COMMUNITY DETECTION
(GRAPH CLUSTERING)
COMMUNITY DETECTION

• Community detection is equivalent to “clustering” in unstructured data

• Clustering: unsupervised machine learning
  ‣ Find groups of elements that are similar to each other
    - People based on DNA, apartments based on characteristics, etc.
  ‣ Hundreds of methods published since 1950 (k-means)
  ‣ Problem: what does “similar to each other” means?
COMMUNITY DETECTION

<table>
<thead>
<tr>
<th>MiniBatchKMeans</th>
<th>AffinityPropagation</th>
<th>MeanShift</th>
<th>SpectralClustering</th>
<th>Ward</th>
<th>AgglomerativeClustering</th>
<th>DBSCAN</th>
<th>Birch</th>
<th>GaussianMixture</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Image of MiniBatchKMeans" /></td>
<td><img src="image" alt="Image of AffinityPropagation" /></td>
<td><img src="image" alt="Image of MeanShift" /></td>
<td><img src="image" alt="Image of SpectralClustering" /></td>
<td><img src="image" alt="Image of Ward" /></td>
<td><img src="image" alt="Image of AgglomerativeClustering" /></td>
<td><img src="image" alt="Image of DBSCAN" /></td>
<td><img src="image" alt="Image of Birch" /></td>
<td><img src="image" alt="Image of GaussianMixture" /></td>
</tr>
<tr>
<td>.01s</td>
<td>4.34s</td>
<td>.07s</td>
<td>1.46s</td>
<td>.23s</td>
<td>.12s</td>
<td>.01s</td>
<td>.04s</td>
<td>.01s</td>
</tr>
<tr>
<td><img src="image" alt="Image of MiniBatchKMeans" /></td>
<td><img src="image" alt="Image of AffinityPropagation" /></td>
<td><img src="image" alt="Image of MeanShift" /></td>
<td><img src="image" alt="Image of SpectralClustering" /></td>
<td><img src="image" alt="Image of Ward" /></td>
<td><img src="image" alt="Image of AgglomerativeClustering" /></td>
<td><img src="image" alt="Image of DBSCAN" /></td>
<td><img src="image" alt="Image of Birch" /></td>
<td><img src="image" alt="Image of GaussianMixture" /></td>
</tr>
<tr>
<td>.02s</td>
<td>4.79s</td>
<td>.05s</td>
<td>2.83s</td>
<td>.22s</td>
<td>.12s</td>
<td>.01s</td>
<td>.04s</td>
<td>.01s</td>
</tr>
<tr>
<td><img src="image" alt="Image of MiniBatchKMeans" /></td>
<td><img src="image" alt="Image of AffinityPropagation" /></td>
<td><img src="image" alt="Image of MeanShift" /></td>
<td><img src="image" alt="Image of SpectralClustering" /></td>
<td><img src="image" alt="Image of Ward" /></td>
<td><img src="image" alt="Image of AgglomerativeClustering" /></td>
<td><img src="image" alt="Image of DBSCAN" /></td>
<td><img src="image" alt="Image of Birch" /></td>
<td><img src="image" alt="Image of GaussianMixture" /></td>
</tr>
<tr>
<td>.02s</td>
<td>2.87s</td>
<td>.10s</td>
<td>3.5s</td>
<td>.84s</td>
<td>.64s</td>
<td>.01s</td>
<td>.04s</td>
<td>.01s</td>
</tr>
<tr>
<td><img src="image" alt="Image of MiniBatchKMeans" /></td>
<td><img src="image" alt="Image of AffinityPropagation" /></td>
<td><img src="image" alt="Image of MeanShift" /></td>
<td><img src="image" alt="Image of SpectralClustering" /></td>
<td><img src="image" alt="Image of Ward" /></td>
<td><img src="image" alt="Image of AgglomerativeClustering" /></td>
<td><img src="image" alt="Image of DBSCAN" /></td>
<td><img src="image" alt="Image of Birch" /></td>
<td><img src="image" alt="Image of GaussianMixture" /></td>
</tr>
<tr>
<td>.02s</td>
<td>2.46s</td>
<td>.05s</td>
<td>4.4s</td>
<td>.62s</td>
<td>.32s</td>
<td>.01s</td>
<td>.04s</td>
<td>.01s</td>
</tr>
<tr>
<td><img src="image" alt="Image of MiniBatchKMeans" /></td>
<td><img src="image" alt="Image of AffinityPropagation" /></td>
<td><img src="image" alt="Image of MeanShift" /></td>
<td><img src="image" alt="Image of SpectralClustering" /></td>
<td><img src="image" alt="Image of Ward" /></td>
<td><img src="image" alt="Image of AgglomerativeClustering" /></td>
<td><img src="image" alt="Image of DBSCAN" /></td>
<td><img src="image" alt="Image of Birch" /></td>
<td><img src="image" alt="Image of GaussianMixture" /></td>
</tr>
<tr>
<td>.02s</td>
<td>2.18s</td>
<td>.05s</td>
<td>.59s</td>
<td>.21s</td>
<td>.12s</td>
<td>.02s</td>
<td>.04s</td>
<td>.01s</td>
</tr>
<tr>
<td><img src="image" alt="Image of MiniBatchKMeans" /></td>
<td><img src="image" alt="Image of AffinityPropagation" /></td>
<td><img src="image" alt="Image of MeanShift" /></td>
<td><img src="image" alt="Image of SpectralClustering" /></td>
<td><img src="image" alt="Image of Ward" /></td>
<td><img src="image" alt="Image of AgglomerativeClustering" /></td>
<td><img src="image" alt="Image of DBSCAN" /></td>
<td><img src="image" alt="Image of Birch" /></td>
<td><img src="image" alt="Image of GaussianMixture" /></td>
</tr>
<tr>
<td>.02s</td>
<td>2.02s</td>
<td>.05s</td>
<td>.47s</td>
<td>.12s</td>
<td>.11s</td>
<td>.01s</td>
<td>.04s</td>
<td>.01s</td>
</tr>
</tbody>
</table>
COMMUNITY DETECTION

• Community detection:
  ‣ Find groups of nodes that are:
    - Strongly connected to each other
    - Weakly connected to the rest of the network
    - Ideal form: each community is 1) A clique, 2) A separate connected component
  ‣ No formal definition
  ‣ Hundreds of methods published since 2003
WHY COMMUNITY DETECTION?

• One of the key properties of complex networks was
  ‣ High clustering coefficient
  ‣ (friends of my friends are my friends)

• Different from random networks. How to explain it?
  ‣ Watts strogatz (spatial structure?)

• => In real networks, presence of dense groups: communities
  ‣ Small, dense (random) networks have high density.
  ‣ Large networks could be interpreted as aggregation of smaller, denser
    networks, with much fewer edges between them
SOME HISTORY

• The graph partitioning problem was a classic problem in graph theory

• It goes like this:
  ‣ How to split a network in $k$ equal parts such that there is a minimal number of edges between parts.
  ‣ Variants were proposed:
    - What if partitions are not exactly same size ?
    - What if the number of parts is not exactly $k$ ?
    - ...
Then in 2002, [Girvan & Newman 2002], introduction of the problem of “community discovery”:

- Observation that social networks are very often composed of groups
- The number and the size of these groups is not known in advance
- Can we design an algorithm to discover automatically those groups?

COMMUNITY STRUCTURE IN REAL GRAPHS

- If you plot the graph of your Facebook friends, it looks like this.
COMMUNITY STRUCTURE IN REAL GRAPHS

• Connections in the brain?
COMMUNITY STRUCTURE IN REAL GRAPHS

• Phone call communications in Belgium?
FIRST METHOD BY GIRVAN & NEWMAN

• 1) Compute the betweenness of all edges
• 2) Remove the edge of highest betweenness
• 3) Repeat until all edges have been removed
  ‣ Connected components are communities
• => It is called a divisive method
• => What you obtain is a dendrogram
• How to cut this dendrogram at the best level?
FIRST METHOD BY GIRVAN & NEWMAN

Cluster Dendrogram

Maximal modularity
FIRST METHOD BY GIRVAN & NEWMAN

• Introduction of the **Modularity**

• The modularity is computed for a partition of a graph
  ‣ (each node belongs to one and only one community)

• It compares:
  ‣ The *observed* fraction of edges inside communities
  ‣ To the *expected* fraction of edges inside communities in a random network
MODULARITY

\[ Q = \frac{1}{(2m)} \sum_{vw} \left[ A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) \]

Original formulation
MODULARITY

\[ Q = \frac{1}{2m} \sum_{vw} \left[ A_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_v, c_w) \]

Sum over all pairs of nodes
MODULARITY

\[ Q = \frac{1}{(2m)} \sum_{vw} \left[ A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) \]

1 if in same community
MODULARITY

\[ Q = \frac{1}{(2m)} \sum_{vw} \left[ A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) \]

1 if there is an edge between them
MODULARITY

\[ Q = \frac{1}{(2m)} \sum_{vw} \left( A_{vw} - \left( \frac{k_v k_w}{(2m)} \right) \right) \delta(c_v, c_w) \]

Probability of an edge in a configuration model
MODULARITY

Can also be defined as a sum by community

\[
Q = \frac{1}{L} \sum_{i=1}^{\mid C \mid} (L_i - \frac{1}{2}K_i^2)
\]

with \(L_i = L(H(c_i))\) the number of edges inside community \(i\) and \(K_i = \sum_{u \in c_i} k_u\) the sum of degrees of nodes in community \(i\).
• Modularity compares the observed network to a null model
  ‣ Usually the configuration model
    - Multi-edges and loops are allowed
  ‣ Other models could be used, such as ER random graphs.

• Natural extension to weighted/multi-edge networks
FIRST METHOD BY GIRVAN & NEWMAN

• Back to the method:
  ‣ Create a dendrogram by removing edges
  ‣ Cut the dendrogram at the best level using modularity

• =>In the end, your objective is... to optimize the Modularity, right?

• Why not optimizing it directly!
MODULARITY OPTIMIZATION

• From 2004 to 2008: The golden age of Modularity

• Scores of methods proposed to optimize it
  ‣ Graph spectral approaches
  ‣ Meta-heuristics approaches (simulated annealing, multi-agent…)
  ‣ Local/Global approaches…

• => 2008: the Louvain algorithm
LOUVAIN ALGORITHM

- Simple, greedy approach
  - Easy to implement
  - Fast

- Yields a hierarchical community structure

- Beat state of the art on all aspects (when introduced)
  - Speed
  - Max modularity obtained
  - Do not fall in some traps (see later)
LOUVAIN ALGORITHM

• Each node starts in its own community

• Repeat until convergence
  ‣ FOR each node:
    ‧ FOR each neighbor:
      if adding node to its community increase modularity, do it

• When converged, create an induced network
  ‣ Each community becomes a node
  ‣ Edge weight is the sum of weights of edges between them

• Trick: Modularity is computed by community
  ‣ Global Modularity = sum of modularities of each community

LOUVAIN ALGORITHM

RESOLUTION LIMIT

• Modularity == Definition of good communities?

• 2006-2008: series of articles [Fortunato, Lancicchinetti, Barthelemy]
  ‣ Resolution limit of Modularity

• Let’s see an example

RESOLUTION LIMIT

Let’s consider a ring of cliques

Cliques are as dense as possible

Single edge between them:

=> As separated as possible

Any acceptable algorithm => Each clique is a community
RESOLUTION LIMIT

But with modularity:

Small graphs => OK

Large graphs =>
The max of modularity obtained by merging cliques
RESOLUTION LIMIT

- Discovery that Modularity has a “favorite scale”:

- For a graph of given \textbf{density} and \textbf{size}:
  - Communities cannot be smaller than a fraction of nodes
  - Communities cannot be larger than a fraction of nodes

- Modularity optimisation will never discover
  - Small communities in large networks
  - Large communities in small networks
RESOLUTION LIMIT

- Multi-resolution modularity

$$\sum_{i}^{c} e_{ii} - a_{i}^{2} \quad \rightarrow \quad \sum_{i}^{c} e_{ii} - \lambda a_{i}^{2}$$

$$\lambda = \text{Resolution parameter}$$

More a patch than a solution…
OTHER WEAKNESSES

• Modularity has other controversial/not-intuitive properties:
  ‣ Global measure => a difference in one side of the network can change communities at the other end (imagine a growing clique ring…)
  ‣ Unable to find no community:
    - Network without community structure: Max modularity for partitions driven by random noise

• To this day, Louvain and modularity remain most used methods
  ‣ Results are usually “good”/useful
  ‣ Some newer methods gain popularity (SBM, Leiden,…)
ALTERNATIVES

• 1000+ Algorithms published, and counting

• What unfortunately many methods still do:
  ‣ They define their own criteria of good communities without being grounded on existing literature
  ‣ They show empirically on a few networks using a single validation method that their method is better than Louvain

• Common saying: “no algorithm is better than other, it depends on the type of network” (no free-lunch theorem) or “it depends on the objective” (I don’t really agree, open to discussion)
ALTERNATIVES

• Most serious alternatives (in my opinion)
  ‣ Infomap (based on information theory —compression)
  ‣ Stochastic block models (bayesian inference)

• These methods have a clear definition of what are good communities. Theoretically grounded
INFOMAP

- [Rovall & Bergstrom 2009]

- Find the partition minimizing the description of any random walk on the network

- We want to compress the description of random walks

**Random walk**

**Description**

- **Without Communities**
- **With communities**

**Huffman coding:** short codes for frequent items

**Prefix free:** no code is a prefix of another one (avoid fix length/separators)
The Infomap method

Finding the optimal partition $M$:

- Shannon’s source coding theorem (Shannon’s entropy)
  
  For a probability distribution $P = \{p_i\}$ such that $\sum p_i = 1$, the lower limit of the per-step code-length is
  
  $$L(P) = H(P) \equiv -\sum_i p_i \log p_i.$$  

- Minimise the expected description length of the random walk
  
  Sum of Shannon entropies of multiple codebooks weighted by the rate of usage

  $$L(M) = q \sum H(\mathcal{Q}) + \sum_{i=1}^{m} p_i \sum H(\mathcal{P}_i)$$

Algorithm

1. Compute the fraction of time each node is visited by the random walker (Power-method on adjacency matrix)

2. Explore the space of possible partitions (deterministic greedy search algorithm - similar to Louvain but here we join nodes if they decrease the description length)

3. Refine the results with simulated annealing (heat-bath algorithm)
INFOMAP

• To sum up:
  ‣ Infomap defines a *quality function* for a partition different than modularity
  ‣ Any algorithm can be used to optimize it (like Modularity)

• Advantage:
  ‣ Infomap can recognize random networks (no communities)
STOCHASTIC BLOCK MODELS

• Stochastic Block Models (SBM) are based on statistical models of networks.

• They are in fact more general than usual communities.

• The model is:
  ‣ Each node belongs to 1 and only 1 community
  ‣ To each pair of communities, there is an associated density (probability of each edge to exist)
Stochastic block models

Stochastic Block Models (SBM)

A stochastic block model is a random graph model defined by:

- \( k \) number of blocks
- \( b \) \( n \times 1 \) vector such as \( b_i \) describes the index of the block of node \( i \).
- \( E \) \( k \times k \) stochastic block matrix, such as \( E_{i,j} \) gives the number of edges between blocks \( i \) and \( j \) (or the probability to observe an edge between any pair of nodes chosen with one node in each of the two blocks).

Generating networks

1. Take \( N \) disconnected nodes
2. Connect each \( u, v \in V \) nodes with probability \( M_{z(u),z(v)} \)

Properties:

- Every vertices in a same module are statistically equivalent
- Vertices in a module are connected by a random graph
- Emergent degree distribution is a combination of Poisson distributions
STOCHASTIC BLOCK MODELS

- SBM can represent different things:
  - Associative SBM: density inside nodes of a same communities >> density of pairs belonging to different communities.
SBM can represent different things:

- Associative SBM: density inside nodes of a same communities $\gg$ density of pairs belonging to different communities.

This is very powerful and potentially relevant.

Problem: Often hard to interpret in real situations.

- SBM can be “constrained”: we impose that intra d.$\gg$inter d.
STOCHASTIC BLOCK MODELS

• Main weakness of SBM:
  ‣ Number of clusters must be specified (avoid trivial solution)

• Usual approach to solve it
  ‣ Similar to k-means in clustering: try different k and measure improvement (elbow-method)
  ‣ Not satisfying

• [2016 Peixoto]
  ‣ Non-parametric SBM
  ‣ Bayesian inference
  ‣ Minimum Description Length (MDL) (Occam’s razor)
STOCHASTIC BLOCK MODELS

Bayesian Formulation

\[ P(A, k, e, b) = P(A \mid k, e, b)P(k \mid e, b)P(e \mid b)P(b) \]

\[ P(b \mid A) = \frac{P(A \mid b)}{P(A)} \]

A: adjacency matrix
k: degree sequence
e: Matrix of edges between blocks
b: partitions

STOCHASTIC BLOCK MODELS

• To sum up:
  ‣ SBM have a convincing definition of communities
  ‣ In practice, inference slower than louvain/infomap
  ‣ But more powerful
  ‣ Can also say if there is no community
  ‣ And also suffer from a form of resolution limit

• Less often used, but regain popularity since works by Peixoto.
EVALUATION OF COMMUNITY STRUCTURE
EVALUATION

• Two main approaches:
  ‣ Intrinsic/Internal evaluation
    - Partition quality function
    - Individual Community quality function
  ‣ Comparison of observed communities and expected communities
    - Synthetic networks with community structure
    - Real networks with Ground Truth
INTRINSIC EVALUATION
INTRINSIC EVALUATION

• Partition quality function
  ‣ Already defined: **Modularity**, **graph compression**, etc.

• Quality function for individual community
  ‣ Internal Clustering Coefficient
    \[
    \frac{|E_{out}|}{|E_{out}| + |E_{in}|}
    \]
  ‣ Conductance:
    \[
    \frac{|E_{out}|}{|E_{out}| + |E_{in}|}
    \]
    - Fraction of external edges
    - \( |E_{in}|, |E_{out}| \): # of links to nodes inside (respectively, outside) the community
COMPARISON WITH GROUND TRUTH
SYNTHETIC NETWORKS

• Planted Partition models:
  ‣ Another name for SBM with manually chosen parameters
    - Assign degrees to nodes
    - Assign nodes to communities
    - Assign density to pairs of communities
    - Attribute randomly edges
  ‣ Problem: how to choose parameters?
    - Either oversimplifying (all nodes same degrees, all communities same #nodes, all intern densities equals…)
    - Or ad-hoc process (sample values from distributions)
SYNTHETIC NETWORKS
SYNTHETIC NETWORKS

- LFR Benchmark [Lancichinetti 2008]
  - High level parameters:
    - Slope of the power law distribution of degrees/community sizes
    - Avg Degree, Avg community size
    - Mixing parameter: fraction of external edges of each node
  - Varying the mixing parameter makes community more or less well defined

- Currently the most popular
SYNTHETIC NETWORKS

LFR Benchmark Networks with 200 Nodes

- $\mu=0.1$
  - #Edges = 2206
- $\mu=0.3$
  - #Edges = 2628
- $\mu=0.5$
  - #Edges = 2462
SYNTHETIC NETWORKS

• Pros of synthetic generators:
  ‣ We know for sure the communities we should find
  ‣ We can control finely the parameters to check robustness of methods
    - For instance, resolution limit…

• Cons:
  ‣ Generated networks are not realistic: simpler than real networks
    - LFR: High CC, scale free, but all nodes have the same mixing coefficient, no overlap, …
    - SBM: depend a lot on parameters, random generation might lead to unexpected ground truth (it is possible to have a node with no connections to other nodes of its own community…)
REAL NETWORKS WITH GT

• In some networks, **ground truth** communities are known:
  ‣ Social networks, people belong to groups (Facebook, Friendsters, Orkut, students in classes…)
  ‣ Products, belonging to categories (Amazon, music…)
  ‣ Other resources with defined groups (Wikipedia articles, Political groups for vote data…)

• Some websites have collected such datasets, e.g.
REAL NETWORKS WITH GT

• Pros of GT communities:
  ‣ Retain the full complexity of networks and communities

• Cons:
  ‣ No guarantee that communities are topological communities.
  ‣ In fact, they are not: some GT communities are not even a single connected component…

• Currently, controversial topic
  ‣ Some authors say it is non-sense to use them for validation
  ‣ Some others consider it necessary
REAL NETWORKS WITH GT

- Example: the most famous of all networks: Zackary Karate Club

If your algorithm find the right communities,
Then it is wrong…
MEASURING PARTITION SIMILARITIES

• Synthetic or GT, we get:
  ‣ Reference communities
  ‣ Communities found by algorithms

• How to measure their similarity?
  ‣ NMI => AMI
  ‣ ARI
  ‣ ...
MEASURING PARTITION SIMILARITIES

• NMI: Normalized Mutual Information

• Classic notion of Information Theory: Mutual Information
  ‣ How much knowing one variable reduces uncertainty about the other
  ‣ Or how much in common between the two variables

\[
I(X; Y) = \sum_{y \in Y} \sum_{x \in X} p(x, y) \log \left( \frac{p(x, y)}{p(x) p(y)} \right)
\]

• Normalized version: NMI
  ‣ 0: independent, 1: identical

• Adjusted for chance: aNMI

\[
AMI(U, V) = \frac{MI(U, V) - E\{MI(U, V)\}}{\max\{H(U), H(V)\} - E\{MI(U, V)\}}
\]
MEASURING PARTITION SIMILARITIES

$$I(X; Y) = \sum_{y \in Y} \sum_{x \in X} p(x, y) \log \left( \frac{p(x, y)}{p(x) p(y)} \right)$$

For all pairs of clusters (y,x)

Probability for a node picked at random to belong to both x and y

Probably for a node picked at random to belong to x
We start by taking a look at the resulting dom walks. The codelength gives us a corroboration of the presence communities compress the walks better because the random walker to close the triangle.

0.47, or 47% of connected node triads have all three edges necessary. However, and this is surprising, clustering is high – transitivity is this is not a result –: only 9% of possible edges are in the network.

Community structure. The network is sparse – by construction, MAX similarity function and setting 0.2 to 0.4 similarity test, that is able to distinguish between subcategories of community discovery and overlapping approaches can.

Figure 1: Again, note that we are not imposing the significance of the edges we include in the resulting network. We show to be connected how overlapping approaches can.

Table 3: The ten nodes with the highest MAX edge weight

<table>
<thead>
<tr>
<th>Rank</th>
<th>Algorithm</th>
<th>oNMI MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>linecomms</td>
<td>165</td>
</tr>
<tr>
<td>2</td>
<td>oslom</td>
<td>73</td>
</tr>
<tr>
<td>3</td>
<td>infomap-overlap</td>
<td>64</td>
</tr>
<tr>
<td>4</td>
<td>savi</td>
<td>62</td>
</tr>
<tr>
<td>5</td>
<td>labelperc</td>
<td>57</td>
</tr>
<tr>
<td>6</td>
<td>rmcl</td>
<td>54</td>
</tr>
<tr>
<td>7</td>
<td>edgebetween</td>
<td>41</td>
</tr>
<tr>
<td>7</td>
<td>leadeig</td>
<td>41</td>
</tr>
<tr>
<td>7</td>
<td>vbmod</td>
<td>41</td>
</tr>
<tr>
<td>10</td>
<td>gce</td>
<td>32</td>
</tr>
</tbody>
</table>

All methods

Overlapping only

OTHER MESO-SCALE ORGANIZATIONS
MESO-SCALE

• MACRO properties of networks:
  ‣ degree distribution, density, average shortest path…

• MICRO properties of networks:
  ‣ Centralities

• MESO-scale: what is in-between
  ‣ Community structure
  ‣ Overlapping Community Structure
  ‣ Core-Periphery
  ‣ Spatial Organization (another class)
CORE-PERIPHERY

- Already introduced in the first class, k-cores, etc.
OVERLAPPING COMMUNITIES

• In real networks, communities are often overlapping
  ‣ Some of your High-School friends might be also University Friends
  ‣ A colleague might be a member of your family
  ‣ …

• Overlapping community detection is considered much harder
  ‣ And is not well defined

• Difference between “attributes” and overlapping communities?
  ‣ Community of Women, Community of 17-19yo, Community of fans of…
OVERLAPPING COMMUNITIES

• Many algorithms
  ‣ Adaptations of modularity, random walks, label propagations…
  ‣ Original methods
  ‣ Many local methods (local criterium), unlike global optimization for non-overlapping methods.
OVERLAPPING COMMUNITIES

• Motif-based definitions:
  ‣ Cliques
    - Of a given size
    - Maximal cliques
  ‣ N-cliques
    - Set of nodes such as there is at least a path of length \( \leq N \) between them
    - Generalization of cliques for \( N > 1 \)
    - Computationally expensive
Link clustering - overlapping communities

Link graphs

- Links are replaced by nodes which are connected if the original links share a node

- Community detection on link graphs allows for overlapping communities
K-CLIQUE PERCOLATION

- (Other name: CPM, C-finder)
- Parameter: size k of atomic cliques
- 1) Find all cliques of size k
- 2) Merge iteratively all cliques having k-1 nodes in common
K-CLIQUE PERCOLATION

Clique for $k=3$: 
{1, 2, 3}, {1, 3, 4}, {2, 5, 6}, 
{5, 6, 7}, {5, 6, 8}, {6, 7, 8}, 
{5, 7, 8}, {5, 7, 9}

$k$-clique Communities:
{1, 2, 3, 4}, 
{2, 5, 6, 7, 8, 9}
HIERARCHICAL COMMUNITIES
