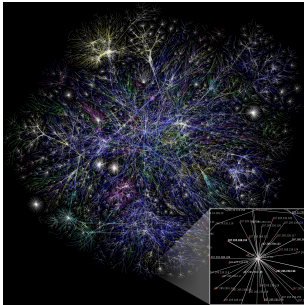


Modeling and Detecting Community Hierarchies

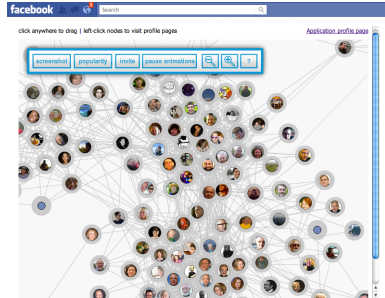
Maria-Florina Balcan, Yingyu Liang
Georgia Institute of Technology

Age of Networks

- Massive amount of network data
- How to understand and utilize?



Internet [1]

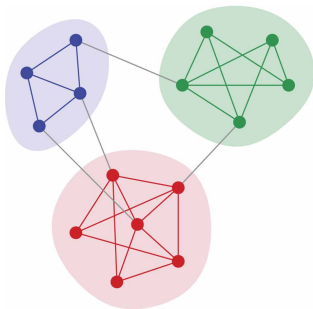


Social Network [2]

Communities

Groups having tighter connections inside than with outside

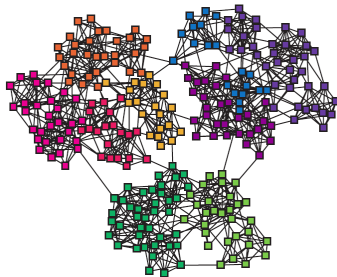
- Reveal network structures
- Help decision making: advertisement, task partitioning, ...



Communities

Hierarchical nature: community contains sub-communities

- Commonly observed but rarely analyzed theoretically



A hierarchical network [3]

Our Contributions

- Theoretical model for community hierarchy
- Efficient algorithm with provable guarantee
- Empirical evaluation

Model Definition

Given similarity function S on n entities

- S obtained from adjacent matrix A of the network,
e.g. $S = A$; $S = \exp\{A\}$; ...

Intuitions about communities

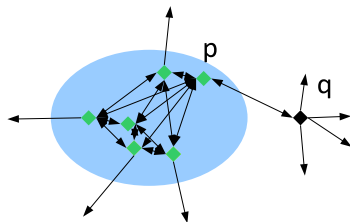
- tighter connections within than with the outside world
- hierarchical organization

Model Definition

Compact Blob

C is a compact blob if out of $|C|$ nearest neighbors,

- [internal] any $p \in C$ has few neighbors outside C
- [external] any $q \notin C$ has few neighbors inside C



Note:

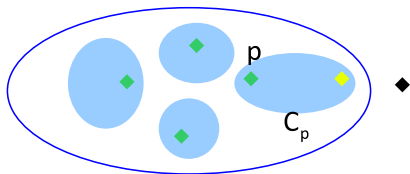
- few neighbors: $\leq \alpha n$
- compact blob should be sufficiently large, i.e. $\geq O(\alpha n)$

Model Definition

Stable Community

C is a stable community if

- [local] any point $p \in C$ falls into a compact blob $C_p \subseteq C$
- [between blobs] a majority of points in the blob C_p have few neighbors outside C out of the $|C|$ nearest neighbors
- [external] any point $q \notin C$ has few neighbors inside C out of the $|C|$ nearest neighbors

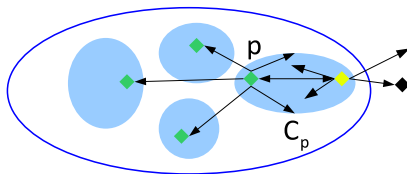


Model Definition

Stable Community

\mathcal{C} is a stable community if

- [local] any point $p \in \mathcal{C}$ falls into a compact blob $\mathcal{C}_p \subseteq \mathcal{C}$
- [between blobs] a majority of points in the blob \mathcal{C}_p have few neighbors outside \mathcal{C} out of the $|\mathcal{C}|$ nearest neighbors
- [external] any point $q \notin \mathcal{C}$ has few neighbors inside \mathcal{C} out of the $|\mathcal{C}|$ nearest neighbors

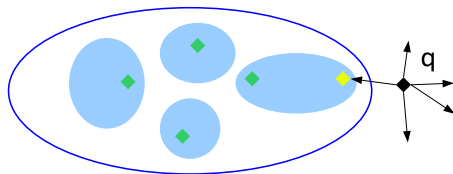


Model Definition

Stable Community

C is a stable community if

- [local] any point $p \in C$ falls into a compact blob $C_p \subseteq C$
- [between blobs] a majority of points in the blob C_p have few neighbors outside C out of the $|C|$ nearest neighbors
- [external] any point $q \notin C$ has few neighbors inside C out of the $|C|$ nearest neighbors

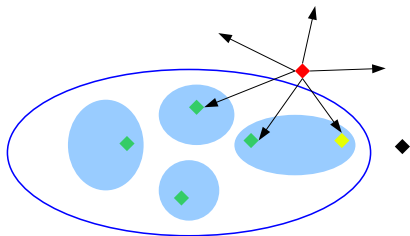


Model Definition

Stable Community

\mathcal{C} is a stable community if **after removing $\leq \nu n$ bad points,**

- [local] any point $p \in \mathcal{C}$ falls into a compact blob $\mathcal{C}_p \subseteq \mathcal{C}$
- [between blobs] a majority of points in the blob \mathcal{C}_p have few neighbors outside \mathcal{C} out of the $|\mathcal{C}|$ nearest neighbors
- [external] any point $q \notin \mathcal{C}$ has few neighbors inside \mathcal{C} out of the $|\mathcal{C}|$ nearest neighbors



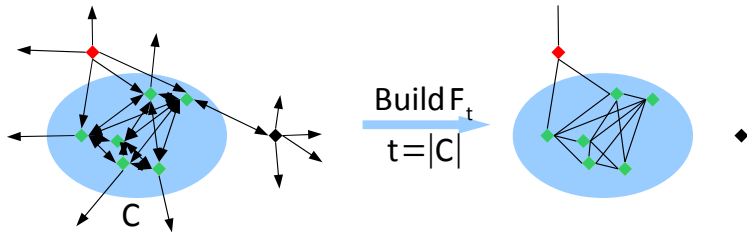
Detection Algorithm

Easy case: blob with known size

Consider a compact blob C with $|C|$ known.

(1) Build F_t by connecting points that share many neighbors out of the $t = |C|$ nearest neighbors

- good points in C and those outside C are disconnected
- good points in C are all connected



Detection Algorithm

Easy Case: Blob with Known Size

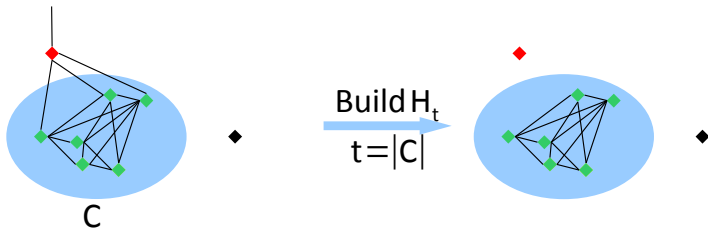
Consider a compact blob C with $|C|$ known.

(2) Build H_t by connecting points with many neighbors in F_t

- Bad point “bridges” are disconnected

(3) Merge components in H_t ;

- One of the components represents C



Detection Algorithm

Easy Case: Blob with Unknown Size

Consider a compact blob C with $|C|$ unknown.

Vary the threshold t :

- Begin with a small t
- Increase t and build F_t, H_t
- When $t = |C|$, a component in H_t represents C

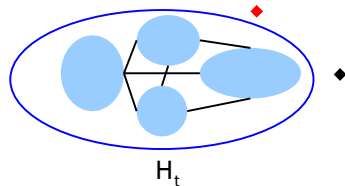
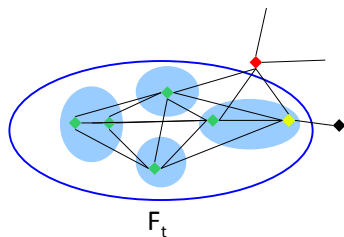
Detection Algorithm

General Case

Consider a stable community C

Build H_t on sets of points instead of on points

- Maintain a list \mathcal{L} of communities
- Build H_t on \mathcal{L} to disconnect bad point “bridges” between sub-communities in C and those outside C
- Merge connected components in H_t to form C



Detection Algorithm

Summary

Hierarchical Community Detection Algorithm

- Initialize \mathcal{L} to be a list of singleton points
Initialize the threshold t to be the size of the minimum blob
- Repeat until all points merged:
 - build F_t, H_t ;
 - update \mathcal{L} by merging large components in H_t ;
 - increase t
- Output a tree with internal nodes corresponding to the merges

Theorem

Any stable community is ν -close to a node in the tree.

Proof

Base Case: Compact Blob

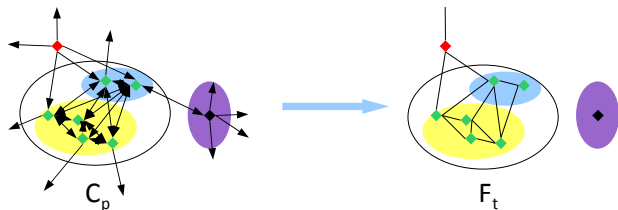
Lemma 1

For any good point p ,

- when $t \leq |C_p|$, good points from C_p will not be merged with good points outside C_p .

Properties of F_t :

- No good point inside C_p is connected to good points outside
- No bad point is connected to both a good point inside and a good point outside



Proof

Base Case: Compact Blob

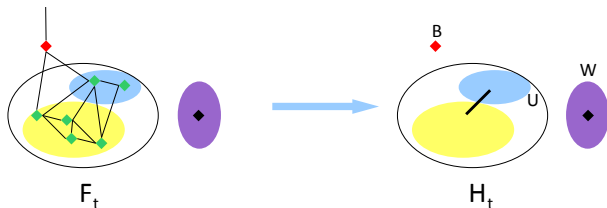
Lemma 1

For any good point p ,

- when $t \leq |C_p|$, good points from C_p will not be merged with good points outside C_p .

Properties of H_t :

- U : community in \mathcal{L} containing good points inside
- W : community in \mathcal{L} containing good points outside
- B : community in \mathcal{L} containing only bad points



Proof

Base Case: Compact Blob

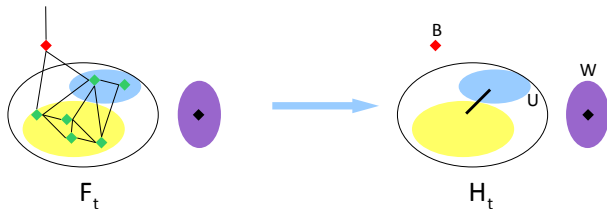
Lemma 1

For any good point p ,

- when $t \leq |C_p|$, good points from C_p will not be merged with good points outside C_p .

Properties of H_t :

- U is not connected to W
- B cannot be connected to both U and W



Proof

Base Case: Compact Blob

Lemma 2

For any good point p ,

- when $t = |C_p|$, all good points in C_p are merged into one community.

Properties of F_t, H_t :

- All good points inside are connected in F_t
- All communities containing good points inside are connected in H_t

Lemma 3

For any stable community C ,

- when $t \leq |C|$, good points from C will not be merged with good points outside C .
- when $t = |C|$, all good points in C are merged into one community.

Proof Sketch:

- Lemma 1 and 2 show:
compact blobs in C are formed
- Similar arguments as in Lemma 1 and 2 then show:
these compact blobs are merged into one community

Experiment

Lift network adjacent matrix A to similarity function S

- direct lifting: $S = A$
- diffusion lifting: $S = \exp\{\lambda A\}$, $\lambda = 0.05$

Evaluation criterion:

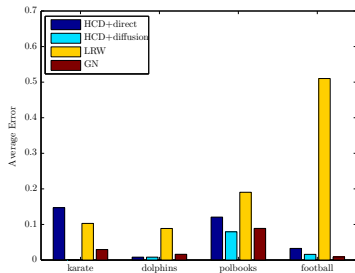
- Recover error of a true community C w.r.t. the tree \mathcal{T}

$$\text{error}(C, \mathcal{T}) = \min_{C' \in \mathcal{T}} \frac{|C \oplus C'|}{n}$$

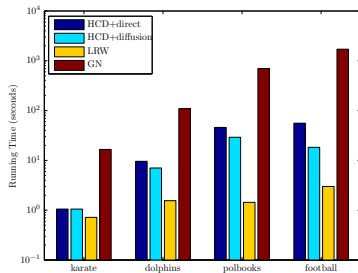
Experiment

Real World Data

Compare our algo (HCD) to:
Lazy Random Walk (LRW [6]), Girvan-Newman algo (GN [4])



Average recover error



Running time (log scale)

4 network type with two level community hierarchies ([5])

Data set	n	m	k	$maxk$
LF50	50	≈ 500	10	15
LF100	100	≈ 1500	15	20
LF150	150	≈ 3000	20	30
LF200	200	≈ 6000	30	40

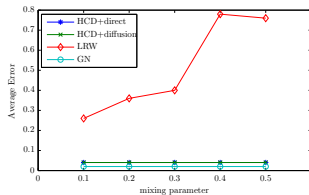
Table: The parameters of the synthetic data sets. n/m : number of nodes/edges; $k/maxk$: average/maximum degree of the nodes.

For each type, vary a mixing parameter to get 5 networks

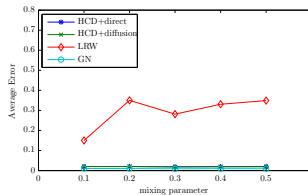
- mixing parameter: probability of connecting points inside a community to points outside
- larger parameter: more difficult to recover communities

Experiment

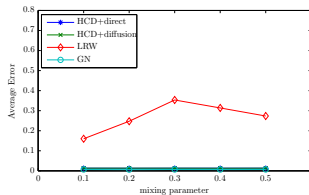
Synthetic Data



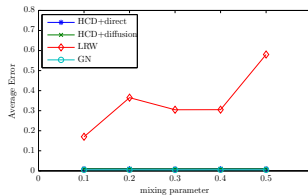
(a) LF50



(b) LF100



(c) LF150

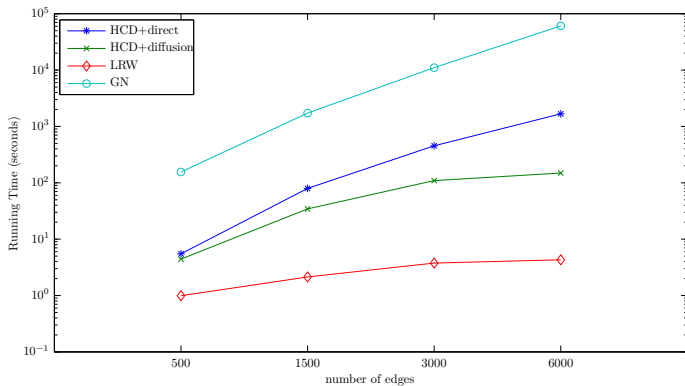


(d) LF200

Average error v.s. mixing parameter






Experiment

Synthetic Data



Running time (log scale) v.s. network size

Thanks!

-  Internet. en.wikipedia.org.
-  Social network. www.steve-dale.net.
-  Aaron Clauset, Cristopher Moore, and M. E. J. Newman.
Hierarchical structure and the prediction of missing links in networks.
Nature, 453(7191):98–101, May 2008.
-  M. Girvan and M. E. J. Newman.
Community structure in social and biological networks.
Proceedings of the National Academy of Sciences,
99(12):7821–7826, 2002.
-  Andrea Lancichinetti and Santo Fortunato.
Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities.
Physical Review E, 80(1):016118, 2009.



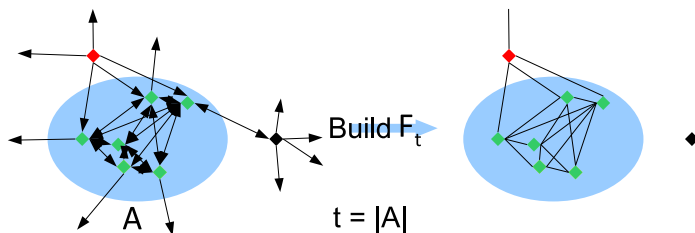
Daniel A. Spielman and Shang-Hua Teng.

Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems.

In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, STOC '04, pages 81–90, New York, NY, USA, 2004. ACM.

Detection Algorithm Details

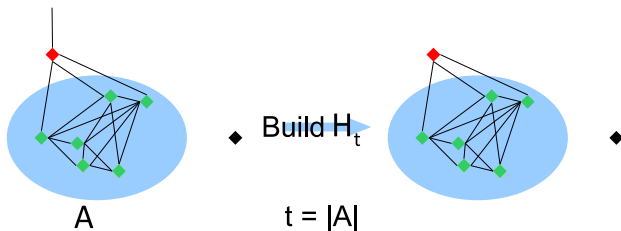
- Initialize \mathcal{L} to be the singleton points.
Initialize the threshold $t = 6(\alpha + \nu)n + 1$.
- For each point, check its t nearest neighbors
- Build F_t by connecting any two points that share more than $t - 2(\alpha + \nu)n$ nearest neighbors



Detection Algorithm Details

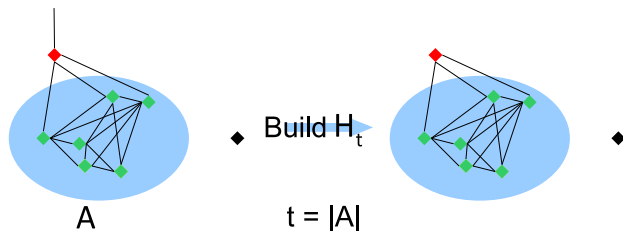
- Build H_t on \mathcal{L} as follows. For any two subsets $u, v \in \mathcal{L}$:
 - if u, v are both singletons: connect them when they share more than νn singleton subsets as neighbors in common in F_t .
 -

■



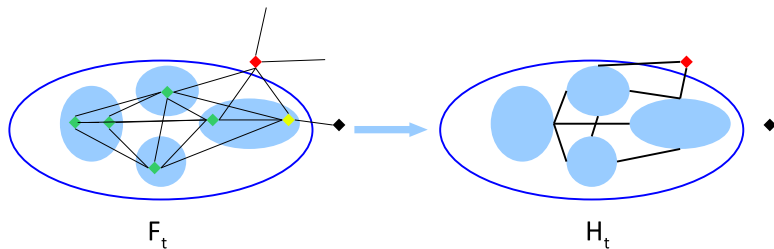
Detection Algorithm Details

- Build H_t on \mathcal{L} as follows. For any two subsets $u, v \in \mathcal{L}$:
 - if u, v are both singletons: connect them when they share more than νn singleton subsets as neighbors in common in F_t .
 - otherwise: for any $x \in u, y \in v$, let $S_t(x, y)$ denote the number of points in $u \cup v$ they share as neighbors in common in F_t . Connect u, v when $\text{median}_{x \in u, y \in v} S_t(x, y)$ is large enough.
- Update \mathcal{L} by merging sufficiently large components in H_t



Detection Algorithm Details

- Increase t and repeat until all points merged.
 - by induction, the sub-communities of C will be formed
 - build F_t to disconnect good points inside and outside
 - build H_t to disconnect the bad point “bridge”
 - merge sufficiently large components in H_t
- Output a tree with internal nodes corresponding to the merges



Detection Algorithm Details

Summary

Hierarchical Community Detection Algorithm

- Initialize \mathcal{L} to be the singleton points.
Initialize the threshold $t = 6(\alpha + \nu)n + 1$.
- Repeat until all points merged:
 build F_t, H_t ; update \mathcal{L} ; increase t
- Output a tree with internal nodes corresponding to the merges

Theorem

Any stable community is ν -close to a node in the tree.