Graph Mining & Community Detection
An Introduction to Social Networks Data Analysis

Etienne Cuvelier[1]

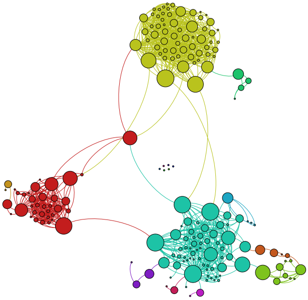
[1]Ecole Centrale Paris

eBISS 2011,
07/07/2011

Graph Mining and SNDA. Why?

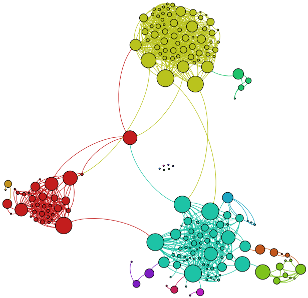
Graph Mining and SNDA. Why?

My Facebook's friends set reduced in 3 main communities.



Graph Mining and SNDA. Why?

My Facebook's friends set reduced in 3 main communities.



What you don't tell, your network can tell it for you (MIT's Gaydar experience).

A talk with a minimum of formulas?

Stephen Hawking

My editor told me that each equation I included in the book would halve sales.

"A Brief History of Time"

A talk with a minimum of formulas?

Stephen Hawking

My editor told me that each equation I included in the book would halve sales.

"A Brief History of Time"

A talk with a minimum of formulas?

Stephen Hawking

My editor told me that each equation I included in the book would halve sales.

"A Brief History of Time"

But...min $>$ 0 ! (Argh! A first formula!)

Outline

Plan

The Graph Theory start

How to walk through the city that would cross each bridge once and only once ? (Typically mathematician's game).

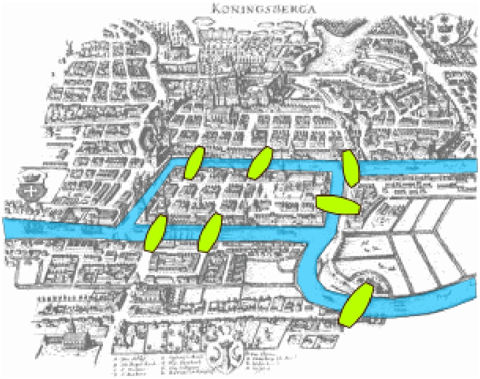


Figure: source: wikipedia

The Graph Theory start

How to walk through the city that would cross each bridge once and only once ? (Typically mathematician's game).

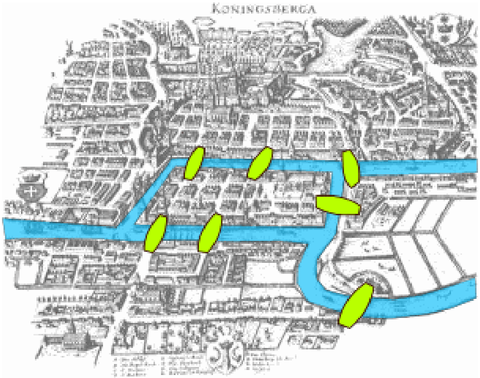


Figure: source: wikipedia

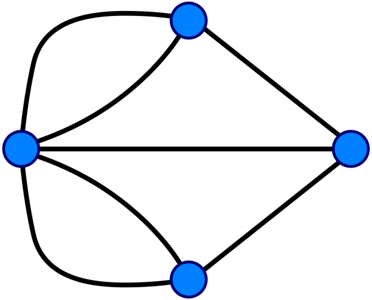


Figure: source: wikipedia

The Graph Theory start

How to walk through the city that would cross each bridge once and only once ? (Typically mathematician's game).

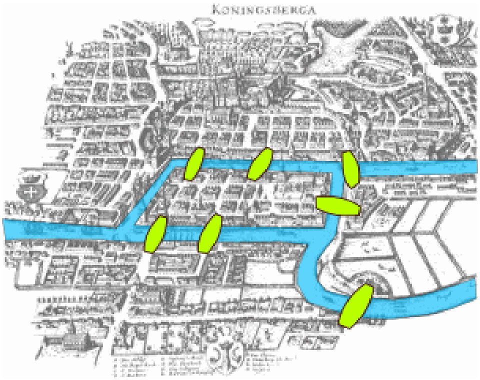


Figure: source: wikipedia

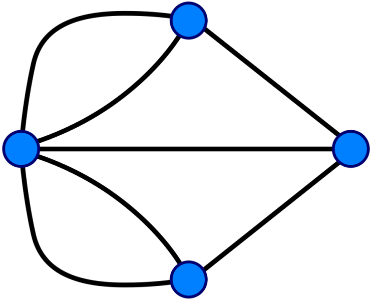


Figure: source: wikipedia

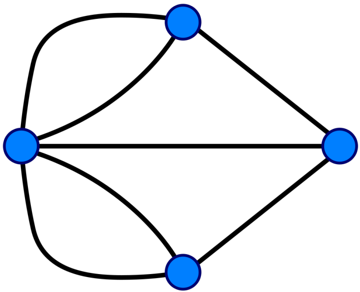
Using graphs Euler proved that the problem has no solution.

The Graph Theory Paradigm

A graph is denoted $G = (V, E)$ with

- V the set of vertices,
- E the set of edges.

$\{v, w\}$ is the edge connecting vertices v and w.



$$A = \left\{ \begin{array}{ccc} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n} \end{array} \right\}$$

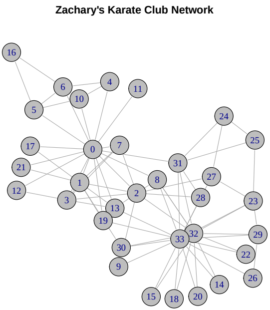
where

$$a_{i,j} = \begin{cases} 1 & \text{if } v_i \text{ and } v_j \text{ are connected,} \\ 0 & \textit{else.} \end{cases}$$

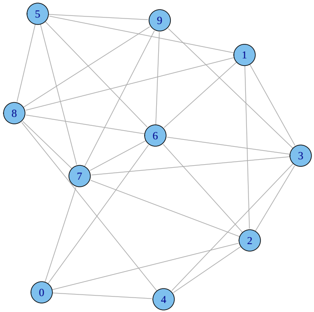
If G is a weighted graph, then $a_{i,j} = \omega(v_i, v_j)$ and then $A = W = (w_{i,j}) = (\omega(v_i, v_j))$.

The Graph Theory Goes Social

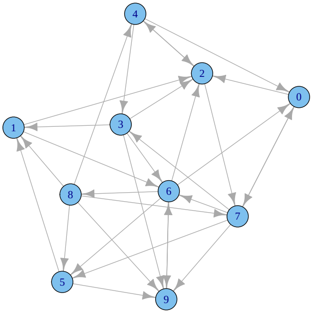
- Moreno (1933) 1st to use points and lines for social configurations,
- Cartwright and Harary (1956) link with the graph theory,
- individuals are represented using points, called *nodes* or *vertices*,
- and social relationships are represented using lines.



Directed or not?



An undirected graph (ex.: Facebook): $\{v, w\}$



A directed graph (ex.: Twitter): (v, w).

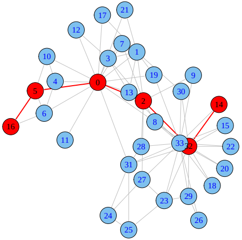
Graph Theory Basic Notions

- $|V| = n$ is called the *order* the graph.
- $|E| = m$ is called the *size* of the graph.
- If $|E| = n(n-1)/2$, i.e. (any pair of vertices are connected), the graph *complete*.
- v and u are *neighbours* if connected by an edge.
- The neighbourhood of a node v is denoted $\Gamma(v)$.
- A *subgraph* $G' = (V', E')$ of $G = (V, E)$ is such $V' \subset V$, $E' \subset E$ and $\{v, u\} \in E' \Rightarrow v, u \in V'$.
- A subset C of V can define an *induced subgraph* $G(C) = (C, E(C))$, where $E(C) = \{(v, u) \in E | v, u \in C\}$.

Graph Theory Basic Notions (ctd)

- A path P is a subgraph $P = (V(P), E(P))$ such $V(P) = \{v_{i_0}, \cdots, v_{i_k}\}$ and $E(P) = \{\{v_{i_0}, v_{i_1}\}, \{v_{i_1}, v_{i_2}\}, \cdots, \{v_{i_{k-1}}, v_{i_k}\}\}$.
- k is the length path.
- If no vertice is repeated, then the path is *simple*. length,
- If there exists a path between v and u, they are *connected*.
- The graph is a *connected graph* if $\forall v, u$, there is, at least, one path connecting v and u.
- A connected subgraph is called a *connected component*.

Diameter of the Zachary Karate Club network



Graph Theory Basic Notions (ctd)

- *Density* of a subgraph $C(V(C), E(C))$: ratio between $|E(C)|$ and the maximum possible number of edges:

$$\delta(G(C)) = \frac{|E(C)|}{|V(C)|(|V(C)|-1)/2} \tag{1}$$

- A partition of V in two subsets C and $V \setminus C$ is called a *cut*.
- The *cut size* is the number of edges joining vertices of C with vertices of $V \setminus C$:

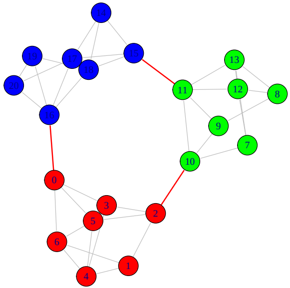
$$c(C, V \setminus C) = |\,\{\{u,v\} \in E | u \in C, v \in V \setminus C\}\,| \tag{2}$$

- For an unweighted graph, *degree* of a vertice v, $deg(v)$: number of incident edges,
- For weighted graph:

$$deg(v_i) = \sum_{j=1}^{n} w_{i,j}. \tag{3}$$

Plan

Detecting Communities



But What is a Community?

- In the clustering framework a *community* is a cluster of nodes in a graph,

But What is a Community?

- In the clustering framework a *community* is a cluster of nodes in a graph,
- But a very important question is *what is a cluster?*,

But What is a Community?

- In the clustering framework a *community* is a cluster of nodes in a graph,
- But a very important question is *what is a cluster?*,
- Even in the clustering literature there is non complete agreement between all authors, ([EC02]),

But What is a Community?

- In the clustering framework a *community* is a cluster of nodes in a graph,
- But a very important question is *what is a cluster?*,
- Even in the clustering literature there is non complete agreement between all authors, ([EC02]),
- But, objects inside a cluster must be more similar than objects outside of this cluster,

But What is a Community?

- In the clustering framework a *community* is a cluster of nodes in a graph,
- But a very important question is *what is a cluster?*,
- Even in the clustering literature there is non complete agreement between all authors, ([EC02]),
- But, objects inside a cluster must be more similar than objects outside of this cluster,
- *Objects are clustered or grouped based in maximizing intra-class similarity and minimizing inter-class similarity* .

But What is a Community?

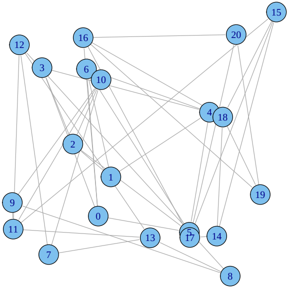
- In the clustering framework a *community* is a cluster of nodes in a graph,
- But a very important question is *what is a cluster?*,
- Even in the clustering literature there is non complete agreement between all authors, ([EC02]),
- But, objects inside a cluster must be more similar than objects outside of this cluster,
- *Objects are clustered or grouped based in maximizing intra-class similarity and minimizing inter-class similarity* .
- In graph framework, clustering is dividing vertices such nodes of a community must be more connected with nodes of this community, than with nodes outside of the cluster ([Sha07], [For10]).

But What is a Community?

- In the clustering framework a *community* is a cluster of nodes in a graph,
- But a very important question is *what is a cluster?*,
- Even in the clustering literature there is non complete agreement between all authors, ([EC02]),
- But, objects inside a cluster must be more similar than objects outside of this cluster,
- *Objects are clustered or grouped based in maximizing intra-class similarity and minimizing inter-class similarity* .
- In graph framework, clustering is dividing vertices such nodes of a community must be more connected with nodes of this community, than with nodes outside of the cluster ([Sha07], [For10]).
- It implies that it must exists at least a path between two nodes of a cluster, and this path must be internal to the cluster.

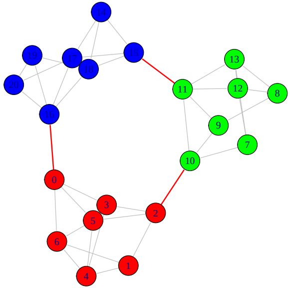
Then in a Community...

Connections must be minimum between groups and maximum within groups.



Then in a Community...

Connections must be minimum between groups and maximum within groups.



Wasserman Criteria

Four criteria:

Wasserman Criteria

Four criteria:

- *Complete mutuality*: all member of a subgroup must be “linked” with all members of the subgroup,

Wasserman Criteria

Four criteria:

- *Complete mutuality*: all member of a subgroup must be “linked” with all members of the subgroup,
- *Reachability*: existence (and length) of paths between vertices of a subgroup,

Wasserman Criteria

Four criteria:

- *Complete mutuality*: all member of a subgroup must be “linked” with all members of the subgroup,
- *Reachability*: existence (and length) of paths between vertices of a subgroup,
- *Nodal degree*: imposes a constraint on the number of adjacent vertices,

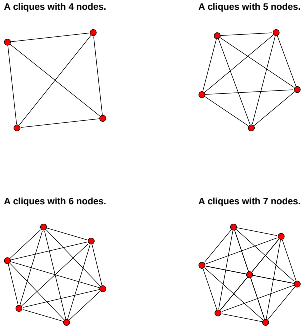
Wasserman Criteria

Four criteria:

- *Complete mutuality*: all member of a subgroup must be “linked” with all members of the subgroup,
- *Reachability*: existence (and length) of paths between vertices of a subgroup,
- *Nodal degree*: imposes a constraint on the number of adjacent vertices,
- *Internal versus external cohesion*: the former must be higher than the latter.

Complete mutuality

- In a very strict sense, in a community, all member of a subgroup must be “friends” with all members of the subgroup,
- In graph theory, it corresponds to a clique,
- But define of a community as a clique is very too strict (Alba 1973) calls it “a quite stingy”), that leads to relaxed definitions of the notion of clique...



Reachability

- An *n-clique* is a maximal subgraph such, for any pair of vertices, there exists at least a geodesic no larger than *n*.
- The classical clique is then a 1-clique.

Reachability

- An *n-clique* is a maximal subgraph such, for any pair of vertices, there exists at least a geodesic no larger than *n*.
- The classical clique is then a 1-clique.
- But a geodesic path of an *n*-clique could run outside of this latter, and then the diameter of the subgraph c an exceed *n*...
- That is why was defined *n-clan* which is an *n-clique* with diameter not larger than *n*.

Nodal degree

For a given graph G and a cluster C:

- A *k-plex* is a maximal subgraph such each vertice is adjacent to all other vertices of the subgraph except for k of them.
- Conversely a *k-core* is a maximal subgraph such each vertice is adjacent to, at least, k other vertices of the subgraph.

Internal versus external cohesion

For a given graph G and a cluster C:

- $deg_i(C)$: internal degree of C is the number of internal edges of C: $deg_i(v, C) = |\Gamma(v) \cap C|$,
- $deg_e(C)$: external degree of C is the number of edges with one vertice inside C, and the other outside of C: $deg_e(v, C) = |\Gamma(v) \cap (V \setminus C)|$,
- $deg_(C) = deg_i(C) + deg_e(C)$: degree of C,

Internal versus external cohesion

For a given graph G and a cluster C:

- $deg_i(C)$: internal degree of C is the number of internal edges of C: $deg_i(v, C) = |\Gamma(v) \cap C|$,
- $deg_e(C)$: external degree of C is the number of edges with one vertice inside C, and the other outside of C: $deg_e(v, C) = |\Gamma(v) \cap (V \setminus C)|$,
- $deg_(C) = deg_i(C) + deg_e(C)$: degree of C,
- If $deg_e(v, C) = 0$, then $v \in C$ is surely a good assignation for v,
- Conversly if $deg_i(v, C) = 0$, then we must have $v \notin C$.

Internal versus external cohesion

For a given graph G and a cluster C:

- $deg_i(C)$: internal degree of C is the number of internal edges of C: $deg_i(v, C) = |\Gamma(v) \cap C|$,
- $deg_e(C)$: external degree of C is the number of edges with one vertice inside C, and the other outside of C: $deg_e(v, C) = |\Gamma(v) \cap (V \setminus C)|$,
- $deg_(C) = deg_i(C) + deg_e(C)$: degree of C,
- If $deg_e(v, C) = 0$, then $v \in C$ is surely a good assignation for v,
- Conversly if $deg_i(v, C) = 0$, then we must have $v \notin C$.
- A *LS-set*, or *strong community* is a subgraph C such for each node $v \in C$ we have $deg_i(v, C) > deg_e(v, C)$.

Internal versus external cohesion

For a given graph G and a cluster C:

- Graph density is the ratio between existing number of edges and maximum possible number of edges: $\delta(G(C)) = \frac{|E(C)|}{|V(C)|(|V(C)|-1)/2}$

Internal versus external cohesion

For a given graph G and a cluster C:

- Graph density is the ratio between existing number of edges and maximum possible number of edges: $\delta(G(C)) = \frac{|E(C)|}{|V(C)|(|V(C)|-1)/2}$
- Intra-cluster density: quotient of the number of internal edges of C and the maximal possible number of internal nodes: $\delta_i(C) = \frac{|\{\{u,v\}|u,v \in C\}|}{|C|(|C|-1)/2}$,,

Internal versus external cohesion

For a given graph G and a cluster C:

- Graph density is the ratio between existing number of edges and maximum possible number of edges: $\delta(G(C)) = \frac{|E(C)|}{|V(C)|(|V(C)|-1)/2}$
- Intra-cluster density: quotient of the number of internal edges of C and the maximal possible number of internal nodes: $\delta_i(C) = \frac{|\{\{u,v\}|u,v \in C\}|}{|C|(|C|-1)/2}$,,
- Inter-cluster density: quotient of the existing number of edges with one vertice inside C, and the other outside of C and maximum possible number of edges in this configuration: $\delta_e(C) = \frac{|\{\{u,v\}|u \in C, v \notin C\}|}{|C|(|G|-|C|)}$,,

Internal versus external cohesion

For a given graph G and a cluster C:

- Graph density is the ratio between existing number of edges and maximum possible number of edges: $\delta(G(C)) = \frac{|E(C)|}{|V(C)|(|V(C)|-1)/2}$
- Intra-cluster density: quotient of the number of internal edges of C and the maximal possible number of internal nodes: $\delta_i(C) = \frac{|\{\{u,v\}|u,v \in C\}|}{|C|(|C|-1)/2}$,,
- Inter-cluster density: quotient of the existing number of edges with one vertice inside C, and the other outside of C and maximum possible number of edges in this configuration: $\delta_e(C) = \frac{|\{\{u,v\}|u \in C, v \notin C\}|}{|C|(|G|-|C|)}$,,

For a given partition $\{C_1, \cdots, C_k\}$ we want

$$\sum_{i=1}^{k} \delta_i(C_i) = \delta_i(G|C_1, \cdots, C_k) >> \delta(G).$$

Internal versus external cohesion

- Using the *cut size*, the number of edges joining vertices of C with vertices of $V \setminus C$,
- The conductance of a community C is defined to taking into account the order of the cluster and the outside of the cluster:

$$\Phi(C) = \frac{c(C, V \setminus C)}{\min\{deg(C), deg(V \setminus C)\}}$$

where $deg(C)$ and $deg(V \setminus C)$ are the total degrees of C and of the rest of the graph. The $min(\Phi(C))$ is obtained when C has a low cut size and when the total degree of the cluster and its complement are equal.

Internal versus external cohesion

- A community with strong inner ties must have a higher *Relative density*

$$\rho(C) = \frac{deg_i(C)}{deg(C)}.$$

Internal versus external cohesion

- A community with strong inner ties must have a higher *Relative density*

$$\rho(C) = \frac{deg_i(C)}{deg(C)}.$$

- The *edge connectivity* of a graph G is the minimal number of nodes to be removed so that G is disconnected, and is denoted $k(G)$.

Internal versus external cohesion

- A community with strong inner ties must have a higher *Relative density*

$$\rho(C) = \frac{deg_i(C)}{deg(C)}.$$

- The *edge connectivity* of a graph G is the minimal number of nodes to be removed so that G is disconnected, and is denoted $k(G)$.
- A community C can be defined as an Highly connected subgraph (HCS) such

$$k(C) > \frac{n}{2}.$$

Measures of Belonging to a Community

Dissimilarity Measures

A distance measure d between two objects must fulfill the following criteria:

1. separation: $d(u, u) = 0$,
2. symmetry: $d(u, v) = d(v, u)$,
3. triangle inequality: $d(u, v) \leq d(u, w) + d(w, v)$.

Similarity Measures

A similarity measure s must fulfill the following criteria:

1. $s(u, u) = k$, where k is a constant,
2. $s(u, v) = s(v, u)$,
3. $s(u, v) \leq s(u, u) = k$.

Measures of Belonging to a Community

Dissimilarity Measures

A distance measure d between two objects must fulfill the following criteria:

1. separation: $d(u, u) = 0$,
2. symmetry: $d(u, v) = d(v, u)$,
3. triangle inequality: $d(u, v) \leq d(u, w) + d(w, v)$.

Similarity Measures

A similarity measure s must fulfill the following criteria:

1. $s(u, u) = k$, where k is a constant,
2. $s(u, v) = s(v, u)$,
3. $s(u, v) \leq s(u, u) = k$.

Link Between Similarities and Dissimilarity

A dissimilarity measure d can be converted into a similarity measure using a strictly decreasing functions ϕ with some boundary conditions :

$$d = \phi(s) \text{ and } s = \phi^{-1}(d).$$

Classical Distance Recall

Recall Distances for Classical Vectors

For two vectors in $\mathbb{R}^2$, $u = (u_1, \cdots, u_n)$ and $v = (v_1, \cdots, v_n)$:

- *Euclidean*:

$$d(u, v) = \sqrt{\sum_{k=1}^{n} (u_k - v_k)^2}$$

- *Manhattan distance*: $d^1(u, v) = \sum_{k=1}^{n} |u_k - v_k|$
- *Tchebychev's distance* $d^\infty(u, v) = \max_{k=1,\cdots,n} |u_k - v_k|$
- *Cosine*:

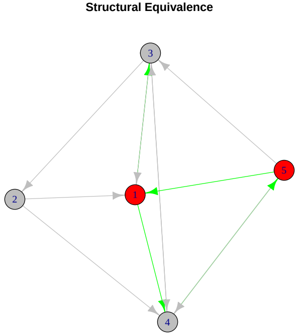
$$\theta(u, v) = \frac{\sum_{k=1}^{n} u_k \cdot v_k}{\sqrt{\sum_{k=1}^{n} (u_k)^2} \sqrt{\sum_{k=1}^{n} (v_k)^2}}.$$

Measures for Graphs: structural equivalence

Given the adjacency matrix $A = \{a_{i,j}\}$, u_i and u_j are structurally equivalent if they have the same neighbors, i.e. if $d_{i,j} = 0$:

$$d_{i,j} = \sqrt{\sum_{k \neq i,j} (a_{i,k} - a_{j,k})^2}$$

$$A = \begin{Bmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \end{Bmatrix}$$



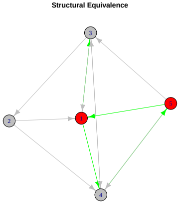
Measures for Graphs: Pearson Correlation

Another measure directly defined on A is the Pearson correlation matrix:

$$c_{i,j} = \frac{\sum_{k=1}^{n}(a_{i,k} - \mu_i)(a_{j,k} - \mu_j)}{n\sigma_j\sigma_j} \tag{4}$$

with $\mu_i = \sum_k a_{i,k}/n$ and $\sigma_i = \sqrt{\sum_k (a_{i,k} - \mu_k)^2/n}$.

$$A = \left\{ \begin{array}{ccccc} 1.00 & 0.16 & -0.16 & 0.16 & 0.66 \\ 0.16 & 1.00 & 0.66 & -0.66 & 0.66 \\ -0.16 & 0.66 & 1.00 & -1.00 & 0.16 \\ 0.16 & -0.66 & -1.00 & 1.00 & -0.16 \\ 0.66 & 0.66 & 0.16 & -0.16 & 1.00 \end{array} \right\}$$



Measures for Graphs: Jaccard Index

Another popular seed to build (dis)similarity measure is the *Jaccard index* which measures similarity between sets:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}. \quad (5)$$

A first use of the Jaccard index in the graph theory context is to measure the *overlap* of the neighborhoods of two nodes v and u:

$$\omega(v, u) = \frac{|\Gamma(v) \cap \Gamma(u)|}{|\Gamma(v) \cup \Gamma(u)|} \quad (6)$$

which is equal to zero when there is no common neighbors, and one when v and u are structurally equivalent. And, as $0 \leq J(A, b) \leq 1$, it is easy to define the Jaccard distance:

$$J_\delta(A, B) = 1 - J(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}. \quad (7)$$

Measures for Graphs: Tanimoto coefficient

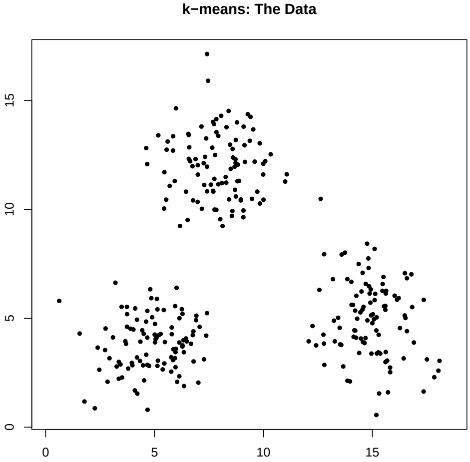
Tanimoto coefficient is an extension of the cosine similarity which coincide with the Jaccard index for binary vectors:

$$T(A,B) = \frac{\sum_{k=1}^{n} a_k \cdot b_k}{\sum_{k=1}^{n} a_k + \sum_{k=1}^{n} b_k - \sum_{k=1}^{n} a_k \cdot b_k}. \tag{8}$$

The Tanimoto coefficient gives the quotient between the number of shared features by A and B, divided by the whole number of features for A and B.

Plan

Partitional Algorithms - Introduction



Partitional Algorithms : k-means

k-means algorithms try to maximize the *intra-cluster dissimilarity*:

$$g_n(\mathcal{C}) = \sum_{i=1}^{k} \sum_{x \in C_i} d(x, c_i)$$

Algorithm:

Starting:Determine an initial vector (or centers) $(c_1^{(0)} \cdots, c_k^{(0)})$,

Repeat until stationarity: $t \leftarrow t + 1$

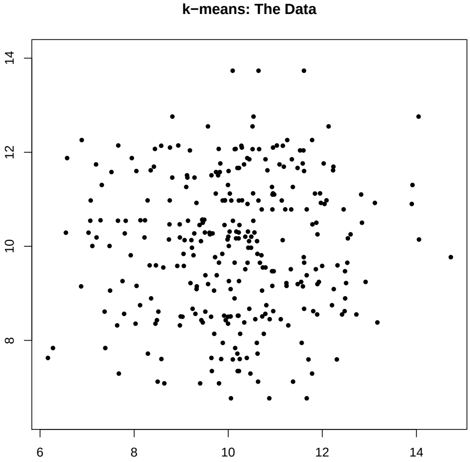
Assignment: observations go to the cluster with the closest centroid:

$$C_i^{(t+1)} = \left\{ x : d(x, c_i^{(t)}) \leq d(x, c_j^{(t)}) \forall j = 1, \ldots, k \right\}$$

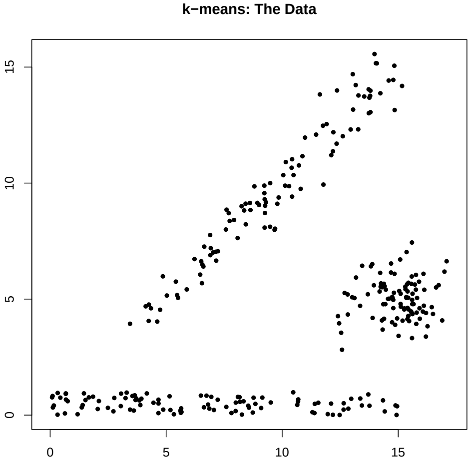
Update: Compute the new centroids $(c_1^{(t+1)}, \cdots, c_k^{(t+1)})$:

$$c_i^{(t+1)} = \frac{1}{|C_i^{(t)}|} \sum_{x \in C_i^{(t)}} x$$

Partitional Algorithms - Instability



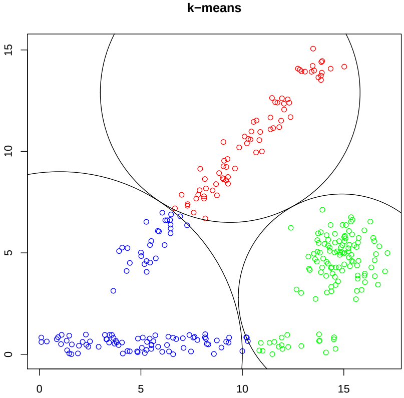
Partitional Algorithms - Spherical Clusters



Partitional Algorithms - Drawbacks

Drawbacks

- choice of k: an inappropriate choice leads too non significant results,
- spherical clusters: algorithms works better when spherical clusters are in data,
- instability: the random starting partition can leads to a local optimum for the criterion function.

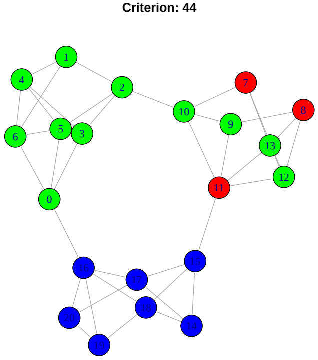


Complexity: $O(m^2 \times n \times k)$ where m is the number of attributes, n the number of objects to cluster and k the number of clusters.

Partitional Algorithms - Drawbacks

Drawbacks

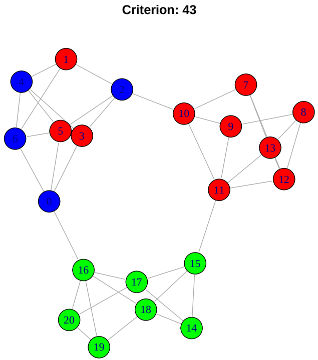
- choice of k: an inappropriate choice leads too non significant results,
- spherical clusters: algorithms works better when spherical clusters are in data,
- instability: the random starting partition can leads to a local optimum for the criterion function.



Complexity: $O(m^2 \times n \times k)$ where m is the number of attributes, n the number of objects to cluster and k the number of clusters.

Drawbacks

- choice of k: an inappropriate choice leads too non significant results,
- spherical clusters: algorithms works better when spherical clusters are in data,
- instability: the random starting partition can leads to a local optimum for the criterion function.

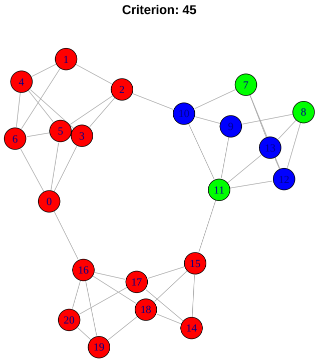


Complexity: $O(m^2 \times n \times k)$ where m is the number of attributes, n the number of objects to cluster and k the number of clusters.

Partitional Algorithms - Drawbacks

Drawbacks

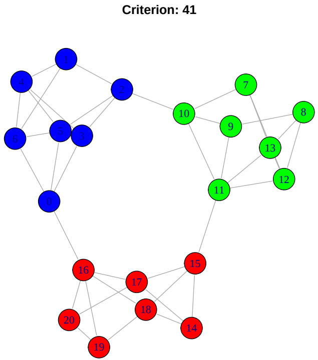
- choice of k: an inappropriate choice leads too non significant results,
- spherical clusters: algorithms works better when spherical clusters are in data,
- instability: the random starting partition can leads to a local optimum for the criterion function.



Complexity: $O(m^2 \times n \times k)$ where m is the number of attributes, n the number of objects to cluster and k the number of clusters.

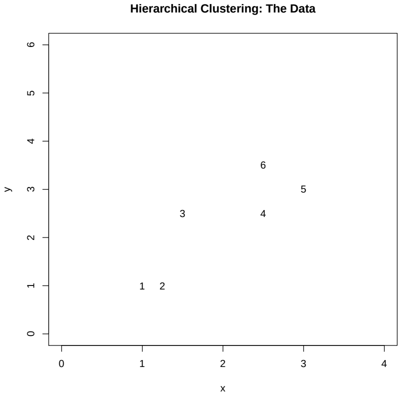
Drawbacks

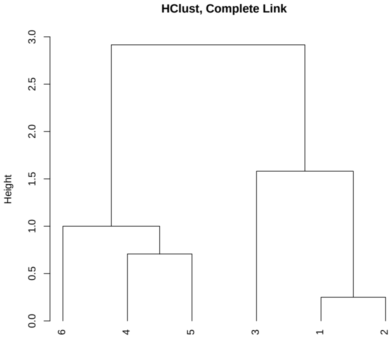
- choice of k: an inappropriate choice leads too non significant results,
- spherical clusters: algorithms works better when spherical clusters are in data,
- instability: the random starting partition can leads to a local optimum for the criterion function.



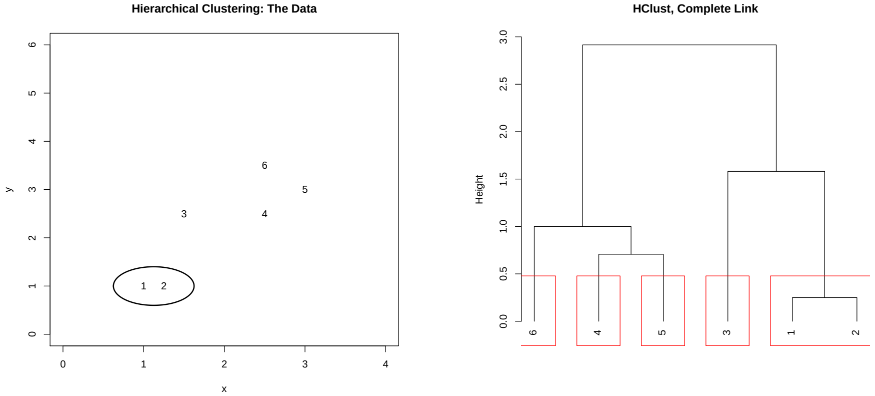
Complexity: $O(m^2 \times n \times k)$ where m is the number of attributes, n the number of objects to cluster and k the number of clusters.

Agglomerative Hierarchical Algorithms - Intro

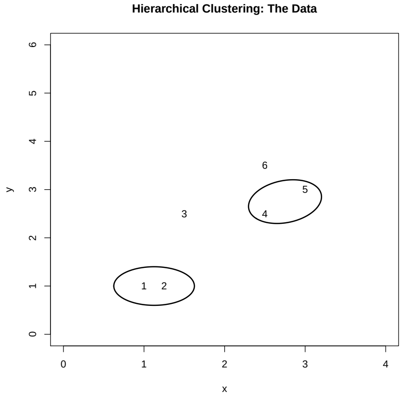


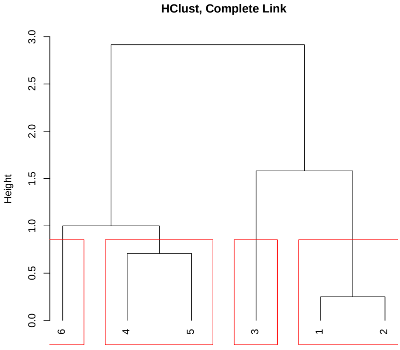


Agglomerative Hierarchical Algorithms - Intro

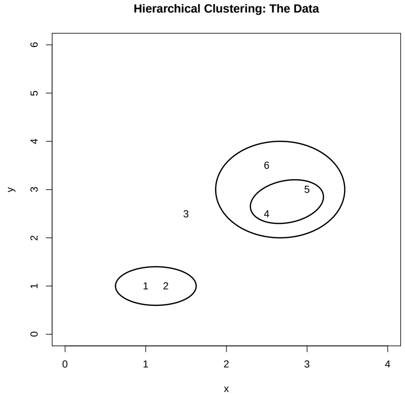


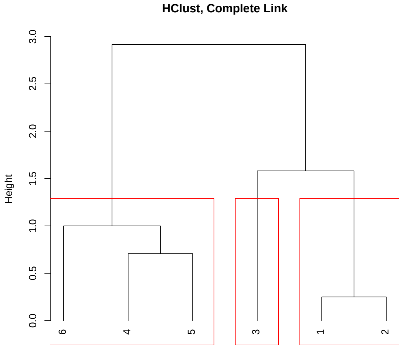
Agglomerative Hierarchical Algorithms - Intro



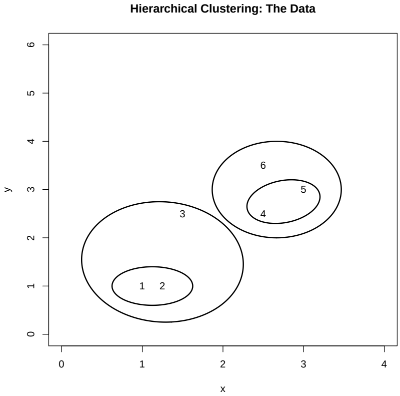


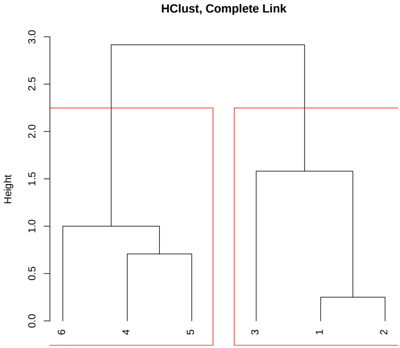
Agglomerative Hierarchical Algorithms - Intro



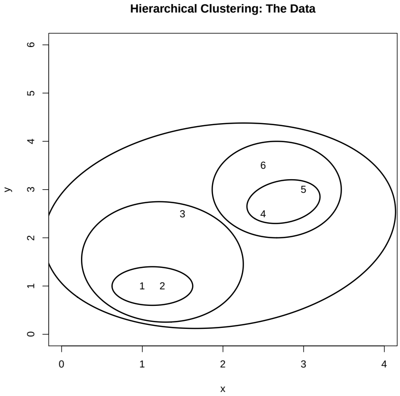


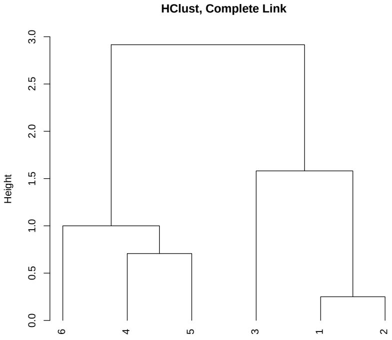
Agglomerative Hierarchical Algorithms - Intro





Agglomerative Hierarchical Algorithms - Intro





Agglomerative Hierarchical Algorithms - For SNA

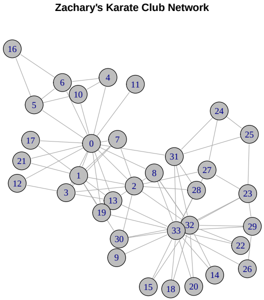


Figure: The Zachary Karate club network.

At the starting point, the n objects to cluster are their own classes: $\{\{x_1\}, \cdots, \{x_n\}\}$, then at each stage we merged the two more similar clusters.

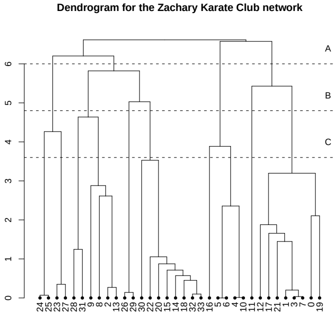


Figure: A dendrogram for the Zachary Karate club network.

At the starting point, the n objects to cluster are their own classes: $\{\{x_1\}, \cdots, \{x_n\}\}$, then at each stage we merged the two more similar clusters.

Agglomerative Hierarchical Algorithms - The Link Choice

For a given dissimilarity measure d between objects, several dissimilarities between clusters D exist:

- the single linkage:

$$D(A,B) = \min\{d(x,y) : x \in A,\ y \in B\},$$

- the complete linkage:

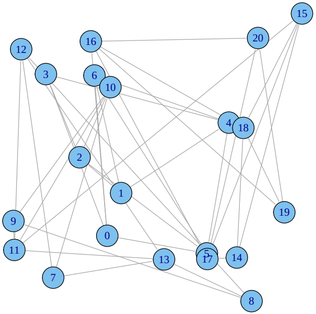
$$D(A,B) = \max\{d(x,y) : x \in A,\ y \in B\},$$

- the average linkage:

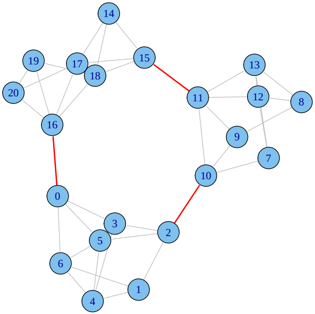
$$D(A,B) = \frac{1}{|A| \cdot |B|} \sum_{x \in A} \sum_{y \in B} d(x,y).$$

Drawbacks: vertices of a community may be not correctly classified.
Complexity: $O(n^2)$ for the single linkage and $O(n^2 \log n)$ for complete and average linkages.

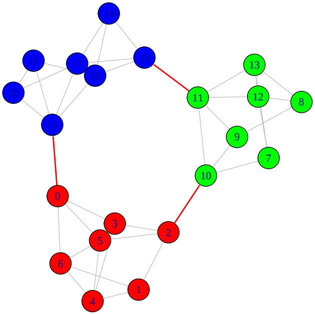
Divisive Hierarchical Algorithms - Betweenness



Divisive Hierarchical Algorithms - Betweenness



Divisive Hierarchical Algorithms - Betweenness



Divisive Hierarchical Algorithms - Betweenness

- Find the connecting edges to find the communities.
- *Edge betweenness* is the number of shortest paths between all vertex pairs that run along the edge.
- Divide the graph finding and "‘removing"’ edges connecting community, two communities, i.e. edges on the maximum of shortest paths between these Communities (max edge betweenness):
  1. compute the edge betweenness for all edges of the running graph,
  2. remove the edge with the largest value (which gives the new running graph).
- Stop when no improvement on a criterion like the *modularity*:

$$\mathcal{M}(C_1, \cdots, C_k) = \sum_i deg_e(C_i) - \sum_i deg_i(C_i)$$

Divisive Hierarchical Algorithms - Betweenness

- Find the connecting edges to find the communities.
- *Edge betweenness* is the number of shortest paths between all vertex pairs that run along the edge.
- Divide the graph finding and "‘removing"’ edges connecting community, two communities, i.e. edges on the maximum of shortest paths between these Communities (max edge betweenness):
  1. compute the edge betweenness for all edges of the running graph,
  2. remove the edge with the largest value (which gives the new running graph).
- Stop when no improvement on a criterion like the *modularity*:

$$\mathcal{M}(C_1, \cdots, C_k) = \sum_i deg_e(C_i) - \sum_i deg_i(C_i)$$

Complexity edge betwneenness on a graph can be computed in $O(n \cdot m)$ for unweighted graphs and in $O(n \cdot m + n^2 logn)$ for the weighted.

Spectral Methods - intro

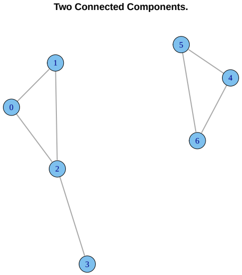


Figure: Two connected components.

Adjacency Matrix

$$W = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

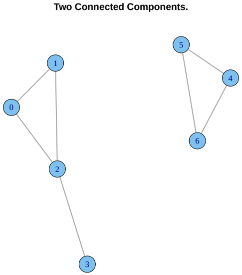


Figure: Two connected components.

Adjacency Matrix

$$W = \left(\begin{array}{cccc|ccc} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{array}\right).$$

Spectral Decomposition

$$W = Q \Lambda Q^{-1}$$

Spectral Methods - intro

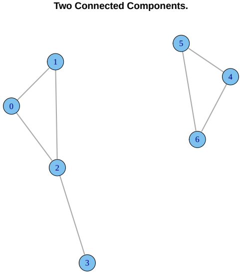


Figure: Two connected components.

Eigenvectors matrix

$$Q = \left(\begin{array}{cccc|ccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & 1 & 0 & 0 & 0 \\ -3 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & -2 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 & 1 \end{array}\right).$$

spectrum=$\{4, 3, 1, 0\} \cup \{3, 3, 0\}$

Spectral Decomposition

$$W = Q \Lambda Q^{-1}$$

Spectral Methods - intro

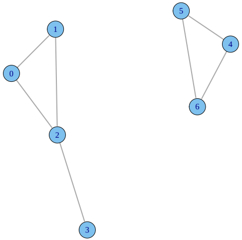


Figure: Two connected components.

Eigenvectors matrix

$$Q = \left(\begin{array}{cccc|ccc} 1 & 1 & 1 & \color{red}{1} & 0 & 0 & \color{red}{0} \\ 1 & -1 & 1 & \color{red}{1} & 0 & 0 & \color{red}{0} \\ -3 & 0 & 0 & \color{red}{1} & 0 & 0 & \color{red}{0} \\ 1 & 0 & -2 & \color{red}{1} & 0 & 0 & \color{red}{0} \\ \hline 0 & 0 & 0 & \color{red}{0} & 0 & -2 & \color{red}{1} \\ 0 & 0 & 0 & \color{red}{0} & 1 & 1 & \color{red}{1} \\ 0 & 0 & 0 & \color{red}{0} & -1 & 1 & \color{red}{1} \end{array}\right).$$

spectrum=$\{4, 3, 1, \color{red}{0}\} \cup \{3, 3, \color{red}{0}\}$

Spectral Decomposition

$$W = Q\Lambda Q^{-1}$$

Spectral Methods - intro

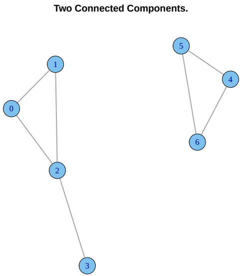


Figure: Two connected components.

Eigenvectors matrix

$$Q = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & -1 & 1 & 0 & 1 \\ -3 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & -2 & 0 & 1 \\ 0 & 0 & -2 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

spectrum=$\{4, 3, 3, 3, 1, 0, 0\}$ Spectral Decomposition

$$W = Q \Lambda Q^{-1}$$

Spectral Methods - intro

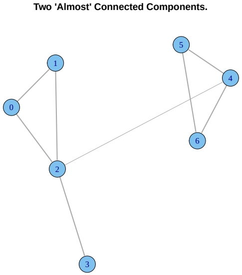


Figure: Two almost connected component.

Adjacency Matrix

$$W = \left(\begin{array}{cccc|ccc} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & \color{red}{1} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{array}\right).$$

Spectral Decomposition

$$W = Q \Lambda Q^{-1}$$

Spectral Methods - intro

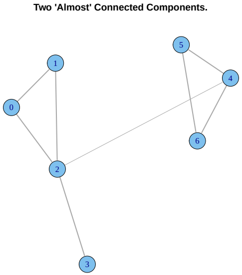


Figure: Two almost connected component.

Eigenvectors matrix

$$Q = \left(\begin{array}{ccccc|cc} . & . & . & . & . & -18 & 1 \\ . & . & . & . & . & -18 & 1 \\ . & . & . & . & . & -10 & 1 \\ . & . & . & . & . & -18 & 1 \\ \hline . & . & . & . & . & 15 & 1 \\ . & . & . & . & . & 25 & 1 \\ . & . & . & . & . & 25 & 1 \end{array}\right).$$

spectrum=$\{\cdots\}$ Spectral Decomposition

$$W = Q\Lambda Q^{-1}$$

Spectral Methods - Algorithm

For efficiency reason it is recommended to not work directly with the adjacency matrix W but with the Laplacian matrix:

$$L = D - W.$$

where D is such that we found the degrees $deg(v_i)$ on the diagonal, and then choose a normalization:

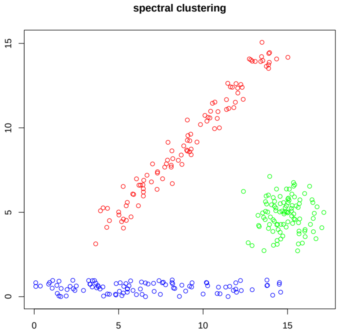
$$L_{\text{rw}} = D^{-1}L \text{ or } L_{\text{sym}} = D^{-1/2}LD^{-1/2}$$

Algorithm for spectral clustering is the following:

1. compute the eigenvalues and sort them such $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$,
2. compute the last k eigenvectors $\vec{u_{n-k}} \cdots, \vec{u_n}$,
3. form matrix $U \in \mathbb{R}^{n \times k}$ with $\vec{u_{n-k}} \cdots, \vec{u_n}$ as columns, and matrix $Y = U^t$,
4. cluster the points $(y_i)_{i=1,\cdots,n}$ using the k-means algorithm into clusters $A_1, \cdots, A_k$,
5. build the communities $C_1, \cdots, C_k$ such $C_i = \{v_j | y_j \in A_i\}$.

Spectral Methods - For More than Graphs

Efficient even for non "'graph"' data: it is sufficient to have a similarity measure $s(u, v)$ to build the weight/adjacency matrix W:



Complexity issue: the computation of the k eigenvectors of the Laplacian matrix require a time in $O(n^3)$.

Galois Lattices - Introduction

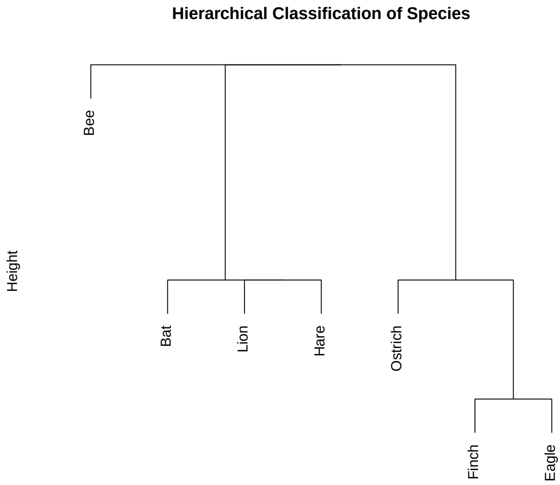
To cluster from a binary table cwith objects and attributes,

	Preying	Flying	Bird	Mammal
Lion	x			x
Finch		x	x	
Eagle	x	x	x	
Hare				x
Ostrich			x	
Bee		x		
Bat		x		x

we can compute a similarity table...

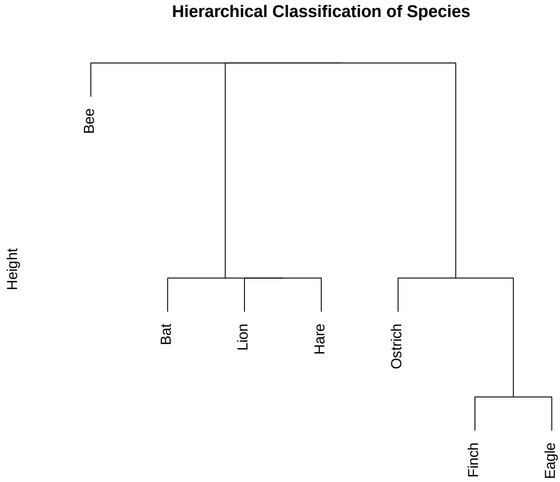
	Lion	Finch	Eagle	Hare	Ostrich	Bee	Bat
Lion	2	0	1	1	0	0	1
Finch	0	2	2	0	1	1	1
Eagle	1	2	3	0	1	1	1
Hare	1	0	0	1	0	0	1
Ostrich	0	1	1	0	1	0	0
Bee	0	1	1	0	0	1	1
Bat	1	1	1	1	0	1	2

do a hierarchical classification



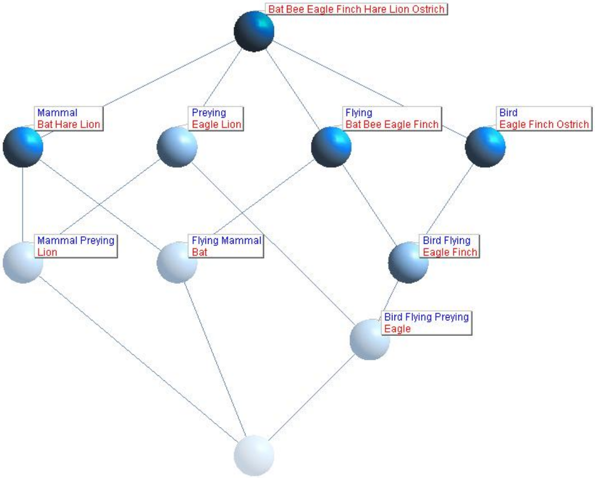
Galois Lattices - Introduction

do a hierarchical classification and choose one class per object!



Galois Lattices - Introduction

...or extract all the concepts and shared propeties!



Galois Lattices: Intension

Attributes A

Objects O

	Preying	Flying	Bird	Mammal
Lion	x			x
Finch		x	x	
Eagle	x	x	x	
Hare				x
Ostrich			x	
Bee		x		
Bat		x		x

For $X \in O$:

$$f(X) = \{a \in A | \forall o \in X, oIa\}.$$

Galois Lattices: Extension

Attributes A

Objects O

	Preying	Flying	Bird	MammifÃ¨re
Lion	x			x
Moineau		x	x	
Eagle	x	x	x	
Hare				x
Ostrich			x	
Bee		x		
Bat		x		x

For $Y \in A$:

$$g(Y) = \{o \in O | \forall a \in Y, oIa\}.$$

Attributes A

Objects O

	Preying	Flying	Bird	MammifÃ¨re
Lion	x			x
Moineau		x	x	
Eagle	x	x	x	
Hare				x
Ostrich			x	
Bee		x		
Bat		x		x

A concept is $(X, Y) \in O \times A$ such:

$$f(X) = Y \;\&\; g(Y) = X.$$

Galois Lattices: POSET

Attributes A

Objects O

	Preying	Flying	Bird	MammifÃ¨re
Lion	x			x
Moineau		x	x	
Eagle	x	x	x	
Hare				x
Ostrich			x	
Bee		x		
Bat		x		x

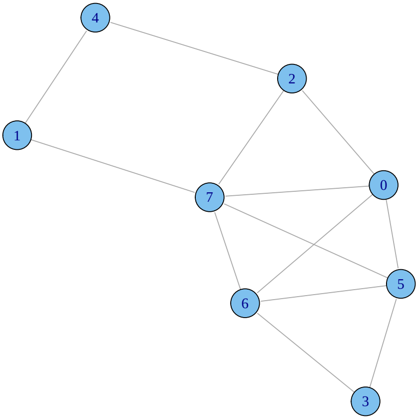
$(X_1, Y_1), (X_2, Y_2) \in O \times A : (X_1, Y_1) \leq (X_2, Y_2) \Leftrightarrow X_1 \subseteq X_2(\text{ or } Y_1 \supseteq Y_2)$.

$(\{\text{Moineau}, \text{Eagle}, \text{Ostrich}\}, \{\text{Birdx}\}) \supset$

$(\{\text{Moineau}, \text{Eagle}\}, \{\text{Flying}, \text{Birdx}\}) \supset$

$(\{\text{Eagle}\}, \{\text{Preying}, \text{Flying}, \text{Birdx}\})$

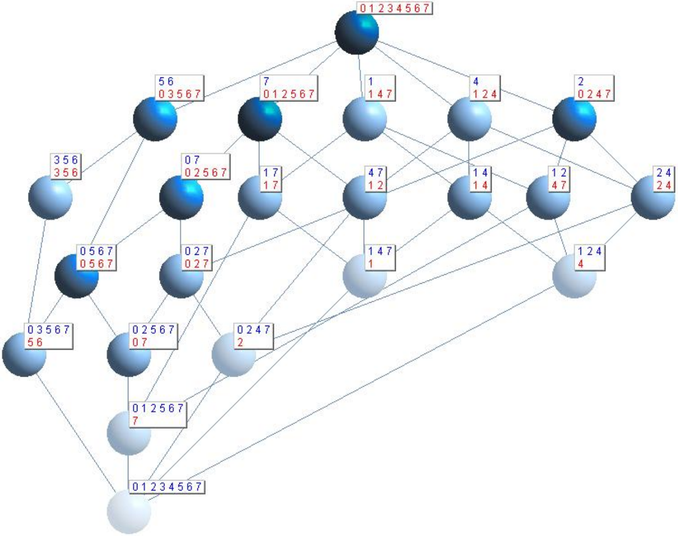
Galois Lattices for Social Networks



Galois Lattices for Social Networks

	1	2	3	4	5	6	7	8
1	×		×			×	×	×
2		×			×			×
3	×		×		×			×
4				×		×	×	
5		×	×		×			
6	×			×		×	×	×
7	×			×		×	×	×
8	×	×	×			×	×	×

Galois Lattices for Social Networks



Galois Lattices for Social Networks

Conceptual Metrics based on Galois Lattices.

Relatedness

Relatedness(O)= % of objects which share properties with O.

Closeness

Closeness(O)= % of shared properties of with its related objects.

Relatedness $\rightarrow$ $\downarrow$ Closeness	High	Low
High	Clustered	
Low		Marginal

Filter the p % of marginal objects for a chosen $p \in]0, 1[$.

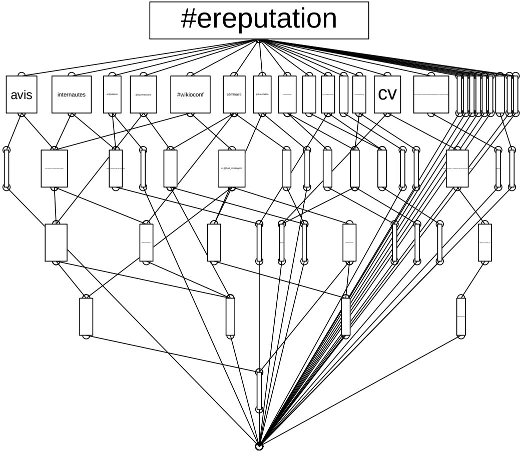
Galois Lattices vs Similarity Based Clusterings

Criteria	GL	SBC
Similarity	Equality	Proximity
Uniqueness of results	Y	N
Completeness of final structure	Y	N
Attribute Weight	N	Y
Continuous value management	Hard	Y

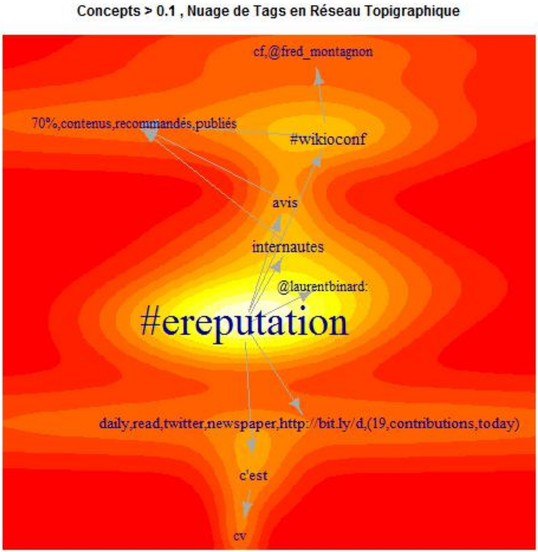
Complexity: if we denote $|O|$ the number of objects, $|A|$ the number of attributes and $|L|$ the size of the lattices (i.e. the number of concepts), then algorithms have a complexity time in $O(|O|^2|A||L|)$ or $O(|O||A|^2|L|)$.

EVARIST: eBuzz Monitoring on Twitter

Treillis Complet



EVARIST: eBuzz Monitoring on Twitter



navigation

- Pajek
- News
- Network Analysis
- How to
- Kako
- Download
- Events
- Resources
- Links
- User's Comments
- Pajek mailing list
- IMFM ePrints

search

toolbox

Index
Recent changes
Backlinks
Login

Show pagesource | Old revisions

Trace: » how_to » network_analysis » resources » pajek » start » download

Download

Pajek

Pajek runs on Windows and is free for noncommercial use.
It also runs on Unix or Mac.
Pajek manual. History.

Pajek for Windows 32 bit

Download the ESNA 2 version 2.04 / Book edition 2, (May 16, 2011) - installation pack. To install it run pajekBE2 and follov
Download the previous version 2.03, (January 26, 2011) of Pajek installation pack. To install it run pajek203 and follow the in
Download the ESNA Book Edition version (October 1, 2004) of Pajek.

Pajek for Windows 64 bit

On Windows 64 bit a special version of Pajek can use up to 4GB of available computer memory.

Download the version 2.04 - 64bit / Book Edition 2, (May 16, 2011) of Pajek installation pack. To install it run pajek64-BE2 ar

Data sets

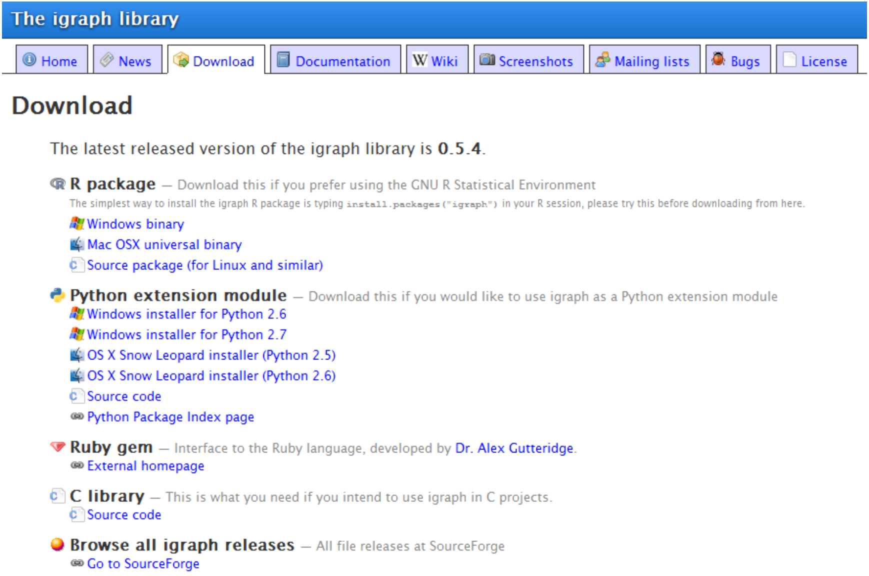
Pajek data sets.

- Data sets for experimenting with Pajek

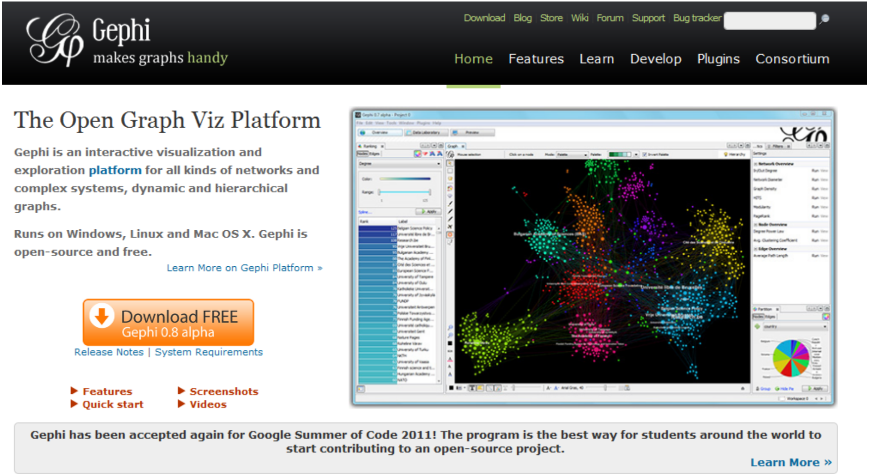
Slides

San Diego Sunbelt XXIX workshop: slides 1, slides 2, data.

igraph


The igraph library
Home
News
Download
Documentation
Wiki
Screenshots
Mailing lists
Bugs
License
Download
The latest released version of the igraph library is 0.5.4.
R package — Download this if you prefer using the GNU R Statistical Environment
The simplest way to install the igraph R package is typing install.packages("igraph") in your R session, please try this before downloading from here.
Windows binary
Mac OSX universal binary
Source package (for Linux and similar)
Python extension module — Download this if you would like to use igraph as a Python extension module
Windows installer for Python 2.6
Windows installer for Python 2.7
OS X Snow Leopard installer (Python 2.5)
OS X Snow Leopard installer (Python 2.6)
Source code
Python Package Index page
Ruby gem — Interface to the Ruby language, developed by Dr. Alex Gutteridge.
External homepage
C library — This is what you need if you intend to use igraph in C projects.
Source code
Browse all igraph releases — All file releases at SourceForge
Go to SourceForge

Gephi


Gephi
makes graphs handy
Download Blog Store Wiki Forum Support Bug tracker
Home Features Learn Develop Plugins Consortium
The Open Graph Viz Platform
Gephi is an interactive visualization and exploration platform for all kinds of networks and complex systems, dynamic and hierarchical graphs.
Runs on Windows, Linux and Mac OS X. Gephi is open-source and free.
Learn More on Gephi Platform »
Download FREE
Gephi 0.8 alpha
Release Notes | System Requirements
Features
Quick start
Screenshots
Videos
Gephi has been accepted again for Google Summer of Code 2011! The program is the best way for students around the world to start contributing to an open-source project.
Learn More »

Conclusions

- Definition of a community or cluster, is not an easy task.
- In fact, what is a cluster is in the eyes of the beholder,
- *One person's noise could be another person's signal*.
- *Cluster analysis is structure seeking although its operation is structure imposing*.
- In data clustering many choices must be done before any analysis (cluster definition , algorithms, measures, ...) which influence strongly the result.
- But, in spite of all these warnings, clustering algorithms allow us to retrieve valuable pieces of information in social networks, by finding communities.

Vladimir Estivill-Castro.
Why so many clustering algorithms: a position paper.
SIGKDD Explor. Newsl., 4(1):65–75, 2002.

Santo Fortunato.
Community detection in graphs.
Physics Reports, 486(3-5):75–174, 2010.

Satu Elisa Shaeffer.
Graph clustering.
Computer Science Review, 1:27–64, 2007.

U. von Luxburg.
A tutorial on spectral clustering.
Technical Report Technical Report 149, Max Planck Institute for Biological Cybernetics, 2006.