Chapter 9. Communities

Konstantinos Petrakis

CEID University of Patras

Project
2020 - 2021

Outline

i. Introduction
ii. Basics on Communities
iii. Hierarchical Clustering
    - Ravasz Algorithm
    - Girvan-Newman Algorithm
iv. Hierarchy in Real Networks

Introduction

Community: a group of nodes that have a higher likelihood of connecting to each other than to nodes from other communities.

- Social Networks (e.g. Zachary's Karate Club)
- Biological Networks (e.g. E.coli Metabolic Network)

The existence of communities is rooted in who connects to whom, hence they cannot be explained based on the degree distribution alone. To extract communities we must therefore inspect a network's detailed wiring diagram.

Theorem 1

Fundamental Hypothesis: A network's community structure is uniquely encoded in its wiring diagram.

Basics on Communities

- What is a community?
- How many communities are there in a network?
- How many different ways can we partition a network into communities?

Basics On Communities

Theorem 2

Connectedness and Density Hypothesis: A community is a locally dense connected subgraph in a network.

- Connectedness: all members of a community must be reached through other members of the same community
- Density: nodes that belong to a community have a higher probability to link to the other members of that community than to nodes that do not belong to the same community.

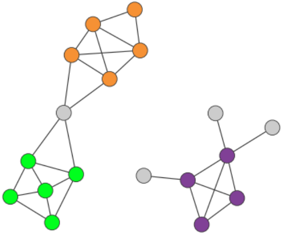


Figure 1: Connectedness and Density Hypothesis

Maximum Cliques

Community as a complete subgraph (clique).

- A clique automatically satisfies Theorem 2, it is a connected subgraph with maximal link density.
- Triangles are frequent in networks, larger cliques are rare.
- Too restrictive.

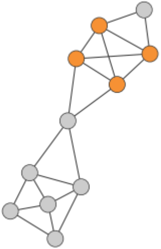


Figure 2: Clique

Strong and Weak Communities

Let's assume a connected subgraph C with N_C nodes.

- k_i^{int}: internal degree of node i.
- k_i^{ext}: external degree of node i.

If $k_i^{ext} = 0$ each neighbor of i is within C, hence C is a good community for node i.
If $k_i^{int} = 0$ then node i should be assigned to a different community.

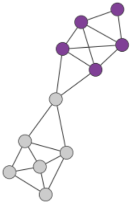


Figure 3: Strong Community

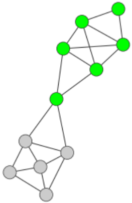


Figure 4: Weak Community

- Strong Community: for every node $i \in C$: $k_i^{int}(C) > k_i^{ext}(C)$
- Weak Community: $\sum_{i \in C} k_i^{int}(C) > \sum_{i \in C} k_i^{ext}(C)$

Number of Communities

A simple community detection problem: *Graph Bisection*

- Divide a network into two non-overlapping subgraphs, such that the number of links between the nodes in the two groups, called the cut size, is minimized.
- Brute force approach for partioning a network of N nodes into groups of N_1 and N_2 nodes: $\frac{N!}{N_1!\,N_2!}$

Using Stirling's formula, $n! = \sqrt{2\pi n}(\frac{n}{e})^n$

$$\frac{N!}{N_1!\,N_2!} = \frac{\sqrt{2\pi N}(\frac{N}{e})^N}{\sqrt{2\pi N_1}(\frac{N_1}{e})^{N_1}\sqrt{2\pi N_2}(\frac{N_2}{e})^{N_2}} \sim \frac{N^{N+\frac{1}{2}}}{N_1^{N_1+\frac{1}{2}}N_2^{N_2+\frac{1}{2}}}$$

If $N_1 = N_2 = \frac{N}{2}$, then $\frac{2^{N+1}}{\sqrt{N}} = e^{(N+1)\ln 2 - \frac{1}{2}\ln N}$

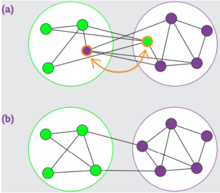


Figure 5: Kerninghan-Lin Algorithm

Community Detection

- Graph Partitioning: the number and the size of communities is predefined.
- Community Detection: both parameters are unknown.

A partition is a division of a network into an arbitrary number of groups, such that each node belongs to one and only one group. The number of possible partitions:

$$B_N = \frac{1}{e}\sum_{j=0}^{\infty}\frac{j^N}{j!}$$

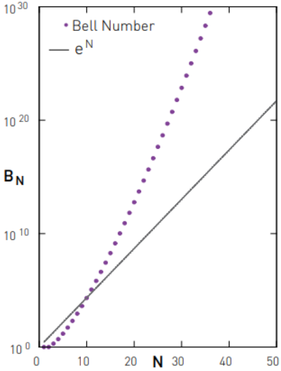


Figure 6: Bell Number

Brute force for community detection is out of the question!

Hierarchical Clustering-Towards Polynomial Time

Based on a similarity matrix, whose elements x_{ij} indicate the distance of node i from node j.

- Similarity is extracted from the relative position of nodes i and j within the network.

Once we have x_{ij}, hierarchical clustering iteratively identifies groups of nodes with high similarity.

- *Agglomerative algorithms* merge nodes with high similarity into the same community.
- *Divisive algorithms* isolate communities by removing low similarity links that tend to connect communities.

The output is a hierarchical tree, called a dendrogram, that predicts the possible community partitions.

Agglomerative procedures: The Ravasz algorithm

Step 1. Define the Similarity Matrix

$$x_{ij}^0 = \frac{J(i,j)}{min(k_i,k_j)+1-\Theta(A_{ij})}$$

- Heaviside step function

$$\Theta(x) = \begin{cases} 0, x \leq 0 \\ 1, x > 0 \end{cases}$$

- $J(i,j)$ is the number of common neighbors of node i and j, to which we add one $(+1)$ if there is a direct link between i and j.
- $min(k_i,k_j)$ is the smaller of the degrees k_i, k_j.

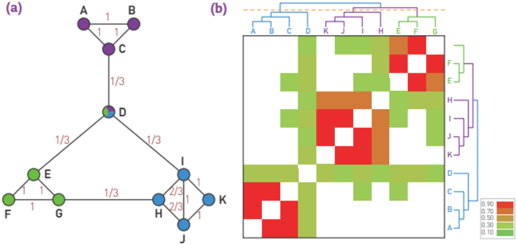


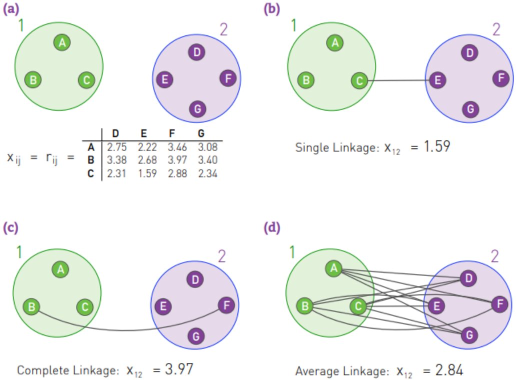
Figure 7: (a)Topological Overlap (b) Topological Overlap Matrix

The Ravasz algorithm

Step 2. Deciding group Similarity.

- single cluster similarity
- complete cluster similarity
- average cluster similarity

Ravasz algorithm uses the average cluster similarity method!



$x_{ij} = r_{ij} =$

	D	E	F	G
A	2.75	2.22	3.46	3.08
B	3.38	2.68	3.97	3.40
C	2.31	1.59	2.88	2.34

Single Linkage: $x_{12} = 1.59$

Complete Linkage: $x_{12} = 3.97$

Average Linkage: $x_{12} = 2.84$

Figure 8: Cluster Similarity

The Ravasz algorithm

Step 3. Apply Hierarchical Clustering.

1. Assign each node to a community of its own and evaluate x_{ij} for all node pairs.
2. Find the community pair or the node pair with the highest similarity and merge them into a single community.
3. Calculate the similarity between the new community and all other communities.
4. Repeat Steps 2 and 3 until all nodes form a single community.

The Ravasz algorithm

Step 4. Dendrogram.

- Eventually all nodes will form a single community.
- The dendrogram is used to extract the underlying organization.
- The dendrogram visualizes the order in which the nodes are assigned to specific communities.
- We must decide where to cut the dendrogram.
    - Hierarchical clustering does not tell us where that cut should be.

Ravasz Algorithm Example

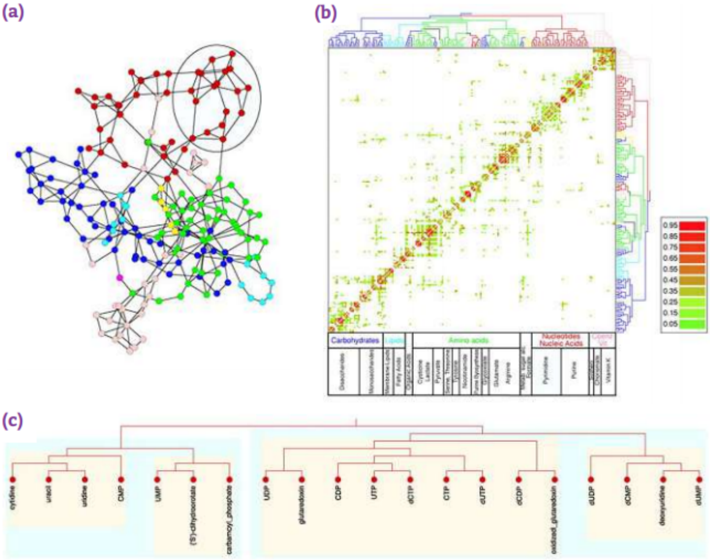


Figure 9: E.coli metabolic network example

The Ravasz Algorithm-Computational Complexity

1. Step 1. The calculation of the similarity matrix x_{ij}^0 requires comparing N^2 node pairs, hence the computational complexity is $O(N^2)$.
2. Step 2. Group similarity requires determining in each step the distance of the new cluster to all other clusters. Doing this N times, with $O(N)$ calculations for each new cluster, we get $O(N^2)$ complexity.
3. Step 3. The construction of the dendrogram requires $O(N \log N)$ steps.

Overall Complexity is $O(N^2) + O(N^2) + O(N \log N) = O(N^2)$.[1]

[1] Much better than the brute force approach which scales as $O(e^N)$.

Divisive Procedures: Girvan-Newman Algorithm

Reminder: Divisive procedures systematically remove the links connecting nodes that belong to different communities, eventually breaking a network into isolated communities.

Step 1. Define Centrality.

- x_{ij} in now called centrality.
- x_{ij} is used to select node pairs that are in different communities.
- x_{ij} is high if nodes i and j belong to different communities and small if they are in the same community.

Girvan-Newman Algorithm

Centrality Options

- Link Betweenness: x_{ij} is proportional to the number of shortest paths that go through the link (i, j).
- Random-Walk Betweenness: A pair of nodes m and n are chosen at random. A walker starts at m, following each adjacent link with equal probability until it reaches n. x_{ij} is the probability that the link $i \to j$ was crossed by the walker after averaging over all possible choices for the starting nodes m and n.

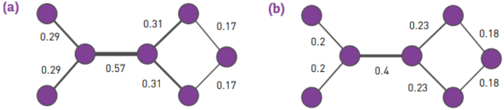


Figure 10: Centrality Measures: (a) Link Betweenness (b)Random-Walk Betweenness

Links connecting different communities are expected to have large x_{ij} while links within a community have small x_{ij}.

Girvan-Newman Algorithm

Step 2. Hierarchical Clustering.

1. Compute the centrality x_{ij} of each link.
2. Remove the link with the largest centrality. In case of a tie, choose one randomly.
3. Recalculate the centrality x_{ij} of each link for the altered network.
4. Repeat steps 2 and 3 until all links are removed.

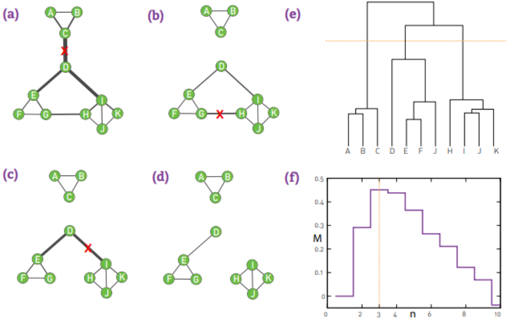


Figure 11: Girvan-Newman Algorithm

Girvan-Newman Algorithm Example

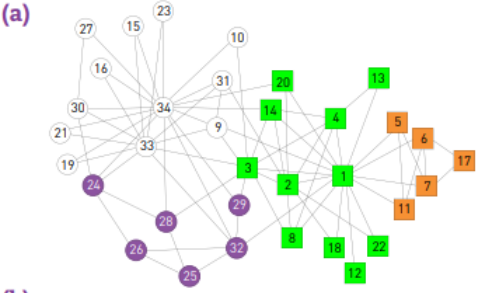


Figure 12: Zachary Karate Club Graph

Girvan-Newman Algorithm-Computational Complexity

- The most time consuming step is the calculation of centrality!
- Computational Complexity depends on the centrality measure.
- Link Betweenness: $O(LN)$.
- Taking into account step 3 of the algorithm the computational complexity is $O(L^2N)$, or $O(N^3)$ for a sparse network.

Hierarchy in Real Networks

Hierarchical clustering raises two fundamental issues.

- Nested Communities
  - Is the hierarchy of nested communities captured by the dendrogram indeed present in the network?
  - Or is it imposed by our algorithms?
- Communities and the Scale-Free Property
  - The density hypothesis states that a network can be partitioned into a collection of subgraphs that are only weakly linked to other subgraphs.
  - How can we have isolated communities in a scale-free network, if the hubs inevitably link multiple communities?

Hierarchy in Real Networks

Answer to both issues: *The Hierarchical Network model*.

- Scale-free Property: The hierarchical model generates a scale-free network with degree exponent $\gamma = 1 + \frac{\ln 5}{\ln 4} = 2.161$.
- Size Independent Clustering Coefficient
  - For the Erdős-Rényi and the Barabási-Albert models the clustering coefficient decreases with N.
  - For the hierarchical network $C = 0.743$ independent of the network size.
  - Such N-independent clustering coefficient has been observed in metabolic networks.
- Hierarchical Modularity
  - The model consists of numerous small communities that form larger communities, which again combine into ever larger communities.
  - The quantitative signature of this nested hierarchical modularity is the dependence of a node's clustering coefficient on the node's degree $C(k) \sim k^{-1}$.
  - The higher a node's degree, the smaller is its clustering coefficient.

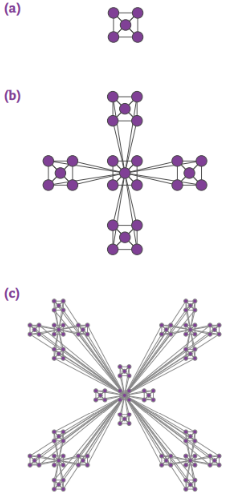


Figure 13: Hierarchical Network

Scaling in Hierarchical Networks

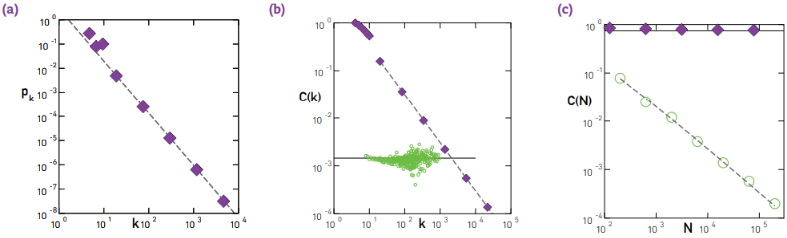


Figure 14: (a)Degree Distribution (b)Hierarchical Clustering (c)Size Independent Clustering Coefficient

- Degree Distribution: The generated network is scale-free as illustrated by the scaling of p_k with slope $\gamma = \frac{\ln 5}{\ln 4}$, shown as a dashed line.
- Hierarchical Clustering: $C(k) \sim k^{-1}$, shown as a dashed line. The circles show $C(k)$ for a randomly wired scale-free network, obtained from the original model by degree-preserving randomization. The lack of scaling indicates that the hierarchical architecture is lost under rewiring. Hence $C(k)$ captures a property that goes beyond the degree distribution.
- Size Independent Clustering Coefficient: The dependence of the clustering coefficient C on the network size N. For the hierarchical model C is independent of N (filled symbols), while for the Barabási-Albert model $C(N)$ decreases (empty symbols).

Hierarchy in Real Networks

The higher the degree of a node, the smaller is its C.

- Small degree nodes have high C because they reside in dense communities.
- High degree nodes have small C because they connect to different communities.
- e.g. in Figure 13 nodes at the center of the five-node modules have $k = 4$ and clustering coefficient $C = 4$. Those at the center of a 25-node module have $k = 20$ and C=3/19. Those at the center of the 125-node modules have $k = 84$ and C=3/83.

Hierarchy in Real Networks

- The hierarchical network model suggests that inspecting $C(k)$ allows us to decide if a network is hierarchical.
- For the Erdős-Rényi and the Barabási-Albert models $C(k)$ is independent of k, indicating that they do not display hierarchical modularity.
- To check whether hierarchical modularity is present in real systems, $C(k)$ was calculated for ten reference networks.

Calculating $C(k)$ for real networks

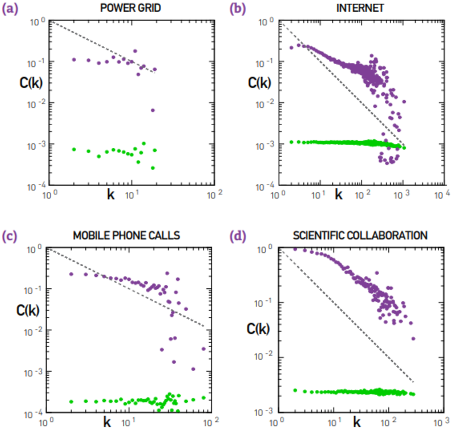


Figure 15: Hierarchy in real networks (1)

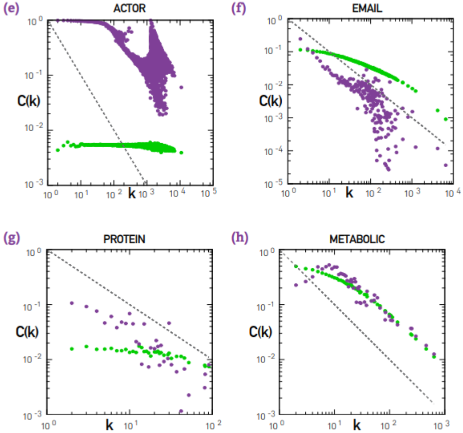


Figure 16: Hierarchy in real networks (2)

Calculating $C(k)$ for real networks

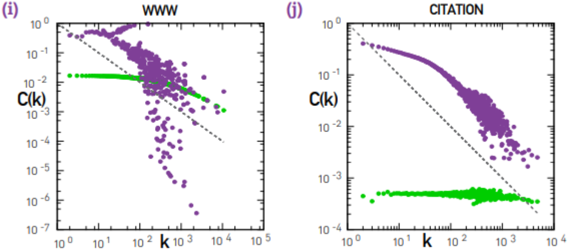


Figure 17: Hierarchy in real networks (3)

Empirical Results

- Only the power grid lacks hierarchical modularity, its $C(k)$ being independent of k.
- For the remaining nine networks $C(k)$ decreases with k. Hence in these networks small nodes are part of small dense communities, while hubs link disparate communities to each other.
- For the scientific collaboration, metabolic, and citation network $C(k) \sim k^{-1}$ in the high-k region. The form of $C(k)$ for the Internet, mobile, email, protein interactions, and the WWW needs to be derived individually, as for those $C(k) \nsim k^{-1}$. More detailed network models predict $C(k) \sim k^{-\beta}$, where β is between 0 and 2.

Ambiguity in Hierarchical Clustering

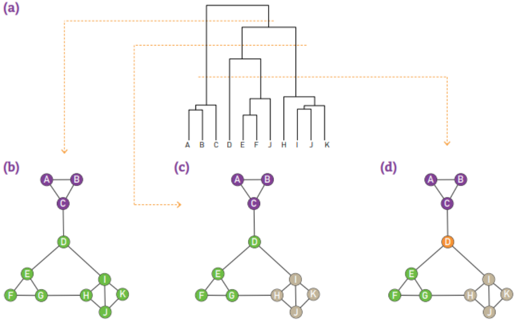


Figure 18: Ambiguity in Hierarchical Clustering

Hierarchical clustering does not tell us where to cut a dendrogram.

- Depending on where we make the cut in the dendrogram (a), we obtain (b) two, (c) three or (d) four communities.
- While for a small network we can visually decide which cut captures best the underlying community structure, it is impossible to do so in larger networks.

Ambiguity in Hierarchical Clustering

In summary

- Hierarchical clustering does not require preliminary knowledge about the number and the size of communities.
- In practice it generates a dendrogram that offers a family of community partitions characterizing the studied network.
- This dendrogram does not tell us which partition captures best the underlying community structure.
  - any cut of the hierarchical tree offers a potentially valid partition.
- This is at odds with our expectation that in each network there is a ground truth, corresponding to a unique community structure.

Also we can check whether the underlying network has hierarchical modularity, inspecting $C(k)$.

- $C(k)$ decreases in most real networks, indicating that most real systems display hierarchical modularity.
- $C(k)$ is independent of k for the Erdős-Rényi or Barabási-Albert models, indicating that these canonical models lack a hierarchical organization.

Thank you!