Core Decomposition of Networks: Concepts, Algorithms and Applications

Fragkiskos D. Malliaros
UC San Diego
USA

Apostolos N. Papadopoulos
Aristotle University of Thessaloniki
Greece

Michalis Vazirgiannis
École Polytechnique
France

ICDM 2016

IEEE International Conference on Data Mining

December 12-15, 2016 | Barcelona, Spain

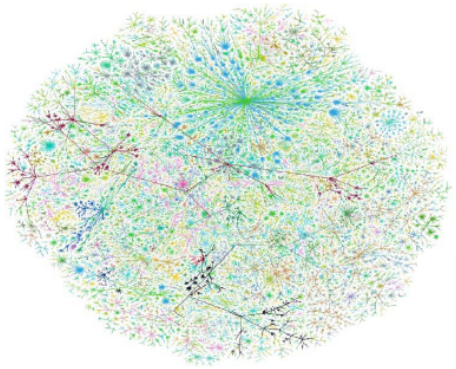
Outline

1. Introduction & Motivation
2. Fundamental Concepts and Definitions of Core Decomposition
3. Algorithms
4. Applications
5. Open Topics and Future Research Directions

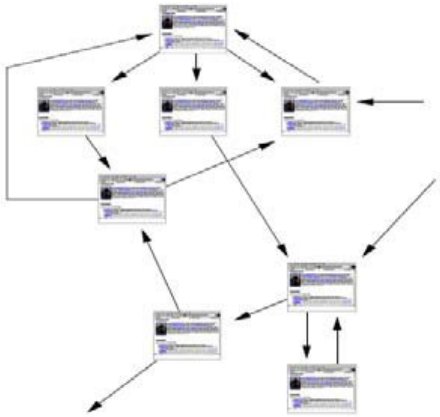
Outline

1. **Introduction & Motivation**
2. Fundamental Concepts and Definitions of Core Decomposition
3. Algorithms
4. Applications
5. Open Topics and Future Research Directions

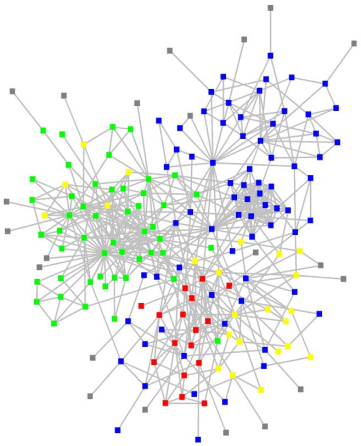
Networks are everywhere



(a) Internet

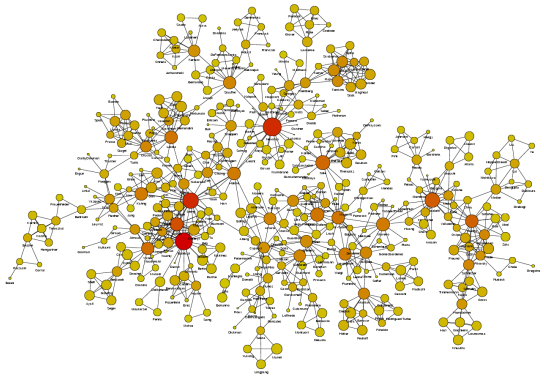


(b) World Wide Web

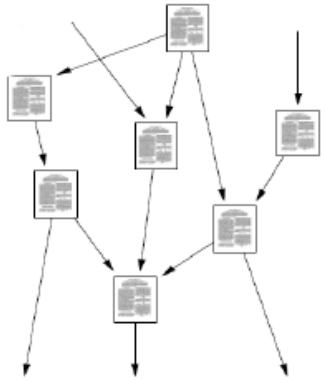


(c) Email network

(d) Social network



(e) Collaboration network



(f) Citation network

Properties of real networks

- Real networks are not **random graphs** (e.g., the Erdos-Renyi random graph model)
- Present fascinating patterns and properties:
  - The **degree distribution** is skewed, following a power-law
  - The **average distance** between the nodes of the network is short (the small-world phenomenon)
  - The edges between the nodes may not represent reciprocal relations, forming **directed networks with non-symmetric links**
  - **Edge density** is inhomogeneous - groups of nodes with high concentration of edges within them and low concentration between different groups

Community detection and clustering structure

- **Community detection** in graphs aims to identify the modules and, possibly, their hierarchical organization, by only using the information encoded in the graph topology
- First attempt dates back to 1955 by Weiss and Jacobson searching for work groups within a government agency
- Applications in various fields
  - Any context that information is encoded as a graph

Communities – application domains

- **Social communities** have been studied for a long time [Coleman, '64], [Freeman, '04], [Kottak, '04], [Moody and White, '03]
- **World Wide Web**: communities correspond to groups of pages dealing with the same or related topics [Dourisboure et al., '07], [Flake et al., '02]
- In **biology**: protein-protein interaction networks, communities are likely to group proteins having the same specific function within the cell [Chen, '06], [Rives and Galitski '03], [Spirin and Mirny, '03]
- In **metabolic networks** they may be related to functional modules such as cycles and pathways [Guimera and Amaral, '05], [Palla et al., 2005]
- In **food webs** they may identify compartments [Krause et al., '03], [Pimm, '79]

Core decomposition in complex networks

- Tools to analyze the structure of real networks
  - Quantify community and clustering structure
- Hierarchical representation of a graph into nested subgraphs of increased connectivity and coherence properties
- **Basic idea:**
  - Set a threshold on the node degree, say **k**
  - Nodes that do not satisfy the threshold are removed from the graph
- Extensions to other node properties (e.g., triangles)
- Plethora of applications
  - Dense subgraph discovery and community detection
  - Evaluation of collaboration in social networks
  - Identification of influential spreaders in social networks
  - Text analytics

Goal of the tutorial

- Introduce **core decomposition** as a tool for complex networks analysis
- Focus on the following points:

1. Fundamental concepts and definitions of core decomposition **[Part 1]**
2. Algorithms under various computation models **[Part 2]**
3. Applications of core decomposition **[Part 3]**

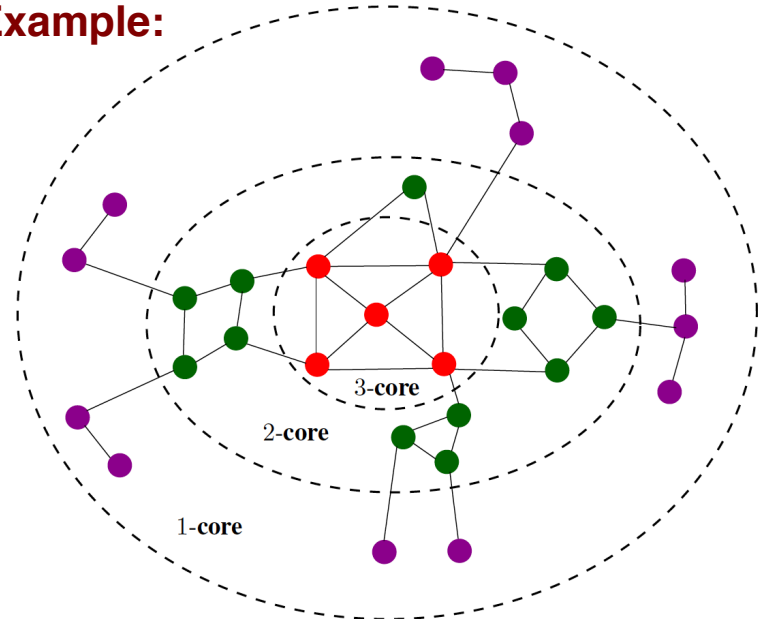
Outline

1. Introduction & Motivation
2. **Fundamental Concepts and Definitions of Core Decomposition**
3. Algorithms
4. Applications
5. Open Topics and Future Research Directions

Graph degeneracy and the k-core decomposition

- Degeneracy for an **undirected** graph G
  - Also known as the **k-core** number
  - The k-core of G is the largest subgraph in which every vertex has degree at least k within the subgraph

Example:



Core number $c_i = 1$

Core number $c_i = 2$

Core number $c_i = 3$

Graph Degeneracy $\delta^*(G) = 3$

$G_0 = G$

$G_1 = 1$-core of G

$G_2 = 2$-core of G

$G_3 = 3$-core of G

$G_0 \supseteq G_1 \supseteq G_2 \supseteq G_3$

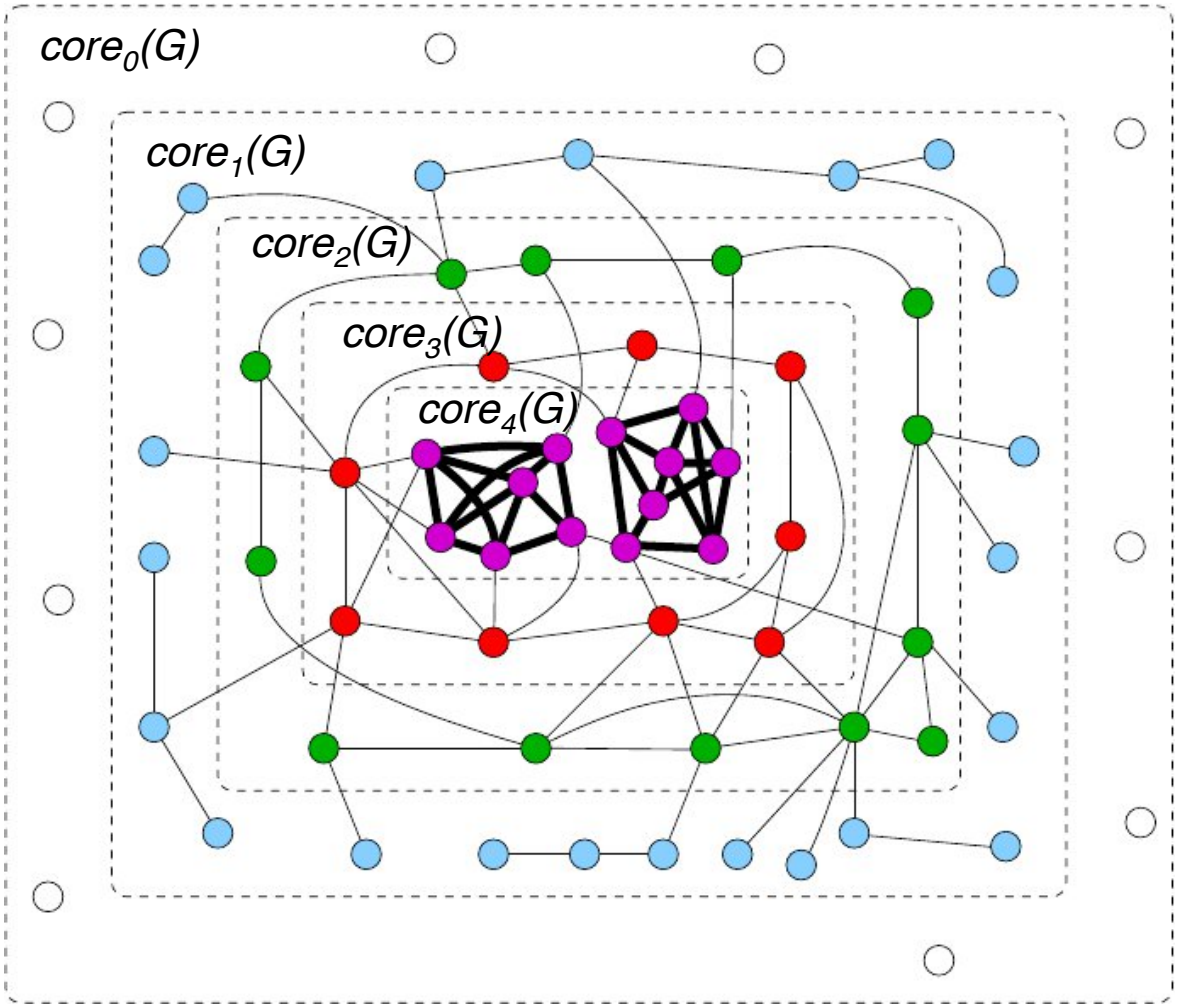
Important property:

- **Fast** and easy to compute
- Linear to the size of the graph
- Scalable to large scale graphs

Note:

The degeneracy and the size of the k-core provide a good indication of the cohesiveness of the graph

Another example



Basic algorithm for k-core decomposition

- An algorithm for computing the *k-th core of a graph:*

  Procedure *$Trim_k$(G, k)*

  Input: An undirected graph G and positive integer k

  *Output: k-**core**(G)*

  1. let F := G

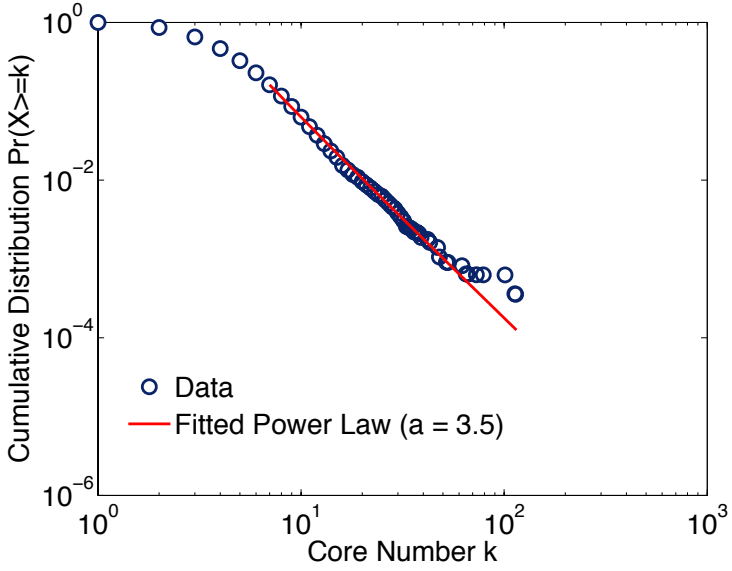
  *2. **while** there is a node x in F such that $\deg_F(x) < k$*

  *   **delete** node x from F*

  *3. **return** F*

- Many efficient algorithms have been given for the computation **[Second part of the tutorial]**
  - E.g., **[Batagelj and Zaversnik, 2003]**
    - *Time complexity: O(m) (m= |E|)*
- Fast! especially in real word data where G is usually sparse

k-core on the DBLP co-authorship graph



- Filtered out 1% of the papers
- Max 15 authors/paper

Kurt Mehlhorn	Joseph S. B. Mitchell	Marc J. van Kreveld
Micha Sharir	David Eppstein	Martin L. Demaine
Pankaj K. Agarwal	Erik D. Demaine	Ferran Hurtado
Mark de Berg	Olivier Devillers	Timothy M. Chan
Rolf Klein	Sándor P. Fekete	Oswin Aichholzer
Mark H. Overmars	Henk Meijer	Bettina Speckmann
Herbert Edelsbrunner	Sariel Har-Peled	Jeff Erickson
Stefanie Wuhrer	John Hershberger	Therese C. Biedl
Jack Snoeyink	Alon Efrat	Greg Aloupis
Joseph O'Rourke	Stefan Langerman	David Bremner
Subhash Suri	Bernard Chazelle	Anna Lubiw
Otfried Cheong	Joachim	Esther M. Arkin
Hazel Everett	Gudmundsson	Boris Aronov
Sylvain Lazard	Giuseppe Liotta	Vida Dujmovic
Helmut Alt	Sue Whitesides	Suneeta Ramaswami
Emo Welzl	Christian Knauer	Thomas C. Shermer
Günter Rote	Raimund Seidel	David R. Wood
Leonidas J. Guibas	Michiel H. M. Smid	Perouz Taslakian
Chee-Keng Yap	Tetsuo Asano	John Iacono
Danny Krizanc	David Rappaport	Sergio Cabello
Pat Morin	Vera Sacristan	Sébastien Collette
Jorge Urrutia	Hee-Kap Ahn	Belén Palop
Diane L. Souvaine	Prosenjit Bose	Mirela Damian
Ileana Streinu	Michael A. Soss	Jirí Matousek
Dan Halperin	Godfried T. Toussaint	Otfried Schwarzkopf
Hervé Brönnimann		Richard Pollack

Authors in the maximal k-core subgraph

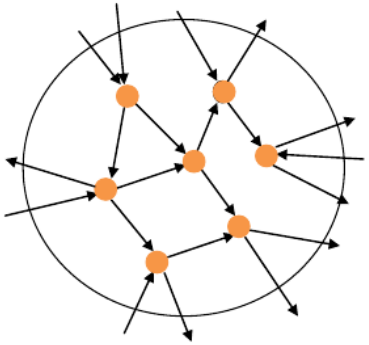
DBLP k-cores

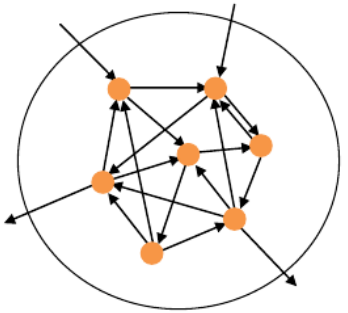
- Extreme k-core: k=15 (DBLP), 76 authors
- **Author ranking metric:** max(k)-core that an author belongs to
  - e.g., Paul Erdős: 14
- On the max(k)-core we can identify the "closest" collaborators**: Hop-1 community**
  - **Erdős hop-1**:

    Boris Aronov, Daniel J. Kleitman, János Pach, Leonard J. Schulman, Nathan Linial, Béla Bollobás, Miklós Ajtai, Endre Szemerédi, Joel Spencer,Fan R. K. Chung, Ronald L. Graham, David Avis, Noga Alon, László Lovász, Shlomo Moran, Richard Pollack, Michael E. Saks, Shmuel Zaks, Peter Winkler, Prasad Tetali, László Babai

k-core for directed networks

- Directed graphs:
  - Wikipedia
  - DBLP – Citation network





- Is there a degeneracy notion for directed graphs?
- We extend the k-core concept in directed graphs by applying a limit on in/out edges respectively
  - This provides a **two dimensional range** where cores degenerate
- Trade off between in/out edges can give us a more specific view of the cohesiveness and the “social" behavior

Degeneracy on directed graphs

(k,l)-D-*core* (G): the (k,l) D-core of graph G for each k,l : $dc_{k,l}$ = | (k,l)-D-*core* (G)|

D-core matrix: D(k,l)=$dc_{k,l}$, k,l integers – each cell stores the size of the respective D-core

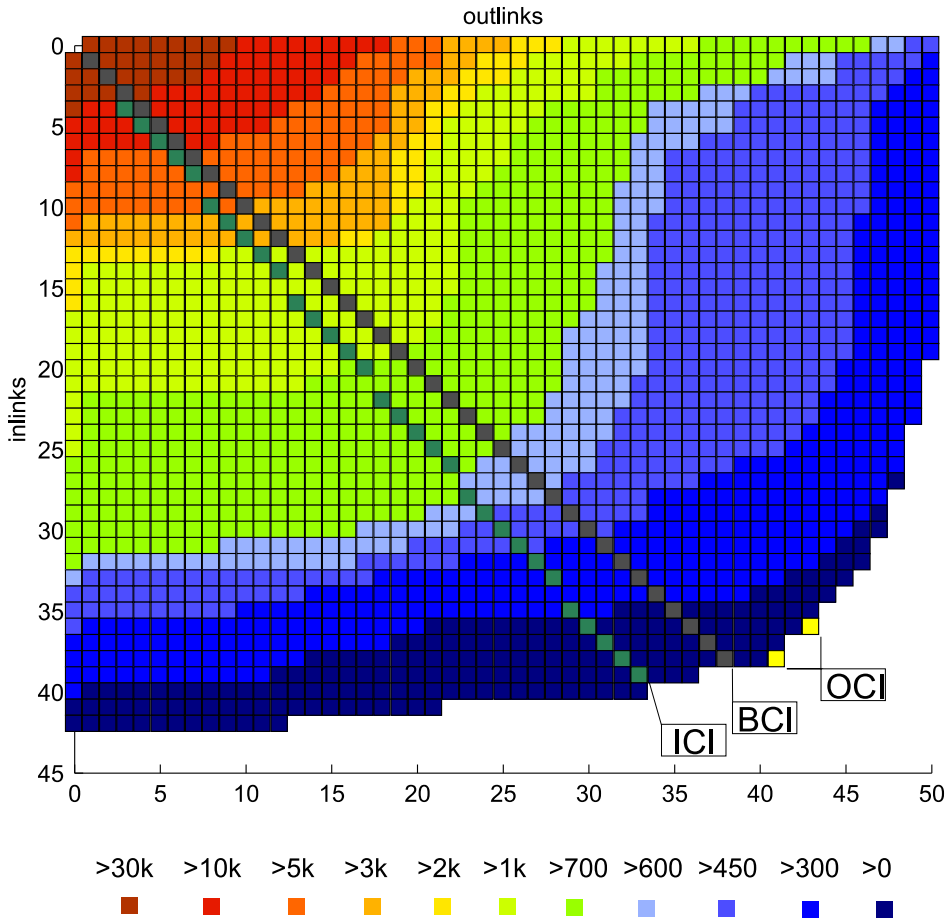
Frontier: F(D) = {(k;l): $dc_{k,l} > 0$ & $dc_{k+1,l+1} = 0$ } : the extreme (k,l)-D-cores

Collaboration indices

- Balanced collaboration index (**BCI**) : Intersection of diagonal D(k,k) with frontier
- Optimal collaboration index (**OCI**) : DC(k,l) where max(($k+l$)/2) distance from D($0,0$)
- Inherent collaboration index (**ICI**): All cores on the angle defined by the average in-links/out-links ratio

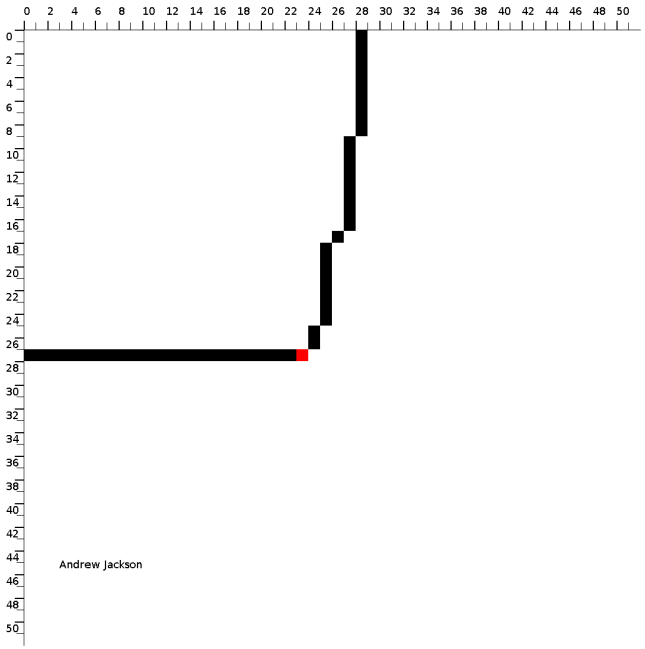
D-core matrix of Wikipedia graph

Extend the notion of degeneracy in directed graphs: (k, l)-D-Core

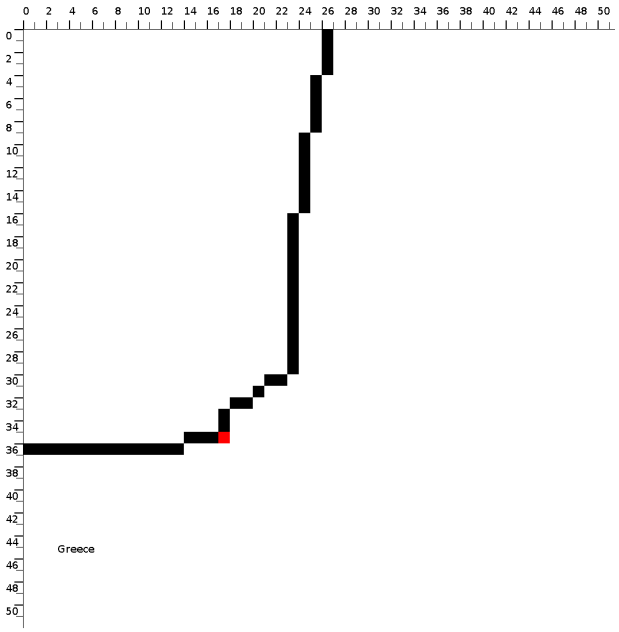


The extreme D-core(38,41) contains 237 Wikipedia articles

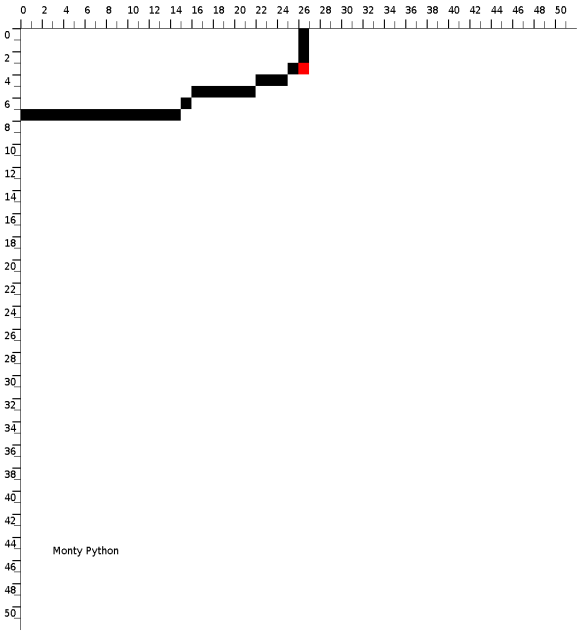
Thematic D-core frontiers in Wikipedia (1/2)



"Andrew Jackson"



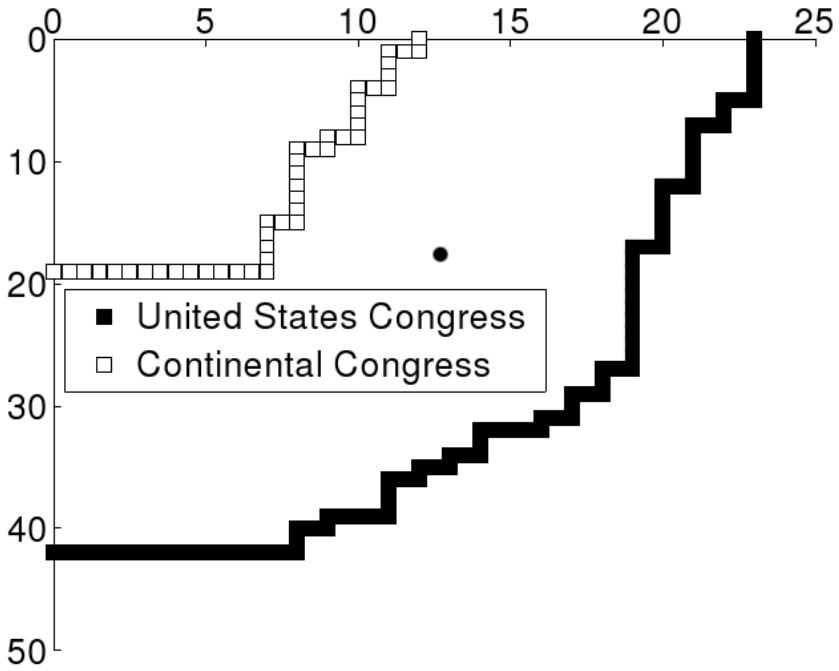
"Greece"

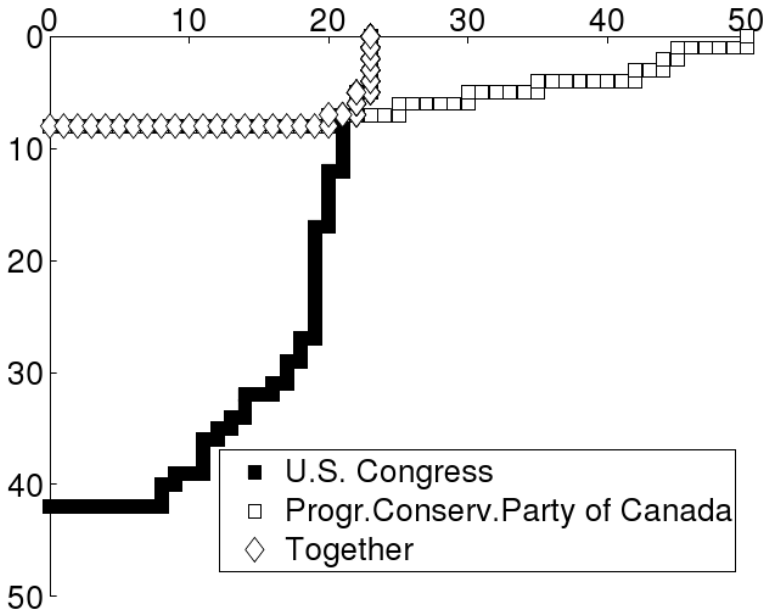


"Monty Pythons"

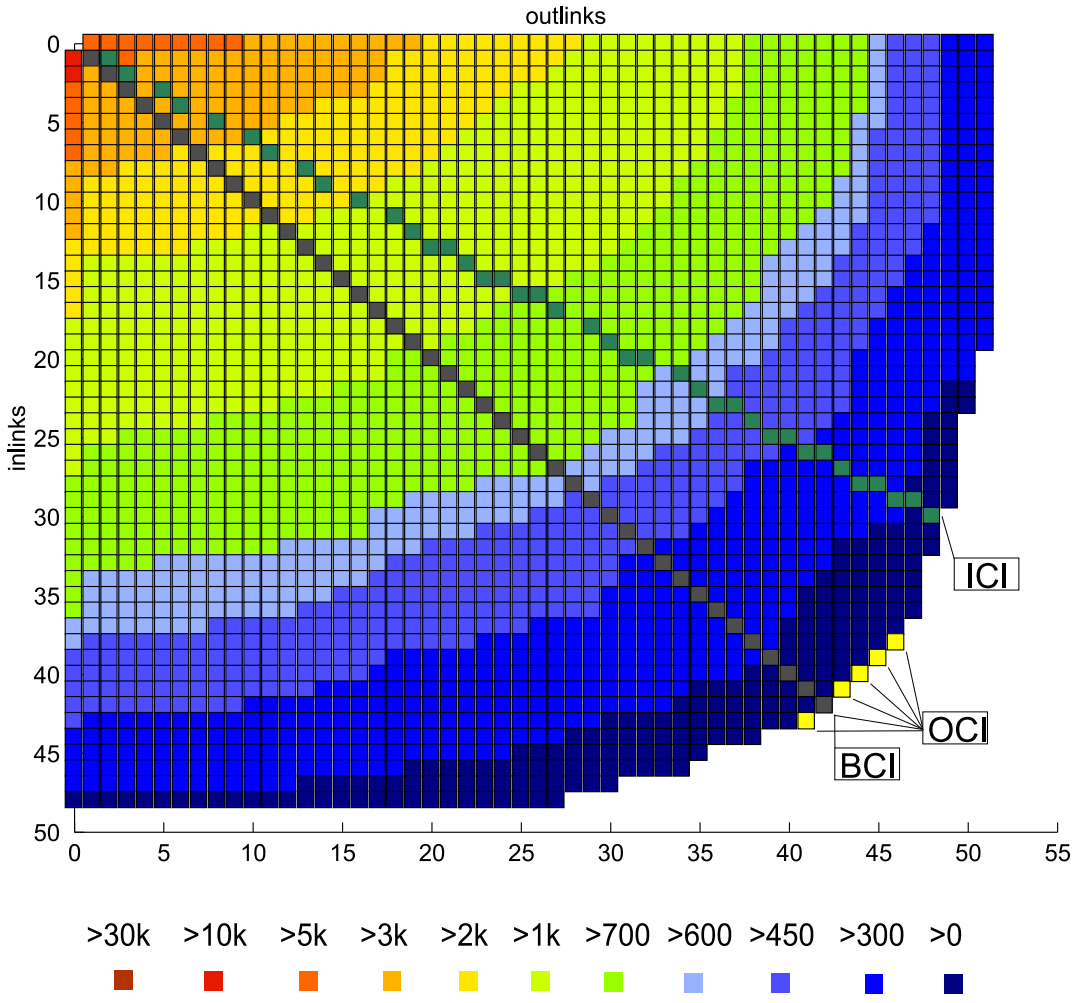
Compare different articles

Thematic D-core frontiers in Wikipedia (2/2)





D-core matrix of the DBLP graph



The extreme DBLP D-core authors

Authoritative and Collaborative Scientists

José A. Blakeley
Hector Garcia-Molina
Abraham Silberschatz
Umeshwar Dayal
Eric N. Hanson
Jennifer Widom
Klaus R. Dittrich
Nathan Goodman
Won Kim
Alfons Kemper
Guido Moerkotte
Clement T. Yu
M. Tamer Ã Zsu
Amit P. Sheth
Ming-Chien Shan
Richard T. Snodgrass
David Maier
Michael J. Carey
David J. DeWitt
Joel E. Richardson
Eugene J. Shekita
Waqar Hasan
Marie-Anne Neimat
Darrell Woelk
Roger King
Stanley B. Zdonik
Lawrence A. Rowe
Michael Stonebraker
Serge Abiteboul
Richard Hull
Victor Vianu
Jeffrey D. Ullman
Michael Kifer
Philip A. Bernstein
Vassos Hadzilacos
Elisa Bertino
Stefano Ceri
Georges Gardarin

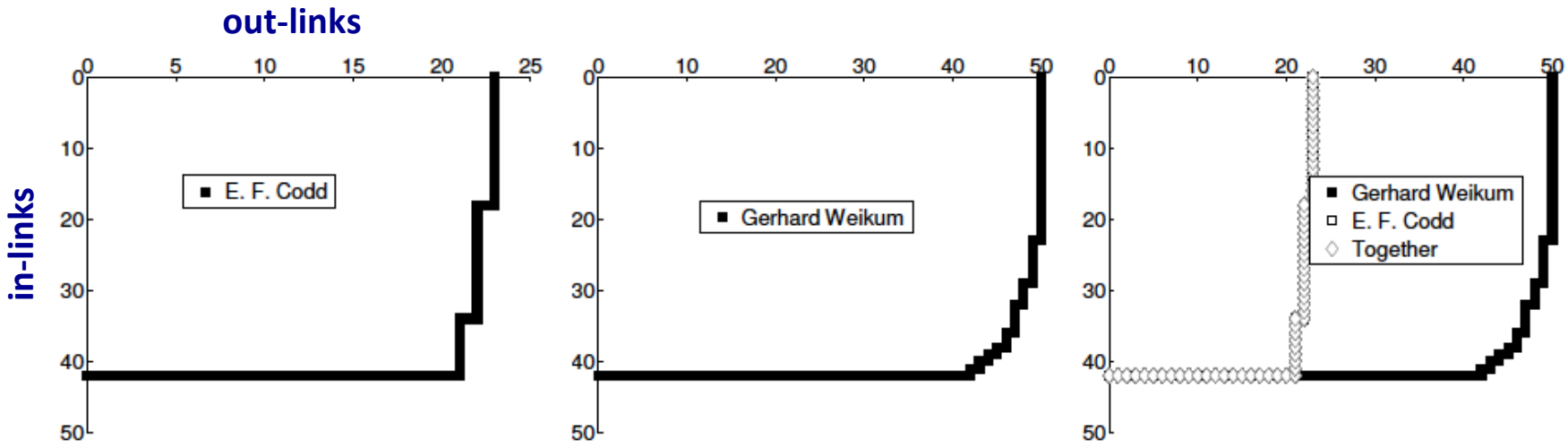
Patrick Valduriez
Ramez Elmasri
Richard R. Muntz
David B. Lomet
Betty Salzberg
Shamkant B. Navathe
Arie Segev
Gio Wiederhold
Witold Litwin
Theo Härder
François Bancilhon
Raghu Ramakrishnan
Michael J. Franklin
Yannis E. Ioannidis
Henry F. Korth
S. Sudarshan
Patrick E. O'Neil
Dennis Shasha
Shamim A. Naqvi
Shalom Tsur
Christos H. Papadimitriou
Georg Lausen
Gerhard Weikum
Kotagiri Ramamohanarao
Maurizio Lenzerini
Domenico Saccà
Giuseppe Pelagatti
Paris C. Kanellakis
Jeffrey Scott Vitter
Letizia Tanca
Sophie Cluet
Timos K. Sellis
Alberto O. Mendelzon
Dennis McLeod
Calton Pu
C. Mohan
Malcolm P. Atkinson
Doron Rotem

Michel E. Adiba
Kyuseok Shim
Goetz Graefe
Jiawei Han
Edward Sciore
Rakesh Agrawal
Carlo Zaniolo
V. S. Subrahmanian
Claude Delobel
Christophe Lecluse
Michel Scholl
Peter C. Lockemann
Peter M. Schwarz
Laura M. Haas
Arnon Rosenthal
Erich J. Neuhold
Hans-Jorg Schek
Dirk Van Gucht
Hamid Pirahesh
Marc H. Scholl
Peter M. G. Apers
Allen Van Gelder
Tomasz Imielinski
Yehoshua Sagiv
Narain H. Gehani
H. V. Jagadish
Eric Simon
Peter Buneman
Dan Suciu
Christos Faloutsos
Donald D. Chamberlin
Setrag Khoshafian
Toby J. Teorey
Randy H. Katz
Miron Livny
Philip S. Yu
Stanley Y. W. Su
Henk M. Blanken

Peter Pistor
Matthias Jarke
Moshe Y. Vardi
Daniel BarbarÃ¡
Uwe Deppisch
H.-Bernhard Paul
Don S. Batory
Marco A. Casanova
Joachim W. Schmidt
Guy M. Lohman
Bruce G. Lindsay
Paul F. Wilms
Z. Meral Özsoyoglu
Gultekin Özsoyoglu
Kyu-Young Whang
Shahram Ghandeharizadeh
Tova Milo
Alon Y. Levy
Georg Gottlob
Johann Christoph Freytag
Klaus Küspert
Louiqa Raschid
John Mylopoulos
Alexander Borgida
Anand Rajaraman
Joseph M. Hellerstein
Masaru Kitsuregawa
Sumit Ganguly
Rudolf Bayer
Raymond T. Ng
Daniela Florescu
Per-Åke Larson
Hongjun Lu
Ravi Krishnamurthy
Arthur M. Keller
Catriel Beeri
Inderpal Singh Mumick
Oded Shmueli

George P. Copeland
Peter Dadam
Susan B. Davidson
Donald Kossmann
Christophe de Maindreville
Yannis Papakonstantinou
Kenneth C. Sevcik
Gabriel M. Kuper
Peter J. Haas
Jeffrey F. Naughton
Nick Roussopoulos
Bernhard Seeger
Georg Walch
R. Erbe
Balakrishna R. Iyer
Ashish Gupta
Praveen Seshadri
Walter Chang
Surajit Chaudhuri
Divesh Srivastava
Kenneth A. Ross
Arun N. Swami
Donovan A. Schneider
S. Seshadri
Edward L. Wimmers
Kenneth Salem
Scott L. Vandenberg
Dallan Quass
Michael V. Mannino
John McPherson
Shaul Dar
Sheldon J. Finkelstein
Leonard D. Shapiro
Anant Jhingran
George Lapis

D-Core frontier for individuals



- The **frontier of an individual**: defined by the outmost D-cores that the individual belongs to
- We can evaluate the citation based robustness of an individual within the community of her/his frontier

Demo at: **www.graphdegeneracy.org**

Degeneracy in signed graphs

- Signed graphs can depict a wide variety of concepts
  - Positive/negative interactions among individuals
  - Common behavior in product review websites (e.g., epinions.com)
- A member of a directed signed graph G can either trust or distrust another but not both simultaneously
- Each vertex **v** has both positive & negative in-degree and both positive & negative out-degree
- **Our solution:** we define and extend the degeneracy concept upon a trust network

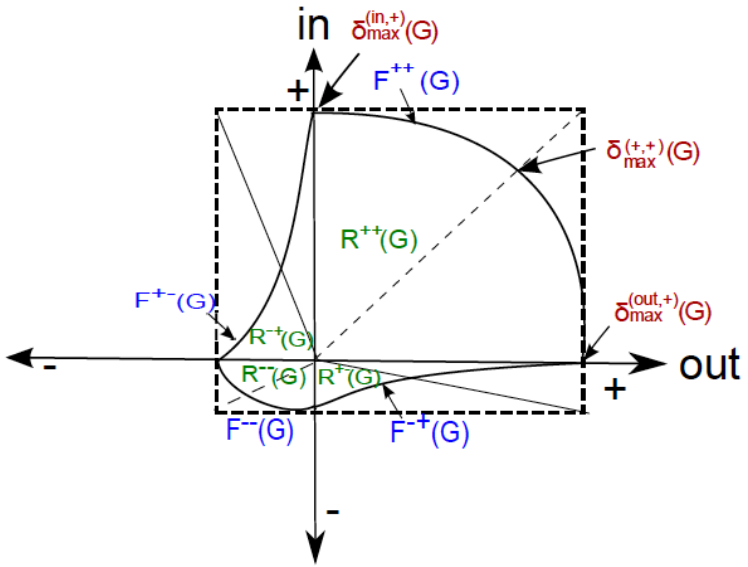
S-cores structure

- We compute the trust network degeneracy along the 4 combinations of **direction** (in, out) and **sign**:
  - **(+,+)**: Mutual Trust
  - **(+,-)**: Trust under distrust (i.e., trust those who do not trust me)
  - **(-,-)**: Mutual distrust
  - **(-,+)**: Distrust under trust

Definitions

- Given a pair **(s, t)** $\in \{+,-\}^2$, we define the **(s, t)**-degeneracy of **G**

- 4 quadrants: We define frontiers $\mathbf{F_G}$ as in D-cores
  - Area under frontier :
    - The extreme non-empty S-cores



Statistics of datasets

Explicit

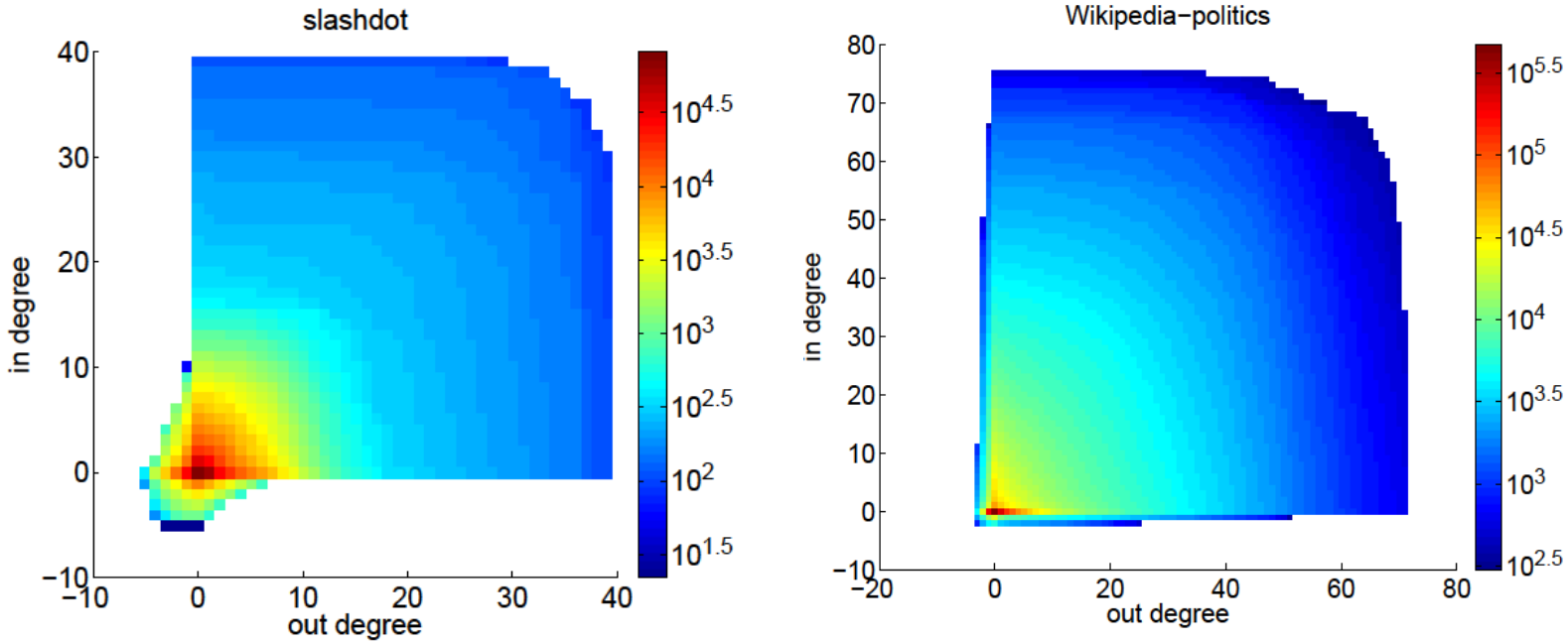
Network	Nodes	Edges	Negative
Epinions	119,217	841,200	15.0%
Slashdot	82,144	549,202	22.6%

Implicit (Wikipedia)

Domain	Articles	Nodes	Edges	Positive	Negative
History	3,331	141,983	534,693	439,193	95,500
Politics	12,921	453,116	2,428,945	2,099,410	329,535
Religion	6,459	277,482	1,423,279	1,244,166	179,113
Mathematics	9,610	158,671	651,450	548,073	103,377

Examples

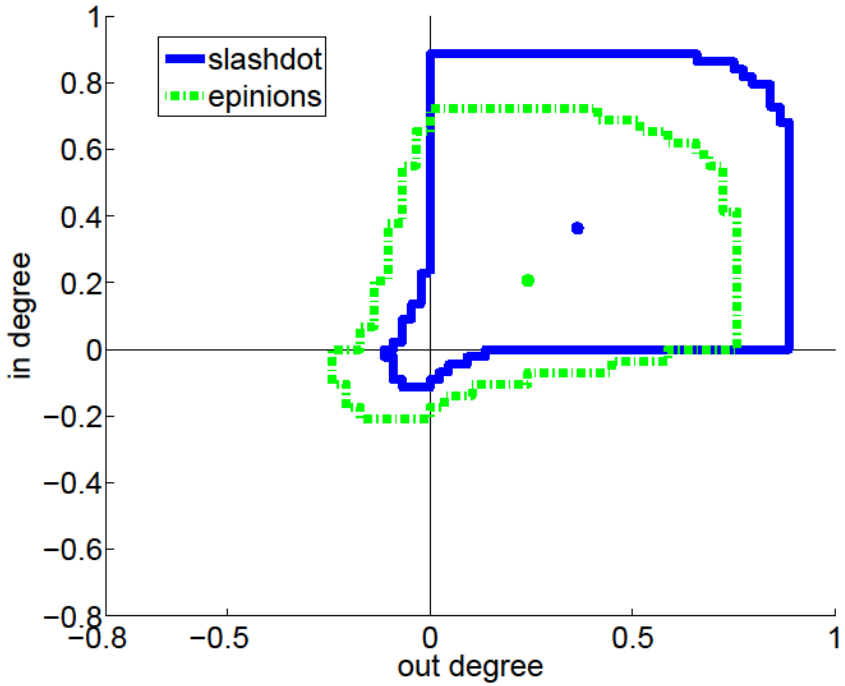
S-Cores sizes on real world data



Observations:

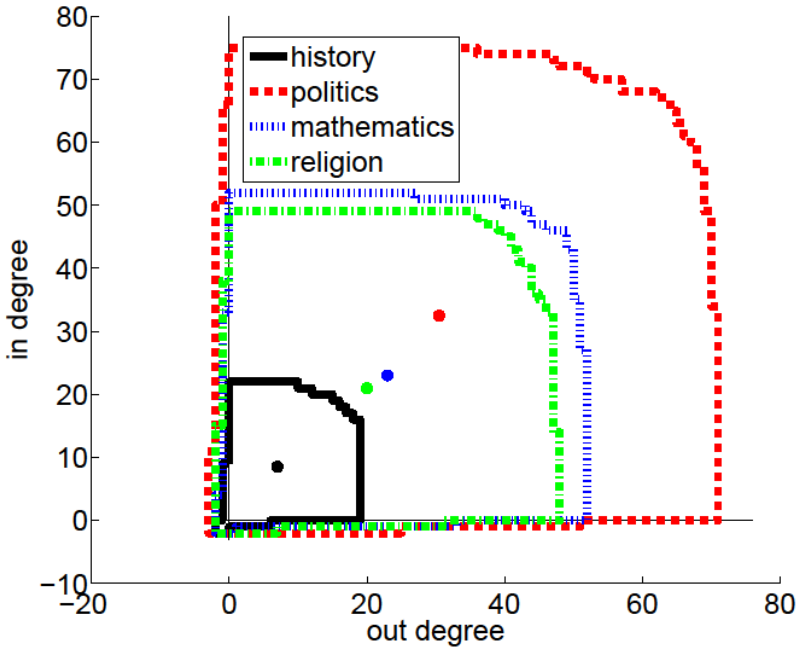
- In both cases positive trust dominates
- In slashdot there is proportionaly much more mutual distrust than in the wikipedia-politics case

Frontiers (explicit graphs)

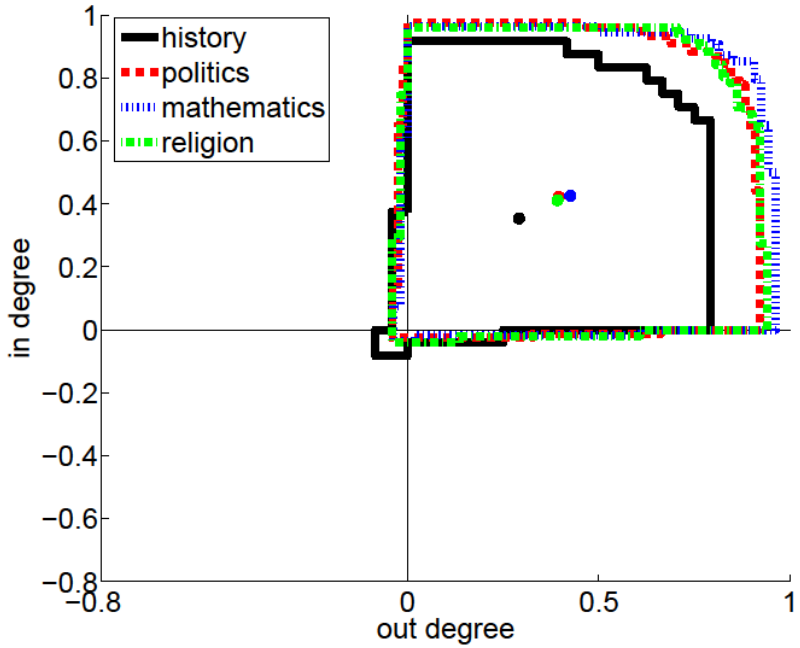


- Slashdot is more robust under degeneracy than the epinions trust graph: i.e., there is denser mutual trust in S than in E
- In epinions the mutual negative distrust is much more important as well as the imbalanced trust graphs (+/-, -/+)

Evaluate Wikipedia topics



Original frontiers



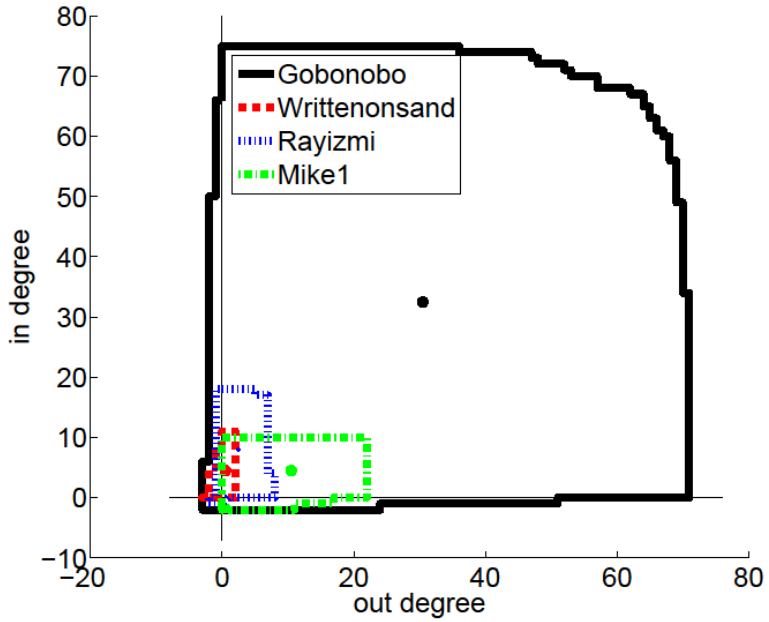
Normalized

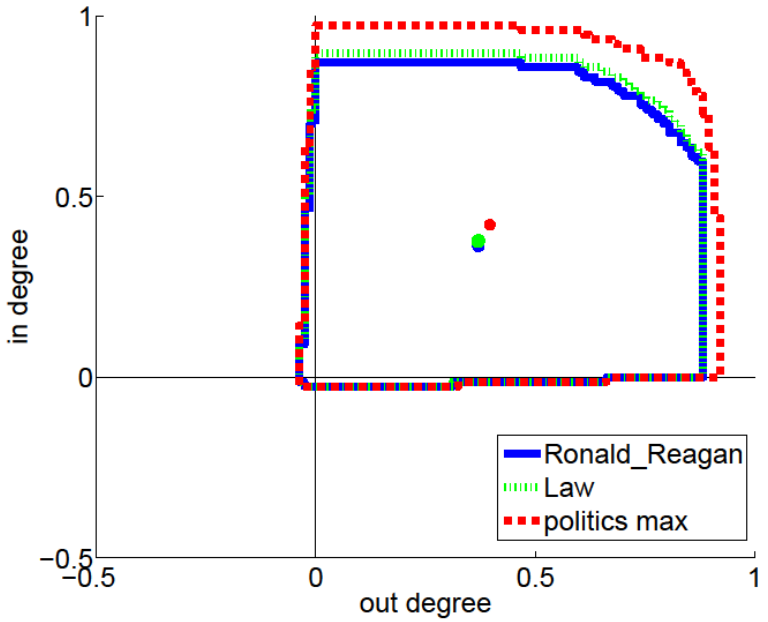
- In the original frontiers, Wikipedia *politics* is the most robust trust network, *history* is the least one
- In the normalized case: history is the one with the largest mutually negative trust constituent

Wikipedia: users and articles frontiers

- We utilize the s-core structure to evaluate the trustworthiness of a **user** or an **article**
- **User's frontier:**
  - The user frontier is defined by intersection of the s-cores she participates
- **Article's frontier:**
  - Multiple users contribute to an article
  - The intersection of the contributing editors individual frontiers is the article frontier

Editors and articles





- **Editors**
  - ''Gobonobo'' is by far the most trusting and trusted one – i.e., a very senior editor
- **Article frontier**
  - "Reagan" article is almost as trusted as the "Politics" topic

Generalized cores (1/2)

- The k-core decomposition was initially introduced for the property of node degree
- **Generalized cores** of graph **G=(V,E)**
  - Let **p** be a node property function: **$p(u, U)$**, **$v \in V$**, **$U \subseteq V$**
  - Examples:
    1. $p_1 (v, U) = deg_U (v)$
    2. $p_2 (v, U) = \text{in-}deg_U (v)$
    3. $p_3 (v, U) = \text{out-}deg_U (v)$
    4. $p_4 (v, U) = \sum_{u \in N(v, U)} w(v, u)$
    5. $p_5 (v, U) =$ # of cycles of length k trough node v

Generalized cores (2/2)

- Subgraph **H** induced by the set is a **p**-core at level **t** iff
  - $v \in C : t \leq p(v, C)$
  - **C** is a maximal set
- Function **p** is monotone iff
  - $C_1 \subset C_2 \rightarrow \forall v \in V: (p(v, C_1) <= p(v, C_2))$
  - In our case (p_1-p_5), all functions are monotone

1. For monotone functions, the **p**-core at level **t** can be determined by successively deleting vertices of value of **p** lower that **t**
2. For monotone functions **p**, the cores are nested

[Batagelj and Zaversnik '02]

References (Fundamentals)

- C. Giatsidis, D. Thilikos, M. Vazirgiannis,"D-cores: Measuring Collaboration of Directed Graphs Based on Degeneracy", Knowledge and Information Systems Journal, Springer, 2012.
- Christos Giatsidis, Dimitrios M. Thilikos, Michalis Vazirgiannis: D-cores: Measuring Collaboration of Directed Graphs Based on Degeneracy. In: ICDM, 2011.
- Christos Giatsidis, Klaus Berberich, Dimitrios M. Thilikos, Michalis Vazirgiannis: Visual exploration of collaboration networks based ongraph degeneracy. In: KDD, 2012.
- Christos Giatsidis, Dimitrios M. Thilikos, Michalis Vazirgiannis: Evaluating Cooperation in Communities with the k-Core Structure. In: ASONAM, 2011.
- S.B. Seidman. Network Structure and Minimum Degree. Social Networks, 1983.
- V. Batagelj and M. Zaveršnik. Generalized Cores, arXiv, 2002.

- An online demo at: **http://www.graphdegeneracy.org/**

Outline

1. Introduction & Motivation
2. Fundamental Concepts and Definitions of Core Decomposition
3. **Algorithms**
4. Applications
5. Open Topics and Future Research Directions

Preliminaries

$G(V,E)$: the graph

V : vertex (node) set

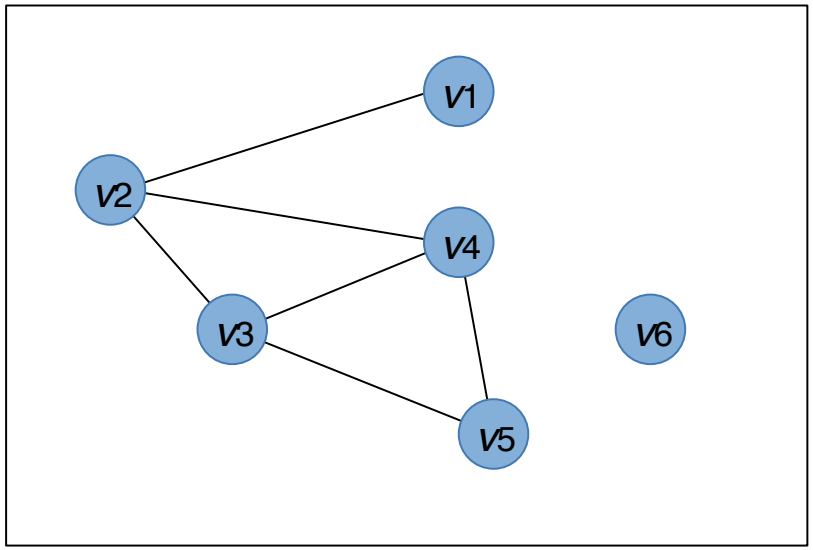
E : edge set

n : number of nodes ($|V|$)

m : number of edges ($|E|$)

v_j : the j-th vertex

v, u, w : some vertices

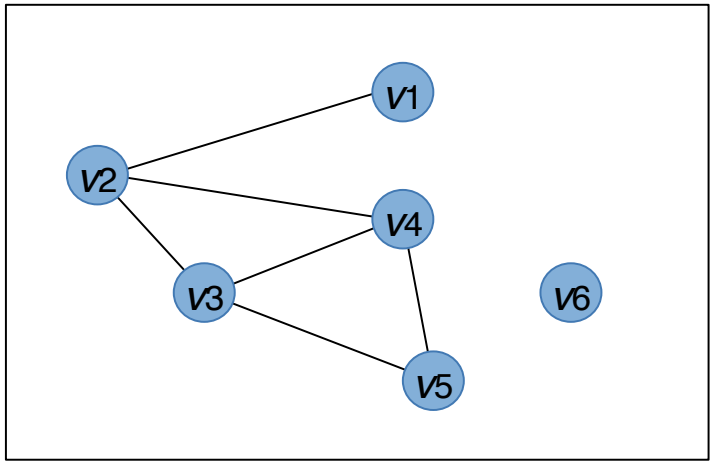


$deg(u)$: degree of vertex u (i.e., number of neighbors)

$c(u)$: core number of u

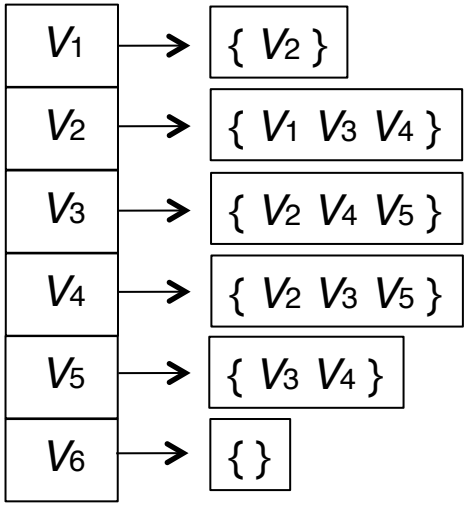
Preliminaries

Data structures for graphs



	1				
1		1	1		
	1		1	1	
	1	1		1	
		1	1		

adjacency matrix



adjacency "lists"

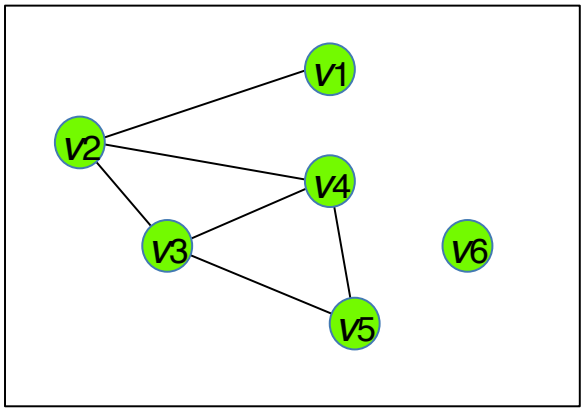
Algorithms

We will cover the following techniques:

- **Brute-Force** Approach
- Main-Memory **Linear**
- Core Decomposition for **Disk-Based** Graphs
- **Local** Core Number Computation
- The **Dynamic** Case
- **Distributed** Core Decomposition
- Cores in **Probabilistic Graphs**

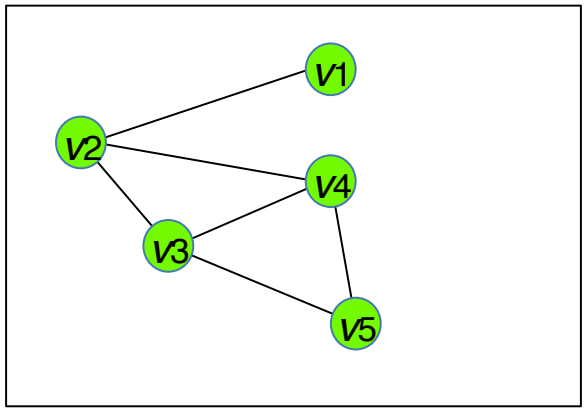
The Brute-Force Algorithm

Basic idea: remove the "weakest" nodes (i.e., the ones with the smallest degree)



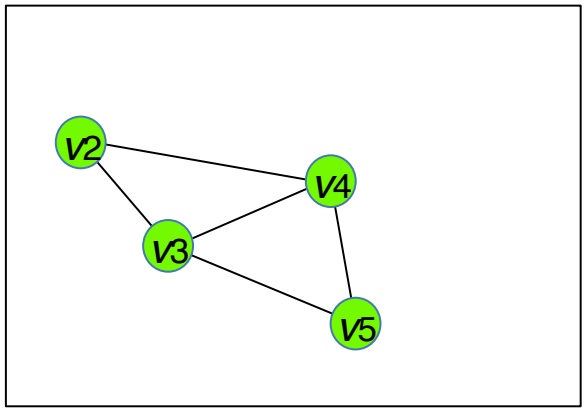
0-core

remove v_6



1-core

remove v_1



2-core
(also the max-core)

If we remove another vertex the graph collapses.

The Brute-Force Algorithm

Complexity

Vertices are stored in a **minheap** prioritized by the degrees. $O(n)$

Remove the node *u* with the smallest degree (top of minheap) and apply deg(*u*) **decrease-key** operations. $O(m \log n)$

Total cost $O(m \log n)$

CAN WE DO BETTER ?

The Linear Algorithm

V. Batagelj, M. Zaversnik,
"An O(m) Algorithm for Cores Decomposition of Networks",
CoRR, cs.DS/0310049, 2002

The Linear Algorithm

Outline

```
compute degrees of vertices;
order nodes in V in non-decreasing order of their degrees;
for each v ∈ V in the order do
     core[v] := degree[v];
     for each u ∈ Neighbors(v) do
          if degree[u] > degree[v] then begin
               degree[u] := degree[u] – 1;
               reorder V accordingly;
          endif
      endfor
endfor
```

The Linear Algorithm

A naïve implementation of the previous algorithm leads to brute-force.

Better implementation leading to O(*m*):

1. use binsort
2. use multiple arrays in main memory

The Linear Algorithm

Let *maxdeg* denote the maximum vertex degree.

Each vertex *u* may belong in one of the *maxdeg* bins, since

1 <= deg(*u*) <= *maxdeg*

Assume *m* > *n* (*we have more edges than nodes*).

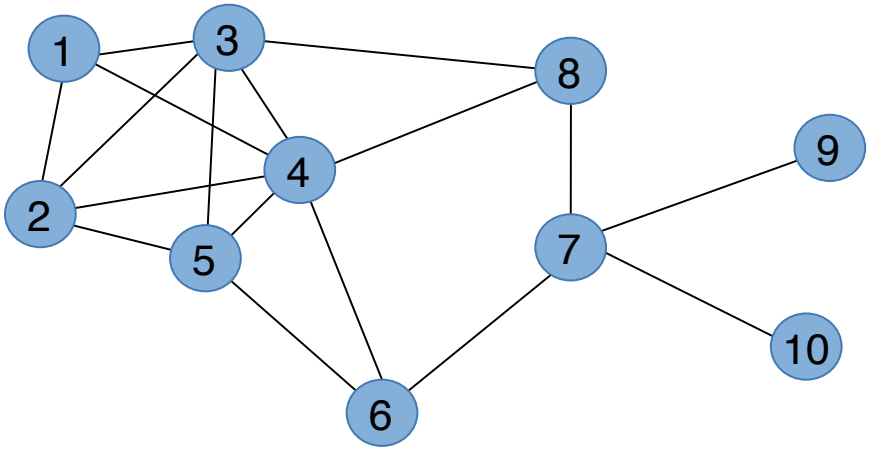
Binsort guarantees **O(*m*)** operations to sort nodes based on degrees.

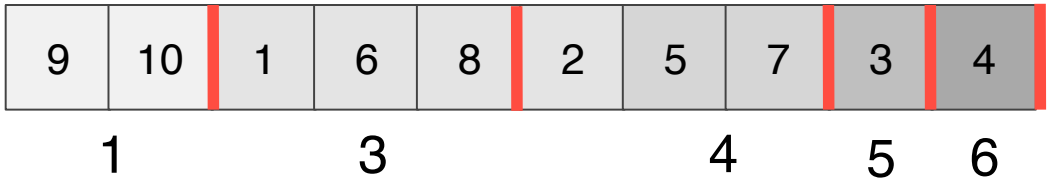
Lowering the degree of a node requires **O(1)** time, and this happens ***m* times**

Time Complexity: O(*m*)

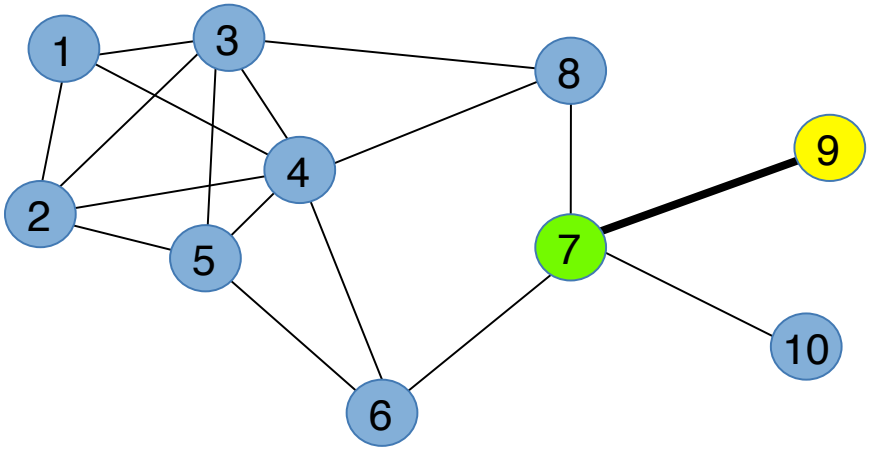
Space Complexity: Ω(*m*) (for the graph)

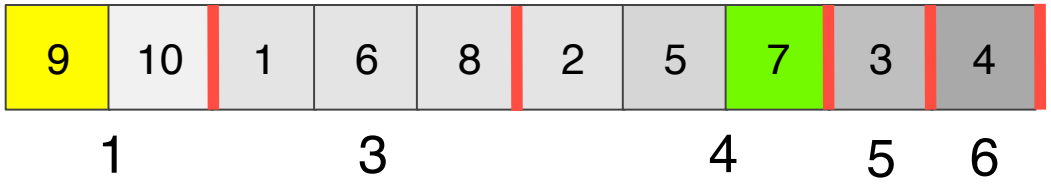
The Linear Algorithm





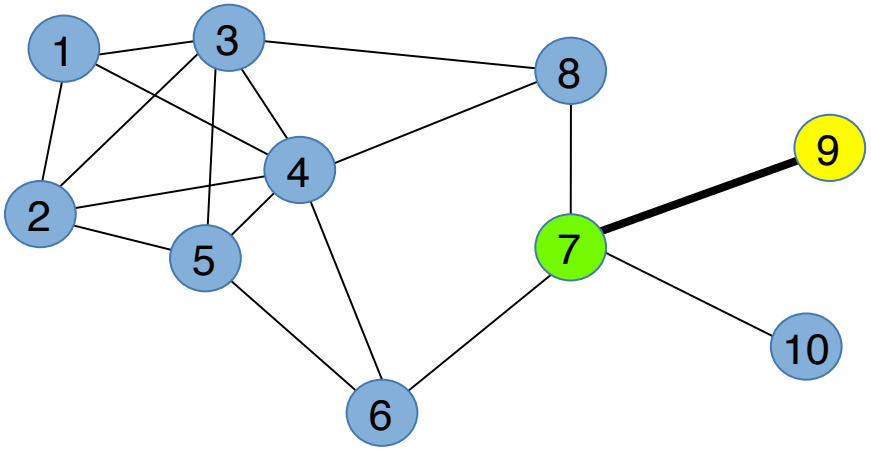
The Linear Algorithm

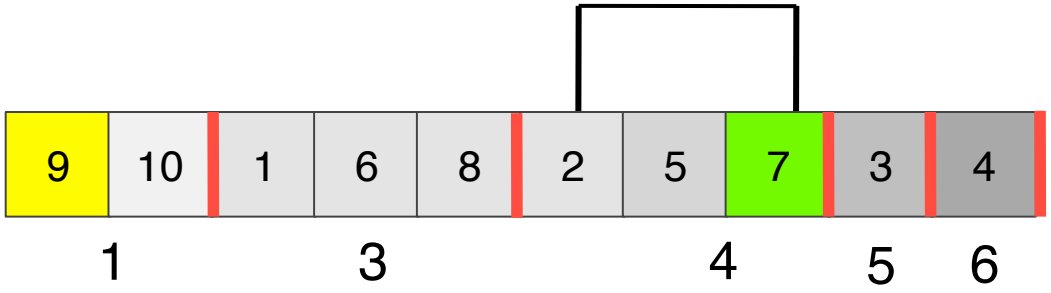




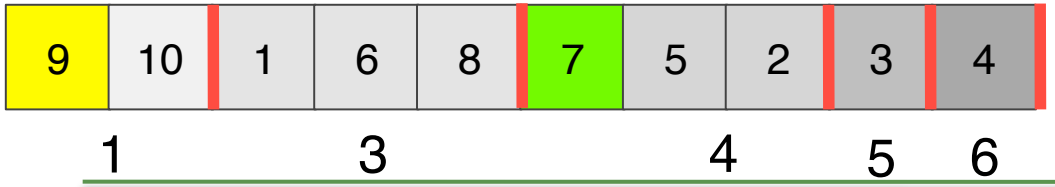
The removal of node 9 leads to decreasing the degree of node 7

The Linear Algorithm



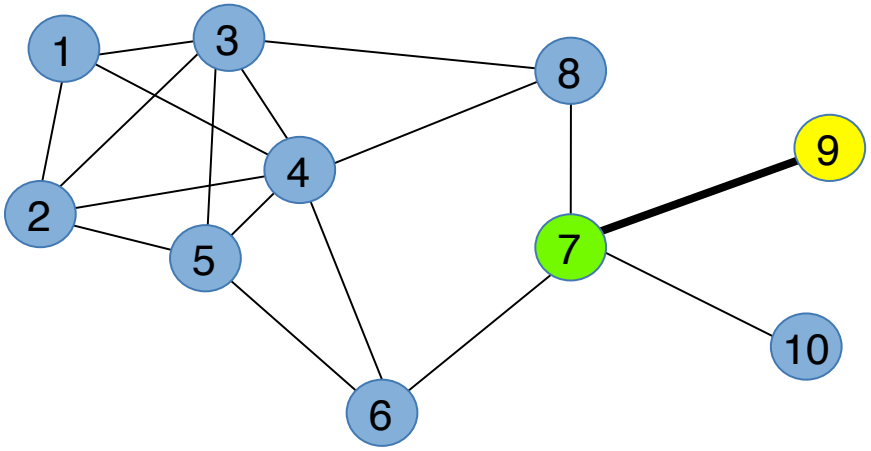


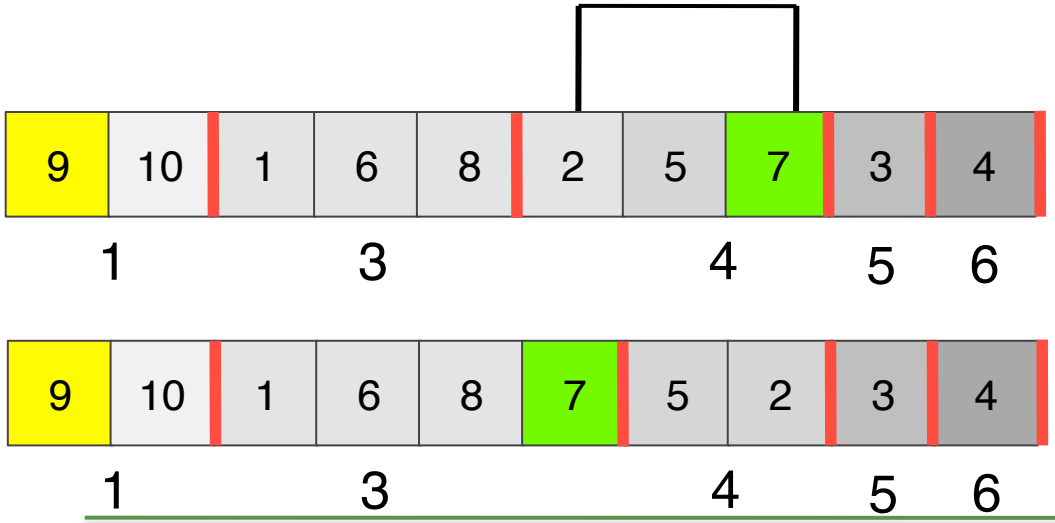
The removal of node 9 leads to decreasing the degree of node 7



Node 7 must change bin

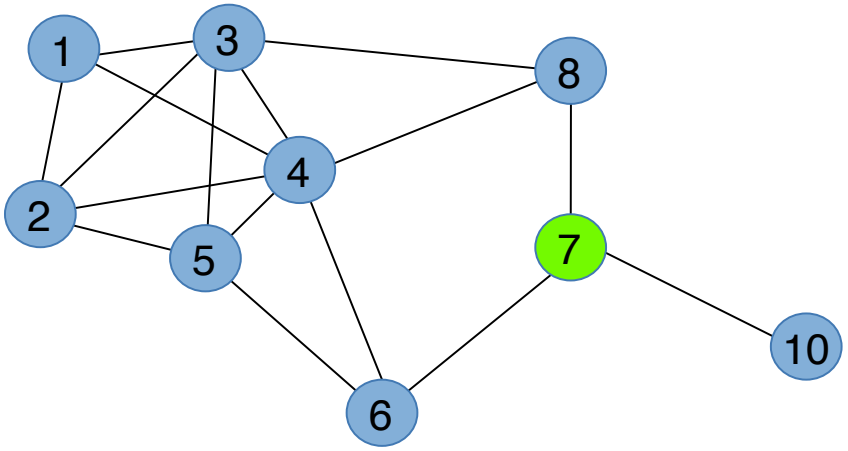
The Linear Algorithm

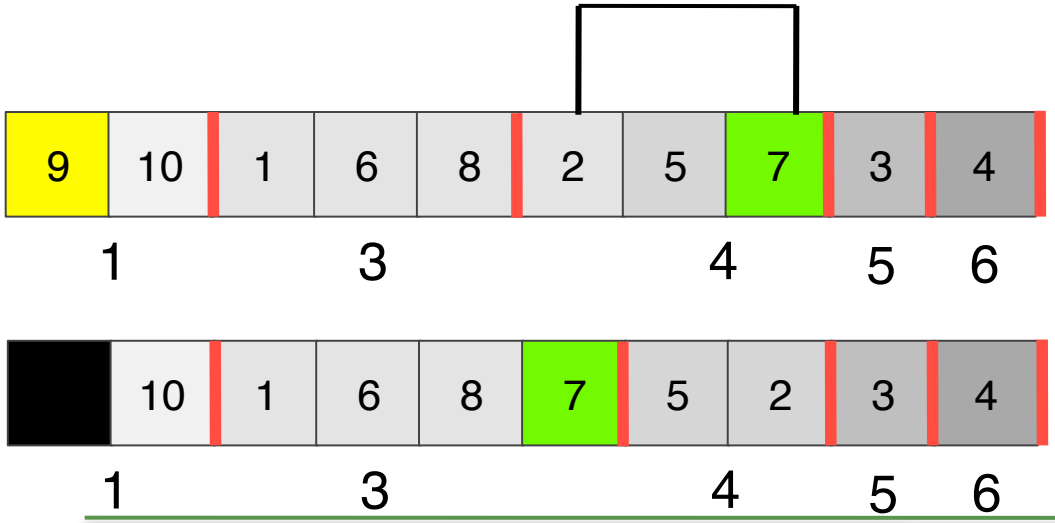




The removal of node 9 leads to decreasing the degree of node 7

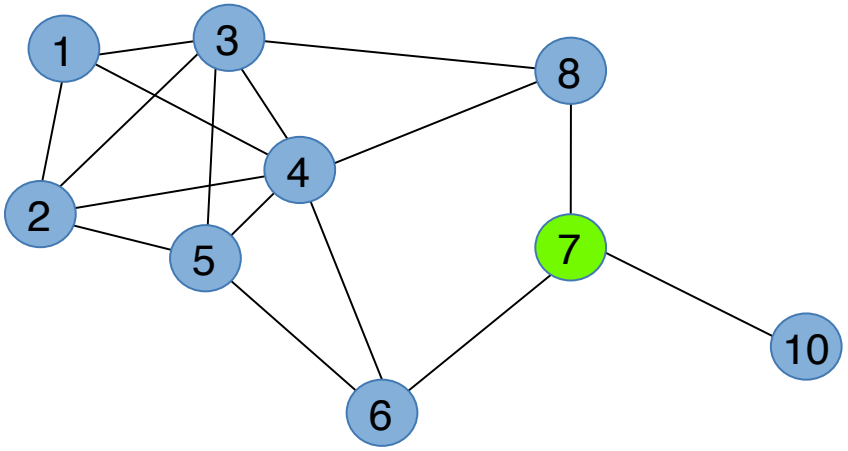
The Linear Algorithm

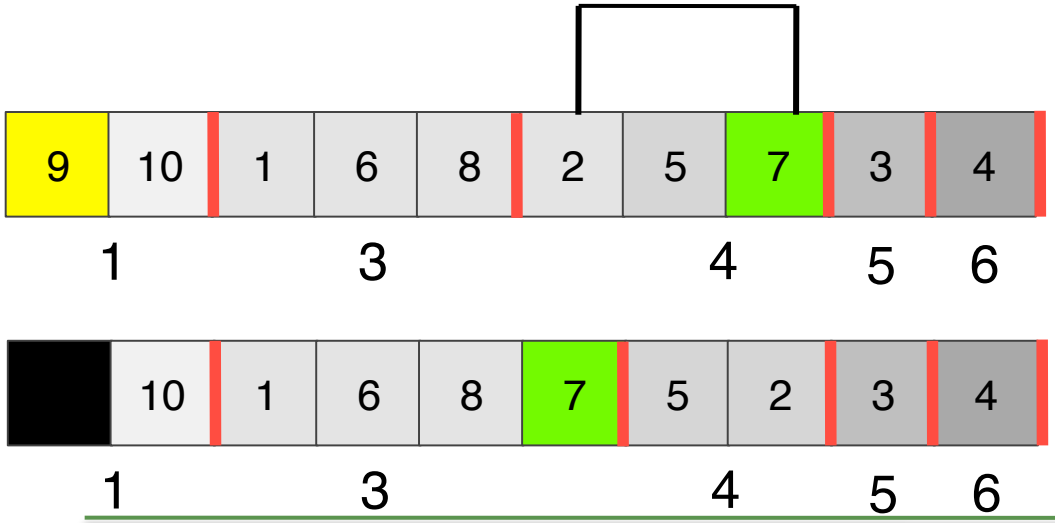




The removal of node 9 leads to decreasing the degree of node 7

The Linear Algorithm





The removal of node 9 leads to decreasing the degree of node 7

The Disk-Based Algorithm

J. Cheng, Y. Ke, S. Chu, T. Ozsu,
"Efficient Core Decomposition in Massive Networks",
Proceedings ICDE, 2011

The Disk-Based Algorithm

So far we assumed that *G* fits in main memory

This is unrealistic for massive graphs required by modern apps

We need an algorithm to compute *k*-cores even **when the graph cannot fit in RAM**

The Disk-Based Algorithm

Can we use a modified version of the Linear algorithm to work for disk-resident graphs?

Yes, but it is expected that it will be inefficient due to excessive use of random I/O operations

The Disk-Based Algorithm

Overview

Phase1: graph partitioning

Phase2: upper bound estimation of core numbers

Phase3: recursive top-down decomposition

The Disk-Based Algorithm

Graph Partitioning

Read the disk-resident graph G once, and partition V into a set of disjoint vertex sets

$$U = \{ U_1, U_2, \ldots, U_t \}$$

The Disk-Based Algorithm

Upper Bound for Core Numbers

For each node u in a partition, find an upper bound $UB(u)$ of its core number. This upper bound is refined progressively at each step of core decomposition.

Initially: $\boldsymbol{UB(u) = deg(u)}$

The Disk-Based Algorithm

Recursive Top-Down Core Decomposition

Compute the *k*-cores with *k* in the range

$$[K_{lower}, K_{upper}]$$

At each recursive step, a subgraph is constructed that is relevant for computing the **real core number** of the nodes *u*, where:

$$K_{lower} <= UB(u) <= K_{upper}$$

K_{lower} is defined using K_{upper} and $b = M / B$

M : main memory capacity (bytes)

B : block size (bytes)

The Disk-Based Algorithm

- It is a **top-down** algorithm: first determines large core numbers
  - Convenient, since we are mostly interested in large core numbers
- It is much more efficient for disk-resident graphs than the linear algorithm
- It can be implemented easily
- It can be used for massive graphs that cannot fit in main memory

Local Computation

M. P. O'Brien, B. D. Sullivan,
"Locally Estimating Core Numbers",
Proceedings ICDM, 2014.

Local Computation

Main Idea

Try to estimate the core number of a node *u* by taking into account the neighborhood of *u*.

Motivation

- interested in specific nodes

- willing to sacrifice accuracy for speed

Local Computation

Main Idea

Try to estimate the core number of a node u by taking into account the neighborhood of u

$N_\delta(u)$ = set of nodes with a **distance** at most δ from u

Local Computation

Main Idea

Try to estimate the core number of a node *u* by taking into account the neighborhood of *u*.

$N_\delta(u)$ = set of nodes with a **distance** at most δ from *u*

Usually the **shortest path distance** is being used

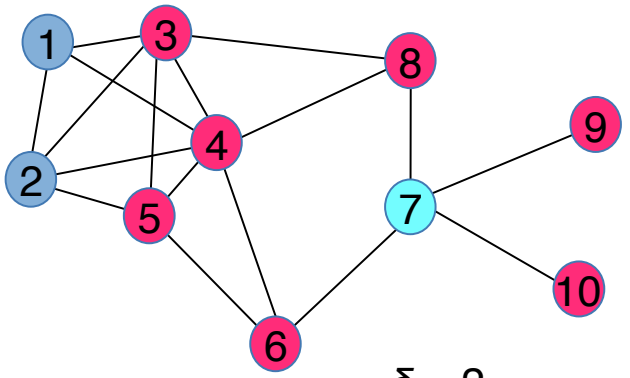
Local Computation

Main Idea

Try to estimate the core number of a node *u* by taking into account the neighborhood of *u*.

$N_\delta(u)$ = set of nodes with a **distance** at most δ from *u*

default is shortest path



Local Computation

Propagating Estimator

Estimation of the **core number** of a node *u* is based on the following formula:

$$k_{\delta}^{+}(u) = \begin{cases} \max\limits_{1 \le i \le d(u)} \left(\min(k_{\delta-1}^{+}(u_i), \deg(u) - i + 1)\right) & \delta > 0 \\ \deg(u) & \delta = 0 \end{cases}$$

where u_1, u_2, … are the neighbors of u ordered by $k_{\delta-1}^{+}$

Local Computation

Complexity

$$O(\delta \cdot |\mathrm{E}_{\delta}(u)|)$$

radius

number of edges induced by $N_{\delta}(u)$

Local Computation

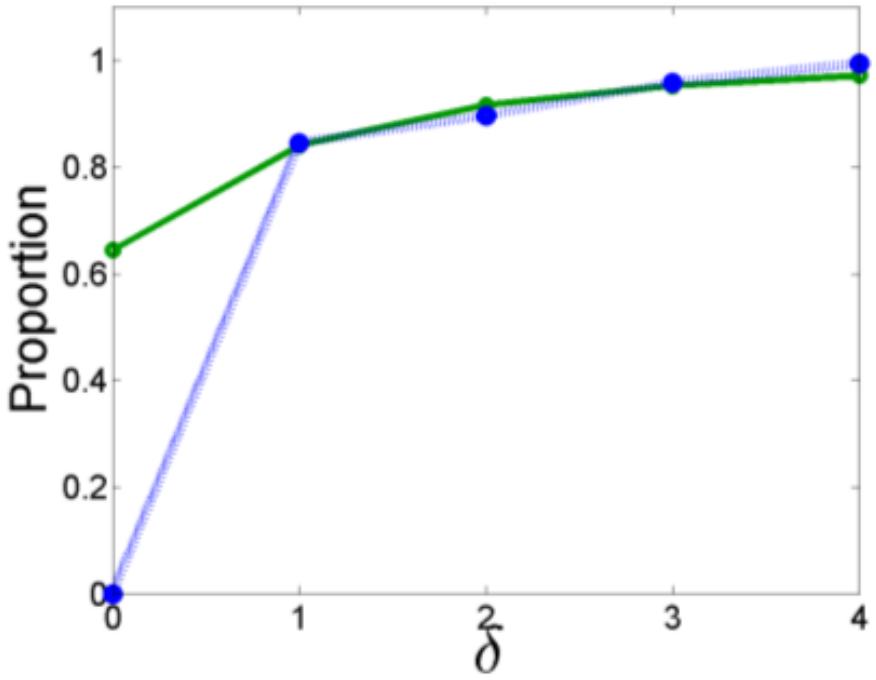
Induced Estimator

Find $N\delta(u)$, and compute the core number of u based only on the subgraph induced by $N\delta(u)$ (ignore the rest of the graph)

$$k_{\delta}^{-}(u)$$

Local Computation

Results for DBLP



blue curve $k_\delta^-(u)$

green curve $k_\delta^+(u)$

Percentage of nodes with the correct core number vs δ

Source: O'Brien et al, 2014

The Dynamic Algorithm

A.E. Sariyuce, B. Gedik, G. Jacques-Silva, et al,
"Streaming Algorithms for k-core Decomposition",
Proceedings VLDB End., 2013.

The Dynamic Algorithm

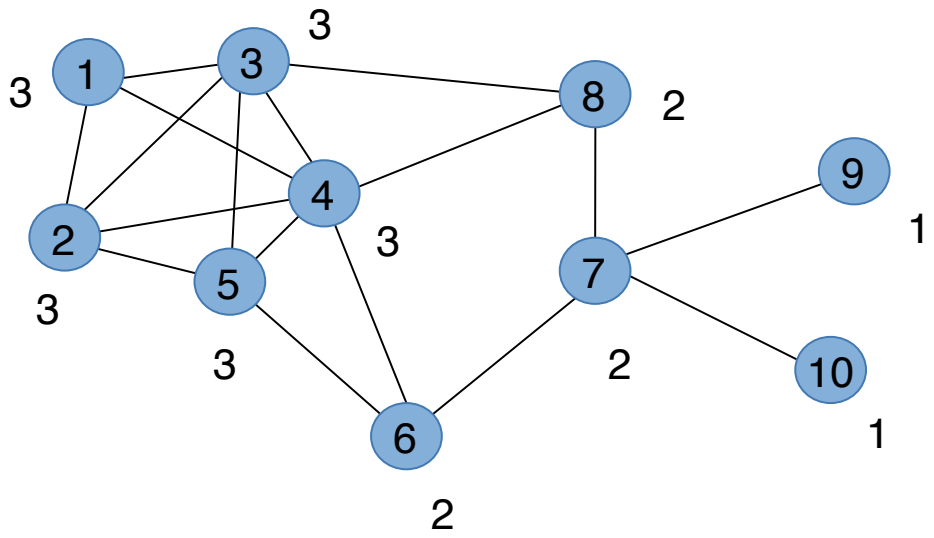
Main Idea

Update core numbers after insertion/deletion of edges/nodes

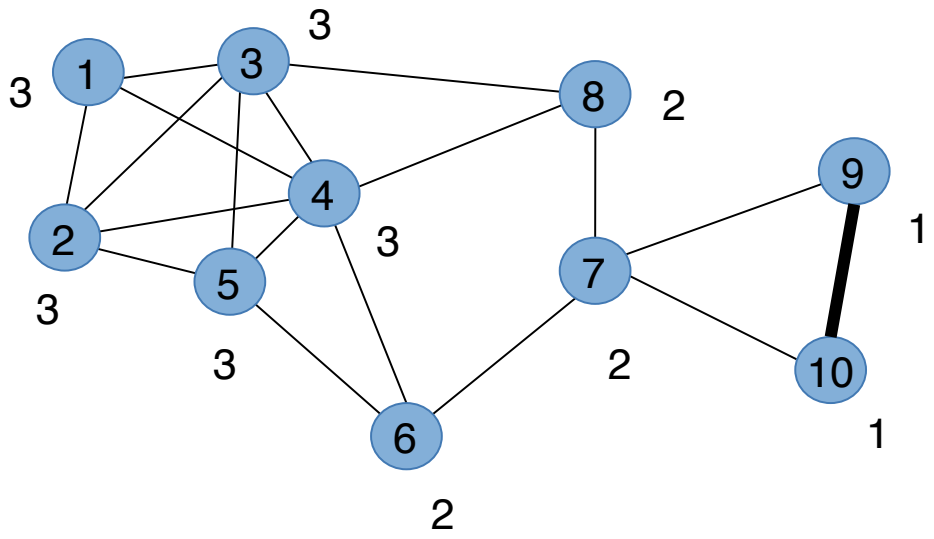
Motivation

- avoid recomputation
- most graphs are dynamic
- enable streaming

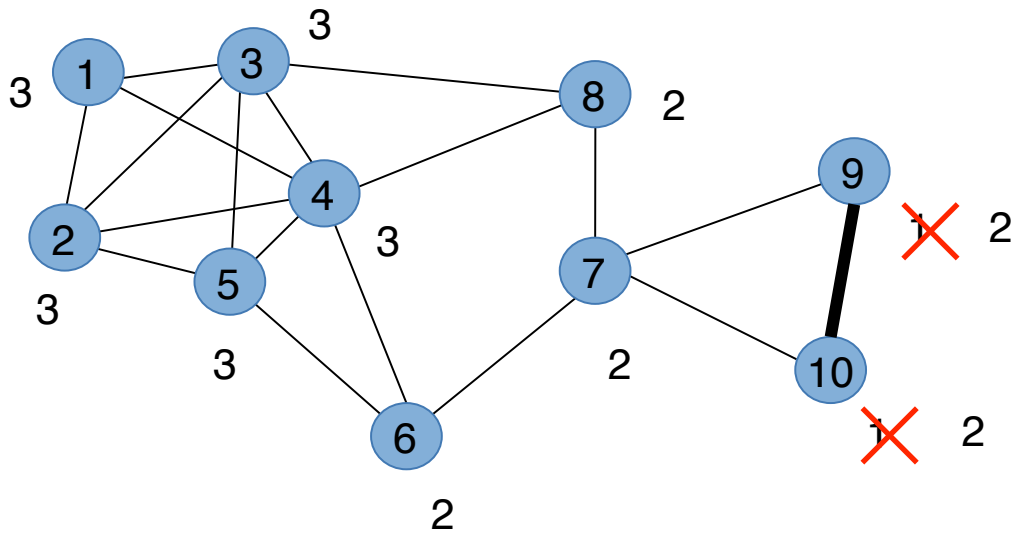
The Dynamic Algorithm



The Dynamic Algorithm



The Dynamic Algorithm



The Dynamic Algorithm

Goal

Determine the set of nodes whose core numbers **may** change. Update core numbers if necessary.

The Dynamic Algorithm

The basic theory behind the algorithm

1. If an edge is inserted to or remover from $G(V,E)$, the core number of any node u can change **by at most one**.

2. If an edge (u, v) is inserted to or removed from $G(V,E)$, where $c(u) < c(v)$, then $c(v)$ **cannot change**.

3. If an edge (u, v) is inserted into $G(V,E)$, then all of the vertices whose core numbers have changed should form a connected subgraph. Similarly, if an edge (u, v) is removed from $G(V,E)$, then all the vertices whose core numbers have changed should form a **connected subgraph**.

4. If an edge (u, v) is inserted (removed) and $c(u) <= c(v)$, then only the vertices w that have $c(w) = c(u)$ and are reachable from u via a path that consists of vertices with core numbers equal to $c(u)$, **may have their core numbers incremented** (decremented).

The Dynamic Algorithm

Three different algorithms are proposed:

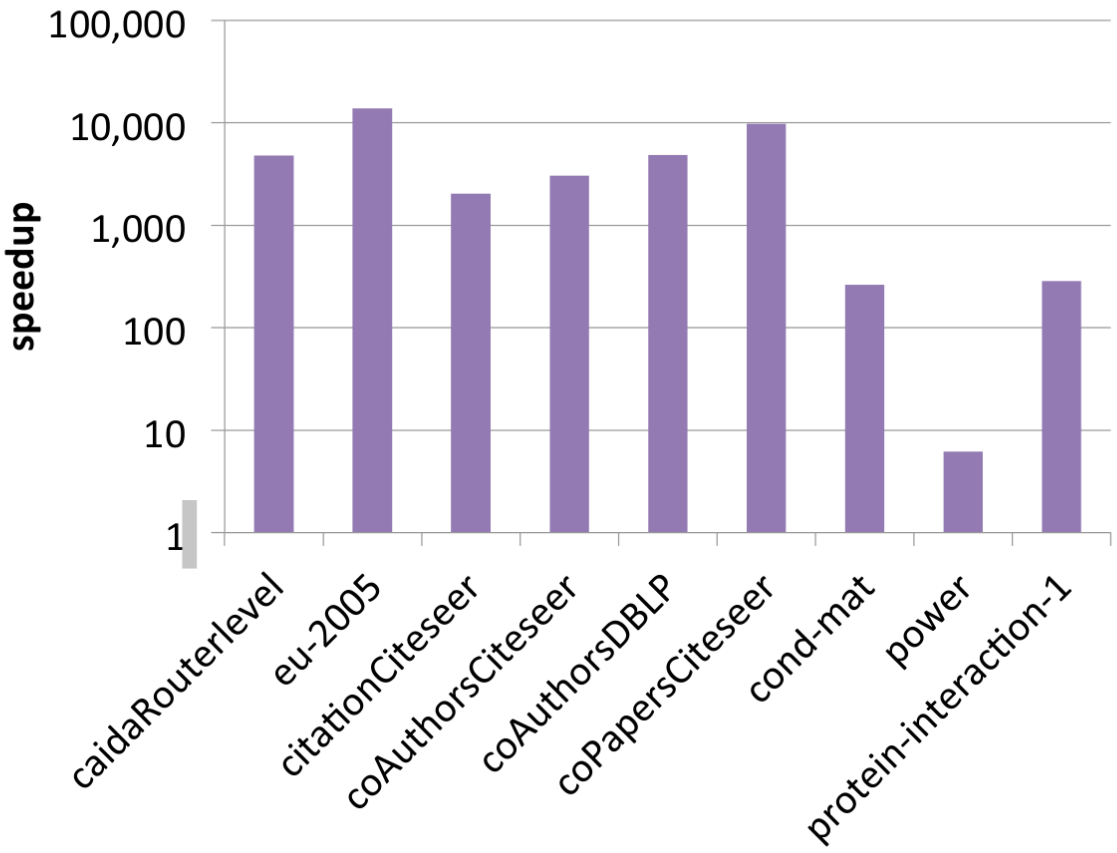
SUBCORE: based on the basic theory

PURECORE: applies some additional optimizations

TRAVERSAL: reduces the number of examined nodes even further (shows the best performance)

The Dynamic Algorithm

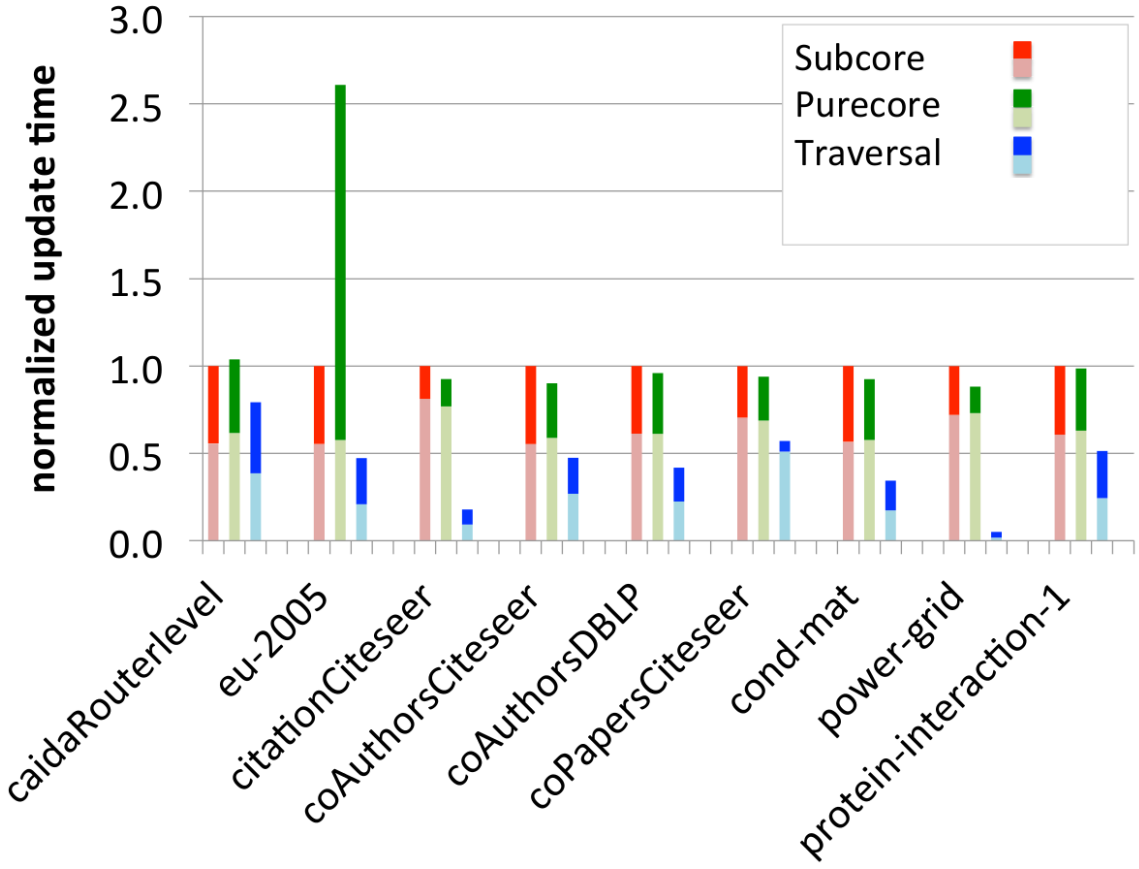
Some Results – Speedup of SUBCORE



Source: Sariyuce et al, 2013

The Dynamic Algorithm

Some Results – Update time



Source: Sariyuce et al, 2013

The Distributed Algorithm

A. Montresor, F. De Pellegrini, D. Miorandi,
"Distributed k-Core Decomposition",
Proceedings PODC, 2011.

The Distributed Algorithm

Main Idea

Try to split work evenly among resources in order to speed up the decomposition process.

Motivation

- faster processing
- massive graphs
- utilize existing engines (e.g., Pregel, GraphLab, Spark GraphX)

The Distributed Algorithm

Computational Models

- **one-to-one**, one computational unit is associated with one node (vertex)
- **one-to-many**, one computational unit is associated with many nodes

We focus on the **one-to-one case**

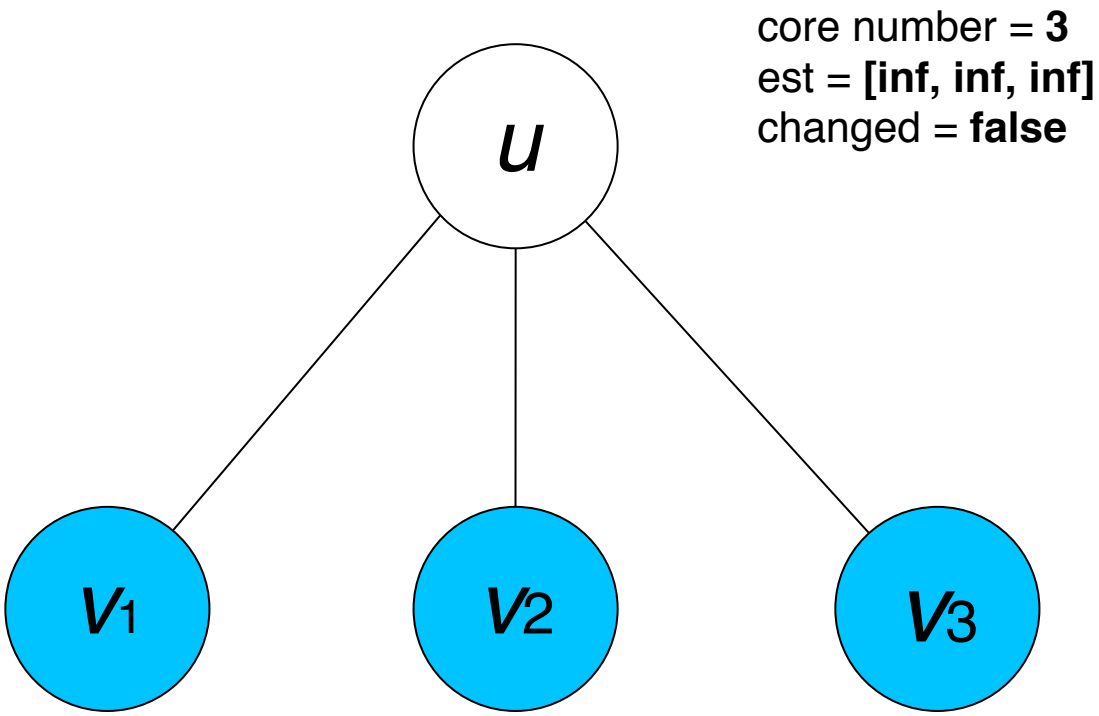
The Distributed Algorithm

Assume that each node *u* is a different processor.

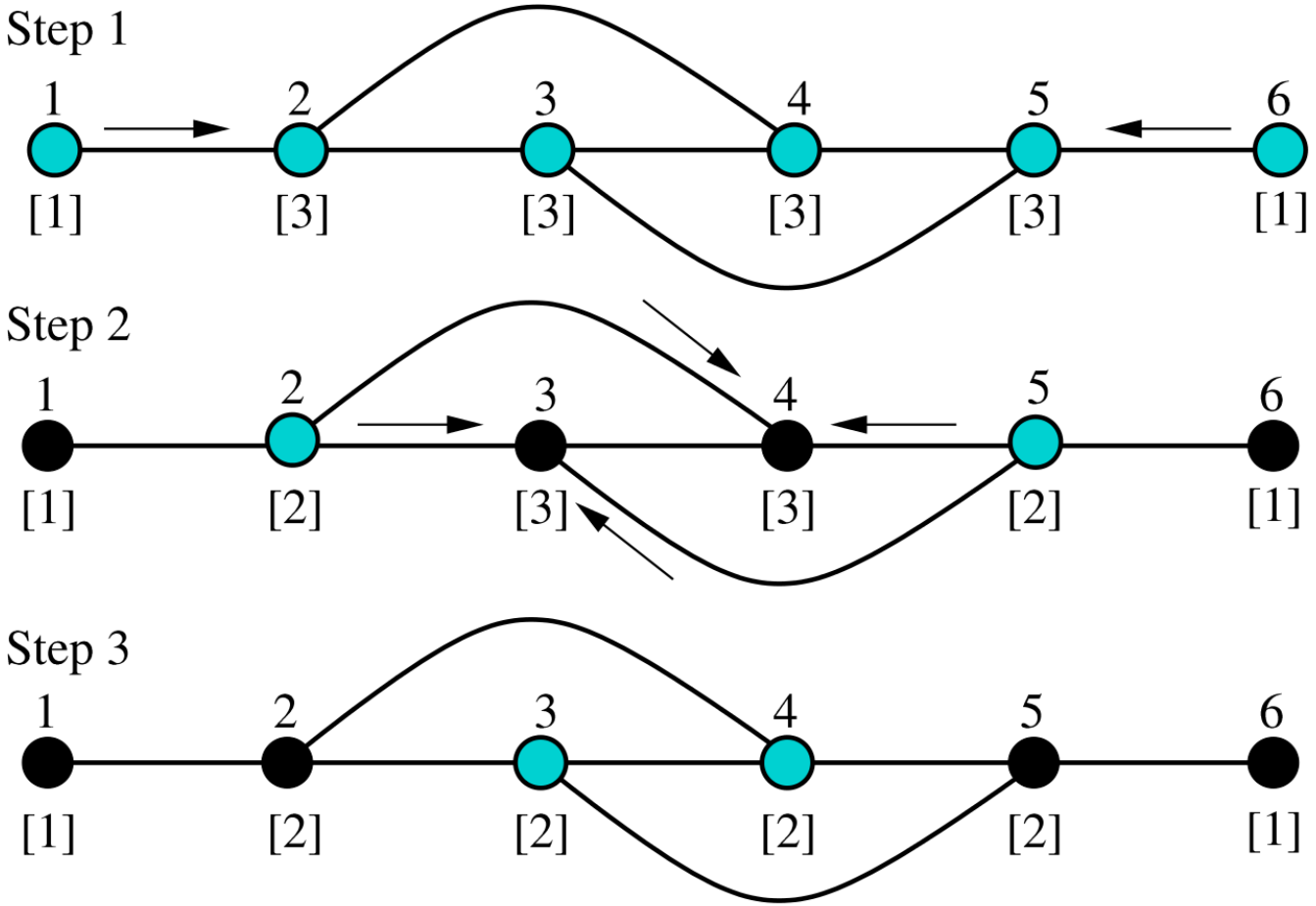
Each node *u* maintains the following information:

- *core* : the **current estimate** for the core number of the vertex
- *est* […] : an **array** containing core estimates for the neighbors of *u*
- *changed :* a **boolean** flag set if *core* has been modified.

The Distributed Algorithm



The Distributed Algorithm



Source: Montresor et al, 2011

The Distributed Algorithm

Some results

Name	$\|V\|$	$\|E\|$	$\oslash$	d_{max}	k_{max}	k_{avg}	t_{avg}	t_{min}	t_{max}	m_{avg}	m_{max}
1) CA-AstroPh	18 772	198 110	14	504	56	12.62	19.55	18	21	47.21	807.05
2) CA-CondMat	23 133	93 497	15	280	25	4.90	15.65	14	17	13.97	410.25
3) p2p-Gnutella31	62 590	147 895	11	95	6	2.52	27.45	25	30	9.30	131.25
4) soc-sign-Slashdot090221	82 145	500 485	11	2 553	54	6.22	25.10	24	26	29.32	3 192.40
5) soc-Slashdot0902	82 173	582 537	12	2 548	56	7.22	21.15	20	22	31.35	3 319.95
6) Amazon0601	403 399	2 443 412	21	2 752	10	7.22	55.65	53	59	24.91	2 900.30
7) web-BerkStan	685 235	6 649 474	669	84 230	201	11.11	306.15	294	322	29.04	86 293.20
8) roadNet-TX	1 379 922	1 921 664	1049	12	3	1.79	98.60	94	103	4.45	19.30
9) wiki-Talk	2 394 390	4 659 569	9	100 029	131	1.96	31.60	30	33	5.89	103 895.35

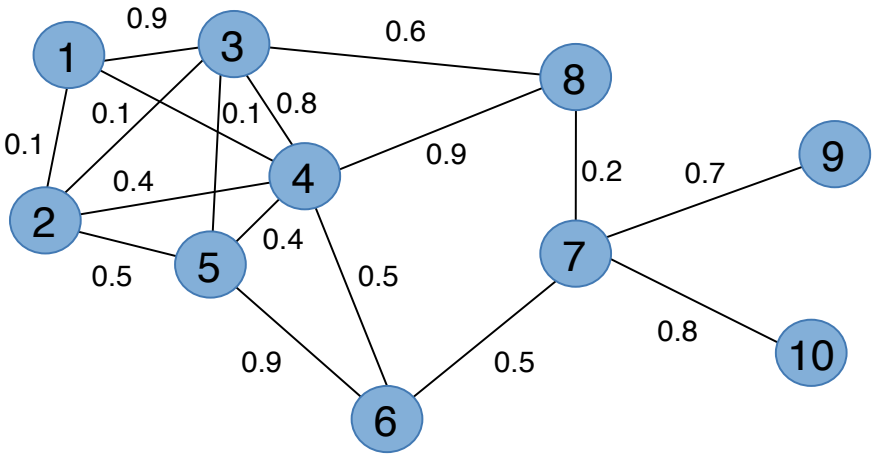
average number of rounds

Core Decomposition of Uncertain Graphs

F. Bonchi, F. Gullo, A. Kaltenbrunner, et al,
"Core Decomposition of Uncertain Graphs",
Proceedings SIGKDD, 2014.

Core Decomposition of Uncertain Graphs

Uncertain Graph or Probabilistic Graph



Each edge e exists with probability $p(e)$

Core Decomposition of Uncertain Graphs

Motivation for Probabilistic Graphs

- edge probabilities may be produced by a link prediction mechanism
- in protein-protein interaction networks denote the probability that two proteins interact in a cell function
- uncertainty may be produced on purpose for privacy reasons
- The probability of an edge may also denote the influence of a person on another (in a social network)
- …

Core Decomposition of Uncertain Graphs

So, where is the challenge?

Core Decomposition of Uncertain Graphs

So, where is the challenge?

Even the simplest tasks may become hard

Example: the problem of testing if two nodes are connected with at least one path (reachability problem) is easily solvable in a deterministic graph, but the problem of computing the probability that two nodes are reachable from each other is **#P-complete** in an uncertain graph

Core Decomposition of Uncertain Graphs

So, where is th

*By definition, a problem is **#P-complete** if and only if it is in **#P**, and every problem in **#P** can be reduced to it by a polynomial-time counting reduction, i.e., a polynomial-time Turing reduction relating the cardinalities of solution sets.*

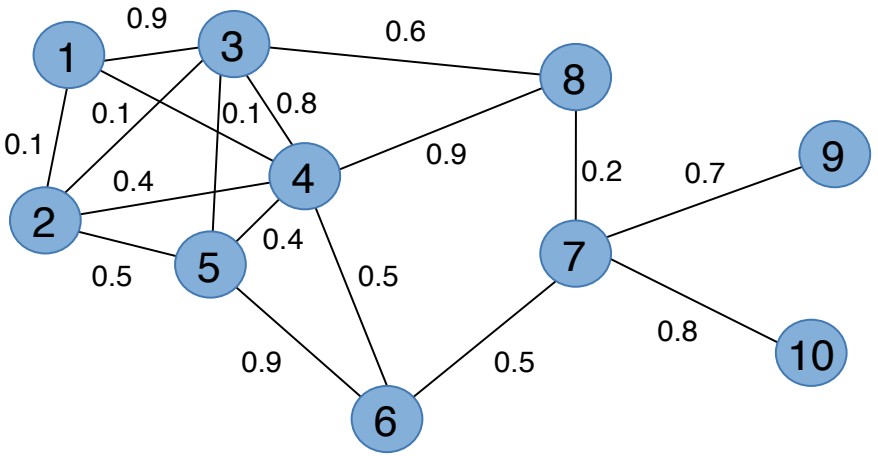
ected

easily solvable

in a dete m of computing the probabilit two nodes are reachable from each other is **#P-complete** in an uncertain graph

Core Decomposition of Uncertain Graphs

Uncertain Graph or Probabilistic Graph



Each edge e exists with probability $p(e)$

Problem: the maximum core may have a very small probability of existence

Core Decomposition of Uncertain Graphs

The concept of probabilistic (*k*,*η*)-cores

Given a probabilistic graph ***G***(*V*,*E*,*p*), the probabilistic (*k*,*η*)-core of ***G*** is a maximal subgraph *H* such that the probability that each node *v* has degree no less than *k* in *H*, is greater than or equal to *η*.

$$\forall u, \Pr[\deg_H(u) \geq k] \geq \eta$$

$$\eta \in [0,1]$$

Core Decomposition of Uncertain Graphs

Problem Definition

Given an uncertain graph $\boldsymbol{G}$ and a probability threshold $\eta \in [0, 1]$, find the (k, η)-core decomposition of $\boldsymbol{G}$, that is the set of all (k, η)-cores of $\boldsymbol{G}$.

Important Property

Given an uncertain graph $\boldsymbol{G}$ and a probability threshold η, the (k, η)-core decomposition of $\boldsymbol{G}$ is unique.

Core Decomposition of Uncertain Graphs

One more definition (η-degree).

Given an uncertain graph $\boldsymbol{G} = (V, E, p)$ and a threshold $\eta \in [0, 1]$, the η-degree of v (η-deg(v)) of a vertex $v \in V$ is defined as

$$\eta\text{-deg}(v) = \max\{k \in [0..d_v] \mid \Pr[\deg(v) \geq k] \geq \eta\}$$

Core Decomposition of Uncertain Graphs

Algorithm (k, η) – cores

Input: An uncertain graph $\mathcal{G} = (V, E, p)$, a threshold $\eta \in [0, 1]$.
Output: An n-dimensional vector $\mathbf{c}$ containing the η-core number of each $v \in V$.

1: compute η-$deg(v)$ for all $v \in V$
2: $\mathbf{c} \leftarrow \emptyset$, $\mathbf{d} \leftarrow \emptyset$, $\mathbf{D} \leftarrow [\emptyset, \ldots, \emptyset]$
3: **for all** $v \in V$ **do**
4: $\quad \mathbf{d}[v] \leftarrow \eta$-$deg(v)$
5: $\quad \mathbf{D}[\eta$-$deg(v)] \leftarrow \mathbf{D}[\eta$-$deg(v)] \cup \{v\}$
6: **end for**
7: **for all** $k = 0, 1, \ldots, n$ **do**
8: $\quad$ **while** $\mathbf{D}[k] \neq \emptyset$ **do**
9: $\quad\quad$ pick and remove a vertex v from $\mathbf{D}[k]$
10: $\quad\quad \mathbf{c}[v] \leftarrow k$
11: $\quad\quad$ **for all** $u : (u, v) \in E$, $\mathbf{d}[u] > k$ **do**
12: $\quad\quad\quad$ recompute η-$deg(u)$
13: $\quad\quad\quad$ move u from $\mathbf{D}[\mathbf{d}[u]]$ to $\mathbf{D}[\eta$-$deg(u)]$
14: $\quad\quad\quad \mathbf{d}[u] \leftarrow \eta$-$deg(u)$
15: $\quad\quad$ **end for**
16: $\quad\quad$ remove v from $\mathcal{G}$
17: $\quad$ **end while**
18: **end for**

Algorithmic Research Directions

- Design of approximation (sublinear) algorithms, trading accuracy for speed
- Distributed implementation in modern engines (e.g., Spark) and scalability tests for massive graphs
- Monitoring the evolution of the cores in time-evolving graphs

Bibliography (algorithms)

V. Batagelj, M. Zaversnik: **"An O(m) Algorithm for Cores Decomposition of Networks"**, ***CoRR*, cs.DS/0310049**, 2002.

J. Cheng, Y. Ke, S. Chu, T. Ozsu, **"Efficient Core Decomposition in Massive Networks"**, ***Proceedings ICDE***, 2011.

A. Montresor, F. De Pellegrini, D. Miorandi, **"Distributed k-Core Decomposition"**, ***Proceedings PODC***, 2011.

M.P. O'Brien, B.D. Sullivan, **"Locally Estimating Core Numbers"**, ***Proceedings ICDM***, 2014.

A.E. Sariyuce, B. Gedik, G. Jacques-Silva, et al, **"Streaming Algorithms for k-Core Decomposition"**, ***Proceedings VLDB End.***, 2013.

F. Bonchi, F. Gullo, A. Kaltenbrunner, et al: **"Core Decomposition of Uncertain Graphs"**, ***Proceedings SIGKDD***, 2014.

Outline

1. Introduction & Motivation
2. Fundamental Concepts and Definitions of Core Decomposition
3. Algorithms
4. **Applications**
5. Open Topics and Future Research Directions

Applications of core decomposition

- Community detection and evaluation
  - Dense subgraph discovery
  - Speed-up community detection algorithms
- Identification of influential spreaders in complex networks
- Dynamics of real-world networks
  - Internet topology
  - Engagement dynamics in social networks
- Network visualization
- Text analytics
- Brain network analysis
- …

Applications of core decomposition

- Community detection and evaluation
  - Dense subgraph discovery
  - Speed-up community detection algorithms
- Identification of influential spreaders in complex networks
- Dynamics of real-world networks
  - Internet topology
  - Engagement dynamics in social networks
- Network visualization
- Text analytics
- Brain network analysis

Community detection

- The notion of **community structure** captures the tendency of nodes to be organized into modules (communities, clusters, groups)
  - Members within a community are **more similar** among each other
- Typically, the communities in graphs (networks) correspond to **densely connected** entities (nodes)
- Set of nodes with **more/better/stronger** connections between its members, than to the rest of the network
- Why this happens?
  - Individuals are typically organized into social groups (e.g., family, associations, profession)
  - Web pages can form groups according to their topic
  - …

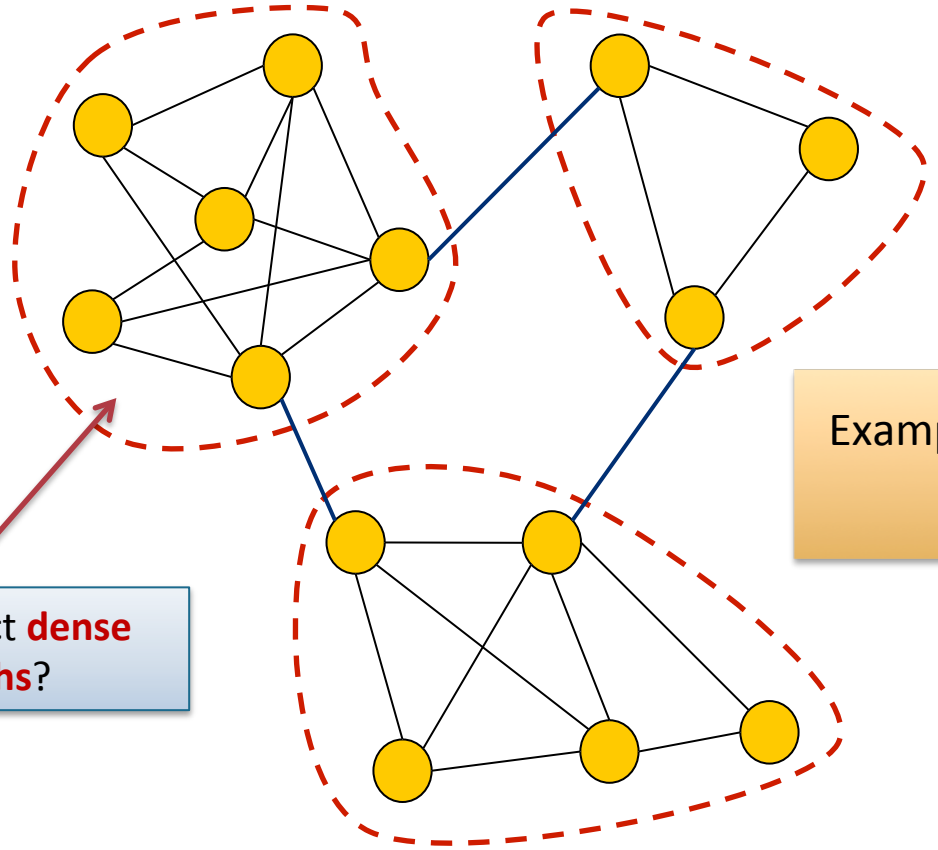
Definition/notion of communities

- How a community in graphs looks like?
- The property of community structure is **difficult** to be defined
  - There is no universal definition of the problem
  - It depends heavily on the application domain and the properties of the graph under consideration
- Most widely used notion/definition of communities is based on the number of edges within a group (density) compared to the number of edges between different groups

A community corresponds to a group of nodes with more **intra-cluster** edges than **inter-clusters** edges

[Newman '03], [Newman and Girvan '04], [Schaeffer '07], [Fortunato '10], [Danon et al. '05], [Coscia et al. 11]

Schematic representation of communities



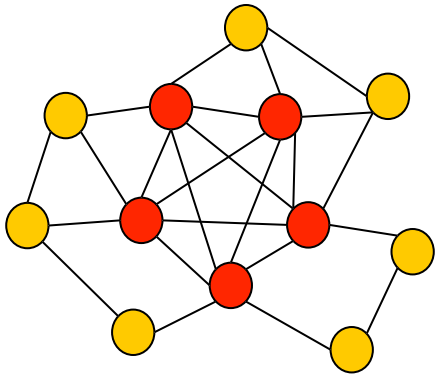
Example graph with three communities

How to detect **dense subgraphs**?

Dense subgraphs

- **Dense subgraphs** are sets of nodes with many edges between them
  - No requirement for **small cuts** (contrary to the community detection problem)
  - **Clique**: complete subgraphs, i.e., all vertices are connected to each other
  - Find the max-size clique is an **NP**-hard problem

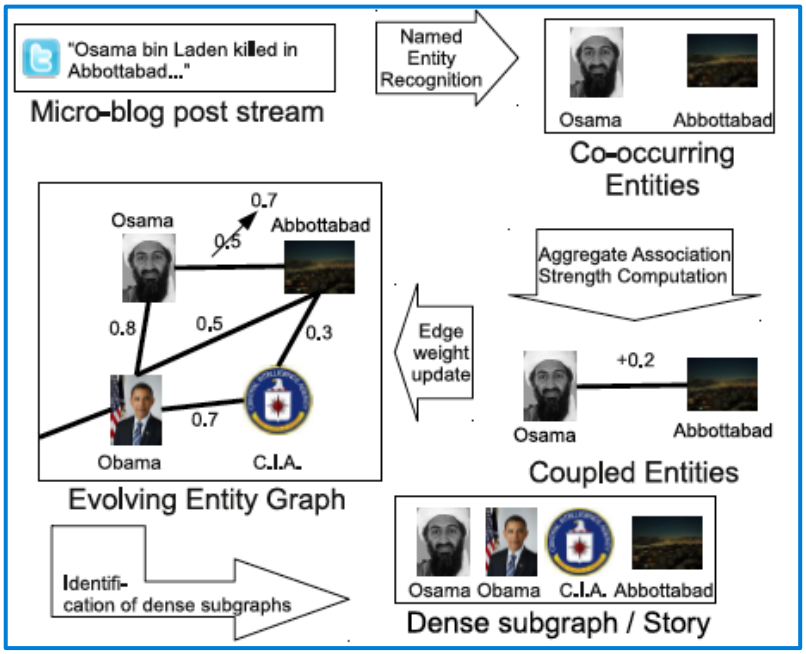
- Primitive operation for various applications



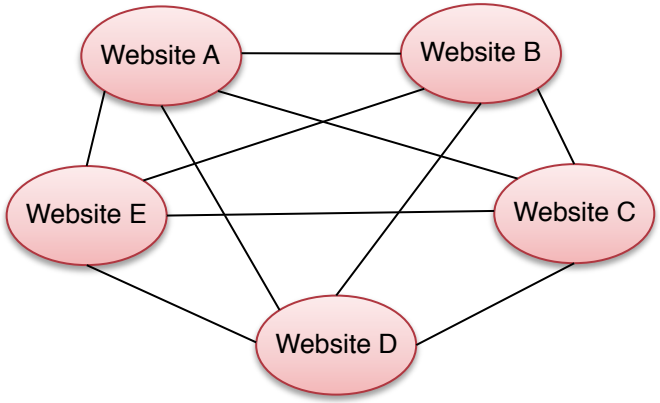
[Håstad '99]

Motivation - Why is the problem important? (1/2)

- Real time story identification [Angel et al. '12]
  - E.g., twitter data stream
  - The **nodes** of the graph correspond to **entities**
  - The **edges** represent **co-occurrence of entities**
  - The **dense subgraphs** capture **news stories**



- Finding **link farms** on the Web [Gibson et al. '04]
  - Group of websites that all link to each other
  - Web spam

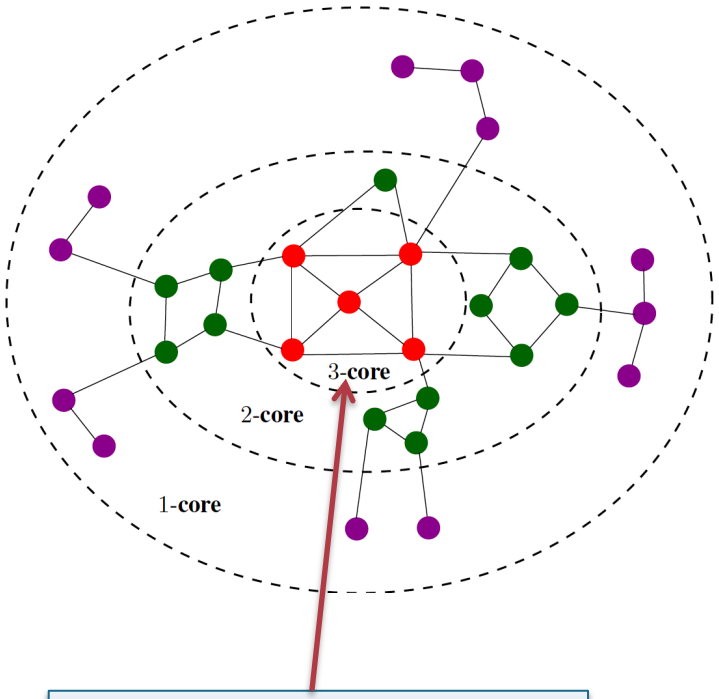


Motivation - Why is the problem important? (2/2)

- Detection of Web communities [Dourisboure et al. '07], [Kumar et al. '99]
- Compression of Web graphs [Karande et al. '09]
- Recommender systems and e-commerce applications
  - Bipartite graph of **users** and **products**
- Detection of frequent itemsets in association rules mining
  - Transaction data can be represented as bipartite graphs
  - **Frequent itemsets** correspond to cliques
- Bioinformatics applications
  - Prediction of protein functions based on the analysis of protein-protein interaction networks [Altaf-Ul-Amin et al. '03]
- …

Maximal k-core subgraphs for dense subgraph discovery

Example:



Core number $c_i = 1$

Core number $c_i = 2$

Core number $c_i = 3$

Graph Degeneracy $\delta^*(G) = 3$

$G_0 = G$

$G_1 = 1$-core of G

$G_2 = 2$-core of G

$G_3 = 3$-core of G

$G_0 \supseteq G_1 \supseteq G_2 \supseteq G_3$

Maximal k-core subgraph

- **Straightforward solution**
1. Apply k-core decomposition
2. Extract the maximal k-core subgraph

No guarantees of how **clique-like** this subgraph is

[Seidman '83]

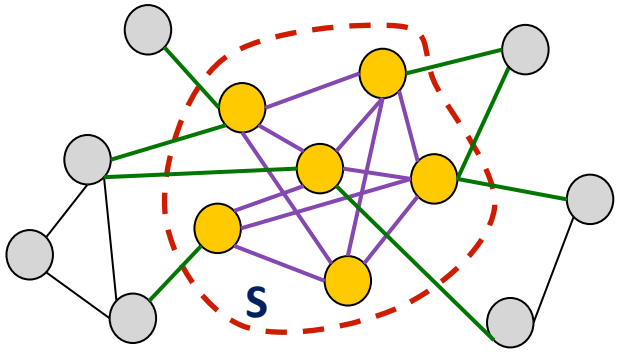
Dense subgraph discovery

- **Densest subgraph** problem:

  Given an undirected graph **G = (V, E)** and a **density measure f: 2^V -> R**

  Find a set of nodes **S** that **maximizes f(S)**

Density of S: $f(S) = \frac{Edges\ within\ S}{|S|}$



- The problem can be solved exactly in **polynomial time** [Goldberg '84]
  - By solving a sequence of maximum flow problems
  - Still, not scalable algorithm for large graphs
- Several variants of the problem: e.g., densest **k**-subgraph (exactly **k** nodes in the extracted subgraph)
  - The complexity increases when adding constraints on the size of the subgraph

Relevant tutorial: [Gionis and Tsourakakis KDD '15]

Variants of the densest k-subgraph

densest k-large-subgraph

- Find a subgraph with at least **k** nodes that has the highest density among all such subgraphs

Algorithm Large-Dense(G, k)

Input: undirected graph **G = (V, E)** and an integer **k**

Output: an induced subgraph of **G** with at least **k** nodes

1. Compute the **k-core decomposition** of the graph: order the nodes according to their core number $v_1 \dots v_n$, where $n = |V|$
   (keep track of the order that nodes are removed by the decomposition)
2. Compute the **density** of each subgraph $H_i = \{v_1 \dots v_i\}$
3. Output the densest subgraph H_i for which $i >= k$

- **Large-Dense(G, k)** is a **(1/3)**-approximation algorithm for the densest **k**-large-subgraph problem
- The running time is **O (m+n)**

[Andersen and Chellapilla '09]

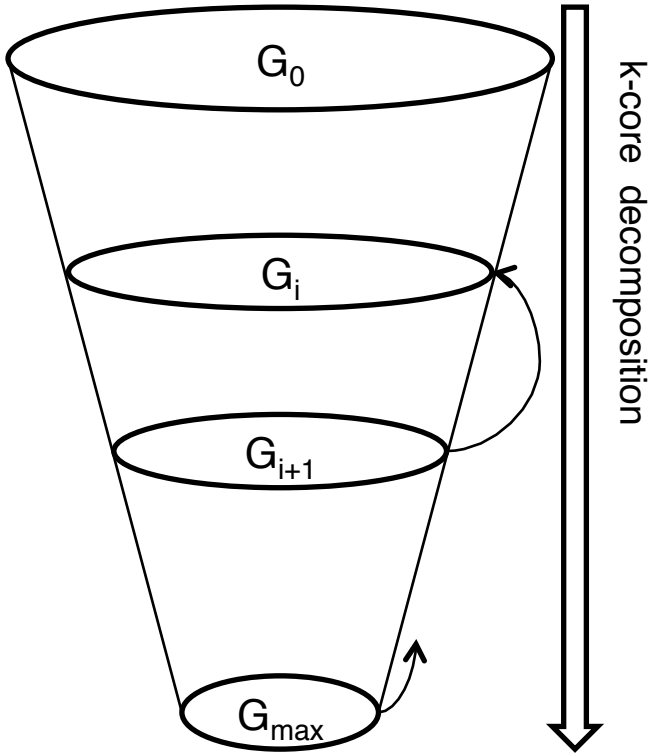
Speeding-up graph clustering

- Assume an expensive algorithm **C** (e.g., Spectral Clustering) as a black box
  - It is less expensive to compute **C** in sections of the data separately, instead of the whole graph

- Utilize the **hierarchical representation** (vertical partition) of the **k**-core decomposition, as incremental input to **C**

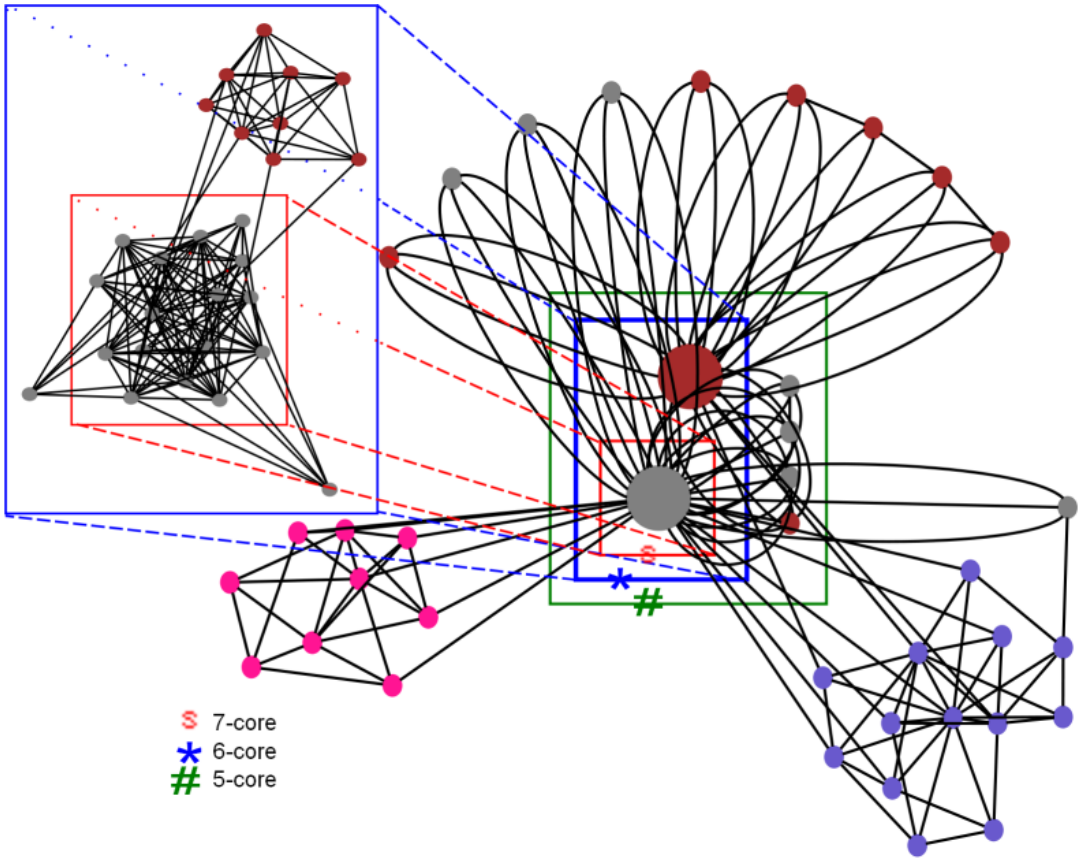
- Starting at the maximal **k**-core, for **i**-core do the following:
  1. Assign with a simple function nodes to existing clusters (from **(i+1)**-core)
  2. Apply **C** to nodes less connected to the existing clusters than nodes of {**(i+1)**-core}-{**i**-core}

CoreCluster framework



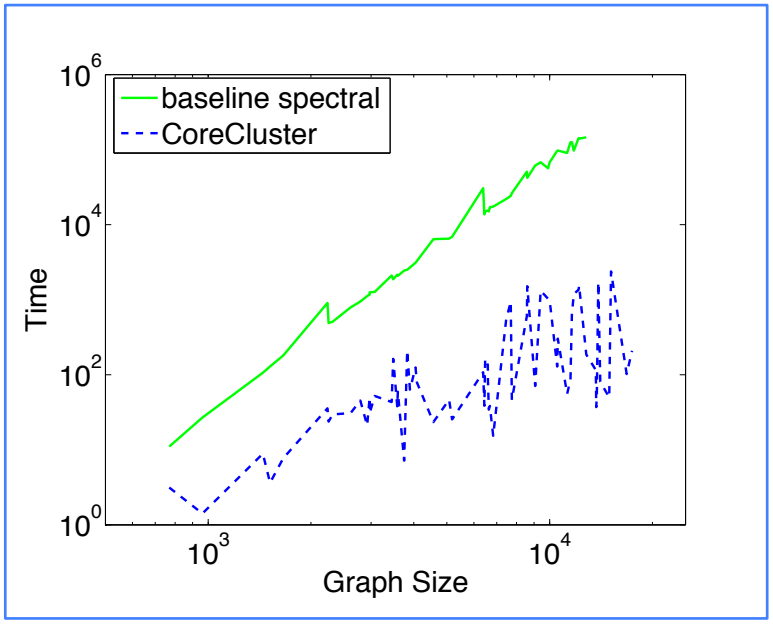
[Giatsidis et al. '14]

CoreCluster framework

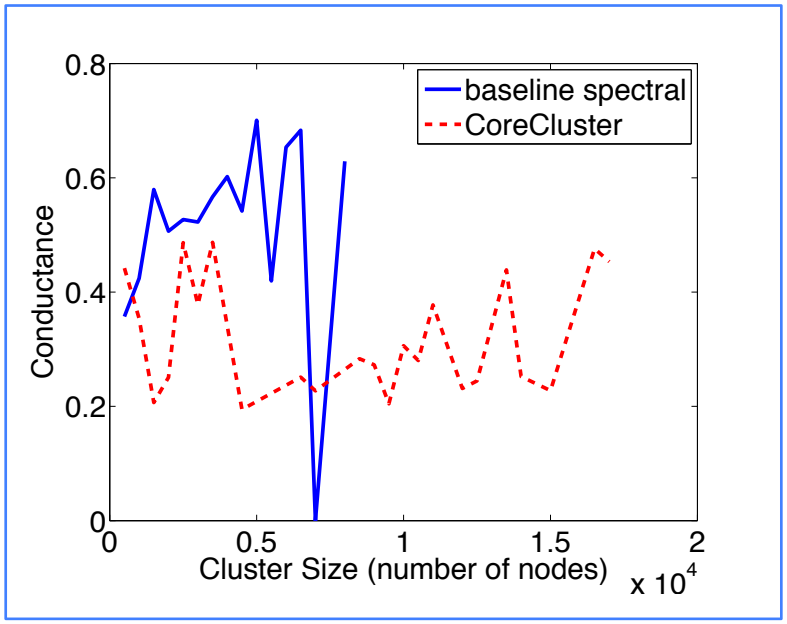


- Core decomposition partitions the graph in a hierarchical nested manner
- We can utilize this structure in graph clustering

Experimental results (Spectral Clustering)



Execution time



Clustering quality

- Significant improvement in execution time
- Clustering quality is retained

[Giatsidis et al. '14]

Applications of core decomposition

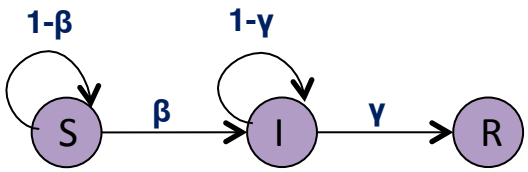
- Community detection and evaluation
  - Dense subgraph discovery
  - Speed-up community detection algorithms
- Identification of influential spreaders in complex networks
- Dynamics of real-world networks
  - Internet topology
  - Engagement dynamics in social networks
- Network visualization
- Text analytics
- Brain network analysis

Identification of influential spreaders

- **Spreading processes** in complex networks
  - Spread of news and ideas
  - Diffusion of influence
  - Disease propagation
  - Viral marketing
  - …
- Identification of **influential spreaders** for understanding and controlling the spreading dynamics
  - Nodes that are able to **diffuse** information to a large part of the network
- **Application in epidemic control:** detect and vaccinate infected individuals with good spreading properties
- **Application in viral marketing:** word-of-mouth effect; target a few initial individuals that can spread the idea/product to a large fraction of the population

Identification of influential nodes: the process

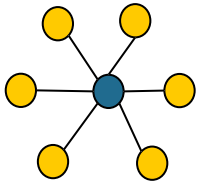
- Typically, a two-step approach [Sei and Makse '13]:
  1. Consider a **topological** or **centrality** criterion of the nodes of the network
  2. Rank the nodes accordingly
  3. The top-ranked nodes are candidates for the most influential ones
  4. Simulate the spreading process over the network to examine the performance of the chosen nodes
- How to simulate the spreading process?
  - Apply models borrowed from **epidemiology** (disease propagation)
  - Susceptible-Infected-Recovered (SIR) model
    1. Set candidate nodes as infected (I state)
    2. An infected node can infect its susceptible neighbors with probability **β**
    3. An infected node can recover (stop being active) with probability **γ**
    4. Count the total number of infected individuals



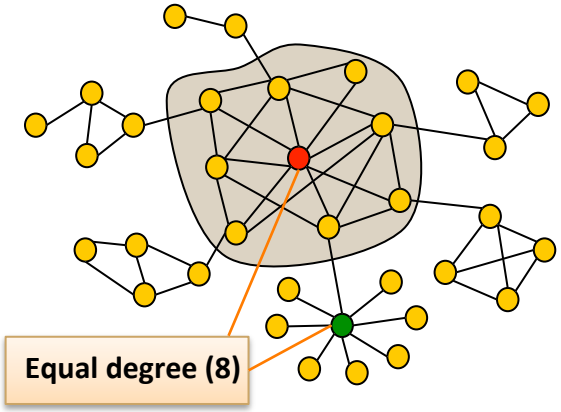
β infection rate
γ recovery rate

Identification of single influential spreaders

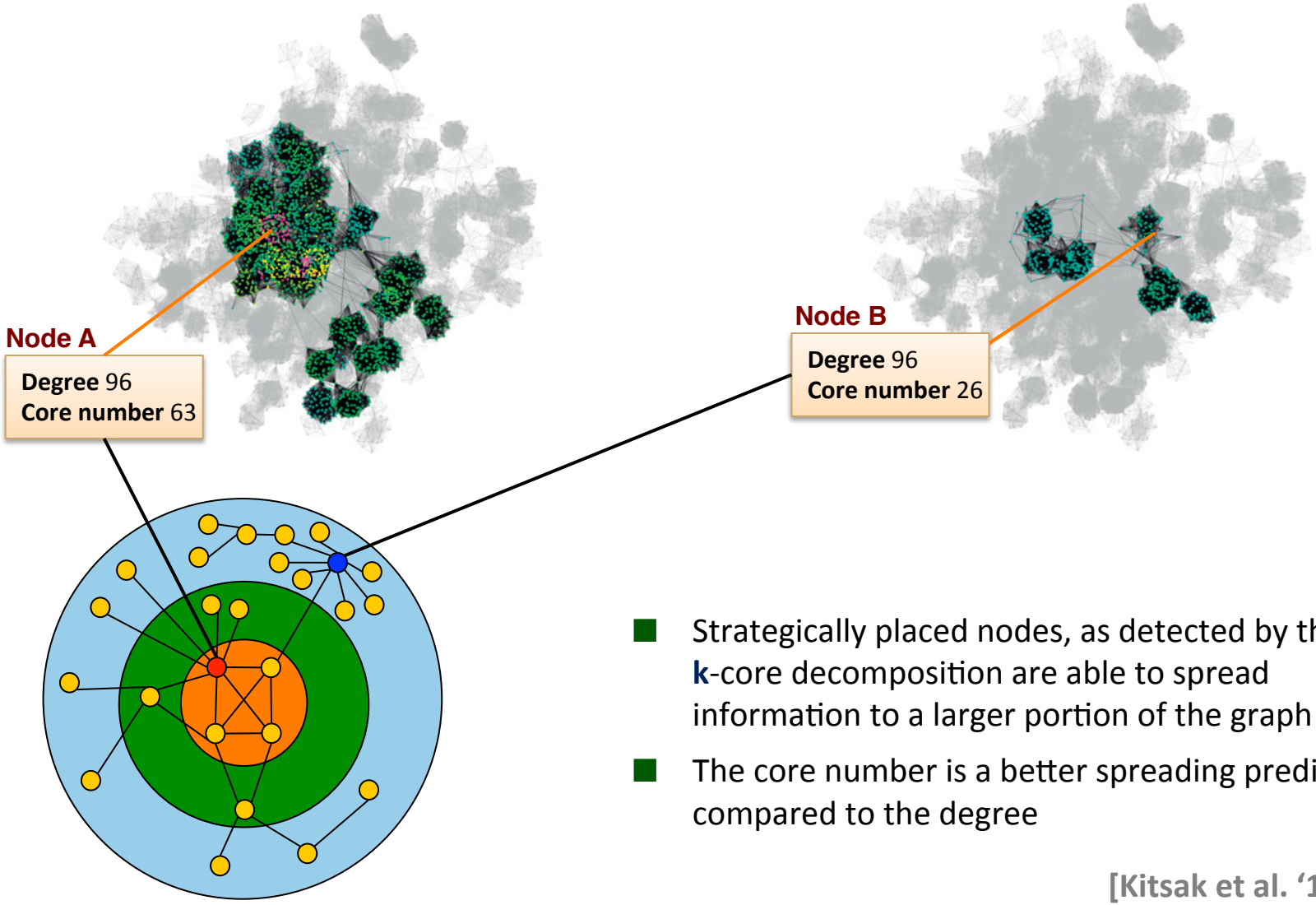
- Straightforward approach: consider **degree centrality**
  - High degree nodes are expected to be good spreaders
  - Results from network science: hub nodes can trigger big cascades affecting the robustness of the system
- However: degree is a **local criterion**
  - Bad instance: **star** subgraph
  - **Core-periphery structure** of real-world networks



Q: How to quantify the core-periphery structure?



The k-core decomposition finds good spreaders



- Strategically placed nodes, as detected by the **k**-core decomposition are able to spread information to a larger portion of the graph
- The core number is a better spreading predictor compared to the degree

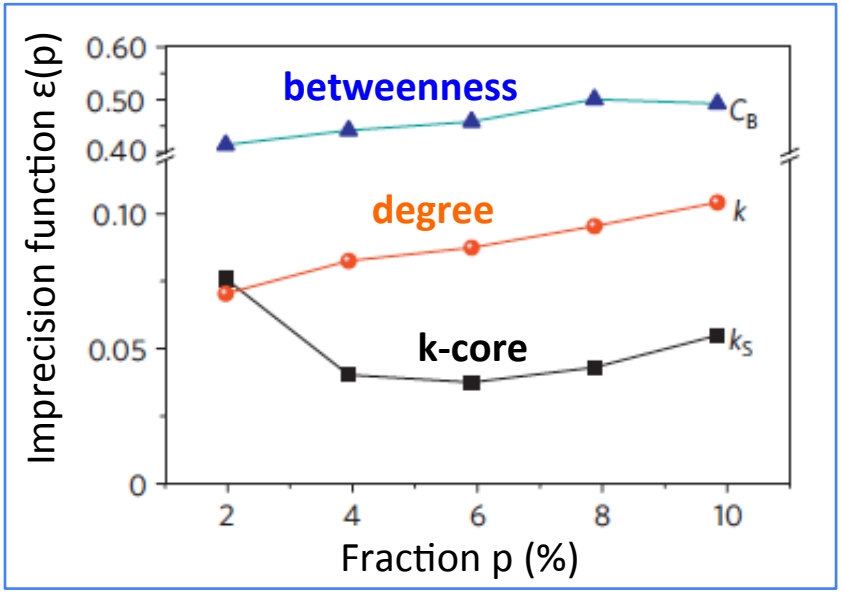
[Kitsak et al. '10]

How close to the optimal spreading?

- Compute the imprecision function **ε(p)** that tests how close is each metric to the optimal spreading
  - For a given fraction **p**:
  1. Find **p%** of the most efficient spreaders and compute the average total spreading $\mathbf{M_{eff}}$
  2. Find **p%** nodes with the highest core number and compute the average total spreading $\mathbf{M_{core}}$
  3. Repeat the same for degree and betweenness centrality

$$\varepsilon(p) = 1 - \frac{M_{core}(p)}{M_{eff}(p)}$$

- The **k**-core is the most accurate spreading predictor
- Degree outperforms the global betweenness centrality



[Kitsak et al. '10]

Extension of the method

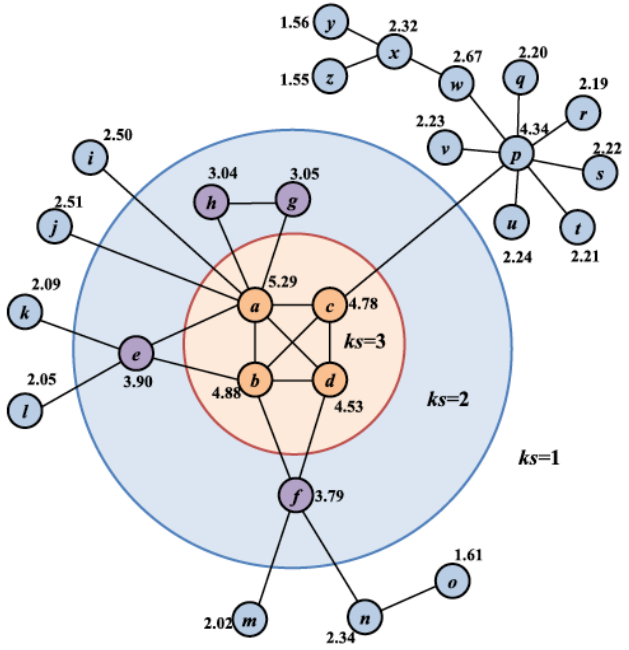
- Observation: the **k**-core decomposition fails to produce a **monotonic ranking** of the nodes
  - Depending on the structure of the network, many nodes will be assigned the same core number …
  - … even if their spreading properties differ to each other

- **Idea:** a node with more neighbors to the core of the network is more powerful
- Define **neighborhood coreness C_{nc}**

$$C_{nc}(v) = \sum_{w \in N(v)} core(w)$$

- New Ranking:
  1. **a**
  2. **b**
  3. **d**
  4. **c**, **p**
  5. **e**, **f**
  6. **…**

[Bae and Kim '14]



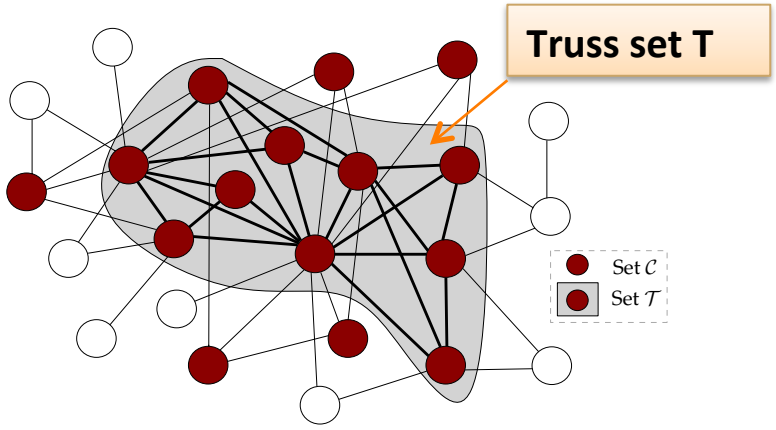
Core number vs. spreading of nodes

K-truss decomposition for detection of spreaders (1/2)

- The **k**-core decomposition often returns a relatively large number of candidate influential spreaders
  - Only a small fraction corresponds to highly influential nodes
- Further **refine** the set of the most influential nodes
- Apply the **K-truss decomposition** [Cohen '08], [Wang and Cheng '12]
  - **Triangle-based** extension of the **k**-core decomposition
  - Each edge of the **K**-truss subgraph participates to at least **K**-2 triangles

Network	Nodes	Edges	|C|	|T|
Email-Enron	33,696	180,811	275	44
Epinions	75,877	405,739	486	61
Wiki-Vote	7,066	100,736	336	50
Email-EuAll	224,832	340,795	292	62

C: set of nodes in the maxima **k**-core subgraph
T: set of nodes in the maximal **K**-truss subgraph



[Malliaros, Rossi and Vazirgiannis '16]

K-truss decomposition for detection of spreaders (2/2)

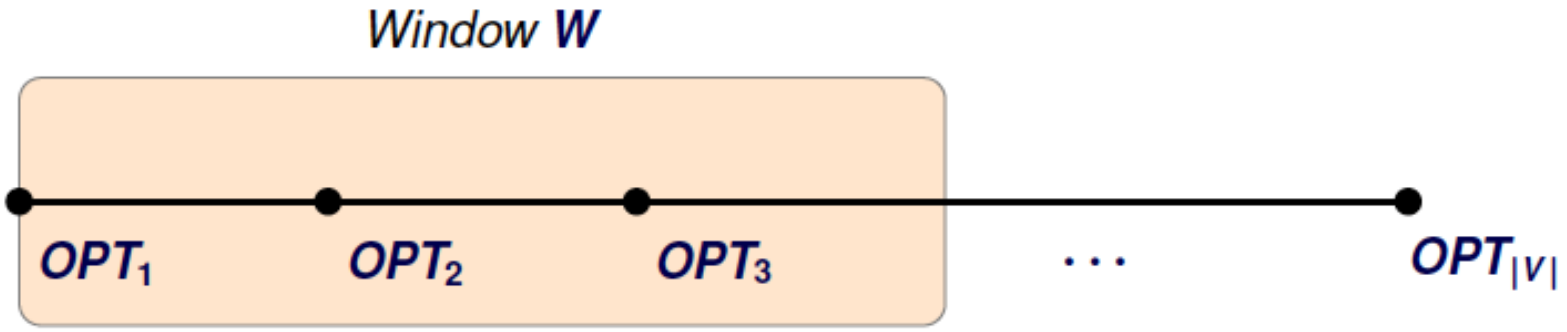
		Time Step							
	Method	2	4	6	8	10	*Final step*	σ	*Max step*
EMAIL-ENRON	truss	8.44	46.66	204.08	418.77	355.84	2,596.52	136.7	33
	core	4.78	31.97	152.55	367.28	364.13	2,465.60	199.6	37
	top degree	6.89	34.13	155.48	360.89	357.08	2,471.67	354.8	36
EPINIONS	truss	4.17	19.70	75.04	204.14	329.08	2,567.69	227.8	37
	core	3.45	14.72	55.27	158.56	280.03	2,325.37	327.2	43
	top degree	4.22	16.03	58.84	166.23	289.49	2,414.99	331.7	47
WIKI-VOTE	truss	2.92	6.92	15.27	28.73	42.46	560.66	114.9	52
	core	1.92	4.78	10.65	20.66	32.40	466.01	104.5	57
	top degree	2.43	5.46	12.05	23.05	35.55	502.88	104.5	62
EMAIL-EUALL	truss	11.62	62.25	240.97	584.87	725.42	5,018.52	487.94	36
	core	9.85	40.82	158.72	433.81	644.76	4,579.84	498.71	38
	top degree	17.96	39.93	144.69	503.18	548.25	4,137.56	1,174.84	39

- **truss**: avg. spreading of the nodes of T
- **core**: avg. spreading of the nodes of C – T
- **top deg**: avg. spreading of the |C – T| top degree nodes

- The **truss** method achieves higher infection rate during the first steps
- The total number of infected nodes at the end of the process is larger, while the epidemic dies out earlier (*Max step*)

[Malliaros, Rossi and Vazirgiannis '16]

Comparison to the optimal spreading (1/2)



1. Rank the nodes according to the total infection size $\mathbf{M_v}$

$$OPT_1 \geq OPT_2 \geq \ldots \geq OPT_{|V|}, \text{ where } OPT_1 = \arg\max_{v \in V} M_v$$

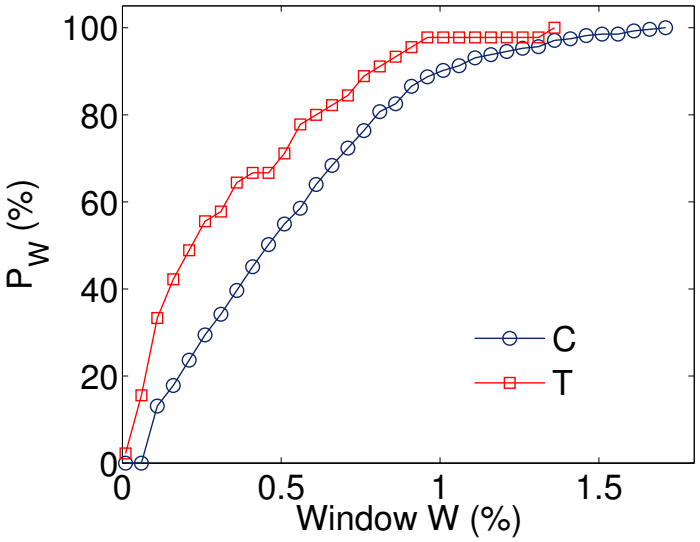
2. Consider window **W** over the ranked nodes

$$P_W^{\mathcal{T}} = \frac{|T_W|/|\mathcal{T}|}{|W|/|V|}$$

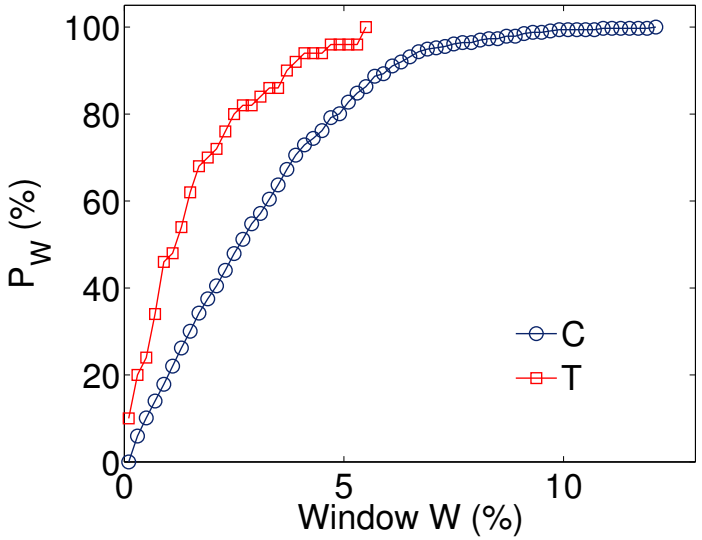
[Malliaros, Rossi and Vazirgiannis '16]

Comparison to the optimal spreading (2/2)

Email-Enron



Wiki-Vote



1. P_w^{truss} reaches the maximum value (i.e., 100%) relatively early and for small window sizes, compared to P_w^{core}
2. The nodes selected by the K-truss decomposition are better distributed among the optimal spreaders

[Malliaros, Rossi and Vazirgiannis '16]

Applications of core decomposition

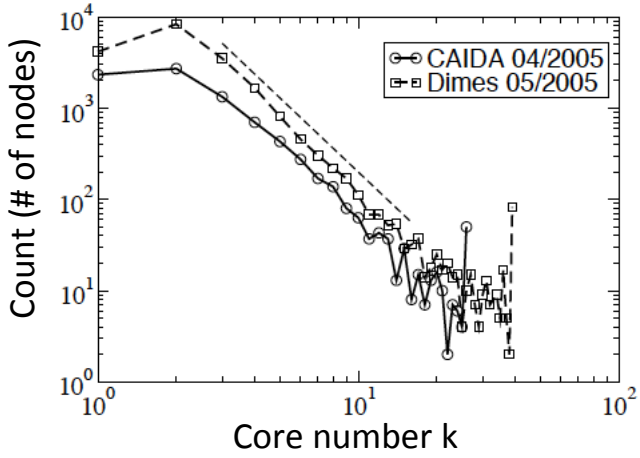
- Community detection and evaluation
  - Dense subgraph discovery
  - Speed-up community detection algorithms
- Identification of influential spreaders in complex networks
- Dynamics of real-world networks
  - Internet topology
  - Engagement dynamics in social networks
- Network visualization
- Text analytics
- Brain network analysis

Analysis of the Internet graph

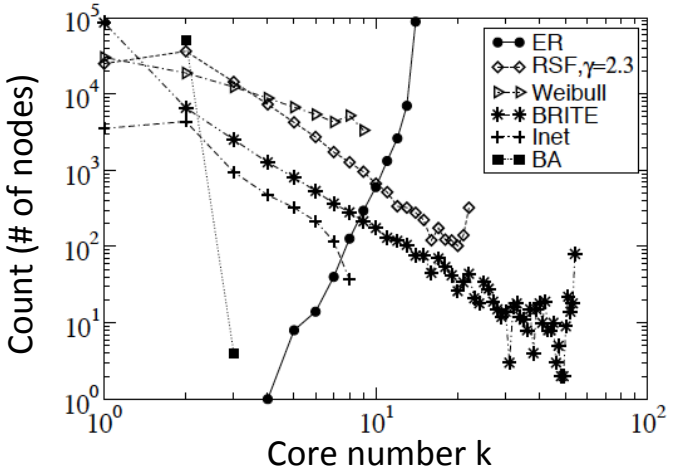
- **k**-core decomposition as a tool to analyze the **structural properties** of the Internet graph
  - Property of connectivity
  - Related to the **robustness** of the network under failures and attacks
  - Important for **routing**, i.e., how to find a path between two nodes

1. Central nodes are **more strongly connected**
   - Larger number of distinct paths between nodes
   - More robust routing
2. Validate **graph generation models** (e.g., Preferential Attachment)
   - The **k**-core structure of the Internet graph obeys properties and rules
   - To what extend network generation models mimic these properties?

Hierarchical structure of the Internet graph



Power law k-core number distribution of the Internet graphs



The graph generation models produce various shapes for the k-core number distribution

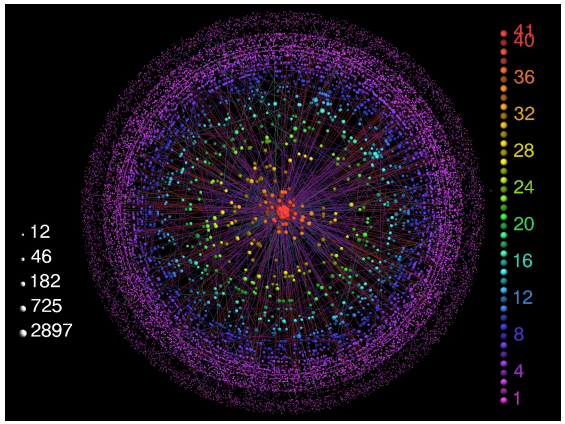
ER: Erdos-Renyi; **BA:** Barabasi-Albert;
RSF: scale-free; **Weibull:** weibull distribution
BRITE, Inet: internet topology generators

Network	Nodes	Edges	Avg. deg	d_{max}	k_{max}
CAIDA, 2005/04	8,542	25,492	5.97	1171	26
DIMES, 2005/05	75,877	405,739	6.04	2800	39

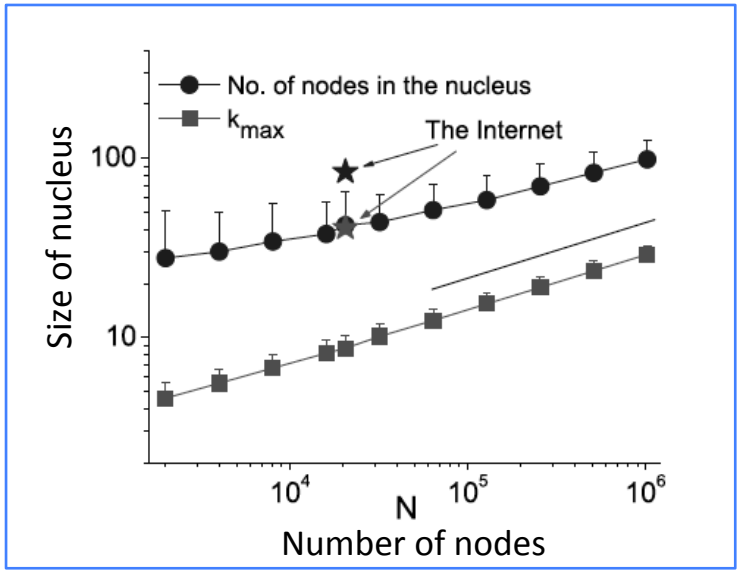
- Interesting properties of the **k**-core structure of the internet graph
- Additional tool for network characterization

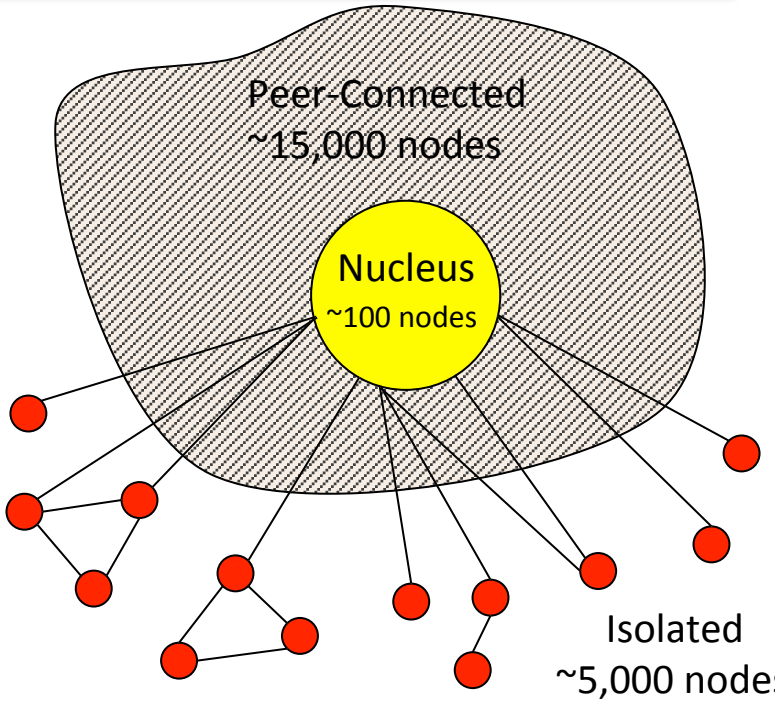
[Alvarez-Hamelin et al. '08]

Medusa: a model of Internet topology based on k-core



Visualization of a snapshot of Internet based on the k-core number





- The **k**-core decomposition provides an intuitive model about the structure of the Internet topology
- Insights about the growth of the Internet

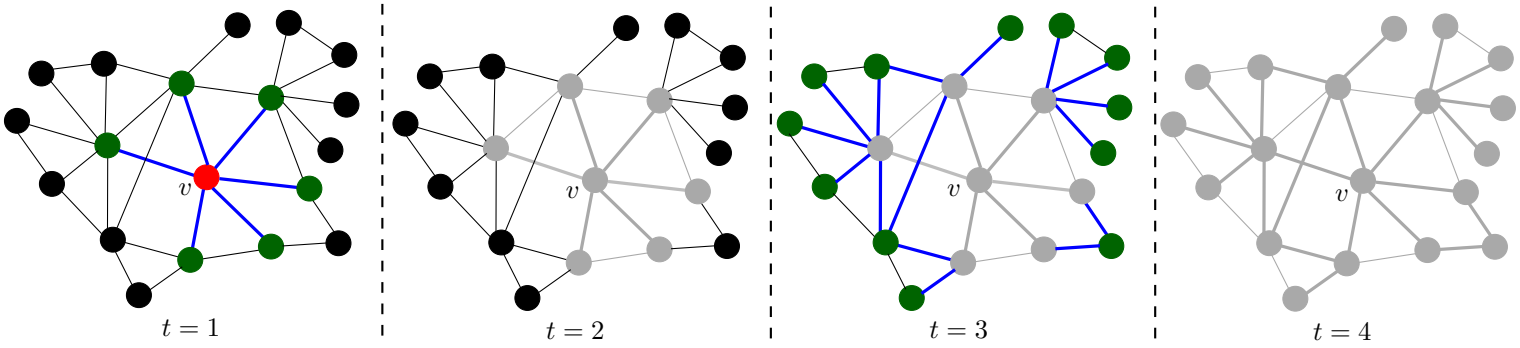
[Carmi et al. '07]

Modeling engagement dynamics in social graphs

- **Question:** Given a large social graph, how can we model and quantify the engagement dynamics?
- User **engagement:** refers to the extend that an individual is encouraged to participate in the activities of a community
  - Closely related to the property of node **departure dynamics**
  - Similar to the decision of becoming member of a community, an individual (node) may also decide to **leave** the graph (e.g., become inactive)
- Each node that participates in a social activity, derives a **benefit**
  1. The benefit emanates from his neighborhood (as captured by the node degree) and …
  2. The degree of interaction among its neighbors **[Ugander et al., PNAS '12]**
     - If ones friends tend to highly interact among each other → the benefit of remaining engaged could potentially be increased

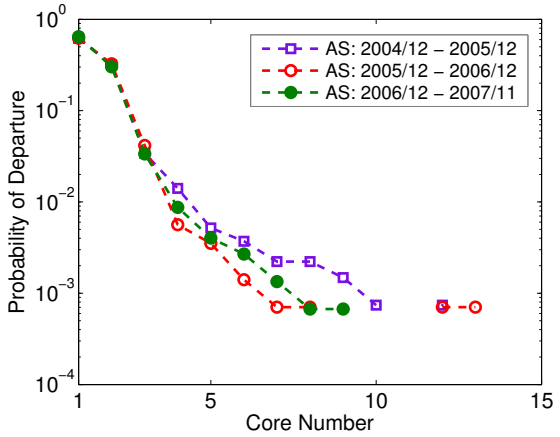
Modeling engagement dynamics: the motivation

- Suppose now that a user (node) decides to **depart** (leave the graph)
  - This will cause direct effects in his/her neighborhood → some of his/her friends may also decide to depart
  - A departure can become an **epidemic** (contagion), forming a **cascade** of individual departures

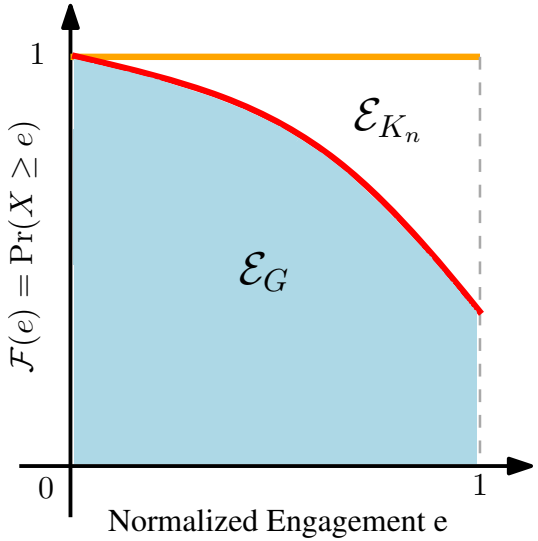


Engagement measures based on the k-core decomposition

- **Node engagement:** the engagement level e_i of node **i** is defined to be its **core number** c_i
  - A degeneracy-based approach
  - Higher core number → better engagement
  - Based on **game theoretic** concepts [Manhadi and Jihari, '09], [Bhawalkar et al. '12] [Harkings, '13]



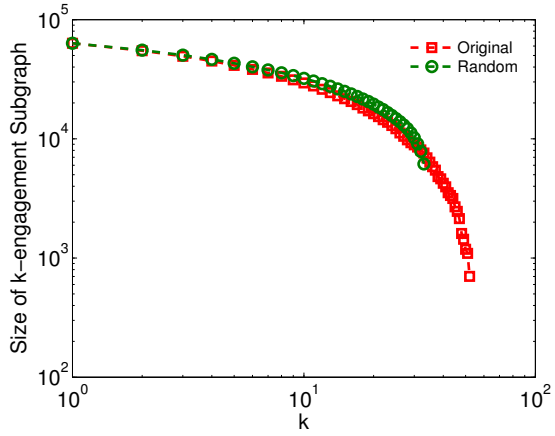
CAIDA dataset



- **Graph engagement:** the total engagement level E_G of the graph based on the aggregated behavior of individual nodes
  - **Area under curve** (light blue region)
  - Values in range **[0, 1]**
  - Higher values → higher total engagement
  - E_{K_n}: complete graph → best engagement properties

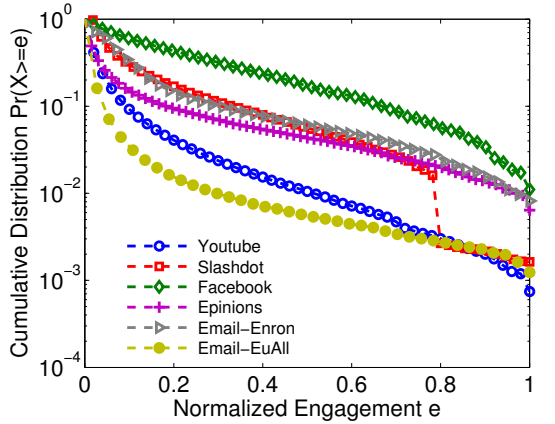
[Malliaros and Vazirgiannis '13], [Garcia et al. '13]

Experimental results on real graphs



Facebook graph

- Cumulative distribution of the number of nodes with engagement level **k** (**red curve**)
- Almost **skewed** distribution → small portion of nodes with high engagement level
- Deviation from random graphs (**green curve**)



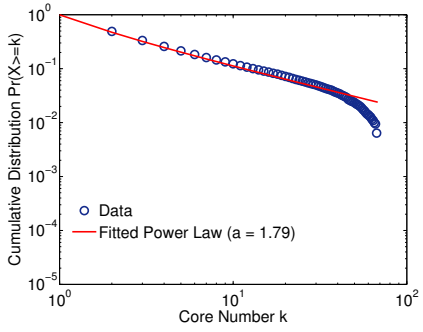
Social graphs

- Facebook graph (**green curve**) has the maximum total engagement index (area under curve)
- A relatively high fraction of nodes has high engagement index **e**

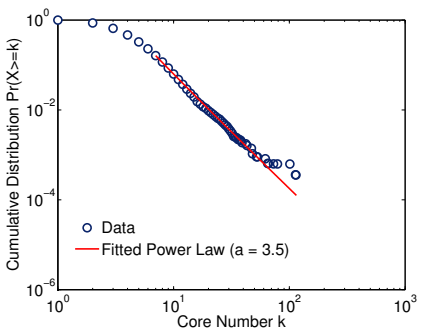
[Malliaros and Vazirgiannis '13]

Network vulnerability under node departures (1/2)

- Nodes with low engagement are more likely to depart from the graph
- This may cause a **disengagement cascading effect** of departures
- **Cascading Departure (CasD)** model
  - **k**-core decomposition-based model to capture the cascading effect due to the departure of a node
  - Remove a node if its core number has changed (decreased)
- Which nodes are more significant?
- Two strategies for selecting the node that will depart first
  1. **Random** departure – more probable to occur in practice
  2. **Targeted** departure of nodes with high engagement (core number)



Epinions graph

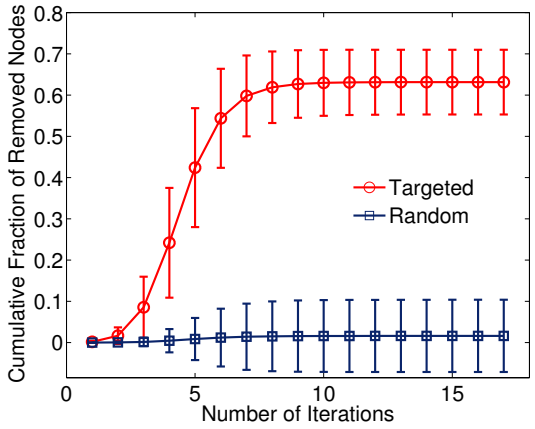


DBLP graph

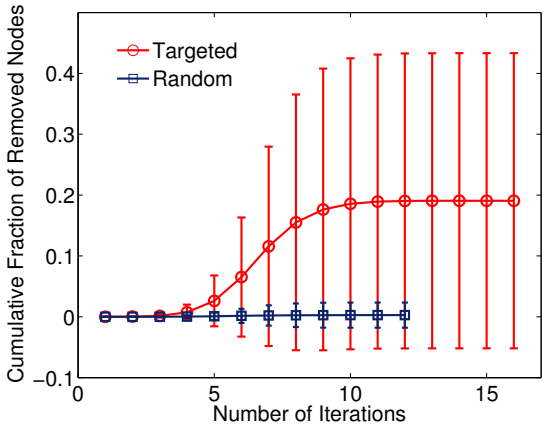
- **Heavy-tailed** core number distribution

[Malliaros and Vazirgiannis '15]

Network vulnerability under node departures (2/2)



Epinions graph



DBLP graph

- Social networks are extremely **robust** against cascades triggered by **departures of randomly selected** nodes …
- … but **vulnerable** under cascades caused by **targeted departures** of high core number nodes

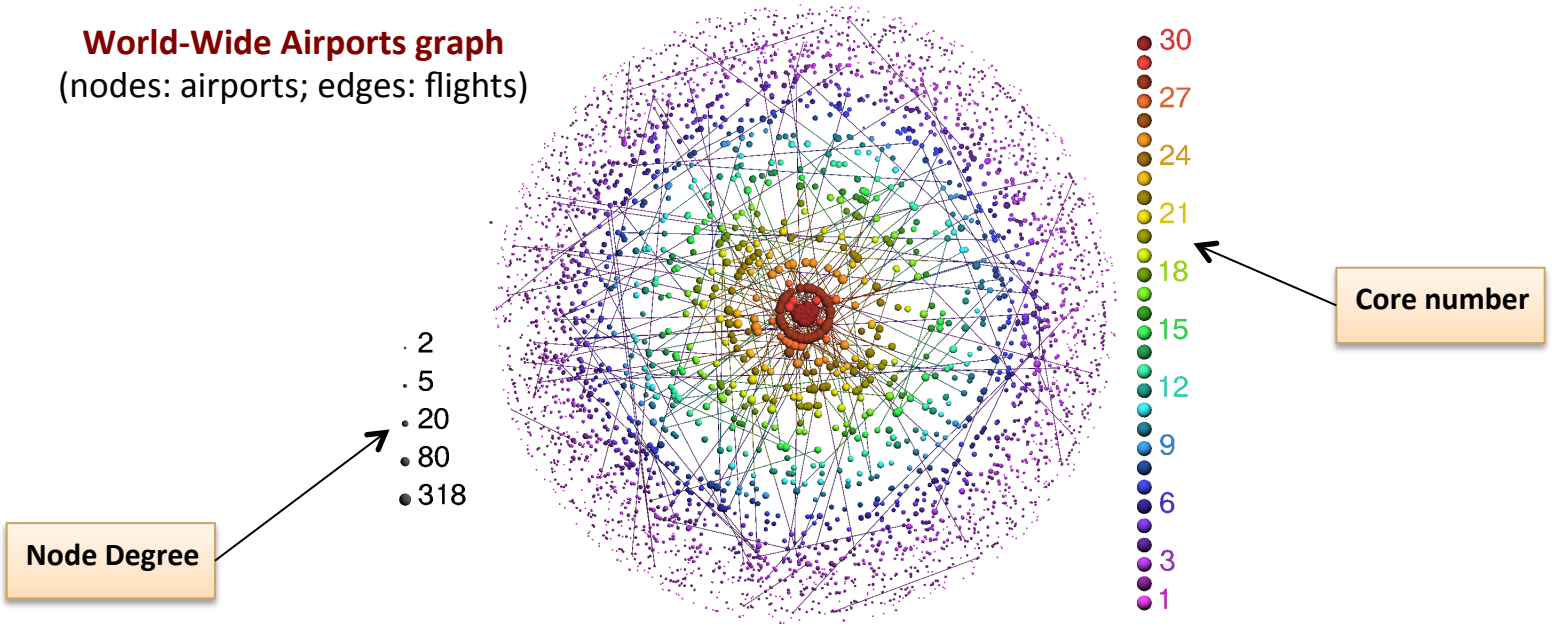
[Malliaros and Vazirgiannis '15]

Applications of core decomposition

- Community detection and evaluation
  - Dense subgraph discovery
  - Speed-up community detection algorithms
- Identification of influential spreaders in complex networks
- Dynamics of real-world networks
  - Internet topology
  - Engagement dynamics in social networks
- Network visualization
- Text analytics
- Brain network analysis

Tool for network visualization

World-Wide Airports graph
(nodes: airports; edges: flights)



Publicly available at: **http://lanet-vi.soic.indiana.edu/**

[Alvarez-Hamelin et al. '06]

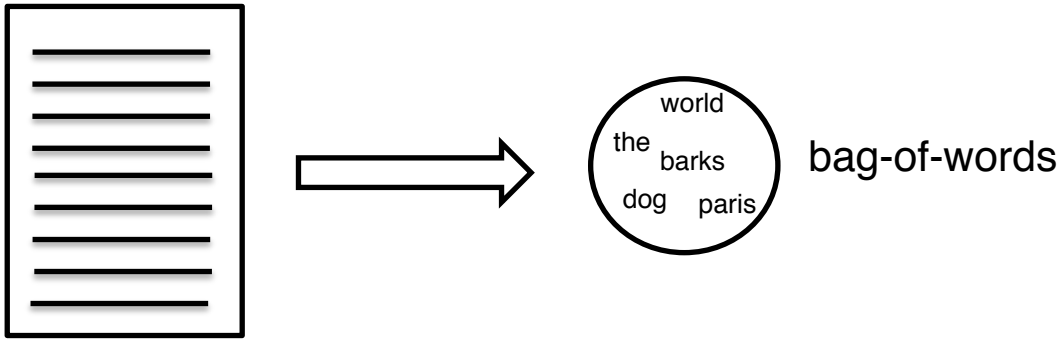
Applications of core decomposition

- Community detection and evaluation
  - Dense subgraph discovery
  - Speed-up community detection algorithms
- Identification of influential spreaders in complex networks
- Dynamics of real-world networks
  - Internet topology
  - Engagement dynamics in social networks
- Network visualization
- Text analytics
- Brain network analysis

Text Mining – Terminology

- Text mining on a collection of documents:
  - The collection is the data set
  - The documents are the data points

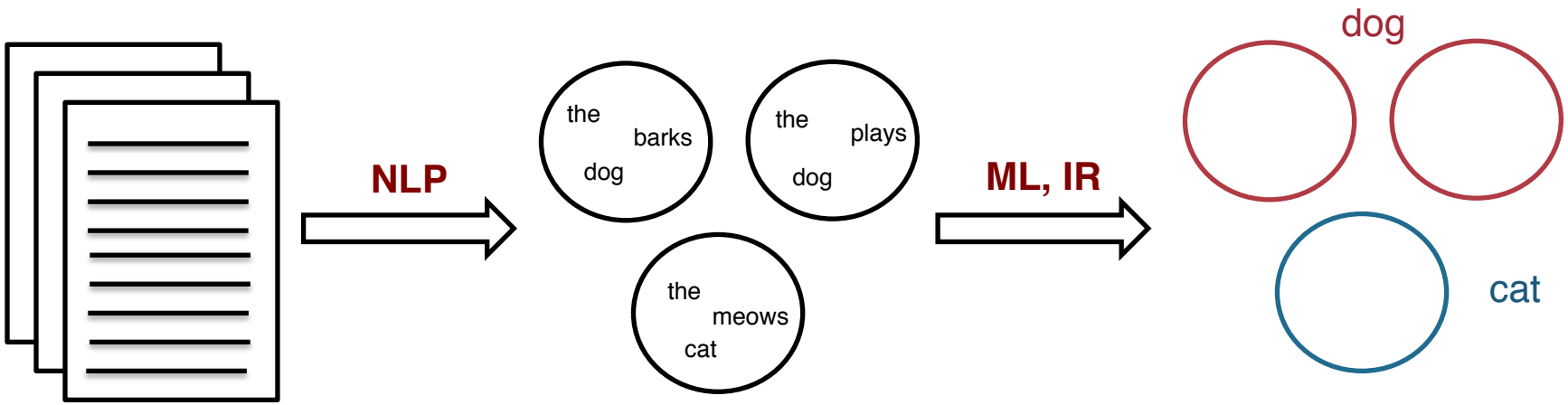
- Since text is unstructured, a document is usually converted in a common representation



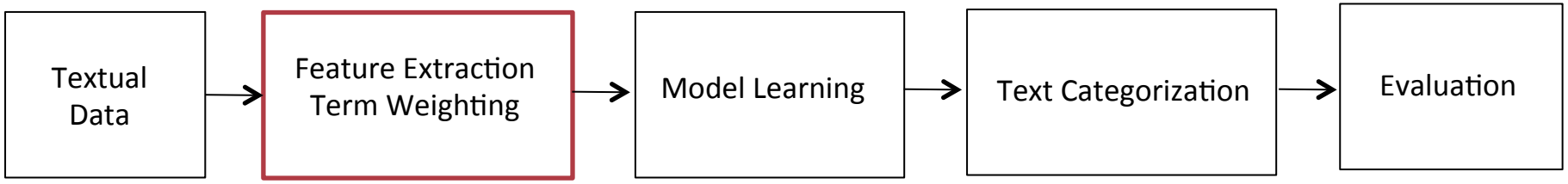
[Vazirgiannis '15]

Text Mining – Pipeline

- **Input:** raw text
- **Output:** application-dependent information
- In-between:
  1) document representation – **bag of words**
  2) feature extraction



Example: text categorization



Applications:

- Opinion mining (sentiment analysis)
- Email spam classification
- Web-pages classification
- …

Bag of Words - issues

Example text

information retrieval is the activity of obtaining

information resources relevant to an information need

from a collection of information resources

Bag of words: [(activity,1), (collection,1)
(information,4), (relevant,1),
(resources, 2), (retrieval, 1), …]

- Term independence assumption
- Term frequency weighting

Graph-based document representation (1/2)

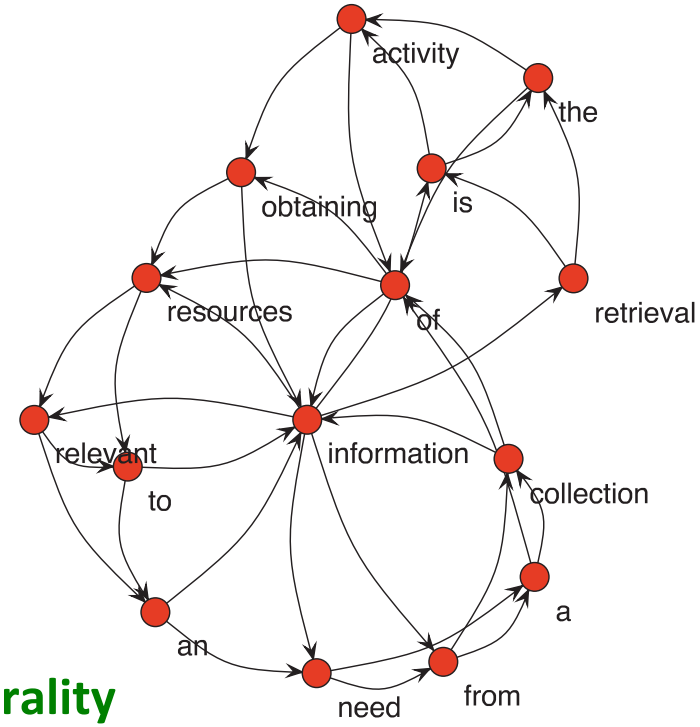
- Challenge **term independence** and **term frequency weighting** assumptions taking into account **word dependence**, **order** and **distance**
- Employ a graph-based document representation capturing the above
- Graphs successfully used in IR to encompass relations and propose meaningful weights (e.g., PageRank)

Graph-based document representation (2/2)

information retrieval is the activity of obtaining

information resources relevant to an information need

from a collection of information resources



Bag of words: [(activity,1), (collection,1)
(information,4), (relevant,1),
(resources, 2), (retrieval, 1), …]

Idea: Replace **term frequency** with **node centrality**

Captures: frequency, order and distance, …

Single document keyword extraction

Keywords are used everywhere:

- looking up information on the Web (e. g., via a search engine bar)
- Finding similar posts on a blog (e. g., tag cloud)
- For ads matching (e. g., AdWords' keyword planner)
- For research paper indexing and retrieval (e. g., SpringerLink)
- For research paper reviewer assignment

Applications are numerous:

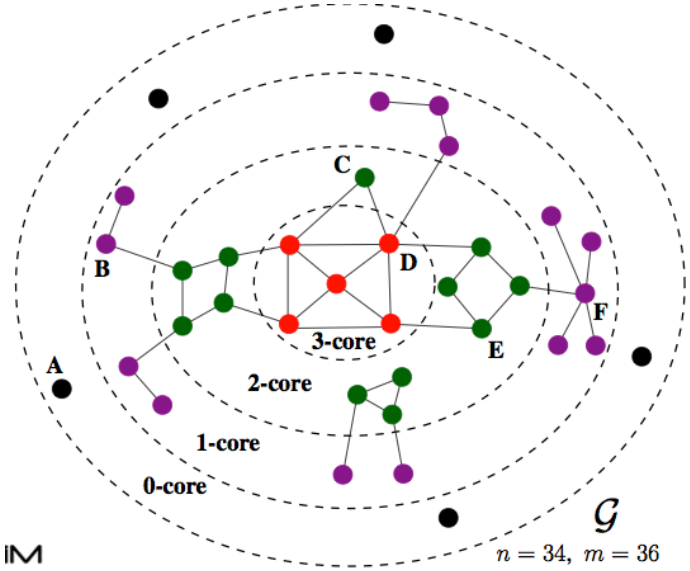
- **Summarization** (to get a gist of the content of a document)
- **Information filtering** (to select specific documents of interest)
- **Indexing** (to answer keyword-based queries)
- **Query expansion** (using additional keywords from top results)

Graph-based keyword extraction (1/2)

Existing graph-based keyword extractors:

- **PageRank** [Mihalcea and Tarau '04]
- **HITS** [Litvak and Last '08]
- Assign a **centrality** based influence score to a node
- Top ranked ones will correspond to the most representative

Idea: retain the **k-core subgraph** of the graph to extract the nodes based on their centrality and cohesiveness

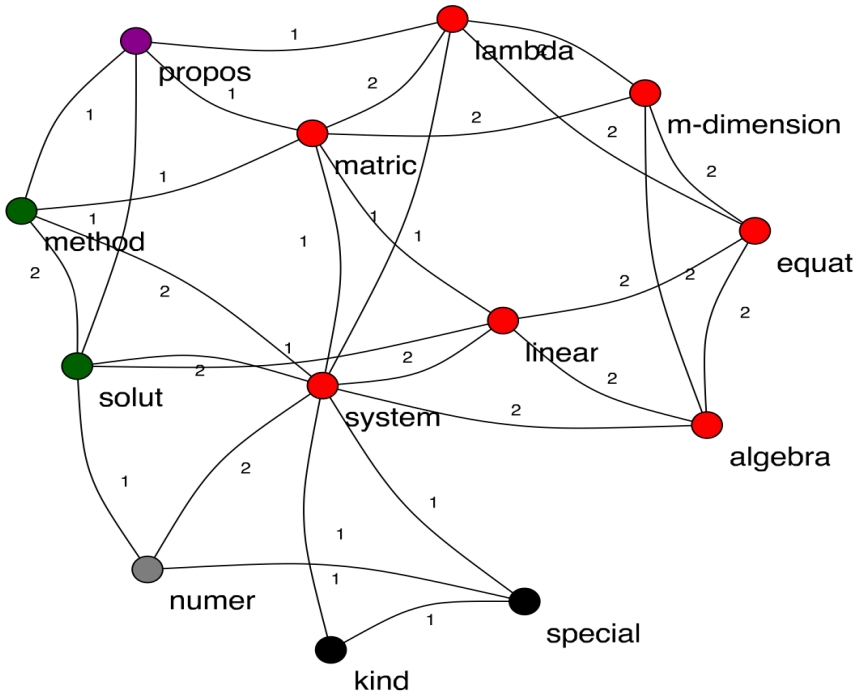


k-core decomposition of the graph

Graph-based keyword extraction (2/2)

A method for solution of systems of linear algebraic equations with m-dimensional lambda matrices.
A system of linear algebraic equations with m-dimensional lambda matrices is considered. The proposed method of searching for the solution of this system lies in reducing it to a numerical system of a special kind.

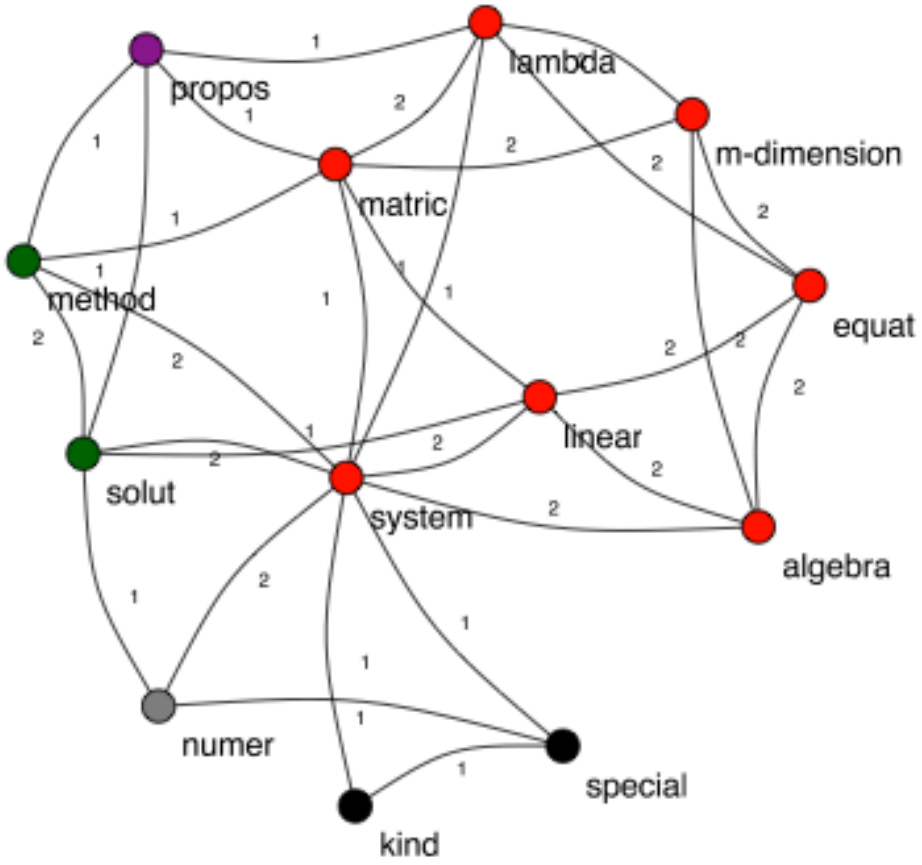
- Single-document keyword extraction
  - Select the most cohesive sets of words in the graph as keywords
  - Use k-core decomposition to extract the main core of the graph
  - Weighted edges



Keywords manually assigned by human annotators
linear algebra equat; numer system; m-dimension lambda matric

[Rousseau and Vazirgiannis '15]

PageRank vs. k-core



Keywords manually assigned by human annotators
linear algebra equat; numer system; m-dimension lambda matric

WK-core		PageRank	
system	6	**system**	1.93
matric	6	**matric**	1.27
lambda	6	solut	1.10
linear	6	**lambda**	1.08
equat	6	**linear**	1.08
algebra	6	**equat**	0.90
m-dim...	6	**algebra**	0.90
method	5	**m-dim...**	0.90
solut	5	propos	0.89
propos	4	method	0.88
numer	3	special	0.78
specia	2	**numer**	0.74
kind	2	kind	0.55

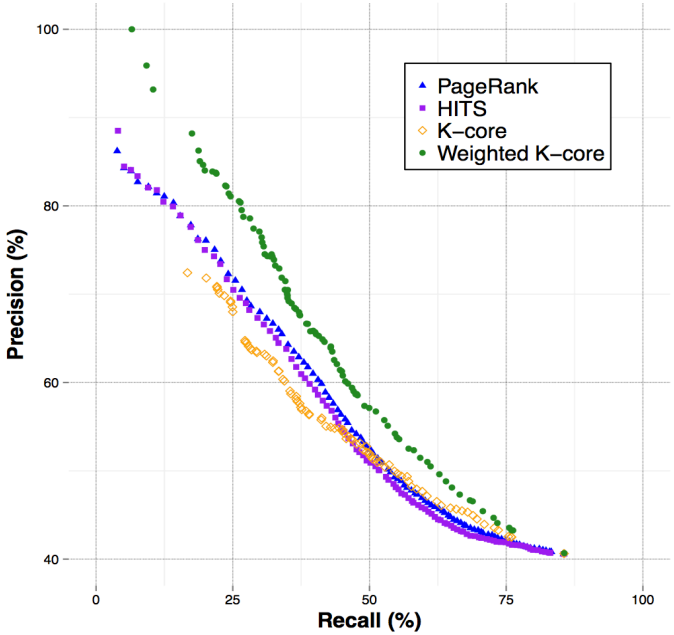
How many keywords?

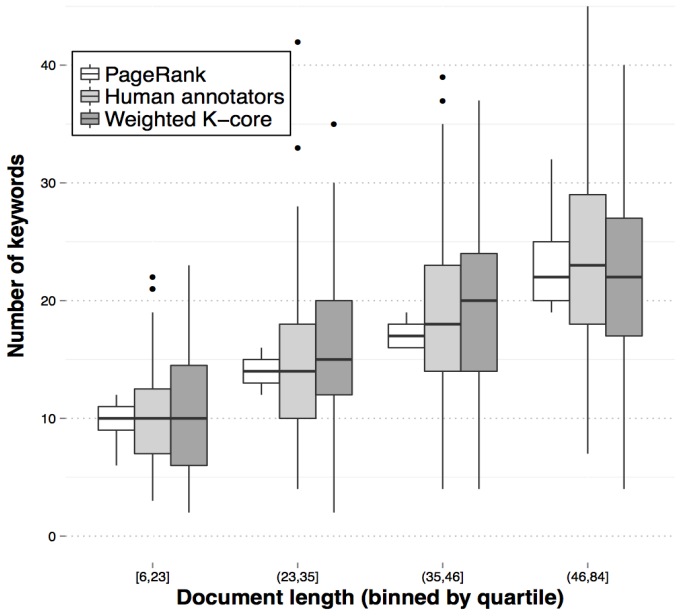
- Most techniques in keyword extraction assign a score to each feature and then take the top ones
- But how many?
  - Absolute number (top X) or relative number (top X%)?
- Besides, at fixed document length, humans may assign more keywords for a document than for another one

X is decided at document level (size of the k-core subgraph)

Performance evaluation

- Precision
- Recall
- F1-score
- Precision/recall
- # of keywords





Graph	Dataset	Macro-averaged precision (%)				Macro-averaged recall (%)				Macro-averaged F1-score (%)			
		PageRank	HITS	K-core	WK-core	PageRank	HITS	K-core	WK-core	PageRank	HITS	K-core	WK-core
undirected edges	Hulth2003	58.94	57.86	46.52	**61.24***	42.19	41.80	**62.51***	50.32*	47.32	46.62	49.06*	**51.92***
	Krapi2009	50.23	49.47	40.46	**53.47***	48.78	47.85	**78.36***	50.21	49.59	47.96	46.61	**50.77***
forward edges	Hulth2003	55.80	54.75	42.45	**56.99***	41.98	40.43	**72.87***	46.93*	45.70	45.03	**51.65***	50.59*
	Krapi2009	47.78	47.03	39.82	**52.19***	44.91	44.19	**79.06***	45.67	45.72	44.95	46.03	**47.01***
backward edges	Hulth2003	59.27	56.41	40.89	**60.24***	42.67	40.66	**70.57***	49.91*	47.57	45.37	45.20	**50.03***
	Krapi2009	51.43	49.11	39.17	**52.14***	49.96	47.00	**77.60***	50.16	**50.51**	47.38	46.93	50.42

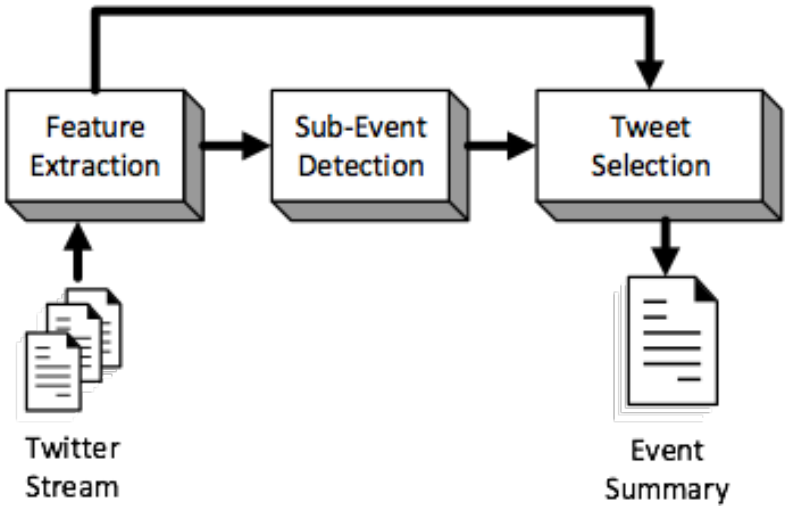
Sub-event detection in Twitter streams

Real Time Event Summarization

1. Feature Extraction: Extracts the terms that best describe the current state of the event

2. Sub-Event Detection: Decides whether a sub-event has occurred

3. Tweet Selection: Ranks all the tweets and selects the first one



System Architecture

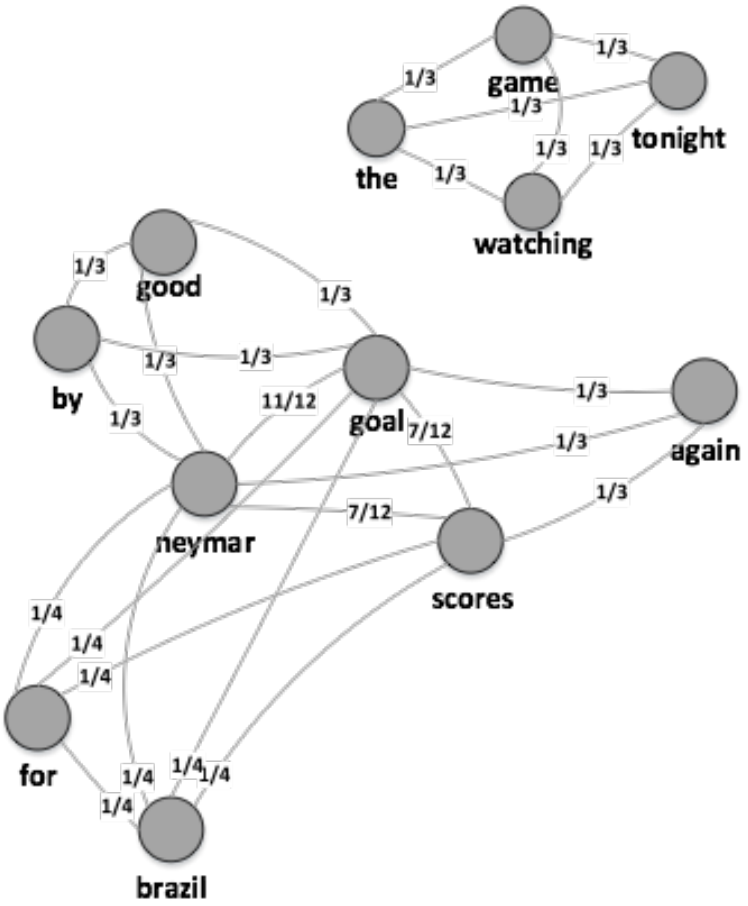
[Meladianos et al. '15]

Graph-based representation of Tweets

- Represents all the input tweets
- node ⟷ unique term
- edge ⟷ #co-occurrences within a tweet (normalized)

Example graph:

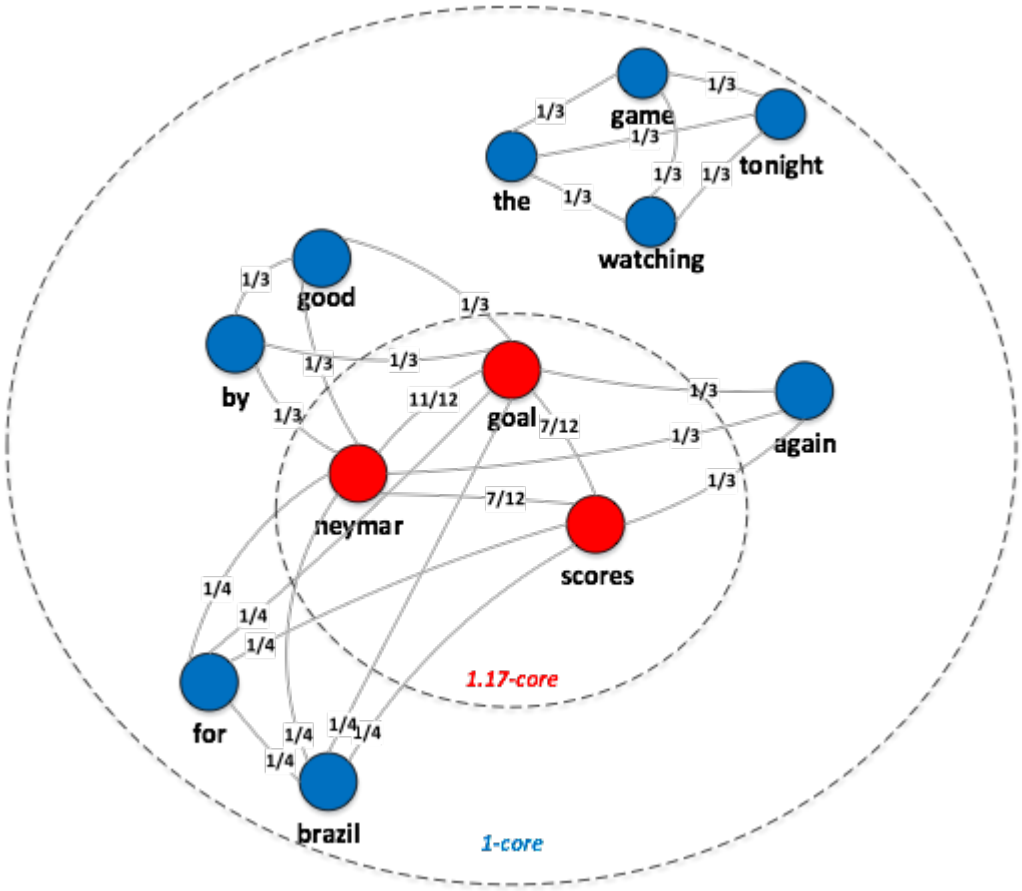
1) Good goal by Neymar
2) Goal! Neymar scores for brazil
3) Goal!! Neymar scores again
4) Watching the game tonight



The graph that was built from 4 tweets

k-core decomposition for feature extraction

- Each term is given a score corresponding to its core number
- Extract the k-core subgraph
- Detect sub-events by considering how the sum of the core numbers extracted from the graph at time t has changed from a previous time point t-1

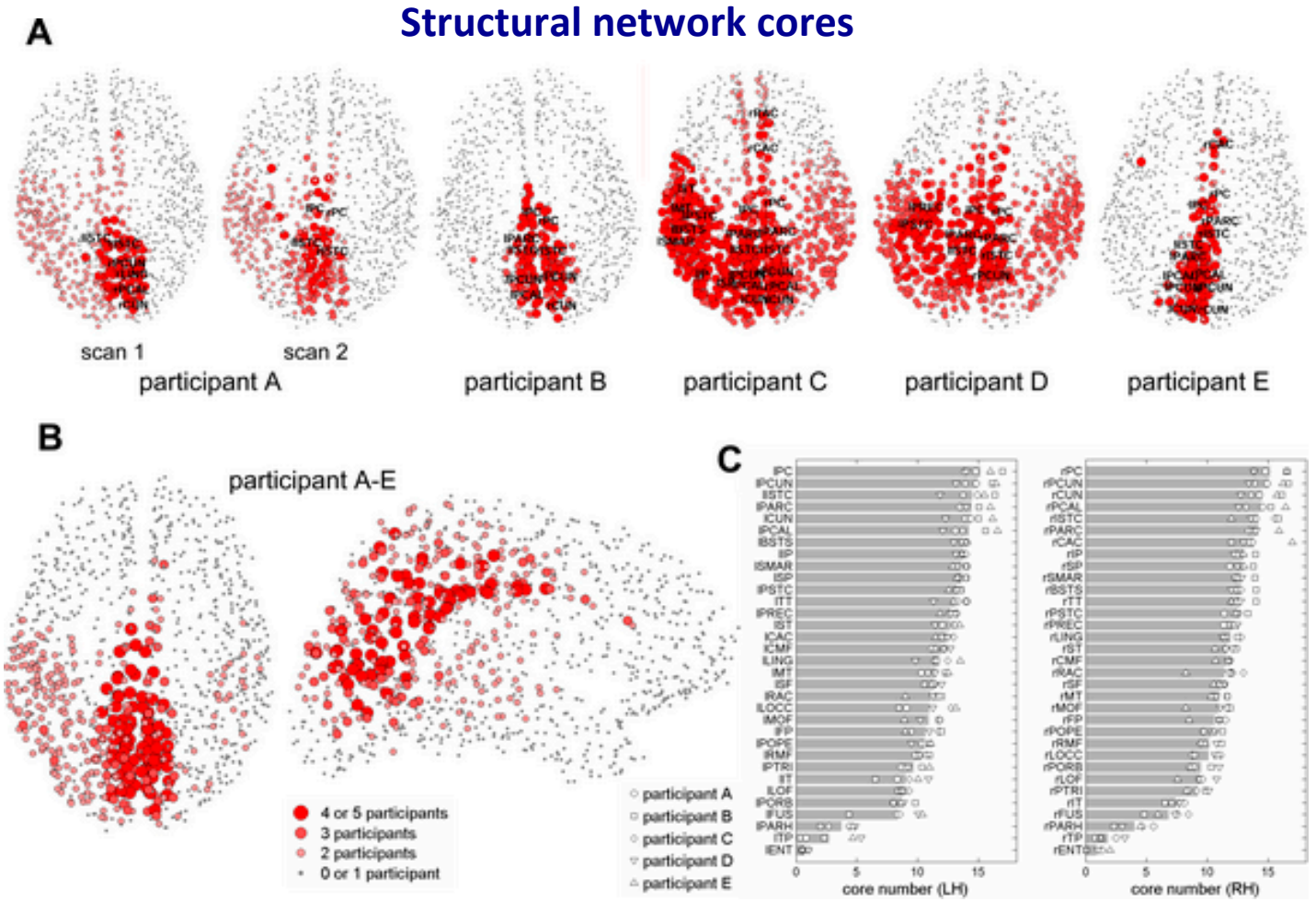


k-core decomposition of the Graph-of-Words

Applications of core decomposition

- Community detection and evaluation
  - Dense subgraph discovery
  - Speed-up community detection algorithms
- Identification of influential spreaders in complex networks
- Dynamics of real-world networks
  - Internet topology
  - Engagement dynamics in social networks
- Network visualization
- Text analytics
- Brain network analysis

k-core decomposition for brain network analysis



Hagmann P, Cammoun L, Gigandet X, Meuli R, Honey CJ, et al. (2008) Mapping the Structural Core of Human Cerebral Cortex. PLoS Biol 6(7): e159. doi: 10.1371/journal.pbio.0060159
http://journals.plos.org/plosbiology/article?id=info:doi/10.1371/journal.pbio.0060159

References (Applications)

- M.E.J. Newman. The structure and function of complex networks. SIAM REVIEW 45, 2003.
- M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. Physical Review E 69(02), 2004.
- S.E. Schaeffer. Graph clustering. Computer Science Review 1(1), 2007.
- S. Fortunato. Community detection in graphs. Physics Reports 486 (3-5), 2010.
- L. Danon, J. Duch, A. Arenas, and A. Diaz-guilera. Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment 9008 , 2005.
- M. Coscia, F. Giannotti, and D. Pedreschi. A classification for community discovery methods in complex networks. Statistical Analysis and Data Mining 4 (5), 2011.
- S.B. Seidman. Network structure and minimum degree. Social Networks 5, pp. 269–287 , 1983.
- J. Håstad. Clique is hard to approximate within n^(1-ε). Acta Mathematica 182, 1999.
- A.V. Goldberg. Finding a maximum density subgraph. Technical report, University of California at Berkeley, 1984.
- R. Andersen and K. Chellapilla. Finding dense subgraphs with size bounds. In: WAW, 2009.

References (Applications)

- A. Angel, N. Sarkas, N. Koudas, and D. Srivastava. Dense subgraph maintenance under streaming edge weight updates for real-time story identification. Proc. VLDB Endow., 2012.
- D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs. In: VLDB, 2004.
- Y. Dourisboure, F. Geraci, and M. Pellegrini. Extraction and classification of dense communities in the web. In: WWW, 2007.
- R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. Computer Networks, 31(11-16), 1999.
- M. Altaf-Ul-Amin, K. Nishikata, T. Koma, T. Miyasato, Y. Shinbo, M. Arifuzzaman, C. Wada, M. Maeda, and T. Oshima. Prediction of protein functions based on k-cores of protein-protein interaction networks and amino acid sequences. Genome Informatics 14: 498–499, 2003.
- C. Giatsidis, F.D. Malliaros, D.M. Thilikos, and M. Vazirgiannis. CoreCluster: A degeneracy-based graph clustering framework. In: AAAI, 2014.
- M. Kitsak, L. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H.E. Stanley, and H. Makse. Identification of Influential Spreaders in Complex Networks. Nature Physics, 6(11):888–893, 2010.

References (Applications)

- J. Bae and S. Kim. Identifying and ranking influential spreaders in complex networks by neighborhood coreness. Physica A: Statistical Mechanics and its Applications, 395:549 – 559, 2014.
- S. Pei and H. A. Makse. Spreading dynamics in complex networks. Journal of Statistical Mechanics: Theory and Experiment, 2013.
- J. Cohen. Trusses: Cohesive subgraphs for social network analysis. National Security Agency Technical Report, 2008.
- J. Wang and J. Cheng. Truss decomposition in massive networks. Proc. VLDB Endow., 5(9):812–823, 2012.
- M.-E.G. Rossi, F.D. Malliaros, and M. Vazirgiannis. Spread it Good, Spread it Fast: Identification of Influential Nodes in Social Networks. In: WWW, 2015.
- F.D. Malliaros, M.-E.G. Rossi and M. Vazirgiannis. Location Influential Nodes in Compex Networks. Scientific Reports, 19307, 2016.
- J.I. Alvarez-Hamelin, L. Dall'Asta, A/ Barrat, and A. Vespignani. k-core decomposition of Internet graphs: hierarchies, self-similarity and measurement biases. NHM 3.2, 2008.
- S. Carmi, S. Havlin, S. Kirkpatrick, Y. Shavitt, and Eran Shir. A Model of Internet Topology using *k*-shell Decomposition. PNAS 104:27, 2007.
- J. Ugander, L. Backstrom, C. Marlow, and J. Kleinberg. Structural Diversity in Social Contagion. PNAS 109(16), 2012.

References (Applications)

- F.D. Malliaros and M. Vazirgiannis. To stay or not to stay: modeling engagement dynamics in social graphs. In: CIKM, 2013.
- K. Bhawalkar, J. Kleinberg, K. Lewi, T. Roughgarden, and A. Sharma. Preventing unraveling in social networks: the anchored k-core problem. In: ICALP, 2012.
- V.H. Manshadi and R. Johari. Supermodular network games. In: Allerton, 2009.
- A. Harkins. Network games with perfect complements. Tech. rep. University of Warwick, 2013.
- D. Garcia, P. Mavrodiev, and F. Schweitzer. Social Resilience in Online Communities: The Autopsy of Friendster. In: COSN, 2013.
- F.D. Malliaros and M. Vazirgiannis. Vulnerability assessment in social networks under cascade-based node departures. EPL (Europhysics Letters) 110(6), 2015.
- J.I. Alvarez-Hamelin , A. Barrat , and A. Vespignani. Large scale networks fingerprinting and visualization using the k-core decomposition. In: NIPS, 2006.
- F. Rousseau and M. Vazirgiannis. Main Core Retention on Graph-of-words for Single-Document Keyword Extraction. In: ECIR, 2015.

Outline

1. Introduction and Motivation
2. Fundamental Concepts and Definitions of Core Decomposition
3. Algorithms
4. Applications
5. **Open Topics and Future Research Directions**

Open problems and future research directions (1)

- **New concepts & applications of degeneracy in social graphs**
  - Reciprocity in signed graphs
    - Reciprocity is defined at node level indicating the average return rate of individual actions
    - Evaluation of reciprocity in trust networks
  - User's engagement in social graphs based on core decomposition
    - Important for prediction and monitoring of social networks evolution
    - Engagement on graphs with rich semantics (e.g., directed, signed)
    - Prediction algorithms for network vulnerability

Open problems and future research directions (2)

- **New concepts & applications of degeneracy in social graphs**
  - Identification of multiple influential spreaders
    - Influence maximization problem
  - Further applications of degeneracy in text retrieval and mining
    - Document summarization and document categorization

Open problems and future research directions (3)

Algorithmic and scalability issues

- Design of approximation algorithms
- Distributed implementation in modern engines (e.g., Spark) and scalability tests for massive graphs
- Degeneracy for large scale graph clustering
  - Degeneracy identifies the cores of the best clusters
  - The degenerated data are exponentially smaller than the original one so the scheme scales

Acknowledgments

- DIGITEO Chair Grant (France) – M. Vazirgiannis

- Visit our prototype:

http://www.graphdegeneracy.org

Thank You!! - Questions?

- **Fragkiskos D. Malliaros**

  University of California, San Diego

  fmalliaros@ucsd.edu

  http://fragkiskos.me

- **Prof. Apostolos N. Papadopoulos**

  Aristotle University of Thessaloniki, Greece

  papadopo@csd.auth.gr

  http://delab.csd.auth.gr/~apostol

- **Prof. Michalis Vazirgiannis**

  École Polytechnique, France

  mvazirg@lix.polytechnique.fr

  http://www.lix.polytechnique.fr/~mvazirg