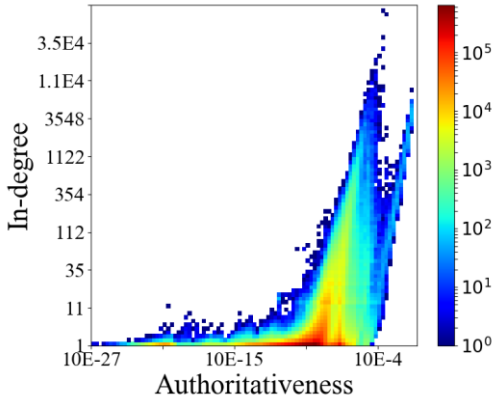
T: # of iteration of learning;

Exp 5. Network pattern discovery

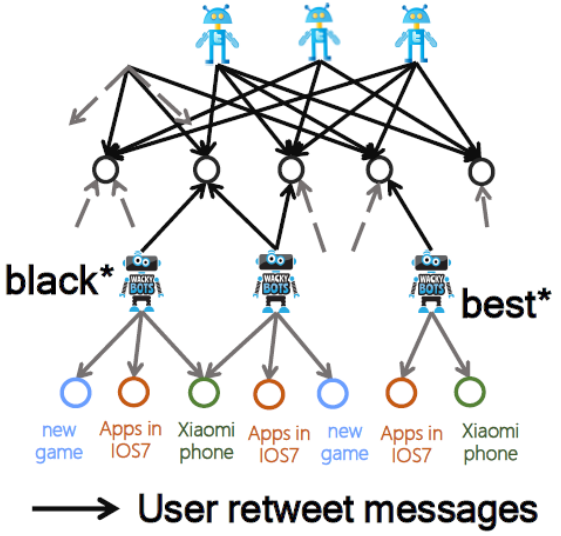
User-Retweet-Msg Graph



user



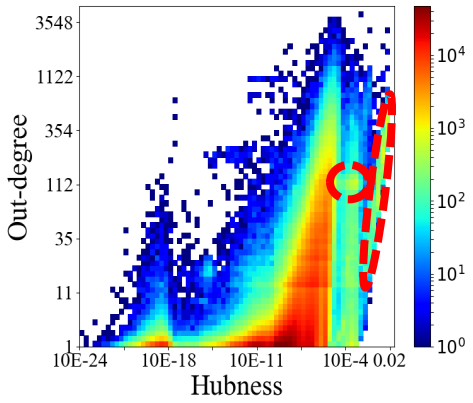
msg



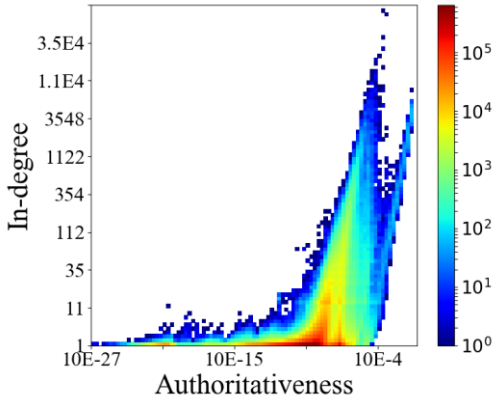
Jellyfish structure anomaly pattern

Exp 5. Network pattern discovery

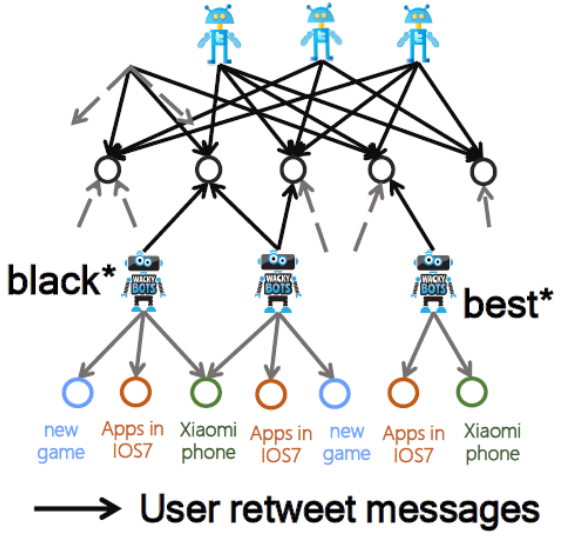
User-Retweet-Msg Graph



user



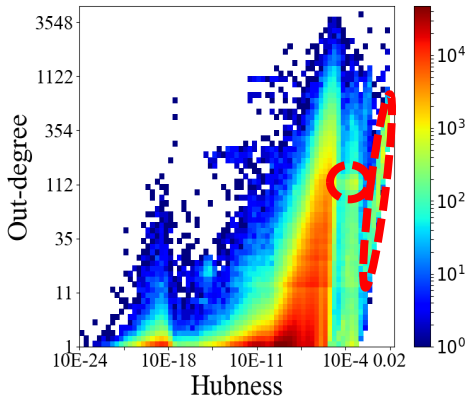
msg



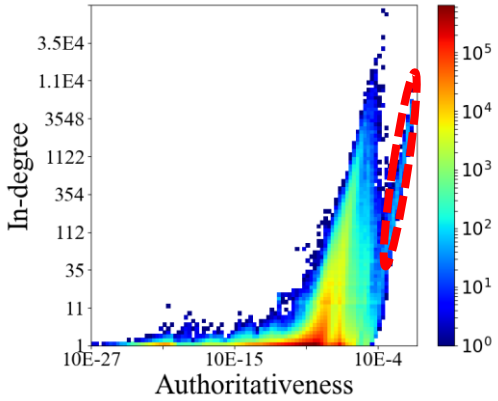
Jellyfish structure anomaly pattern

Exp 5. Network pattern discovery

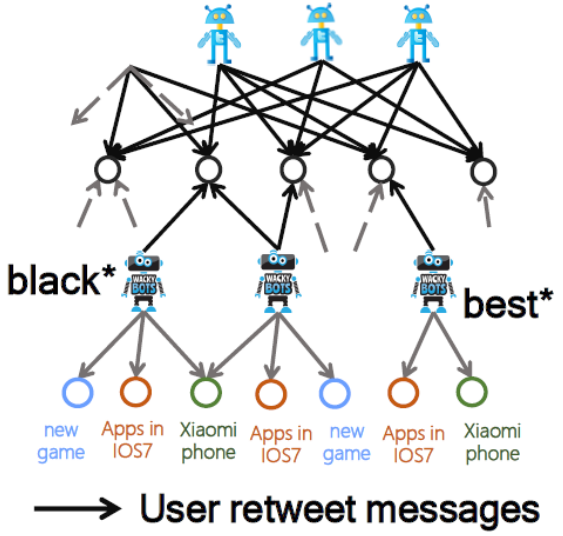
User-Retweet-Msg Graph



user



msg



Jellyfish structure anomaly pattern

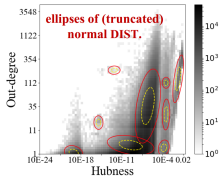
Outline

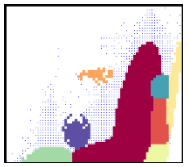
- Introduction
- Overview
- Proposed Method: **EagleMine**
- Experiments
- Conclusion <<

Conclusion

- **EagleMine**: vision-guided large graph mining

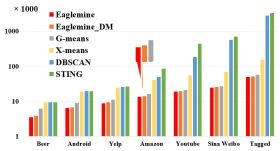
☑ *Automated summarization:*

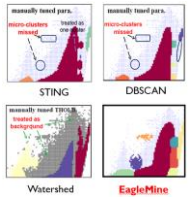




graph nodes distribution in correlated feature spaces

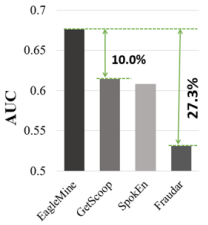
☑ *Effectiveness:* describe and detect node groups,

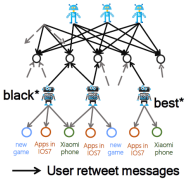




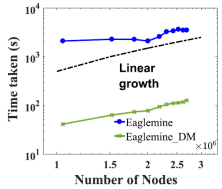
and get visually sense-making results

☑ *Anomaly detection:* spot explainable anomaly patterns





☑ *Scalability:* runs linearly in # of graph node



☑ *Reproducible:* open source code & data

GitHub https://github.com/wenchieh/eaglemine

Reference

- **[L. Wikinson et al, 05]** Leland Wilkinson, Anushka Anand, and Robert Grossman. Graph-theoretical scagnostics. Proceedings - IEEE Symposium on Information Visualisation, INFOVIS'05.
- **[J. W. Tukey et al, 85]** J. W. Tukey and P. A. Tukey. Computer graphics and exploratory data analysis: An introduction. Nat Computer Graphics Association.
- **[L. Vincent 91]** Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. PAMI'91.
- **[M. Ester et al 96]** Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density based algorithm for discovering clusters in large spatial databases with noise. In KDD'96.
- **[W. Wang et al 97]** Wei Wang, Jiong Yang, Richard Muntz, et al. STING: A statistical information grid approach to spatial data mining. In VLDB'97.
- **[M. Jiang et al 14]** Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. Inferring strange behavior from connectivity pattern in social networks. In PAKDD'14.
- **[B.Prakash 10]** B Aditya Prakash, Ashwin Sridharan, Mukund Seshadri, Sridhar Machiraju, and Christos Faloutsos. Eigenspokes: Surprising patterns and scalable community chipping in large graphs. PAKDD'10.
- **[B. Hooi et al. 16]** Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. Fraudar: Bounding graph fraud in the face of camouflage. In SIGKDD'16.
- **[J. DiCarlo et al.12]** James J. DiCarlo, Davide Zoccolan, and Nicole C. Rust. 2012. How Does the Brain Solve Visual Object Recognition? Neuron'12.
- **[Rafael C Gonzalez et al 2007]** Rafael C Gonzalez and Richard E Woods. 2007. Image processing. Digital image processing.
- **[J. McAuley, R. Pandey & J. Leskovec, 15]** Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring Networks of Substitutable and Complementary Products. In KDD.

Reference

- **[J. McAuley & J. Leskovc, 13]** Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In WWW.
- **[A. Mislove, M. Marcon et al, 07]** Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2007. Measurement and Analysis of Online Social Networks. In SIGCOMM.
- **[S. Fakraei et al. 15]** Shobeir Fakhraei, James Foulds, Madhusudana Shashanka, and Lise Getoor. 2015. Collective Spammer Detection in Evolving Multi-Relational Social Networks. In KDD'15.

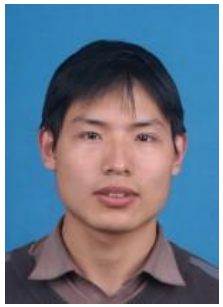
中国科学院计算技术研究所

中国科学院 CHINESE ACADEMY OF SCIENCES

Carnegie Mellon SCHOOL OF COMPUTER SCIENCE








Thank you !

Source codes and datasets used in the paper are available at
https://github.com/wenchieh/eaglemine