Quality Functions in Graph Clustering

Leonidas Pitsoulis

Department of Electrical and Computer Engineering

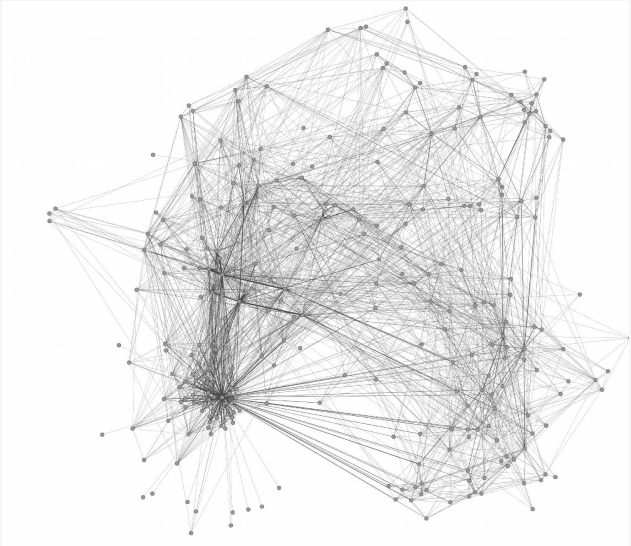
Aristotle University of Thessaloniki, Greece

Workshop on clustering and search techniques in large scale networks
Nizhny Novgorod, Russia
3-8 November 2014

Outline of the talk

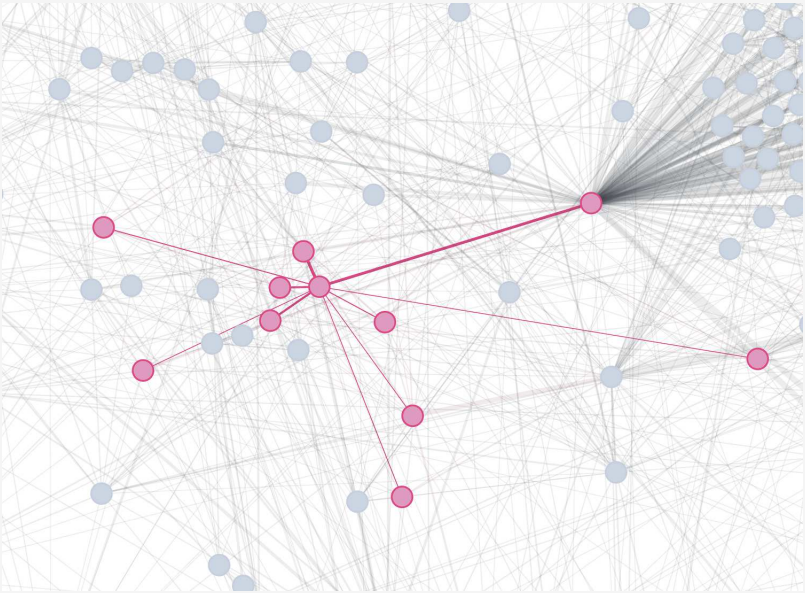
Communities in (complex) Networks

Neural network of Caenorhabditis Elegans (D. J. Watts and S. H. Strogatz, Nature 393, 440-442 (1998))



Communities in (complex) Networks

Neural network of Caenorhabditis Elegans (D. J. Watts and S. H. Strogatz, Nature 393, 440-442 (1998))

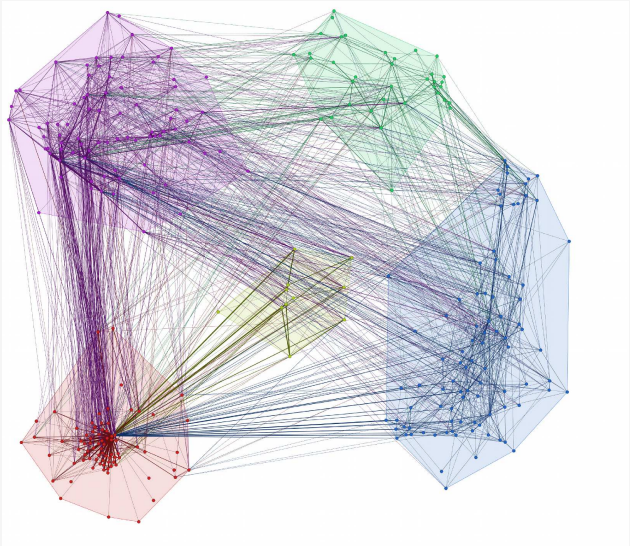


Communities in (complex) Networks

Neural network of Caenorhabditis Elegans (D. J. Watts and S. H. Strogatz, Nature 393, 440-442 (1998))

Communities in (complex) Networks

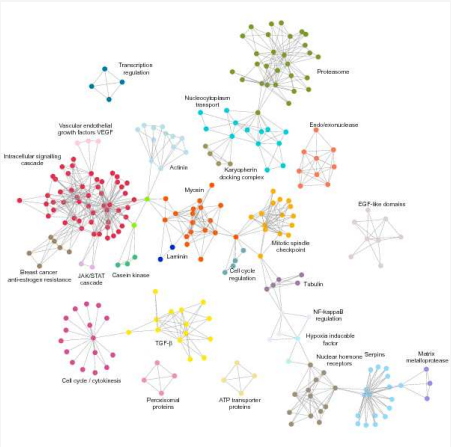
Neural network of Caenorhabditis Elegans (D. J. Watts and S. H. Strogatz, Nature 393, 440-442 (1998))



Community Detection

Community detection appears as a problem in many real-life networks

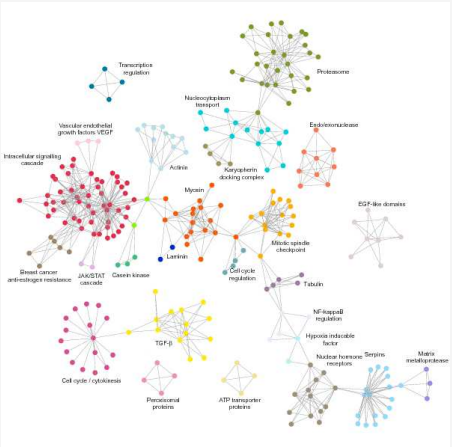
- protein-protein interaction networks
- metabolic networks
- social networks
- WWW (search engines)
- scientific collaboration networks
- mobile phone networks



Community Detection

Community detection appears as a problem in many real-life networks

- protein-protein interaction networks
- metabolic networks
- social networks
- WWW (search engines)
- scientific collaboration networks
- mobile phone networks

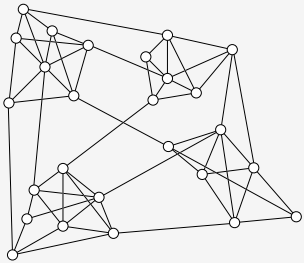


In all cases we are interested in mesoscopic system behavior, derived from the known microscopic dynamics.

Outline of the talk

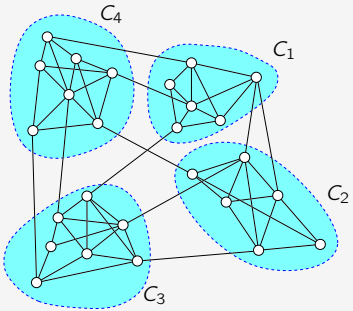
Clustering in Graphs = Community Structure

- Graph $G = (V, E)$ and let $|V(G)| = n$ while $|E(G)| = m$



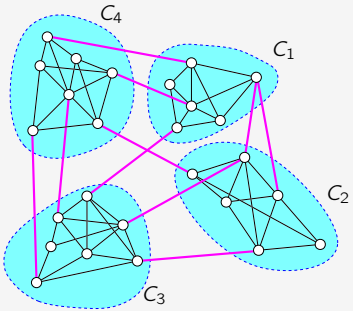
Clustering in Graphs = Community Structure

- Graph $G = (V, E)$ and let $|V(G)| = n$ while $|E(G)| = m$
- A **clustering** $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ is a partition of $V(G)$
- The $C_i \in \mathcal{C}$ are the **clusters**



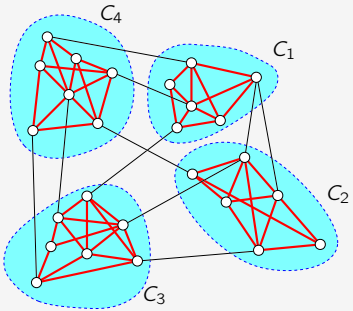
Clustering in Graphs = Community Structure

- Graph $G = (V, E)$ and let $|V(G)| = n$ while $|E(G)| = m$
- A **clustering** $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ is a partition of $V(G)$
- The $C_i \in \mathcal{C}$ are the **clusters**
- $\mathcal{C}$ partitions also $|E(G)|$ into:
- **extra-cluster** edges denoted $E^-(G, \mathcal{C})$



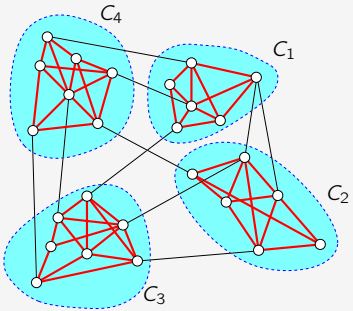
Clustering in Graphs = Community Structure

- Graph $G = (V, E)$ and let $|V(G)| = n$ while $|E(G)| = m$
- A **clustering** $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ is a partition of $V(G)$
- The $C_i \in \mathcal{C}$ are the **clusters**
- $\mathcal{C}$ partitions also $|E(G)|$ into:
- **extra-cluster** edges denoted $E^-(G, \mathcal{C})$
- **intra-cluster** edges denoted $E^+(G, \mathcal{C})$



Clustering in Graphs = Community Structure

- Graph $G = (V, E)$ and let $|V(G)| = n$ while $|E(G)| = m$
- A **clustering** $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ is a partition of $V(G)$
- The $C_i \in \mathcal{C}$ are the **clusters**
- $\mathcal{C}$ partitions also $|E(G)|$ into:
- **extra-cluster** edges denoted $E^-(G, \mathcal{C})$
- **intra-cluster** edges denoted $E^+(G, \mathcal{C})$



A network exhibits community structure, if there is a partition of the vertices into groups where the **density** of edges joining the vertices within the groups is higher than the density of edges joining the groups themselves

Comparing clusterings

Definition (**Jaccard similarity coefficient**)

Given $G(V, E)$ and two clusterings $\mathcal{C}_1, \mathcal{C}_2$ let

$a_{1,1}$ = number of vertex pairs which belong to same cluster in both $\mathcal{C}_1$ and $\mathcal{C}_2$

$a_{1,0}$ = number of vertex pairs which belong to same cluster in $\mathcal{C}_1$ only

$a_{0,1}$ = number of vertex pairs which belong to same cluster in $\mathcal{C}_2$ only

The Jaccard similarity coefficient is defined as

$$J(\mathcal{C}_1, \mathcal{C}_2) = \frac{a_{1,1}}{a_{1,0} + a_{0,1} + a_{1,1}}$$

Comparing clusterings

Definition (**Jaccard similarity coefficient**)

Given $G(V, E)$ and two clusterings $\mathcal{C}_1, \mathcal{C}_2$ let

$a_{1,1}$ = number of vertex pairs which belong to same cluster in both $\mathcal{C}_1$ and $\mathcal{C}_2$

$a_{1,0}$ = number of vertex pairs which belong to same cluster in $\mathcal{C}_1$ only

$a_{0,1}$ = number of vertex pairs which belong to same cluster in $\mathcal{C}_2$ only

The Jaccard similarity coefficient is defined as

$$J(\mathcal{C}_1, \mathcal{C}_2) = \frac{a_{1,1}}{a_{1,0} + a_{0,1} + a_{1,1}}$$

- $J(\mathcal{C}_1, \mathcal{C}_2) \in [0, 1]$ with higher values directly proportional to similarity
- more *exact* algebraic metric for clusterings in Pitsoulis Nanscimento (COR 2013) but requires $\mathcal{O}(n^3)$ time.

Distance between clusterings

Definition

A matrix $S = (s_{ij}) \in \{0,1\}^{k \times n}$ is called a basic clustering matrix if

i) it has no zero rows

ii) $\sum_{i=1}^{k} s_{ij} = 1$ for all $j = 1, \ldots, n$

iii) if s_{ij} is the first nonzero element of row i then $s_{lt} = 0$ for $l = i+1, \ldots, n$ and $t = 1, \ldots, j$.

If only conditions i) and ii) are satisfied then the matrix is called clustering matrix.

$\Rightarrow$ there is a one-to-one correspondence between the set of clusterings of size k and the $\{0,1\}^{k \times n}$ basic clustering matrices

Given any two clustering matrices $S \in \{0,1\}^{k_1 \times n}$ and $T \in \{0,1\}^{k_2 \times n}$ we define their difference set as the set

$$\Delta(S,T) := \{j : S_{ij} \neq T_{ij}, i = 1, \ldots, \min\{k_1, k_2\}, j = 1, \ldots, n\},$$

which is the set of columns that these matrices differ.

Distance between clusterings

Definition

A matrix $S = (s_{ij}) \in \{0,1\}^{k \times n}$ is called a basic clustering matrix if

i) it has no zero rows

ii) $\sum_{i=1}^{k} s_{ij} = 1$ for all $j = 1, \ldots, n$

iii) if s_{ij} is the first nonzero element of row i then $s_{lt} = 0$ for $l = i+1, \ldots, n$ and $t = 1, \ldots, j$.

If only conditions i) and ii) are satisfied then the matrix is called clustering matrix.

$\Rightarrow$ there is a one-to-one correspondence between the set of clusterings of size k and the $\{0,1\}^{k \times n}$ basic clustering matrices

Given any two clustering matrices $S \in \{0,1\}^{k_1 \times n}$ and $T \in \{0,1\}^{k_2 \times n}$ we define their difference set as the set

$$\Delta(S,T) := \{j : S_{ij} \neq T_{ij}, i = 1, \ldots, \min\{k_1, k_2\}, j = 1, \ldots, n\},$$

which is the set of columns that these matrices differ.

Distance between clusterings

Definition

A matrix $S = (s_{ij}) \in \{0,1\}^{k \times n}$ is called a basic clustering matrix if

i) it has no zero rows

ii) $\sum_{i=1}^{k} s_{ij} = 1$ for all $j = 1, \ldots, n$

iii) if s_{ij} is the first nonzero element of row i then $s_{lt} = 0$ for $l = i+1, \ldots, n$ and $t = 1, \ldots, j$.

If only conditions i) and ii) are satisfied then the matrix is called clustering matrix.

$\Rightarrow$ there is a one-to-one correspondence between the set of clusterings of size k and the $\{0,1\}^{k \times n}$ basic clustering matrices

Given any two clustering matrices $S \in \{0,1\}^{k_1 \times n}$ and $T \in \{0,1\}^{k_2 \times n}$ we define their difference set as the set

$$\Delta(S, T) := \{j : S_{ij} \neq T_{ij}, i = 1, \ldots, \min\{k_1, k_2\}, j = 1, \ldots, n\},$$

which is the set of columns that these matrices differ.

Distance beween clusterings

The distance between any two basic clustering matrices $S^1 \in \{0,1\}^{k_1 \times n}$ and $S^2 \in \{0,1\}^{k_2 \times n}$ is thus defined as

$$d(S^1, S^2) := \min\{|\Delta(S, T)| : S \in \mathcal{M}(S^1),\, T \in \mathcal{M}(S^2)\}.$$

Distance beween clusterings

The distance between any two basic clustering matrices $S^1 \in \{0,1\}^{k_1 \times n}$ and $S^2 \in \{0,1\}^{k_2 \times n}$ is thus defined as

$$d(S^1, S^2) := \min\{|\Delta(S, T)| : S \in \mathcal{M}(S^1),\, T \in \mathcal{M}(S^2)\}.$$

$\Rightarrow$ $d(S^1, S^2)$ is the minimum number of *moves* of elements between the clusters in the clusterings associated with the basic clustering matrices S^1 and S^2, needed to transform one clustering to another

Lemma

For some graph $G(V, E)$ and any three clusterings $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$ the following statements are true:

i) $d(S_{\mathcal{C}_1}, S_{\mathcal{C}_2}) \geq 0$ *with equality iff* $S_{\mathcal{C}_1} = S_{\mathcal{C}_2}$

ii) $d(S_{\mathcal{C}_1}, S_{\mathcal{C}_2}) = d(S_{\mathcal{C}_2}, S_{\mathcal{C}_1})$

iii) $d(S_{\mathcal{C}_1}, S_{\mathcal{C}_2}) + d(S_{\mathcal{C}_2}, S_{\mathcal{C}_3}) \geq d(S_{\mathcal{C}_1}, S_{\mathcal{C}_3})$

Clusterings - Distance

Problem: direct computation of the distance requires $(\min\{k_1, k_2\})!$ steps.

Clusterings - Distance

Problem: direct computation of the distance requires $(\min\{k_1, k_2\})!$ steps.

Proposition (Pitsoulis & Nascimento (COR 2013))

Given two basic clustering matrices $S^1 \in \{0,1\}^{k_1 \times n}$ *and* $S^2 \in \{0,1\}^{k_2 \times n}$ *their distance* $d(S^1, S^2)$ *can be computed in* $\mathcal{O}(k^3)$ *time, where* $k := \min\{k_1, k_2\}$.

Clusterings - Distance

Problem: direct computation of the distance requires $(\min\{k_1, k_2\})!$ steps.

Proposition (Pitsoulis & Nascimento (COR 2013))

Given two basic clustering matrices $S^1 \in \{0,1\}^{k_1 \times n}$ and $S^2 \in \{0,1\}^{k_2 \times n}$ their distance $d(S^1, S^2)$ can be computed in $\mathcal{O}(k^3)$ time, where $k := \min\{k_1, k_2\}$.

Proof.

Given the two basic clustering matrices $S^1 = (s^1_{ij})$ and $S^2 = (s^2_{ij})$, construct a $k \times k$ cost matrix $C = (c_{ij})$

$$c_{ij} := \sum_{l=1}^{n} |s^1_{il} - s^2_{jl}|,$$

for $i, j = 1, \ldots, k$. Then optimum solution to related LAP gives the distance. □

Example

Say we have $n = 10$ vertices and two clusterings

$$\mathcal{C}_1 = \{\{1, 4, 5\}, \{2\}, \{3, 8\}, \{6, 7\}, \{9, 10\}\},$$

$$\mathcal{C}_2 = \{\{1, 2, 9\}, \{3, 8\}, \{4, 5, 10\}, \{6, 7\}\}$$

Then the basic clustering matrices

$$S_{\mathcal{C}_1} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}, S_{\mathcal{C}_2} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

For the distance computation

$$C = \begin{bmatrix} 4 & \boxed{2} & 5 & 5 \\ 5 & 3 & \boxed{0} & 4 \\ \boxed{2} & 4 & 4 & 5 \\ 5 & 3 & 4 & \boxed{0} \end{bmatrix},$$

Example

So the optimum permutation is $p = (3, 1, 2, 4)$ and

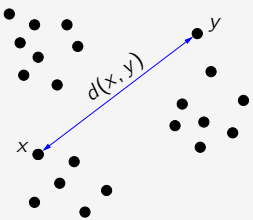
$$S = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

So we have $\Delta^*(S_{\mathcal{C}_1}, S_{\mathcal{C}_2}) = \Delta(S_{\mathcal{C}_1}, S) = \{1, 9, 10\}$, which implies that $d(S_{\mathcal{C}_1}, S_{\mathcal{C}_2}) = 3$.

Outline of the talk

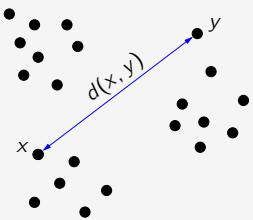
Distance based clustering

- $X = \{x_1, x_2, \ldots, x_n\}$ is a **data set**
- $d : X \times X \to \mathbb{R}$ is a **distance function**:



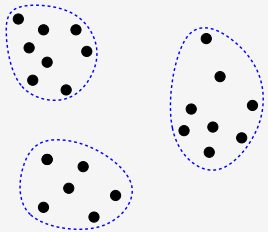
Distance based clustering

- $X = \{x_1, x_2, \ldots, x_n\}$ is a **data set**
- $d : X \times X \to \mathbb{R}$ is a **distance function**:
- i. $d(x, y) \geq 0, \forall x, y \in X$
- ii. $d(x, y) = 0$ iff $x = y$
- iii. $d(x, y) = d(y, x)$



Distance based clustering

- $X = \{x_1, x_2, \ldots, x_n\}$ is a **data set**
- $d : X \times X \to \mathbb{R}$ is a **distance function**:
- i. $d(x, y) \geq 0, \forall x, y \in X$
- ii. $d(x, y) = 0$ iff $x = y$
- iii. $d(x, y) = d(y, x)$
- A **clustering** $\mathcal{C}$ is a partition of X



Clustering functions

Definition

- A **clustering function** is a function F which given a data set X and a distance function d it returns a partition $\mathcal{C}$ of X.

$$F : (X, d) \to \mathcal{C}$$

- A **clustering quality function** is any function Q which given a data set X, a partioning $\mathcal{C}$ of X and a distance function d it returns a real number.

$$Q : (X, d, \mathcal{C}) \to \mathbb{R}$$

Clustering functions

Definition

- A **clustering function** is a function F which given a data set X and a distance function d it returns a partition $\mathcal{C}$ of X.

$$F : (X, d) \to \mathcal{C}$$

- A **clustering quality function** is any function Q which given a data set X, a partioning $\mathcal{C}$ of X and a distance function d it returns a real number.

$$Q : (X, d, \mathcal{C}) \to \mathbb{R}$$

Given Q we can define F as the extrema

$$F(X, d) = \arg\max_{\mathcal{C}} Q(X, d, \mathcal{C})$$

$\Rightarrow$ any property of *clustering functions* can stated for *clustering quality functions*

Kleinberg's Impossibility Theorem

Kleinberg's axioms for clustering functions $F(X, d)$

i. Scale Invariance: F produces the same clustering if distances between points are scaled uniformly.

ii. Richness: if any clustering of the points can be produced by modifying the distances between the points.

iii. Consistency: for any clustering that F produces, decreasing inner cluster distances or increasing outer cluster distances gives a set of points that F produces the same clustering.

Kleinberg's Impossibility Theorem

Kleinberg's axioms for clustering functions $F(X, d)$

i. Scale Invariance: F produces the same clustering if distances between points are scaled uniformly.

ii. Richness: if any clustering of the points can be produced by modifying the distances between the points.

iii. Consistency: for any clustering that F produces, decreasing inner cluster distances or increasing outer cluster distances gives a set of points that F produces the same clustering.

Theorem (Kleinberg (NIPS 2002))

There is no clustering function that satisfies scale invariance, richness and consistency at the same time.

Consistency through quality functions

Ackerman and Ben-David (NIPS 2009) properties for quality functions.

i. Scale Invariance: Q is **scale invariant** if for every clustering $\mathcal{C}$ of (X, d) and every positive λ

$$Q(X, d, \mathcal{C}) = Q(X, \lambda d, \mathcal{C})$$

ii. Richness: Q is **rich** if for any $\mathcal{C}^*$ of X there exists some d over X such that

$$\mathcal{C}^* = \arg\max_{\mathcal{C}} Q(X, d, \mathcal{C})$$

iii. Consistency: Q is **consistent** if for any $\mathcal{C}$ of X, if $d_{\mathcal{C}}$ corresponds to d where intra (extra) cluster distances are decreased (increased) then

$$Q(X, d, \mathcal{C}) \geq Q(X, d_{\mathcal{C}}, \mathcal{C})$$

Consistency through quality functions

Ackerman and Ben-David (NIPS 2009) properties for quality functions.

i. Scale Invariance: Q is **scale invariant** if for every clustering $\mathcal{C}$ of (X, d) and every positive λ

$$Q(X, d, \mathcal{C}) = Q(X, \lambda d, \mathcal{C})$$

ii. Richness: Q is **rich** if for any $\mathcal{C}^*$ of X there exists some d over X such that

$$\mathcal{C}^* = \arg\max_{\mathcal{C}} Q(X, d, \mathcal{C})$$

iii. Consistency: Q is **consistent** if for any $\mathcal{C}$ of X, if $d_{\mathcal{C}}$ corresponds to d where intra (extra) cluster distances are decreased (increased) then

$$Q(X, d, \mathcal{C}) \geq Q(X, d_{\mathcal{C}}, \mathcal{C})$$

- presented a number of quality functions which constitute the above set of axioms consistent
- propose a set of axioms which include relaxations of the above plus **isomorphism invariance**
- the above results can be extended to *graph clustering quality functions*

Outline of the talk

Properties of graph clustering quality functions

We have identified the following properties

i. Isomorphism invariance
ii. Scale invariance
iii. Richness
iv. Monotonicity
v. Perfectness
vi. Connectivity
vii. Convexity
viii. Complementarity
ix. Resolution limit free

Isomorphism

Property (**Isomorphism invariance**)

A quality function Q is **isomorphism invariant** *if for any pair of isomorphic graphs $G_1 \cong G_2$ with isomorphism ϕ, we have*

$$Q(G_1, \mathcal{C}) = Q(G_2, \phi(\mathcal{C})), \qquad \text{for all } \mathcal{C} \in 2^{|V|} \tag{1}$$

where $\phi(\mathcal{C}) = \{\{\phi(v) : v \in C\} : C \in \mathcal{C}\}$.

quality function values of two isomorphic graphs should be equal for clusterings under the same isomorphism

Scaling

Property (**Scale invariance**)

A quality function Q is **scale invariant** *if for a graph G with weight function $w : E(G) \to \mathbb{R}$ and a constant $\alpha > 0$, we have*

$$Q(G, \mathcal{C}) = Q(\alpha G, \mathcal{C}), \qquad \text{for all } \mathcal{C} \in 2^{|V|}, \tag{2}$$

where the weighted graph αG is defined as $E(\alpha G) = E(G)$, $V(\alpha G) = V(G)$ with weight function $z(e) = \alpha w(e)$, $e \in E(\alpha G)$.

quality function should be invariant under a uniform scaling of the edge weights in a graph

Richness

Property (**Richness**)

A quality function Q is **rich** *if for any finite set of vertices V and a partition $\mathcal{C}^* \in 2^{|V|}$ there exists a set of edges E such that for $G = (V, E)$*

$$\mathcal{C}^* = \arg\max\{Q(G, \mathcal{C}) : \mathcal{C} \in 2^{|V|}\}. \tag{3}$$

for any partition of a finite set V we can find a graph with V as its vertex set such that the partition will be the maximum value of the clustering quality function

Monotonicity

Property (**Monotonicity**)

A quality function is **monotone** *if for any graph G, clustering $\mathcal{C}$ of $V(G)$, and any graph G' satisfying:*

(i) $V(G') = V(G)$,

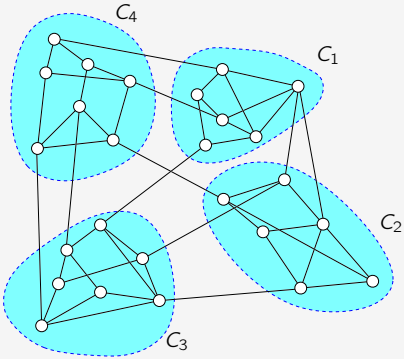
(ii) $E^+(G, \mathcal{C}) \subseteq E^+(G', \mathcal{C})$ *and* $E^-(G', \mathcal{C}) \subseteq E^-(G, \mathcal{C})$,

we have

$$Q(G, \mathcal{C}) \leq Q(G', \mathcal{C}). \tag{4}$$

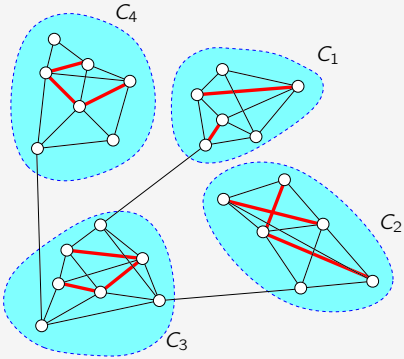
Monotonicity

the value given by the quality function to a clustering upon which we delete extra-cluster edges and/or add of intra-cluster edges should not decrease



Monotonicity

the value given by the quality function to a clustering upon which we delete extra-cluster edges and/or add of intra-cluster edges should not decrease



Perfectness

Property (**Perfectness**)

A quality function is **perfect** *if for any graph* $G(V, E)$*, the following are true*

(i) *if* $\mathcal{C}^*$ *is a clustering on* $V(G)$ *such that we cannot* add *an intra-cluster edge nor* remove *an extra-cluster edge, then*

$$Q(G, \mathcal{C}^*) = \max\{Q(G', \mathcal{C}) : \textit{all } G' \textit{ such that } V(G') = V, \mathcal{C} \in 2^{|V|}\}.$$

(ii) *if* $\mathcal{C}^*$ *is a clustering on* $V(G)$ *such that we cannot* add *an extra-cluster edge nor* remove *an intra-cluster edge, then*

$$Q(G, \mathcal{C}^*) = \min\{Q(G', \mathcal{C}) : \textit{all } G' \textit{ such that } V(G') = V, \mathcal{C} \in 2^{|V|}\}.$$

quality function should provide the maximum value among all possible graphs and clusterings on this vertex set

Connectivity

Property (**Connectivity**)

Let a graph G, a clustering $\mathcal{C}$ that contains a dissconnected cluster C with a partition $\{C_1, C_2, \ldots, C_k\}$ such that $G[C_1], \ldots, G[C_k]$ are the connected components of $G[C]$, and a clustering $\mathcal{D}$ obtained from $\mathcal{C}$ by replacing C with $\{C_1, C_2, \ldots, C_k\}$. A quality function Q is called **connected** *if for any such triple $G, \mathcal{C}, \mathcal{D}$ we have*

$$Q(G, \mathcal{C}) \leq Q(G, \mathcal{D})$$

minimum requirement for a cluster to be be classified as a community is that the associated induced subgraph should be connected

Convexity

Definition

Given a graph $G(V, E)$ some set of vertices $X \subseteq V(G)$ is called **convex** in G if for any pair of vertices $v, w \in X$ the shortest $v - w$ path contains vertices only from X.

Property (**Convexity**)

Let a graph G, a clustering $\mathcal{C}$ that contains a nonconvex cluster C with a partition $\{C_1, C_2, \ldots, C_k\}$ such that $C_1, \ldots, C_k$ are convex, and a clustering $\mathcal{D}$ obtained from $\mathcal{C}$ by replacing C with $\{C_1, C_2, \ldots, C_k\}$. A quality function Q is called **convex** *if for any such triple $G, \mathcal{C}, \mathcal{D}$ we have*

$$Q(G, \mathcal{C}) \leq Q(G, \mathcal{D})$$

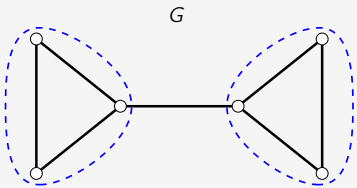
Complentarity

Property (**Complementarity**)

A quality function Q is **complementary** *if for any graph G, its complement $\bar{G}$, and any clustering $\mathcal{C}$ of $V(G)$,*

$$Q'(G, \mathcal{C}) = 1 - Q'(\bar{G}, \mathcal{C})$$

where Q' the function which results as a uniform scaling on the range of Q in the interval $[0, 1]$.



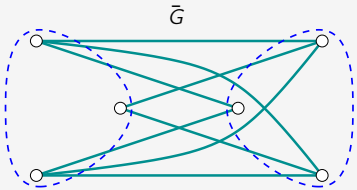
Complentarity

Property (**Complementarity**)

A quality function Q is **complementary** *if for any graph G, its complement $\bar{G}$, and any clustering $\mathcal{C}$ of $V(G)$,*

$$Q'(G, \mathcal{C}) = 1 - Q'(\bar{G}, \mathcal{C})$$

where Q' the function which results as a uniform scaling on the range of Q in the interval $[0, 1]$.



Resolution-limit-free

Introduced by Traag, van Dooren, and Nesterov (2011)

Property (**Resolution-limit-freedom**)

Let $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ be a Q-optimal clustering of a graph G, for some quality function Q. Then, Q is called **resolution-limit-free** *if for each subgraph of G induced by $\mathcal{D} \subset \mathcal{C}$, the partition $\mathcal{D}$ is also Q-optimal.*

attempt to rigorously define the resolution limit of some quality functions

Axiomatic system

Consider that we have an axiomatic system say AQF. Then it should be:

- **consistent**: there exists at least one quality function which satisfies all axioms
- **independent**: there does not exist a set of axioms $\mathcal{A}$ of AQF and an axiom A of AQF such that $\mathcal{A} \not\Rightarrow$ A.

Axiomatic system

Consider that we have an axiomatic system say AQF. Then it should be:

- **consistent**: there exists at least one quality function which satisfies all axioms
- **independent**: there does not exist a set of axioms $\mathcal{A}$ of AQF and an axiom A of AQF such that $\mathcal{A} \not\Rightarrow$ A.

But we would like to have results of the form

Theorem

Let Q_1 and Q_2 be two graph clustering quality functions which satisfy AQF *and G a graph. Then*

$$\arg\max\{Q_1(G,\mathcal{C}) : \mathcal{C} \in 2^{|V(G)|}\} = \arg\max\{Q_2(G,\mathcal{C}) : \mathcal{C} \in 2^{|V(G)|}\}.$$

Outline of the talk

Graph clustering quality functions

We have examined the following types of graph clustering quality functions

i. modularity
ii. density
iii. distance
iv. node membership
v. connectivity

Graph clustering quality functions

We have examined the following types of graph clustering quality functions

i. modularity
ii. density
iii. distance
iv. node membership
v. connectivity

- all functions other than the modularity are new
- in each type of function we can formulated it based on a random model

Modularity

Modularity is a quality function introduced by Newman and Girvan that quantifies the community structure by providing a value for every clustering of a given graph.

- *Newman MJ, Girvan M.* Finding and evaluating community structure in networks, *Physical Review E 2004, 69(026113).*

Modularity - Main Idea

Employ a random graph on the same vertex set that does not have any community structure and compare the edge densities of the clusters in the original graph and the random graph.

The modularity of a clustering $\mathcal{C}$ for some graph G, is defined by the following normalized sum of differences

$$Q_m(\mathcal{C}, G) := \frac{1}{2m} \sum_{\mathcal{C} \in \mathcal{C}} \sum_{i,j \in C} (a_{ij} - p_{ij})$$

- a_{ij} = number of edges between vertices i and j in G
- p_{ij} = is the *expected number* of edges between vertices i and j in the random graph

Modularity - Main Idea

Employ a random graph on the same vertex set that does not have any community structure and compare the edge densities of the clusters in the original graph and the random graph.

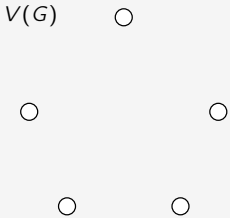
The modularity of a clustering $\mathcal{C}$ for some graph G, is defined by the following normalized sum of differences

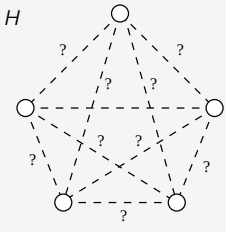
$$Q_m(\mathcal{C}, G) := \frac{1}{2m} \sum_{C \in \mathcal{C}} \sum_{i,j \in C} (a_{ij} - p_{ij})$$

- a_{ij} = number of edges between vertices i and j in G
- p_{ij} = is the *expected number* of edges between vertices i and j in the random graph

Modularity - Main Idea

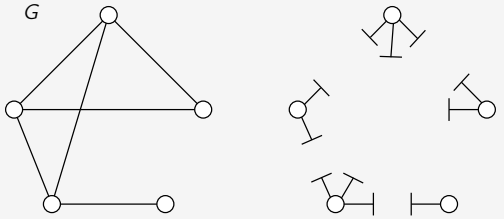
Question: How do we define the random graph (equivalently the p_{ij}) ?

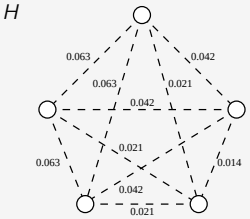




Modularity - the Random Graph

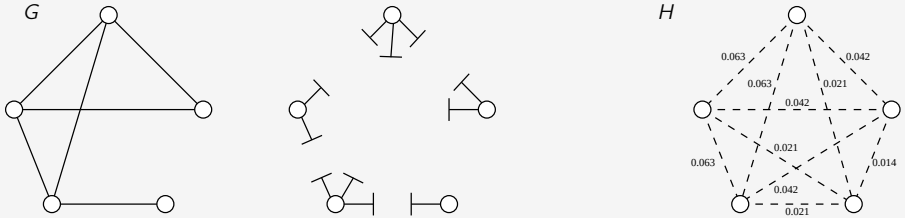
Random Graph Property: Keep the same degree distribution as in the original graph





Modularity - the Random Graph

Random Graph Property: Keep the same degree distribution as in the original graph

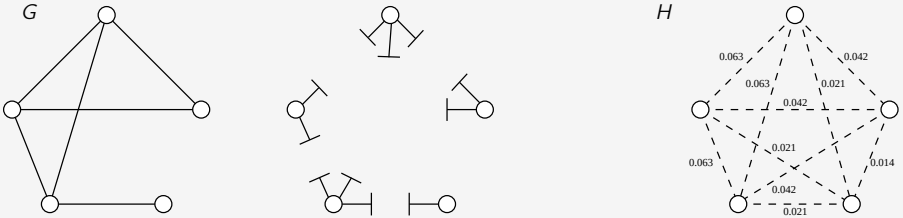


Random graph H will have $V(H) = V(G)$ and $E(G)$ defined by

$$Pr[(i,j) \in E(H)] \quad = \quad \frac{d_G(i)}{2m} \cdot \frac{d_G(j)}{2m}.$$

Modularity - the Random Graph

Random Graph Property: Keep the same degree distribution as in the original graph



Random graph H will have $V(H) = V(G)$ and $E(G)$ defined by

$$Pr[(i,j) \in E(H)] \quad = \quad \frac{d_G(i)}{2m} \cdot \frac{d_G(j)}{2m}.$$

$\Rightarrow$ expected number of edges between i and j then is

$$p_{ij} = 2m \times Pr[(i,j) \in E(H)] = \frac{d_G(i)d_G(j)}{2m}$$

Modularity - unweighted graphs

we thus have

$$Q_m(G, \mathcal{C}) = \frac{1}{2m} \sum_{C \in \mathcal{C}} \sum_{i,j \in C} \left(a_{ij} - \frac{d_G(i) d_G(j)}{2m} \right)$$

where a_{ij} is the number of edges between vertices i and j in G. Its is *straighforward* to show that

$$Q_m(G, \mathcal{C}) = \sum_{C \in \mathcal{C}} \left[\frac{m_C}{m} - \left(\frac{d_G(C)}{2m} \right)^2 \right] \tag{5}$$

where $m = |E(G)|$ and $m_C = |E(G[C])|$, and the terms

$$\frac{m_C}{m} \quad : \quad \text{fraction of edges within cluster } C$$

$$\left(\frac{d_G(C)}{2m} \right)^2 \quad : \quad \text{expected fraction of edges within cluster } C$$

Modularity - weighted graphs

Given a weight function $w : E(G) \to \mathbb{R}$ on the edges of a graph, we can define the **strength** of a vertex $i \in V(G)$ as

$$s_G(i) := \sum_{j \in V(G)} w(i,j).$$

We can then write for the modularity of a clustering $\mathcal{C}$ for some weighted graph G

$$Q_{m_w}(G, \mathcal{C}) = \frac{1}{2\sum_{e \in E(G)} w(e)} \sum_{C \in \mathcal{C}} \sum_{i,j \in C} \left(w(i,j) - \frac{s_G(i) s_G(j)}{2\sum_{e \in E(G)} w(e)} \right).$$

Modularity - directed graphs

If G directed let a_{ij} denote the number of **directed** edges from vertex i to vertex j while $d_G^+(i)$ and $d_G^-(i)$ be in-degree and out-degree of vertex i, respectively. We will therefore have

$$d_G^+(i) = \sum_j a_{ji}, \quad d_G^-(j) = \sum_i a_{ij}.$$

In order to generalize modularity for directed graphs, it is enough to construct a random directed graph without any community structure for where the expected in-degree and out-degree sequence will be the same as in G. The modularity of a clustering $\mathcal{C}$ in a directed graph G is given by the following

$$Q_{m_d}(G, \mathcal{C}) = \frac{1}{m} \sum_{C \in \mathcal{C}} \sum_{i,j \in C} \left(a_{ij} - \frac{d_G^-(i) d_G^+(j)}{m} \right).$$

Modularity - weighted directed graphs

Generalizing the strength of a vertex $i \in V(G)$ into **in-strength** and **out-strength** for a weighted directed graph G as follows,

$$s_G^-(i) = \sum_{j \in V(G)} w(i,j), \quad s_G^+(i) := \sum_{j \in V(G)} w(j,i), \tag{6}$$

we can combine the expressions for Q_{m_d} and Q_{m_w} to derive the an expression for modularity for weighted directed graphs

$$Q_{m_{w,d}}(G, \mathcal{C}) = \frac{1}{\sum_{i,j \in V(G)} (w(i,j) + w(j,i))} \sum_{C \in \mathcal{C}} \sum_{i,j \in C} \left(w(i,j) - \frac{s_G^-(i) s_G^+(j)}{\sum_{i,j \in V(G)} (w(i,j) + w(j,i))} \right).$$

Modularity maximization - IP Formulation

Define n^2 binary variables x_{ij} for each pair of nodes $i, j \in V(G)$ as

$$x_{ij} := \begin{cases} 1, & \text{if vertices } i \text{ and } j \text{ belong in the same cluster,} \\ 0, & \text{otherwise.} \end{cases}$$

Modularity maximization - IP Formulation

Define n^2 binary variables x_{ij} for each pair of nodes $i, j \in V(G)$ as

$$x_{ij} := \begin{cases} 1, & \text{if vertices } i \text{ and } j \text{ belong in the same cluster,} \\ 0, & \text{otherwise.} \end{cases}$$

This results in the following $\{0, 1\}$ program

$$\begin{aligned} \max \quad & \frac{1}{2m} \sum_{i,j \in V(G)} \left(a_{ij} - \frac{d_G(i) d_G(j)}{2m} \right) x_{ij} \\ \text{s.t.} \quad & x_{ii} = 1, \ \ \forall i \in V(G) \\ & x_{ij} = x_{ji}, \ \ \forall i, j \in V(G) \\ & x_{ij} + x_{jk} \leq 2x_{ik} + 1, \ \ \forall i, j, k \in V(G) \\ & x_{ij} \in 0, 1, \ \ \forall i, j \in V(G) \end{aligned}$$

Modularity properties

Theorem (Gevezes, Kehagias and Pitsoulis, 2013)

The modularity function is not:

- *monotone*
- *connected*
- *convex*
- *complementary*
- *resolution-limit free*

so it seems that modularity fails in almost all theoretical properties, but is the most widely used!

Anti-modularity

Based on the same random model as modularity, but instead of maximizing intra-cluster edge density it **minimizes extra-cluster edge density**.

$$Q_{anti-m}(G, \mathcal{C}) = - \sum_{\substack{C_1, C_2 \in \mathcal{C} \\ C_1 \neq C_2}} \left[\frac{m_{C_1 \leftrightarrow C_2}}{m} - \left(\frac{d_G(C_1) d_G(C_2)}{4m^2} \right) \right]$$

where $m_{C_1 \leftrightarrow C_2}$ denotes the number of edges with an end-vertex in cluster C_1 and an end-vertex in cluster C_2.

Anti-modularity

Based on the same random model as modularity, but instead of maximizing intra-cluster edge density it **minimizes extra-cluster edge density**.

$$Q_{anti-m}(G, \mathcal{C}) = - \sum_{\substack{C_1, C_2 \in \mathcal{C} \\ C_1 \neq C_2}} \left[\frac{m_{C_1 \leftrightarrow C_2}}{m} - \left(\frac{d_G(C_1) d_G(C_2)}{4m^2} \right) \right]$$

where $m_{C_1 \leftrightarrow C_2}$ denotes the number of edges with an end-vertex in cluster C_1 and an end-vertex in cluster C_2.

- similar behavior as modularity
- performs better in *unbalanced community structure*
- open problem: has not been examined yet w.r.t. properties

Components quality function

Definition

A graph is connected if for any $v, w \in V(G)$ there exists a $v - w$ path. The number of connected components of a graph G will be denoted by k_G.

The **components quality function** takes the value of 1 for clusterings which identify with the connected components of the graph and 0 elsewhere. It is defined as follows

$$Q_{coco} = \begin{cases} 1 & \text{if the members of } \mathcal{C} \text{ are the connected components of } G, \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

Components quality function

Definition

A graph is connected if for any $v, w \in V(G)$ there exists a $v - w$ path. The number of connected components of a graph G will be denoted by k_G.

The **components quality function** takes the value of 1 for clusterings which identify with the connected components of the graph and 0 elsewhere. It is defined as follows

$$Q_{coco} = \begin{cases} 1 & \text{if the members of } \mathcal{C} \text{ are the connected components of } G, \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

Theorem (Gevezes, Kehagias and Pitsoulis, 2013)

Q_{coco} is isomorphishm invariant, scale invariant, rich, connected, monotone, complementary and perfect.

Higher connectivity

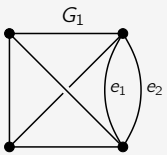
Definition (edge connectivity)

- For $k \in \mathbb{N}$ we say that a graph G is k-edge-connected, if $|E(G)| > k$ and $G \setminus Y$ is connected for any $Y \subseteq E(G)$ with $|Y| < k$.
- Equivalently G is k-edge-connected if k is the minimum number of edges that you can delete and make G disconnected or the trivial graph K_1.
- We will write $\alpha(G)$ for the edge connectivity number of a graph.

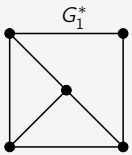
Higher connectivity

Definition (edge connectivity)

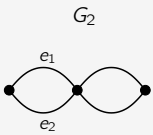
- For $k \in \mathbb{N}$ we say that a graph G is k-edge-connected, if $|E(G)| > k$ and $G \backslash Y$ is connected for any $Y \subseteq E(G)$ with $|Y| < k$.
- Equivalently G is k-edge-connected if k is the minimum number of edges that you can delete and make G disconnected or the trivial graph K_1.
- We will write $\alpha(G)$ for the edge connectivity number of a graph.



$\alpha(G_1) = 3$



$\alpha(G_1^*) = 2$



$\alpha(G_2) = 2$

Edge connectivity quality function

Using the same random graph H definition as in modularity we define the edge connectivity quality function as

$$Q_\alpha(G, \mathcal{C}) = \sum_{C \in \mathcal{C}} \left[\frac{\alpha(G[C])}{\alpha(G)} - \frac{mincut(H[C])}{mincut(H)} \right]$$

where $G[C]$ and $H[C]$ are the induced subgraphs of G and H by the set of vertices C, and

$$\frac{\alpha(G[C])}{\alpha(G)} \quad : \quad \text{relative edge connectivity of cluster } C$$

$$\frac{mincut(H[C])}{mincut(H)} \quad : \quad \text{expected edge connectivity of cluster } C$$

Edge connectivity quality function

Using the same random graph H definition as in modularity we define the edge connectivity quality function as

$$Q_\alpha(G, \mathcal{C}) = \sum_{C \in \mathcal{C}} \left[\frac{\alpha(G[C])}{\alpha(G)} - \frac{mincut(H[C])}{mincut(H)} \right]$$

where $G[C]$ and $H[C]$ are the induced subgraphs of G and H by the set of vertices C, and

$$\frac{\alpha(G[C])}{\alpha(G)} \quad : \quad \text{relative edge connectivity of cluster } C$$

$$\frac{mincut(H[C])}{mincut(H)} \quad : \quad \text{expected edge connectivity of cluster } C$$

- other variations using Tutte-connectivity and vertex connectivity
- computationally not attractive
- open problem: is it monotone, rich, etc. ?

Local density

These functions are based on the **densities** of intra-cluster and extra-cluster edges. We are given a graph G and a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$. Let

$$E_C \quad : \quad \text{the edges of } G \text{ with both end-vertices in cluster } C$$
$$E'_C \quad : \quad \text{the edges of } G \text{ with one end-vertex in cluster } C$$

Local density

These functions are based on the **densities** of intra-cluster and extra-cluster edges. We are given a graph G and a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$. Let

$$E_C \quad : \quad \text{the edges of } G \text{ with both end-vertices in cluster } C$$
$$E'_C \quad : \quad \text{the edges of } G \text{ with one end-vertex in cluster } C$$

The **local density** quality function is defined as

$$Q_{ld}(G, \mathcal{C}) := \frac{1}{2k} \sum_{C \in \mathcal{C}} \left[\frac{|E_C|}{|C| \cdot (|C| - 1)/2} + \left(1 - \frac{|E'_C|}{|C| \cdot |V(G) - C|} \right) \right]$$

Local density

These functions are based on the **densities** of intra-cluster and extra-cluster edges. We are given a graph G and a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$. Let

$$\begin{aligned} E_C &: \text{the edges of } G \text{ with both end-vertices in cluster } C \\ E'_C &: \text{the edges of } G \text{ with one end-vertex in cluster } C \end{aligned}$$

The **local density** quality function is defined as

$$Q_{ld}(G, \mathcal{C}) := \frac{1}{2k} \sum_{C \in \mathcal{C}} \left[\frac{|E_C|}{|C| \cdot (|C| - 1)/2} + \left(1 - \frac{|E'_C|}{|C| \cdot |V(G) - C|} \right) \right]$$

$$\begin{aligned} \frac{|E_C|}{|C| \cdot (|C| - 1)/2} &: \text{density of intra-cluster edges of cluster } C \\ \frac{|E'_C|}{|C| \cdot |V(G) - C|} &: \text{density of extra-cluster edges of cluster } C \\ Q_{ld}(G, \mathcal{C}) &: \text{average of cluster densities} \end{aligned}$$

Global density

The **global density** quality function as

$$Q_{gd}(G, \mathcal{C}) := \frac{1}{2}\left[\frac{\sum_{C\in\mathcal{C}} |E_C|}{\sum_{C\in\mathcal{C}} |C| \cdot (|C|-1)/2} + \left(1 - \frac{\sum_{C\in\mathcal{C}} |E'_C|}{\sum_{C\in\mathcal{C}} |C| \cdot |V(G) - C|}\right)\right]$$

where

$$\frac{\sum_{C\in\mathcal{C}} |E_C|}{\sum_{C\in\mathcal{C}} |C| \cdot (|C|-1)/2} \quad : \quad \text{density of } \textbf{all} \text{ intra-cluster edges}$$

$$\frac{\sum_{C\in\mathcal{C}} |E'_C|}{\sum_{C\in\mathcal{C}} |C| \cdot |V(G) - C|} \quad : \quad \text{density of } \textbf{all} \text{ extra-cluster edges}$$

$$Q_{gd}(G, \mathcal{C}) \quad : \quad \text{average of cluster densities}$$

Density based quality functions

Theorem (Gevezes, Kehagias and Pitsoulis, 2013)

Q_{ld} and Q_{gd} are isomorphishm invariant, scale invariant, monotone, complementary and perfect.

– Let graph G, its complement $\bar{G}$ and a clustering $\mathcal{C}$ of $V(G)$.
– Since the range of both functions Q_{ld} and Q_{gd} is $[0, 1]$ scaling will not be necessary.

We will first prove the statement for Q_{ld}

– For some $C \in \mathcal{C}$ let

$$m_C = |E_C| + |\bar{E}_C| \quad : \quad \text{number of possible edges with both end-vertices in } G[C] \tag{8}$$
$$m'_C = |E'_C| + |\bar{E}'_C| \quad : \quad \text{number of possible edges with one end-vertex in } G[C] \tag{9}$$

– It follows that

$$\frac{|E_C|}{|C| \cdot (|C|-1)/2} = \frac{|E_C|}{m_C} = 1 - \frac{|\bar{E}_C|}{m_C},$$
$$\frac{|E'_C|}{|C| \cdot |V(G) - C|} = \frac{|E'_C|}{m'_C} = 1 - \frac{|\bar{E}'_C|}{m'_C}.$$

– Letting

$$a_C = \frac{|E_C|}{m_C}, \bar{a}_C = \frac{|\bar{E}_C|}{m_C}, \tag{10}$$

and

$$e_C = \frac{|E'_C|}{m'_C}, \bar{e}_C = \frac{|\bar{E}'_C|}{m'_C}, \tag{11}$$

we have that

$$a_C = 1 - \bar{a}_C, \quad e_C = 1 - \bar{e}_C.$$

Substituting (10) and (11) in the expression for $Q_{ld}(G, \mathcal{C})$ we get

$$\begin{aligned} Q_{ld}(G, \mathcal{C}) &= \frac{1}{2k} \sum_{C \in \mathcal{C}} [a_C + (1 - e_C)] \\ &= \frac{1}{2k} \sum_{C \in \mathcal{C}} [(1 - \bar{a}_C) + 1 - (1 - \bar{e}_C)] \\ &= \frac{1}{2} - \frac{1}{2k} \sum_{C \in \mathcal{C}} (\bar{a}_C - \bar{e}_C) \\ &= 1 - Q_{ld}(\bar{G}, \mathcal{C}). \end{aligned}$$

For Q_{gd} we extend the analysis by summing up the values of (8) and (9)
– Let

$$m = \sum_{C \in \mathcal{C}} m_C, \quad m' = \sum_{C \in \mathcal{C}} m'_C$$

and

$$a = \frac{\sum_{C \in \mathcal{C}} |E_C|}{m}, \bar{a} = \frac{\sum_{C \in \mathcal{C}} |\bar{E}_C|}{m}, \tag{12}$$

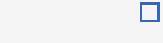
$$e = \frac{\sum_{C \in \mathcal{C}} |E'_C|}{m}, \bar{e} = \frac{\sum_{C \in \mathcal{C}} |\bar{E}'_C|}{m}, \tag{13}$$

while it follows that

$$a = 1 - \bar{a}, \quad e = 1 - \bar{e}.$$

–Substituting (12) and (13) in the expression for $Q_{gd}(G, \mathcal{C})$ we get

$$\begin{aligned} Q_{gd}(G, \mathcal{C}) &= \frac{1}{2}[a + 1 - e] \\ &= \frac{1}{2}[(1 - \bar{a}) - (1 - \bar{e})] + \frac{1}{2} \\ &= 1 - Q_{gd}(\bar{G}, \mathcal{C}) \end{aligned}$$



Preliminaries

Definition

Given a graph $G(V, E)$ we define the following:

- $v - w$ **walk** is an alternating sequence of vertices and edges, begining with vertex v and ending with vertex w
- **trail** is a walk with distinct edges
- **path** is a walk with distinct vertices
- **shortest path** between two vertices is a path with the smallest number of edges (may not be unique)

Preliminaries

Definition (adjacency matrix)

The adjacency matrix of a graph $G(V, E)$, is a $n \times n$ matrix A_G defined as

$$A_G(i,j) = \begin{cases} 1 & \text{if vertex } v_j \text{ is adjacent to vertex } v_i, \\ 0 & \text{otherwise.} \end{cases}$$

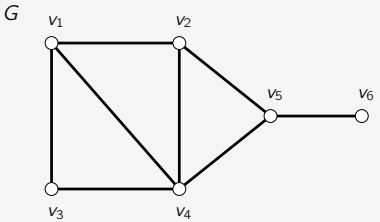
Preliminaries

Definition (adjacency matrix)

The adjacency matrix of a graph $G(V, E)$, is a $n \times n$ matrix A_G defined as

$$A_G(i,j) = \begin{cases} 1 & \text{if vertex } v_j \text{ is adjacent to vertex } v_i, \\ 0 & \text{otherwise.} \end{cases}$$

$$A_G = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$



Preliminaries

It is well know that by taking the powers of the adjacency matrix A_G^k we have

$$A_G^k = \text{is the number of } v_i - v_j \text{ walks}$$

So we have for our example

$$A_G^2 = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} 3 & 1 & 1 & 2 & 2 & 0 \\ 1 & 3 & 2 & 2 & 1 & 1 \\ 1 & 2 & 2 & 1 & 1 & 0 \\ 2 & 2 & 1 & 4 & 1 & 1 \\ 2 & 1 & 1 & 1 & 3 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \end{array}$$

and the diagonal of A_G^2 corresponds to the degrees of the vertices in G (if the graph is simple).

Distance matrix

Definition

The **distance matrix** of a graph $G(V, E)$ is a $n \times n$ matrix D_V

$$D_V = \min\{k : A^k_G(i, j) \neq 0\}$$

and it contains the distances between pairs of vertices. If only a subset of the vertices $U \subseteq V$ is used we write $D_U(i, j)$ to denote the distance of vertices v_i and v_j in $G[U]$.

- the **diameter** of $G(V, E)$ is $diam(G) = \max\{D_V(i, j) : \forall i, j \in V\}$
- for $C \subseteq W \subseteq V$ we denote $D_W(C) = \sum_{i,j \in C} D_W(i, j)$.
- so for $C \subseteq V$ by $D_V(C)$ we mean the sum of distances of vertex pairs in C using **all** vertices of the graph, while
- by $D_C(C)$ we mean the sum of distances of vertex pairs in C in the subgraph $G[C]$.

Paths matrix

Definition

The **paths matrix** of a graph $G(V, E)$ is defined as an $n \times n$ matrix P_V

$$P_V(i,j) = A^l_G(i,j) \text{ where } l = \min\{k : A^k_G(i,j) \neq 0\}$$

and it contains number of different shortest paths between pairs of vertices. If only a subset of the vertices $U \subseteq V$ is used we write $P_U(i,j)$ to denote the number of shortest paths between vertices v_i and v_j in $G[U]$.

Paths matrix

Definition

The **paths matrix** of a graph $G(V, E)$ is defined as an $n \times n$ matrix P_V

$$P_V(i,j) = A^l_G(i,j) \text{ where } l = \min\{k : A^k_G(i,j) \neq 0\}$$

and it contains number of different shortest paths between pairs of vertices. If only a subset of the vertices $U \subseteq V$ is used we write $P_U(i,j)$ to denote the number of shortest paths between vertices v_i and v_j in $G[U]$.

So the distance and paths matrices for our example:

$$D_V = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 & 2 & 3 \\ 1 & 0 & 2 & 1 & 1 & 2 \\ 1 & 2 & 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 0 & 1 & 2 \\ 2 & 1 & 2 & 1 & 0 & 1 \\ 3 & 2 & 3 & 2 & 1 & 0 \end{bmatrix} \end{array} \quad P_V = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & 2 & 2 \\ 1 & 1 & 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Generalized degree

Definition (generalized degree)

The k-**degree** of a vertex v denoted by $d_k(v)$ is the number of shortest paths of length k that this vertex participates as a source vertex.

Generalized degree

Definition (generalized degree)

The ***k*-degree** of a vertex v denoted by $d_k(v)$ is the number of shortest paths of length k that this vertex participates as a source vertex.

- we have $d_k(v) = \sum\{P_V(v,i) : D_V(v,i) = k\}$
- given a graph $G(V,E)$ the total number of shortest paths of length $k \leq diam(G)$ is

$$m_k(G) = \frac{1}{2} \sum_{v \in V(G)} d_k(v)$$

- for $k = 1$ we have the degree of a vertex and the familiar $m_1(G) = |E(G)|$

Distance quality function

we are now ready to formulate the distance quality function using a random graph

- the probability that vertices i, j are *joined* by a path of length k

$$Pr[i, j, k] = \frac{d_k(i)}{2m_k(G)} \frac{d_k(j)}{2m_k(G)}$$

- expected distance between vertices i, j

$$\overline{D_V(i,j)} = \sum_{k=1}^{diam(G)} kPr[i, j, k]$$

- sum of expected pairwise distances in cluster C

$$\overline{D_V(C)} = \frac{1}{2} \sum_{i,j \in C} \overline{D_V(i,j)}$$

- given a cluster of vertices C we want to have the smallest sum of pairwise distances w.r.t. a random model

$$Q_d(G, \mathcal{C}) = \sum_{C \in \mathcal{C}} \left(\overline{D_V(C)} - D_V(C) \right)$$

Outline of the talk

Resolution limit: underestimation of clusters

- *Resolution limit in community detection*, S. Fortunato and M. Barthelemy , Proceedings of the National Academy of Sciences, Vol. 104, pp. 36-41 (2007).

We have n cliques K_m

$$Q_m = 1 - \frac{2}{m(m-2)+2} - \frac{1}{n}$$

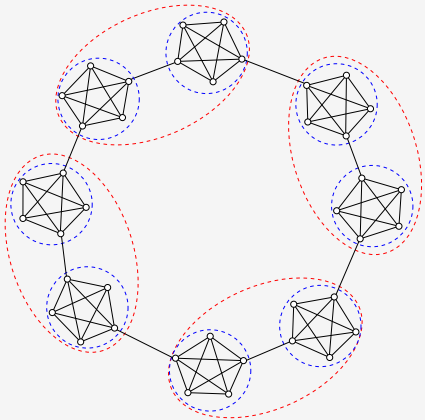
$$Q_m = 1 - \frac{1}{m(m-2)+2} - \frac{2}{n}$$

so $Q_m > Q_m$ only if

$$m(m-1)+2 > n$$

So for $m = 5, n = 30$

$$Q_m = 0.876 < 0.888 = Q_m$$



Overestimation of clusters

For a clustering $\mathcal{C} = \{C_1, \ldots, C_K\}$ we can decompose modularity

$$Q_m(\mathcal{C}, G) = \underbrace{\sum_{C \in \mathcal{C}} \frac{m_C}{m}}_{Q_f(\mathcal{C}, G)} - \underbrace{\sum_{C \in \mathcal{C}} \left(\frac{d_G(C)}{2m} \right)^2}_{Q_0(\mathcal{C}, G)}$$

Overestimation of clusters

For a clustering $\mathcal{C} = \{C_1, \ldots, C_K\}$ we can decompose modularity

$$Q_m(\mathcal{C}, G) = \underbrace{\sum_{C \in \mathcal{C}} \frac{m_C}{m}}_{Q_f(\mathcal{C}, G)} - \underbrace{\sum_{C \in \mathcal{C}} \left(\frac{d_G(C)}{2m} \right)^2}_{Q_0(\mathcal{C}, G)}$$

- $Q_f(\mathcal{C}, G)$ gets maximized at $K = 1$.

Overestimation of clusters

For a clustering $\mathcal{C} = \{C_1, \ldots, C_K\}$ we can decompose modularity

$$Q_m(\mathcal{C}, G) = \underbrace{\sum_{C \in \mathcal{C}} \frac{m_C}{m}}_{Q_f(\mathcal{C}, G)} - \underbrace{\sum_{C \in \mathcal{C}} \left(\frac{d_G(C)}{2m} \right)^2}_{Q_0(\mathcal{C}, G)}$$

- $Q_f(\mathcal{C}, G)$ gets maximized at $K = 1$.
- $Q_0(\mathcal{C}, G)$ gets minimized at $K = n$.

Overestimation of clusters

For a clustering $\mathcal{C} = \{C_1, \ldots, C_K\}$ we can decompose modularity

$$Q_m(\mathcal{C}, G) = \underbrace{\sum_{C \in \mathcal{C}} \frac{m_C}{m}}_{Q_f(\mathcal{C}, G)} - \underbrace{\sum_{C \in \mathcal{C}} \left(\frac{d_G(C)}{2m} \right)^2}_{Q_0(\mathcal{C}, G)}$$

- $Q_f(\mathcal{C}, G)$ gets maximized at $K = 1$.
- $Q_0(\mathcal{C}, G)$ gets minimized at $K = n$.
- Q_f term favors clusterings with few extra-cluster edges.

Overestimation of clusters

For a clustering $\mathcal{C} = \{C_1, \ldots, C_K\}$ we can decompose modularity

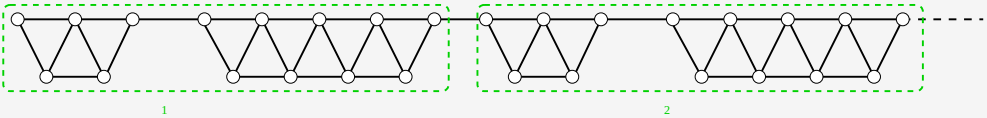
$$Q_m(\mathcal{C}, G) = \underbrace{\sum_{C \in \mathcal{C}} \frac{m_C}{m}}_{Q_f(\mathcal{C}, G)} - \underbrace{\sum_{C \in \mathcal{C}} \left(\frac{d_G(C)}{2m} \right)^2}_{Q_0(\mathcal{C}, G)}$$

- $Q_f(\mathcal{C}, G)$ gets maximized at $K = 1$.
- $Q_0(\mathcal{C}, G)$ gets minimized at $K = n$.
- Q_f term favors clusterings with few extra-cluster edges.
- Q_0 term favors clusterings with *balanced* clusters.

Overestimation of clusters

Consider the following family of graphs

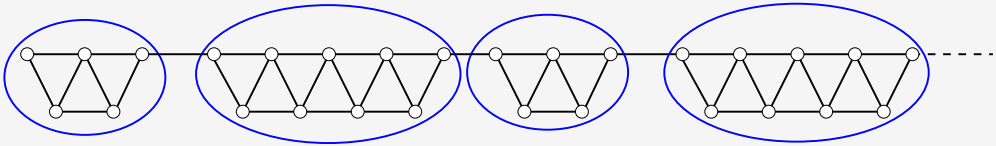
- family H_{k,n_1,n_2} of graphs



Overestimation of clusters

Consider the following family of graphs

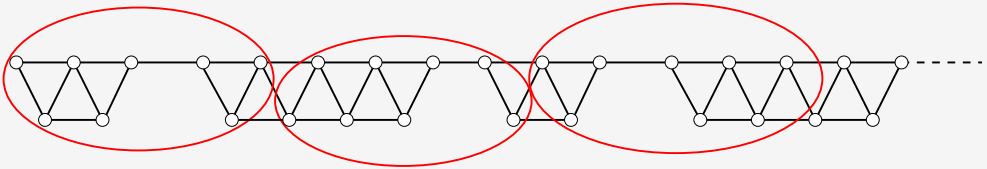
- family H_{k,n_1,n_2} of graphs
- **natural** clustering $\mathcal{C}_N$



Overestimation of clusters

Consider the following family of graphs

- family H_{k,n_1,n_2} of graphs
- **natural** clustering $\mathcal{C}_N$
- **balanced** clustering $\mathcal{C}_B(J)$ for $J = 8$



Overestimation of clusters

Theorem (Kehagias and Pitsoulis (EPJ 2013))

For every $k \in \mathbb{N}$ *and* $\epsilon \in (0, \frac{1}{2k})$ *there exist* n_1, n_2, J *such that*

$$Q(\mathcal{C}_N, H_{k,n_1,n_2}) < 1 - \epsilon < Q(\mathcal{C}_B(J), H_{k,n_1,n_2})$$

and

$$J(\mathcal{C}_N, \mathcal{C}_B(J)) < \epsilon.$$

Overestimation of clusters

Theorem (Kehagias and Pitsoulis (EPJ 2013))

For every $k \in \mathbb{N}$ *and* $\epsilon \in (0, \frac{1}{2k})$ *there exist* n_1, n_2, J *such that*

$$Q(\mathcal{C}_N, H_{k,n_1,n_2}) < 1 - \epsilon < Q(\mathcal{C}_B(J), H_{k,n_1,n_2})$$

and

$$J(\mathcal{C}_N, \mathcal{C}_B(J)) < \epsilon.$$

- $J(\mathcal{C}_1, \mathcal{C}_2) \in [0, 1]$ is the Jaccard similarity coefficient.

Overestimation of clusters

Theorem (Kehagias and Pitsoulis (EPJ 2013))

For every $k \in \mathbb{N}$ *and* $\epsilon \in (0, \frac{1}{2k})$ *there exist* n_1, n_2, J *such that*

$$Q(\mathcal{C}_N, H_{k,n_1,n_2}) < 1 - \epsilon < Q(\mathcal{C}_B(J), H_{k,n_1,n_2})$$

and

$$J(\mathcal{C}_N, \mathcal{C}_B(J)) < \epsilon.$$

- $J(\mathcal{C}_1, \mathcal{C}_2) \in [0, 1]$ is the Jaccard similarity coefficient.
- $\Rightarrow$ natural clustering does not have maximum modularity.

Overestimation of clusters

Theorem (Kehagias and Pitsoulis (EPJ 2013))

For every $k \in \mathbb{N}$ *and* $\epsilon \in (0, \frac{1}{2k})$ *there exist* n_1, n_2, J *such that*

$$Q(\mathcal{C}_N, H_{k,n_1,n_2}) < 1 - \epsilon < Q(\mathcal{C}_B(J), H_{k,n_1,n_2})$$

and

$$J(\mathcal{C}_N, \mathcal{C}_B(J)) < \epsilon.$$

- $J(\mathcal{C}_1, \mathcal{C}_2) \in [0, 1]$ is the Jaccard similarity coefficient.
- $\Rightarrow$ natural clustering does not have maximum modularity.
- $\Rightarrow$ balanced "bad" clustering can achieve an almost maximum modularity.

Overestimation of clusters

Theorem (Kehagias and Pitsoulis (EPJ 2013))

For every $k \in \mathbb{N}$ *and* $\epsilon \in (0, \frac{1}{2k})$ *there exist* n_1, n_2, J *such that*

$$Q(\mathcal{C}_N, H_{k,n_1,n_2}) < 1 - \epsilon < Q(\mathcal{C}_B(J), H_{k,n_1,n_2})$$

and

$$J(\mathcal{C}_N, \mathcal{C}_B(J)) < \epsilon.$$

- $J(\mathcal{C}_1, \mathcal{C}_2) \in [0, 1]$ is the Jaccard similarity coefficient.
- $\Rightarrow$ natural clustering does not have maximum modularity.
- $\Rightarrow$ balanced "bad" clustering can achieve an almost maximum modularity.
- $\Rightarrow$ natural clustering can be arbitrarily different than balanced clustering.

Overestimation of clusters

Theorem (Kehagias and Pitsoulis (EPJ 2013))

For every $k \in \mathbb{N}$ *and* $\epsilon \in (0, \frac{1}{2k})$ *there exist* n_1, n_2, J *such that*

$$Q(\mathcal{C}_N, H_{k,n_1,n_2}) < 1 - \epsilon < Q(\mathcal{C}_B(J), H_{k,n_1,n_2})$$

and

$$J(\mathcal{C}_N, \mathcal{C}_B(J)) < \epsilon.$$

- $J(\mathcal{C}_1, \mathcal{C}_2) \in [0, 1]$ is the Jaccard similarity coefficient.
- $\Rightarrow$ natural clustering does not have maximum modularity.
- $\Rightarrow$ balanced "bad" clustering can achieve an almost maximum modularity.
- $\Rightarrow$ natural clustering can be arbitrarily different than balanced clustering.
- $\Rightarrow$ modularity maximization can **overestimate** the number of clusters.

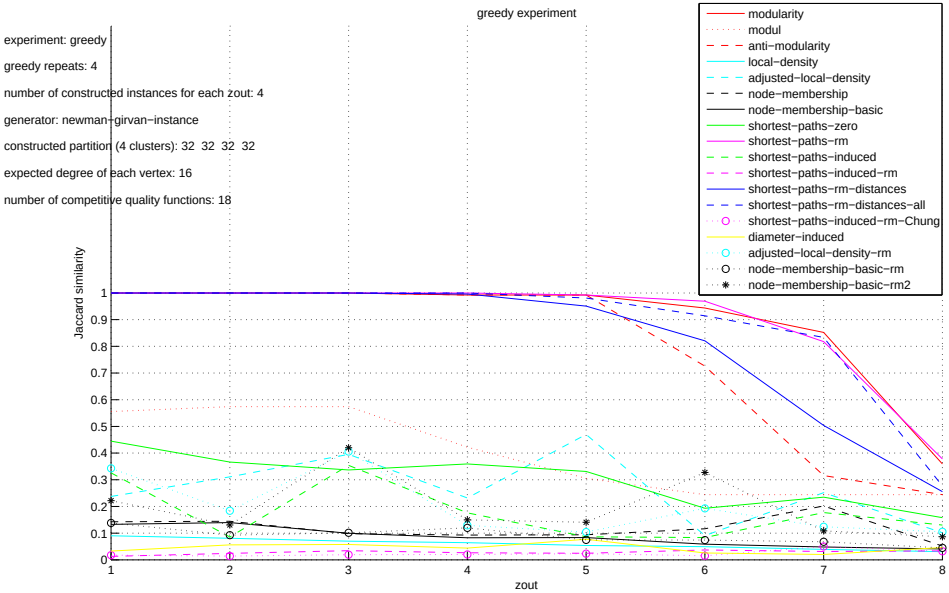
Outline of the talk

Girvan-Newman artificial graphs

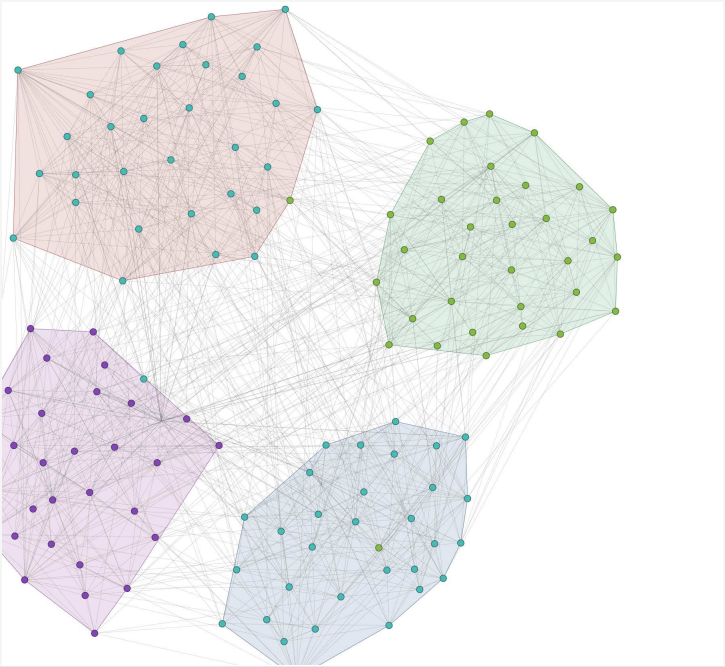
- preliminary results with the GN graphs
- $n = 128$, 4 communities with 32 vertices each
- expected degree of each vertex $= 16$
- p_{in}, p_{out}: probabilities for an intra-cluster and extra-cluster edge respectively
- more tests with benchmark instances with heterogeneous cluster sizes and degree distributions

  A. Lancichinetti, S. Fortunato and F. Radicchi (2008). "Benchmark graphs for testing community detection algorithms". Phys. Rev. E 78 (4): 046110.

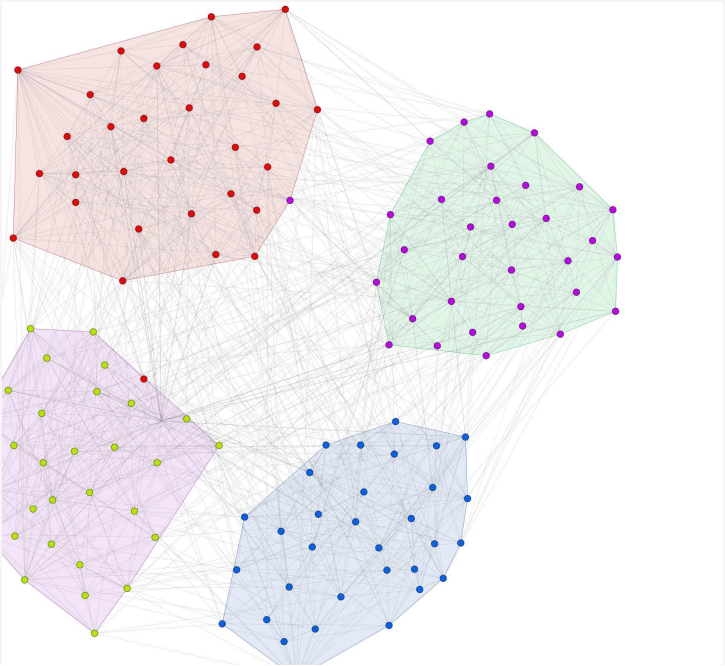
{32, 32, 32, 32}


greedy experiment
experiment: greedy
greedy repeats: 4
number of constructed instances for each zout: 4
generator: newman−girvan−instance
constructed partition (4 clusters): 32 32 32 32
expected degree of each vertex: 16
number of competitive quality functions: 18
modularity
modul
anti−modularity
local−density
adjusted−local−density
node−membership
node−membership−basic
shortest−paths−zero
shortest−paths−rm
shortest−paths−induced
shortest−paths−induced−rm
shortest−paths−rm−distances
shortest−paths−rm−distances−all
shortest−paths−induced−rm−Chung
diameter−induced
adjusted−local−density−rm
node−membership−basic−rm
node−membership−basic−rm2
Jaccard similarity
zout
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1
2
3
4
5
6
7
8

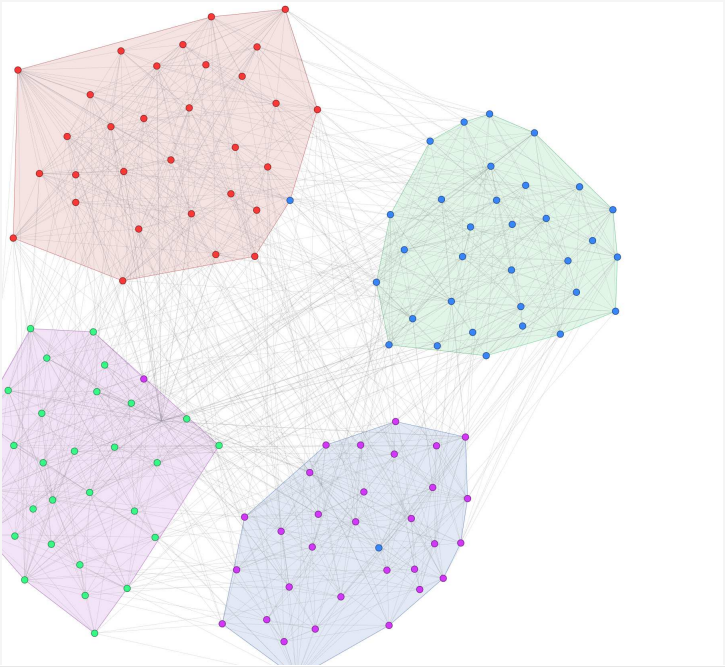
$\{32, 32, 32, 32\}$, $z_{out} = 6$, antimodularity



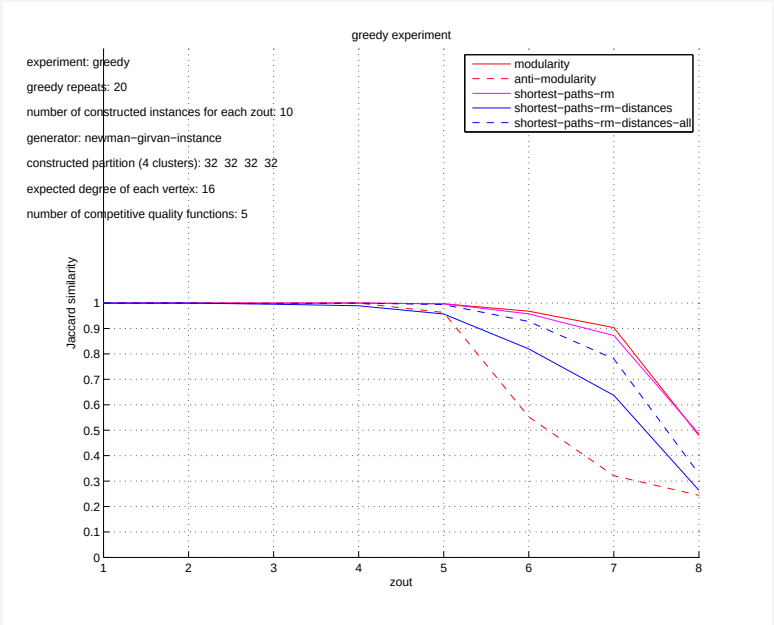
$\{32, 32, 32, 32\}$, $z_{out} = 6$ modularity



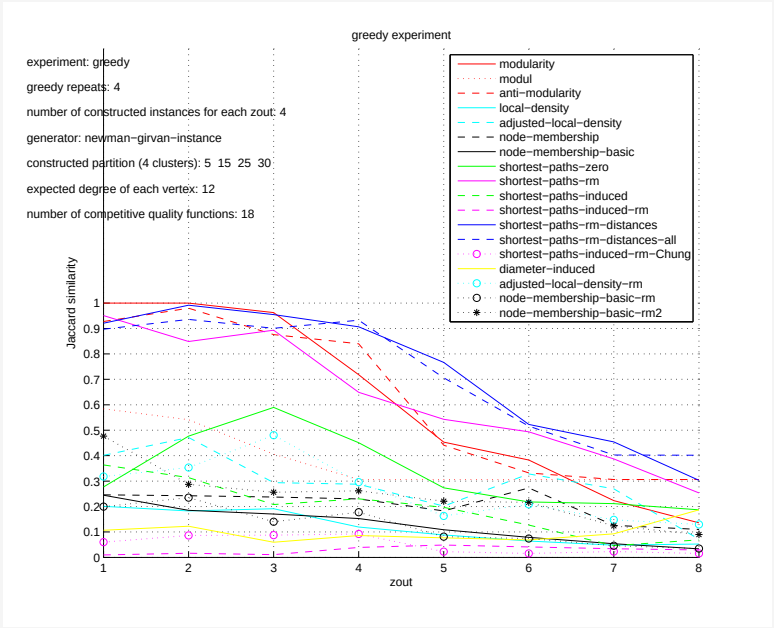
$\{32, 32, 32, 32\}$, $z_{out} = 6$ distance



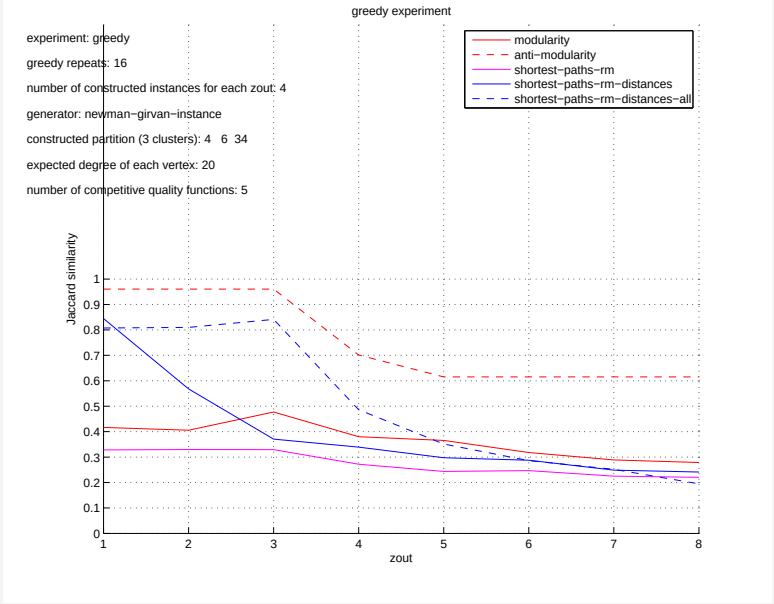
{32, 32, 32, 32}


greedy experiment
experiment: greedy
greedy repeats: 20
number of constructed instances for each zout: 10
generator: newman−girvan−instance
constructed partition (4 clusters): 32 32 32 32
expected degree of each vertex: 16
number of competitive quality functions: 5
modularity
anti−modularity
shortest−paths−rm
shortest−paths−rm−distances
shortest−paths−rm−distances−all
Jaccard similarity
zout
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1
2
3
4
5
6
7
8

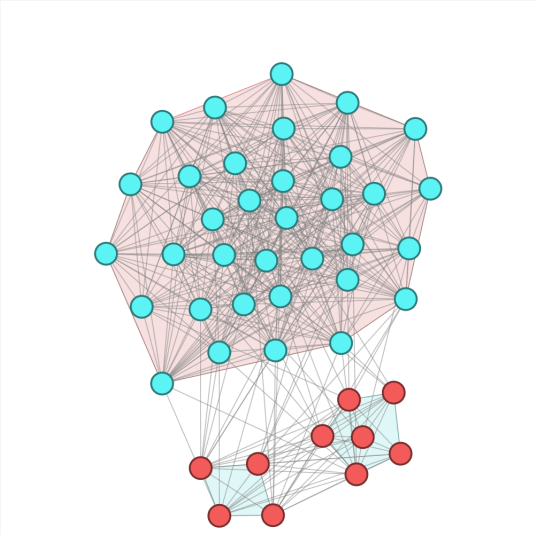
$\{5, 15, 25, 30\}$


greedy experiment
experiment: greedy
greedy repeats: 4
number of constructed instances for each zout: 4
generator: newman−girvan−instance
constructed partition (4 clusters): 5 15 25 30
expected degree of each vertex: 12
number of competitive quality functions: 18
modularity
modul
anti−modularity
local−density
adjusted−local−density
node−membership
node−membership−basic
shortest−paths−zero
shortest−paths−rm
shortest−paths−induced
shortest−paths−induced−rm
shortest−paths−rm−distances
shortest−paths−rm−distances−all
shortest−paths−induced−rm−Chung
diameter−induced
adjusted−local−density−rm
node−membership−basic−rm
node−membership−basic−rm2
Jaccard similarity
zout
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1
2
3
4
5
6
7
8

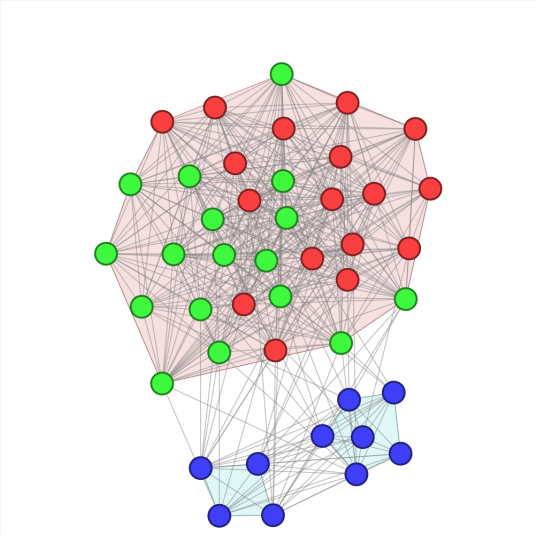
$\{4, 6, 34\}$


greedy experiment
experiment: greedy
greedy repeats: 16
number of constructed instances for each zout: 4
generator: newman−girvan−instance
constructed partition (3 clusters): 4 6 34
expected degree of each vertex: 20
number of competitive quality functions: 5
modularity
anti−modularity
shortest−paths−rm
shortest−paths−rm−distances
shortest−paths−rm−distances−all
Jaccard similarity
zout
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1
2
3
4
5
6
7
8

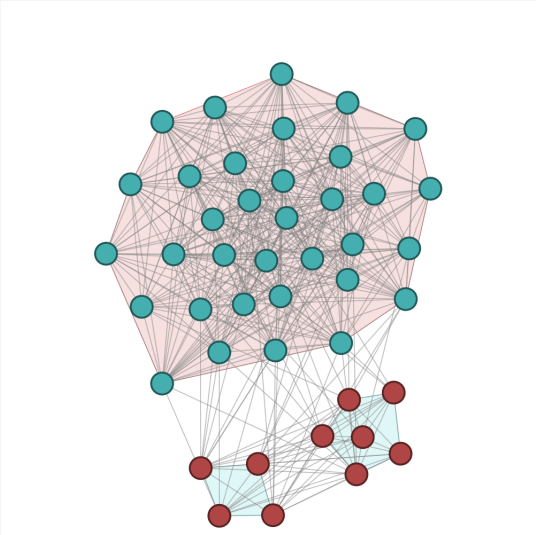
$\{4, 6, 34\}$ antimodularity



$\{4, 6, 34\}$ modularity



$\{4, 6, 34\}$ distance



resolution limit

bash bash

```
5 -  30:  modularity      :       0.8758 -->       0.8879 [     -0.0121]
          anti modularity :       0.3924 -->       0.4212 [     -0.0288]
          modul           :       0.3924 -->       0.4212 [     -0.0288]
          NM basic rm     :      57.8000 -->      29.3000 [     28.5000]   --- good ---
          NM              :       0.8248 -->       0.4486 [      0.3763]   --- good ---
          NM basic rm2    :       1.9267 -->       1.9533 [     -0.0267]
          LD adjstd rm    :       0.0165 -->       0.0154 [      0.0011]   --- good ---
          LD              :       0.0165 -->       0.0154 [      0.0011]   --- good ---
          SP rm dist all  :    4478.2019 -->    9514.2970 [  -5036.0951]
          SP rm           :    1592.8626 -->    2606.7470 [  -1013.8844]
          SP rm dist      :    4483.3091 -->    9523.0786 [  -5039.7694]
          SP indcd rm dist:     150.0000 -->    -306.7966 [    456.7966]   --- good ---
          SP indcd rm Chun:       2.1658 -->      -0.1583 [      2.3241]   --- good ---
          SP zero         :       2.1658 -->      -0.1583 [      2.3241]   --- good ---
          SP inf          :       2.1658 -->      -0.1583 [      2.3241]   --- good ---
          SP induced      :    -120.0000 -->    -255.0000 [    135.0000]   --- good ---

5 -  32:  modularity      :       0.8778 -->       0.8920 [     -0.0142]
          anti modularity :       0.3935 -->       0.4233 [     -0.0298]
          modul           :       0.3935 -->       0.4233 [     -0.0298]
          NM basic rm     :      61.8000 -->      31.4000 [     30.4000]   --- good ---
          NM              :       0.8232 -->       0.4467 [      0.3766]   --- good ---
          NM basic rm2    :       1.9312 -->       1.9625 [     -0.0313]
          LD adjstd rm    :       0.0155 -->       0.0144 [      0.0010]   --- good ---
          LD              :       0.0155 -->       0.0144 [      0.0010]   --- good ---
          SP rm dist all  :    5097.1651 -->   10869.6378 [  -5772.4727]
          SP rm           :    1732.4676 -->    2855.6838 [  -1123.2161]
          SP rm dist      :    5102.6855 -->   10879.1751 [  -5776.4895]
          SP indcd rm dist:     160.0000 -->    -327.2497 [    487.2497]   --- good ---
          SP indcd rm Chun:       2.3102 -->      -0.1689 [      2.4790]   --- good ---
          SP zero         :       2.3102 -->      -0.1689 [      2.4790]   --- good ---
          SP inf          :       2.3102 -->      -0.1689 [      2.4790]   --- good ---
          SP induced      :    -128.0000 -->    -272.0000 [    144.0000]   --- good ---
```

Outline of the talk

Probability that a GN-graph meets the community criteria

Definition (Community in the Strong Sense)

Given a graph $G(V, E)$ some $C \subseteq V(G)$ is a community in the **strong sense** if

$$d_{in}(v) > d_{out}(v), \quad \forall v \in C,$$

where $d_{in}(v)$ and $d_{out}(v)$ are the incident intra-cluster and extra-cluster edges respectively.

Probability that a GN-graph meets the community criteria

Definition (Community in the Strong Sense)

Given a graph $G(V, E)$ some $C \subseteq V(G)$ is a community in the **strong sense** if

$$d_{in}(v) > d_{out}(v), \quad \forall v \in C,$$

where $d_{in}(v)$ and $d_{out}(v)$ are the incident intra-cluster and extra-cluster edges respectively.

Consider a GN graph with no fixed expected degree, k clusters each with size n and

$$\begin{aligned} p_{in} &: \text{probability of intra-cluster edge} \\ p_{out} &: \text{probability of extra-cluster edge} \end{aligned}$$

Probability that a GN-graph meets the community criteria

Then we have the following:

- probability that a vertex is incident to m_{in} intra-cluster edges

$$\pi^{+}(m_i) = \binom{n}{m_i} p_{in}^{m_i} (1 - p_{in})^{n - m_i}$$

Probability that a GN-graph meets the community criteria

Then we have the following:

- probability that a vertex is incident to m_{in} intra-cluster edges

$$\pi^{+}(m_i) = \binom{n}{m_i} p_{in}^{m_i}(1-p_{in})^{n-m_i}$$

- probability that a vertex is incident to m_{out} extra-cluster edges

$$\pi^{-}(m_o) = \binom{n(k-1)}{m_o} p_{out}^{m_o}(1-p_{out})^{n(k-1)-m_o}$$

Probability that a GN-graph meets the community criteria

Then we have the following:

- probability that a vertex is incident to m_{in} intra-cluster edges

$$\pi^{+}(m_i) = \binom{n}{m_i} p_{in}^{m_i}(1 - p_{in})^{n-m_i}$$

- probability that a vertex is incident to m_{out} extra-cluster edges

$$\pi^{-}(m_o) = \binom{n(k-1)}{m_o} p_{out}^{m_o}(1 - p_{out})^{n(k-1)-m_o}$$

- probability that a vertex is incident to m_{in} intra-cluster and m_{out} extra-cluster edges

$$\pi(m_i, m_o) = \pi^{+}(m_i)\pi^{-}(m_o)$$

Probability that a GN-graph meets the community criteria

Then we have the following:

- probability that a vertex is incident to m_{in} intra-cluster edges

$$\pi^+(m_i) = \binom{n}{m_i} p_{in}^{m_i}(1-p_{in})^{n-m_i}$$

- probability that a vertex is incident to m_{out} extra-cluster edges

$$\pi^-(m_o) = \binom{n(k-1)}{m_o} p_{out}^{m_o}(1-p_{out})^{n(k-1)-m_o}$$

- probability that a vertex is incident to m_{in} intra-cluster and m_{out} extra-cluster edges

$$\pi(m_i, m_o) = \pi^+(m_i)\pi^-(m_o)$$

- probability that a vertex satisfies the strong condition, assuming independence between the events of having different degrees

$$Pr[i \text{ is strong}] = \sum_{m_o < m_i} \pi(m_i, m_o)$$

Probability that a GN-graph meets the community criteria

- probability that a cluster with n vertices satisfies the strong condition

$$Pr[C \text{ is strong}] = (Pr[i \text{ is strong}])^n$$

- probability that a clustering with k clusters of size n satisfies the strong condition

$$(Pr[i \text{ is strong}])^{nk} = \left(\sum_{m_o < m_i} \pi(m_i, m_o) \right)^{nk}$$

Thank You!