Graph Mining
SD212
5. Clustering

Thomas Bonald

2018 – 2019

Motivation

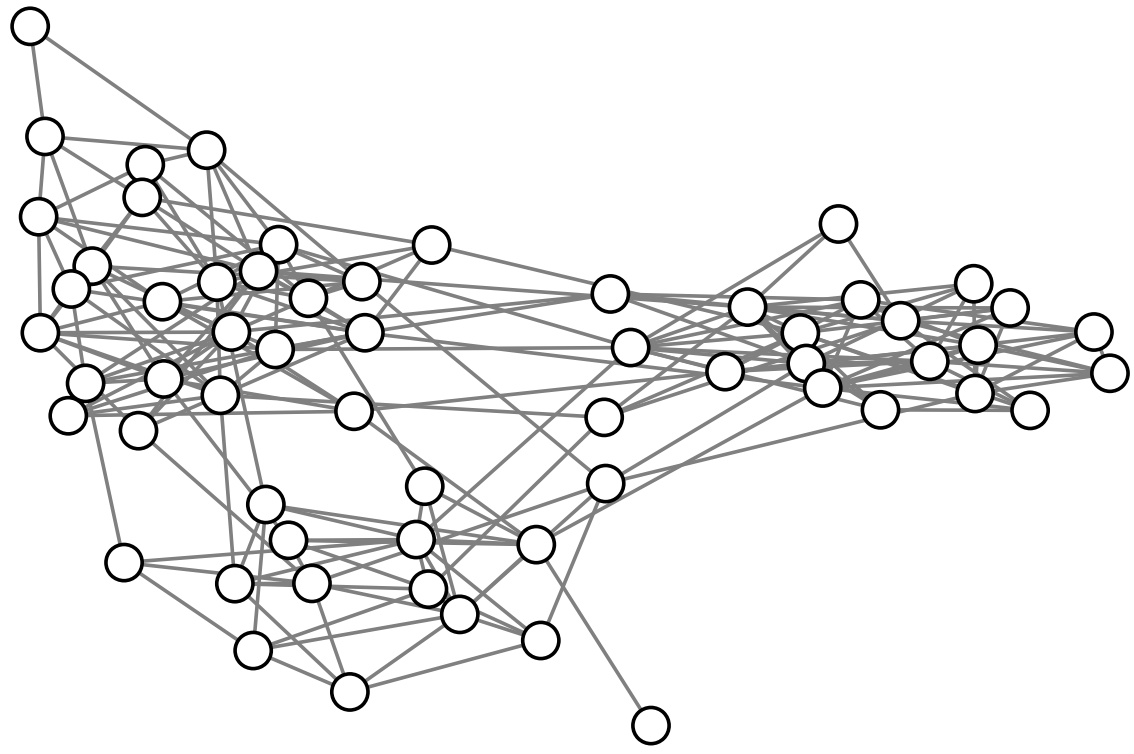
How to identify relevant groups of nodes in a graph?

This is the problem of **graph clustering**, also known as **community detection** in the context of social networks

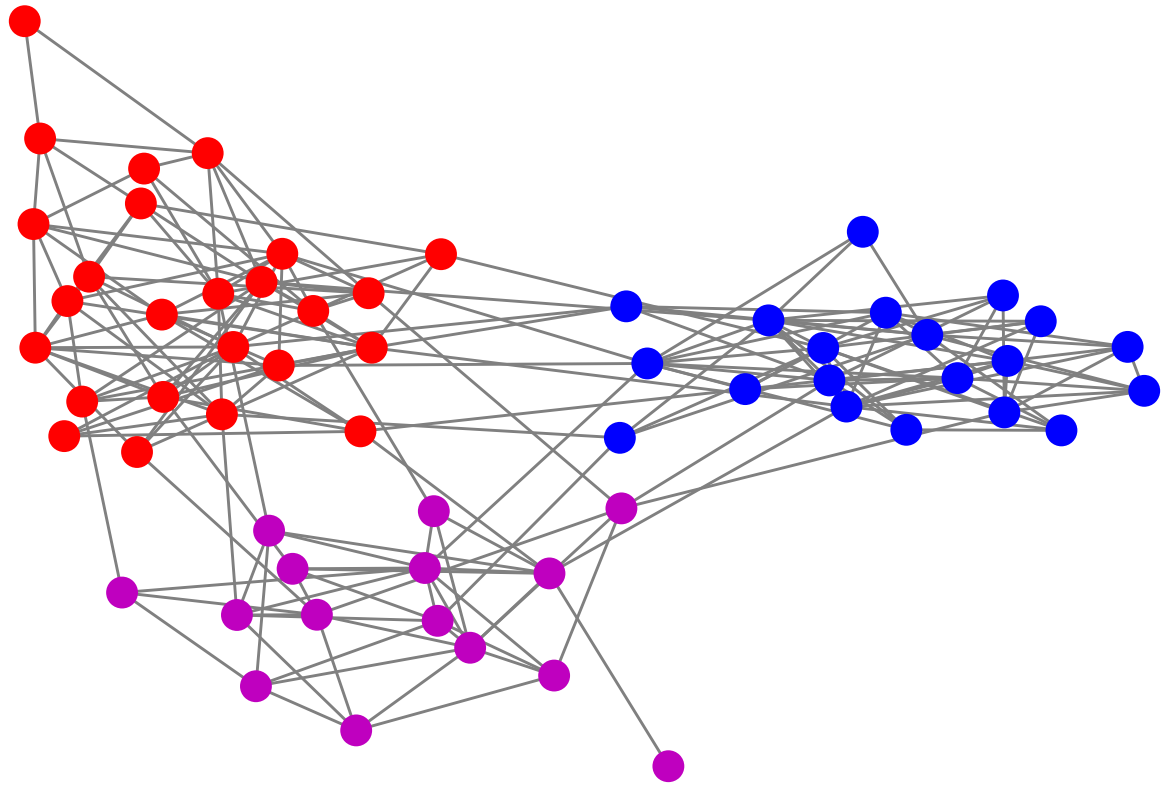
Useful for:

- data visualization
- information retrieval
- content / friend recommendation
- prediction
- anomaly detection

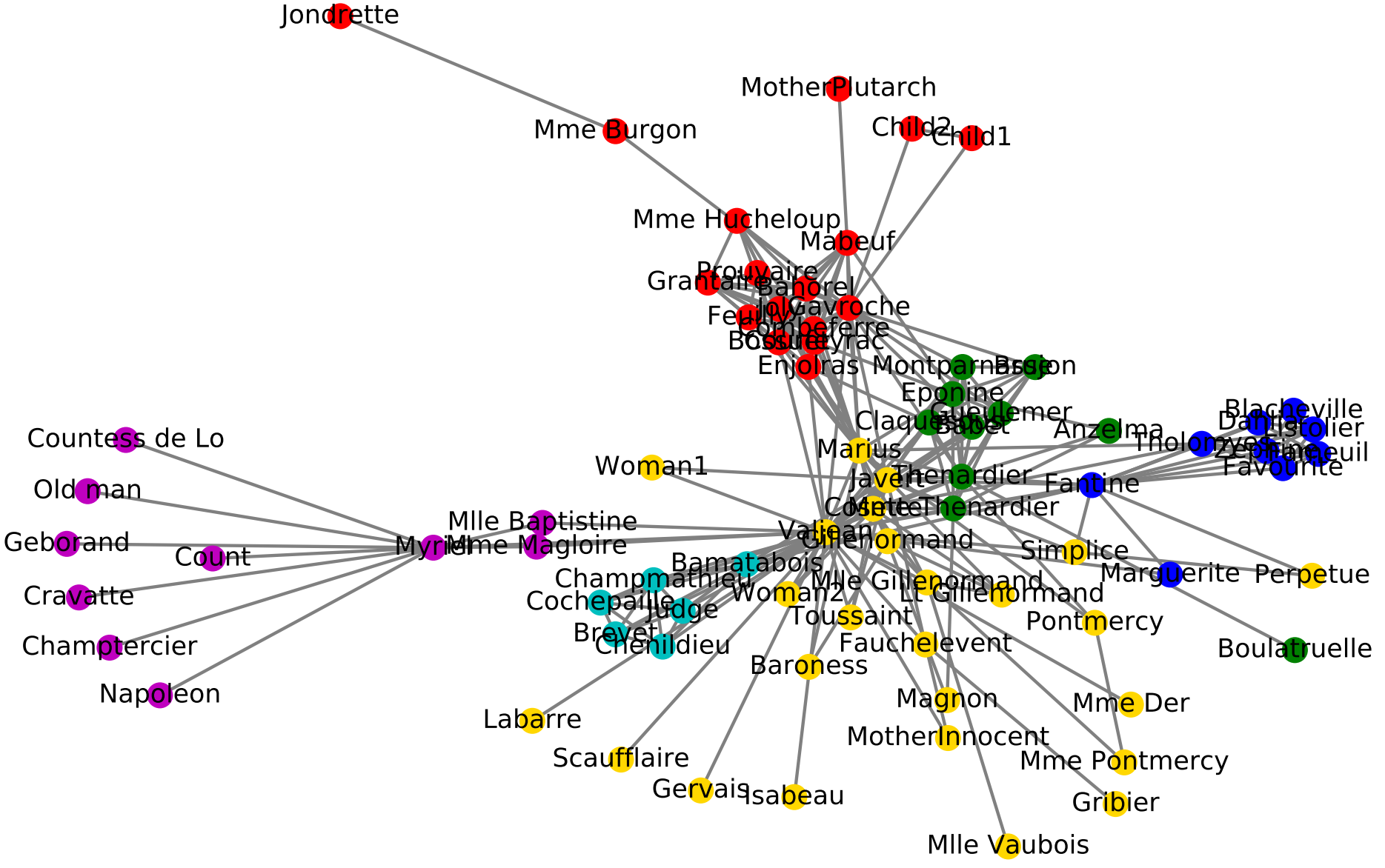
Example



Graph clustering



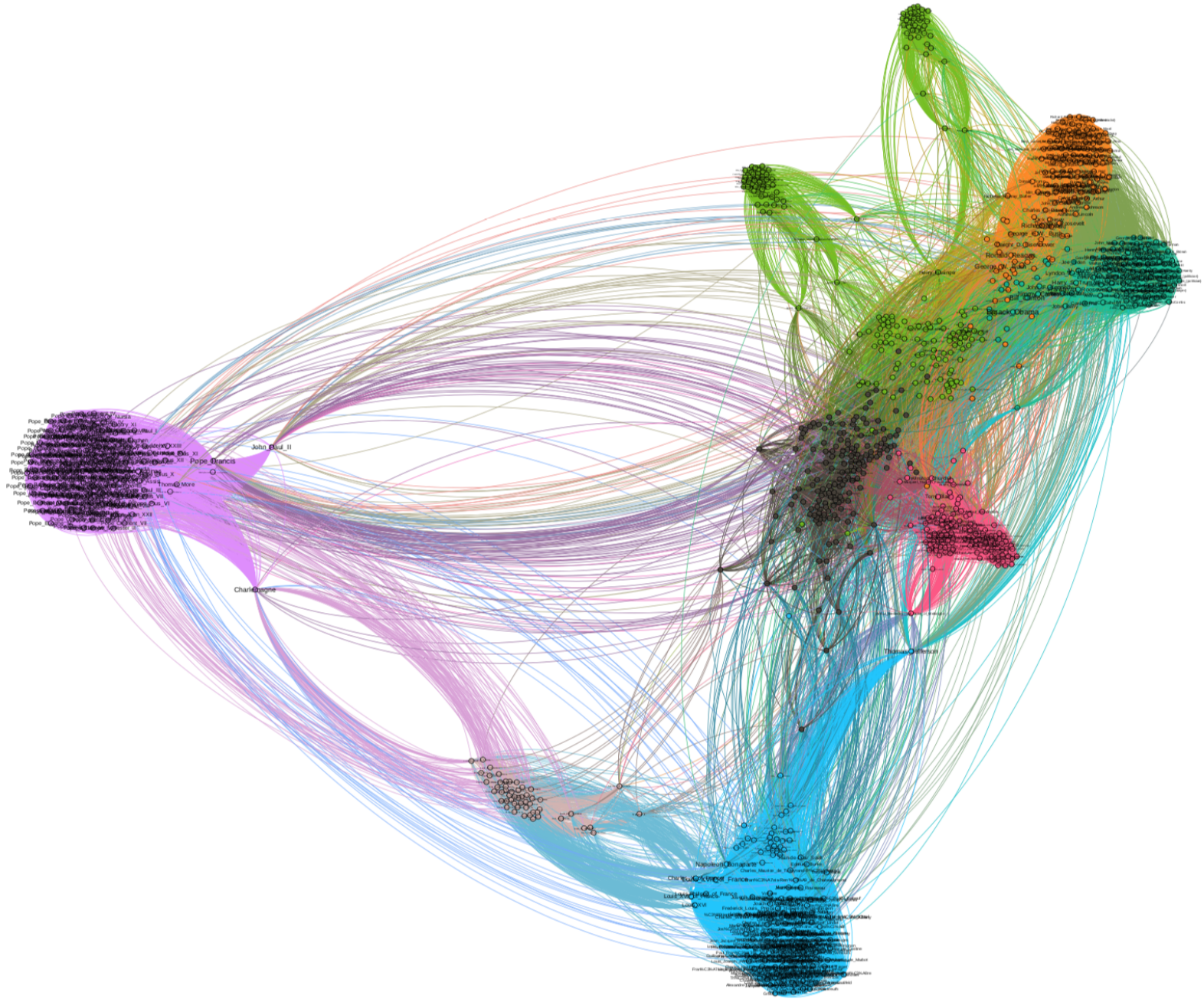
Characters of Les Miserables



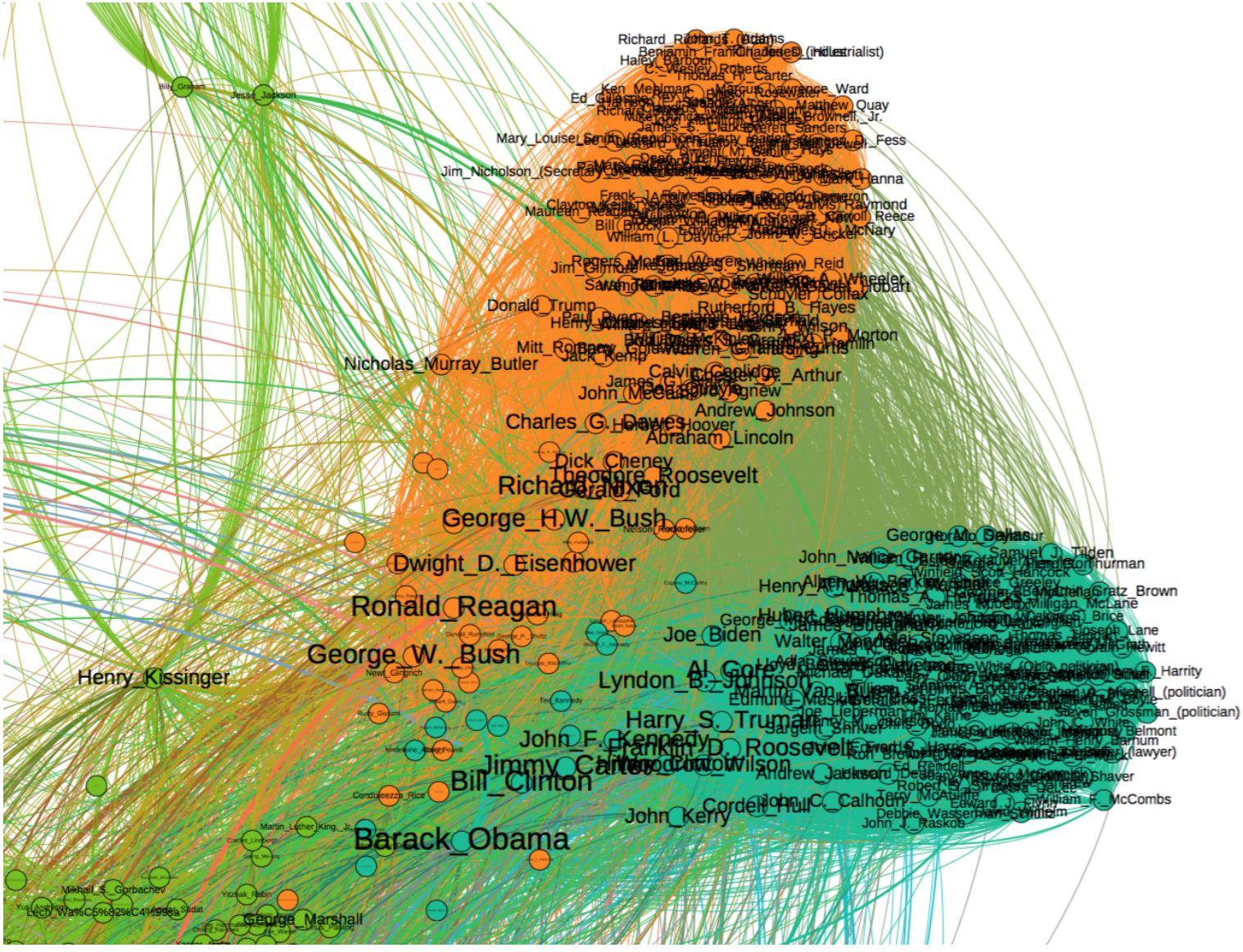
OpenFlights

OpenFlights (higher resolution)

Wikipedia restricted to Political Figures

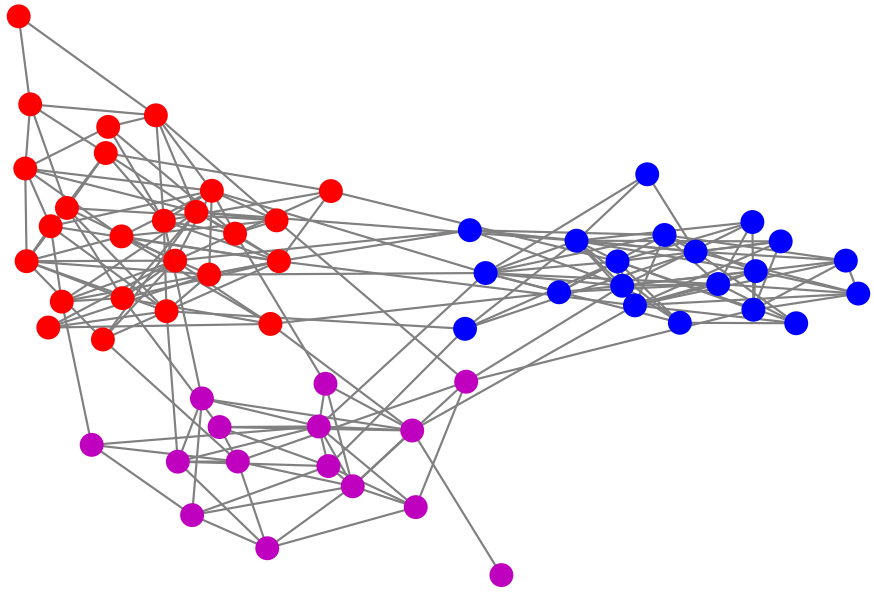


Wikipedia restricted to Political Figures (zoom)



Graph clustering

- The clustering of a graph $G = (V, E)$ of n nodes and m edges is any function $C : V \to \{1, \ldots, K\}$



- In general, K is unknown (unlike K-means) and we look for the best clustering **irrespective** of the value of K
- We first assume that the graph is **undirected**, **unweighted** and **without self-loops**

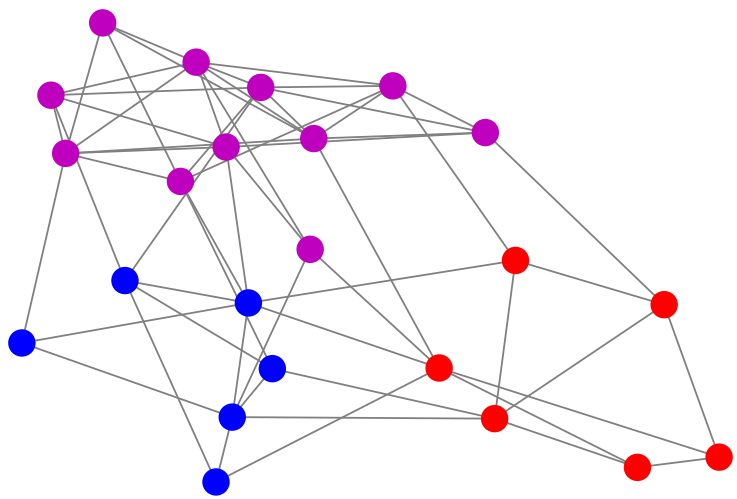
Outline

1. Modularity
2. Resolution
3. The Louvain algorithm
4. Cluster ranking
5. Extensions

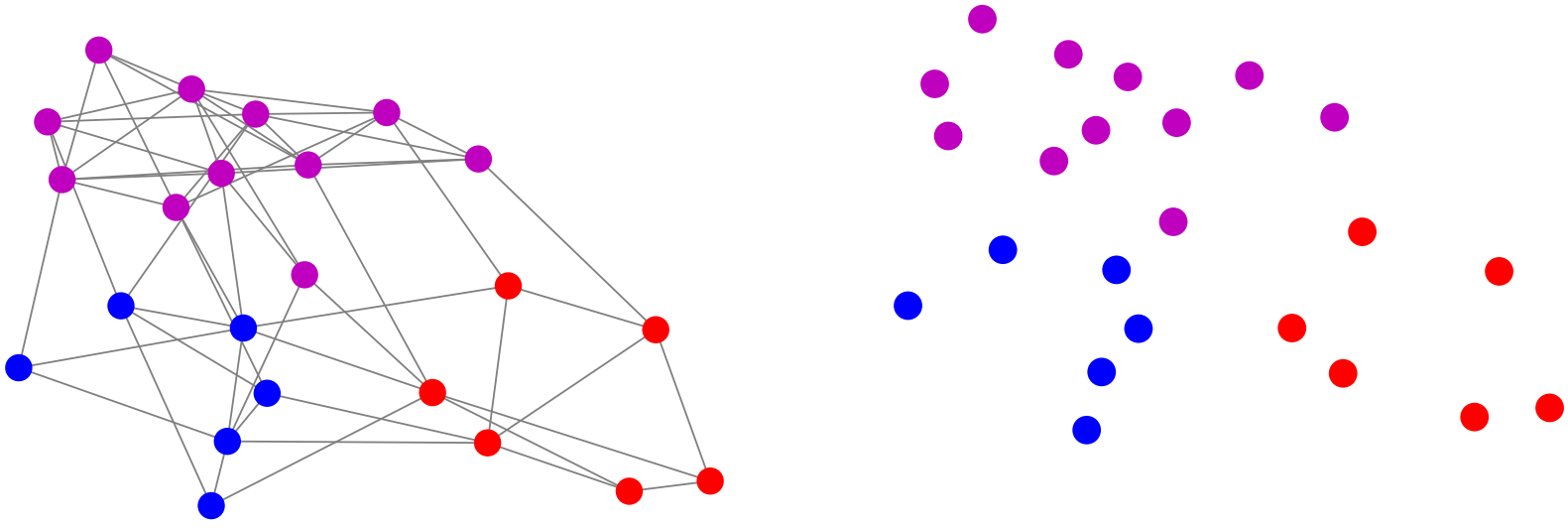
Modularity

The modularity of clustering C is defined by:

$$Q(C) = \frac{1}{2m} \sum_{i,j \in V} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \delta_{C(i),C(j)}$$



A random graph



Node sampling

- The edges of the graph induce a probability distribution on node pairs:

$$p(i,j) = \frac{A_{ij}}{2m}$$

- Marginal distribution:

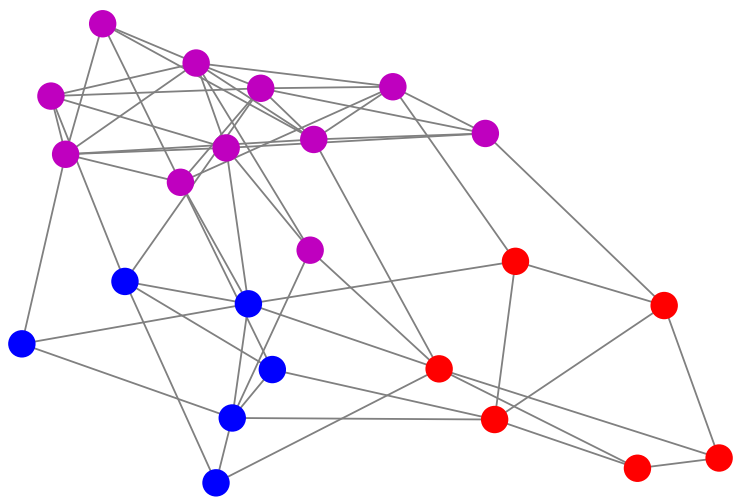
$$p(i) = \sum_{j \in V} p(i,j) = \frac{d_i}{2m}$$

- Modularity:

$$Q(C) = \frac{1}{2m} \sum_{i,j \in V} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \delta_{C(i),C(j)}$$

Modularity and node sampling

$$Q(C) = \sum_{i,j \in V} (p(i,j) - p(i)p(j))\delta_{C(i),C(j)}$$



Cluster sampling

- The distribution on node pairs induces a probability distribution on cluster pairs:

$$\forall k, l, \quad p_C(k, l) = \sum_{i,j:C(i)=k,C(j)=l} p(i, j)$$

- Marginal distribution:

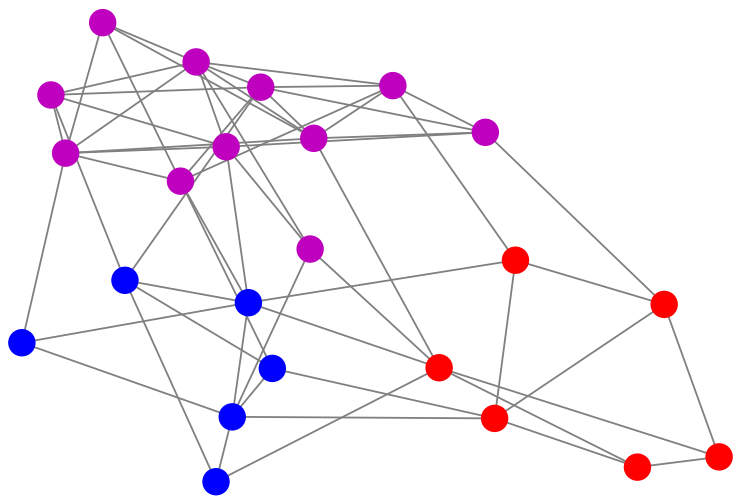
$$\forall k, \quad p_C(k) = \sum_l p_C(k, l) = \sum_{i:C(i)=k} p(i)$$

- Modularity:

$$Q(C) = \sum_{i,j \in V} (p(i, j) - p(i)p(j))\delta_{C(i),C(j)}$$

Modularity and cluster sampling

$$Q(C) = \sum_k (p_C(k,k) - p_C(k)^2)$$



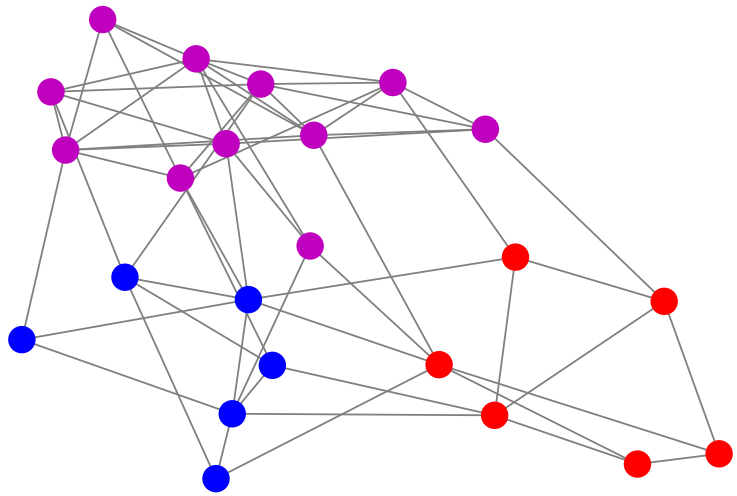
Cluster-level expression of modularity

$$p_C(k,k) = \frac{m_k}{m}, \quad p_C(k) = \frac{v_k}{v}$$

where

- m_k is the number of edges in cluster k
- v_k is the **volume** of cluster k (total degree)

$$Q(C) = \sum_k \frac{m_k}{m} - \sum_k \left(\frac{v_k}{v}\right)^2$$



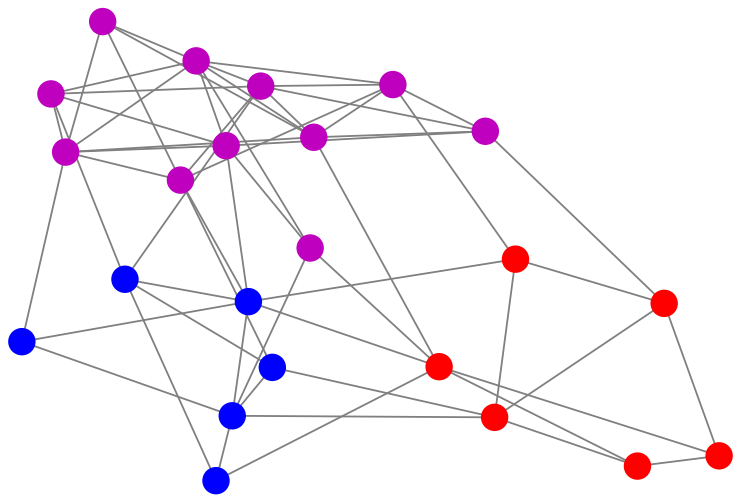
The Simpson index (1949)

- Let $q_1, \ldots, q_K$ be any probability distribution over $\{1, \ldots, K\}$
- Simpson's index:

$$S = \sum_k q_k^2$$

The fit-diversity trade-off

$$Q(C) = \sum_k \frac{m_k}{m} - \sum_k \left(\frac{v_k}{v}\right)^2$$



Outline

1. Modularity
2. **Resolution**
3. The Louvain algorithm
4. Cluster ranking
5. Extensions

The resolution limit of modularity

Recall that

$$Q(C) = \sum_k \frac{m_k}{m} - \sum_k \left(\frac{v_k}{v}\right)^2$$

For a large number of clusters of (approximately) equal weights,

$$\sum_k \left(\frac{v_k}{v}\right)^2 \approx \frac{1}{K} \approx 0$$

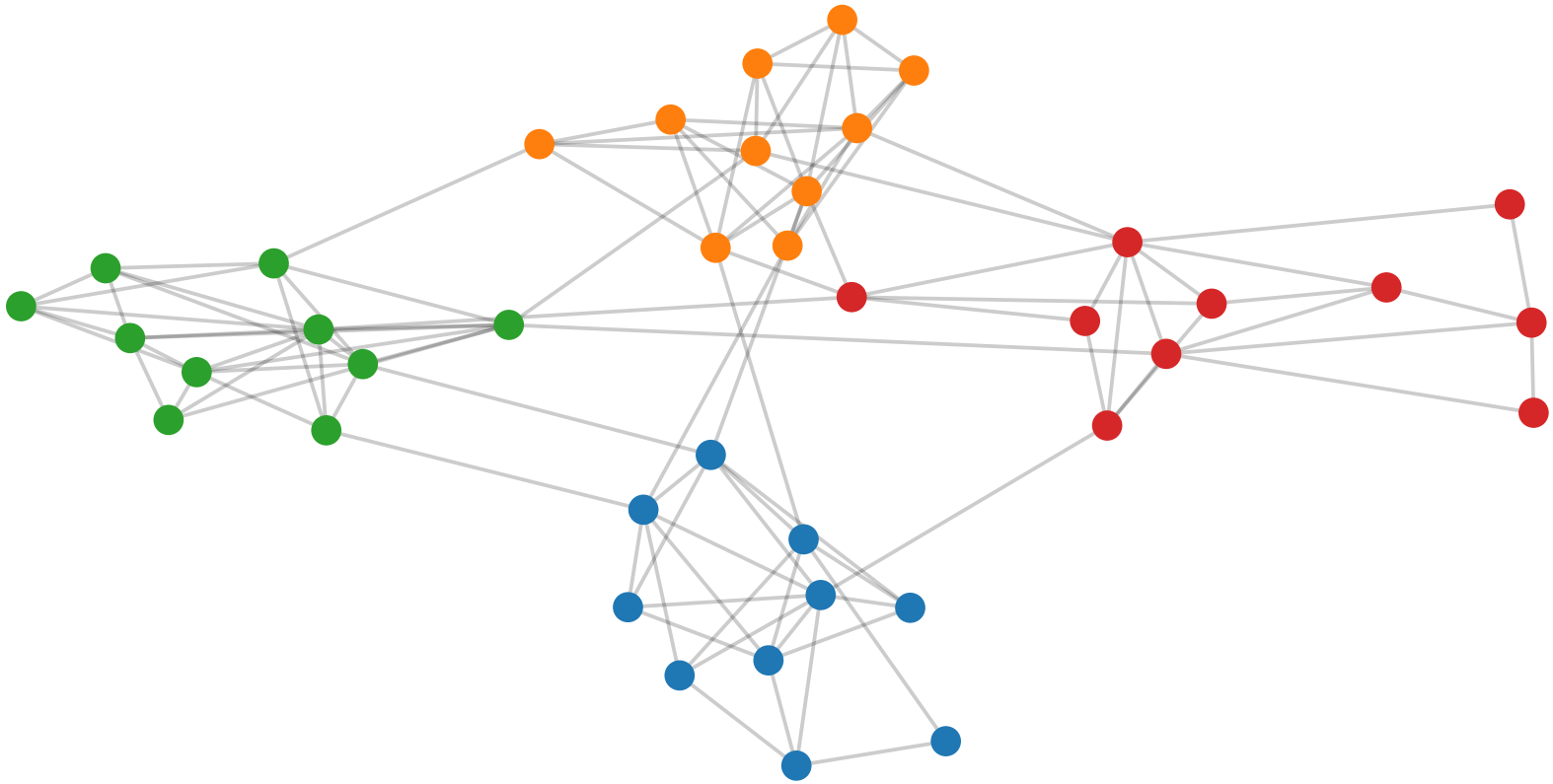
Modularity is not able to detect **high-resolution** clusterings!

Modularity with resolution

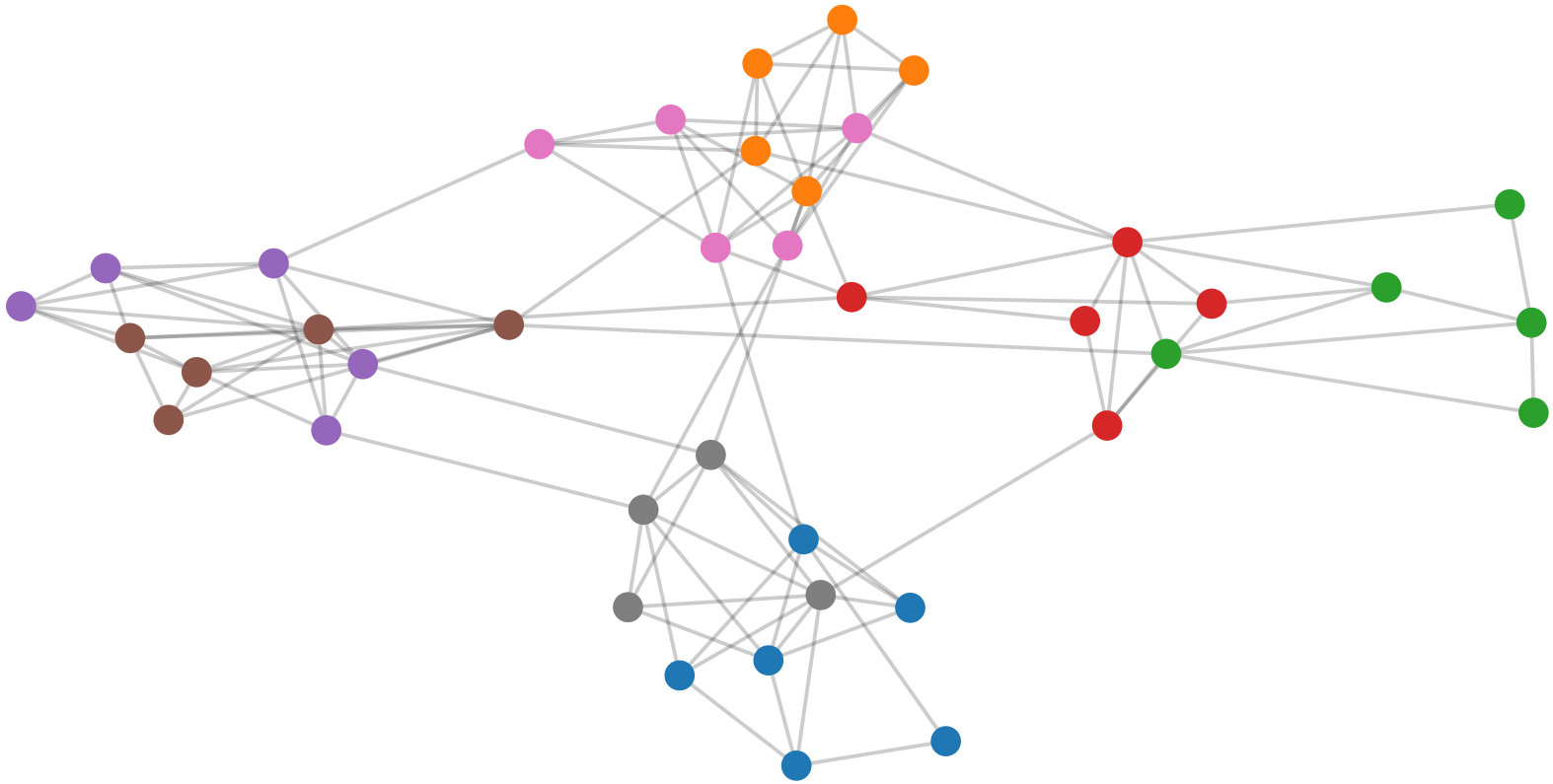
Resolution parameter $\gamma > 0$ that controls the **fit-diversity** trade-off:

$$Q_\gamma(C) = \frac{1}{2m} \sum_{i,j \in V} \left(A_{ij} - \gamma \frac{d_i d_j}{2m} \right) \delta_{C(i),C(j)}$$

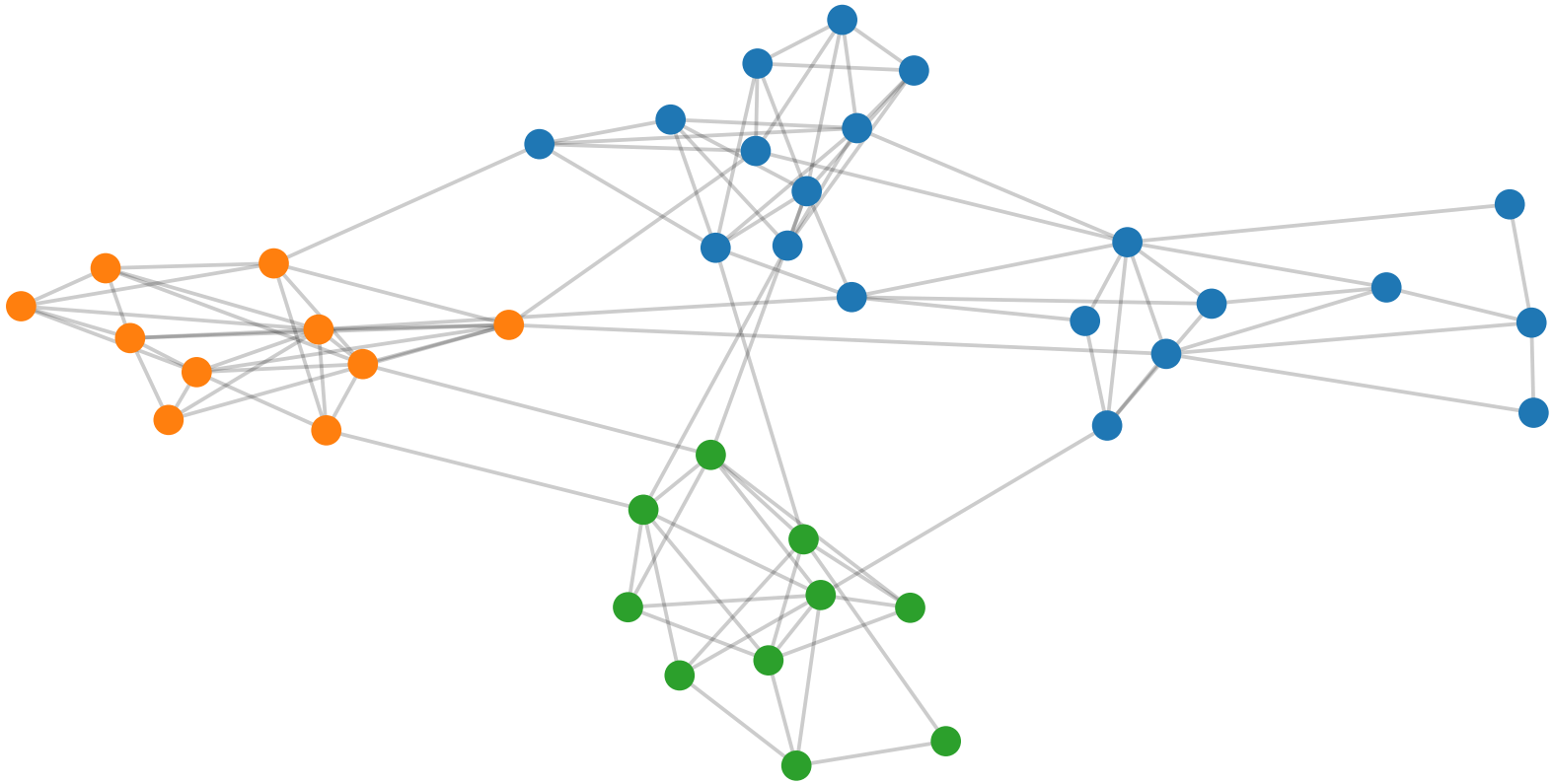
Example: $\gamma = 1$



Example: $\gamma = 3$



Example: $\gamma = 0.3$

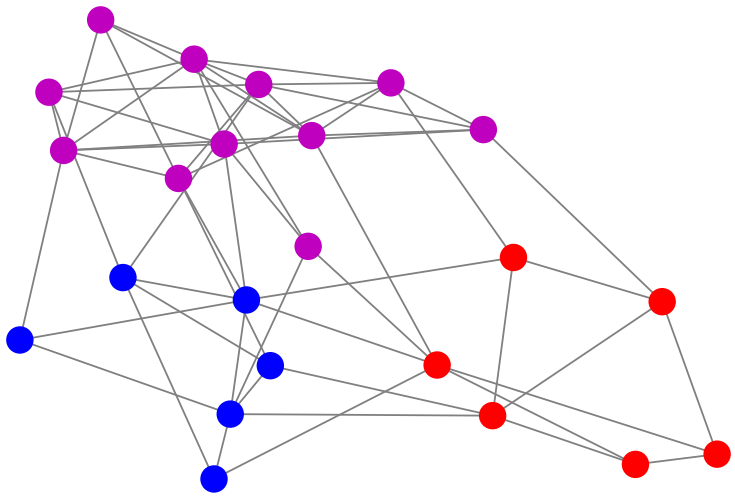


Outline

1. Modularity
2. Resolution
3. **The Louvain algorithm**
4. Cluster ranking
5. Extensions

Key observations

- The modularity depends only on the **number of edges** m_k and the **volume** v_k of each cluster k
- In particular, nodes in the same cluster can be **merged**



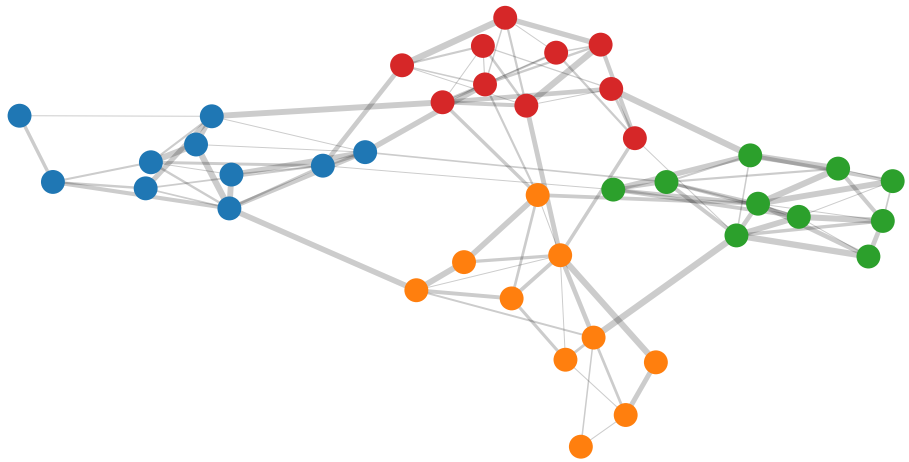
Modularity for weighted graphs

For a weighted graph, the modularity of clustering C is defined by:

$$Q_\gamma(C) = \frac{1}{2w} \sum_{i,j \in V} \left(A_{ij} - \gamma \frac{w_i w_j}{2w} \right) \delta_{C(i),C(j)}$$

where

- $w_i = \sum_j A_{ij}$ is the weight of node i
- $w = \sum_{i<j} A_{ij}$ is the total weight of edges



Adding self-loops

For a weighted graph with self-loops,

$$Q_\gamma(C) = \frac{1}{2w} \sum_{i,j \in V} \left(A_{ij} - \gamma \frac{w_i w_j}{2w} \right) \delta_{C(i),C(j)} + \frac{1}{2w} \sum_{i \in V} A_{ii}$$

where

- $w_i = \sum_{j \neq i} A_{ij} + 2A_{ii}$ is the weight of node i
- $w = \sum_{i<j} A_{ij} + \sum_i A_{ii}$ is the total weight of edges

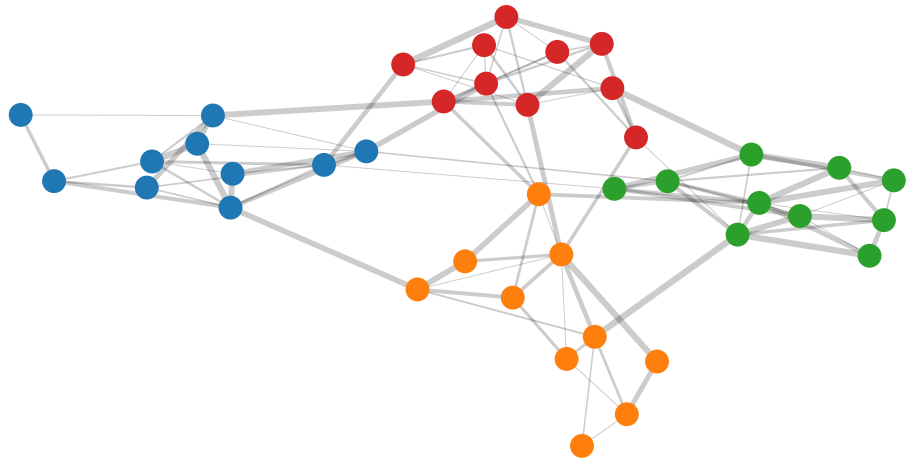
Modularity at cluster level

For a weighted graph,

$$Q_\gamma(C) = \sum_k \frac{w_k}{w} - \gamma \sum_k \left(\frac{v_k}{v}\right)^2,$$

where

- w_k is the **weight** of cluster k (sum of edge weights)
- v_k is the **volume** of cluster k (sum of node weights)
- $v = \sum_i w_i = 2w$ is the volume of the graph

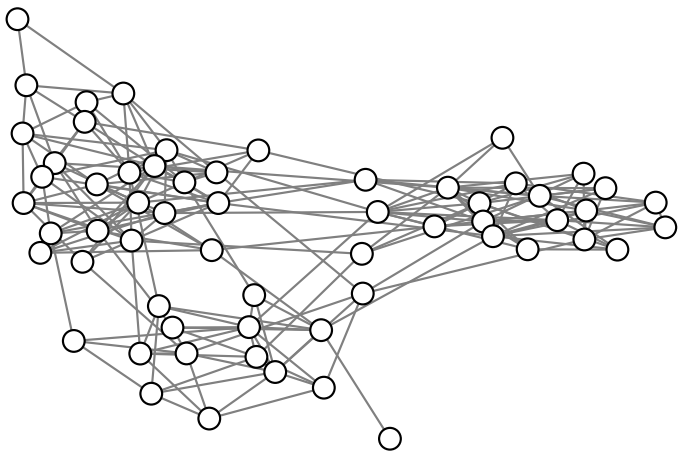


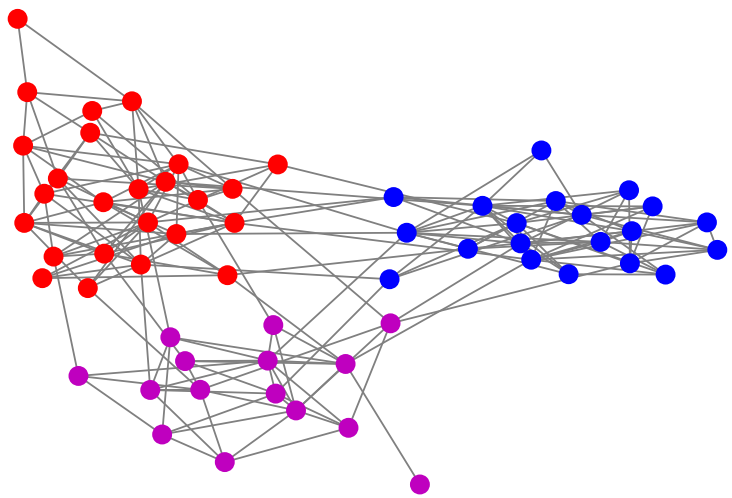
Maximizing the modularity

Consider the following problem:

$$\max_{C} Q_\gamma(C)$$

- Combinatorial problem!
- NP-hard





The Louvain algorithm (2008)

Greedy algorithm:

1. **(Initialization)** $C \leftarrow$ identity
2. **(Maximization)** While modularity $Q_\gamma(C)$ increases, update C by moving one node from one cluster to another
3. **(Aggregation)** Merge all nodes belonging to the same cluster into a single node, update the weights accordingly and apply step 2 to the aggregate graph

Note: The outcome depends on the order in which nodes are considered!

Local optimization

$$Q_\gamma(C) = \sum_k \frac{w_k}{w} - \gamma \sum_k \left(\frac{v_k}{v}\right)^2$$

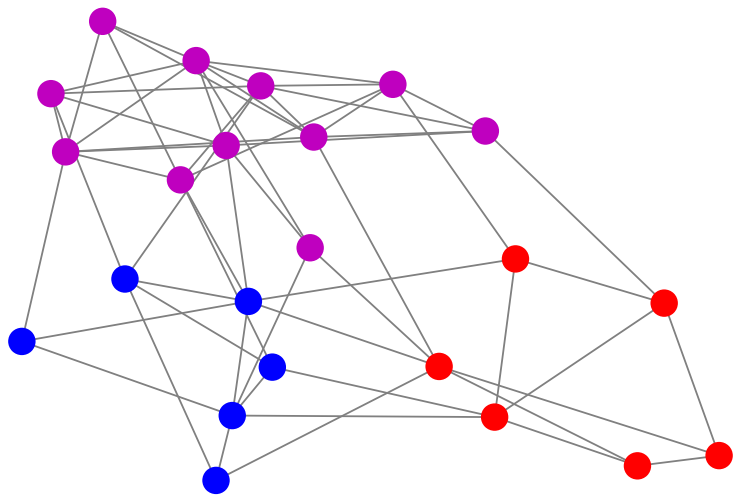
Outline

1. Modularity
2. Resolution
3. The Louvain algorithm
4. **Cluster ranking**
5. Extensions

Cluster strength

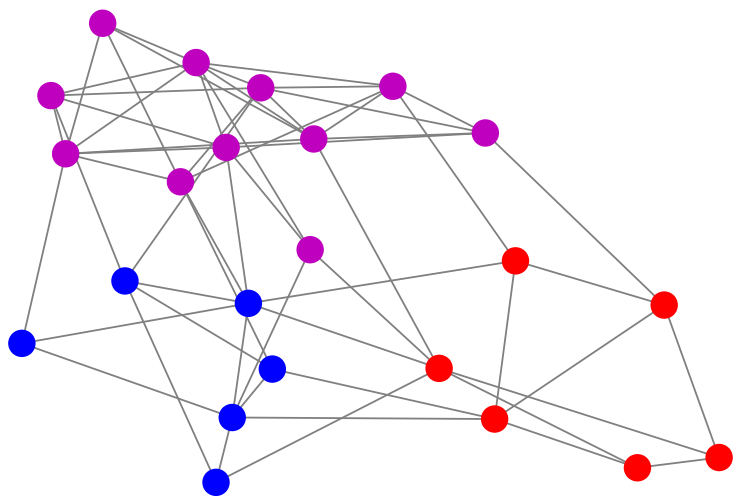
Ratio of weight to volume:

$$\rho_k = \frac{2w_k}{v_k}$$



Random walk

- $P_{ij} = A_{ij}/w_i$, probability of moving from i to j
- A Markov chain $X_0, X_1, X_2, \ldots$ with transition matrix P
- Stationary distribution: $\pi = \frac{1}{v}(w_1, \ldots, w_n)$
- Relative frequency of moves from i to j: $\pi_i P_{ij}$



Interpretation of cluster strength

Proposition

The cluster strength is the probability that the random walk (in steady state) **stays** in that cluster after one move:

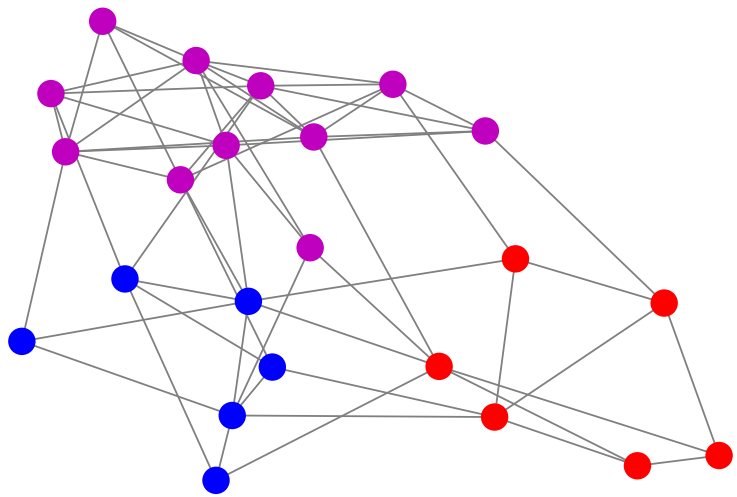
$$\rho_k = P(C(X_{t+1}) = k |\ C(X_t) = k)$$

Cluster strength and modularity

We expect ρ_k to be higher than π_k (the probability to be in cluster k in steady state)

Proposition

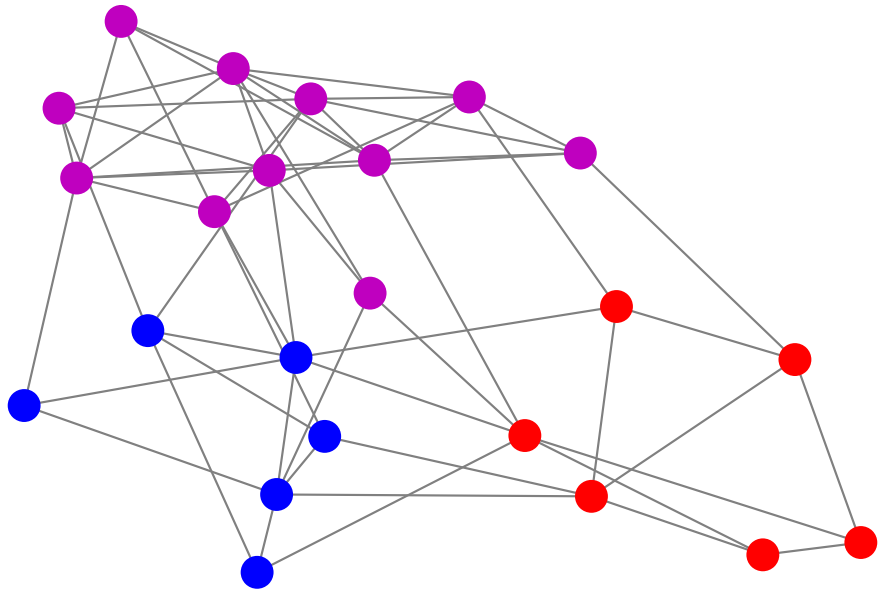
$$Q(C) = \sum_k \pi_k(\rho_k - \pi_k)$$



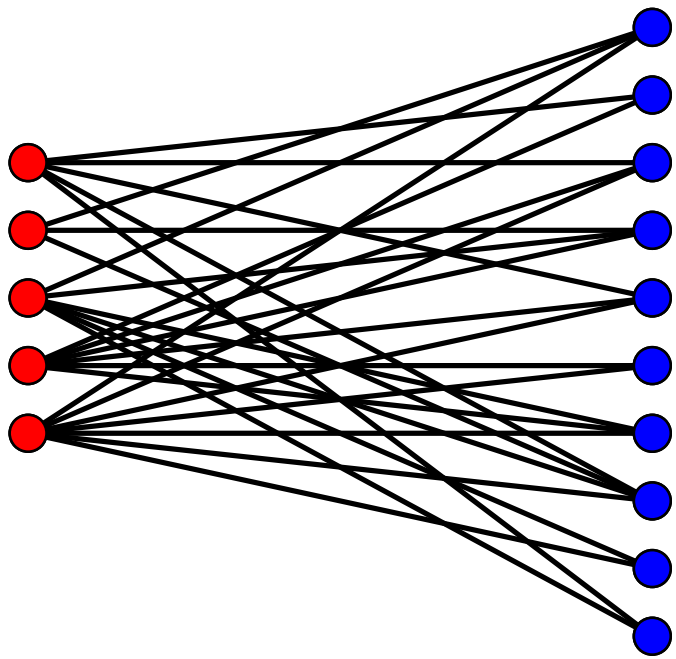
Outline

1. Modularity
2. Resolution
3. The Louvain algorithm
4. Cluster ranking
5. **Extensions**

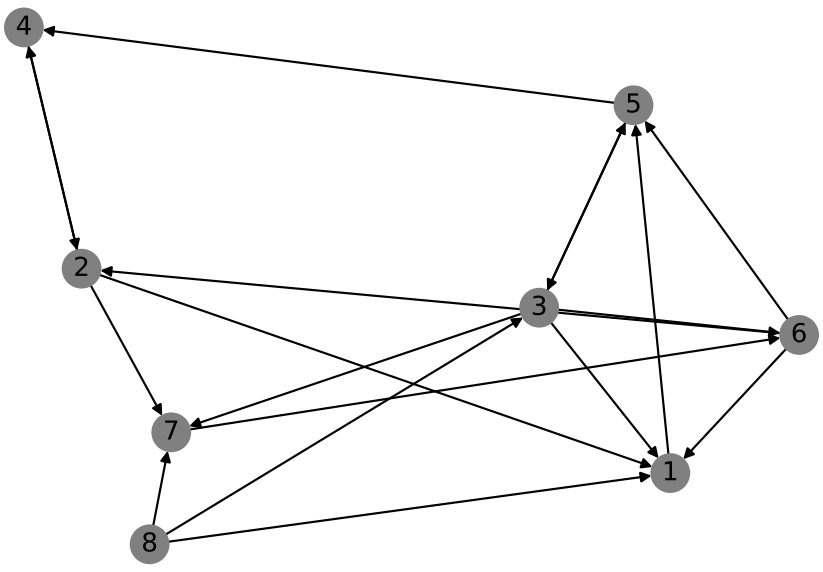
Soft clustering

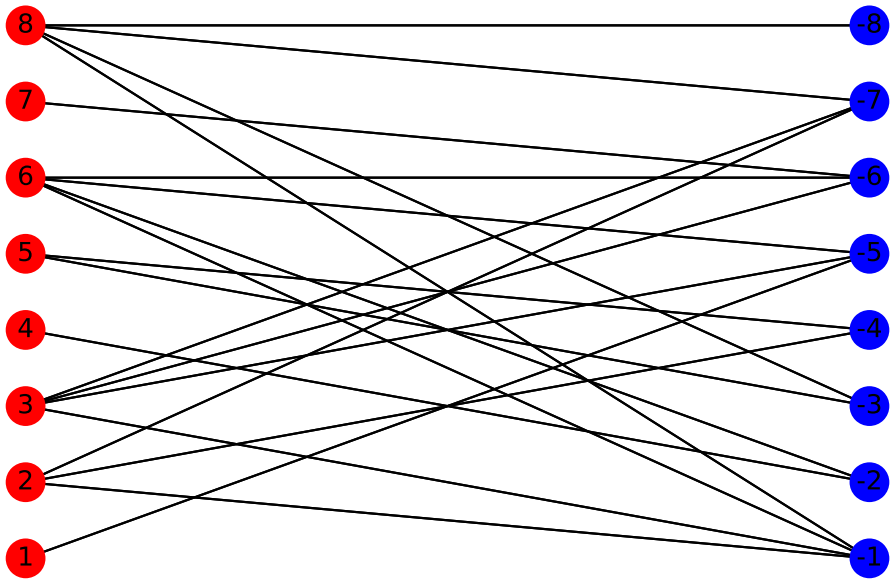


The case of bipartite graphs



Directed graphs





Summary

Clustering is a key technique of graph analysis, revealing the structure of the graph:

- **Modularity**, a fundamental quality metric
- **Resolution**, a parameter to explore the graph structure at different scales
- The **Louvain algorithm**, the most efficient algorithm for graph clustering, able to process massive graphs
- Applicable to **bipartite** and **directed** graphs