Graph Learning
SD212
3. Graph Clustering

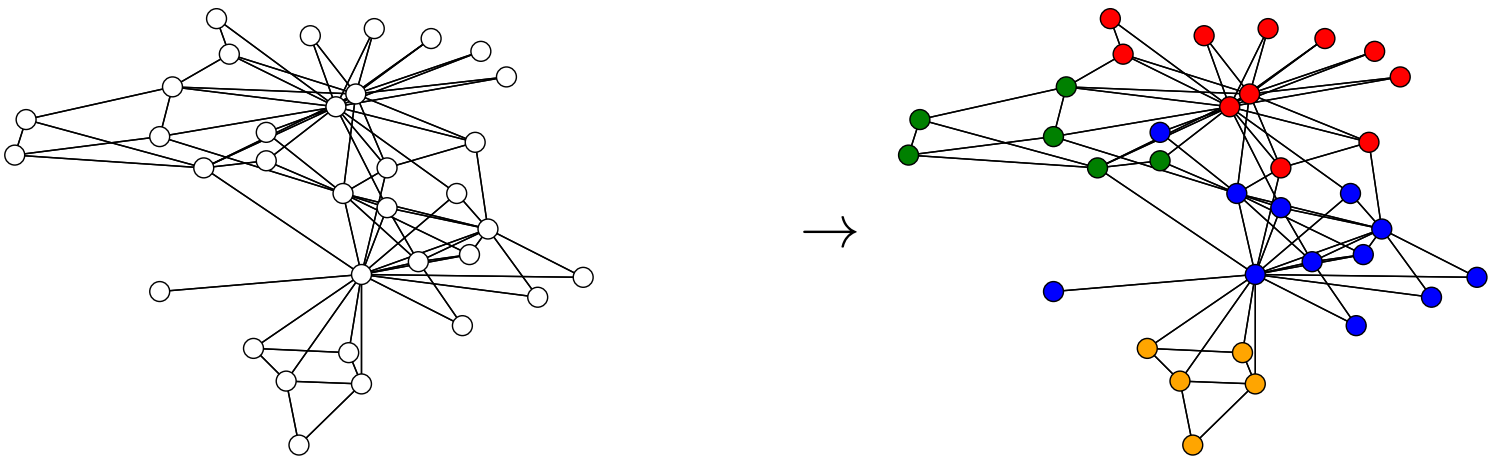
Thomas Bonald

2023 – 2024

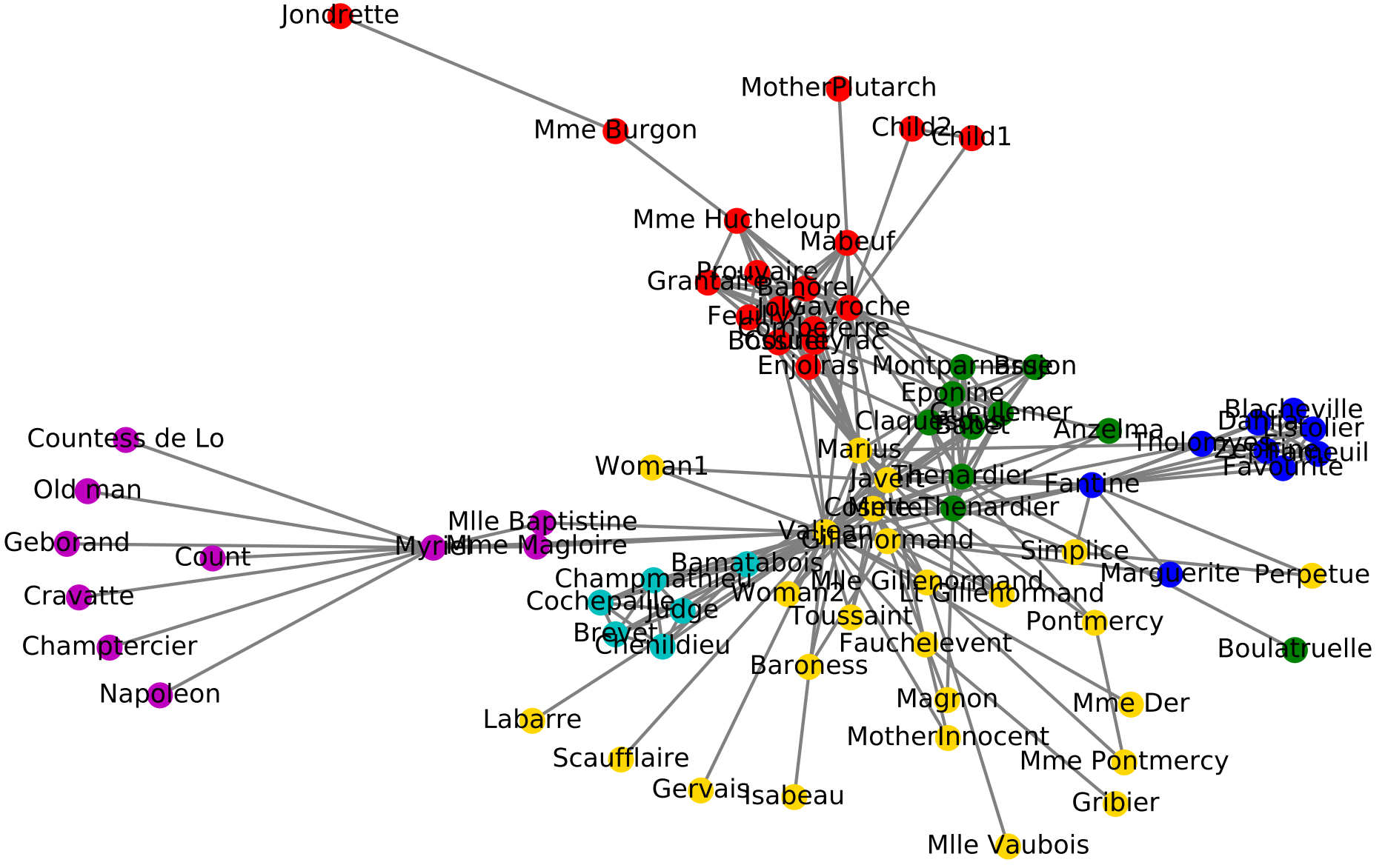
Motivation

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

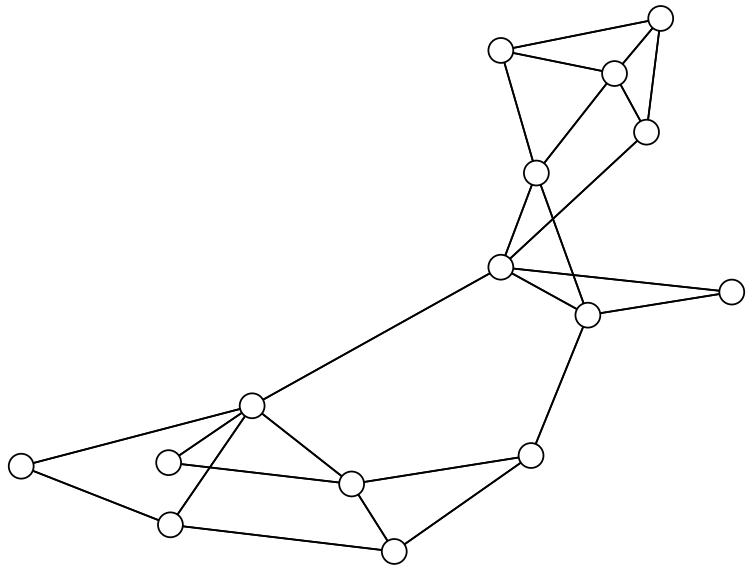


Characters of Les Miserables



Graph clustering

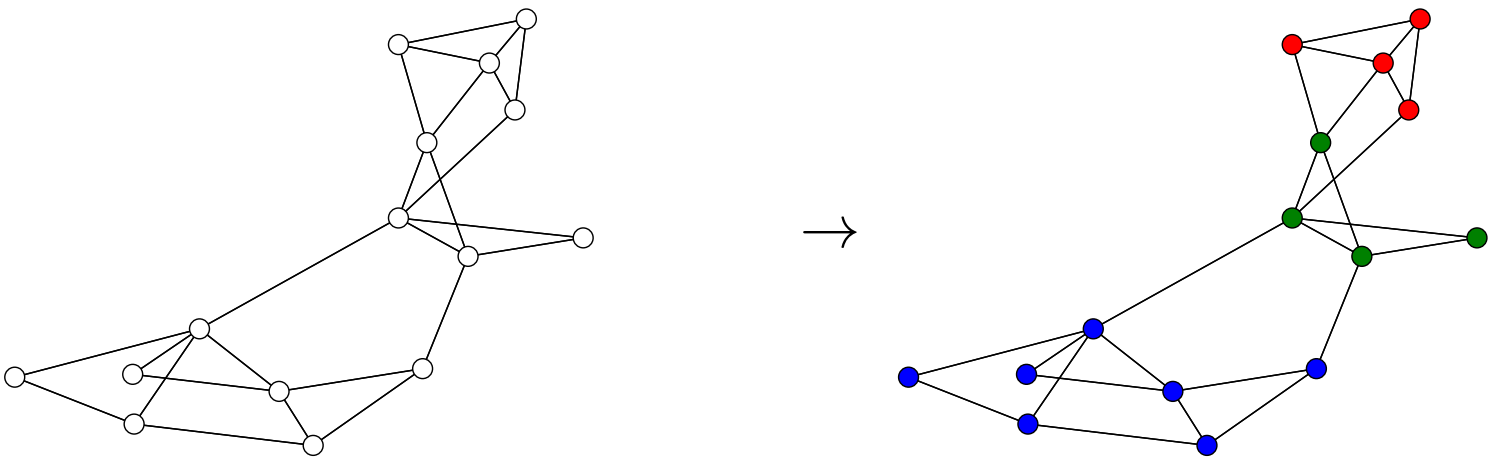
The clustering of a graph $G = (V, E)$ is any function $C : V \to \{1, \ldots, K\}$



In general, K is unknown (unlike K-means) and we look for the best clustering **irrespective** of the value of K

Outline

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

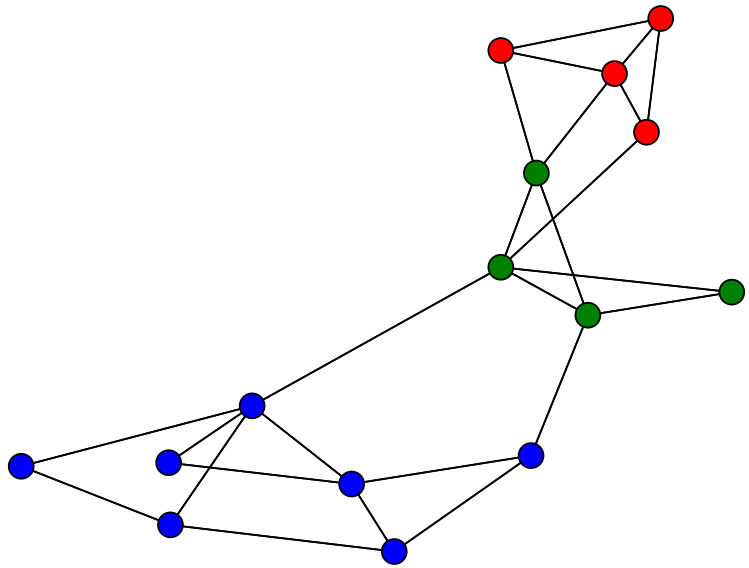


Fitness of a clustering

Let $G = (V, E)$ be an undirected graph with adjacency matrix A
The **fitness** of clustering C is the fraction of edges within clusters:

$$F(C) = \frac{1}{v} \sum_{i,j \in V} A_{ij} \delta_{C(i),C(j)}$$

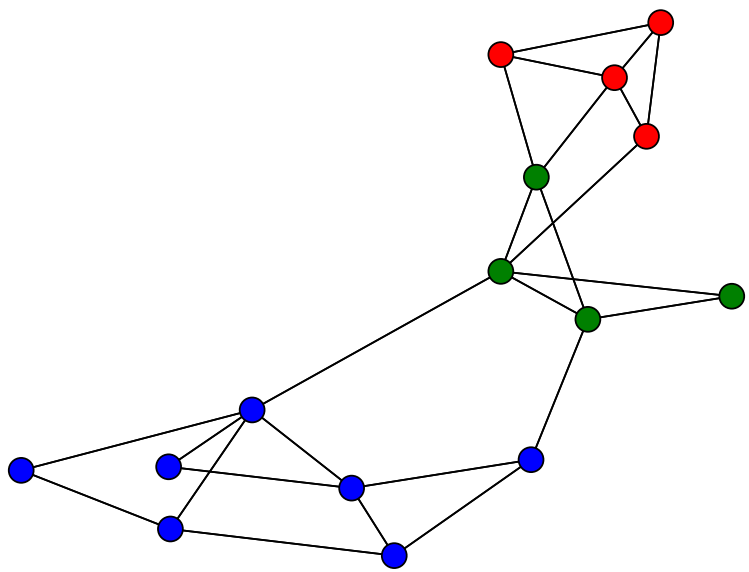
where $v = 1^T A 1$ is the volume of the graph



Modularity

The **modularity** of clustering C is defined by:

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

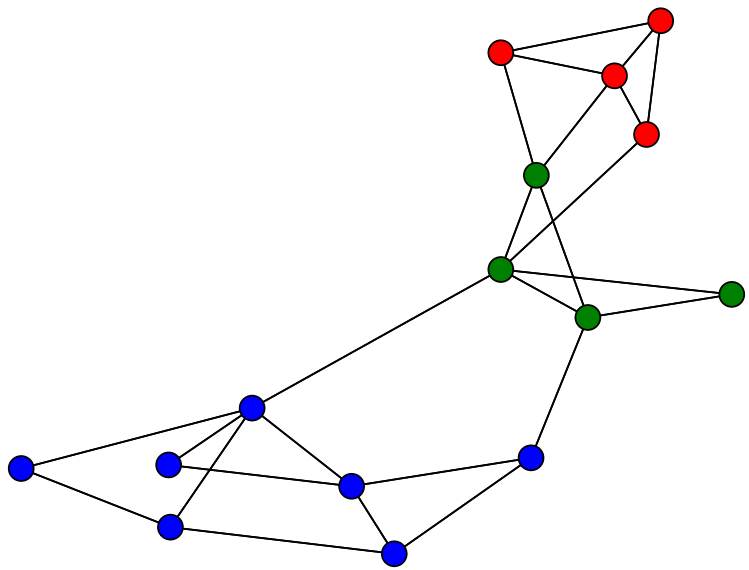


Cluster-level expression

In the absence of self-loops, the modularity can be written:

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

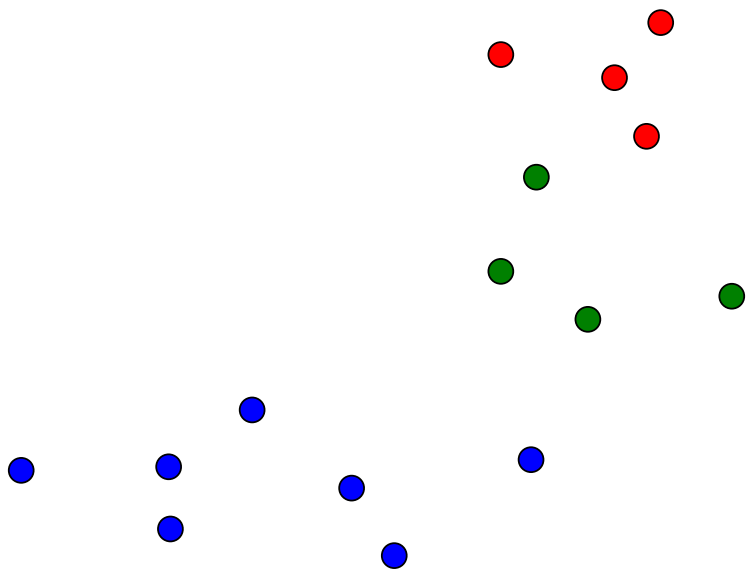
with m_k the **size** (number of edges) and v_k the **volume** (total degree) of cluster k



The Simpson index (1949)

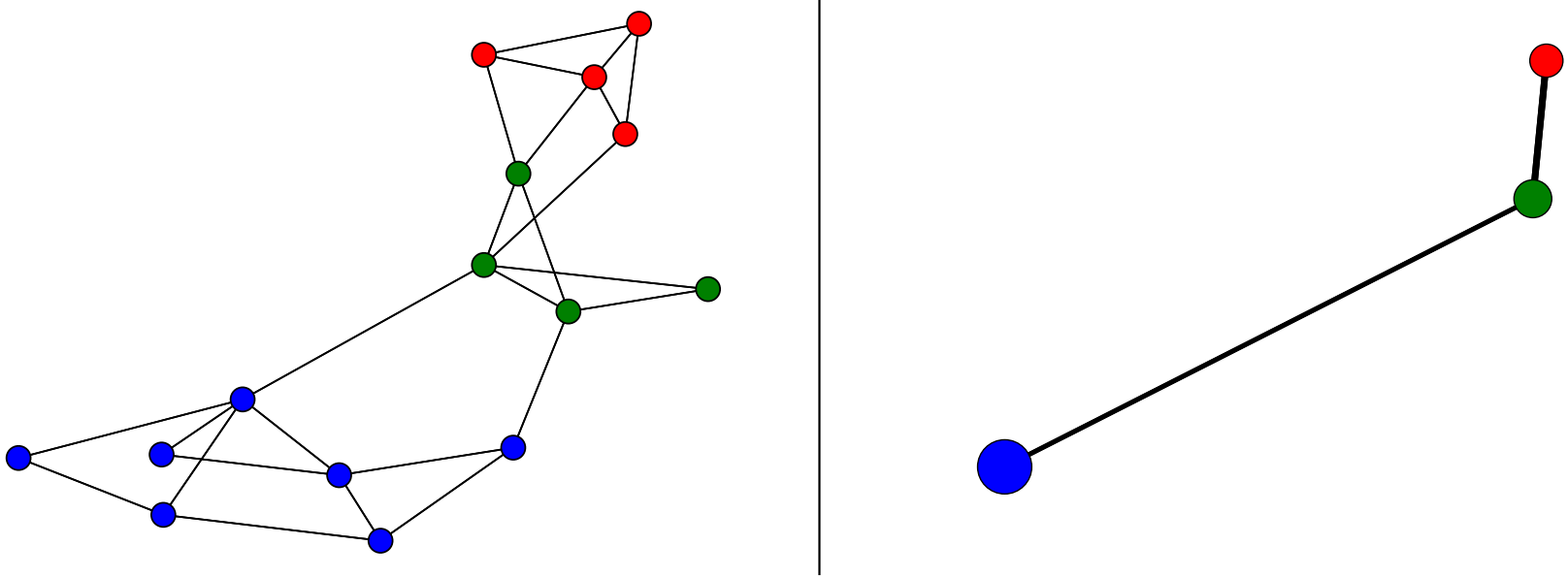
Let $p_1, \ldots, p_K$ be any probability distribution over $\{1, \ldots, K\}$
Simpson's index is a measure of **concentration** of this distribution:

$$S = \sum_{k=1}^{K} p_k^2$$



Aggregation

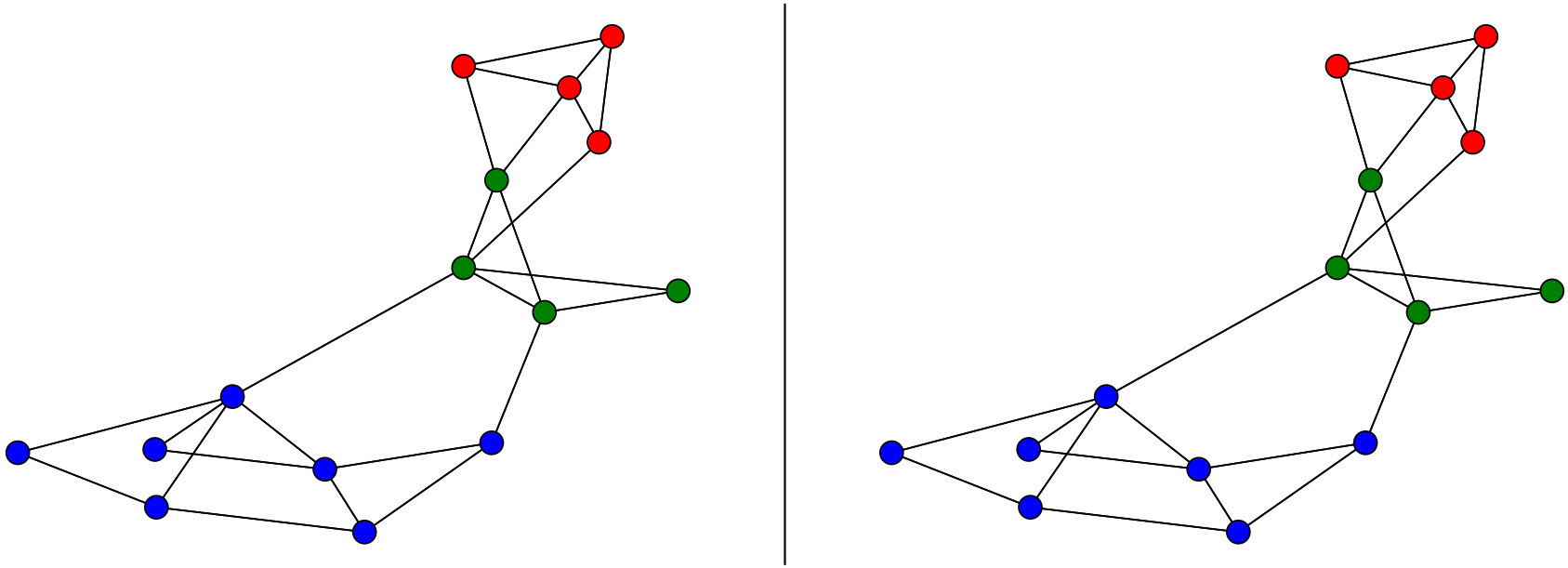
The modularity is preserved by **aggregation**
Edges within clusters $\rightarrow$ **self-loops** in the aggregate graph



Random walk

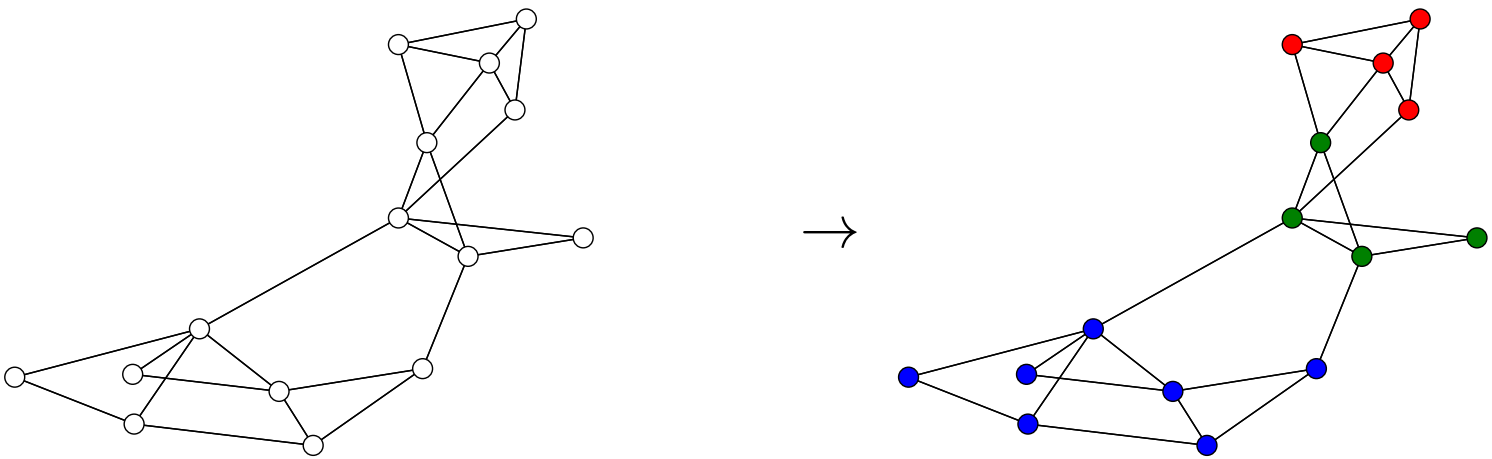
Let X_t, Y_t be two independent random walks in the graph
The modularity can be written:

$$\boxed{Q(C) = \mathrm{P}(C(X_{t+1}) = C(X_t)) - \mathrm{P}(C(X_t) = C(Y_t))}$$



Outline

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

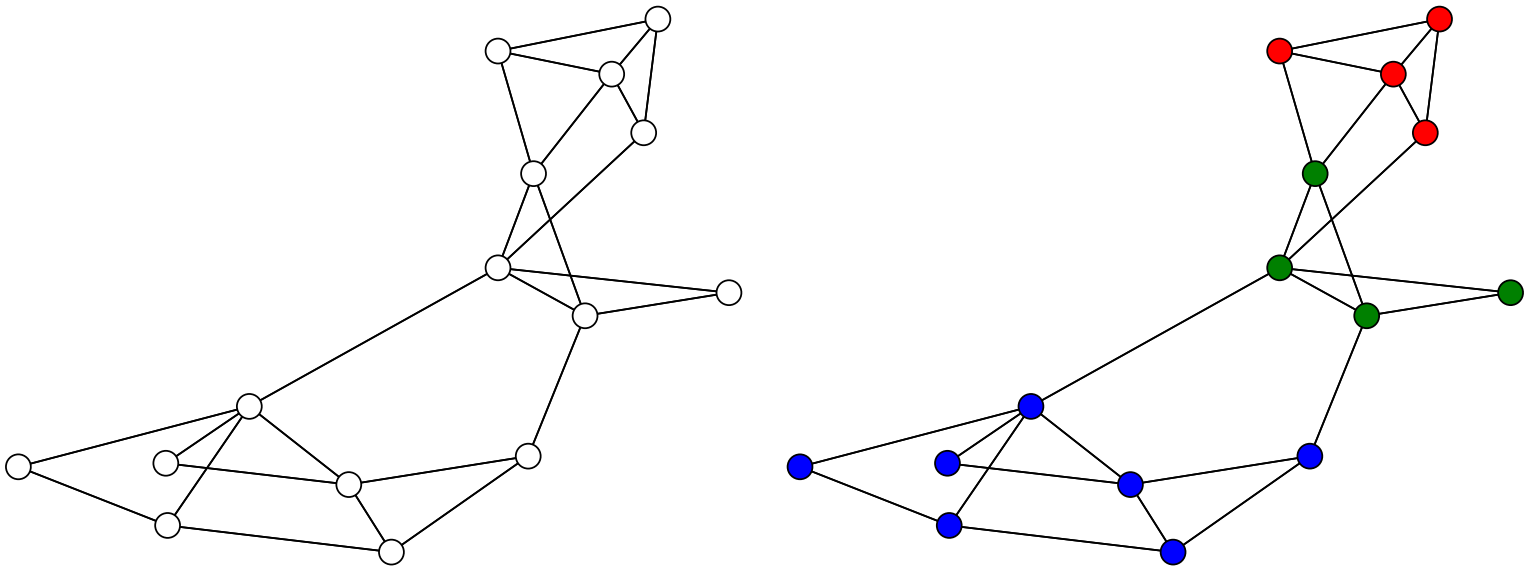


Maximizing modularity

Consider the following problem:

$$\max_{C} Q(C)$$

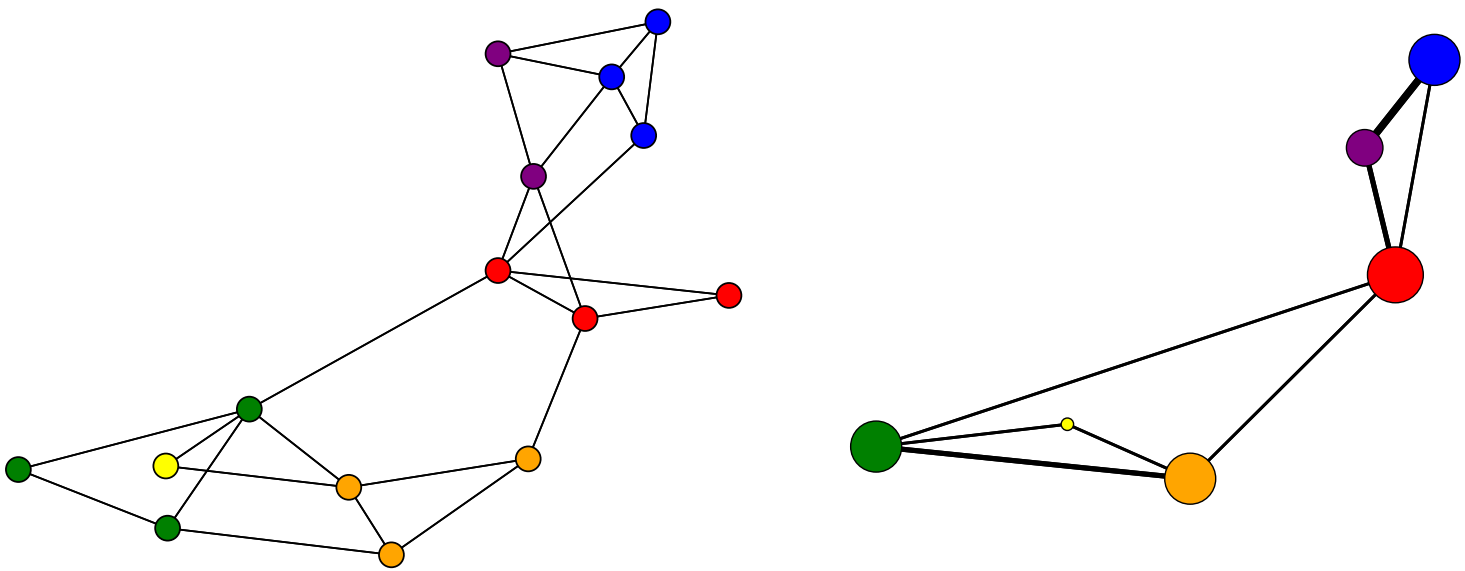
- This problem is combinatorial!
- NP-hard



The Louvain algorithm[1]

Greedy algorithm:

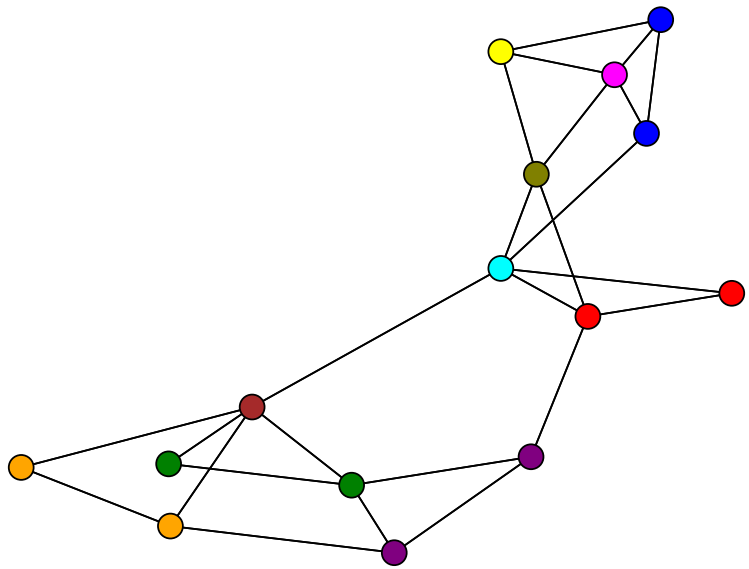
1. **(Initialization)** $C \leftarrow$ identity
2. **(Maximization)** Consider each node sequentially and change its cluster if the modularity $Q(C)$ increases
3. **(Aggregation)** Aggregate the graph and go to step 2



[1] Blondel, Guillaume, Lambiotte & Lefebvre 2008

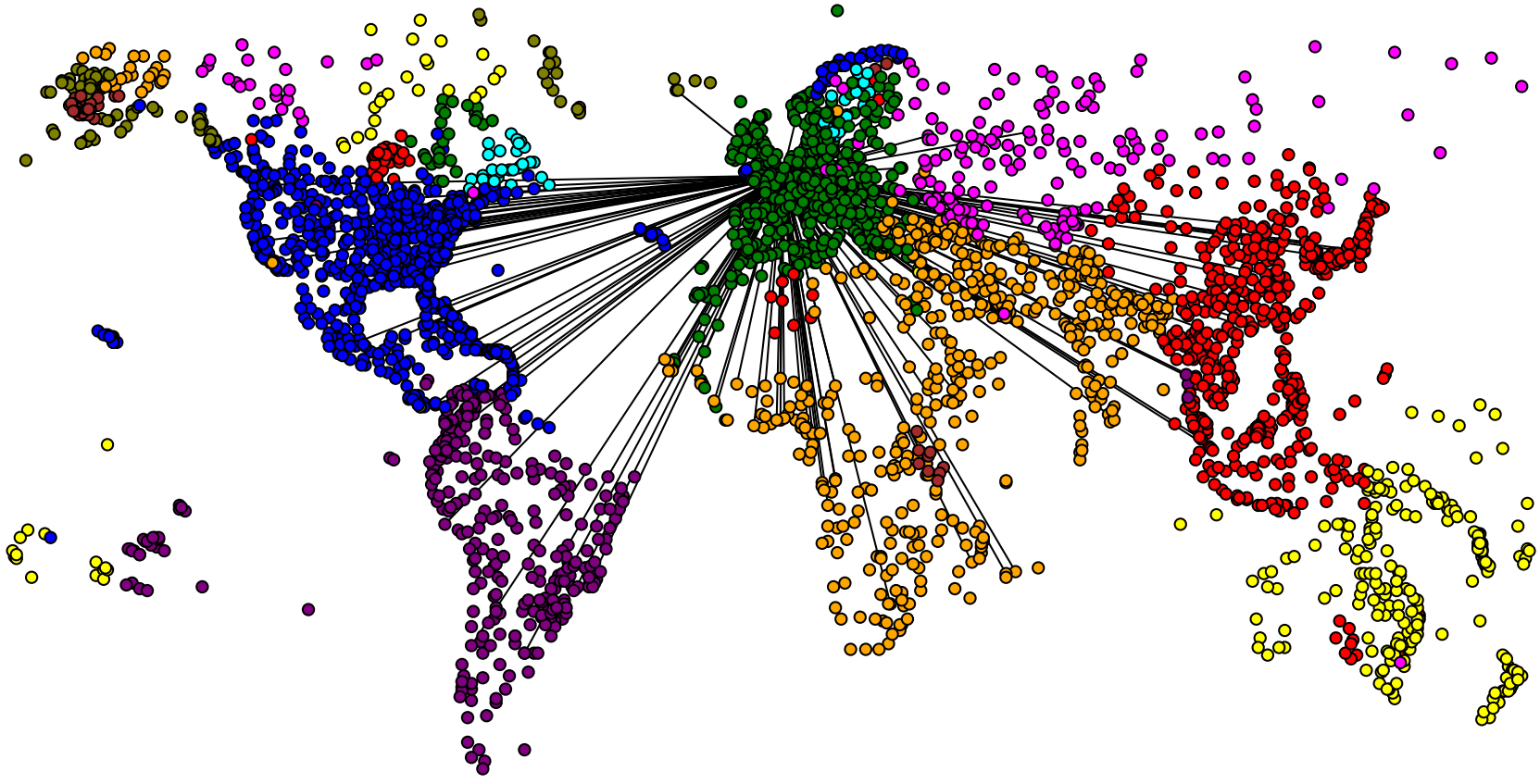
Some observations

- The outcome depends on the **order** in which nodes are considered in the maximization
- The **time complexity** of the maximization is $O(m)$ per iteration, where m is the number of edges
- A **tolerance** parameter can added to speed up the algorithm
- Some **variants** exist (e.g., Leiden algorithm[1])



[1] Traag, Waltman & Van Eck 2019

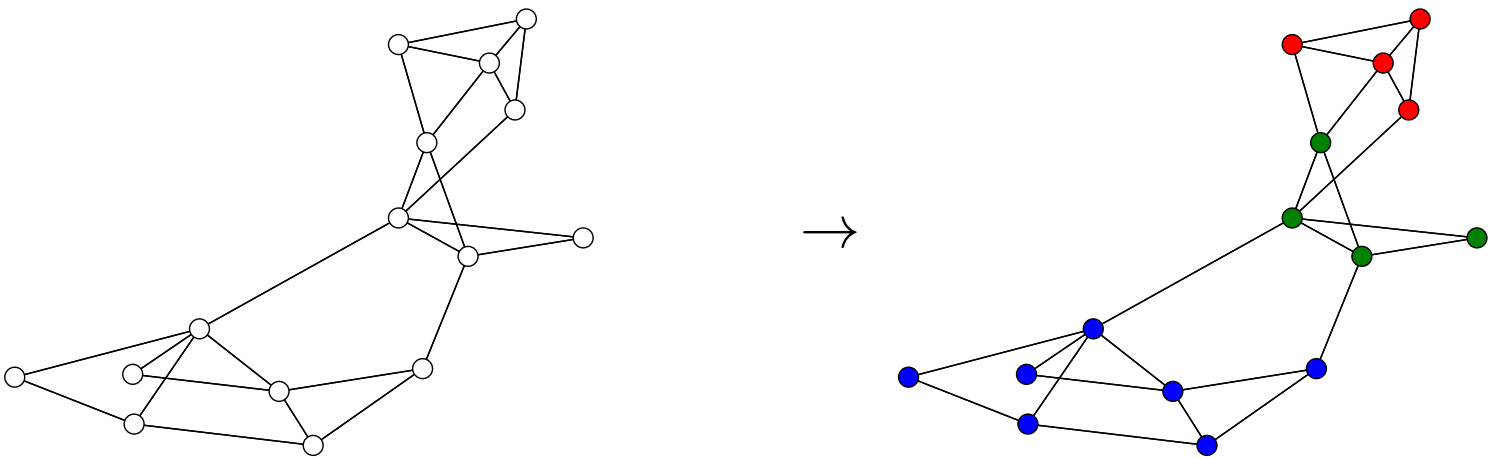
Example



Clustering of Openflights by Louvain
(3,097 nodes, 36,386 edges)

Outline

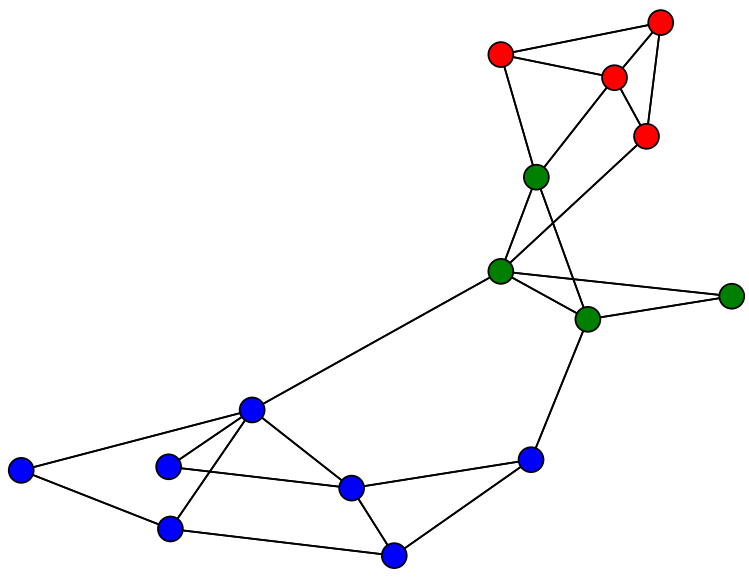
1. Modularity
2. The Louvain algorithm
3. **Cluster strengths**
4. Resolution
5. Extensions



Cluster strength

The **strength** of cluster k is defined by:

$$\sigma_k = \frac{\text{total internal degree}}{\text{total degree}} = \frac{2m_k}{v_k}$$

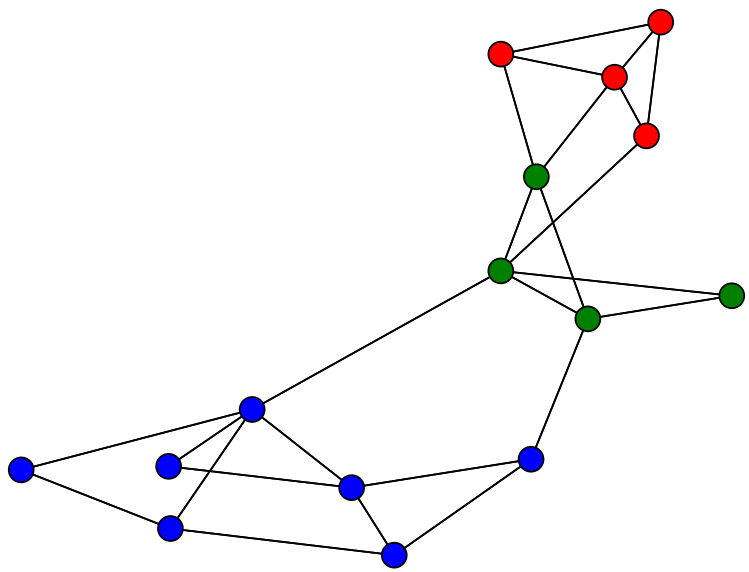


Link with modularity

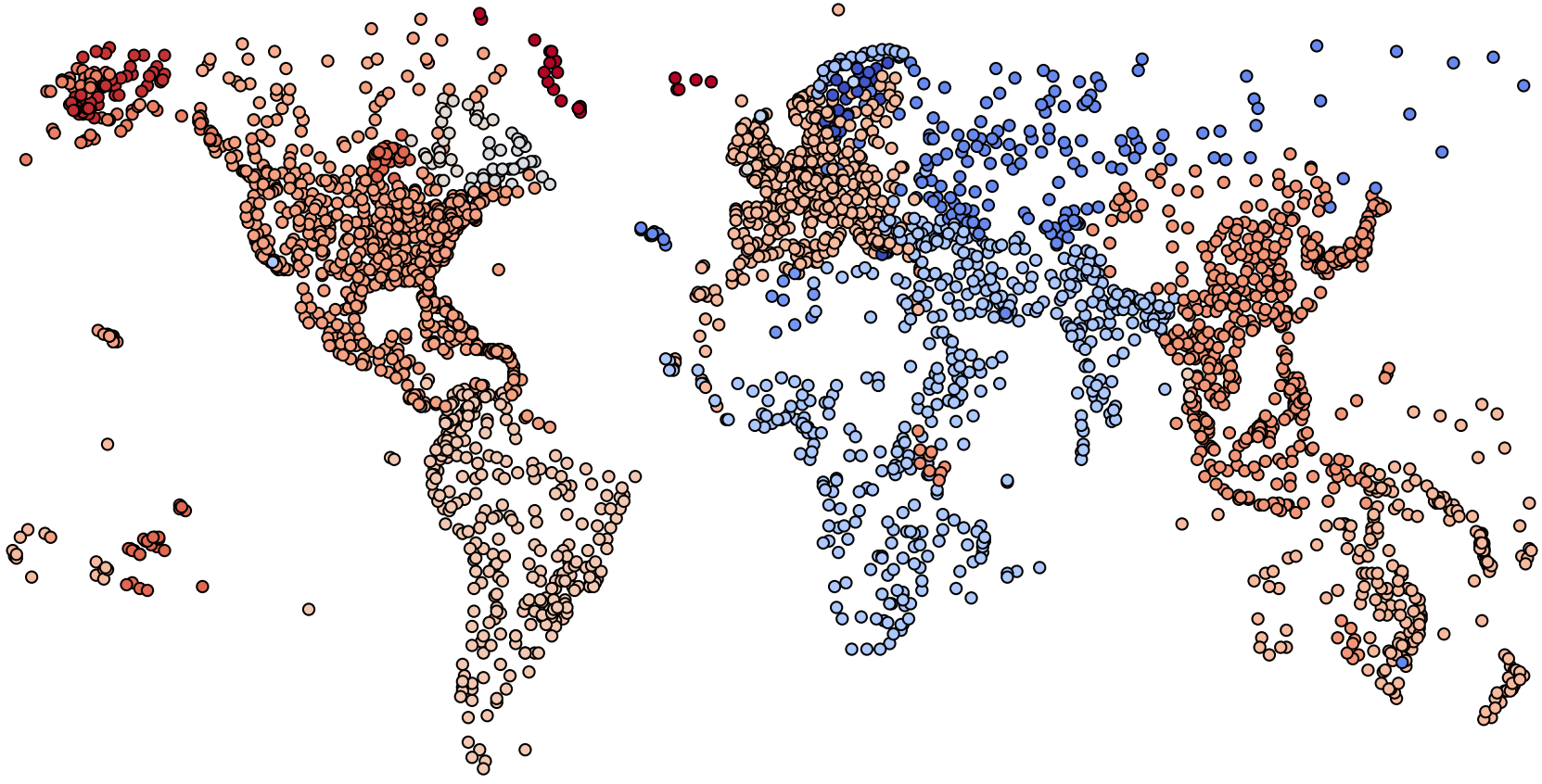
$$Q(C) = \sum_k \pi_k(\sigma_k - \pi_k)$$

σ_k = probability of staying in cluster k after one move

π_k = probability of being in cluster k



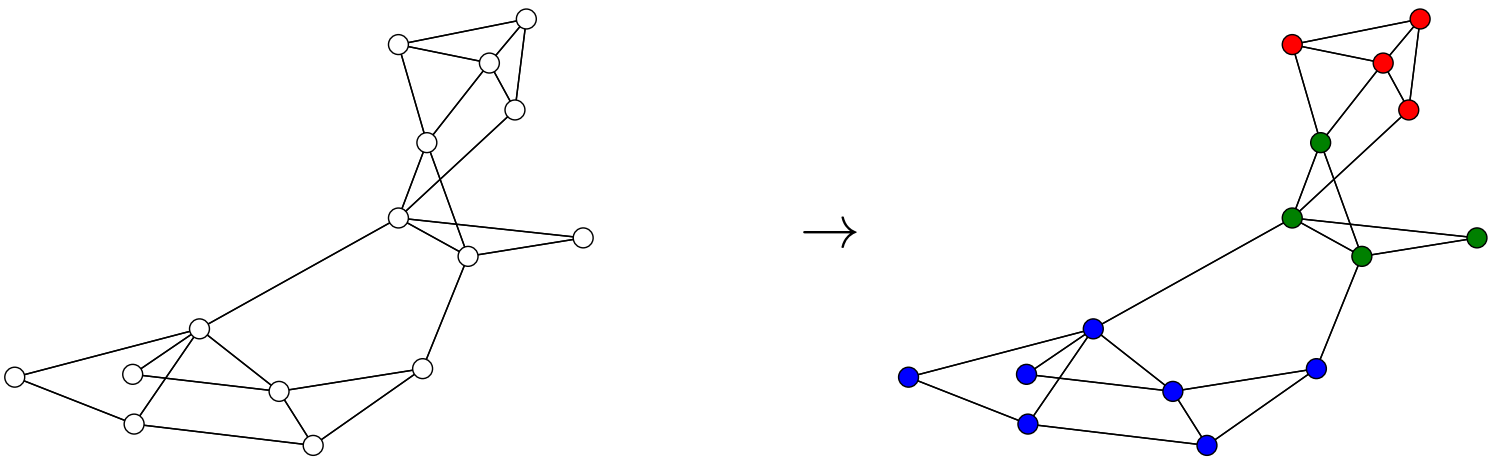
Example



Cluster strengths of Openflights
(3,097 nodes, 36,386 edges)

Outline

1. Modularity
2. The Louvain algorithm
3. Cluster strengths
4. **Resolution**
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,

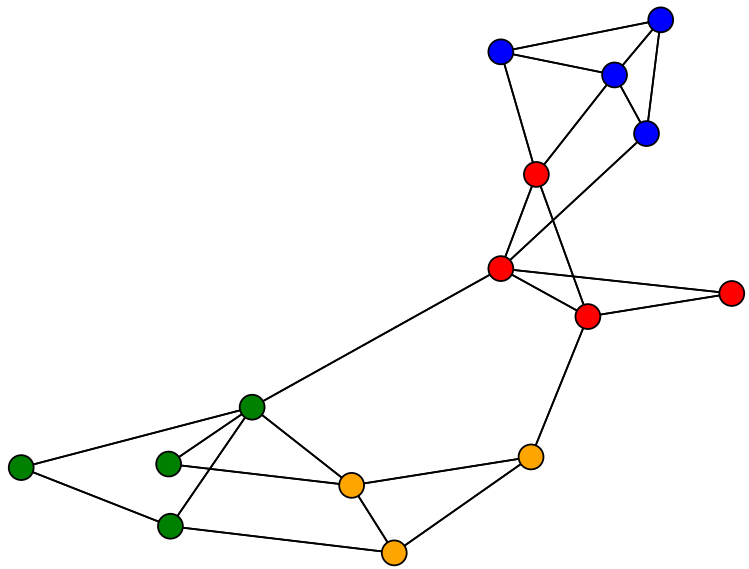
$$\sum_k \left(\frac{v_k}{v}\right)^2 \approx \frac{1}{K} \approx 0$$

Modularity is not able to detect **small** clusters!

Modularity with resolution

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

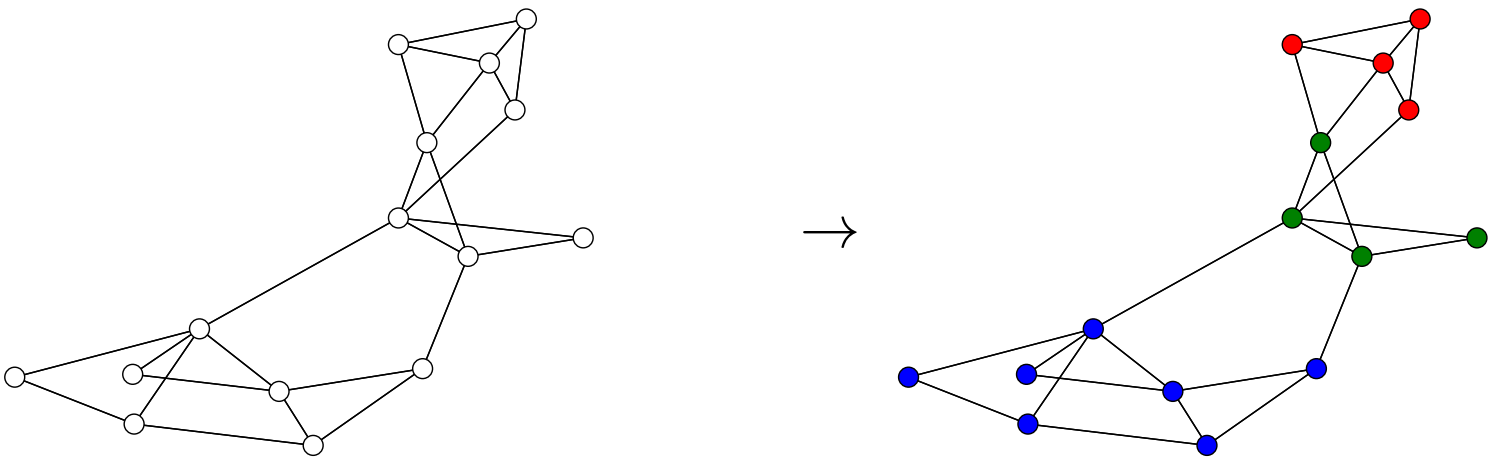
$$Q_\gamma(C) = \frac{1}{v} \sum_{i,j \in V} \left(A_{ij} - \gamma \frac{d_i d_j}{v} \right) \delta_{C(i),C(j)}$$



$\gamma = 2$

Outline

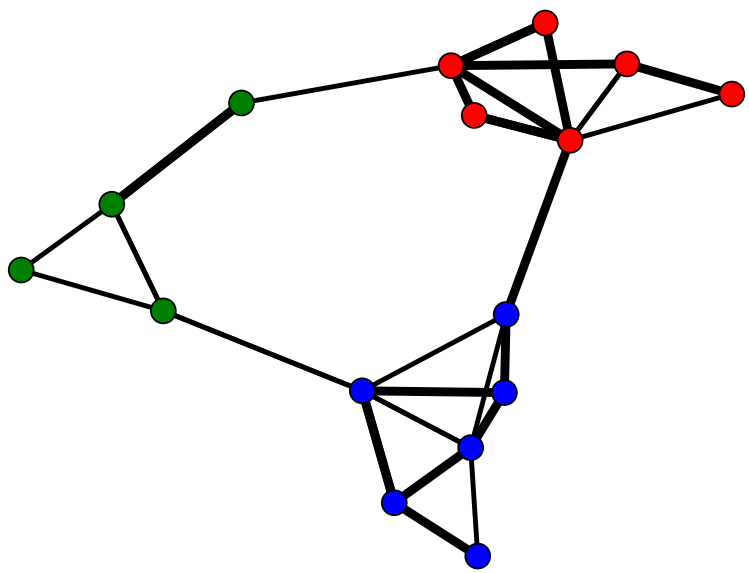
1. Modularity
2. The Louvain algorithm
3. Cluster strengths
4. Resolution
5. **Extensions**



Case of weighted graphs

Let $G = (V, E)$ be a **weighted** graph with adjacency matrix A
Let $w = A1$ be the vector of node weights
The **modularity** of clustering C is defined by:

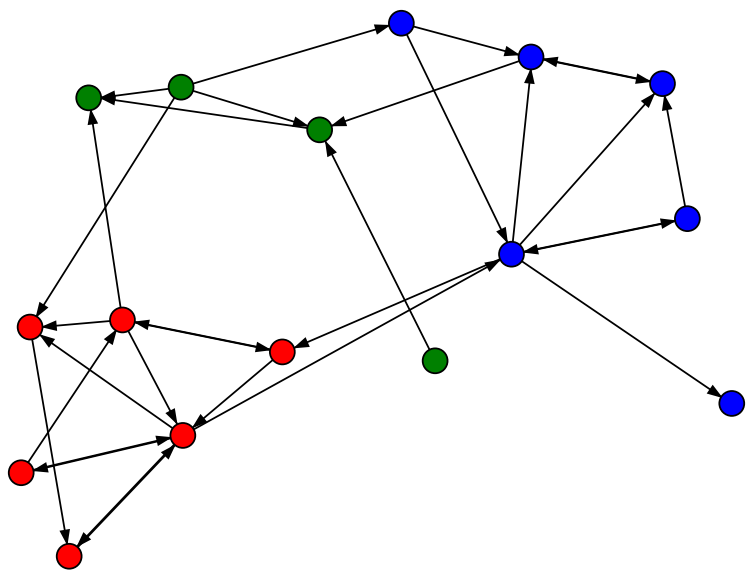
$$Q(C) = \frac{1}{v} \sum_{i,j \in V} \left(A_{ij} - \frac{w_i w_j}{v} \right) \delta_{C(i),C(j)}$$



Case of directed graphs

Let $G = (V, E)$ be a **directed** graph with adjacency matrix A
The **modularity** of clustering C is defined by[2]:

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



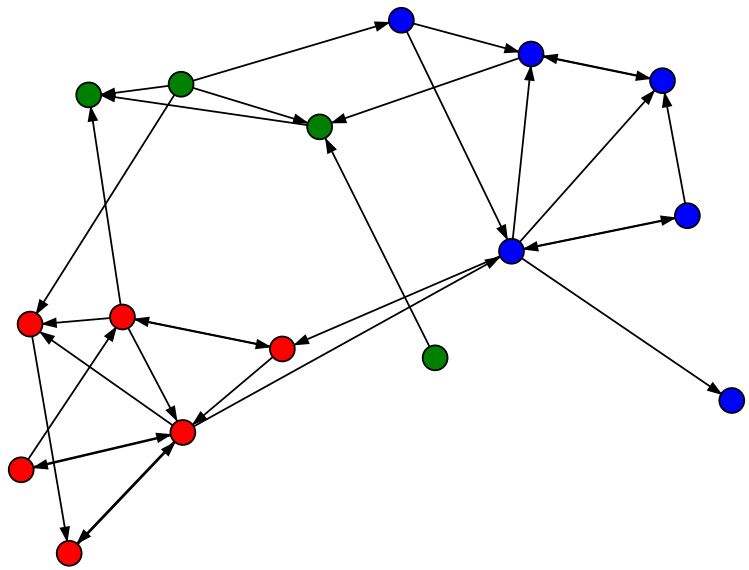
[2] Dugué 2015

Cluster-level expression

The modularity can be written:

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

with m_k the **size** (number of edges) and v_k^+, v_k^- the total out-degrees and in-degrees of cluster k

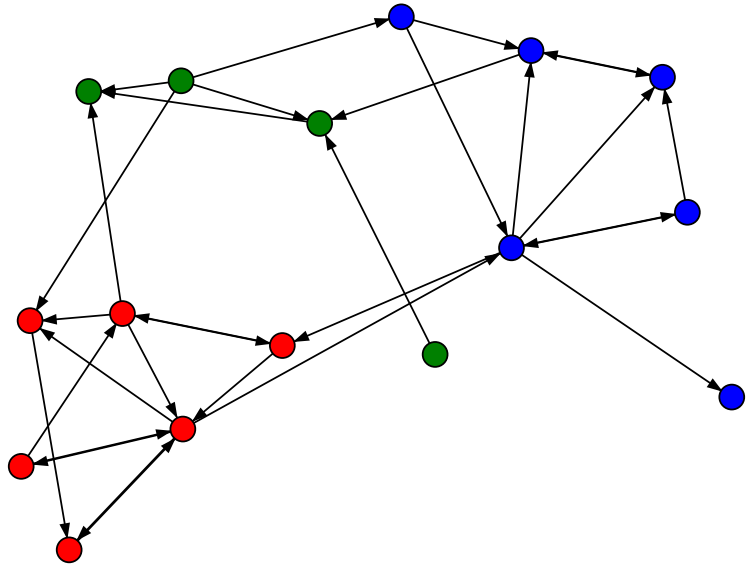


Cluster strength

The **strength** of cluster k is defined by:

$$\sigma_k = \frac{\text{total internal degree}}{\text{total out-degree}} = \frac{m_k}{v_k^+}$$

$=$ probability of staying in cluster k after one move

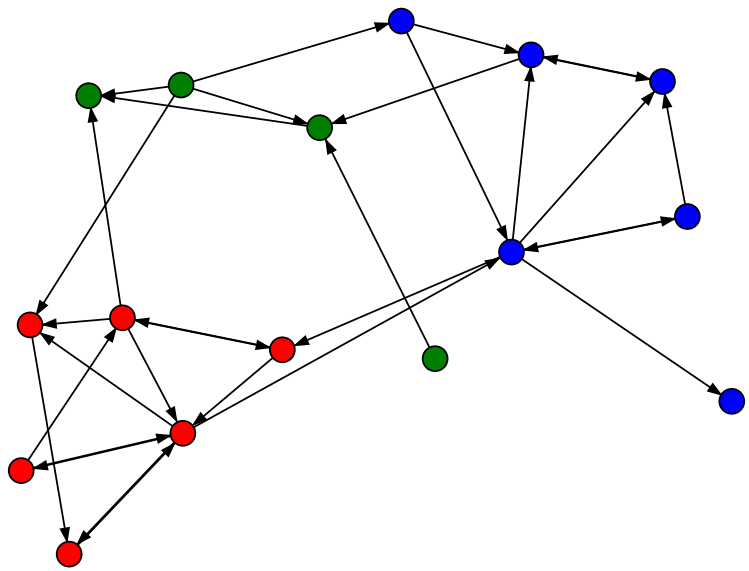


Link with modularity

The modularity can be written:

$$\boxed{Q(C) = \sum_k \pi_k^+ (\sigma_k - \pi_k^-)}$$

σ_k = probability of staying in cluster k after one move
π_k^+, π_k^- = probability of sampling cluster k from out/in-degrees



Bipartite graphs

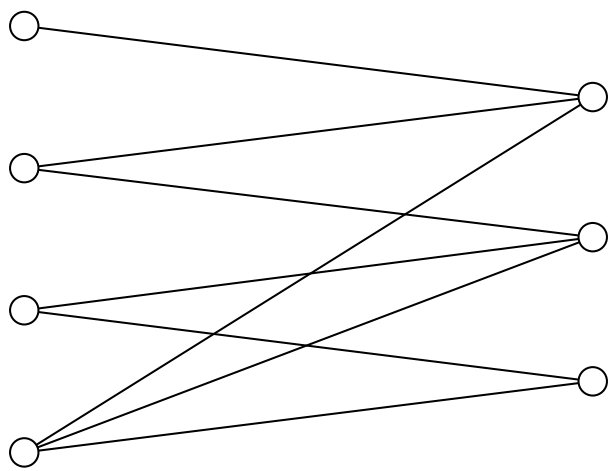
Seen as undirected...

$$A = \begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix}$$

or directed[3]...

$$A = \begin{bmatrix} 0 & B \\ 0 & 0 \end{bmatrix}$$

Co-clustering!



[3] Barber 2007

Summary

Graph clustering

- Notion of **modularity** $\rightarrow$ quality metric
- The **Louvain** algorithm $\rightarrow$ applicable to massive graphs
- The **resolution** parameter $\rightarrow$ to explore different scales
- Applicable to **weighted**, **directed** and **bipartite** graphs