Hierarchical Graph Clustering: Quality Metrics & Algorithms

Thomas Bonald

Joint work with
Bertrand Charpentier, Alexis Galland & Alexandre Hollocou

LTCI Data Science seminar
March 2019

INSTITUT
POLYTECHNIQUE
DE PARIS

TECHNISCHE
UNIVERSITÄT
MÜNCHEN

Motivation

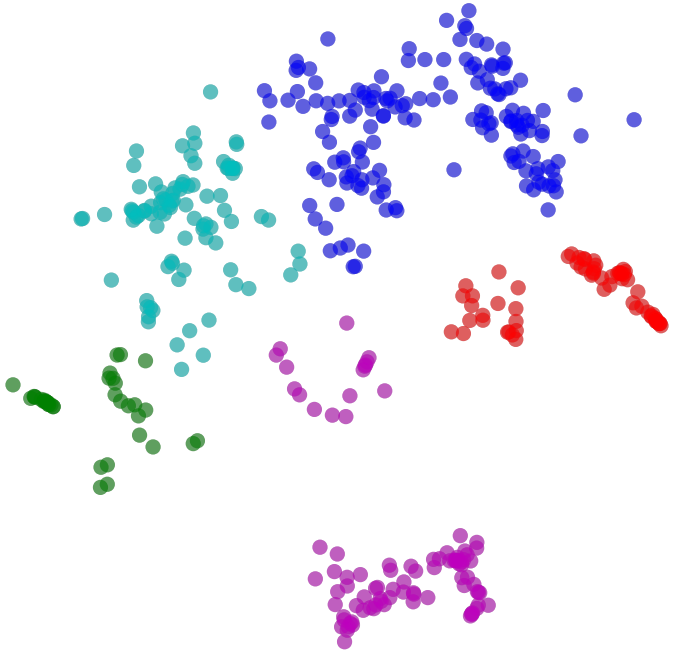
- **Clustering** is a fundamental problem in data science
- The objective is to group together items that are "similar" to each other $\longrightarrow$ **unsupervised learning**

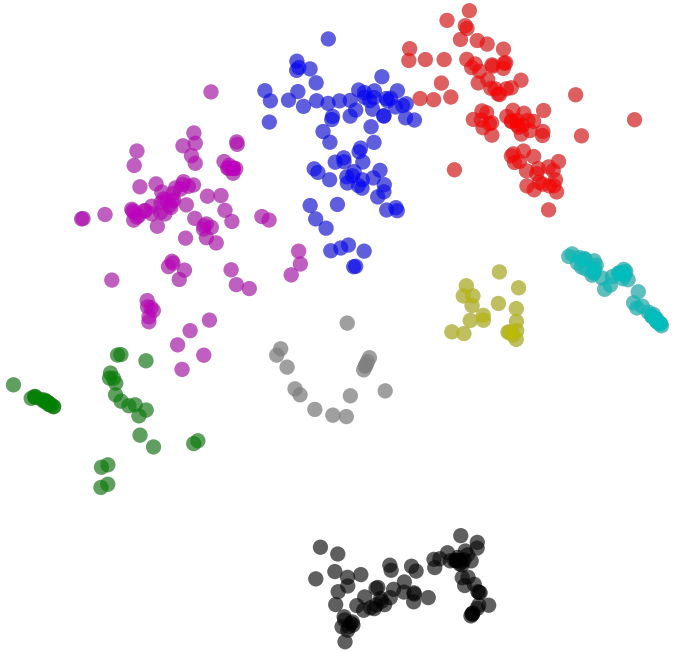
Many applications:

- Recommendation
- Anomaly detection
- Visualization
- Storage / processing
- Search engines
- Image segmentation
- NLP

An ill-posed problem

- What is a **good** clustering?
- **How many** clusters?





Kleinberg's impossibility theorem

Viewing clustering as a function $f : \mathbb{R}^{n \times d} \to P(\{1, \ldots, n\})$

Axioms

1. **Scale-invariance:** $\forall \alpha > 0, f(\alpha x) = f(x)$
2. **Richness:** f surjective
3. **Consistency:** $\forall y \succ x, f(y) = f(x)$

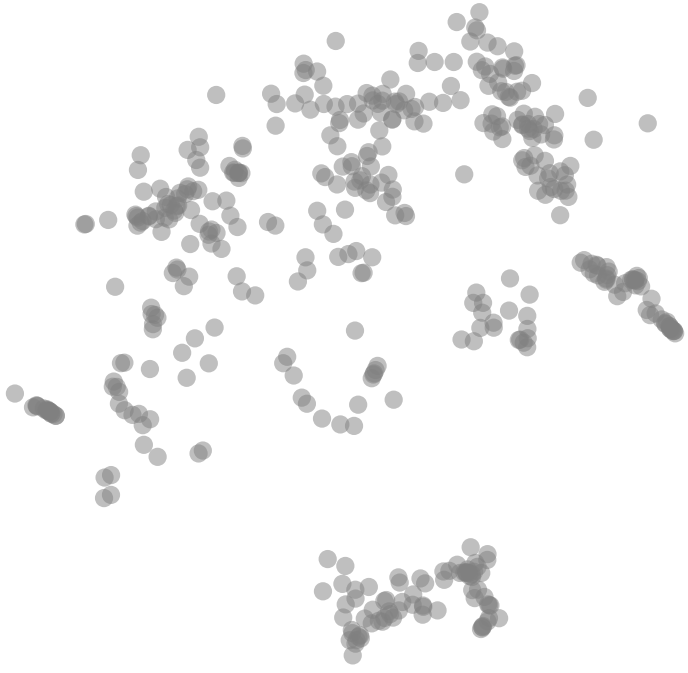
There is **no** clustering function f satisfying these 3 axioms!
Kleinberg, NIPS 2002

In fact, this is possible with 3 replaced by:

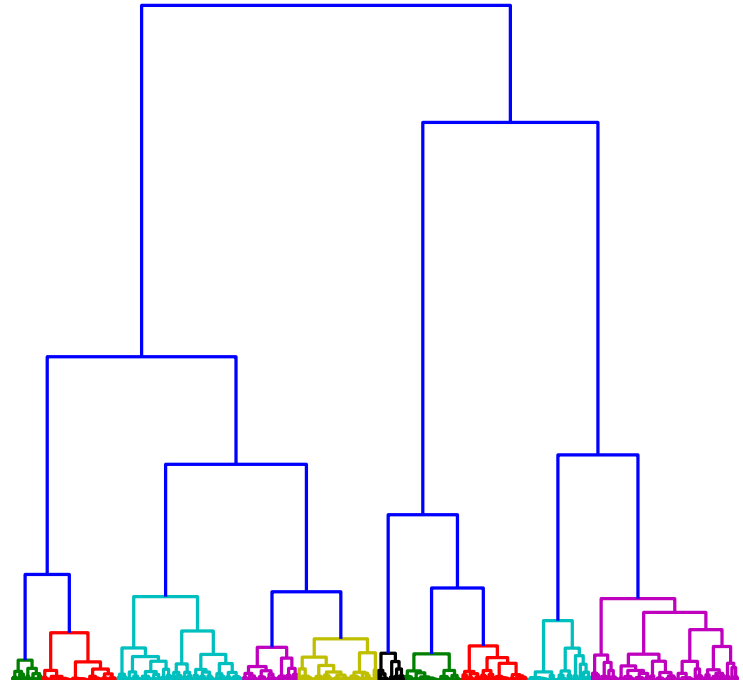
3'. **Refined consistency:** $\forall y \succ x, f(y) = f(x)$ or $|f(y)| \neq |f(x)|$

Cohen-Addad, Kanade & Mallmann-Trenn, NIPS 2018

Hierarchical clustering



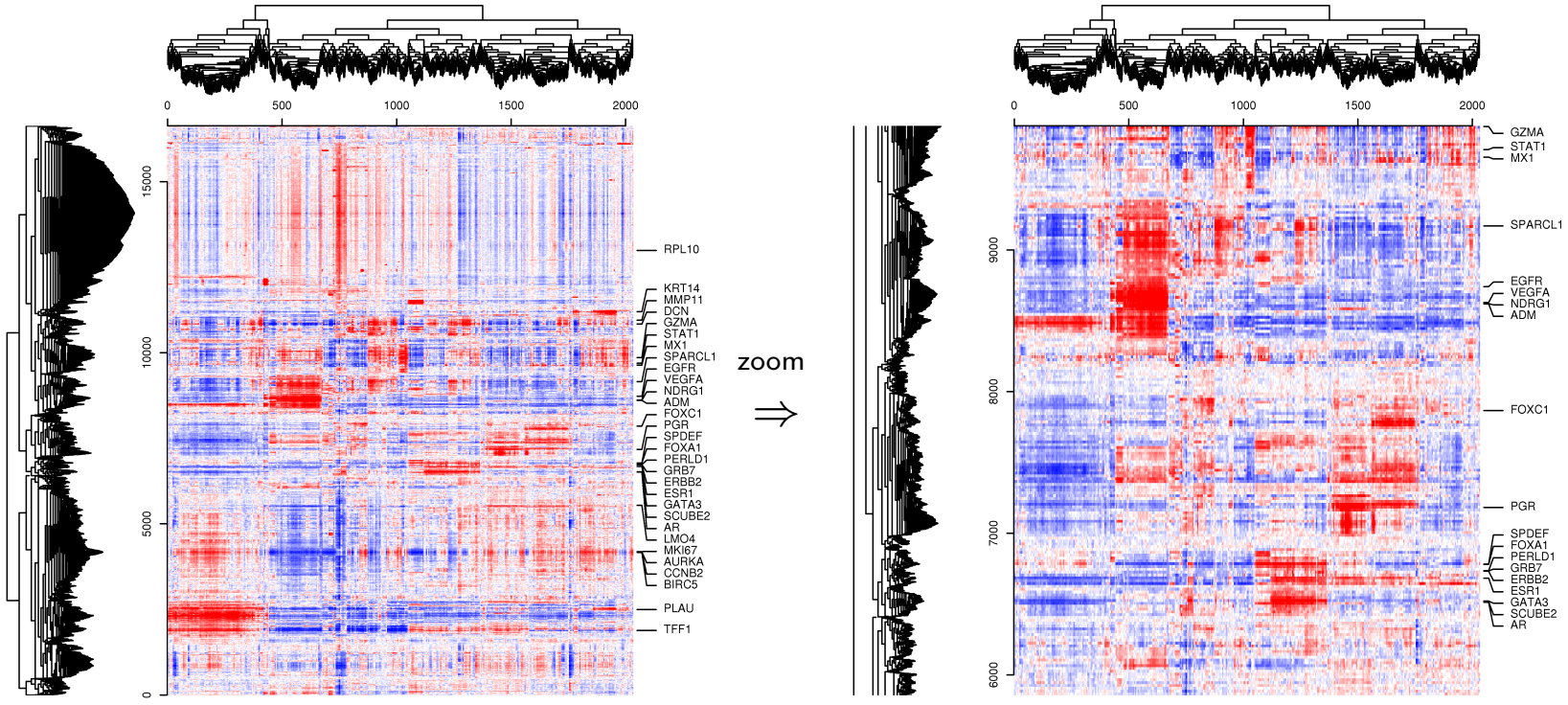
Data



Dendrogram

Example in biology

2,035 tumors, 16,634 non-redundant genes



Wirapati 2009

Hierarchical clustering algorithms

Divisive algorithms

- e.g., through successive k-means

Agglomerative algorithms

- Successive merges of the closest clusters $a, b \subset \{1, \ldots, n\}$

Linkage	$d(a, b)$
Single	$\min_{i \in a, j \in b} \|\|x_i - x_j\|\|$
Complete	$\max_{i \in a, j \in b} \|\|x_i - x_j\|\|$
Average	$\frac{1}{\|a\|\|b\|} \sum_{i \in a, j \in b} \|\|x_i - x_j\|\|$
Ward	$\frac{\|a\|\|b\|}{\|a\|+\|b\|} \|\|g_a - g_b\|\|^2$

Lance & Williams 1967

- Local search by the **nearest-neighbor chain**
  Bruynooghe 1977, Benzécri 1982, Murtagh 1983

Graph data

Many datasets can be represented by graphs:

- social networks, transport networks, databases, etc.
  $\rightarrow$ **explicit links**
- authors-papers, words-documents, consumers-products, etc.
  $\rightarrow$ **implicit links**

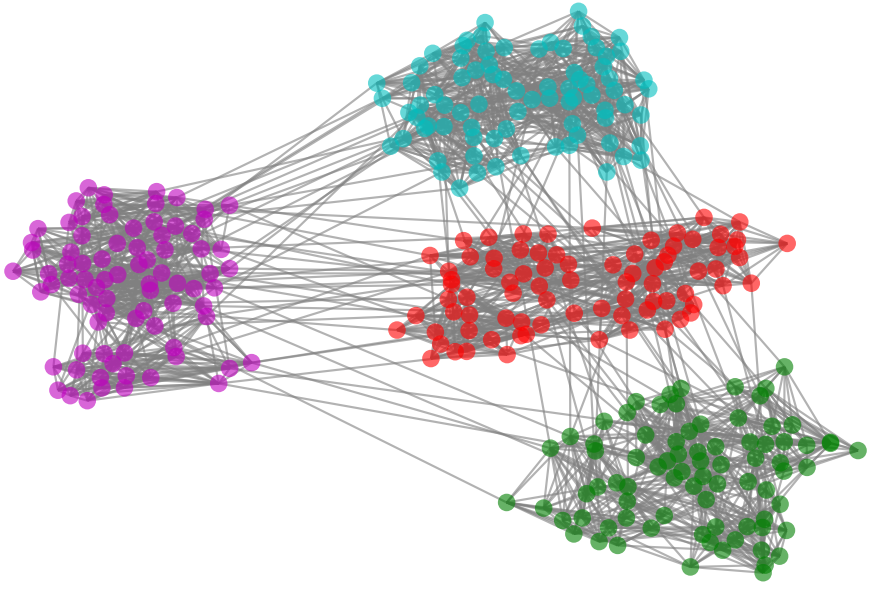
These graphs can be represented by **sparse** matrices

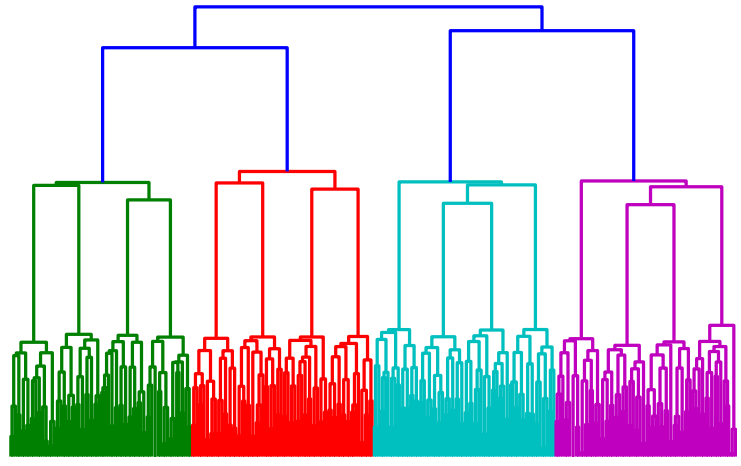
Dataset	#nodes	#edges	density
Amazon	335k	925k	$\approx 10^{-5}$
Wikipedia	12M	378M	$\approx 10^{-6}$
Twitter	42M	1.5G	$\approx 10^{-6}$

Usual clustering algorithms do not apply as pairwise distances are **not** defined and the number of node pairs is **huge**!

Questions

1. How to **cluster** a graph?
2. How to assess the **quality** of this clustering?





Outline

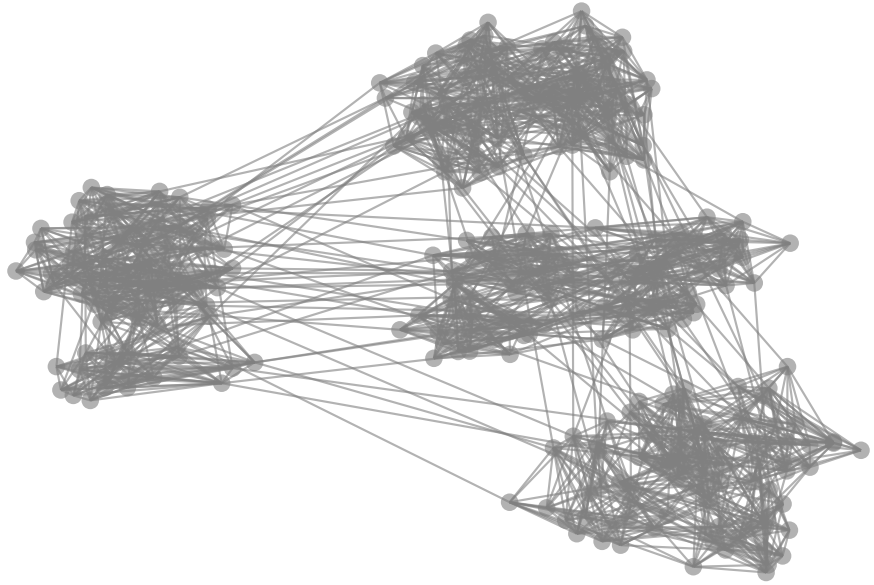
1. Node pair sampling
2. Flat clustering
3. Hierarchical clustering
4. Quality metric
5. Experiments

Notation

Weighted, **undirected** graph G of n nodes
The weights represent the **strengths** of the links

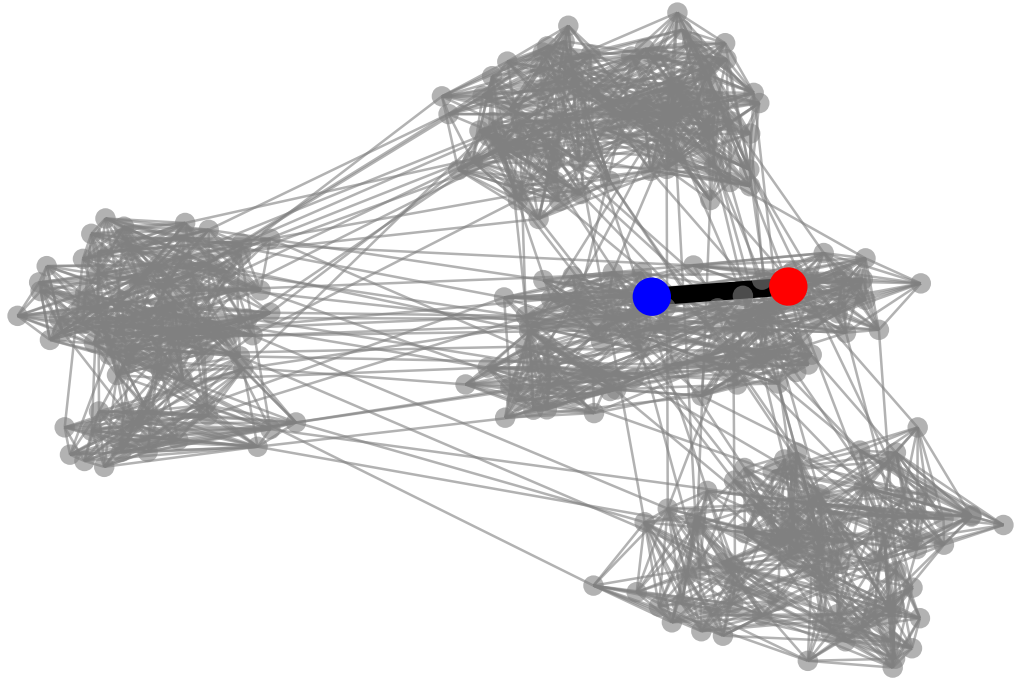
$$w_{ij} = \begin{cases} \text{weight of edge } i,j, \text{ if any} \\ 0 \text{ otherwise} \end{cases}$$

$$w_i = \sum_j w_{ij} \qquad w = \sum_i w_i = \sum_{i,j} w_{ij}$$



Node pair sampling

$$p(i,j) = \frac{w_{ij}}{w} \qquad p(i) = \sum_j p(i,j) = \frac{w_i}{w}$$

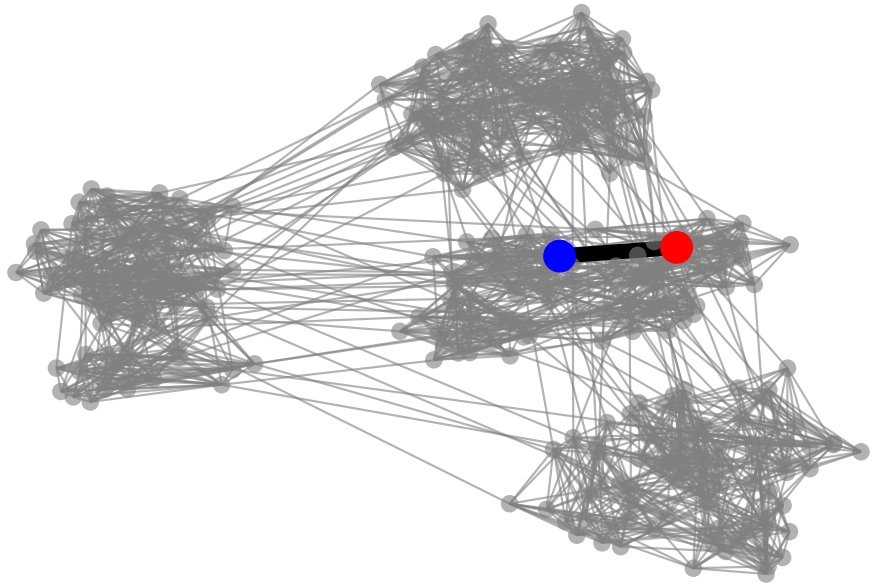


Entropy

A simple metric for assessing the **complexity** of the graph:

$$H = -\sum_{i,j} p(i,j) \log p(i,j)$$

Note: This is not what is known as graph entropy...
Körner 1973



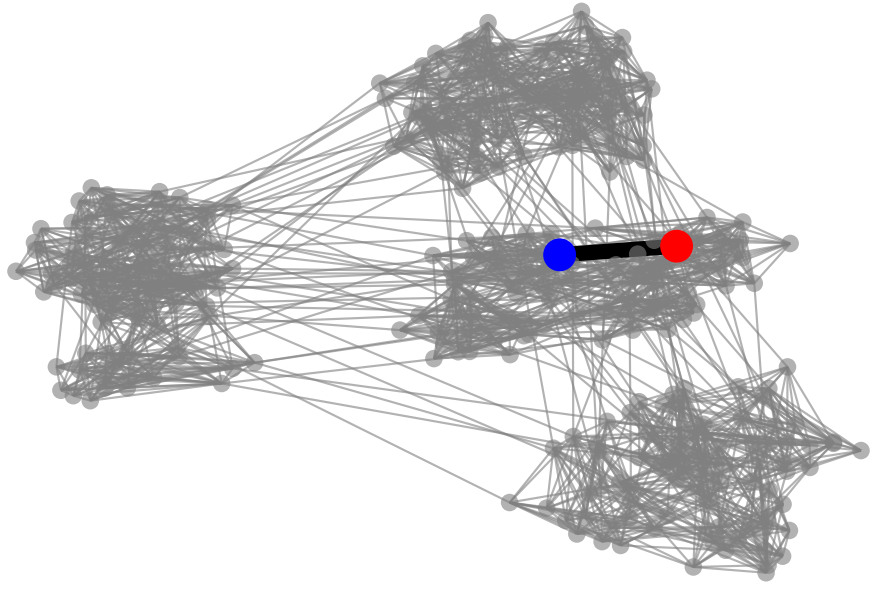
$H \approx 12$ bits

Mutual information

A simple metric for assessing the **clustering** structure of the graph:

$$I = \sum_{i,j} p(i,j) \log \frac{p(i,j)}{p(i)p(j)}$$

Alush, Friedman & Goldberger 2016



$I \approx 4$ bits

Outline

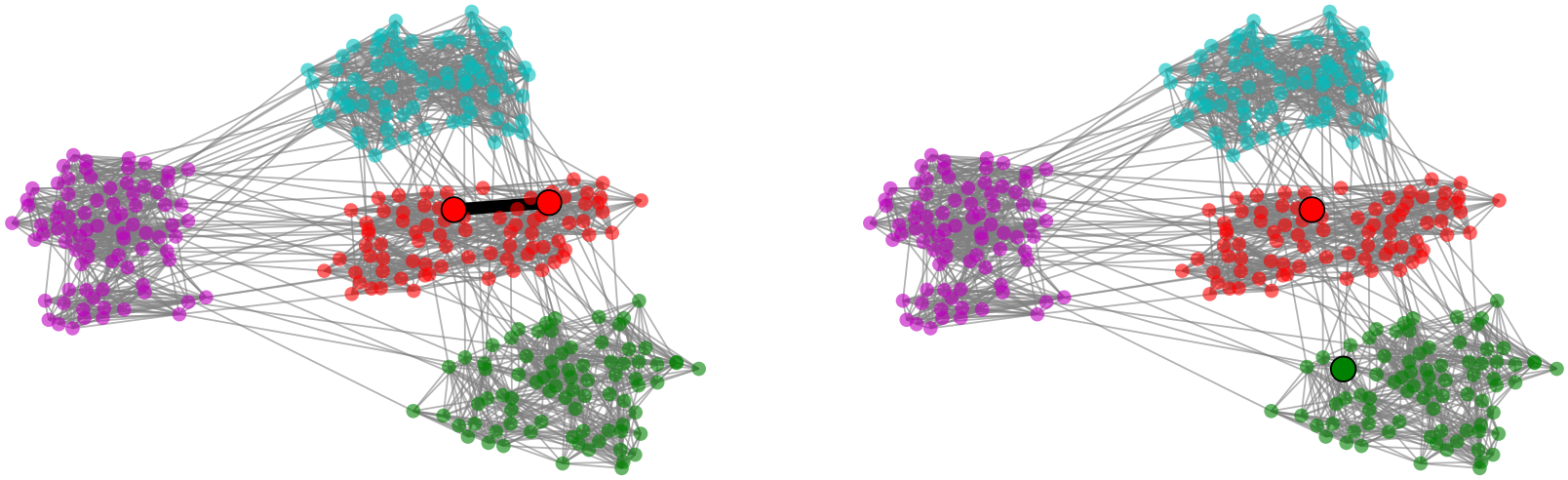
1. Node pair sampling
2. **Flat clustering**
3. Hierarchical clustering
4. Quality metric
5. Experiments

Modularity

Quality of a clustering $c : \{1, \ldots, n\} \to \{1, \ldots, k\}$

$$M(c) = \sum_{i,j} (p(i,j) - p(i)p(j))\, \delta_{c(i),c(j)}$$

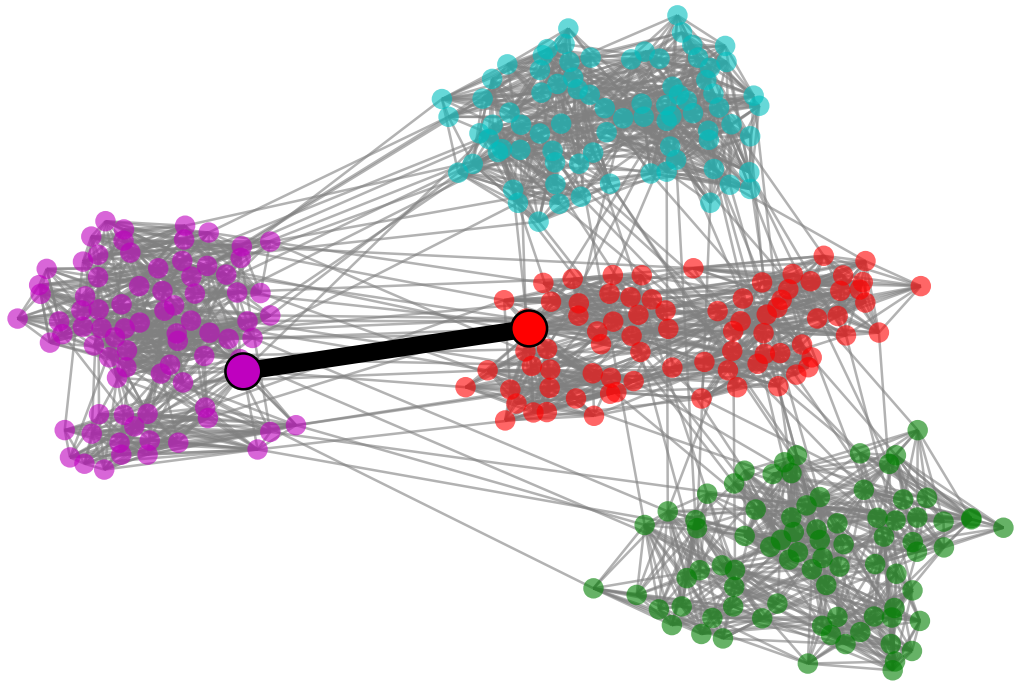
Newman & Girvan 2004



Cluster pair sampling

For any clustering $C \in P(\{1, \ldots, n\})$:

$$\forall a, b \in C, \quad p(a, b) = \sum_{i \in a, j \in b} p(i, j) \qquad p(a) = \sum_{i \in a} p(i)$$

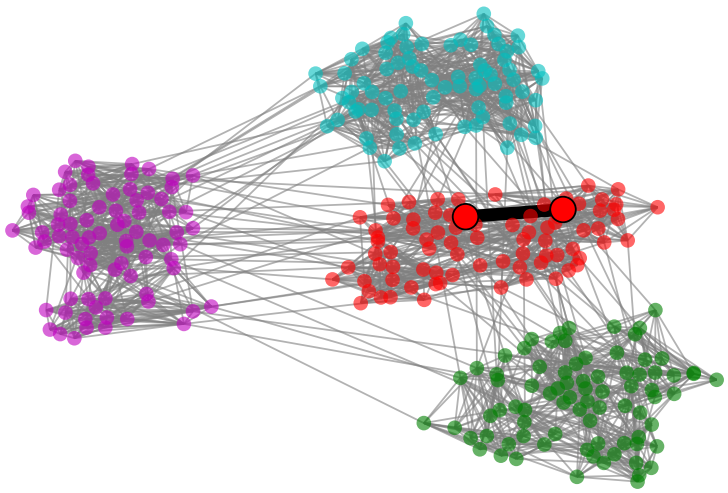


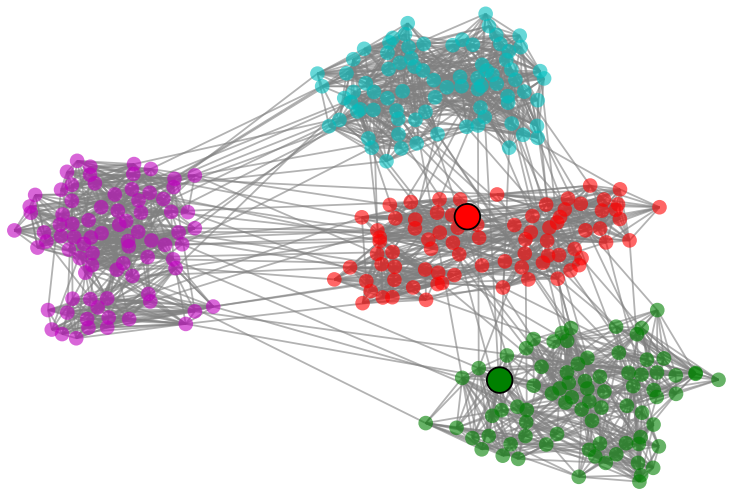
Modularity at cluster level

Quality of a clustering $C \in P(\{1, \ldots, n\})$:

$$M(C) = \sum_{c \in C} p(c, c) - \sum_{c \in C} p(c)^2$$

Simpson 1949

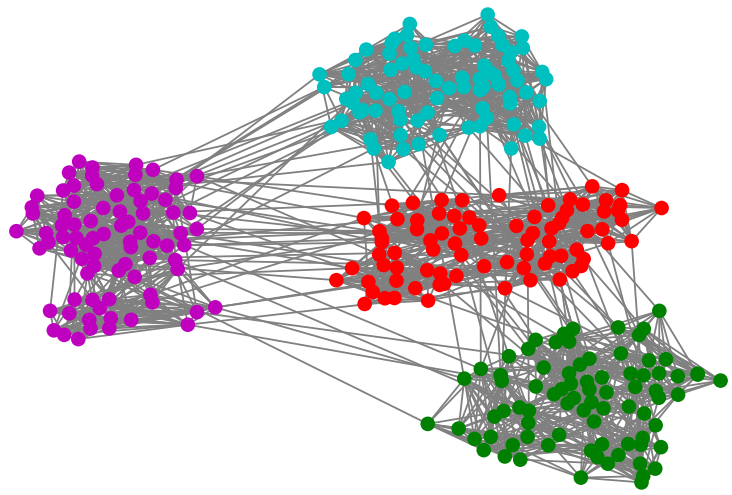




Modularity maximization

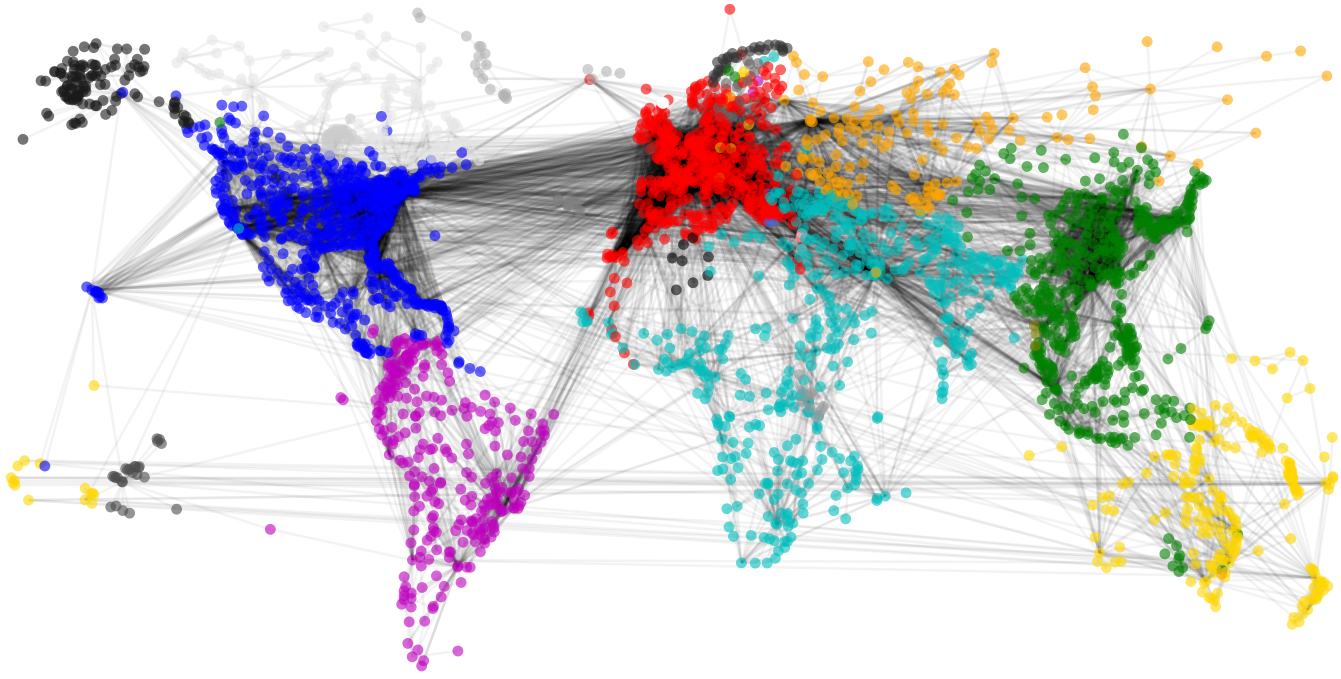
$$\max_C M(C)$$

- NP-hard problem
- The **Louvain** algorithm, fast and efficient
  Blondel, Guillaume, Lambiotte & Lefebvre 2008



Clustering of OpenFlights by Louvain

3,097 airports, 18,193 flights



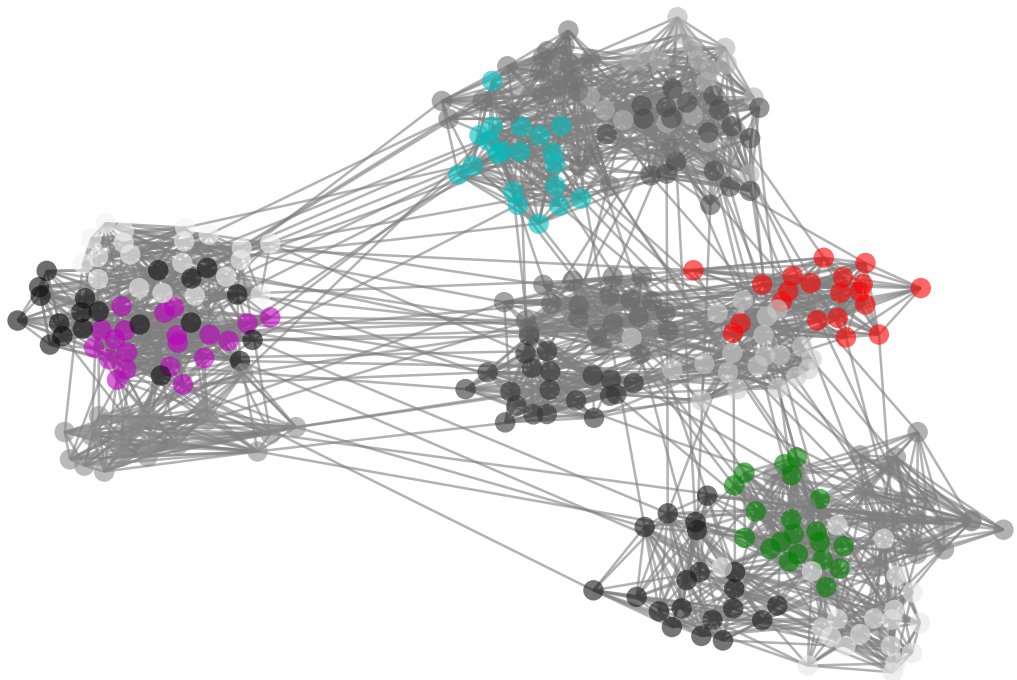
$M(C) \approx 0.66$

Resolution parameter

For some parameter $\gamma > 0$:

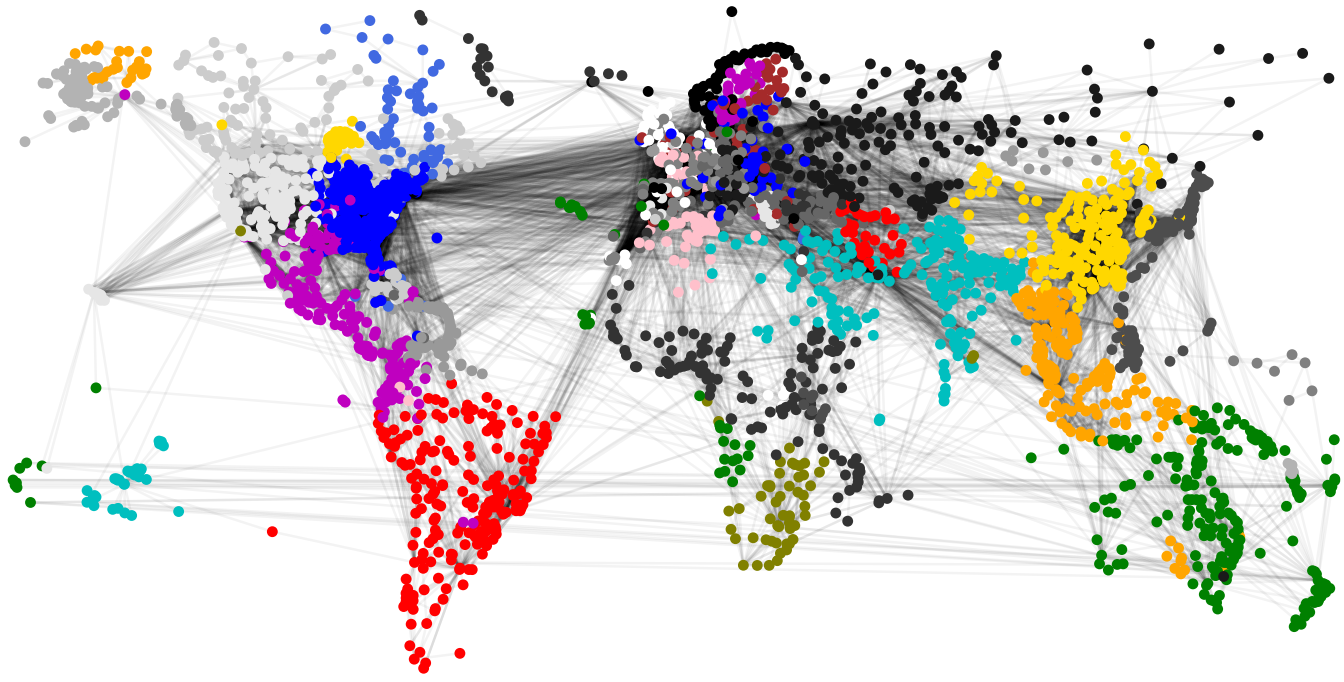
$$M_\gamma(c) = \sum_{i,j} \left(p(i,j) - \gamma p(i)p(j)\right) \delta_{c(i),c(j)}$$

Reichardt & Bornholdt 2006



Clustering of OpenFlights by Louvain

3,097 airports, 18,193 flights



$\gamma = 2$

Outline

1. Node pair sampling
2. Flat clustering
3. **Hierarchical clustering**
4. Quality metric
5. Experiments

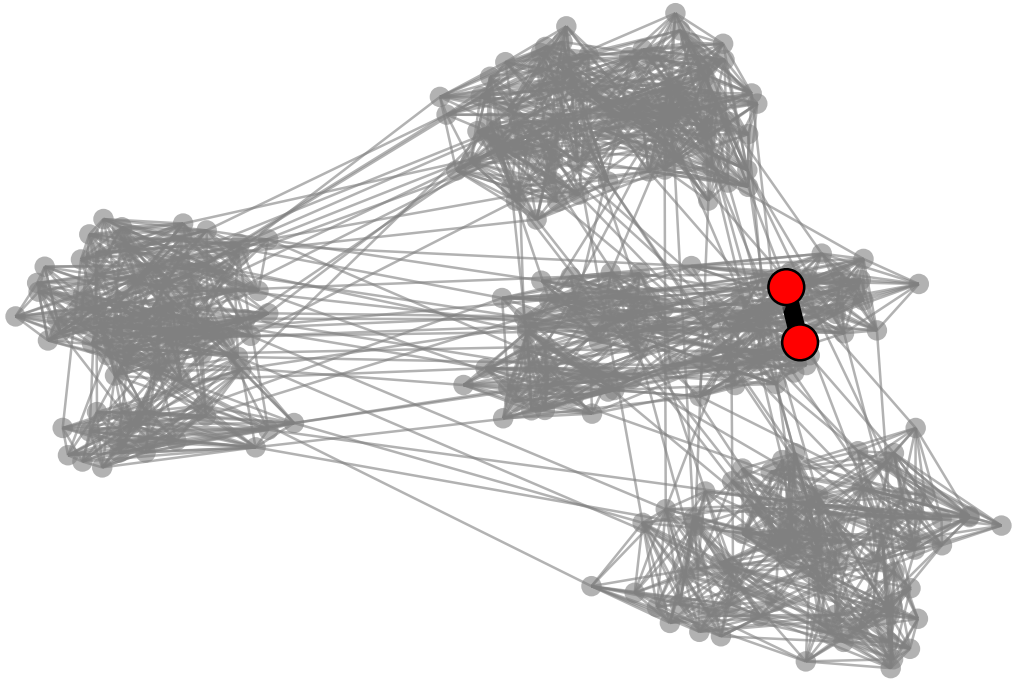
An agglomerative algorithm

We need a measure of **proximity** between nodes

$$\sigma(i,j) = \frac{p(i,j)}{p(i)p(j)}$$

Observe that

$$\sigma(i,j) = \frac{p(i|j)}{p(i)} = \frac{p(j|i)}{p(j)}$$



The maximum resolution

$$M_\gamma(c) = \sum_{i,j} \left(p(i,j) - \gamma p(i)p(j)\right) \delta_{c(i),c(j)}$$

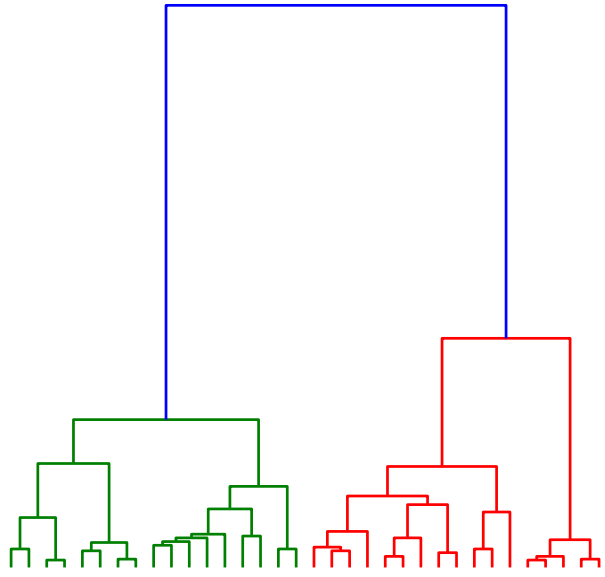
$$\gamma^+ = \max_{i,j} \frac{p(i,j)}{p(i)p(j)}$$

Algorithm

While there are at least 2 nodes:

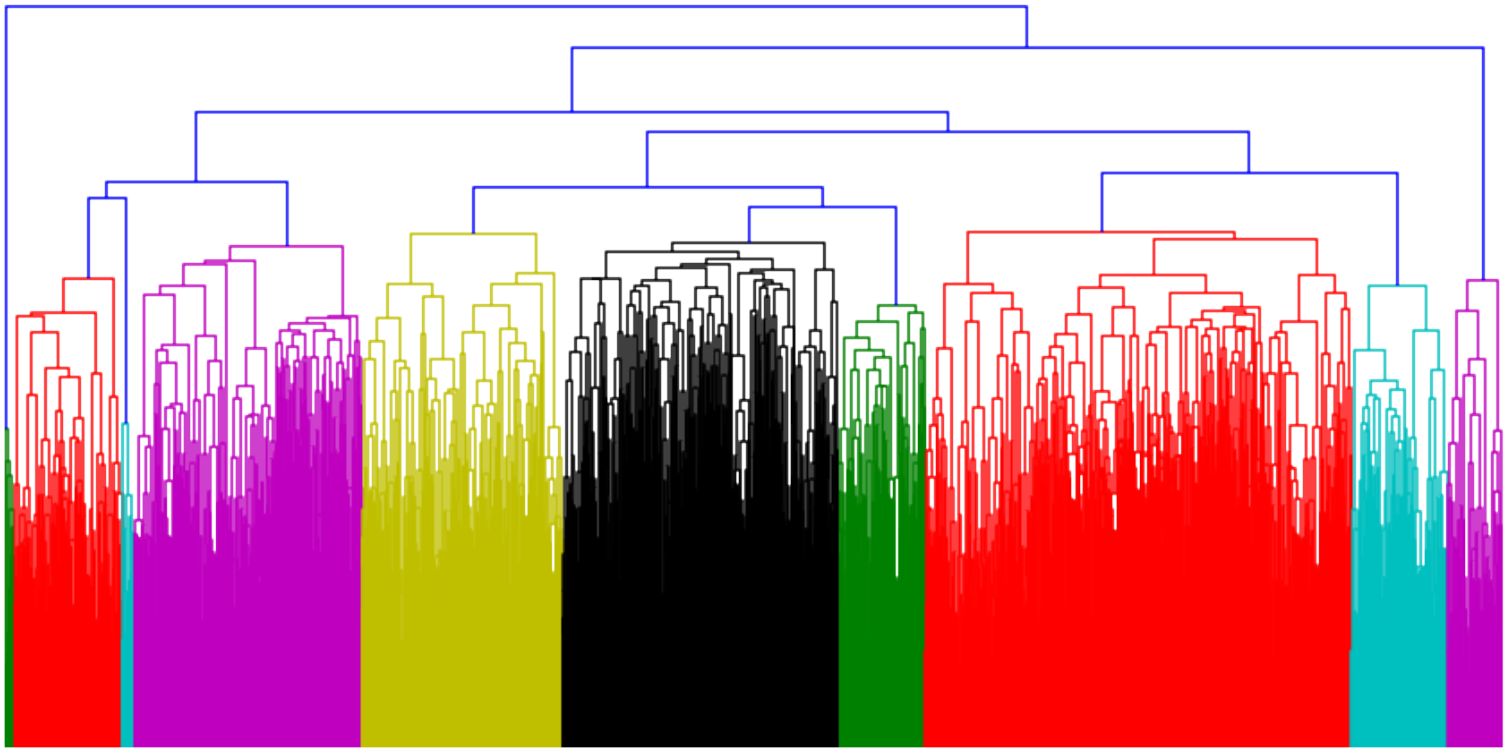
- **find** the node pair i, j maximizing $\sigma(i, j)$
- **merge** nodes i, j
- **update** σ

Sequence of similarities / resolutions $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{n-1}$



Hierarchical clustering of Openflights

3,097 airports, 18,193 flights



Other hierarchical clustering algorithms

Divisive algorithms

- e.g., through successive bisections

Agglomerative algorithms

- Successive merges of the two closest clusters $a, b \subset \{1, \ldots, n\}$

Linkage	$\sigma(a, b)$
Single	$\max_{i \in a, j \in b} p(i, j)$
Average	$\frac{1}{\lvert a \rvert \lvert b \rvert} p(a, b)$
Sampling ratio	$\frac{p(a,b)}{p(a)p(b)}$

- Local search by the **nearest-neighbor chain**

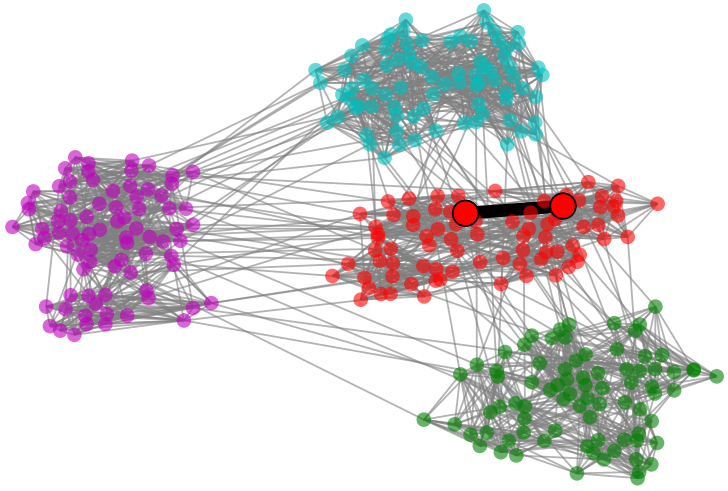
See also Newman 2004, Pons & Latapy 2005, Chang 2011

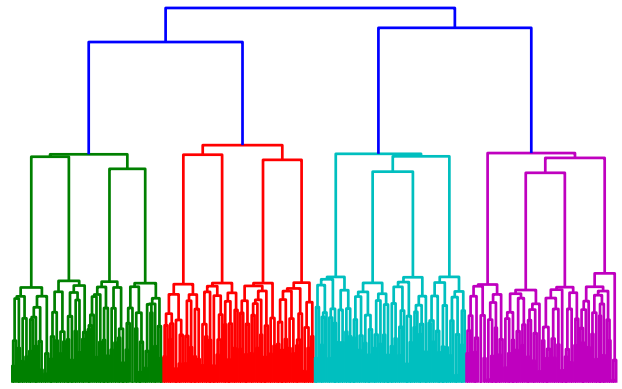
Outline

1. Node pair sampling
2. Flat clustering
3. Hierarchical clustering
4. **Quality metric**
5. Experiments

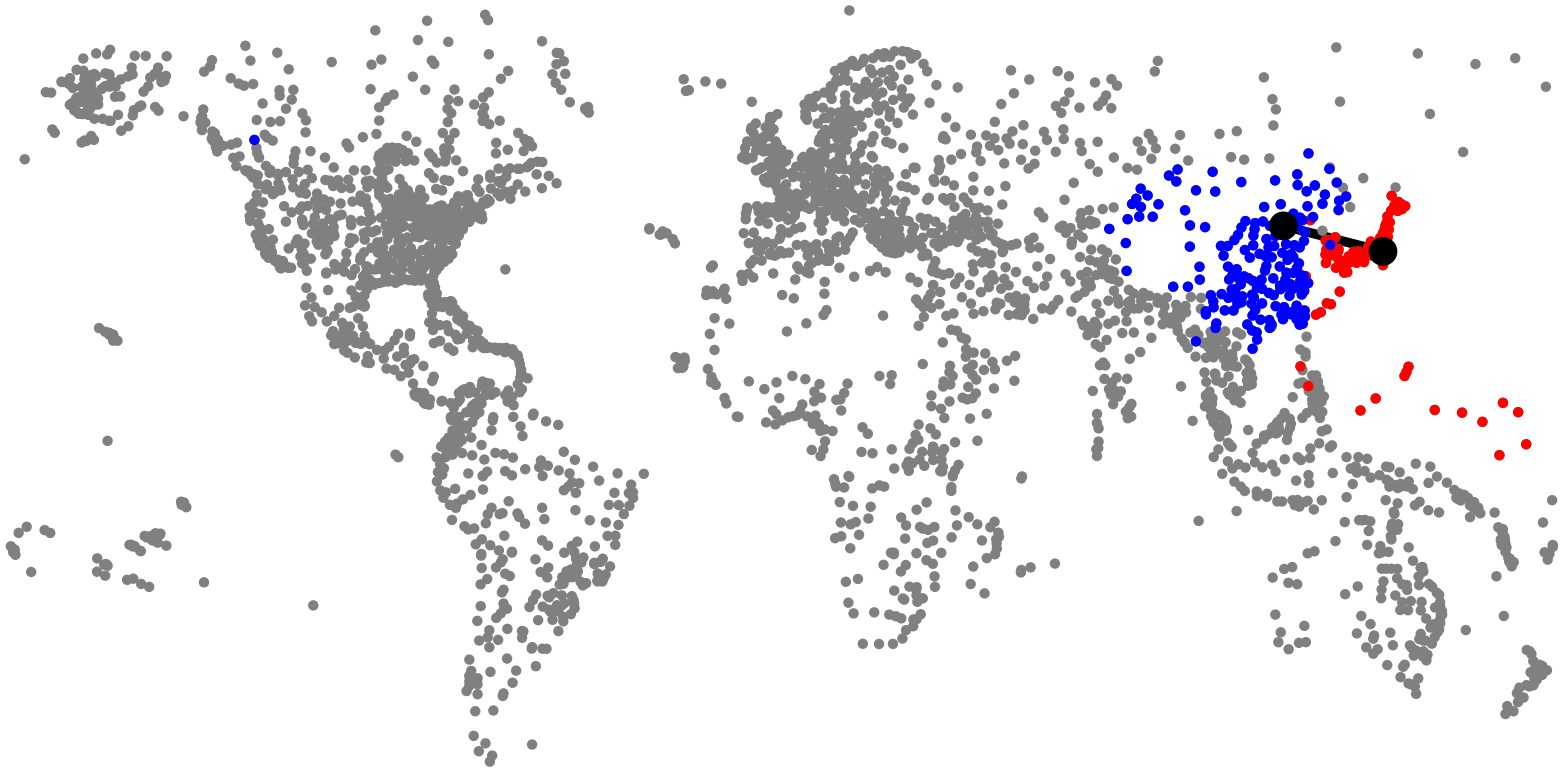
Intuition

- Two nodes sampled from the edges are expected to have a **common ancestor** relatively low in the hierarchy
- This corresponds to the **smallest cluster** of the hierarchy containing these two nodes

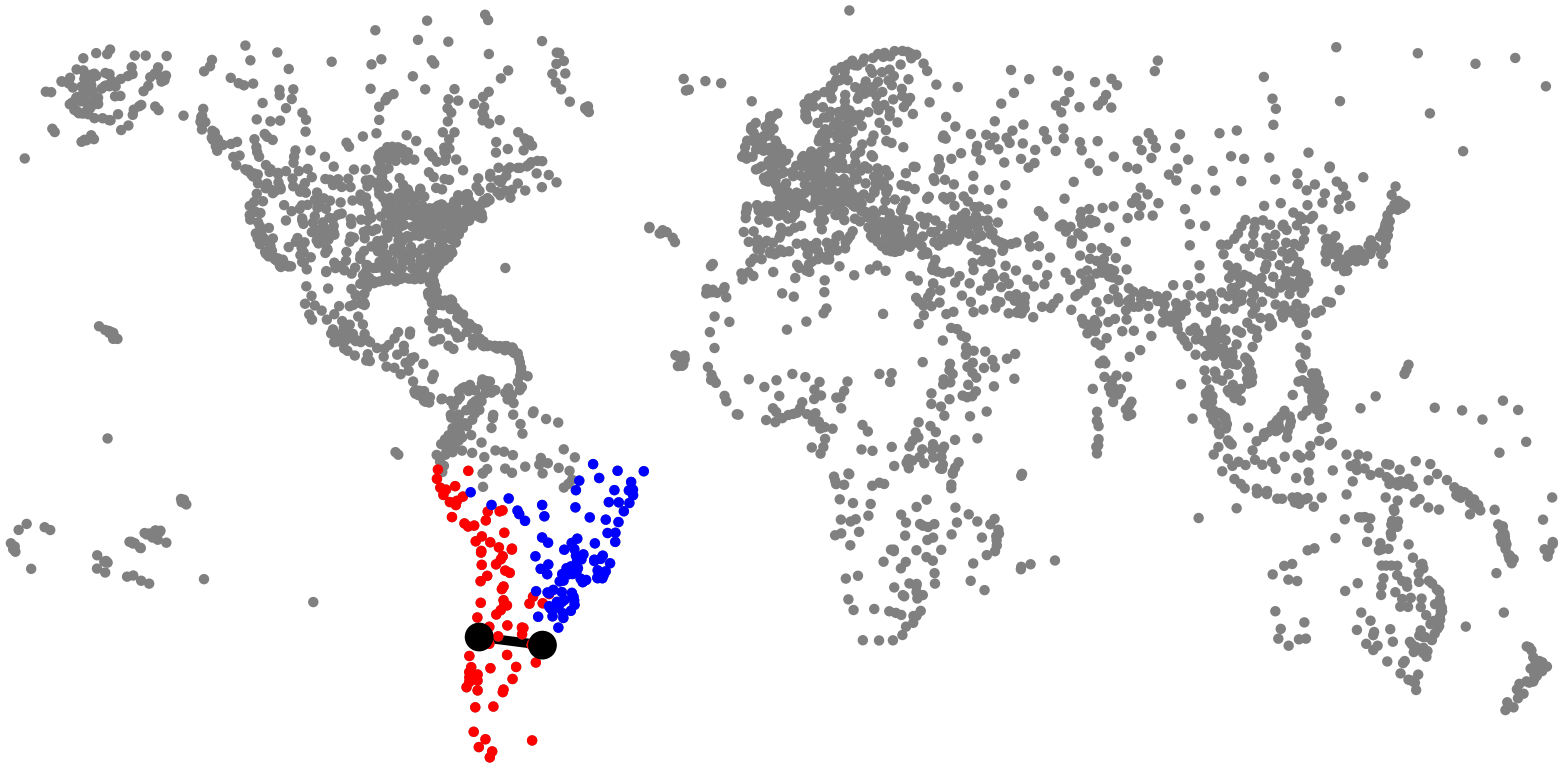




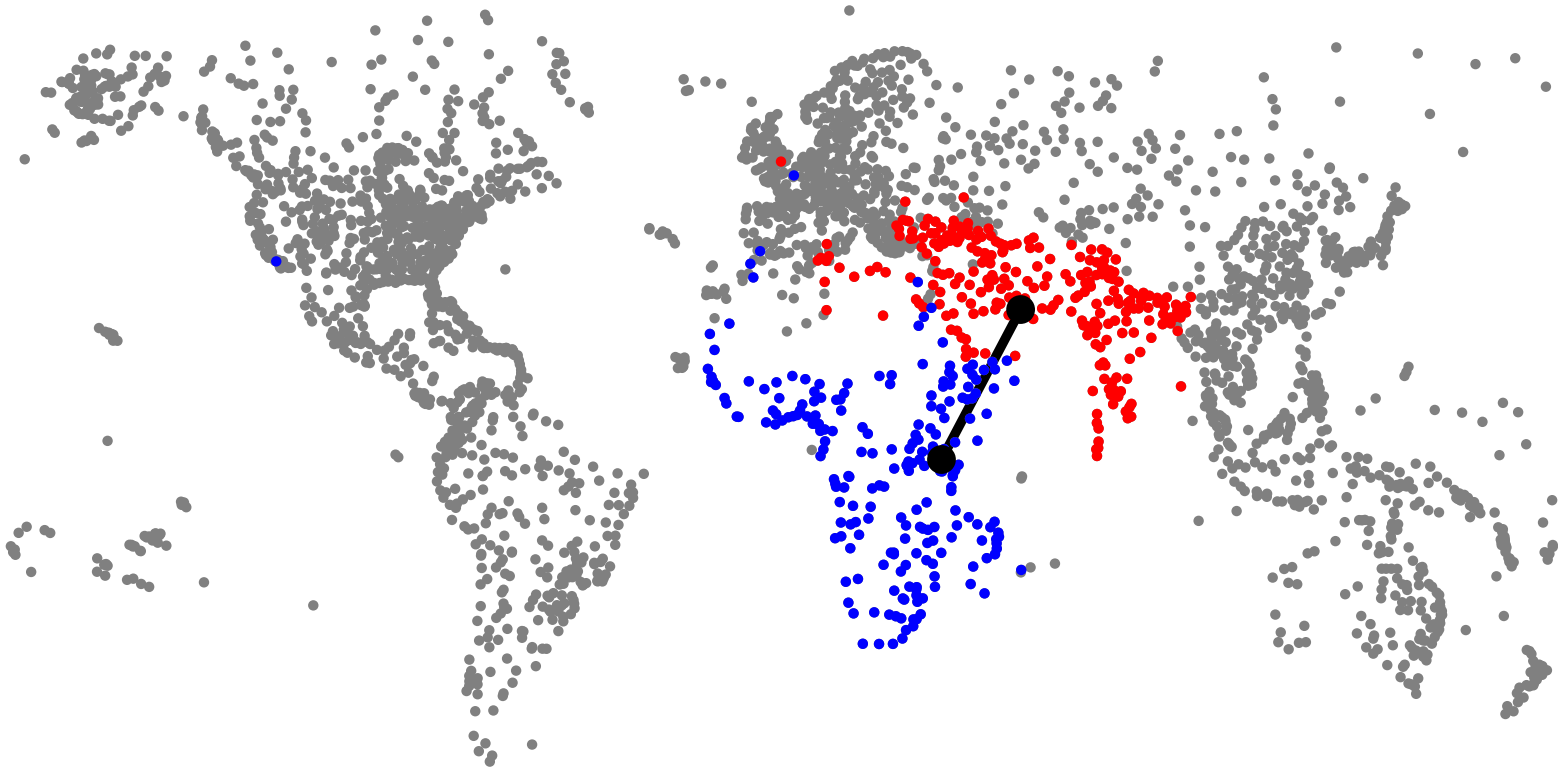
Example on Openflights



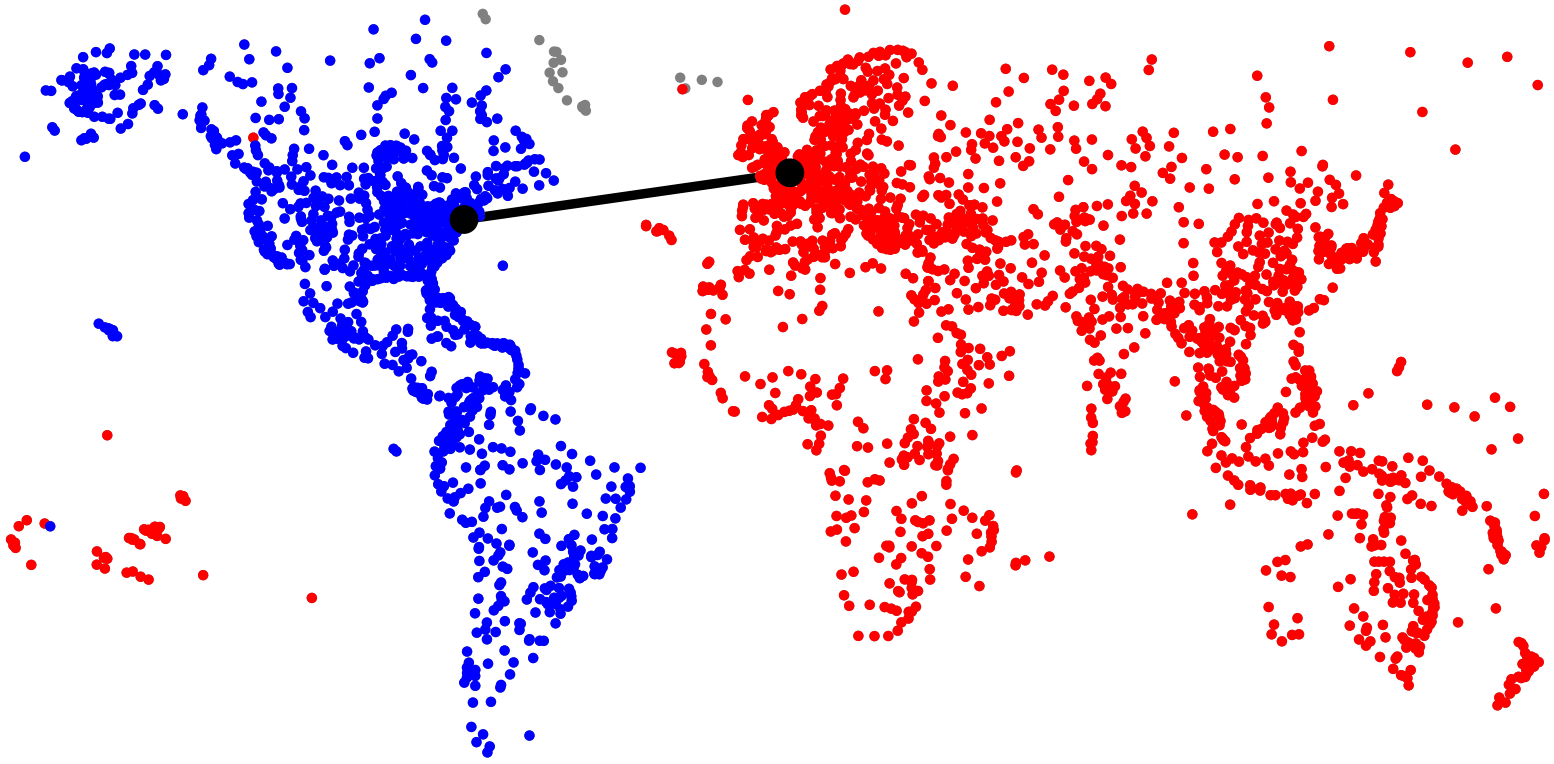
Example on Openflights



Example on Openflights



Example on Openflights

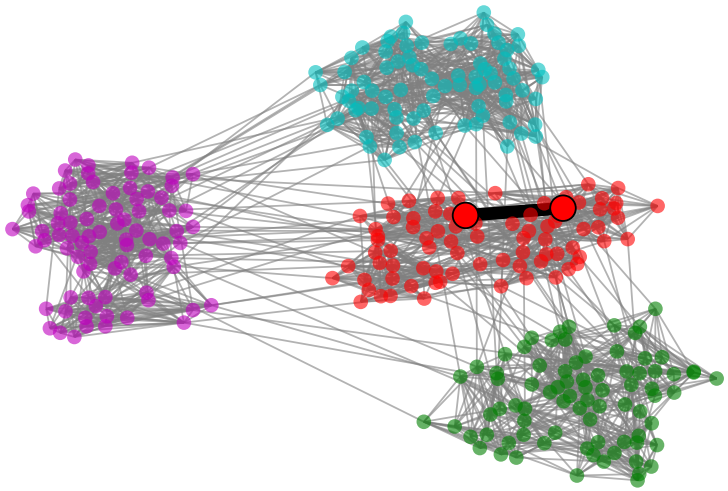


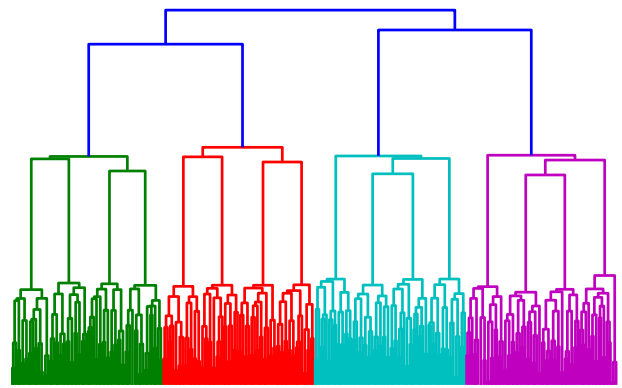
Tree sampling

- Let T be any rooted binary tree with leaves $\{1, \ldots, n\}$
- For any node $x \in T$,

$$p(x) = \sum_{i,j:i \wedge j = x} p(i,j)$$

- We denote by $c(x)$ the corresponding cluster



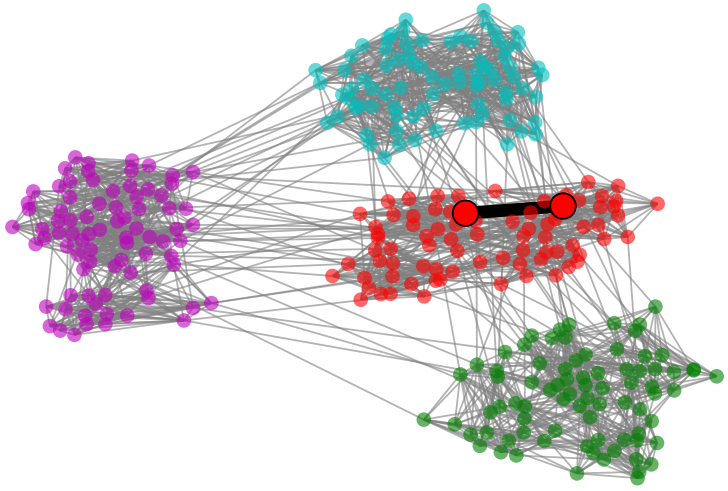


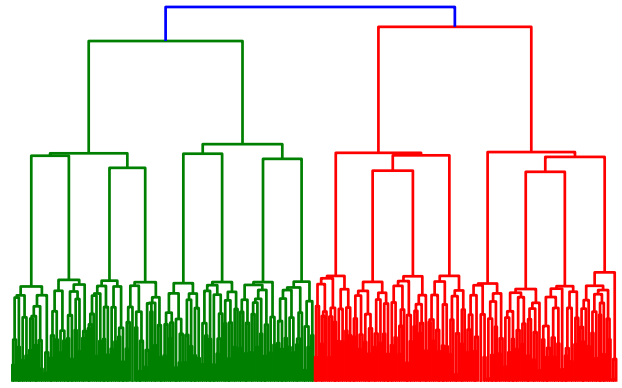
Dasgupta's cost

- Average cluster size:

$$\sum_{x \in T} p(x)|c(x)|$$

Dasgupta 2016
Cohen-Addad et. al. 2017

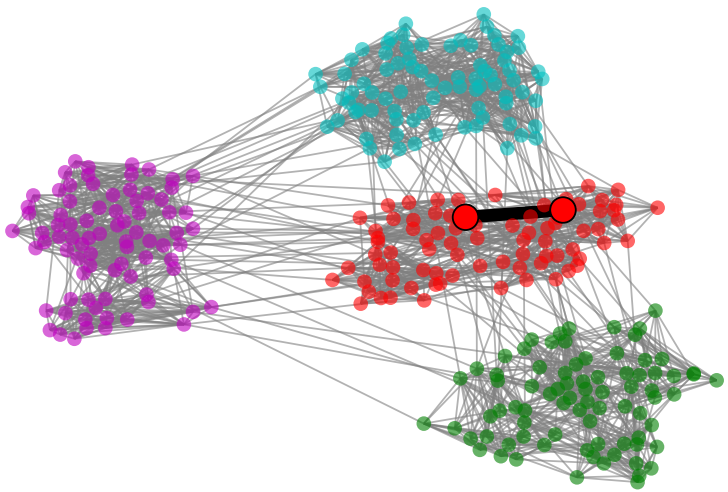


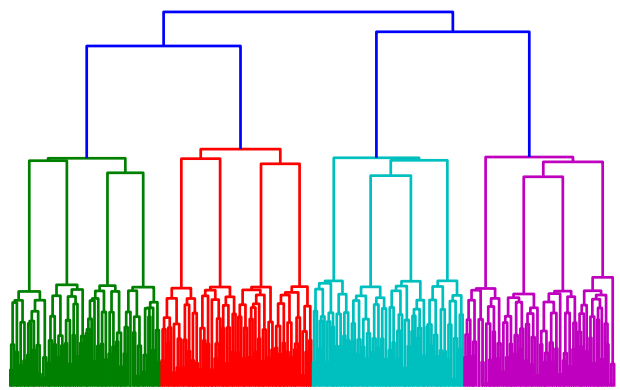


Back to tree sampling

- Let T be any rooted binary tree with leaves $\{1, \ldots, n\}$
- For any node $x \in T$,

$$p(x) = \sum_{i,j:i \wedge j = x} p(i,j) \quad q(x) = \sum_{i,j:i \wedge j = x} p(i)p(j)$$

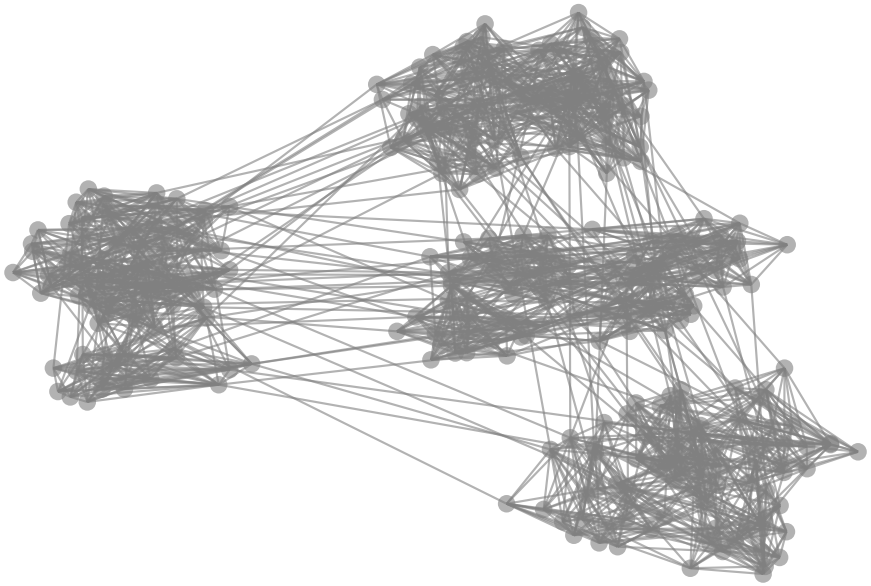


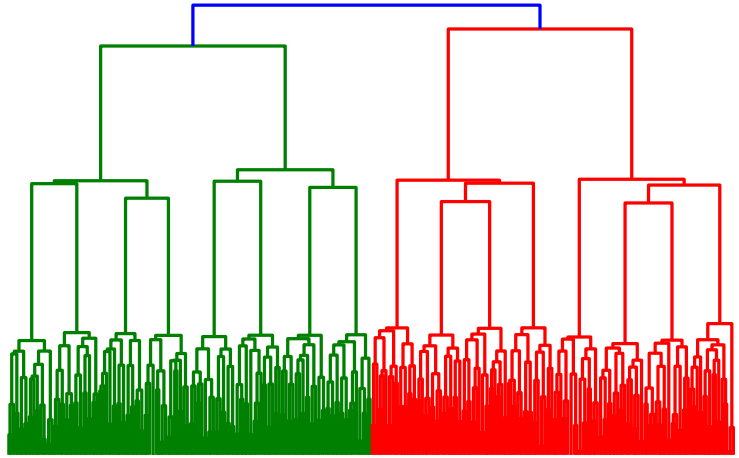


Tree sampling divergence

- Kullback-Leibler divergence between sampling distributions:

$$Q(T) = \sum_{x \in T} p(x) \log \frac{p(x)}{q(x)}$$



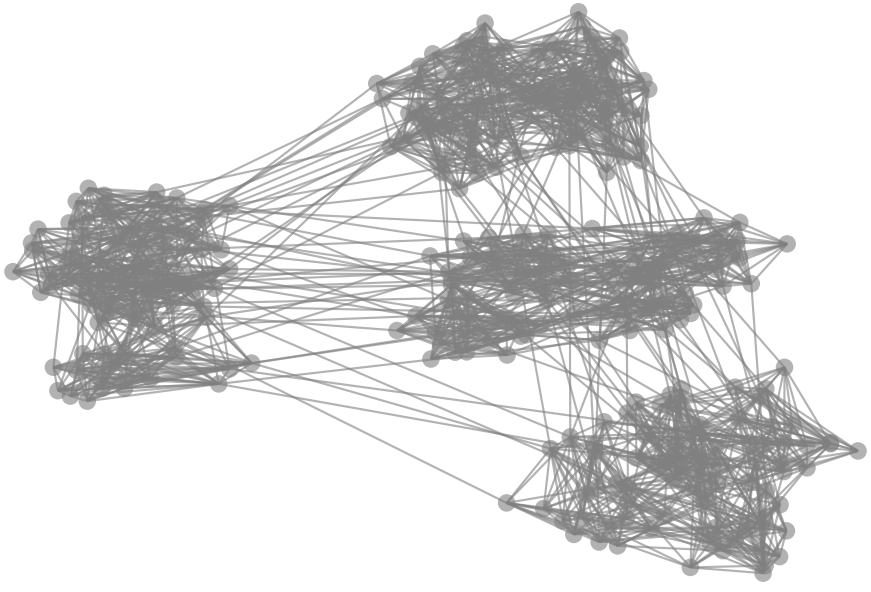


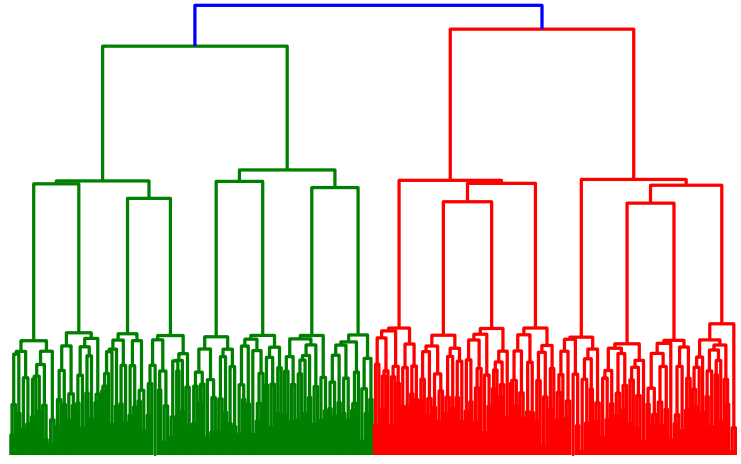
Tree sampling divergence

- Kullback-Leibler divergence between sampling distributions:

$$Q(T) = \sum_{x \in T} p(x) \log \frac{p(x)}{q(x)}$$

- Interpretable in terms of graph **reconstruction**!





Graph reconstruction

Given a tree T and the node weights $w_1, \ldots, w_n$, what is the best reconstruction of the graph (say $\hat{G}$)?

- Build the graph $\hat{G}$ with weights:

$$\hat{w}_{ij} \propto w_i w_j \hat{\sigma}(x)$$

  where $\hat{\sigma}(x)$ is some similarity attached to $x = i \wedge j$

- Apply the loss function:

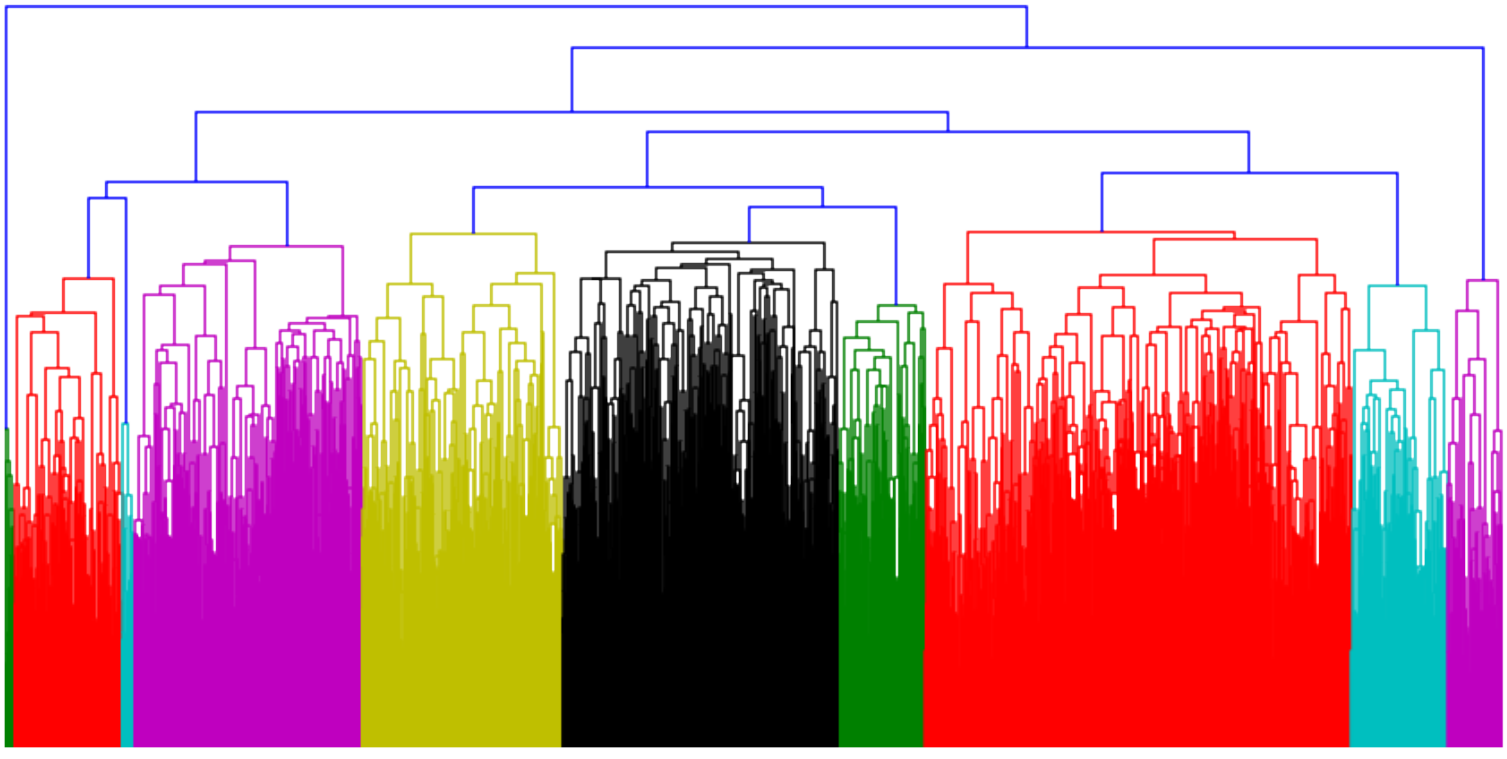
$$D(p||\hat{p}) = \sum_{i,j} p(i,j) \log \frac{p(i,j)}{\hat{p}(i,j)}$$

Main result

$$\min_{\hat{p} \leftarrow T} D(p||\hat{p}) = I - Q(T)$$

Hierarchical clustering of Openflights

3,097 airports, 18,193 flights

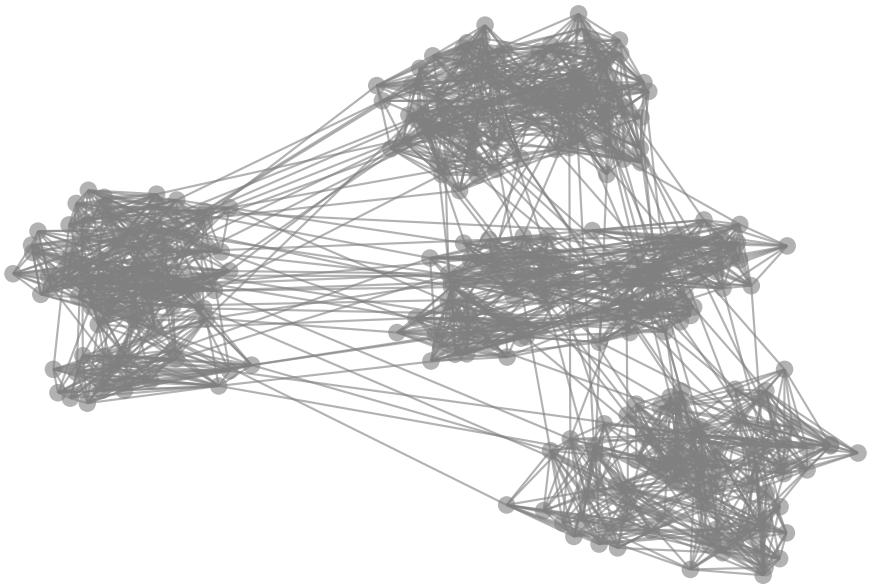


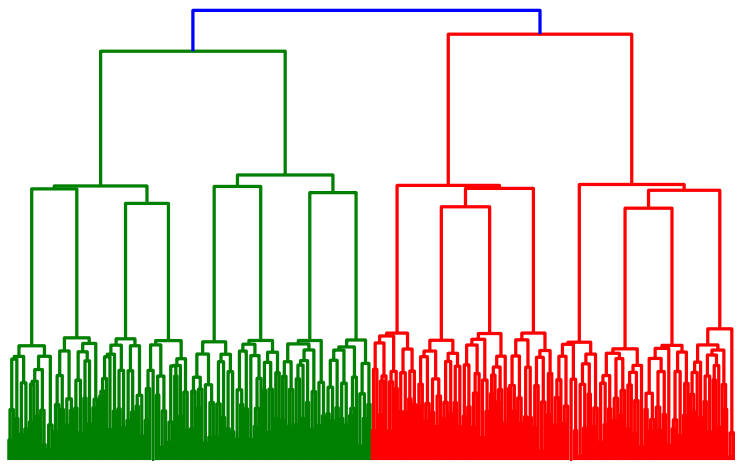
$H \approx 15$ bits $\quad I \approx 4$ bits $\quad Q \approx 2.6$ bits $\quad \bar{Q} = \dfrac{Q}{I} \approx 0.65$

General trees

The tree sampling divergence is applicable to **any** tree T:

$$Q(T) = \sum_{x \in T} p(x) \log \frac{p(x)}{q(x)}$$





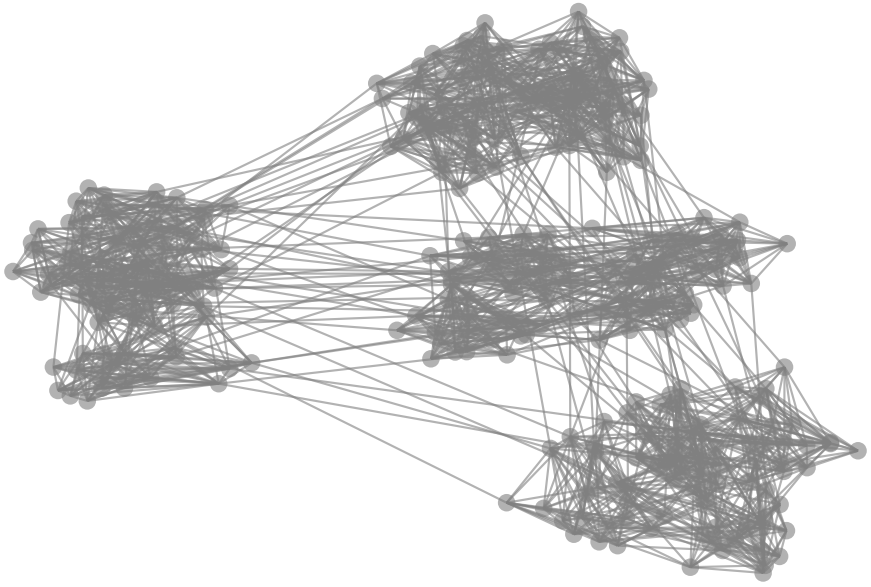
General trees

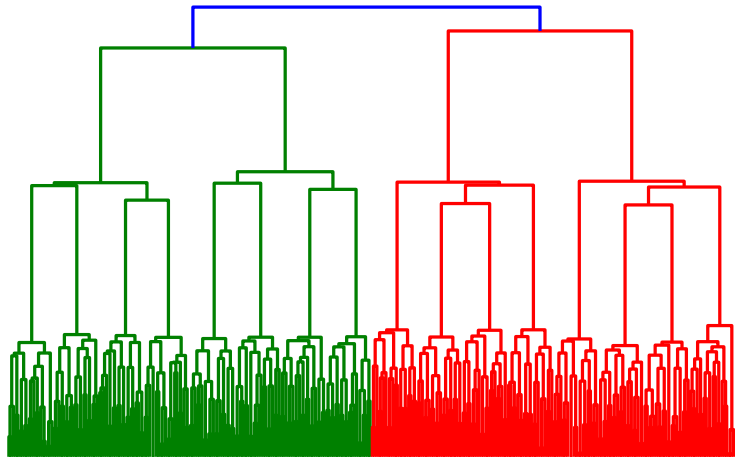
The tree sampling divergence is applicable to **any** tree T:

$$Q(T) = \sum_{x \in T} p(x) \log \frac{p(x)}{q(x)}$$

In particular, it can be used for:

- Flat clustering (trees of height 2)
- Tree compression

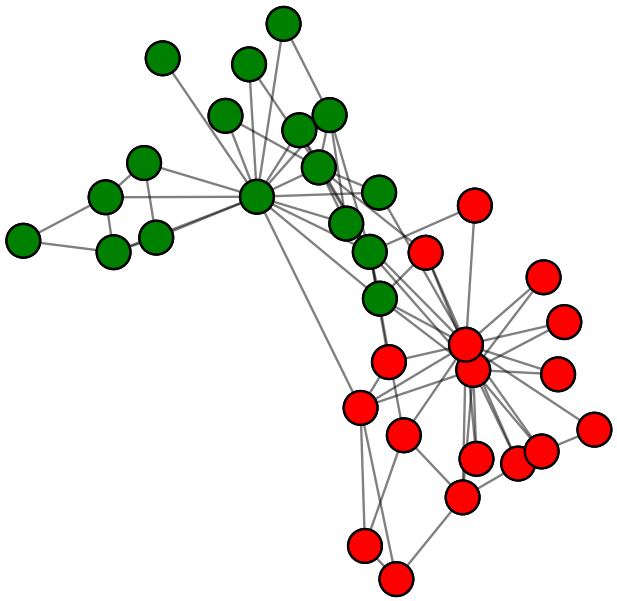


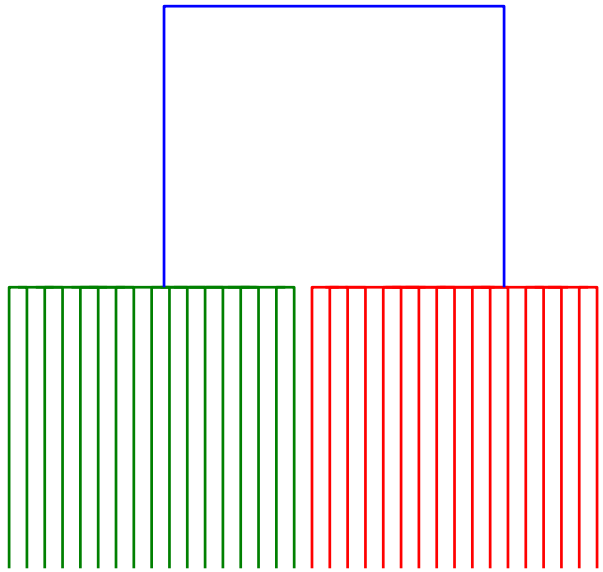


Flat clustering

For any clustering $C \in P(\{1, \ldots, n\})$:

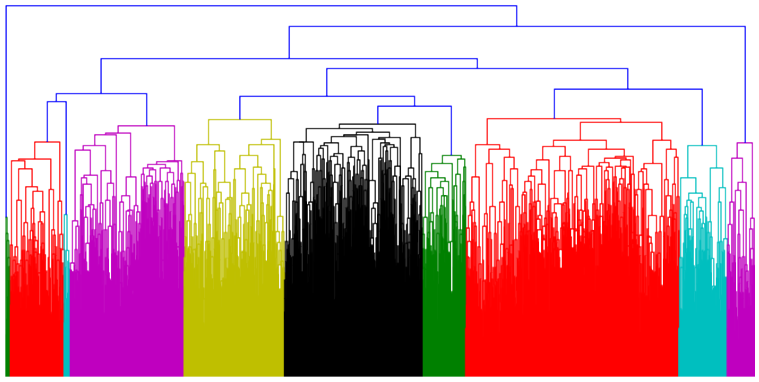
$$Q(C) = \sum_{c \in C} p(c,c) \log \frac{p(c,c)}{p(c)^2} + \left(1 - \sum_{c \in C} p(c,c)\right) \log \frac{1 - \sum_c p(c,c)}{1 - \sum_c p(c)^2}$$



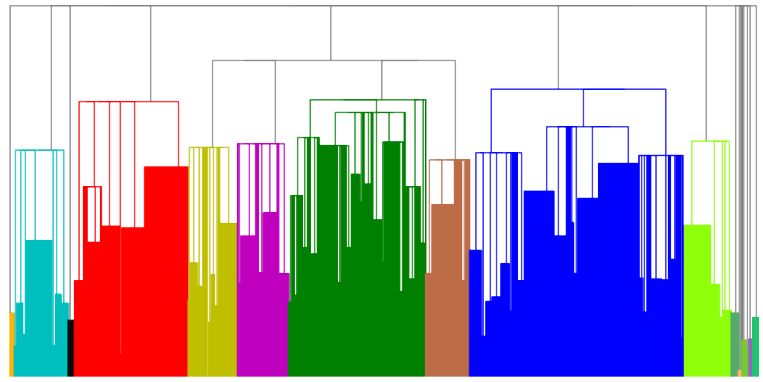


Tree compression of Openflights

3,097 airports, 18,193 flights

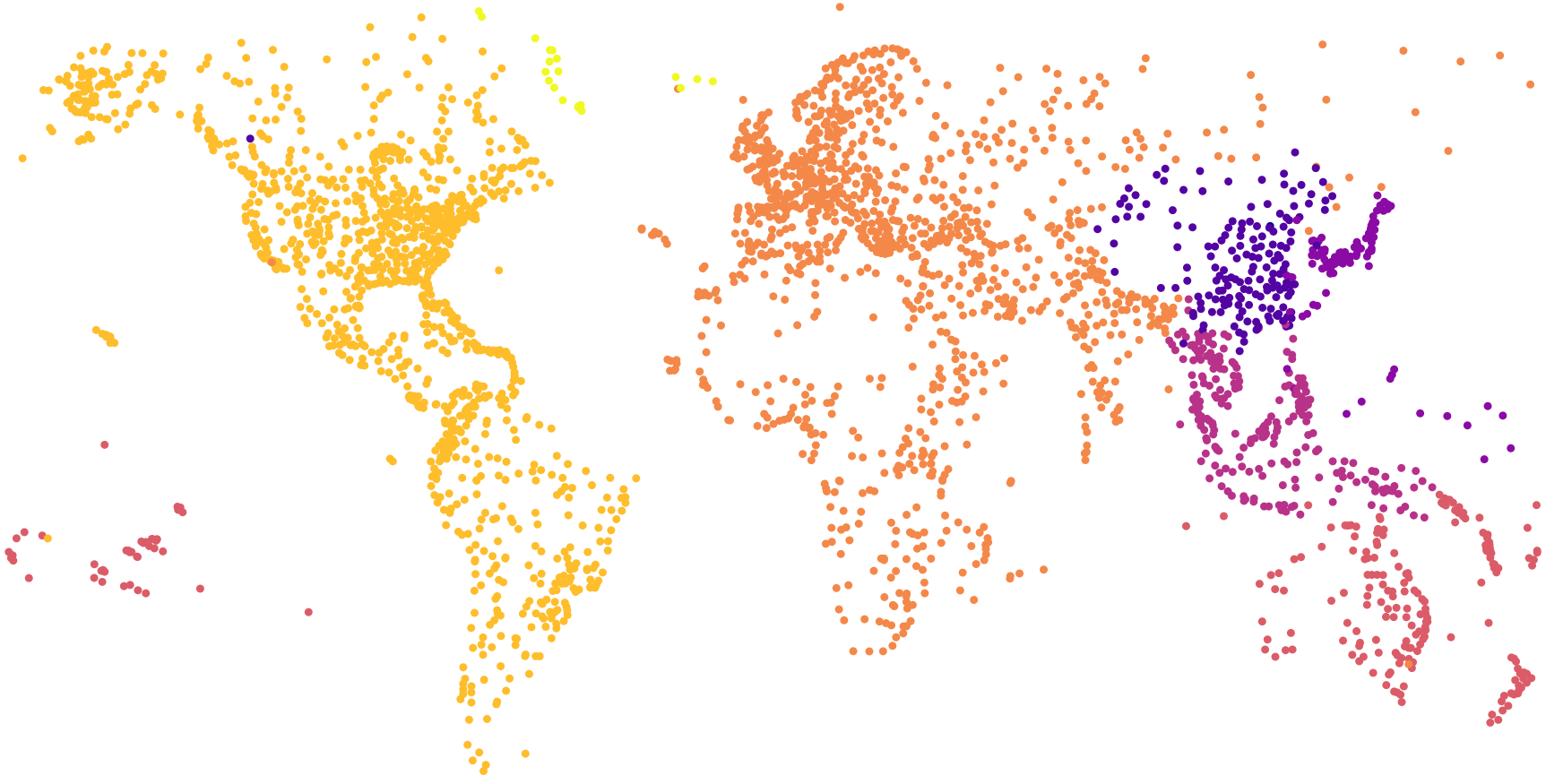


Full hierarchy (3097 levels)

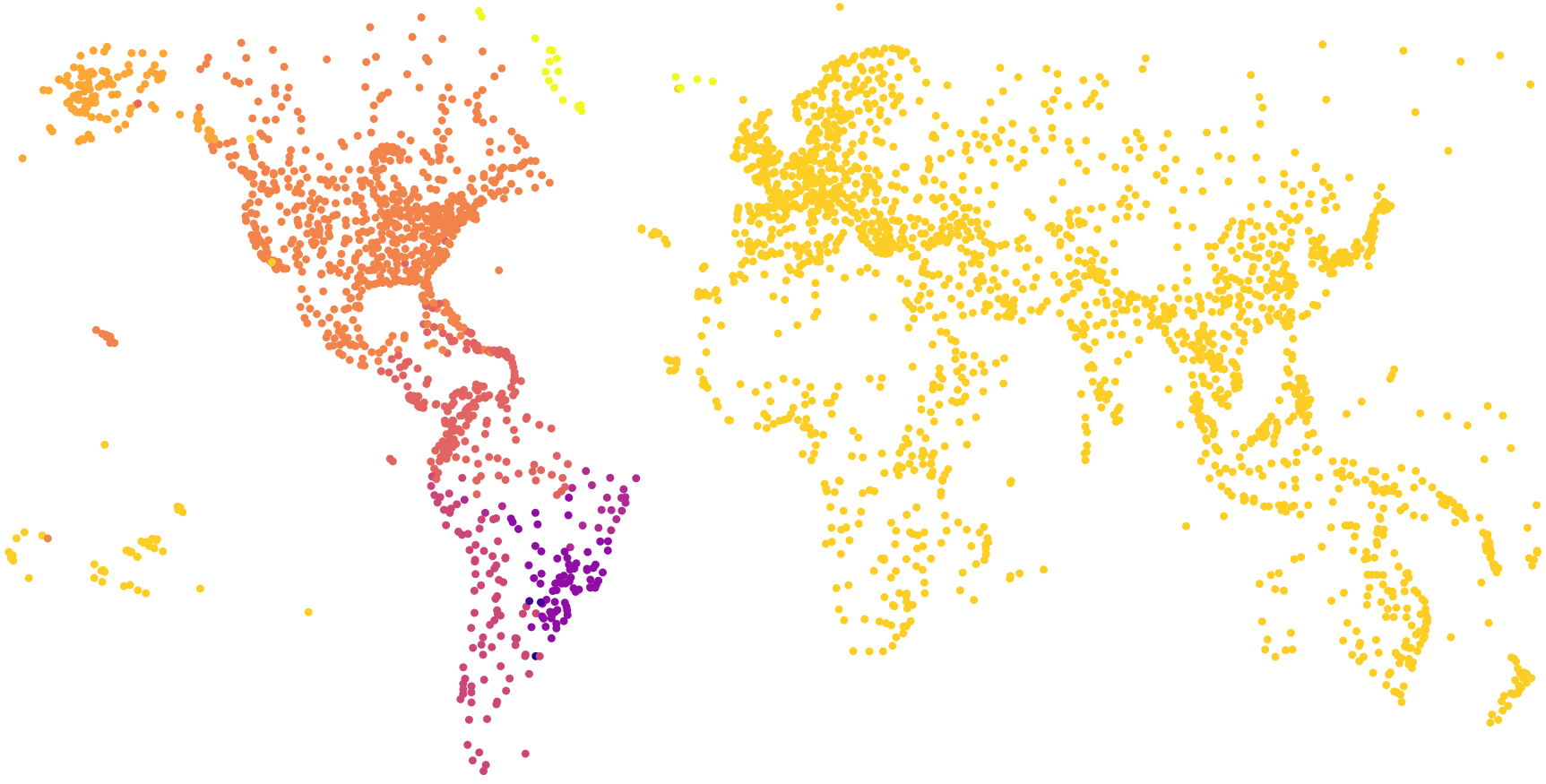


Compact hierarchy (97 levels)

Local hierarchy: Beijing Capital International Airport



Local hierarchy: Carrasco International Airport



Outline

1. Node pair sampling
2. Flat clustering
3. Hierarchical clustering
4. Quality metric
5. **Experiments**

Experiments

Remind the two metrics:

- Dasgupta's cost

$$\sum_{x \in T} p(x)|c(x)|$$

- Tree sampling divergence

$$\sum_{x \in T} p(x) \log \frac{p(x)}{q(x)}$$

Comparison of these metrics on two tasks:

1. Tree detection
2. Graph reconstruction

Tree detection

Idea:

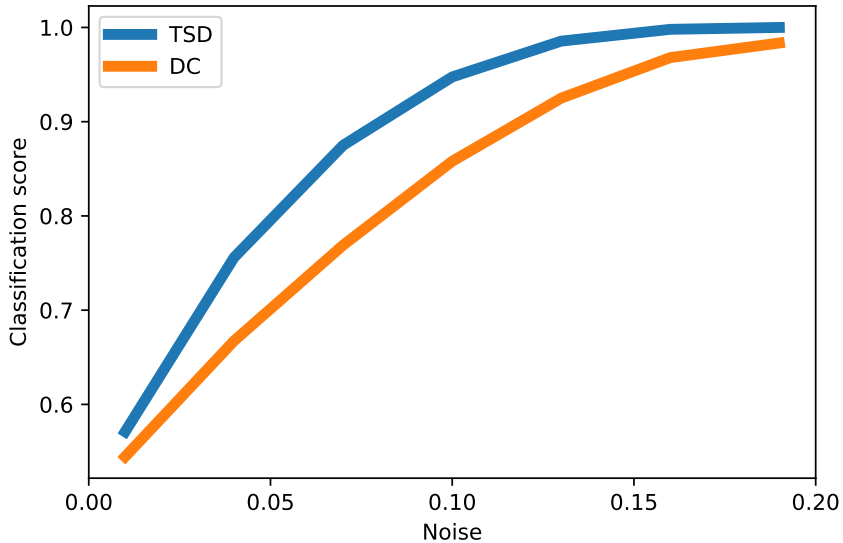
- Generate two noisy versions G_1, G_2 of some graph G
- Compute the corresponding trees T_1, T_2
- Guess the tree associated with each graph G_1, G_2

$$\hat{T}_1 = \arg \max_{T = T_1, T_2} Q_1(T) \quad \hat{T}_2 = \arg \max_{T = T_1, T_2} Q_2(T)$$

The **score** is the fraction of correct answers:

$$\frac{1}{2}(P(\hat{T}_1 = T_1) + P(\hat{T}_2 = T_2))$$

Results



Graph G = HSBM

16 blocks of size 20, 2 levels of hierarchy

Graph reconstruction

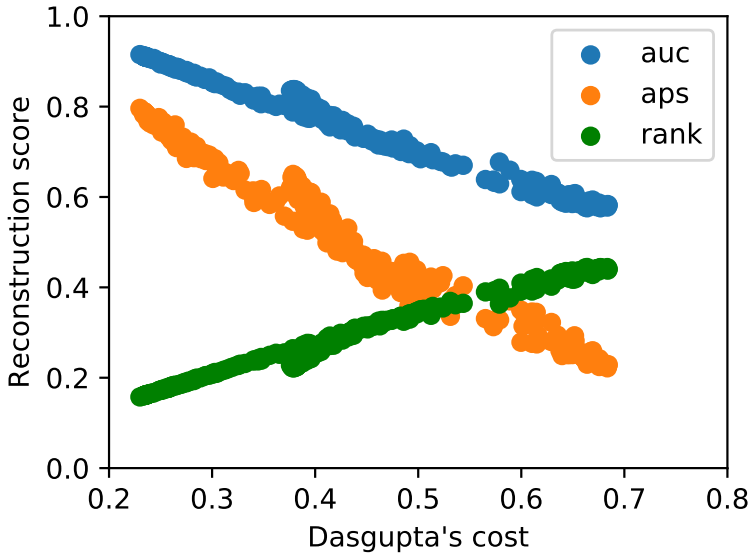
Idea:

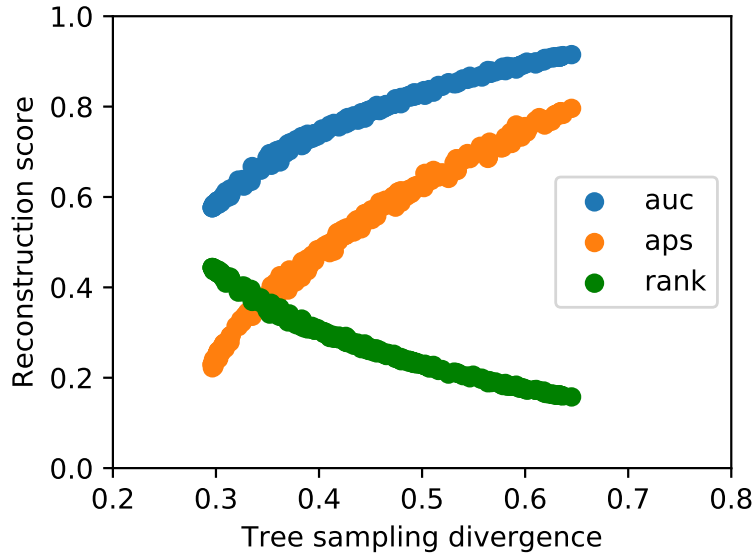
- Generate some hierarchical random graph G
- Compute trees with different linkages
- For each tree, reconstruct the graph $\hat{G}$
- Compare the **quality** of the tree and the **reconstruction scores**

Reconstruction scores:
Streaming the edges of $\hat{G}$ in decreasing order of weights,

- Area-Under-ROC
- Average-Precision-Score
- Average rank of each edge of G

Results


1.0
0.8
0.6
0.4
0.2
0.0
Reconstruction score
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Dasgupta's cost
auc
aps
rank


1.0
0.8
0.6
0.4
0.2
0.0
Reconstruction score
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Tree sampling divergence
auc
aps
rank

Summary

Viewing graphs as **probability** measures:

- A novel agglomerative **algorithm**, based on the linkage:

$$\sigma(i,j) = \frac{p(i,j)}{p(i)p(j)}$$

- A novel **quality** metric, the Tree Sampling Divergence:

$$\sum_{x \in T} p(x) \log \frac{p(x)}{q(x)}$$

  interpretable in terms of **graph reconstruction**

Ongoing work on:

- TSD for **flat clustering**
- **Fast** hierarchical clustering

scikit-network

A Python package under development, inspired by scikit-learn:
`https://github.com/sknetwork-team/scikit-network`

```
In [ ]: from sknetwork import clustering
In [ ]: louvain = clustering.Louvain(resolution = 4)
In [ ]: louvain.fit(X)
```