



# Analisi e Visualizzazione delle Reti Complesse

**NS11 - Communities**

**Prof. Rossano Schifanella**





## Outline

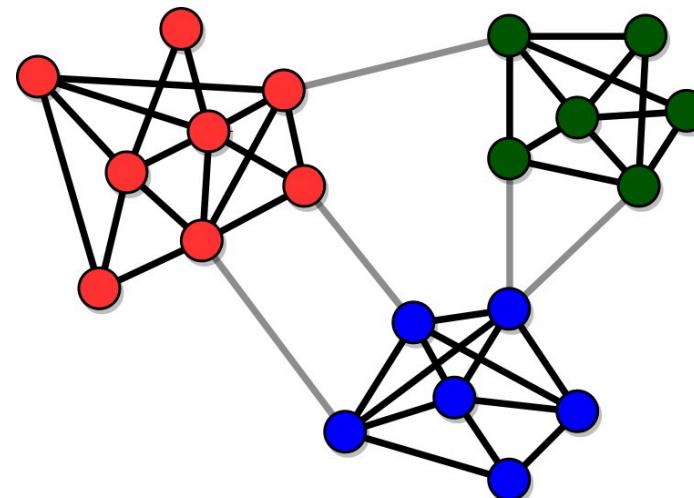
- Basic definitions
  - **community variables**
  - **community definitions**
  - **partitions**
- Related problems
  - **network partitioning**
  - **data clustering and dendograms**



# Basic definitions

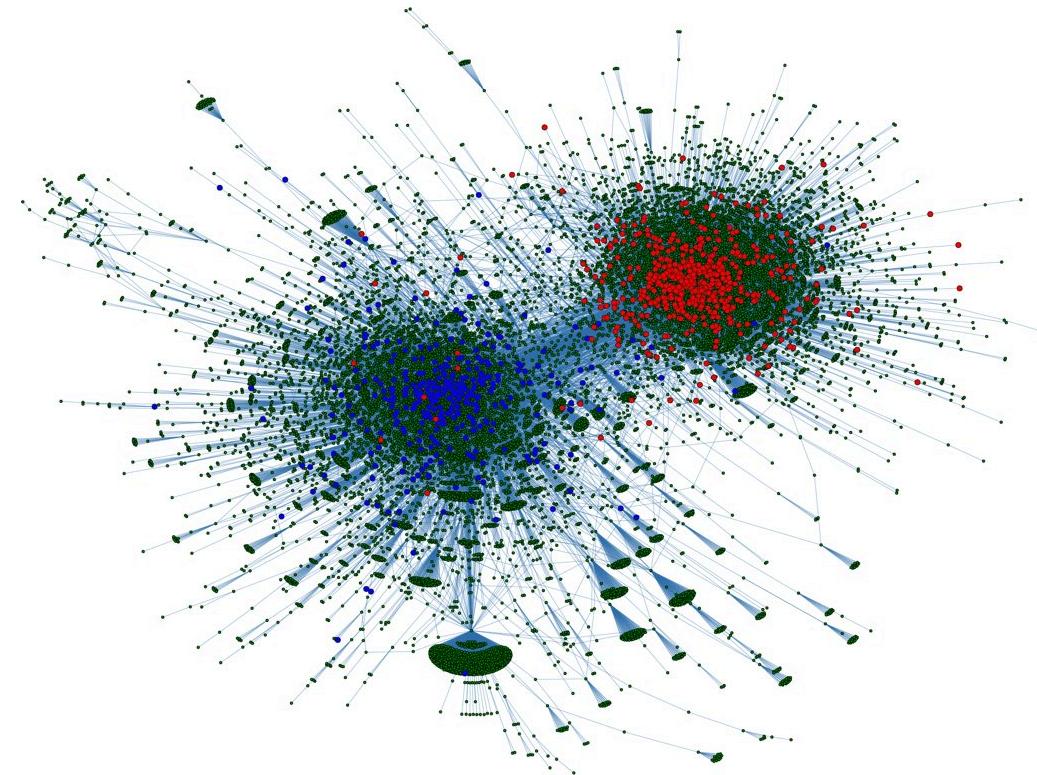
## Community structure

Communities (or clusters): sets of tightly connected nodes



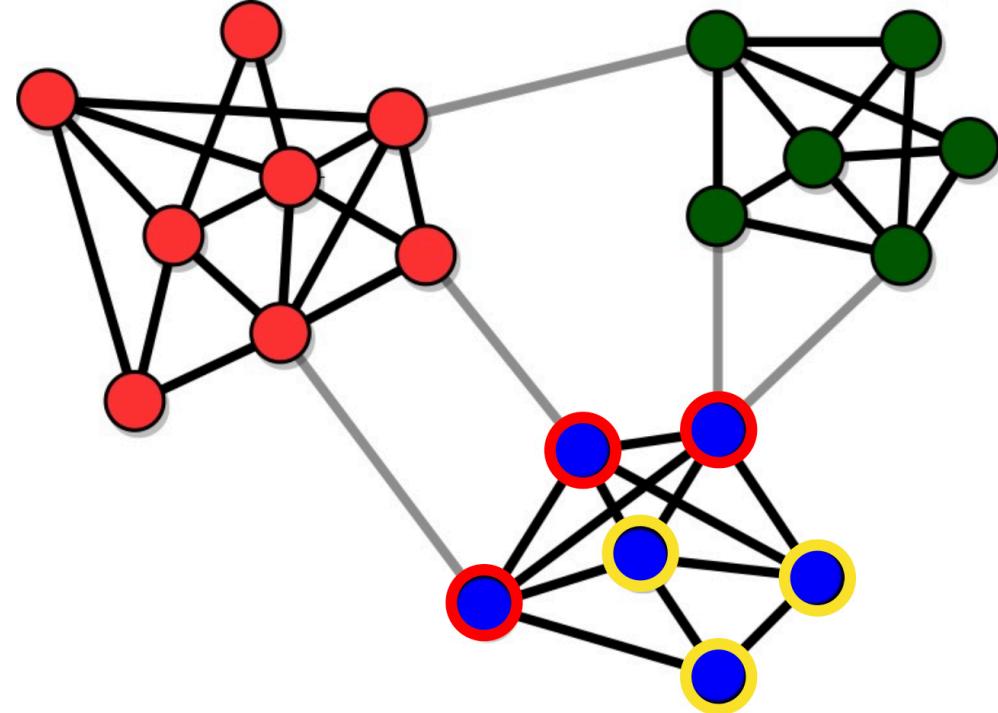
## Community structure

- Example: Twitter users with strong political preferences tend to follow those aligned with them and not to follow users with different political orientation
- Other examples: social circles in social networks, functional modules in protein interaction networks, groups of pages about the same topic on the Web, etc.



## Why study communities?

- Uncover the organization of the network
- Identify features of the nodes
- Classify the nodes based on their position in the clusters
- Find missing links

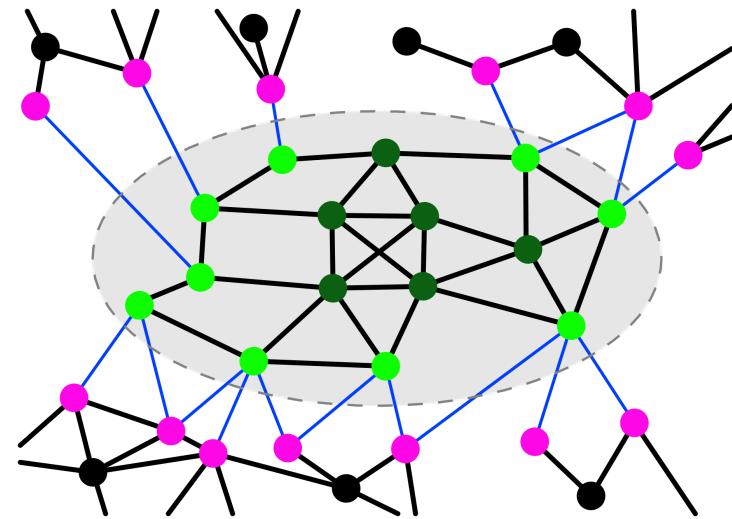


## Basic definitions: variables

- **Internal degree of a node:** number of neighbors of the node in its community
- **External degree of a node:** number of neighbors of the node outside of its community
- **Community degree:** sum of the degrees of the nodes in the community
- **Internal link density:** ratio between the number of links  $L_C$  inside a community  $C$  and the maximal possible number of links that can lie inside  $C$ :

$$\delta_C^{int} = \frac{L_C}{L_c^{max}} = \frac{L_C}{\binom{N_C}{2}} = \frac{2L_C}{N_C(N_C - 1)}$$

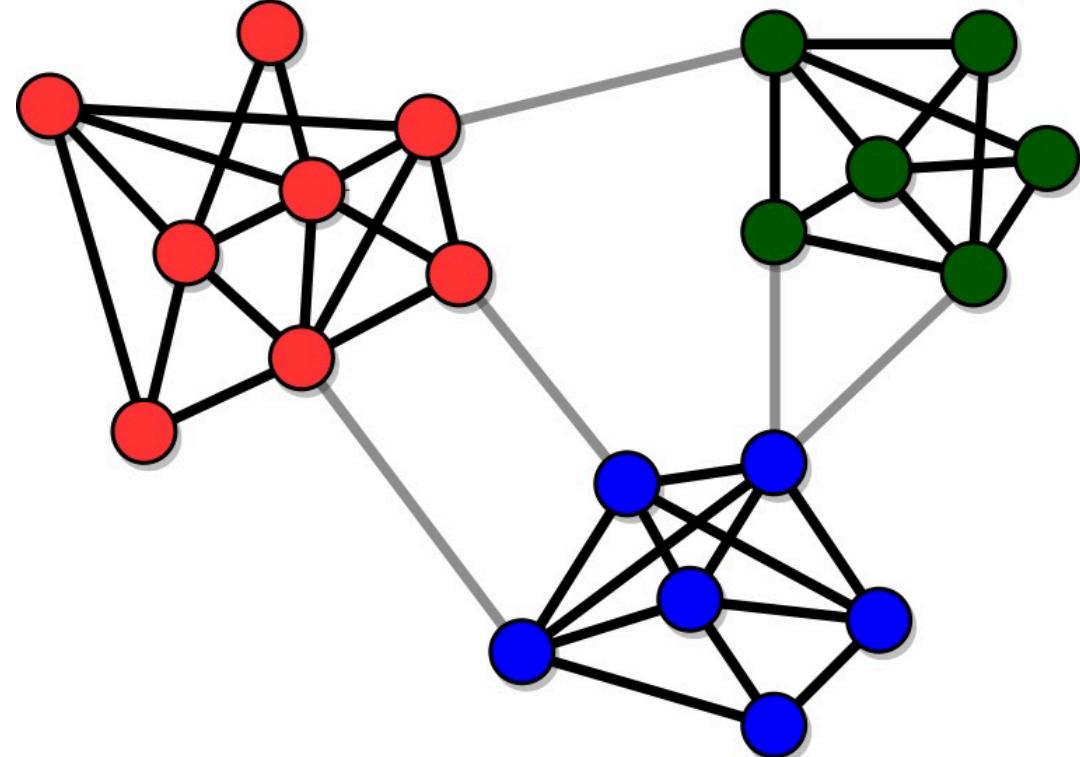
- where  $N_C$  is the number of nodes in  $C$



## Basic definitions: community

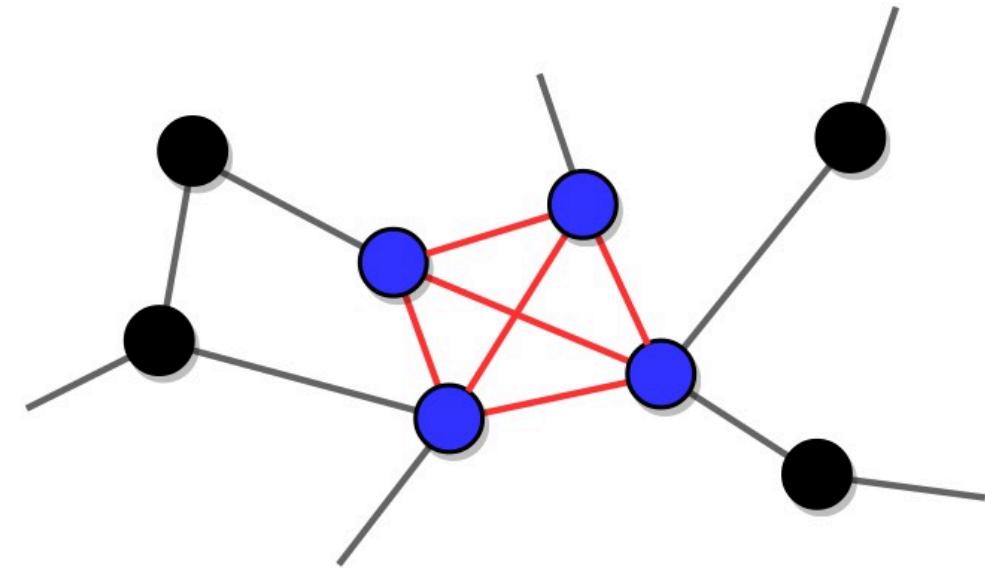
Two main features:

- **High cohesion:** communities have many internal links, so their nodes stick together
- **High separation:** communities are connected by a few links



## Community definitions based on cohesion

- **Principle:** focus on the cluster's properties, disregarding the rest of the network
- **Example:** clique — all internal links are there (maximal cohesion)
- **Problem:** nodes are connected to all others in the cluster, whereas in real communities, they have different roles, which is reflected in heterogeneous linking patterns



# Community definitions based on cohesion versus separation

## Principle:

Definition tends to achieve high cohesion and high separation

## Popular idea:

The number of internal links exceeds the number of external links

## Two concepts:

- **Strong community:** subnetwork such that the internal degree of each node is greater than its external degree
- **Weak community:** subnetwork such that the sum of the internal degrees of its nodes is greater than the sum of their external degrees

## Community definitions based on cohesion versus separation

**A strong community is also a weak community:**

If the inequality between internal and external degree holds for each node, then it must hold for the sum over all nodes

**A weak community is not a strong community, in general:**

If the inequality between internal and external degree holds for the sum, it may be violated for one or more nodes

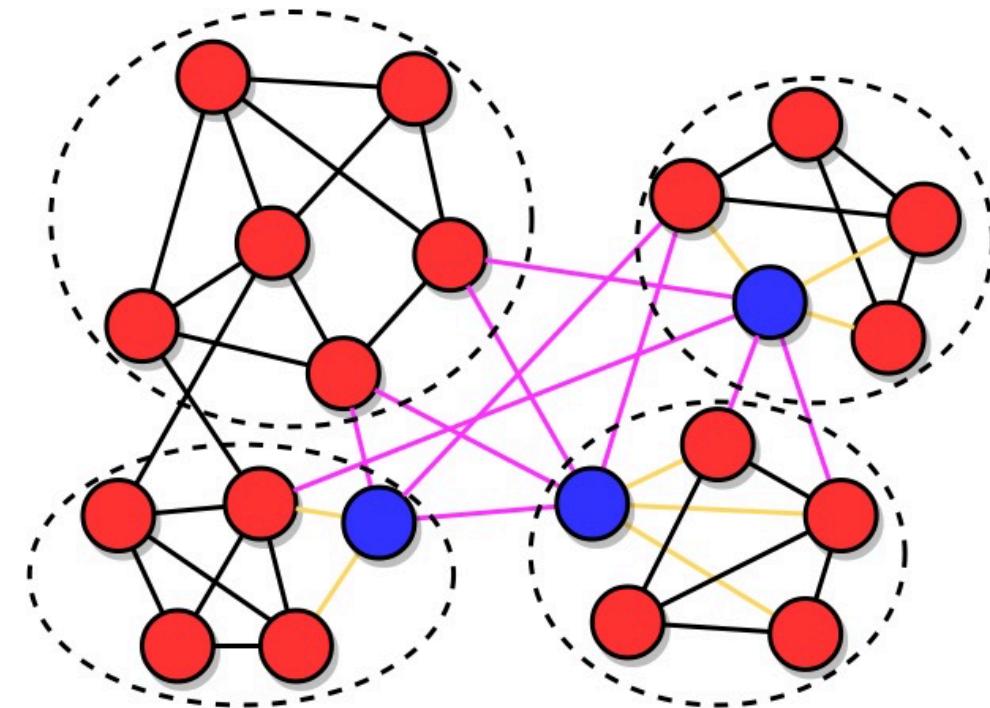
**Problem:**

In the definition of strong and weak community, one compares a subnetwork with the rest of the network. It makes more sense to **compare subnetworks to each other!**

## Community definitions based on cohesion versus separation

Less stringent definitions of strong and weak community:

- **Strong community:** subnetwork such that each node has more neighbors inside it than in any other community
- **Weak community:** subnetwork such that the sum of the internal degrees of its nodes exceeds the total number of neighbors that the nodes have in any other community



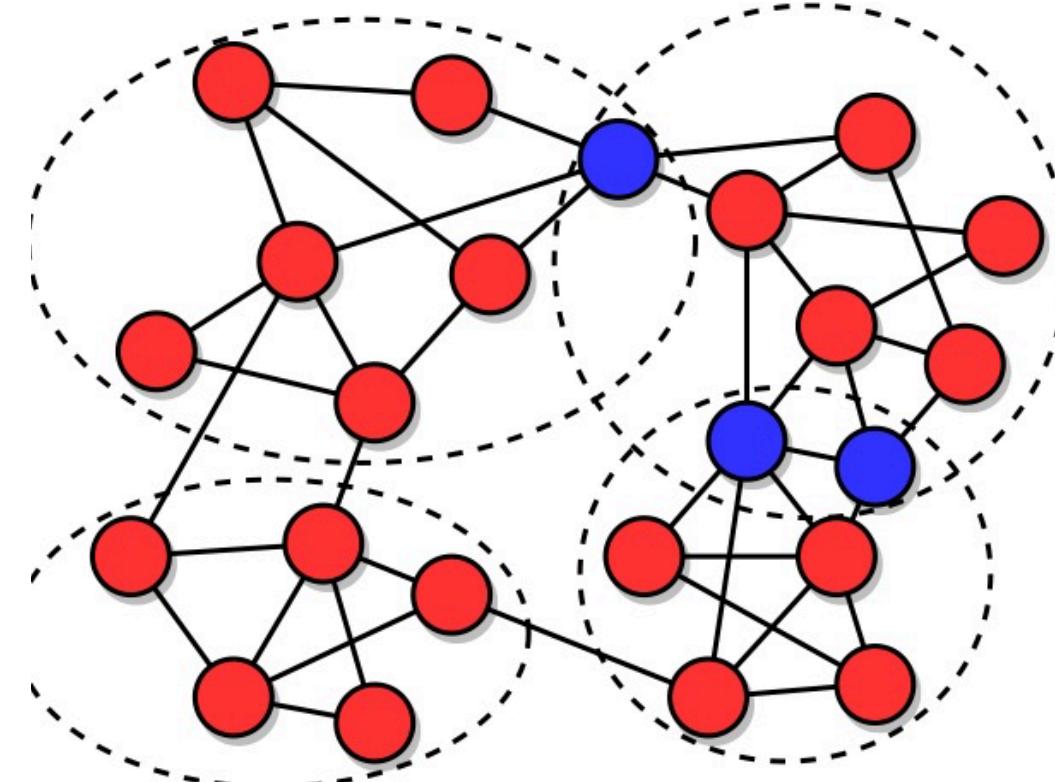
# Partitions

- The number of partitions of  $n$  objects is the **Bell number**  $B_n$
- The Bell number grows **faster than exponentially with  $n$**
- **Conclusion:** it makes no sense to look for interesting community structures by exploring the whole space of partitions! A smart exploration of the partition space must be performed.

$n$	$B_n$
1	1
2	2
3	5
4	15
5	52
6	203
7	877
8	4140
9	21147
10	115975
11	678570
12	4213597
13	27644437
14	190899322
15	1382958545

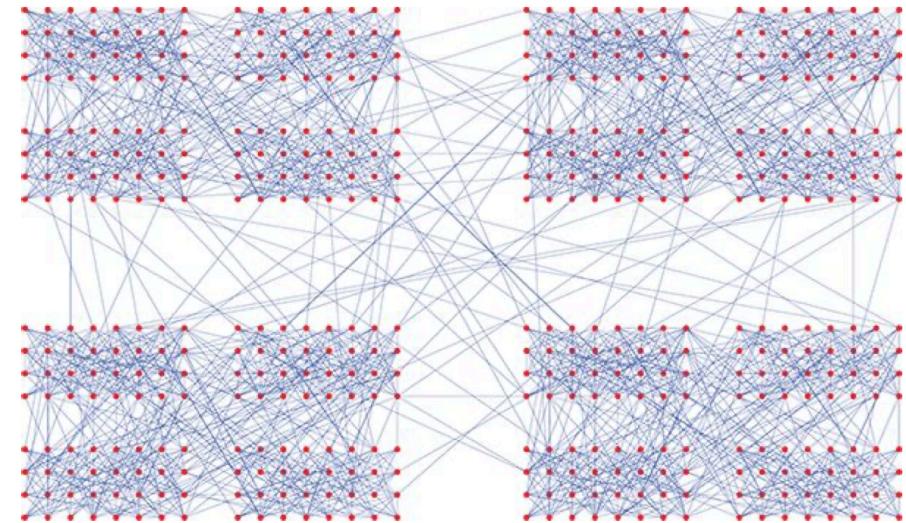
## Overlapping communities

- Communities in many real networks **overlap**
- A division of a network into overlapping communities is called **cover**
- The number of possible covers of a network is **far higher** than the number of partitions due to the many ways clusters can overlap



## Hierarchical communities

- If the network has multiple levels of organization, its communities could form a **hierarchy**, with small communities within larger ones
- **Example:** branches in a company, in turn divided into departments
- **All hierarchical partitions are meaningful:** a good clustering algorithm should detect all of them





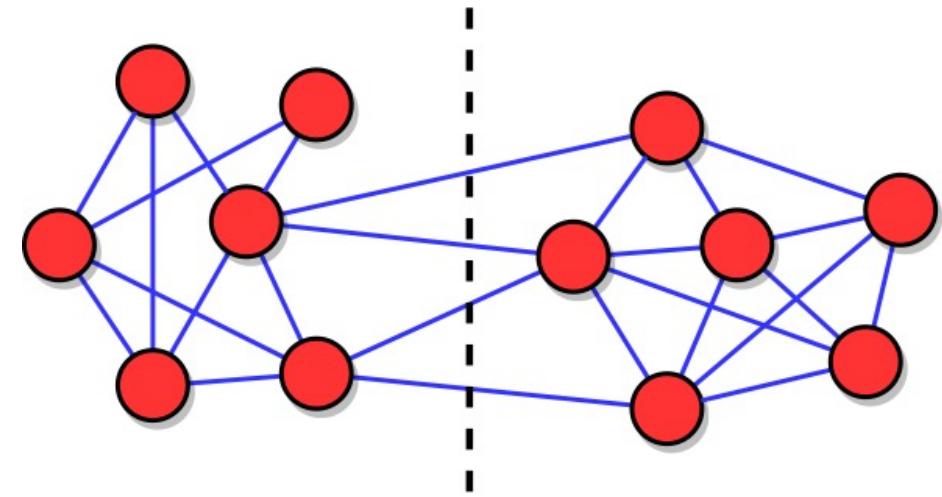
# Related problems

# Clusters

- Networks are made of **tightly-knit regions** connected by means of sparser connections
- Now we have a more formal definition of such regions, along with some other useful measures:
  - clustering coefficients / triadic closure
  - (local) bridges / neighborhood overlap / betweenness
  - cores / k-cores / k-shells
- The problem of finding denser regions in a network is called: **graph partitioning or community detection** (with some differences)
- Another related problem is **data clustering**

## Related problems: network partitioning

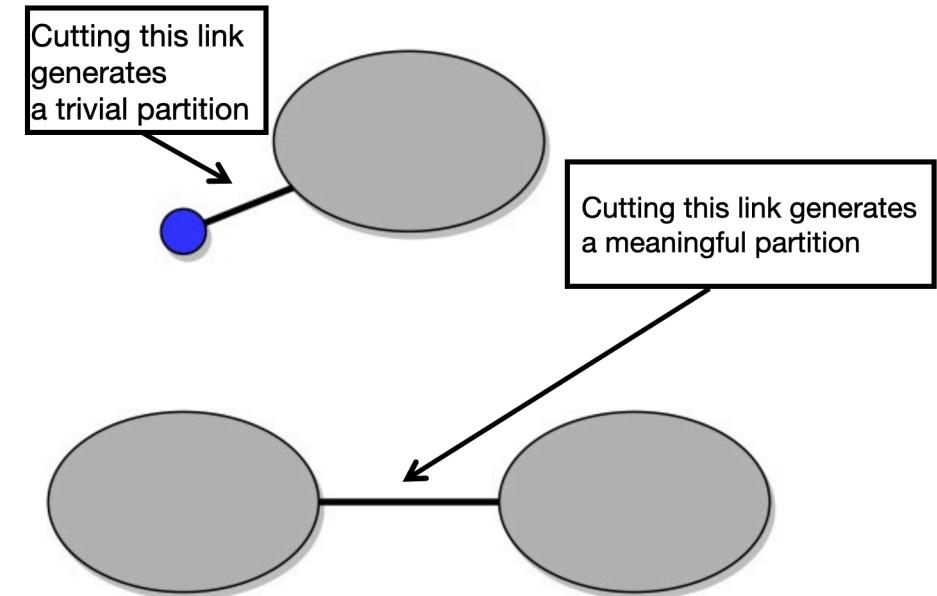
- **The problem:** dividing the nodes of a network into a number of groups of predefined size, such that the number of links between the groups is minimal
- The number of links between groups is called **cut size**



# Related problems: network partitioning

## Problems:

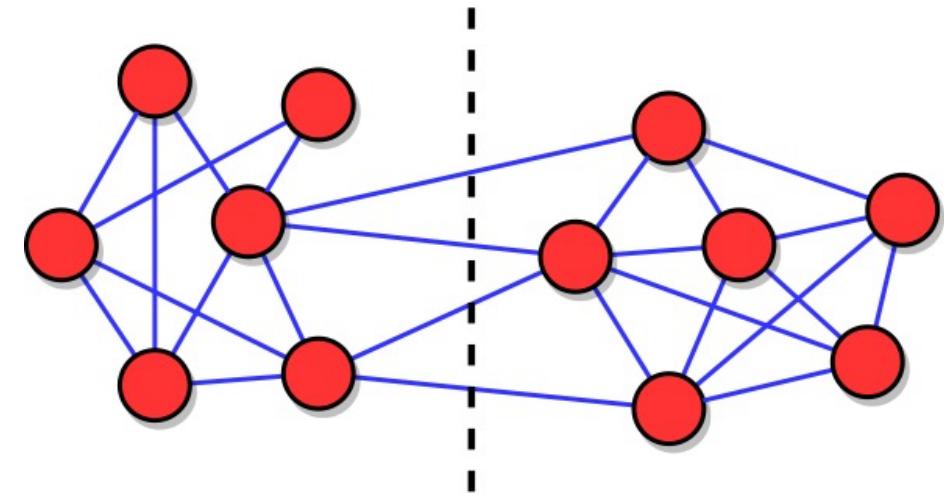
- If the number of clusters is not given beforehand, the trivial solution is a single cluster including everything
- If the size of the clusters is not indicated, there may be trivial solutions by removing the nodes with lowest degree



# Network bisection

## The problem:

Dividing the nodes of a network into **two groups of equal size**, such that the number of **links between the groups is minimal**

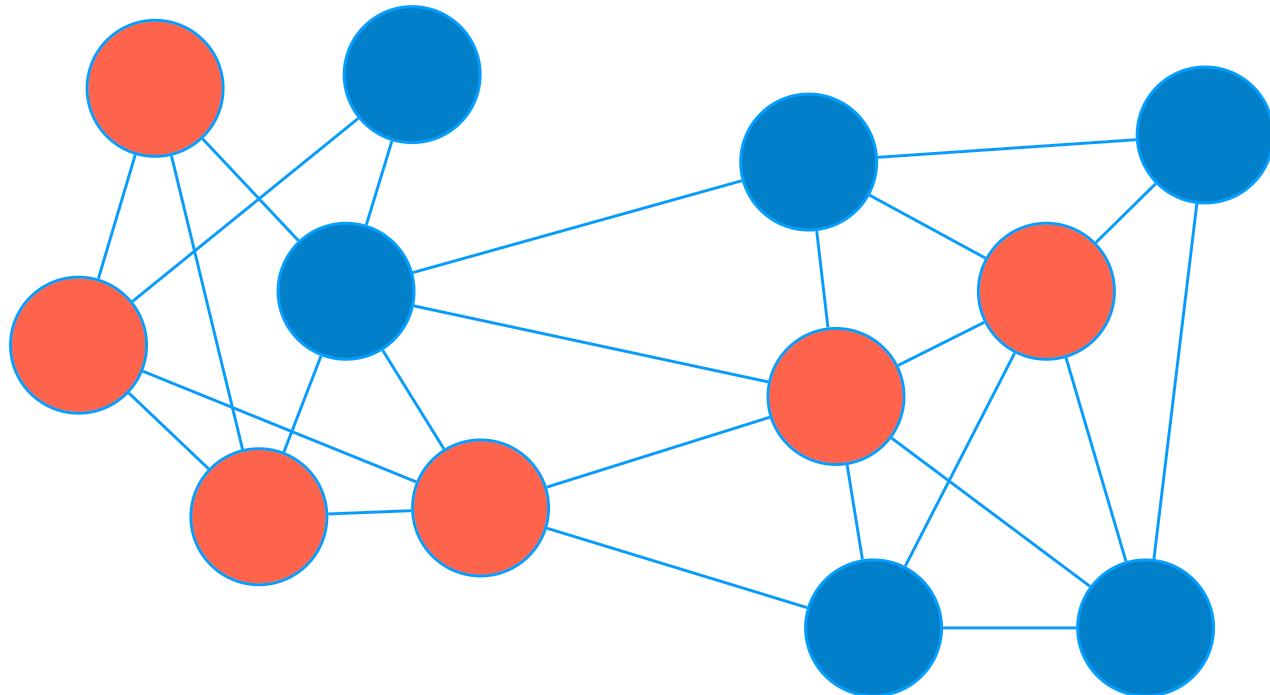


# Kernighan-Lin algorithm

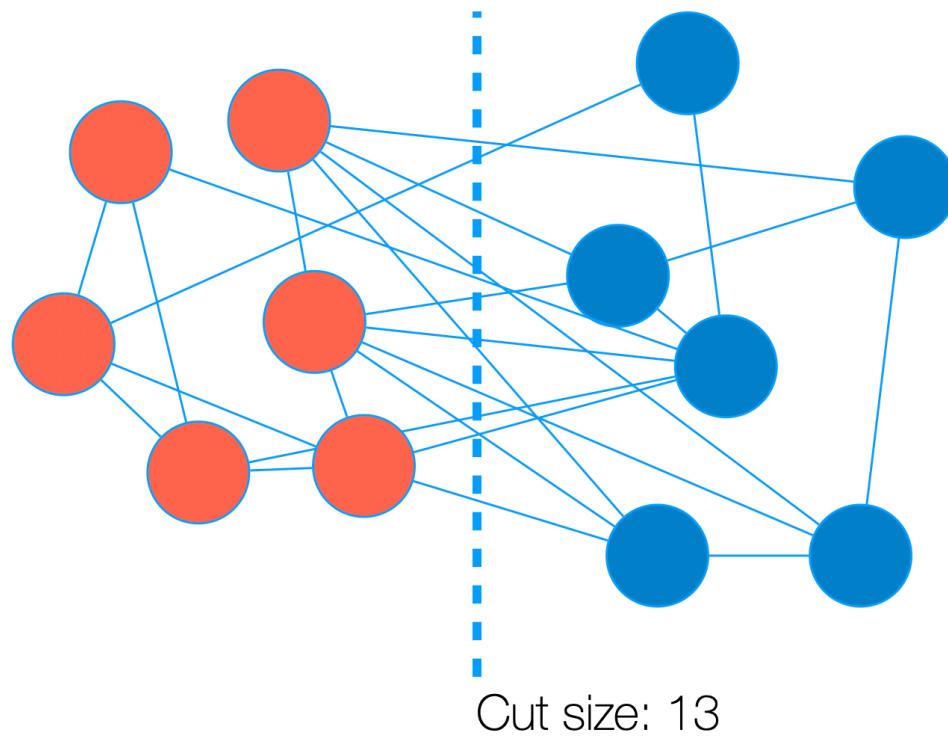
## Procedure:

1. Split in two groups  $A$  and  $B$  of predefined sizes  $N_A$  and  $N_B$ , e.g., by randomly assigning nodes to either group (for bisection  $N_A = N_B$ )
2. For each pair of nodes  $i, j$ , with  $i \in A$  and  $j \in B$ , compute the variation in cut size between the current partition and the one obtained by swapping  $i$  and  $j$
3. The pair of nodes  $i$  and  $j$  yielding the largest decrease in cut size is selected and swapped. This pair of nodes is locked; they will not be touched again during this iteration
4. Repeat steps 2 and 3 until no more swaps of unlocked nodes yield a decrease in cut size. This yields a new bipartition, that is used as a starting configuration for the next iteration

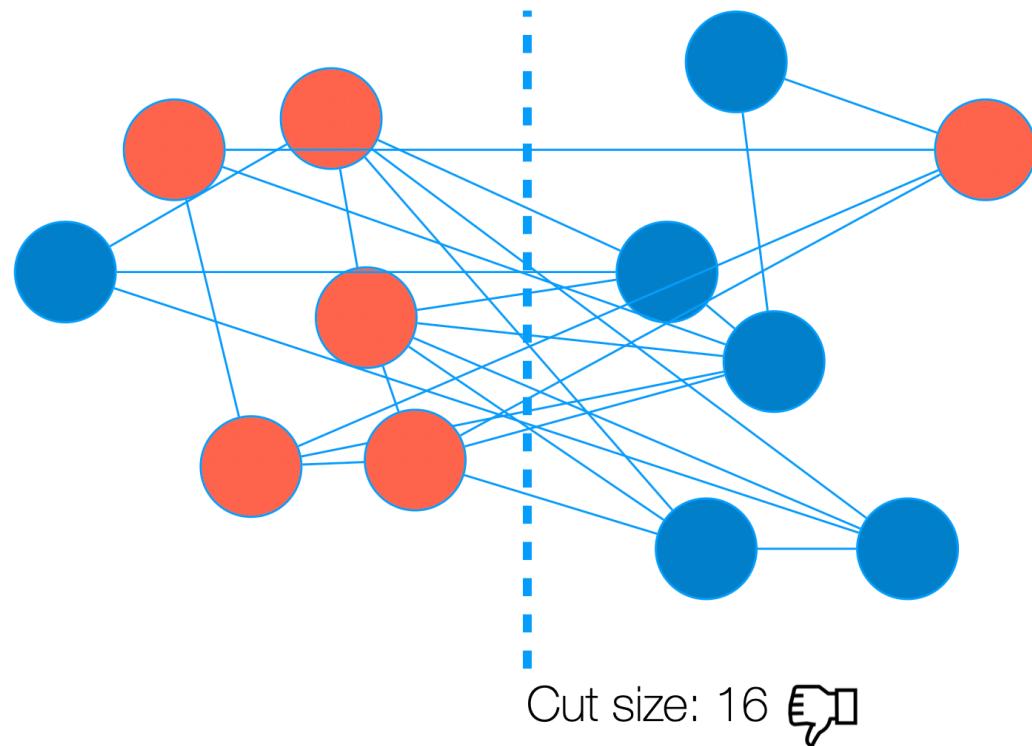
## Kernighan-Lin algorithm: step 1



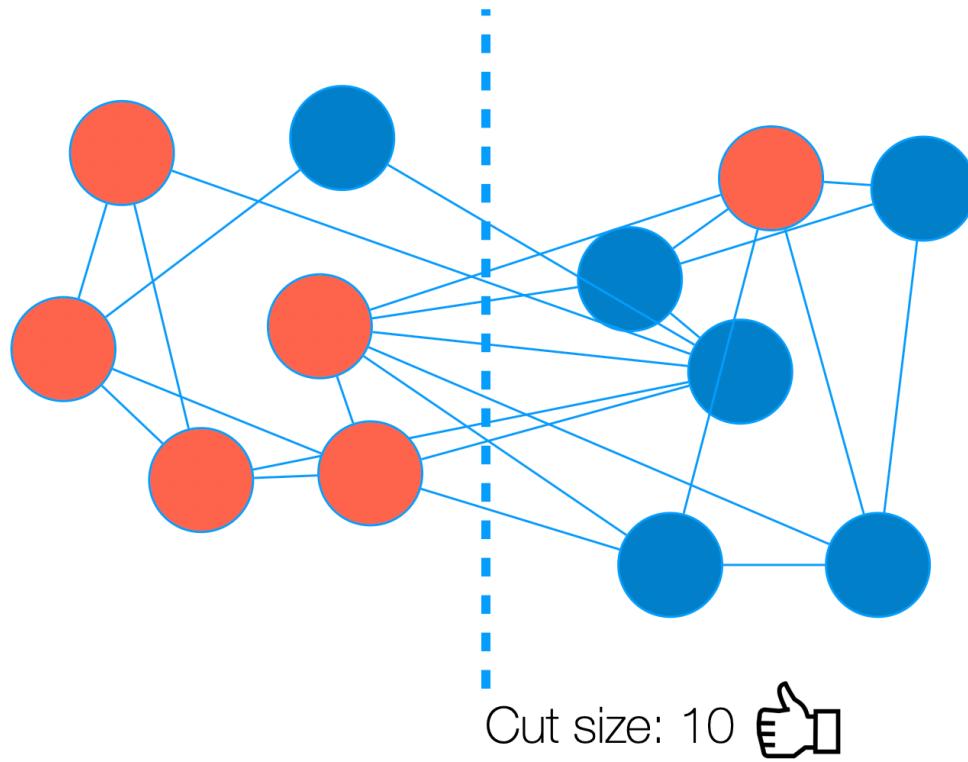
## Kernighan-Lin algorithm: step 1



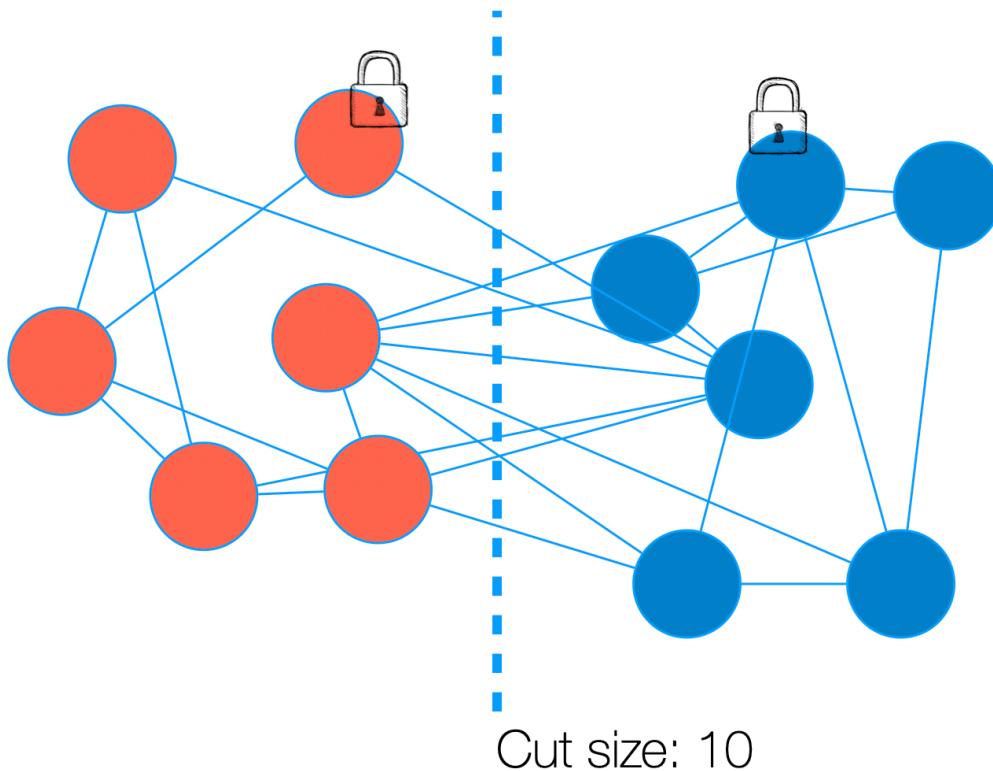
## Kernighan-Lin algorithm: step 2



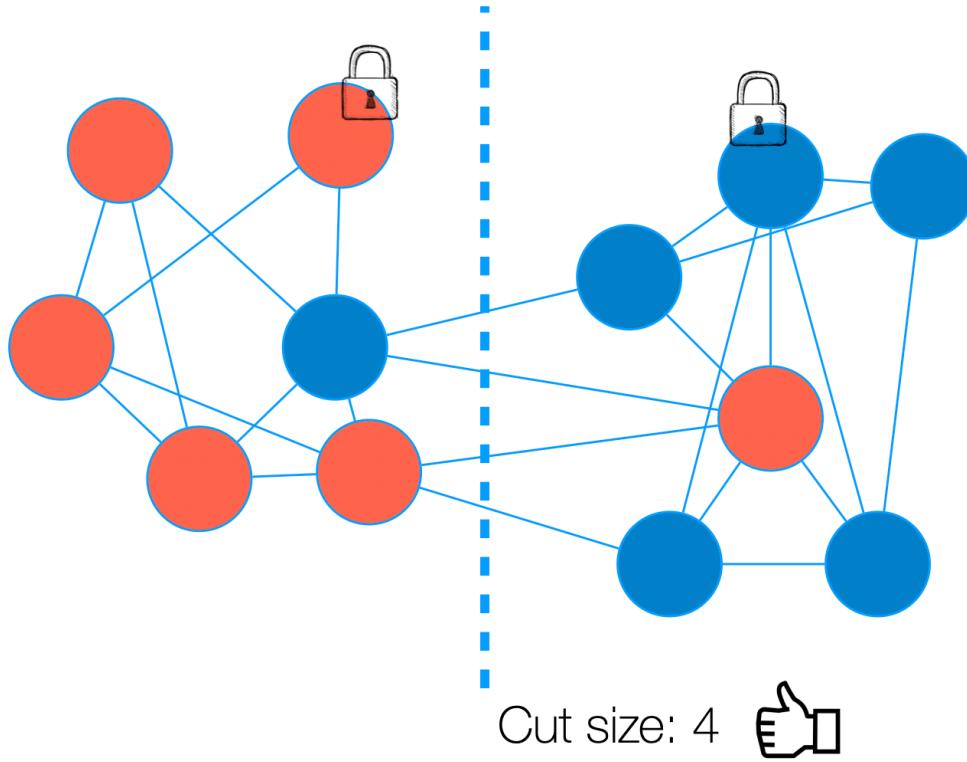
## Kernighan-Lin algorithm: step 2



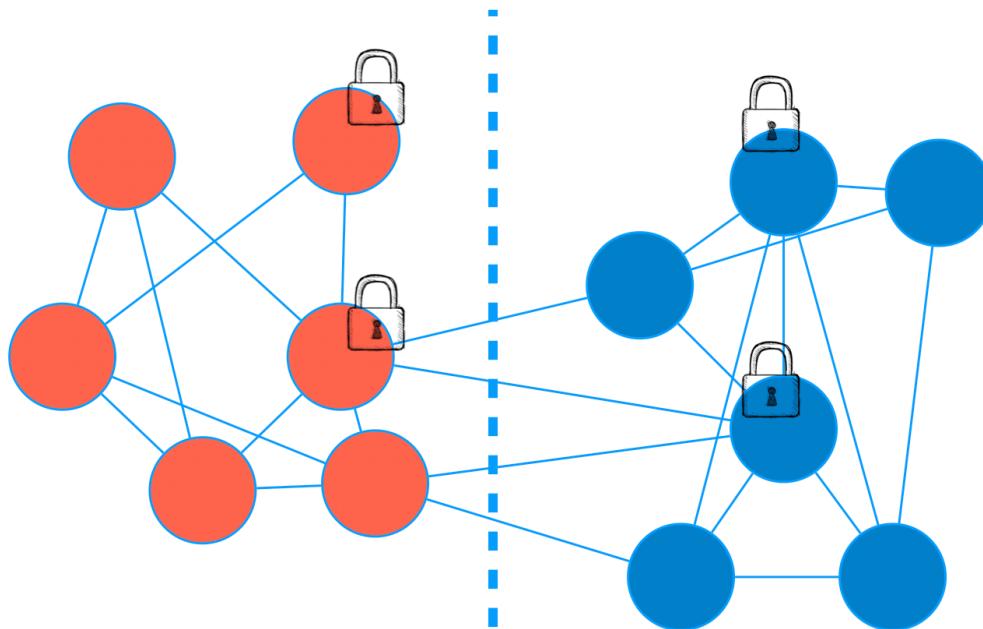
## Kernighan-Lin algorithm: step 3



## Kernighan-Lin algorithm: step 2



## Kernighan-Lin algorithm: step 3



Cut size: 4

## Kernighan-Lin algorithm

- **Convergence:** The procedure ends when the cut size of partitions obtained after consecutive iterations is the same, meaning that the algorithm is unable to improve the result
- The algorithm can be applied to minimize the cut size of partitions into more than two clusters, by swapping nodes between pairs of clusters
- **Problem:** **the choice of the initial partition heavily affects the final result.** The larger the cut size of the initial partition, the worse the final solution and the longer the time to reach convergence
- **Solution:** creating multiple random partitions and picking the one with the lowest cut size as initial partition

## Kernighan-Lin algorithm

- The described procedure is **greedy**, in that at each step one looks for the partition with the smallest cut size. Because of that, the algorithm gets stuck in **local optima**, i.e., solutions whose cut size is not as low as it can be
- The problem can be mitigated by occasionally allowing swaps of nodes that increase the cut size
- The Kernighan-Lin algorithm is widely applied as a post-processing technique, to improve partitions delivered by other methods. Such partitions can be used as starting points for the method, which might return solutions with lower cut size

## Network partitioning: limits

- Clusters have to be well-separated, but they do not need to have high internal link density
- Clusters found via graph partitioning are not communities, in general
- The number of clusters must be given as input, but it is usually unknown

In NetworkX:

```
# minimum cut bisection: returns a pair of sets of nodes
partition = nx.community.kernighan_lin_bisection(G)
```

# Data clustering

- Grouping objects based on how similar to each other they are
- Two main classes of algorithms:
  - Hierarchical clustering
  - Partitional clustering

# Hierarchical clustering

- Hierarchical clustering delivers a nested set of partitions
- Main ingredient: **similarity measure**
- Examples:
  - In a social network it could indicate how close the profiles of two people are based on their interests
  - If nodes are embedded in space (i.e., they are points in a metric space), the (dis)similarity between two nodes can be expressed by their distance
  - If nodes are not embedded in space, similarity measures can be derived from the network structure

## Similarity: structural equivalence

- **Concept:** nodes are similar if their neighbors are similar

$$S_{ij}^{SE} = \frac{\text{number of neighbors shared by } i \text{ and } j}{\text{total number of nodes neighboring only } i, \text{ only } j \text{ or both}}$$

- **Examples:**

- If the neighbors of  $i$  and  $j$  are  $(v_1, v_2, v_3)$  and  $(v_1, v_2, v_4, v_5)$ , respectively,  $S_{ij} = 2/5 = 0.4$ , because there are two common neighbors ( $v_1$  and  $v_2$ ) out of five distinct neighbors in total ( $v_1, v_2, v_3, v_4, v_5$ )
- If  $i$  and  $j$  have no neighbors in common,  $S_{ij} = 0$
- If  $i$  and  $j$  have the same neighbors,  $S_{ij} = 1$

## Similarity of node groups

- **Question:** how can we define the similarity  $S_{G_1, G_2}$  between two groups of nodes  $G_1$  and  $G_2$  via the similarity  $S$  between pairs of nodes?
- **Answer:** multiple approaches. The first step is to compute the pairwise similarity  $S_{ij}$  between each node  $i$  in  $G_1$  and each node  $j$  in  $G_2$ .

Then the following strategies can be adopted:

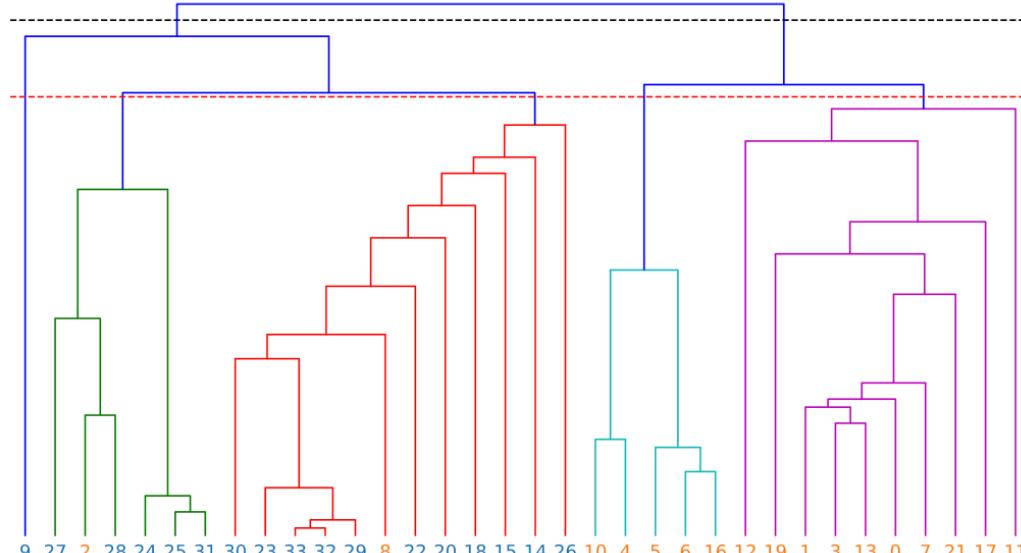
- **Single linkage:** take the maximum pairwise similarity  $\rightarrow S_{G_1, G_2} = \max_{i,j} S_{ij}$
- **Complete linkage:** take the minimum pairwise similarity  $\rightarrow S_{G_1, G_2} = \min_{i,j} S_{ij}$
- **Average linkage:** take the average pairwise similarity  $\rightarrow S_{G_1, G_2} = \langle S_{ij} \rangle_{i,j}$

## Hierarchical clustering

- **Two approaches**
  - Agglomerative hierarchical clustering: partitions are generated by iteratively merging groups of nodes
  - Divisive hierarchical clustering: partitions are generated by iteratively splitting groups of nodes
- **Agglomerative hierarchical clustering:**
  - Start: partition into  $N$  groups, each group consisting of one node
  - At each step the pair of groups with the largest similarity are merged

# Dendograms

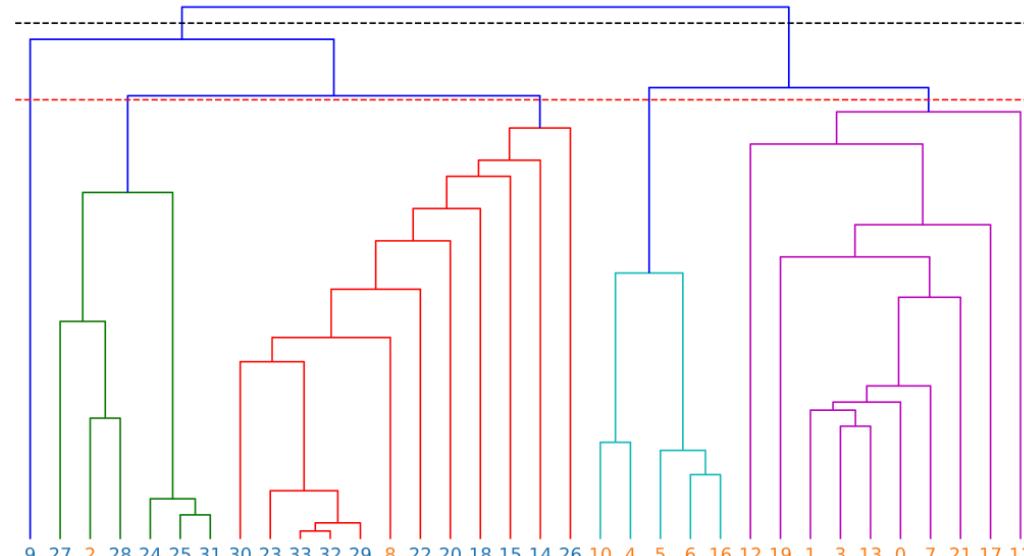
- Outcome: **dendrogram (hierarchical tree)**
- A dendrogram is a compact summary of all partitions created by hierarchical clustering
- Since each merger reduces the number of groups by one, **the total number of partitions is  $N$**



# Dendograms

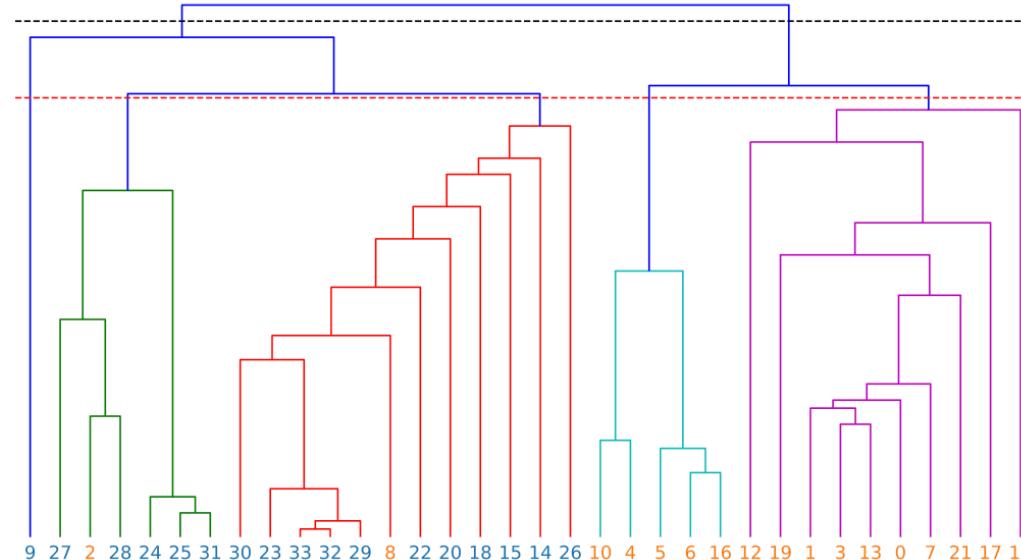
## Features:

- **Bottom: leaves** of the tree, indicated by the labels of the nodes
- Going upwards, pairs of clusters are merged. Mergers are illustrated by horizontal lines joining two vertical lines, each representing a cluster
- The nodes of each cluster can be identified by following the vertical line representing the cluster all the way down



## Dendograms

- Partitions are selected via **horizontal cuts** of the dendrogram: the clusters are the ones corresponding to the vertical lines severed by the cut
- High cuts yield partitions into a few large clusters, low cuts yield partitions into many small clusters
- **Hierarchy:** each partition has clusters including clusters of all partitions lying lower in the dendrogram





# Reading material

References

[ns1] Chapter 6 - Communities



# Q & A

