Culprits and Islands

Jilles Vreeken

4 July 2014 (TADA)

UNIVERSITÄT DES SAARLANDES
CLUSTER OF EXCELLENCE
max planck institut informatik

Service Announcement #1

Introduction

- Is DM science?
- DM in action

Tensors

- Introduction to tensors
- Tensors in DM
- Special topics in tensors

Information Theory

- MDL + patterns
- Entropy + correlation
- MaxEnt + iterative DM

Mixed Grill

- Influence Propagation
- Redescription Mining
- ***<special request>***

Service Announcement #1

<special request>?

Let us know (asap, mail) what topic **you** would like to see discussed

- MaxEnt + iterative DM
- ***<special request>***

Who are the Culprits?

B. Aditya Prakash VT
Jilles Vreeken
Christos Faloutsos Carnegie Mellon University

4 July 2014 (TADA)

UNIVERSITÄT DES SAARLANDES
M2CI CLUSTER OF EXCELLENCE
mpi max planck institut informatik

First question of the day

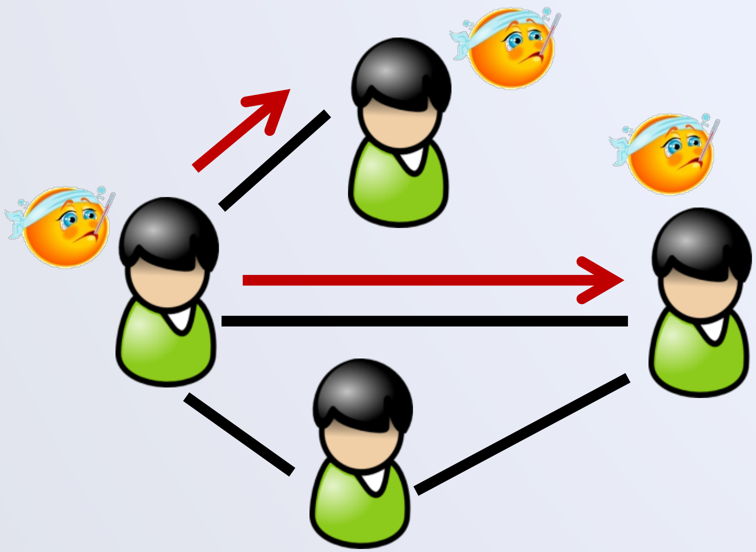
How can we find the
number *and* location of starting
points for epidemics in graphs?

– or –

Who are the culprits?

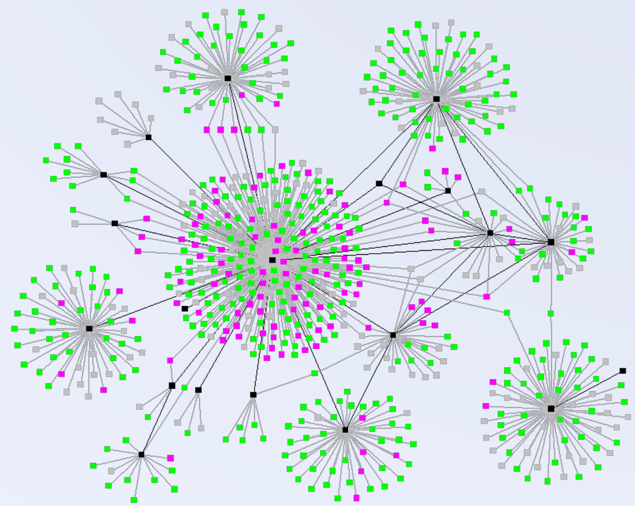
Virus Propagation

Susceptible-Infected (SI) Model



Diseases over contact networks

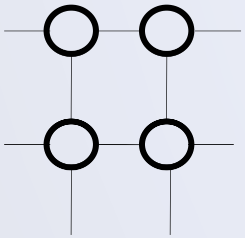
[AJPH 2007]

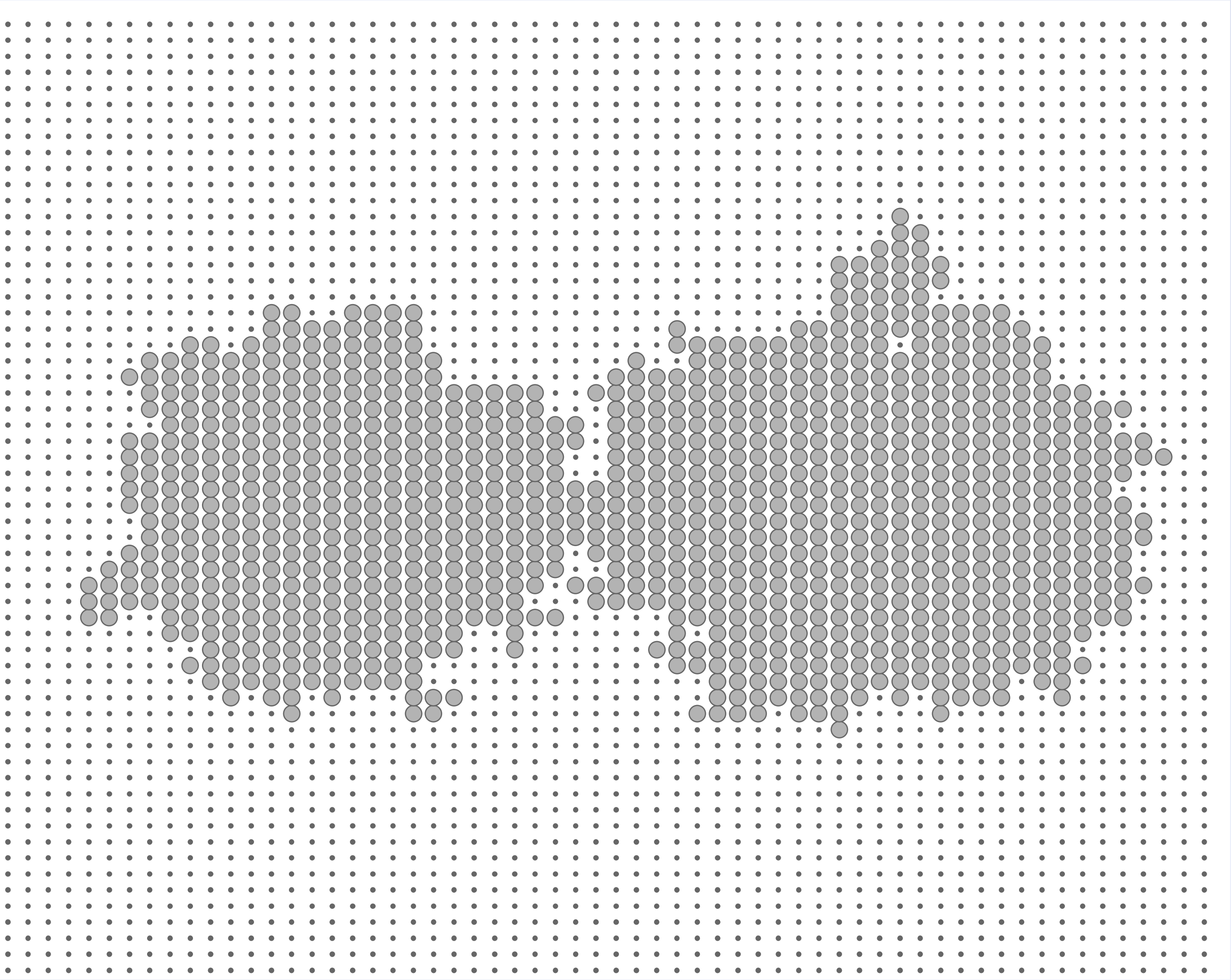


CDC data:
Visualization of the first 35 tuberculosis (TB) patients and their 1039 contacts

Culprits: Problem definition

2d grid

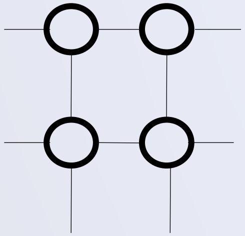


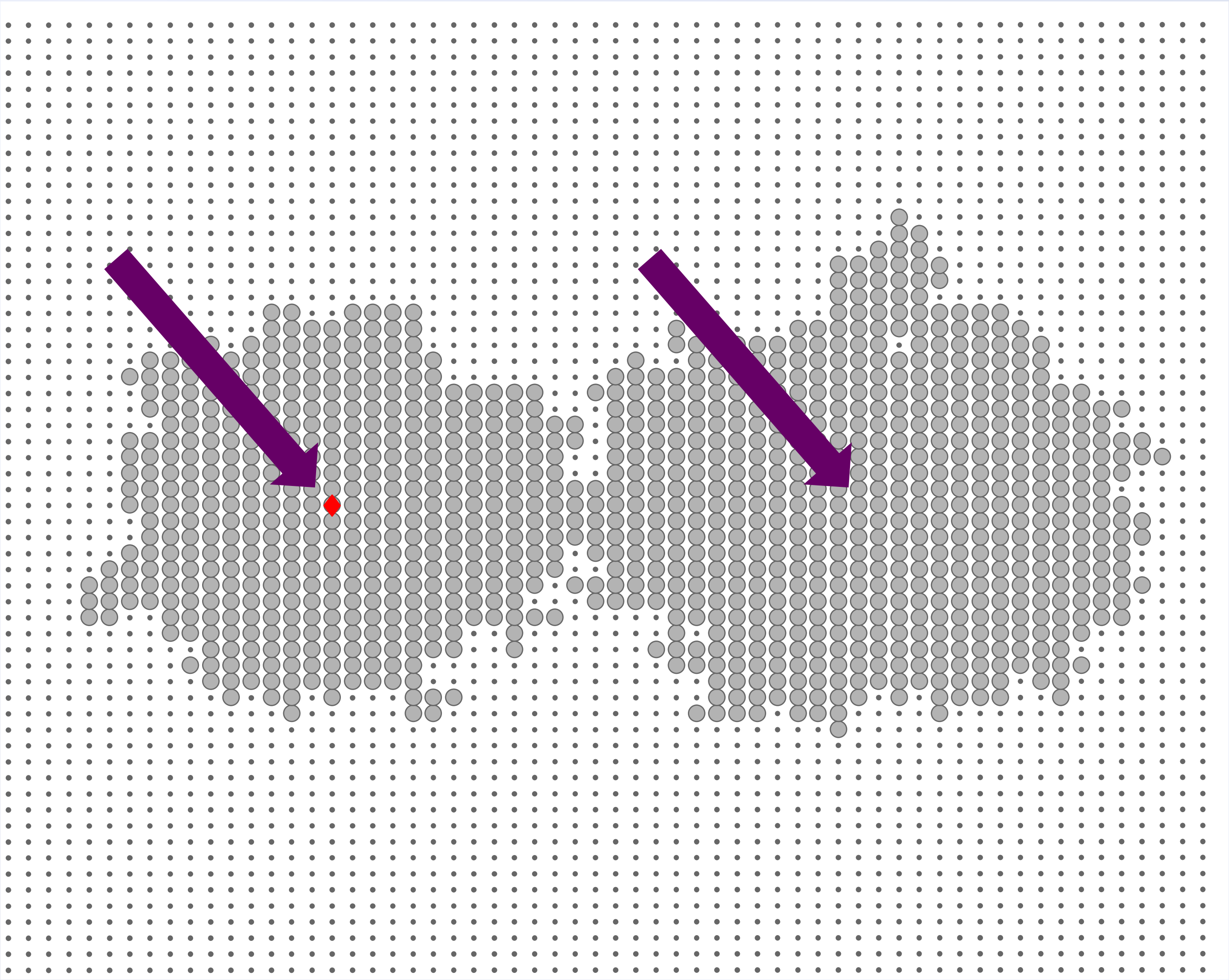


Question:
Who started it?

Culprits: Problem definition

2d grid

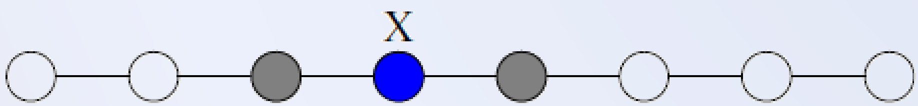




Question:
Who started it?

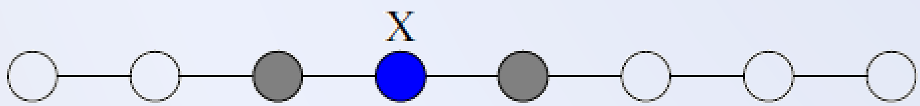
Prior work: [Lappas et al. 2010, Shah et al. 2011]

Culprits: Exoneration

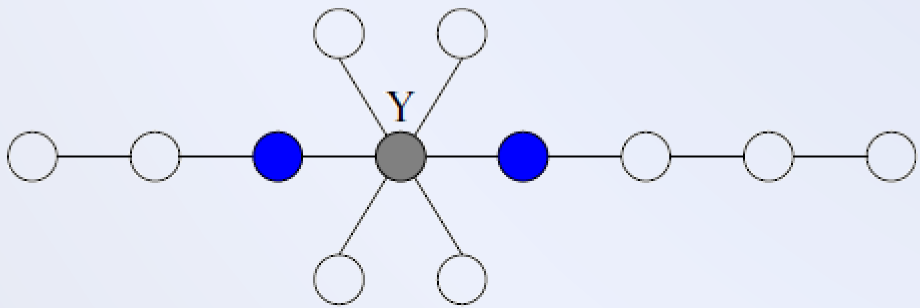


(a) A chain

Culprits: Exoneration

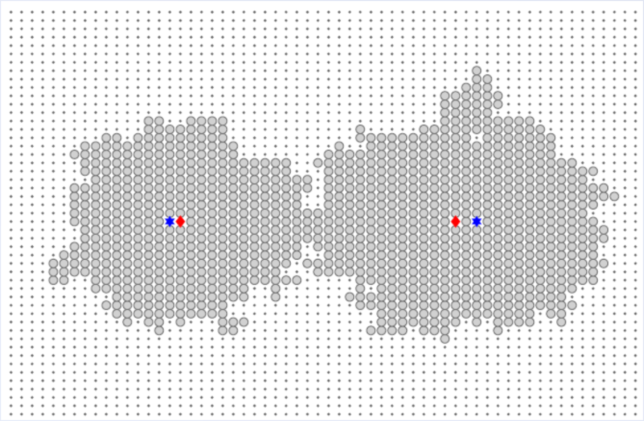


(a) A chain



(b) A chain-star

Who are the culprits



Two-step solution

1) use MDL for *number* of seeds
2) for a given number:
   exoneration =
   centrality + penalty

Running time linear!
(in edges and nodes)

$$O(k^*(E_I + E_F + V_I))$$

Modeling using MDL

Minimum Description Length principle

Induction by Compression

Related to Bayesian approaches

MDL = Model + Data

Cost of a Model:

scoring the seed-set

$$\mathcal{L}(\mathcal{S}) = \mathcal{L}_{\mathbb{N}}(|\mathcal{S}|) + \log \binom{N}{|\mathcal{S}|}$$

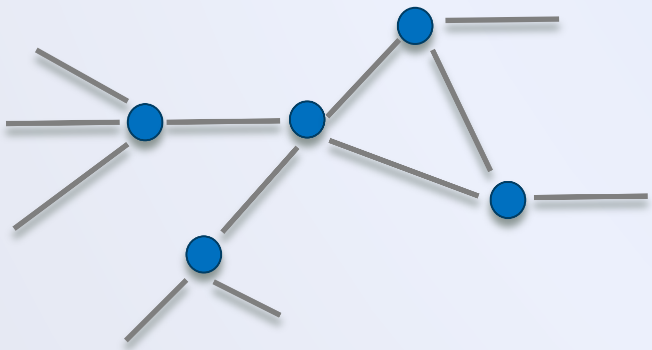
Encoding *integer* $|S|$

Number of *possible* $|S|$-sized sets

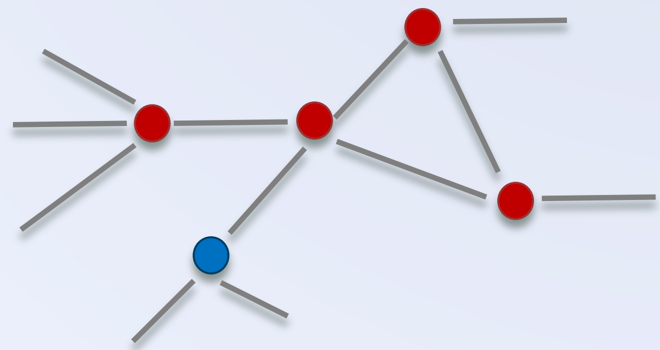
Modeling using MDL

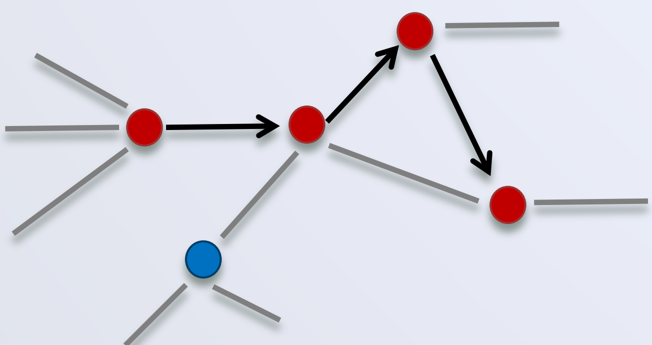
Encoding the Data: Propagation Ripples

Original Graph

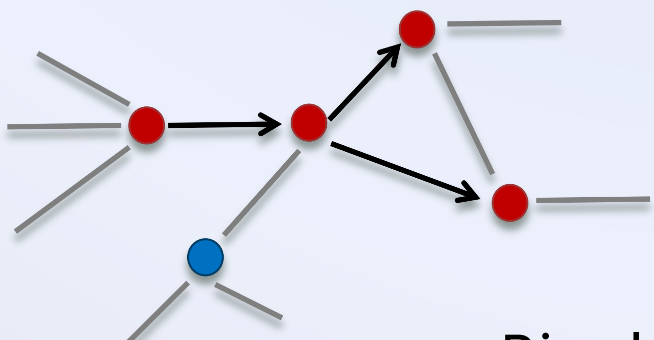


Infected Snapshot





Ripple *R1*



Ripple *R2*

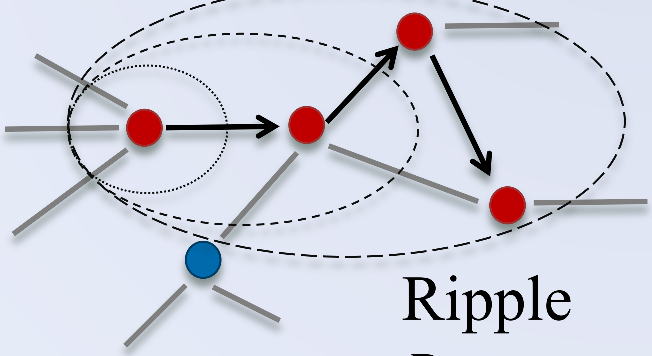
Modeling using MDL

Ripple cost

$$\mathcal{L}(R \mid \mathcal{S}) = \mathcal{L}_{\mathbb{N}}(T) + \sum_{t}^{T} \mathcal{L}(\mathcal{F}^{t})$$

How long is the ripple

How the 'frontier' advances



Total MDL cost

$$\mathcal{L}(G_I, \mathcal{S}, R) = \mathcal{L}(\mathcal{S}) + \mathcal{L}(R \mid \mathcal{S})$$

How to optimize the score?

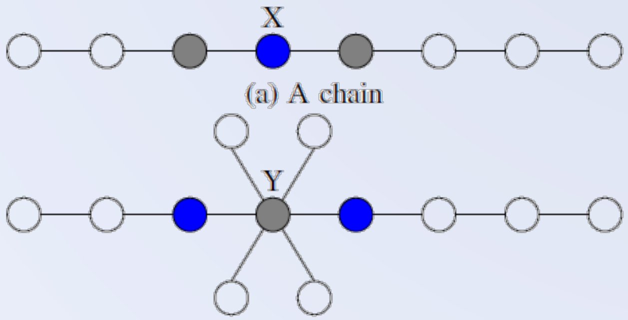
Two-step process

- Given *k* quickly identify high-quality set *S*
- Given set *S*, optimize the ripple *R*

Optimizing the score

High-quality *k*-seed-set

- exoneration



(a) A chain
(b) A chain-star

Best single seed:

- *smallest* eigenvector of Laplacian *sub-matrix*
- analyze a ***Constrained*** SI epidemic

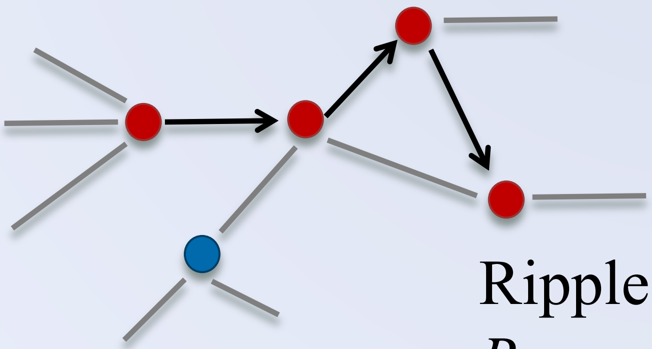
Exonerate neighbors

Repeat

Optimizing the score

Optimizing R

- Get the MLE ripple!



Finally use MDL score to tell us the best set

NETSLEUTH: Linear running time in nodes and edges

$$O(k^*(E_I + E_F + V_I))$$

Experiments

Evaluation functions:

- MDL based

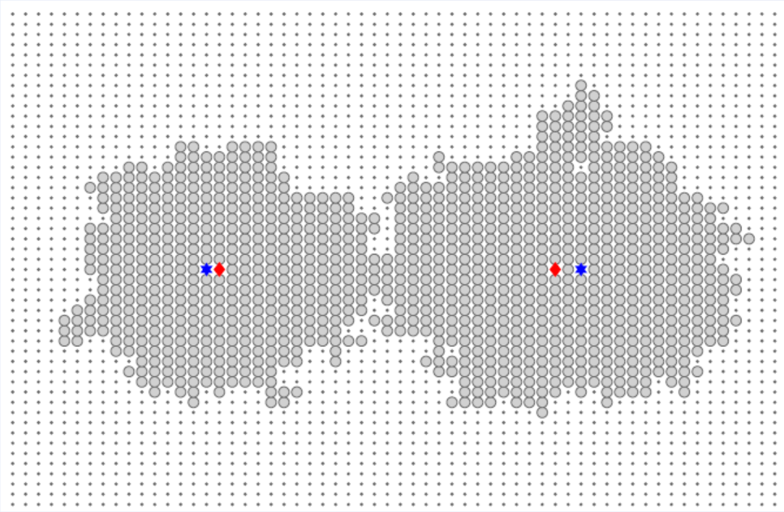
$$Q_{\text{MDL}} = \frac{\mathcal{L}(G_I, \mathcal{S}, R)}{\mathcal{L}(G_I, \mathcal{S}^*, R^*)}$$

- Overlap based

$$Q_{\text{JD}} = \frac{\mathbb{E}[JD_{\mathcal{S}}(\mathcal{V}_I)]}{\mathbb{E}[JD_{\mathcal{S}^*}(\mathcal{V}_I)]}$$

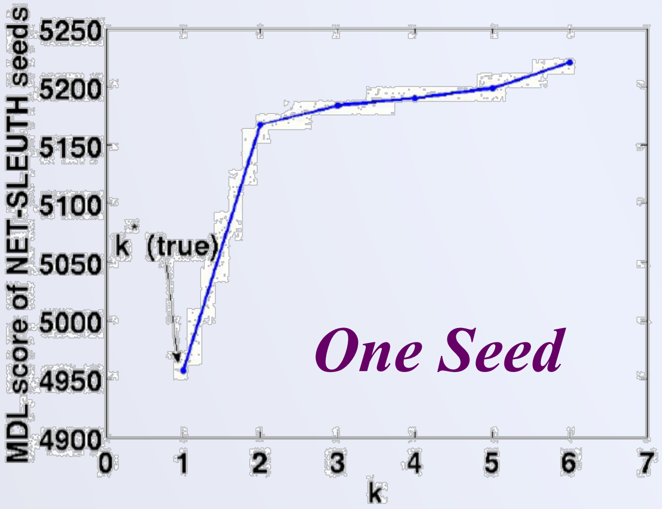
(*JD* = Jaccard distance)

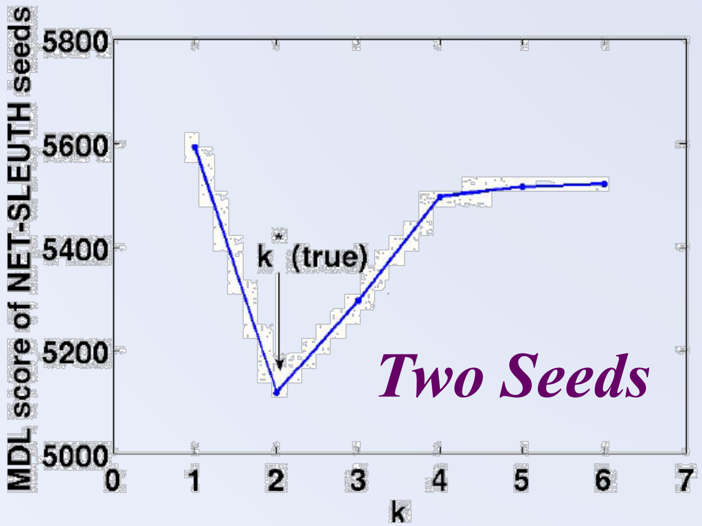
How far are they?

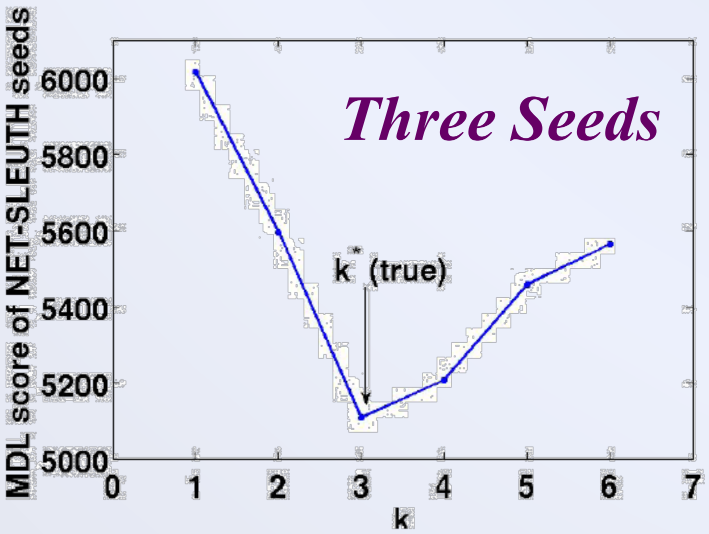


Closer to 1 the better

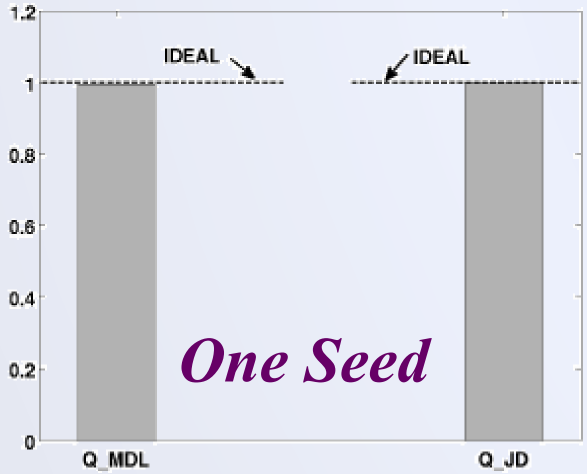
Experiments: # of Seeds

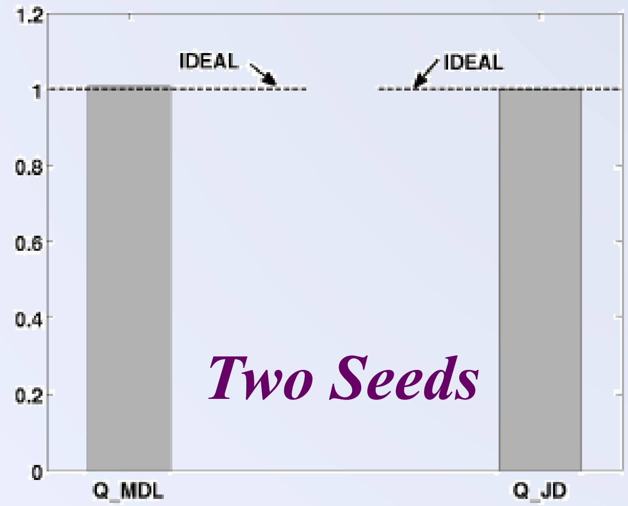






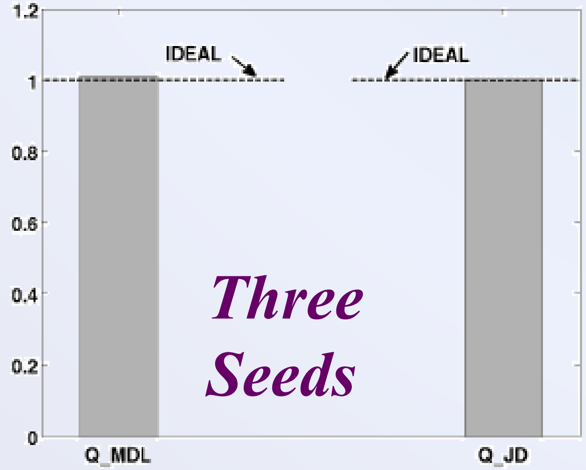
Experiments: Quality (MDL and JD)





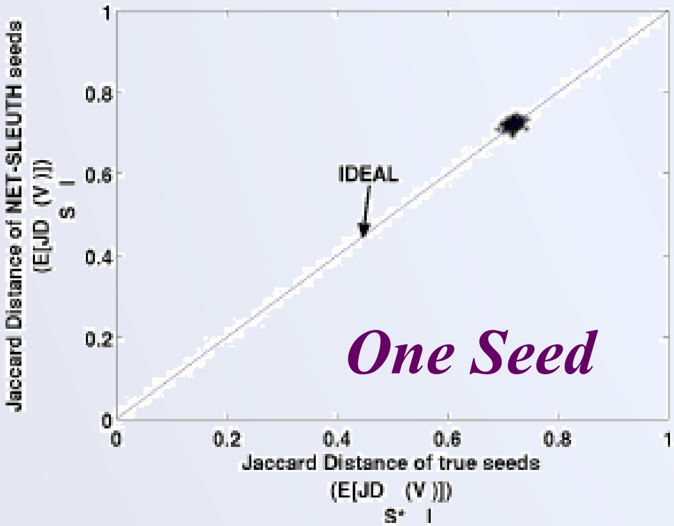
$$Q_{\text{MDL}} = \frac{\mathcal{L}(G_I, \mathcal{S}, R)}{\mathcal{L}(G_I, \mathcal{S}^*, R^*)}$$

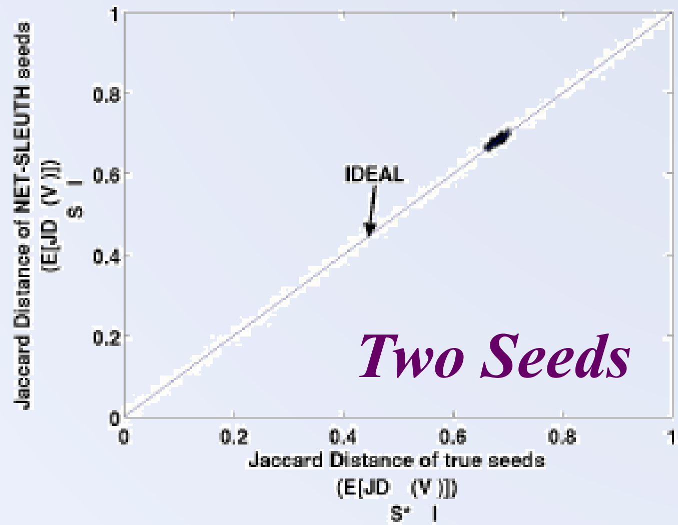
$$Q_{\text{JD}} = \frac{\mathbb{E}[JD_{\mathcal{S}}(\mathcal{V}_I)]}{\mathbb{E}[JD_{\mathcal{S}^*}(\mathcal{V}_I)]}$$

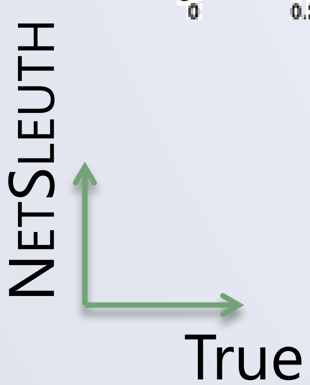


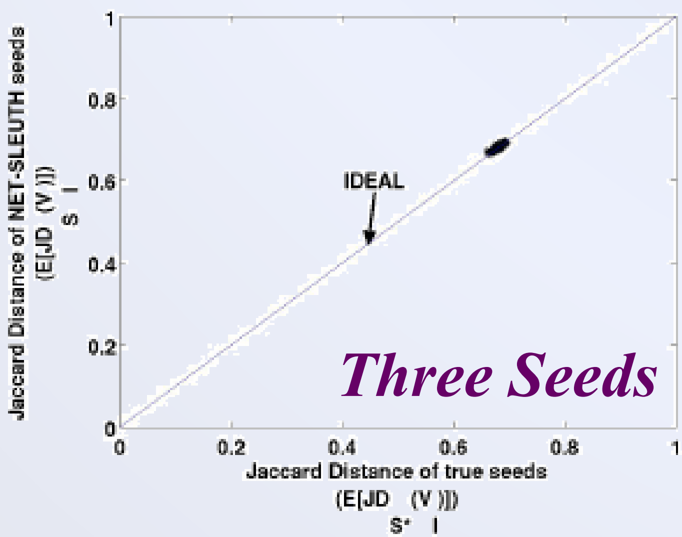
Ideal = 1

Experiments: Quality (Jaccard Scores)



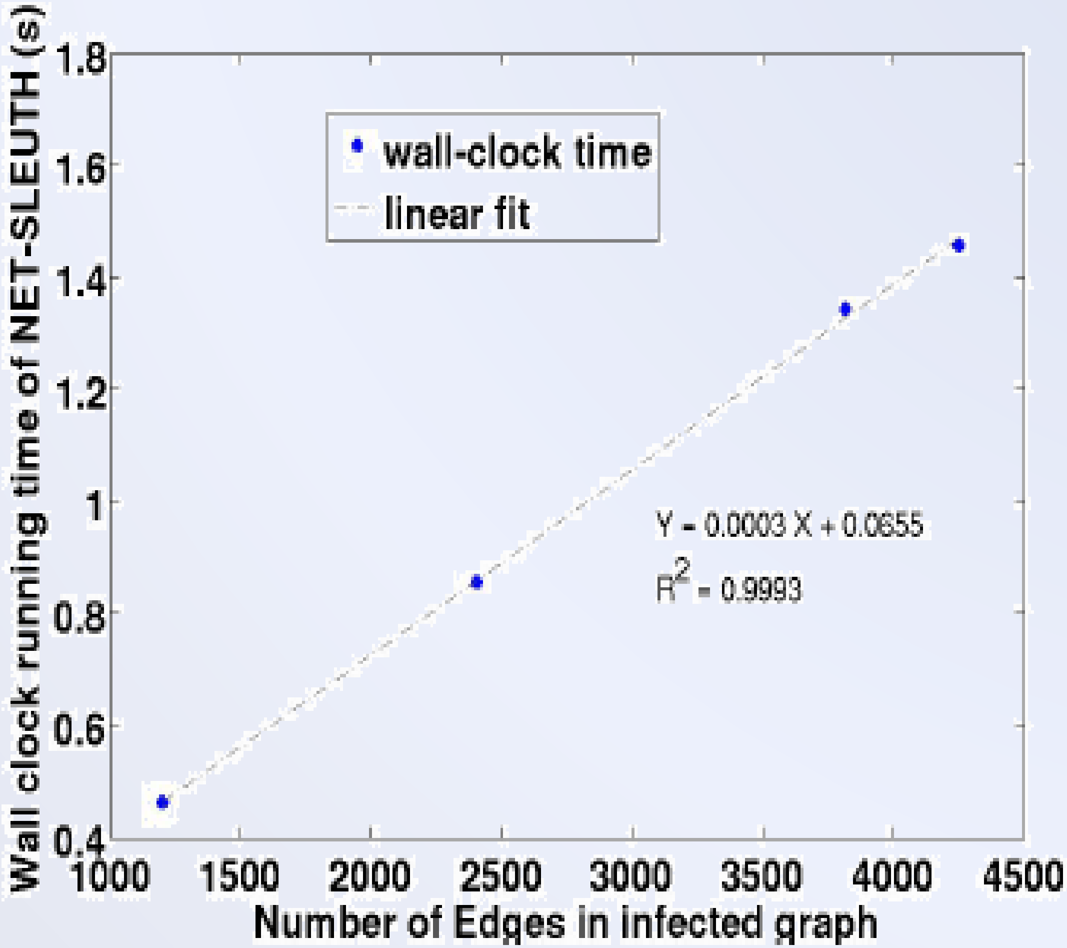






Closer to diagonal, the better

Experiments: Scalability



Conclusion

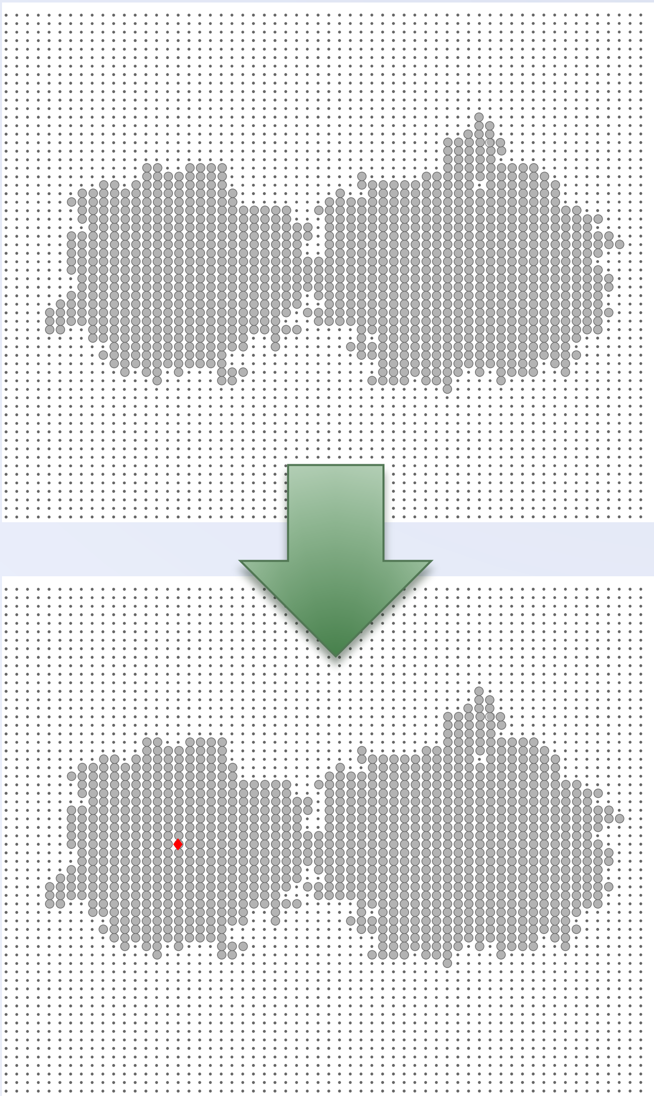
Given: Graph and Infections

Find: Best 'Culprits'

Two-step solution

- use **MDL** for *number* of seeds
- for a given number:

  exoneration = centrality + penalty



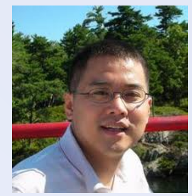
- **NetSleuth**:
  - Linear running time in nodes and edges

$$O(k^*(E_I + E_F + V_I))$$

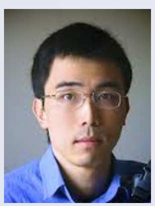
Connection Pathways

Leman Akoglu

Jilles Vreeken



Hanghang Tong



Polo Chau

Nikolaj Tatti

Christos Faloutsos

(Akoglu et al. SDM'13)

Question at hand

How can we use a graph
to **explain** a few **selected nodes**?

Given a ‘list’ of authors…

What can we say?

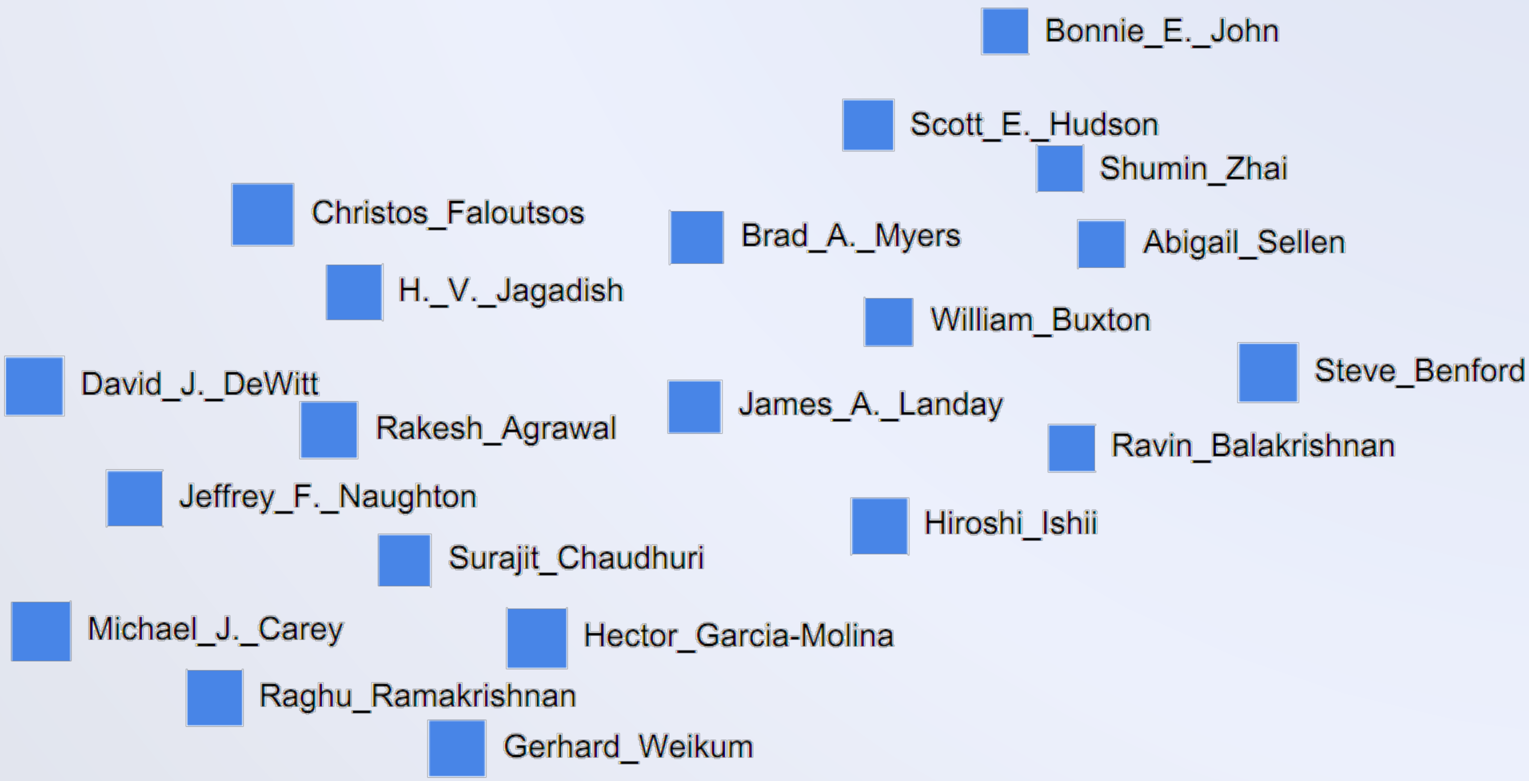
- let’s use relational information

- Christos Faloutsos
- H. V. Jagadish
- David J. DeWitt
- Bonnie E. John
- Hector Garcia Molina
- James A. Landay
- Brad A. Myers
- Jeffrey F. Naughton
- Hiroshi Ishii
- Gerhard Weikum
- William Buxton
- Raghu Ramakrishnan
- Michael J. Carey
- Rakesh Agrawal
- Surajit Chaudhuri
- Scott E. Hudson
- Shumin Zhai
- Abigail Sellen
- Steve Benford
- Ravin Balakrishnan

Given a ‘list’ of authors…

What can we say?

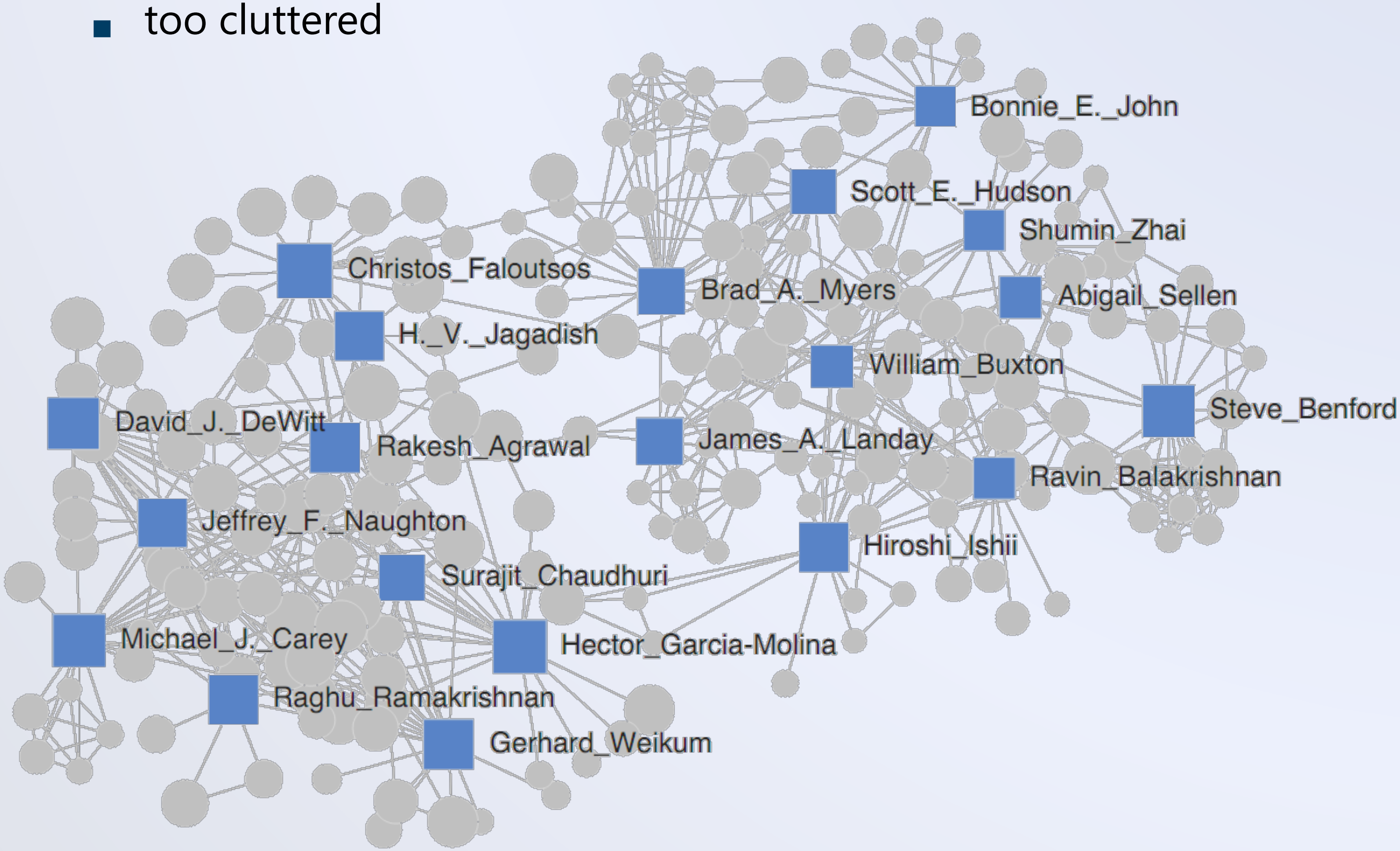
- let’s use relational information



Using the co-authorship graph…

Any structure?

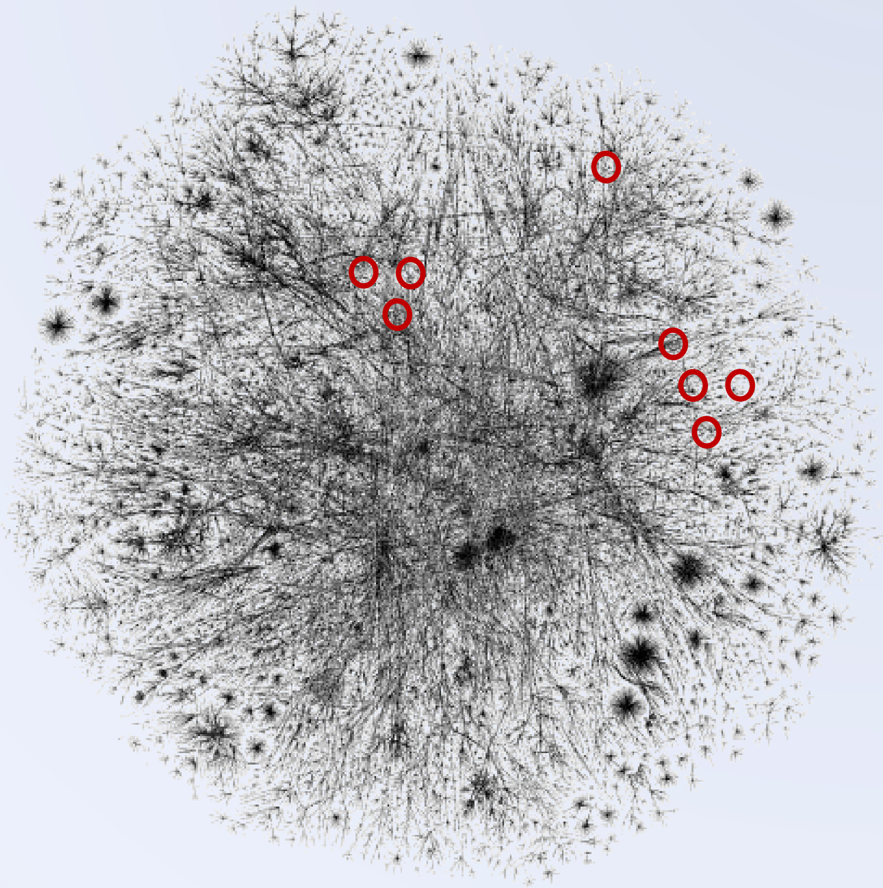
- too cluttered



The Problem

Given

- a **large graph G**
- a **handful of nodes S**
  marked by an external process



What can we say about **S**?

- are they **close by**?
- are they **segregated**?
- do they form **groups**?

Can we connect them?

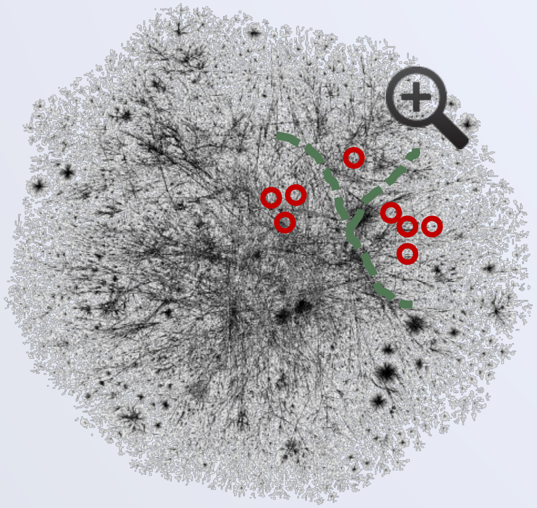
- with **simple** paths?
- maybe using a few **connectors**?

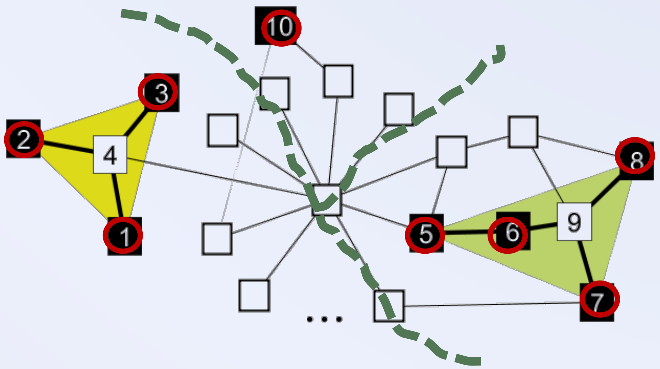
Our approach

Use the network structure to explain S

Partition S into **groups** of nodes, such that

- "simple" **paths** in G connect the nodes in each **group**,
- nodes in different groups are "not easily reachable"



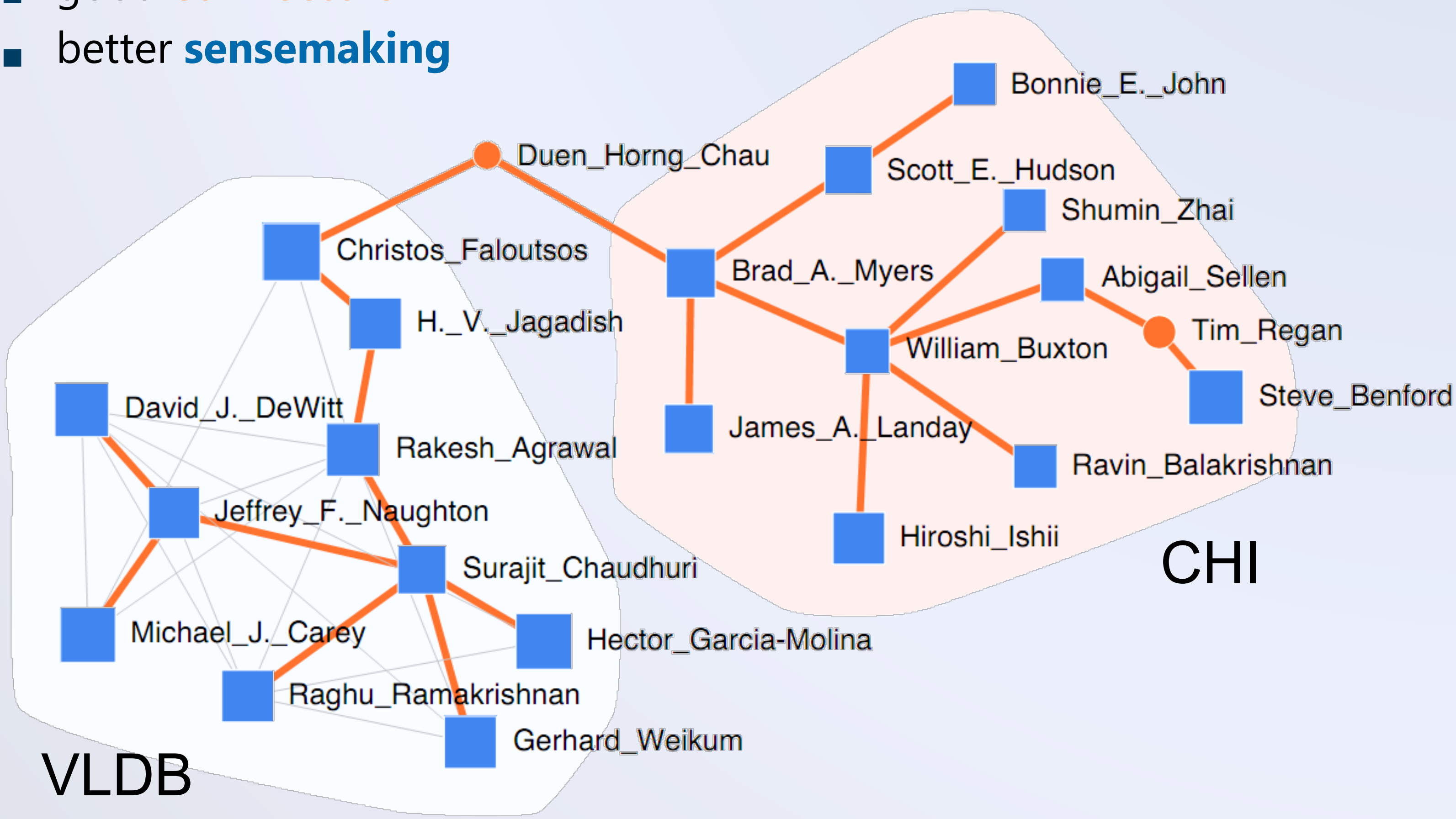


Use MDL to decide '**simple**' and '**best**' partitioning

Example

Simple connection pathways

- good **connectors**
- better **sensemaking**

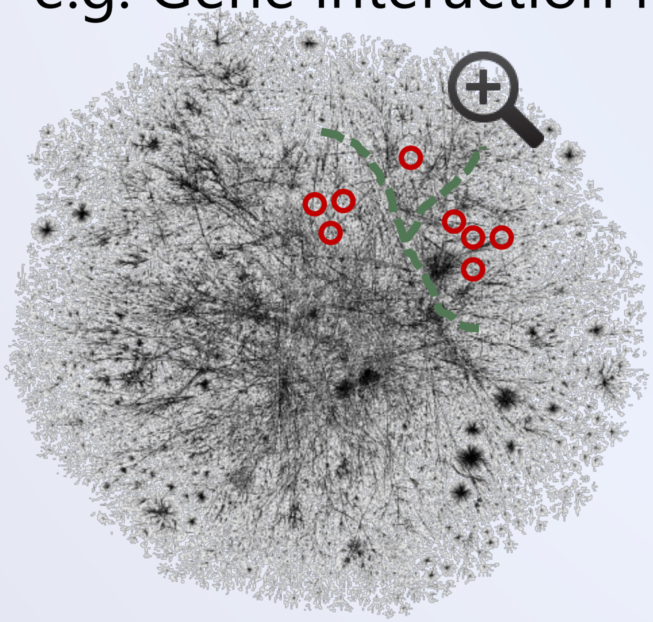


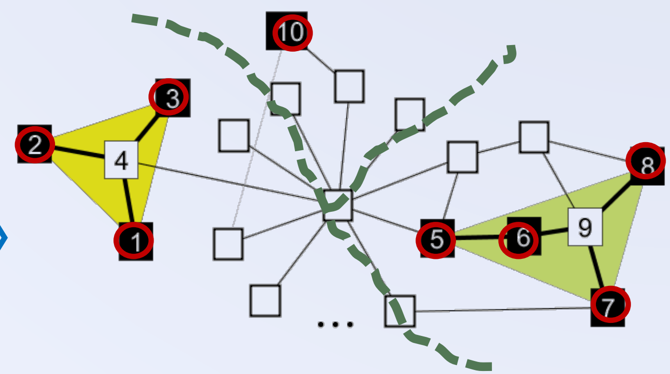
Applications

1. Graph anomaly description/summarization

e.g. Gene interaction network

Top-k anomalies





✓ Summarize top-k node anomalies by groups

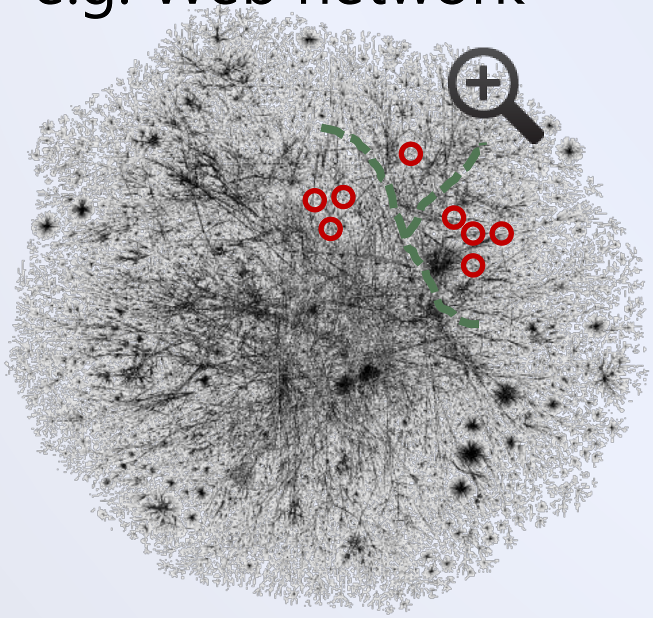
✓ Find connections/connectors among groups

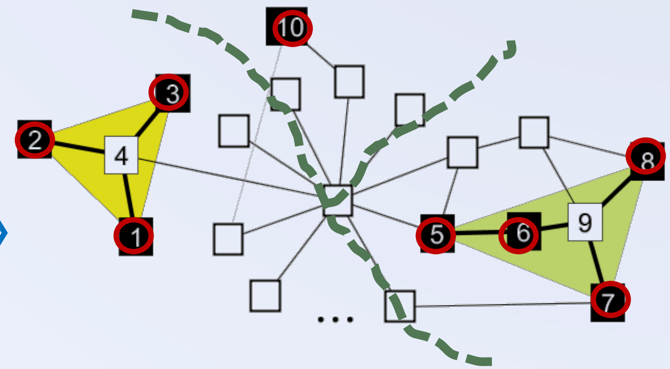
Applications

2. Query summarization

e.g. Web network

Top-ranked pages





✓ Summarize top-k query pages by groups

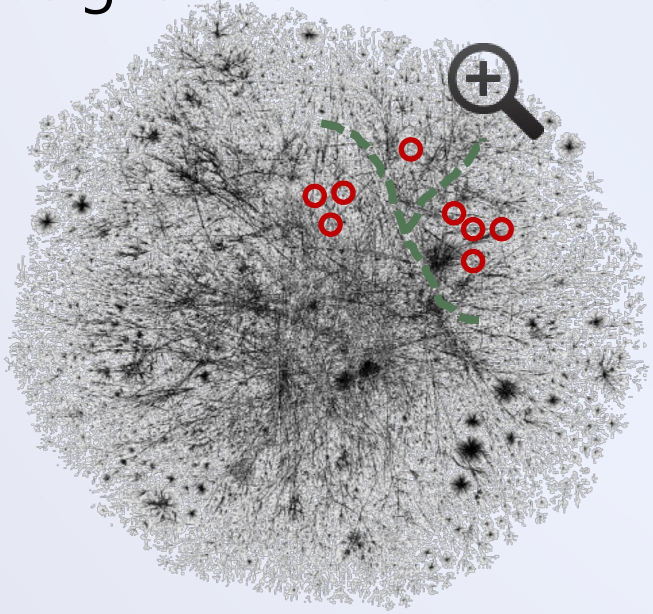
✓ Find connections/connectors among groups

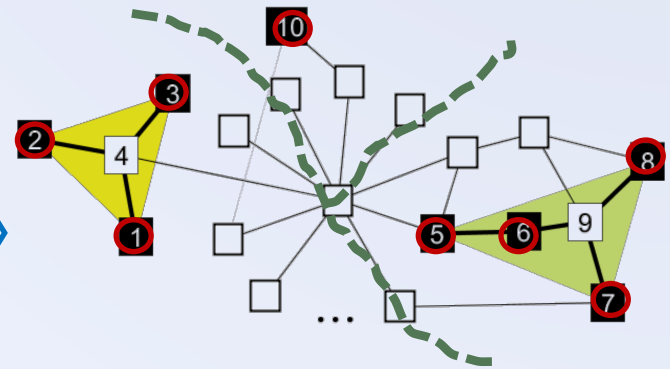
Applications

3. Understanding dynamic events in graphs

e.g. Social network

Affected people





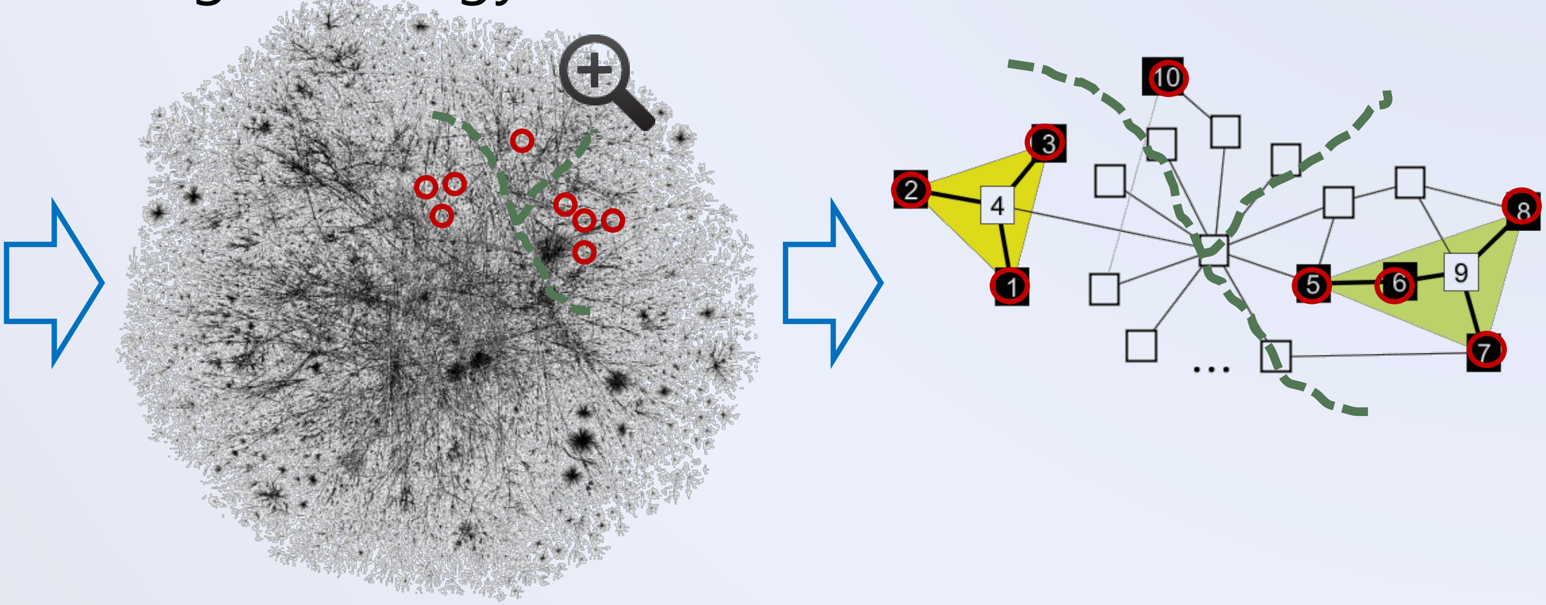
- ✓ Event spread within groups explained by the network
- ✓ Event spread between groups due to external influence

Applications

4. Understanding semantic coherence

e.g. Ontology network

Set of words



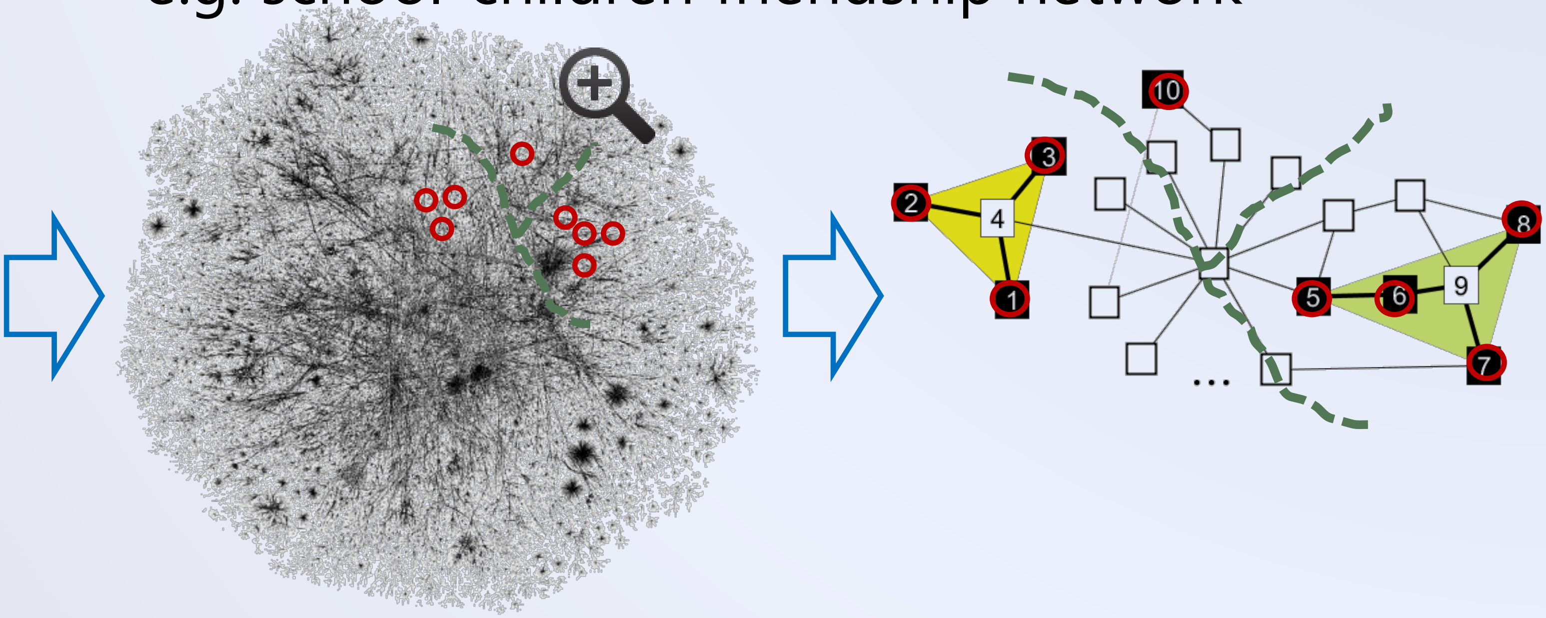
- ✓ Summarize words by semantically coherent groups
- ✓ Find connectors (other relevant words) per group

Applications

5. Understanding segregation (social science)

e.g. school-children friendship network

Students with attributes of interest



✓ Summarize students by their social "circles"
✓ Study groups (and groups within groups)

Problem: Formally

Problem Definition

Given a graph G=(V,E) and a set of marked nodes M subseteq V

Problem 1. Optimal partitioning

Find a **coherent** partitioning P of M.
Find the optimal **number of partitions** $|P|$.

Problem 2. Optimal connection subgraphs

Efficiently find the minimum **cost** set of subgraphs connecting the nodes in each part $p_i \in P$

Objective: Informally

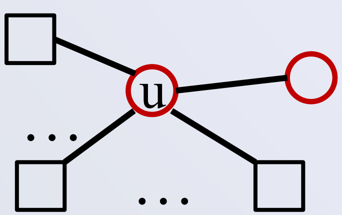
Our **key idea** is to use information theory

Imagine a sender and a receiver.

- both sender and receiver know graph structure G,
- only the sender knows the set of marked nodes M
- goal: transmit M **using as few bits as possible**.

Why would this work?

- **naïve**: encode ID of each marked node with $\log |V|$ bits
- **better**: **exploit "close-by" nodes, restart for farther nodes**



$\log |V| + \log d(u)$ vs. $2 \log |V|$

Objective: Intuition

We think of encoding as

- **hopping** from node to node to encode **close-by** nodes
- and **flying** to a new node to encode **farther** nodes
- until all marked nodes are identified

Simplicity of connection tree T is determined by:

- the **amount of flights** we make across the graph;
- ease of **identifying the edges to follow** next;
- ease of **identifying the marked nodes** in our tour;

Objective: Formally

minimize P, T_i $$L(P, M \mid G) = L(|P|) + \sum_i L(p_i)$$

- encode #partitions $L(|P|) = \log |V|$
- encode each part

$$L(p_i) = \log |V| + L(t) + \log |T| + \log \binom{|T|}{||T||}$$

root node

spanning tree t of p_i

number of marked nodes in p_i

identities of marked nodes

- encoding of tree per part

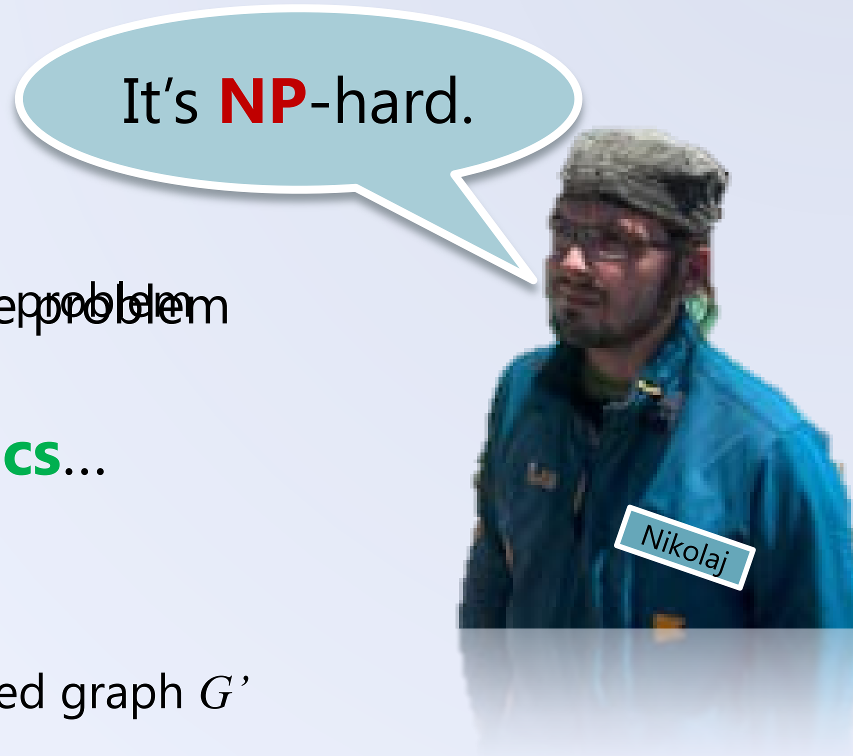
$$L(t) = L_{\mathbb{N}}(|t| + 1) + \log \binom{d(v_t)}{|t|} + \sum_j^{|t|} L(b(t, j))$$

#branches of node t

identities of branch nodes

recursively encode all tree nodes

Solution: Intuition



The problem is hard

The problem is NP-hard

- Related to the directed Steiner tree problem
- Reduces to directed Steiner tree problem

Hence, we resort to **heuristics**…

The general idea:

- transform G into a directed weighted graph G'
- chop G' into sub-graphs
- find low-cost minimal spanning trees per sub-graph
  (we give 4 efficient algorithms)

Solution: Preliminaries

Graph transformation

- given undirected unweighted $G(V, E)$
- we transform it into directed weighted $G'(V, E, W)$
  where $w(u, v) = \log d(u)$ and $w(v, u) = \log d(v)$

Given G', the problem becomes: find *the set of trees* with minimum total cost on the marked nodes.

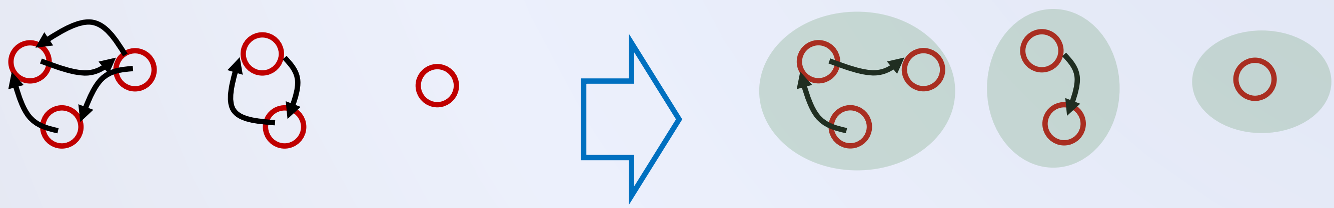
Finding bounded-length paths

- (multiple) short paths of length up to $\log |V|$ between marked nodes in G'
- employ BFS-like expansion

Algorithms

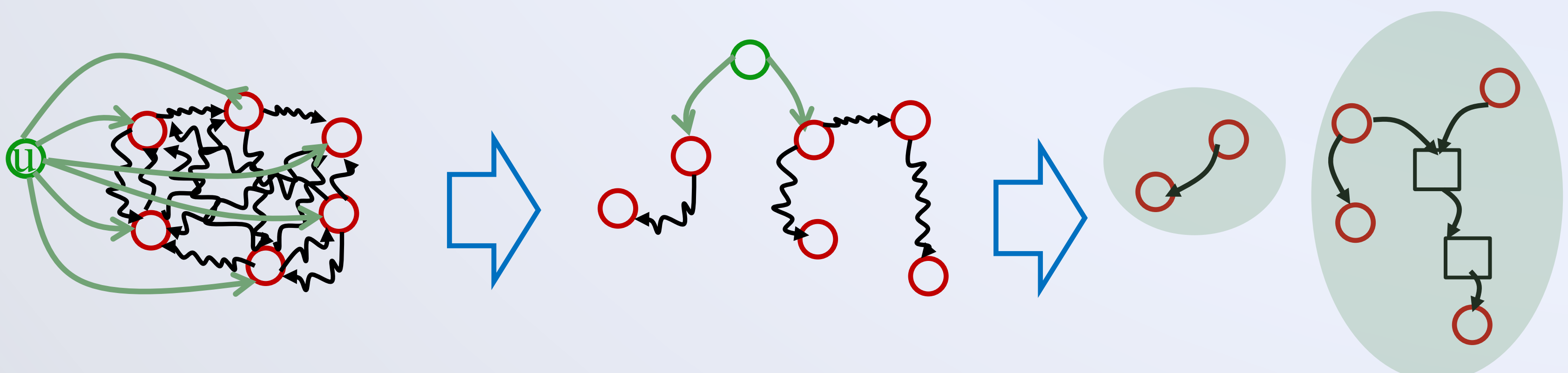
1) Connected components (CC)

- find induced subgraph(s) on marked nodes in *G'*
- find minimum cost directed tree(s)



2) Minimum arborescence (ARB)

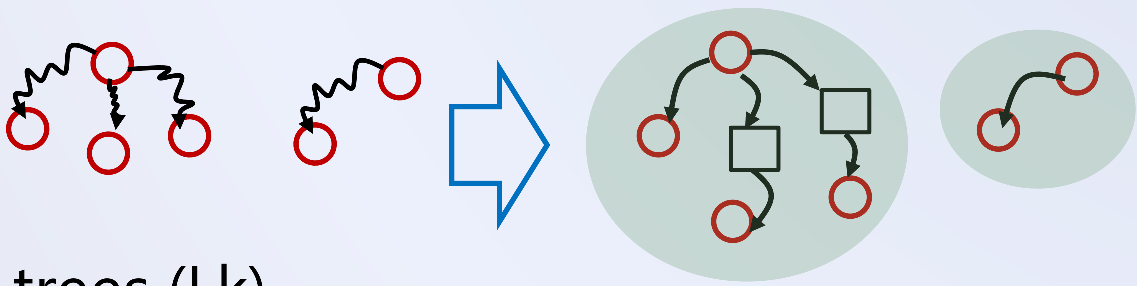
- construct transitive closure graph *CG* (with bounded paths)
- add universal node u with out-edges $w(u,m) = \log|V|$
- find minimum cost directed tree(s), remove u, re-expand paths



Algorithms

3) Level-1 trees (L1)

- find minimum cost depth-1 trees in CG
- expand paths

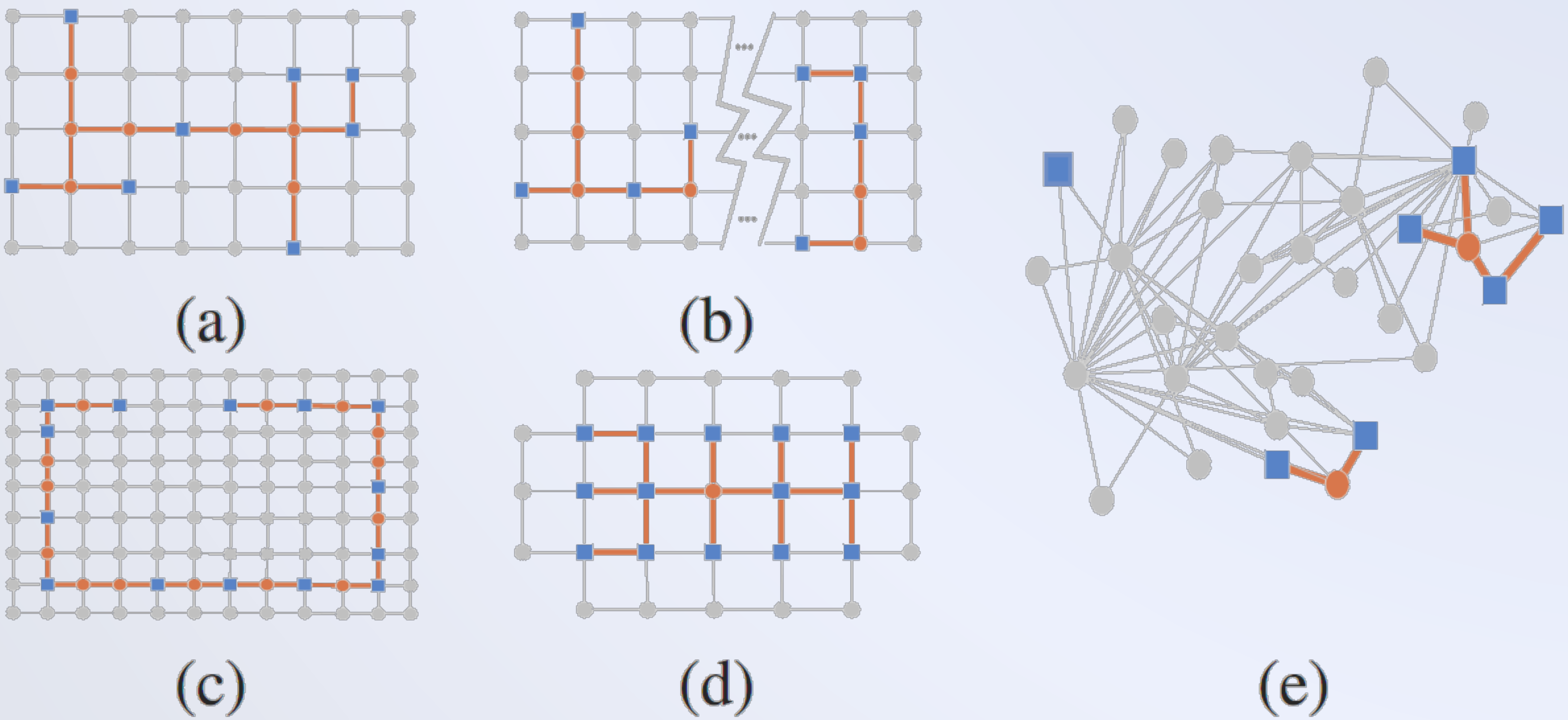


4) Level-k trees (Lk)

- refine level-(k-1) trees by finding intermediate node v's
- minimizing total cost, i.e. sum of cost to each v and subtrees

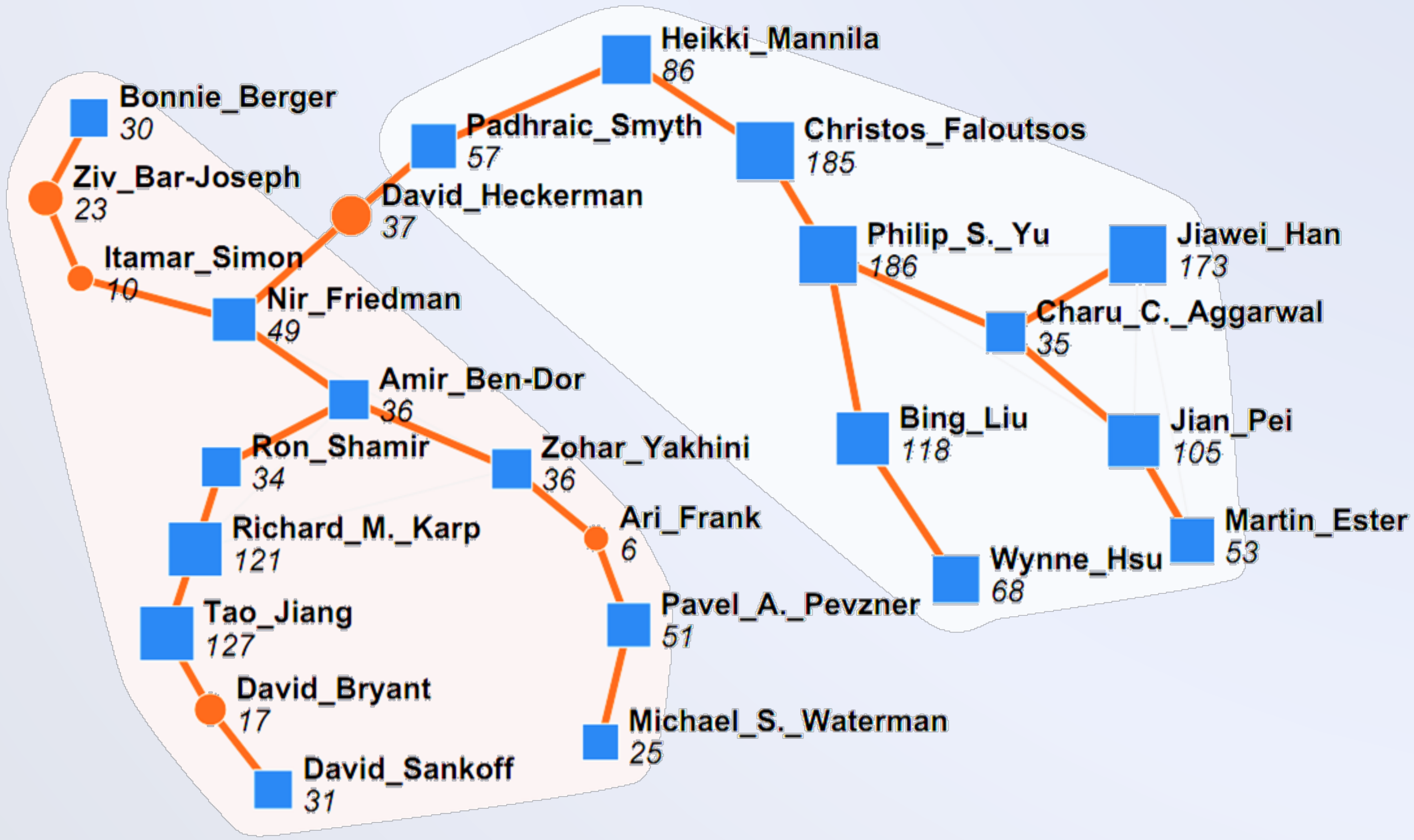
Experiments

Synthetic examples



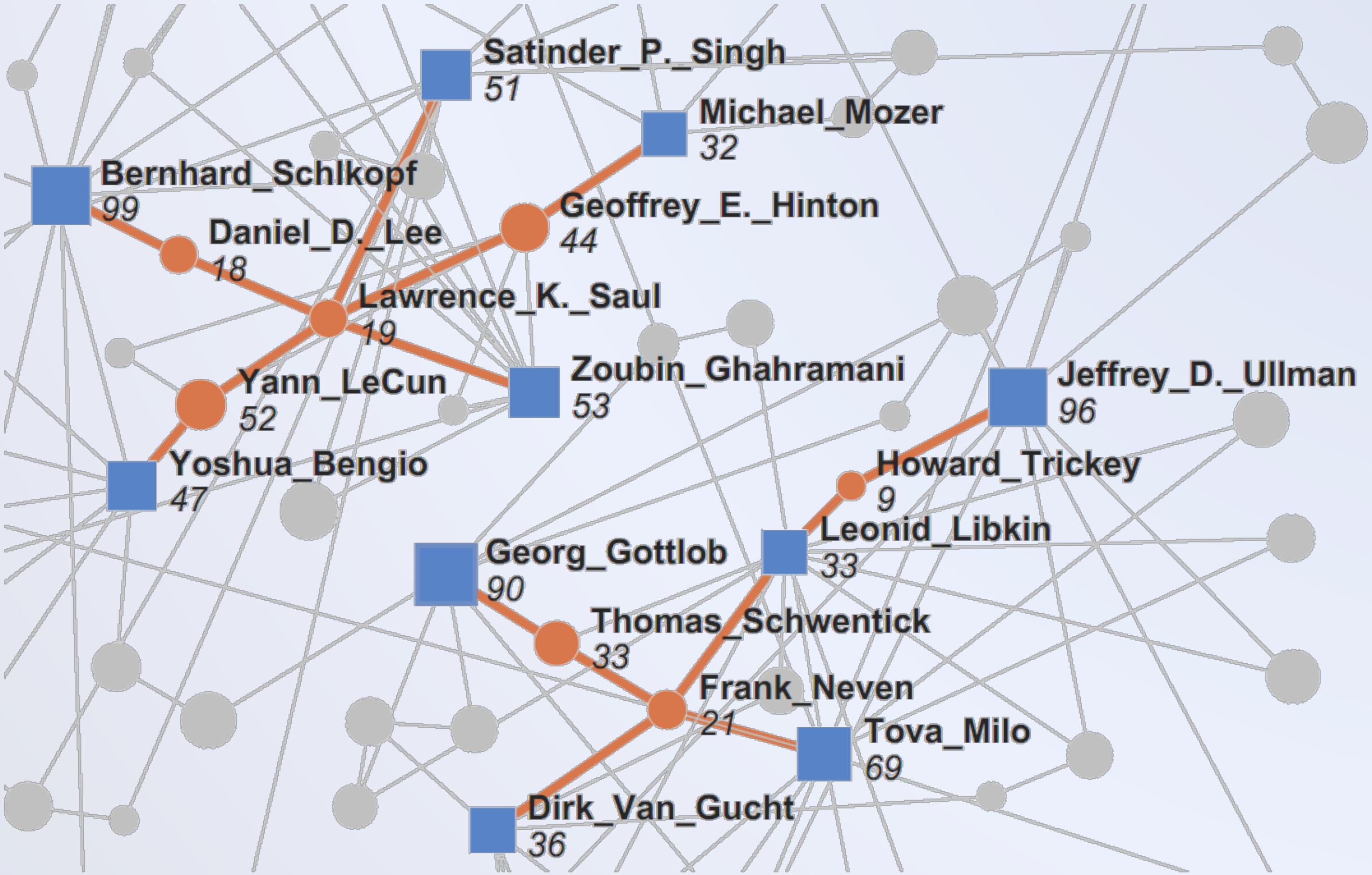
Experiments

- Case studies on DBLP



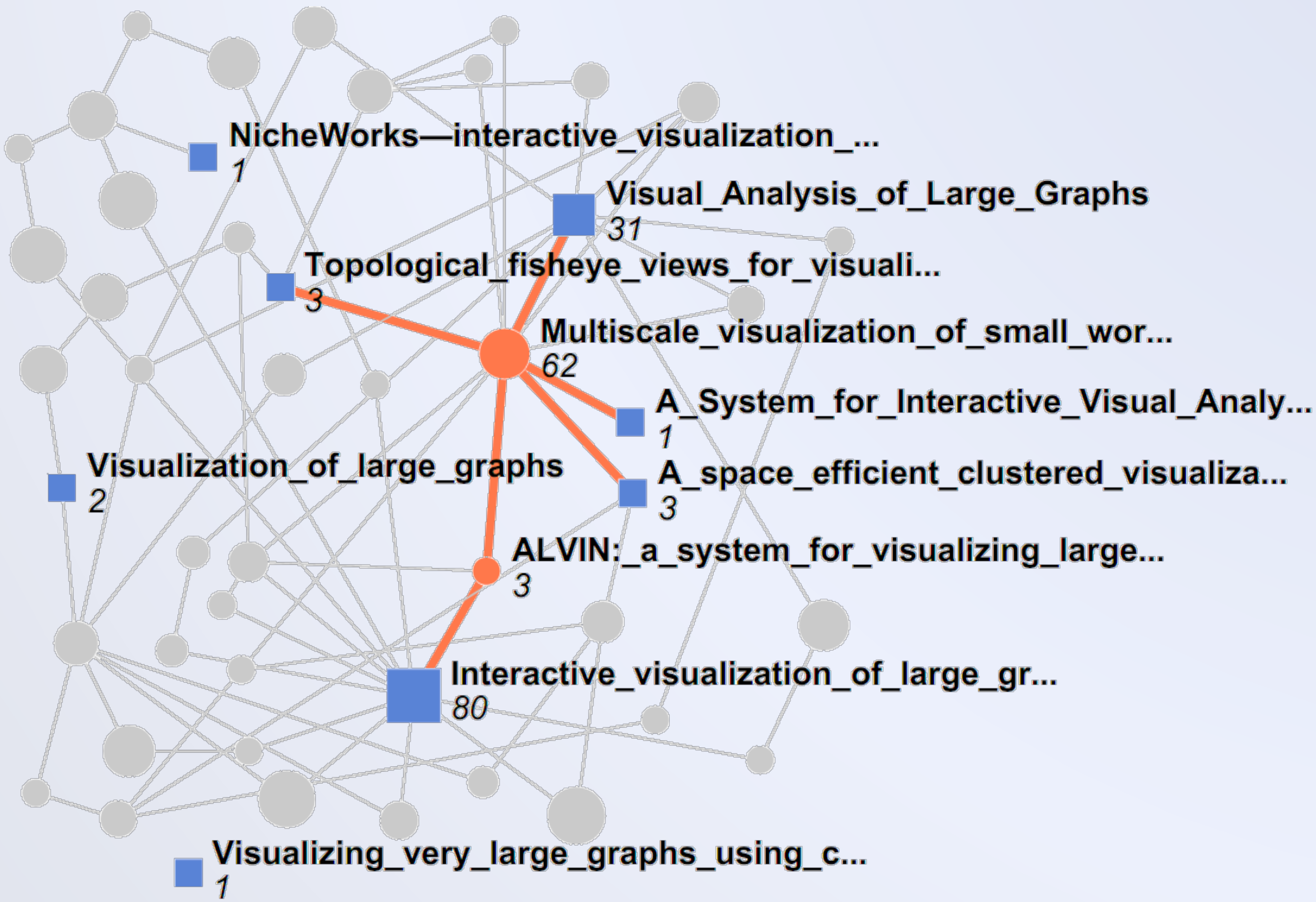
DBLP: RECOMB vs. KDD

Experiments



DBLP: NIPS vs. PODS

Experiments



GScholar: ‘large graphs’ vs. ‘visual’

Intermediate **Conclusions**

Dot2Dot

- principled approach to describe sets of **marked nodes** using structure of the graph
- automatically finds good **connectors**
- automatically determines number of groups

New problem, but many **applications** in the wild

Conclusions

Graphs problems are often difficult

- solutions are typically very ad hoc, very heuristic

Information theory

- offers a clean and principled way to define solutions

Identifying Infection Sources

- first to identify multiple sources – extensions currently underway

Explaining Node Sets

- first to define the problem – many applications in the wild

Thank you!

Graphs problems are often difficult

- solutions are typically very ad hoc, very heuristic

Information theory

- offers a clean and principled way to define solutions

Identifying Infection Sources

- first to identify multiple sources – extensions currently underway

Explaining Node Sets

- first to define the problem – many applications in the wild