Graph Mining
SD212
6. Clustering

Thomas Bonald

2017 – 2018

Motivation

How to identify relevant groups of nodes in a graph?

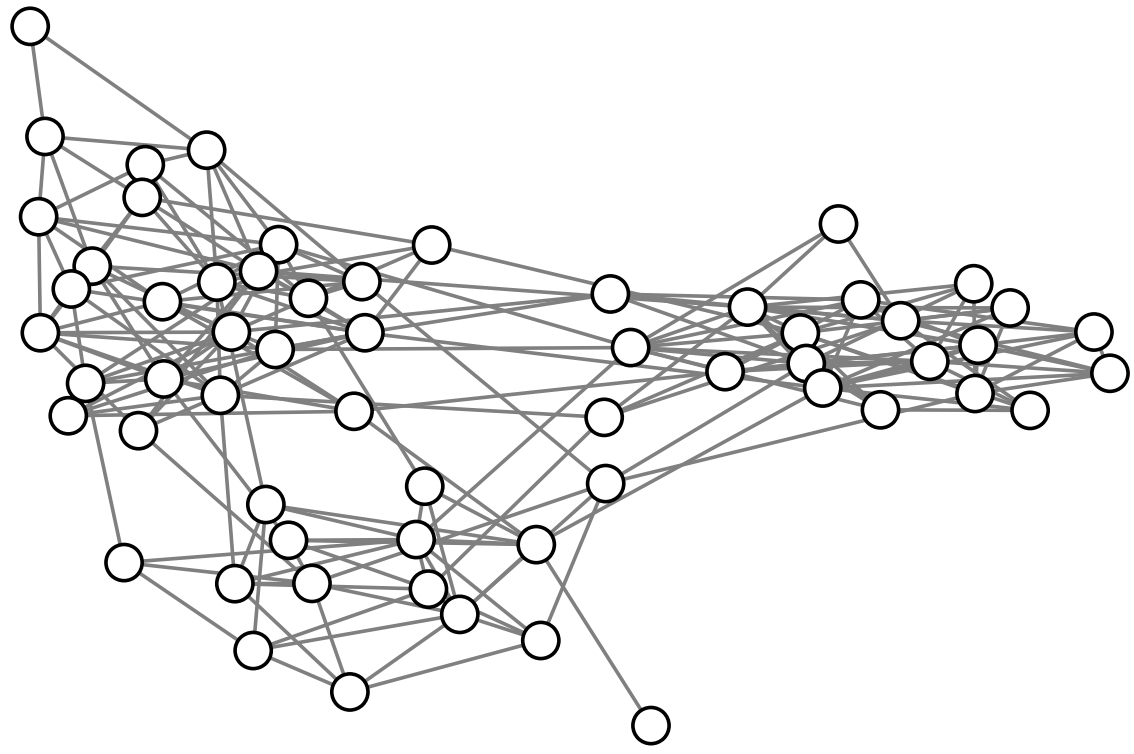
This is the problem of **graph clustering**, also known as **community detection**.

Useful for:

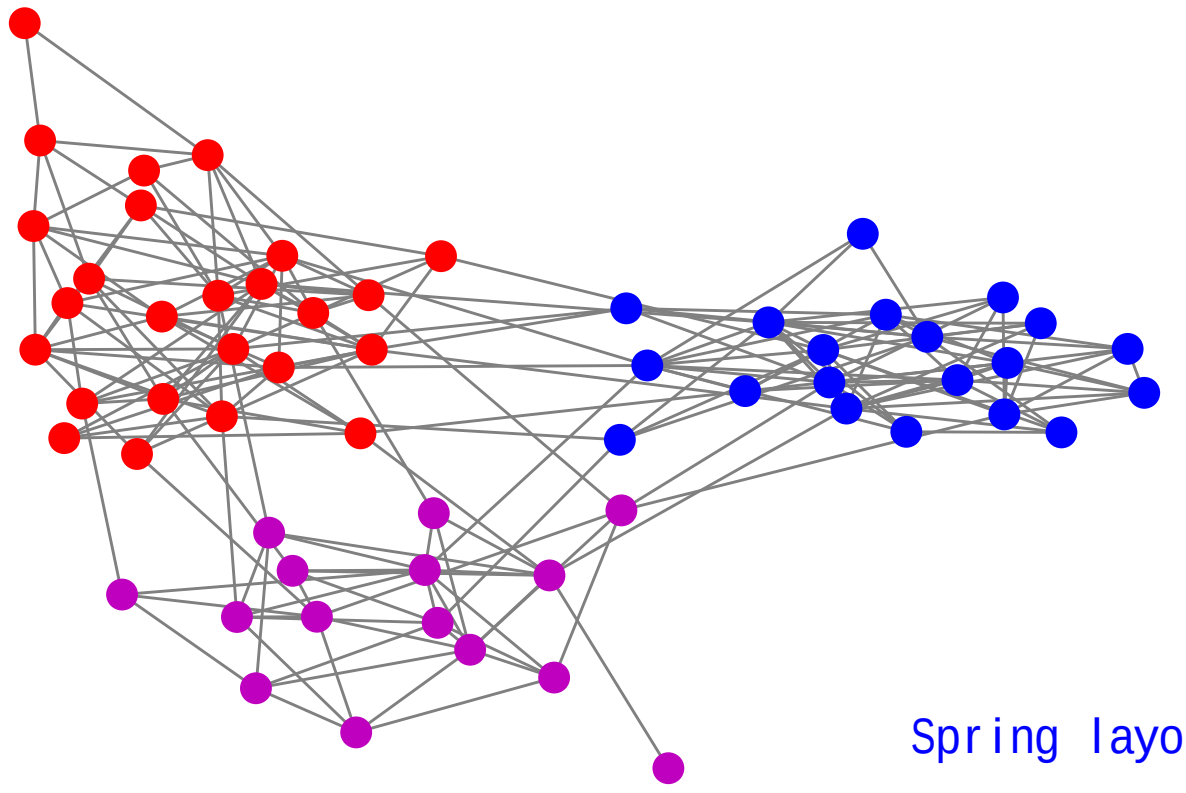
- data visualization
- information retrieval / structure
- content / friend recommendation
- feature extraction
- anomaly detection

We consider **undirected** graphs, possibly weighted.

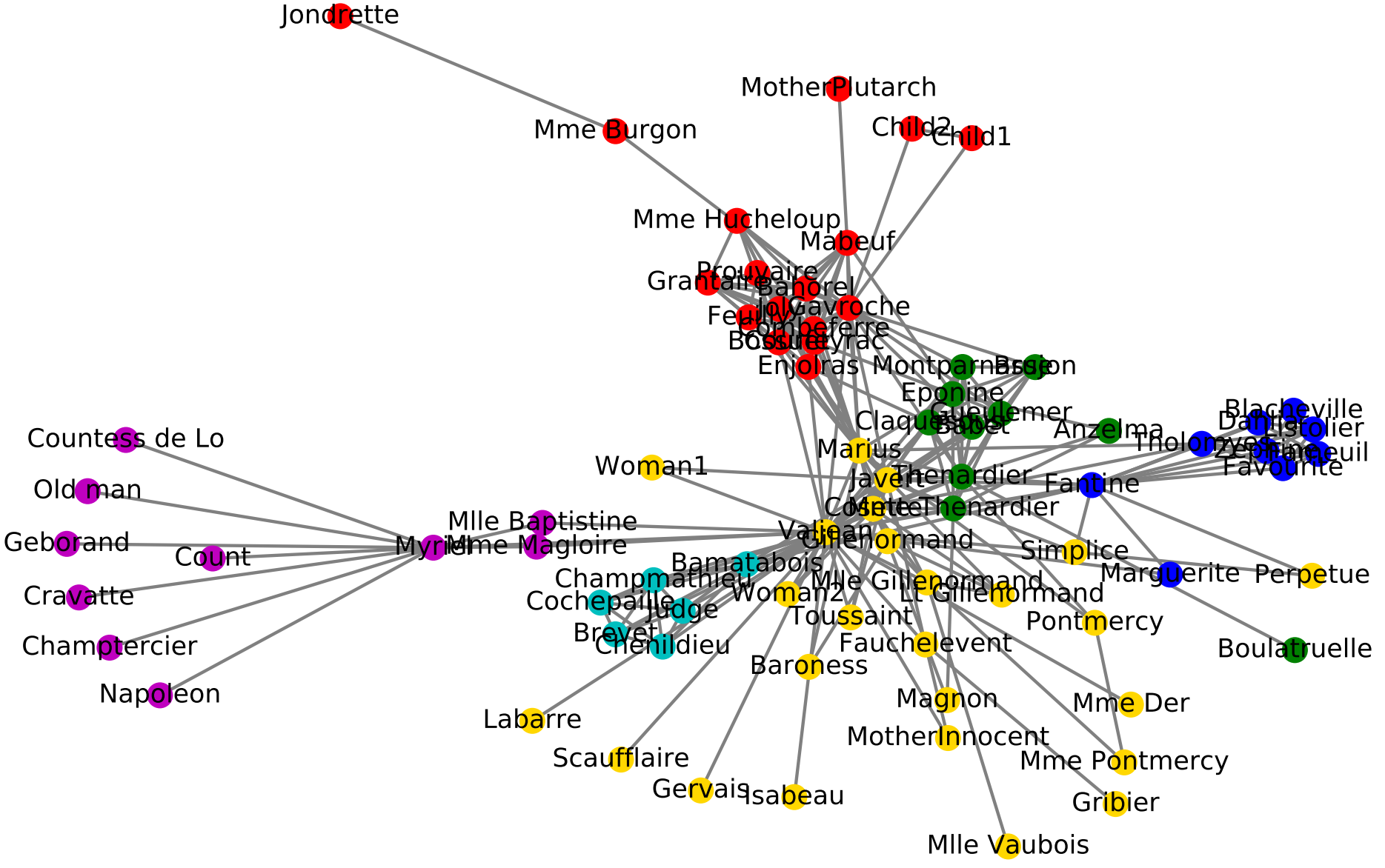
Example



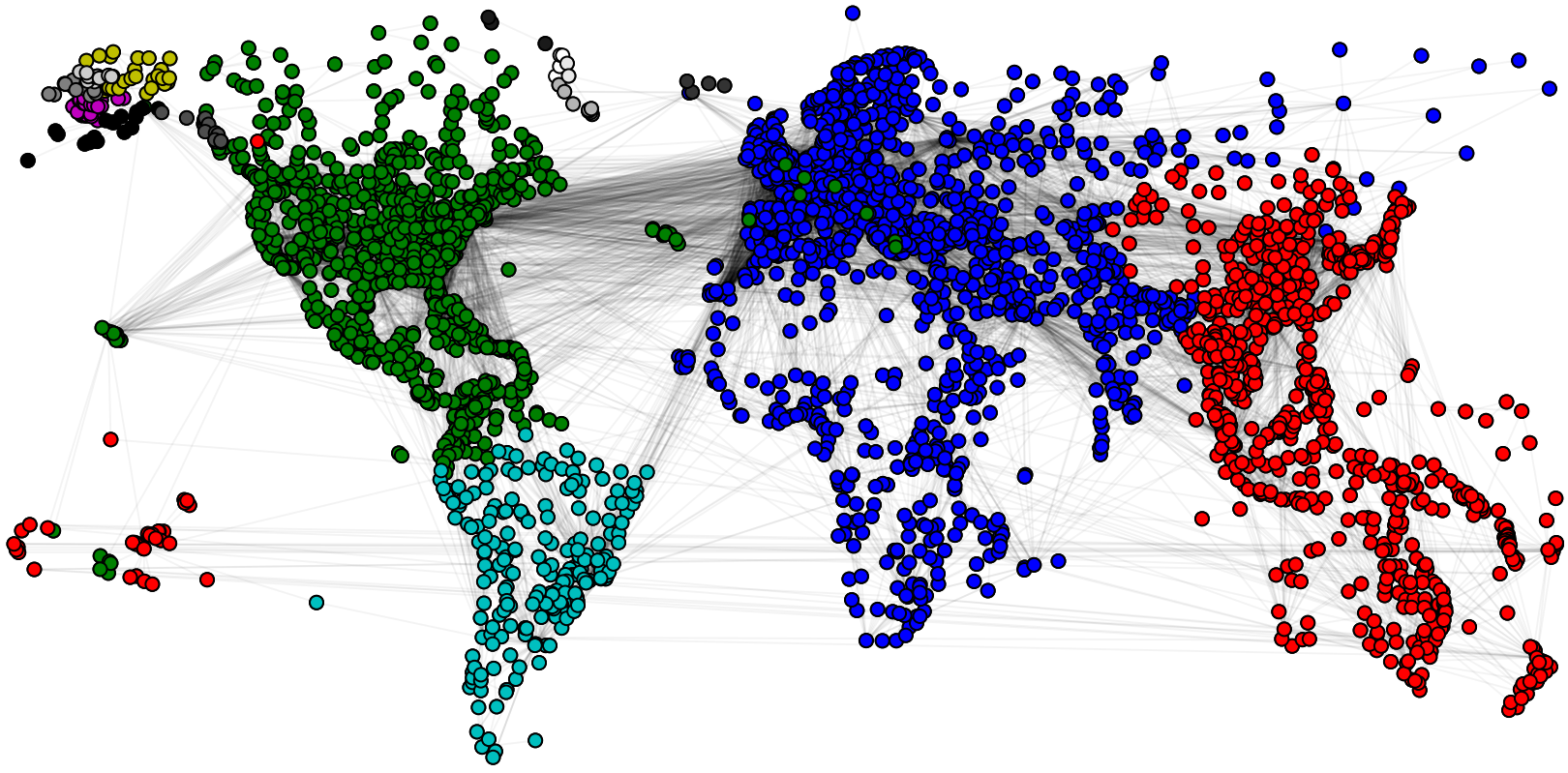
Graph clustering



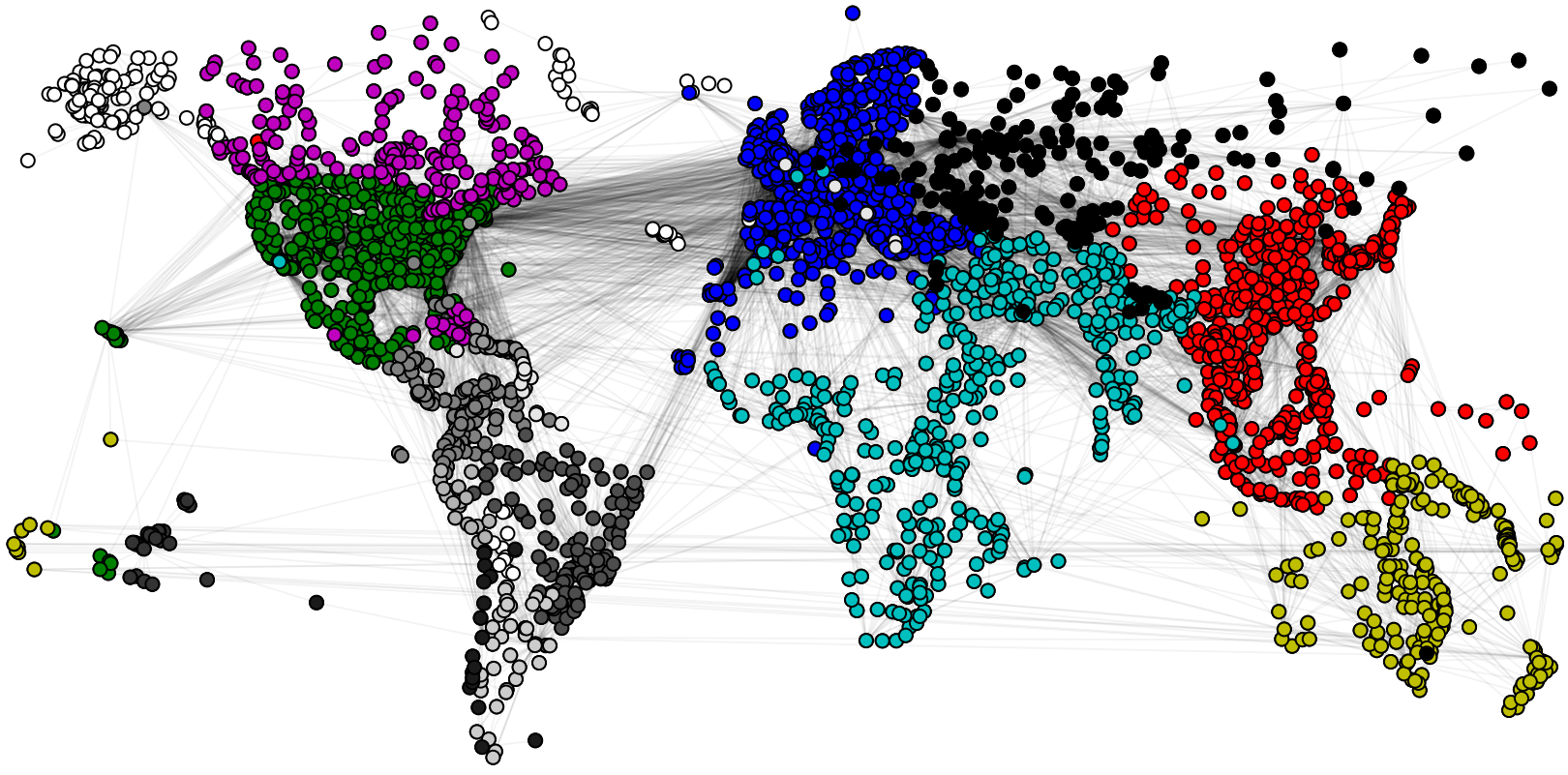
Characters of Les Miserables



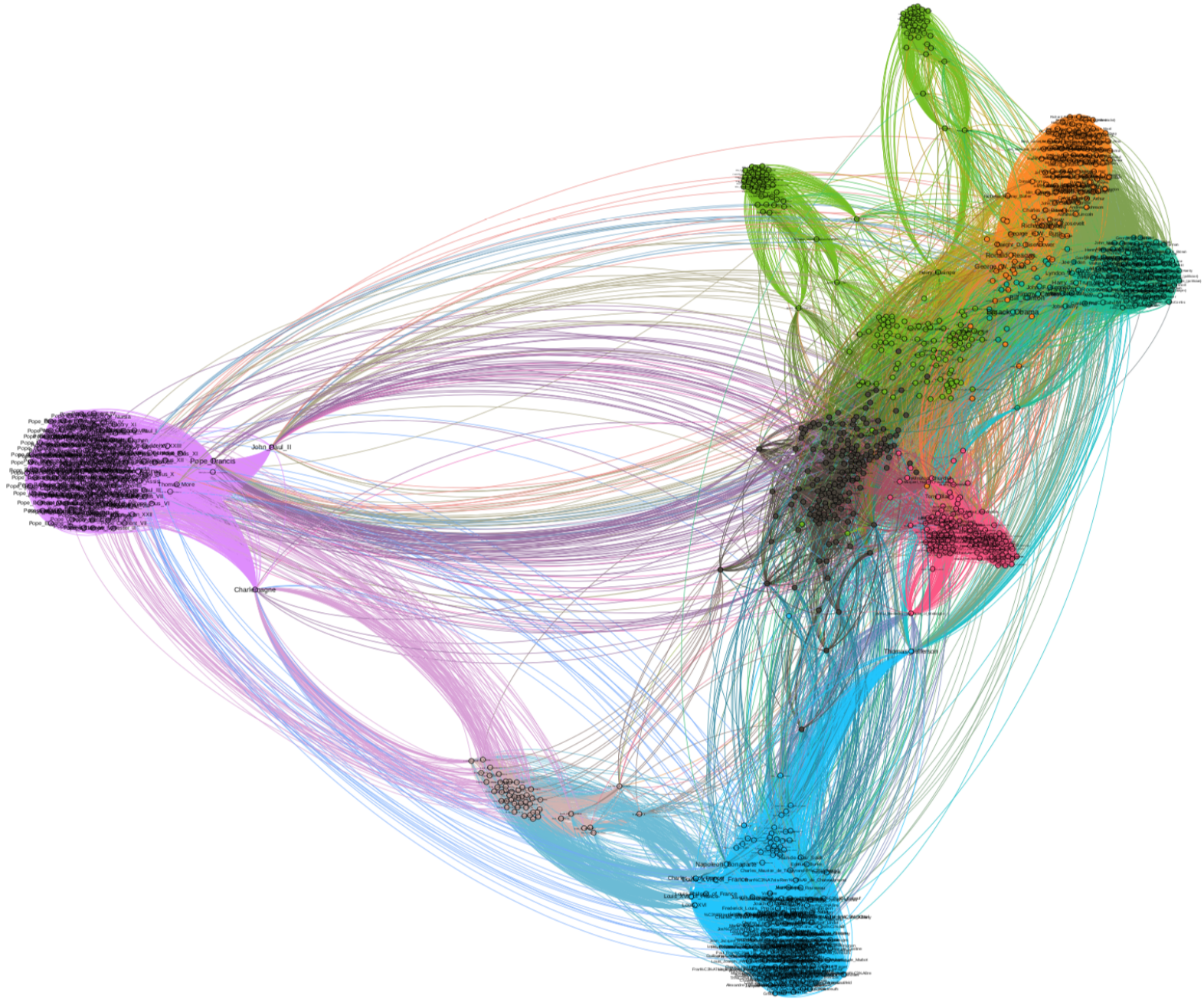
OpenFlights



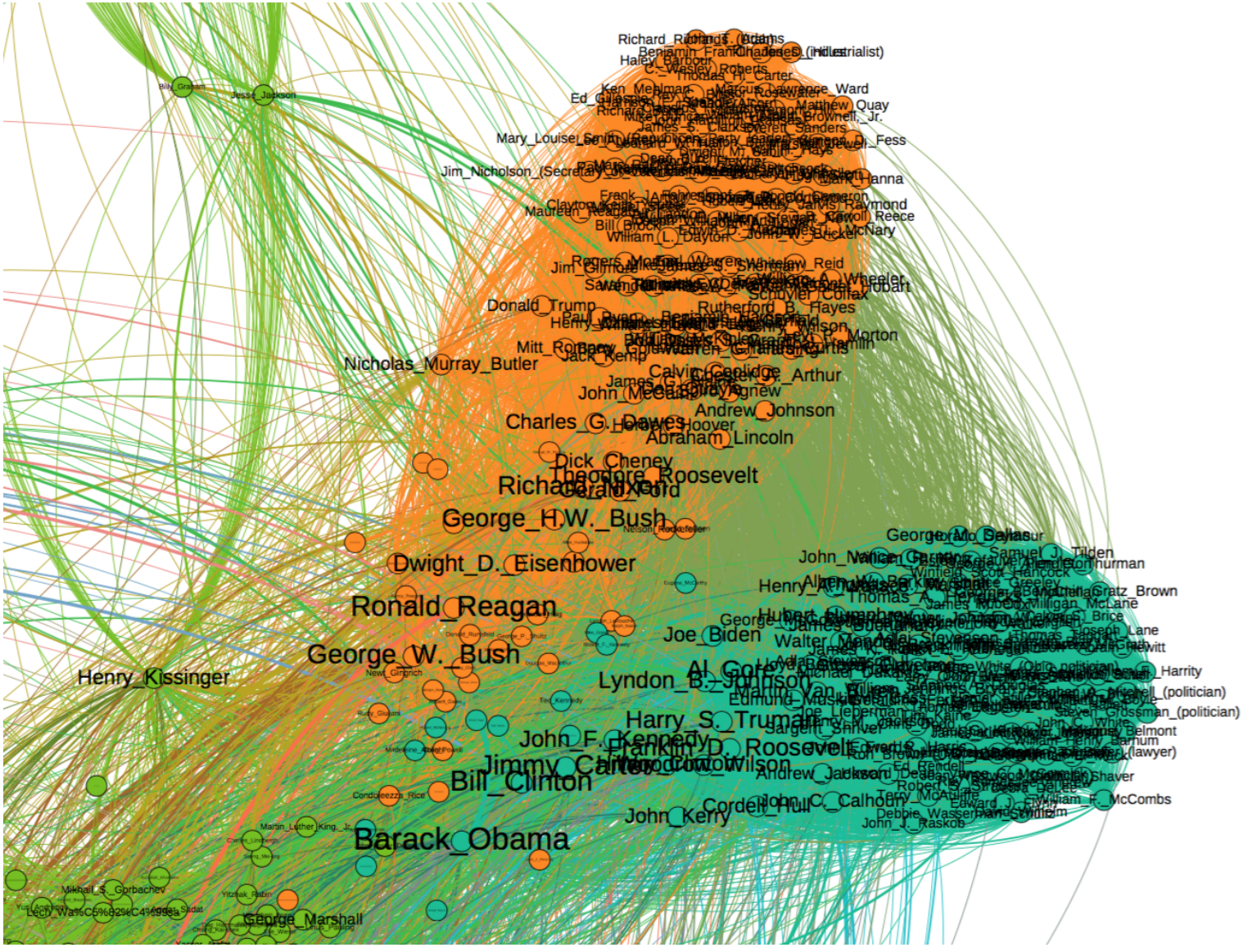
OpenFlights



Wikipedia restricted to Political Figures



Wikipedia restricted to Political Figures (zoom)

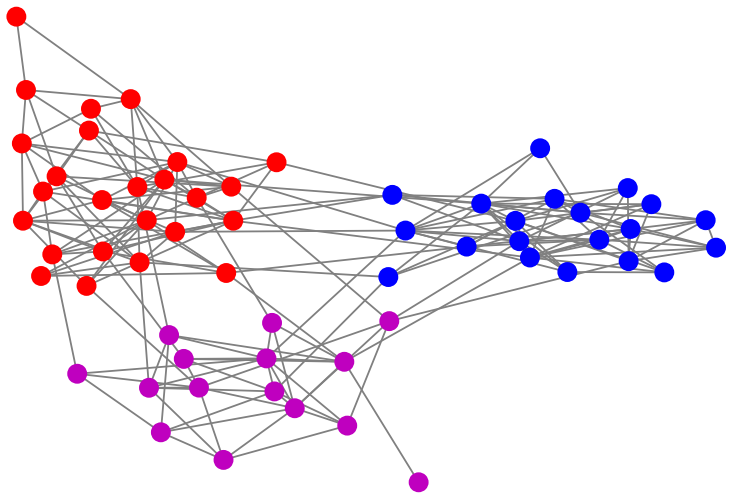


Outline

1. Notion of clustering
2. Modularity
3. Resolution
4. The Louvain algorithm
5. Cluster ranking

Graph clustering

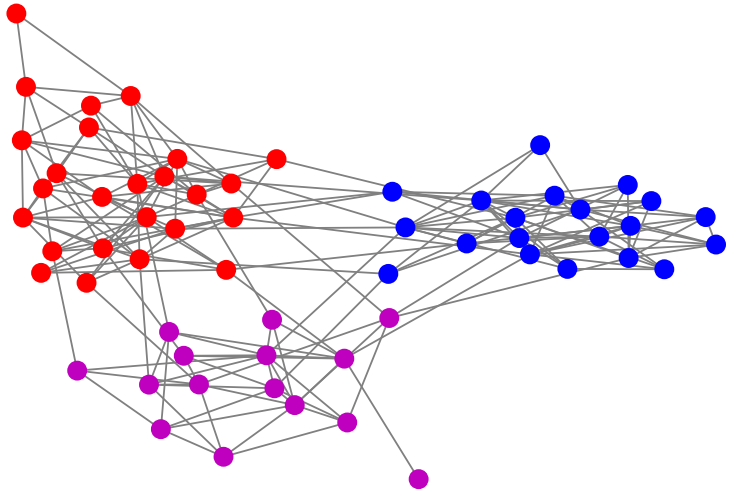
The clustering of a graph $G = (V, E)$ of n nodes and m edges is any surjective function $C : V \to \{1, \ldots, K\}$



Graph clustering

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

- We refer to $C^{-1}(k)$ as cluster k
- Trivial clusterings: $K = 1$ and $K = n$



In general, K is unknown (unlike K-means) and we look for the best clustering over **all** possible values of K.

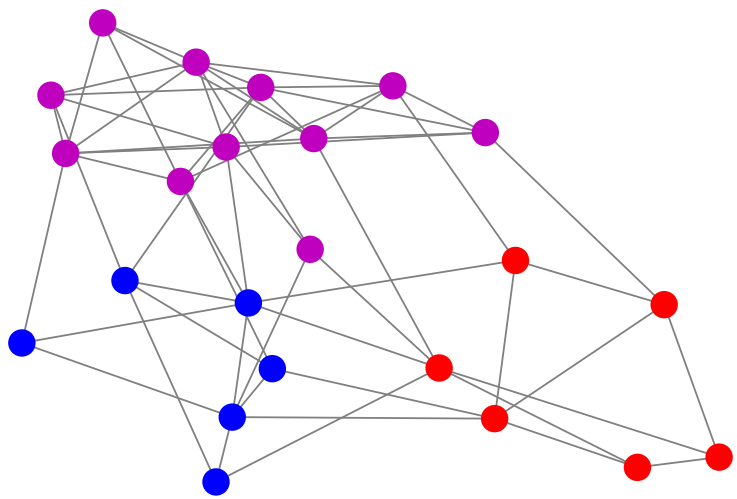
Outline

1. Notion of clustering
2. **Modularity**
3. Resolution
4. The Louvain algorithm
5. Cluster ranking

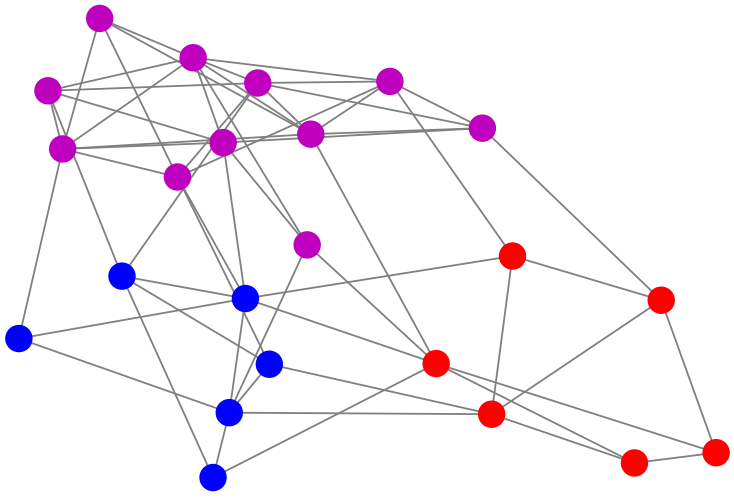
Modularity

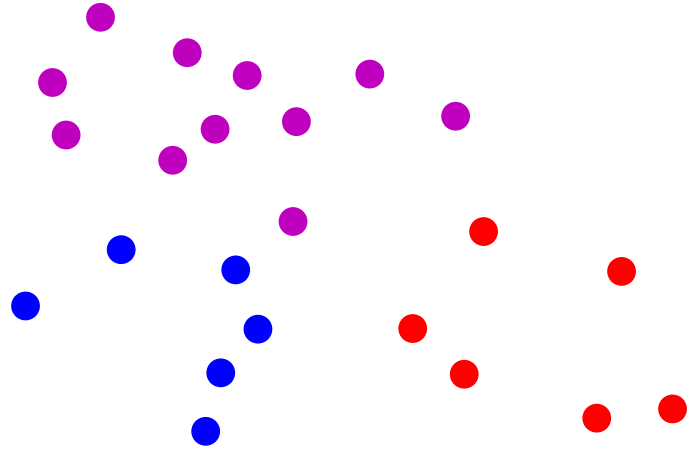
The modularity of clustering C is defined by:

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



The null model





Node sampling

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

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

- Marginal distribution:

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

- Modularity:

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

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:

$$\forall k, \quad p_C(k) = \sum_{l=1}^{K} p_C(k, l) = \sum_{i:C(i)=k} p(i)$$

- Modularity:

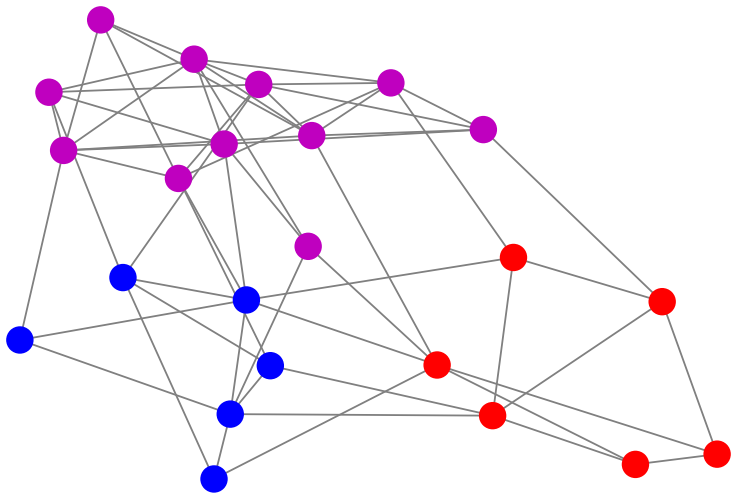
$$\begin{aligned} Q(C) &= \sum_{i,j \in V} (p(i, j) - p(i)p(j))\delta_{C(i),C(j)} \\ &= \sum_{k,l=1}^{K} (p_C(k, l) - p_C(k)p_C(l))\delta_{k,l} \end{aligned}$$

Cluster-level expression of modularity

$$Q(C) = \sum_{k=1}^{K} (p_C(k,k) - p_C(k)^2)$$
$$= \sum_{k=1}^{K} \frac{m_k}{m} - \sum_{k=1}^{K} \left(\frac{w_k}{w}\right)^2$$

where

- m_k is the number of edges in cluster k
- w_k is the total degree in cluster k



The Simpson index (1949)

Resolution

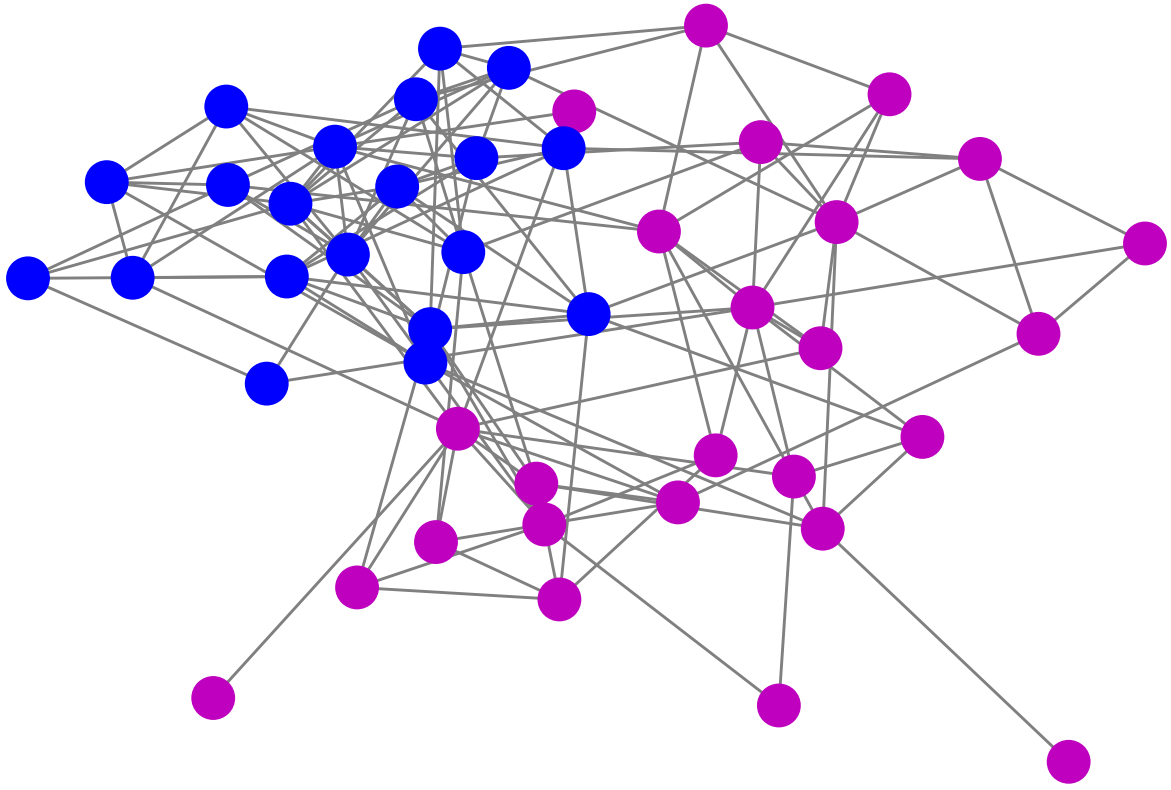
- Parameter $\gamma > 0$ that controls the **quality-diversity** trade-off
- Node level:

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

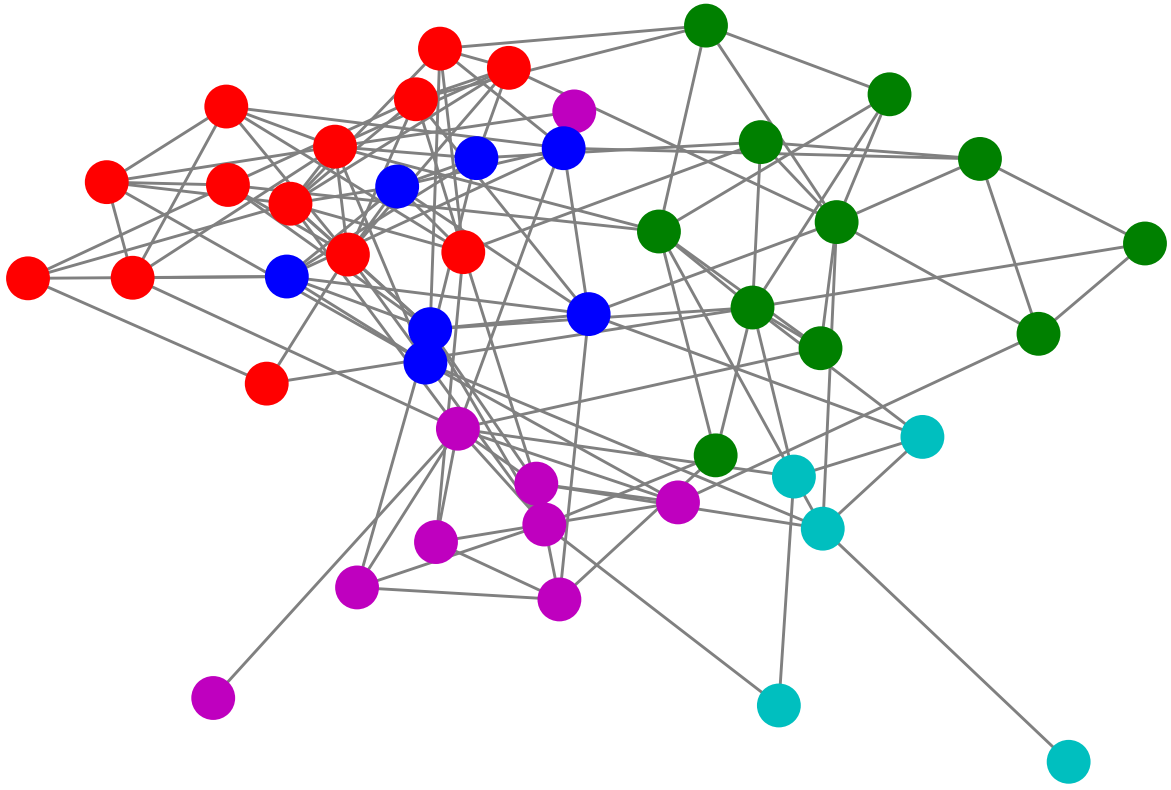
- Cluster level:

$$Q_\gamma(C) = \sum_{k=1}^{K} \frac{m_k}{m} - \gamma \sum_{k=1}^{K} \left(\frac{w_k}{w} \right)^2$$

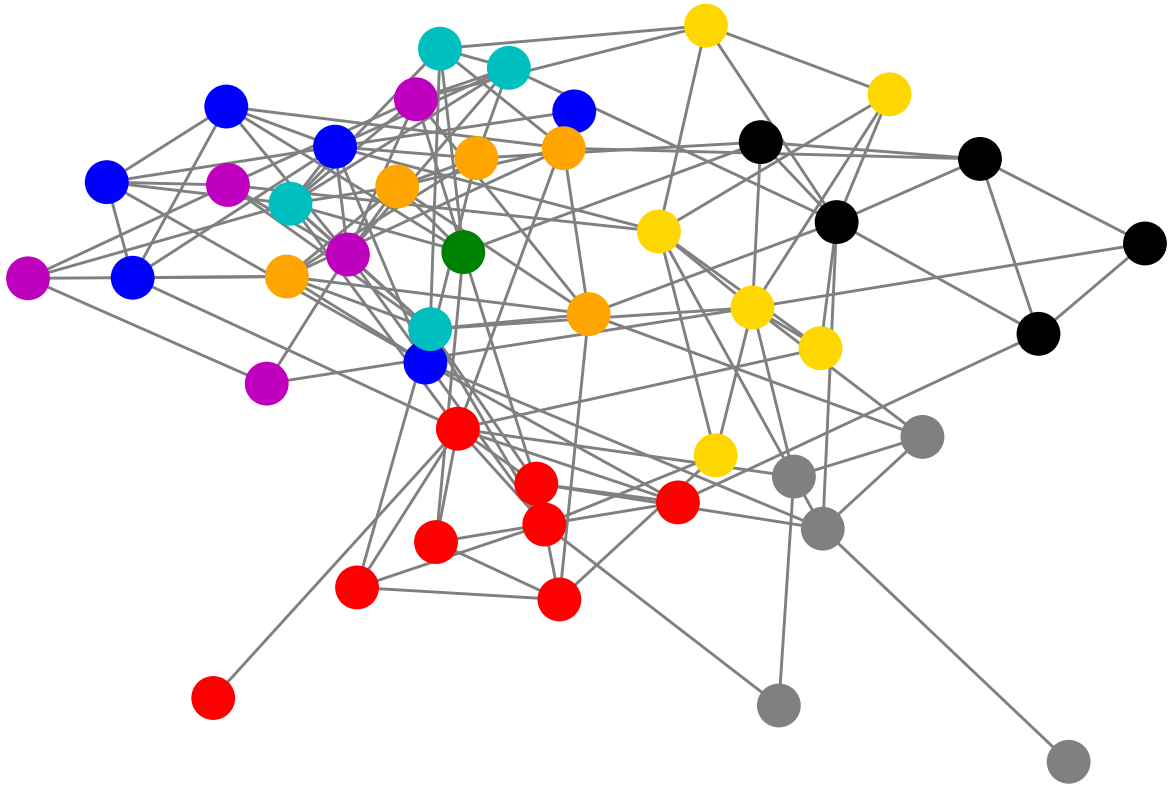
Example: $\gamma = 0.5$



Example: $\gamma = 1$



Example: $\gamma = 2$



Extension to weighted graphs

- Let A be the **weighted** adjacency matrix
- Let $w_i = \sum_j A_{ij}$ be the weight of node i
- Let $w = \sum_i w_i$ be the total weight of nodes
- Modularity:

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

Node sampling

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

$$\forall i, j \in V, \quad p(i,j) = \frac{A_{ij}}{w}$$

- Marginal distribution:

$$\forall i \in V, \quad p(i) = \sum_{j \in V} p(i,j) = \frac{w_i}{w}$$

- Modularity:

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

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:

$$\forall k, \quad p_C(k) = \sum_{l=1}^{K} p_C(k, l) = \sum_{i:C(i)=k} p(i)$$

- Modularity:

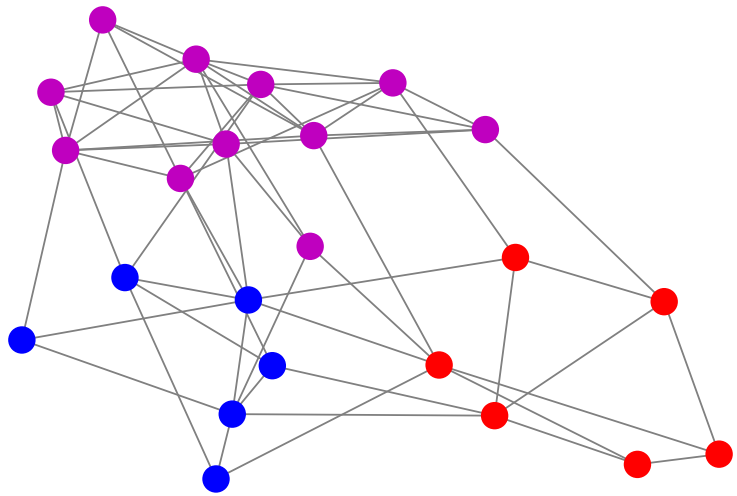
$$Q(C) = \sum_{k,l=1}^{K} (p_C(k, l) - p_C(k)p_C(l))\delta_{k,l}$$

Cluster-level expression of modularity

$$Q_\gamma(C) = \sum_{k=1}^{K} (p_C(k,k) - p_C(k)^2)$$
$$= \sum_{k=1}^{K} \frac{w_k^{(E)}}{w^{(E)}} - \gamma \sum_{k=1}^{K} \left(\frac{w_k^{(V)}}{w^{(V)}} \right)^2$$

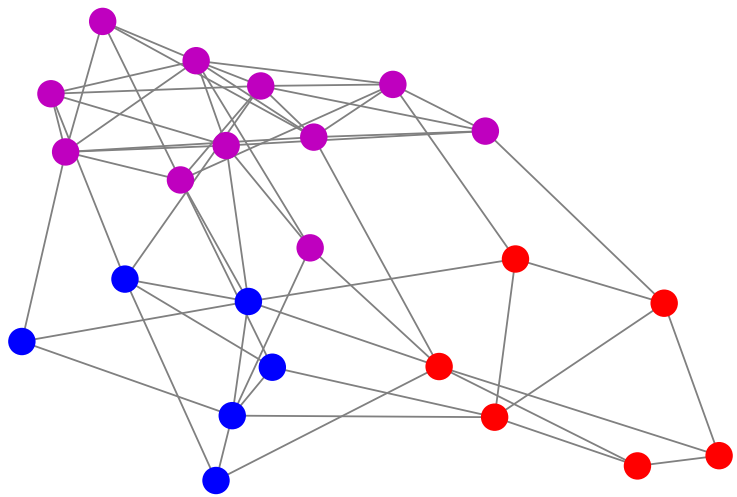
where

- $w_k^{(E)}$ is the total weight of edges in cluster k
- $w_k^{(V)}$ is the total weight of nodes in cluster k

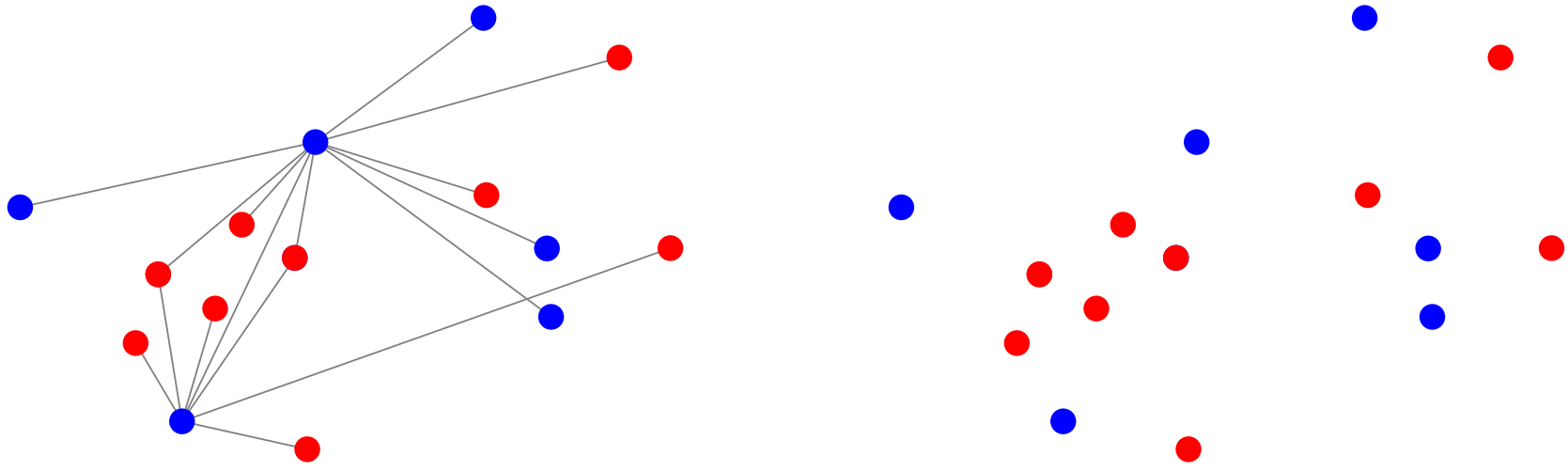


Key observation

The modularity depends on the clustering through $w_k^{(E)}$ and $w_k^{(V)}$ only, for each cluster k



Aggregation



Self-loops

Modularity with self-loops

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

Outline

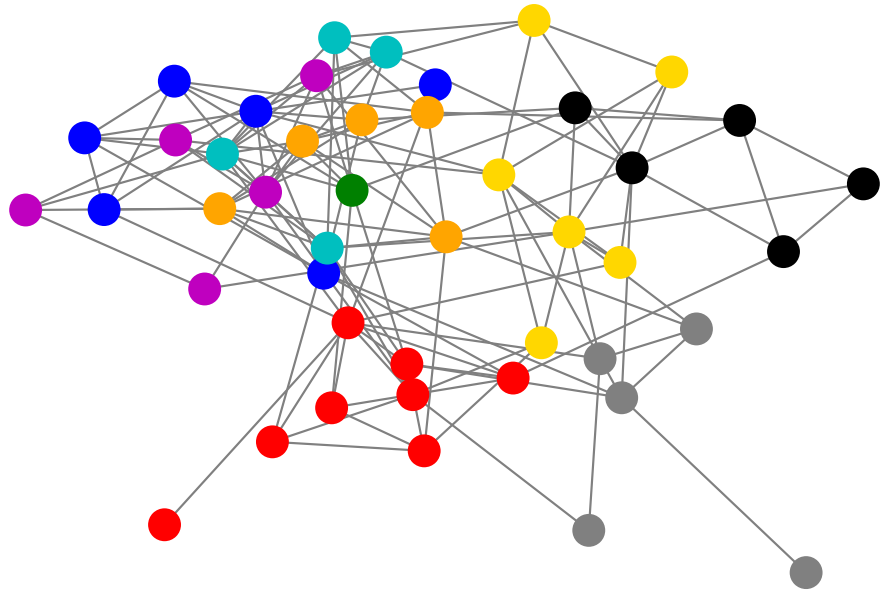
1. Notion of clustering
2. Modularity
3. Resolution
4. **The Louvain algorithm**
5. Cluster ranking

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!

Changing the cluster of a node

$$Q_\gamma(C) = \sum_{k=1}^{K} \frac{w_k^{(E)}}{w^{(E)}} - \gamma \sum_{k=1}^{K} \left(\frac{w_k^{(V)}}{w^{(V)}} \right)^2$$

Variable updates

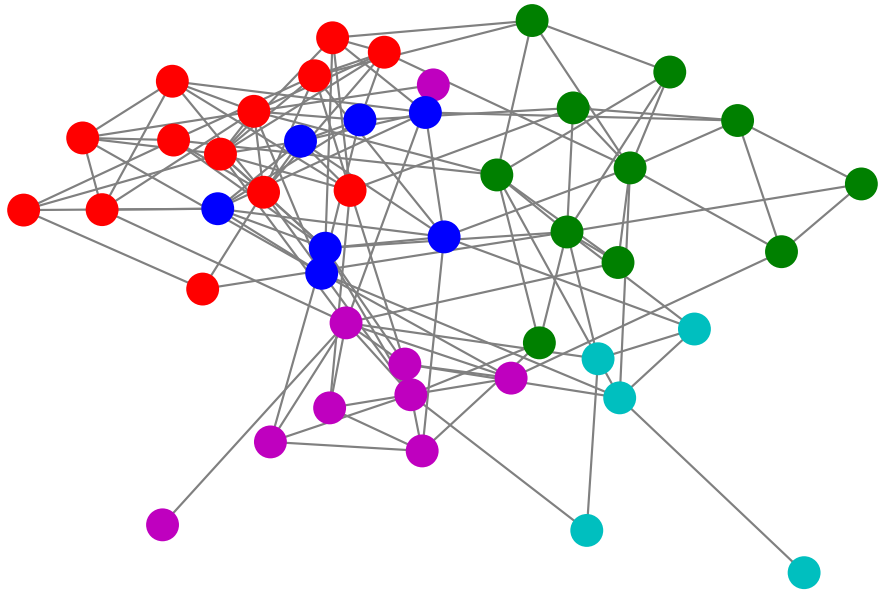
Outline

1. Notion of clustering
2. Modularity
3. Resolution
4. The Louvain algorithm
5. **Cluster ranking**

Cluster strength

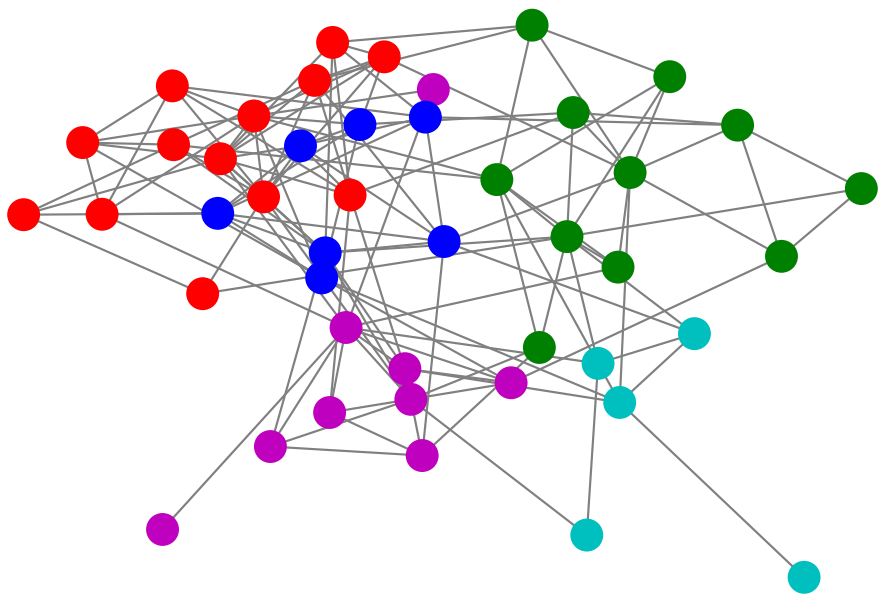
Ratio of inside weight to total weight:

$$\sigma_k = \frac{2w_k^{(E)}}{w_k^{(V)}}$$



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: $\pi \propto w$
- Relative frequency of moves from i to j: $\pi_i P_{ij}$



Cluster sampling by the random walk

- Relative frequency of moves from cluster k to cluster l,

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

- Probability to be in cluster k:

$$p_C(k) = \sum_{l=1}^{K} p_C(k,l) = \sum_{i:C(i)=k} p(i)$$

- We have

$$p_C(k,k) = \frac{w_k^{(E)}}{w^{(E)}}, \quad p_C(k) = \frac{w_k^{(V)}}{w^{(V)}}$$

Interpretation of cluster strength

Proposition

$$\sigma_k = \frac{p_C(k,k)}{p_C(k)} = p_C(k|k)$$

Cluster strength and modularity

We expect σ_k to be higher than $\pi_k \equiv p_C(k)$

Proposition

$$Q(C) = \sum_{k=1}^{K} \pi_k(\sigma_k - \pi_k)$$

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 (greedy) modularity maximization, able to process huge graphs

We have recently proposed an approach based on **hierarchical clustering** to finding the best resolution parameters