Graph Learning
4. Hierarchical Clustering

Thomas Bonald
Institut Polytechnique de Paris

2024 – 2025

These lecture notes introduce the notion of hierarchical graph clustering, useful for capturing the multi-scale nature of real graphs. We refer the reader to [1, 2] for more details on this topic.

1 Notion of hierarchical clustering

Consider an undirected graph $G = (V, E)$ of n nodes and m edges, with $V = \{1, \dots, n\}$. We denote by A the adjacency matrix and by $d = A1$ the vector of degrees. The volume of the graph is defined by:

$$v = \sum_{i \in V} d_i = \sum_{i,j \in V} A_{ij}.$$

We seek to represent the graph as a binary tree whose leaves are the nodes of the graph, as illustrated by Figure 1. Observe that this hierarchical representation of the graph reveals subclusters within the two top clusters.

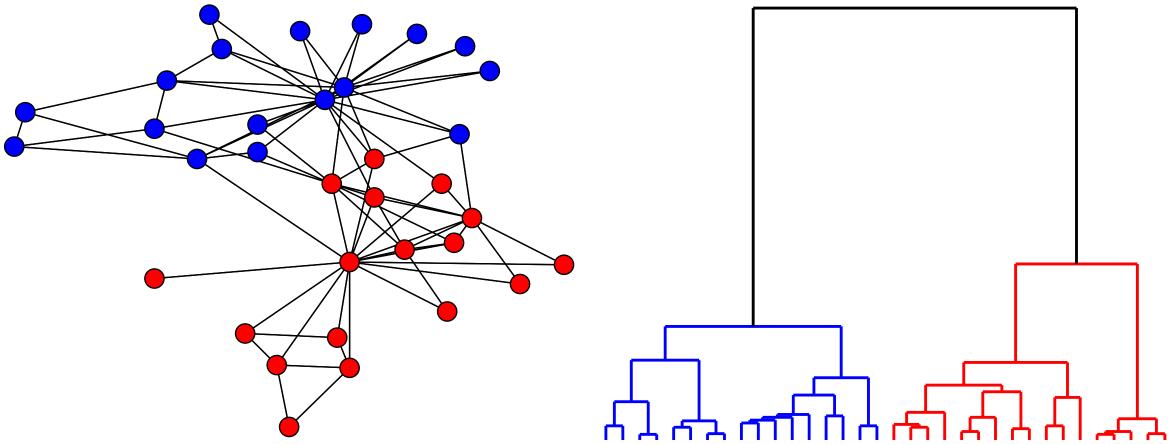

Figure 1: The Karate Club graph and its representation as a tree.

2 Sampling

Under edge sampling, each node pair i, j (in this order) is sampled with probability:

$$p(i,j) = \frac{A_{ij}}{v}.$$

This is a symmetric joint distribution with marginal distribution:

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

We deduce the probability of sampling node j given the sampling of node i:

$$p(j|i) = \frac{p(i,j)}{p(i)} = \frac{A_{ij}}{d_i}.$$

Similarly,

$$p(i|j) = \frac{p(i,j)}{p(j)} = \frac{A_{ij}}{d_j}.$$

3 Paris algorithm

The most efficient algorithms for hierarchical clustering are agglomerative: they consists in successively merge the two nodes with the strongest link, in a certain sense. A natural notion of link "strength" follows from sampling. We say that the link between nodes i and j is strong if the probability of sampling node j *given* the sampling of node i is much higher than the probability of sampling node j. We get the following definition of link strength between nodes i and j:

$$\sigma(i,j) = \frac{p(j|i)}{p(j)} = \frac{p(i,j)}{p(i)p(j)} = v\frac{A_{ij}}{d_i d_j}. \tag{1}$$

Observe that this definition is symmetric in i, j so that

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

We get the following algorithm[1], each merge generating a new "node" that takes the first available index:

```
Input: Graph $G = (V, E)$
Output: List of merges, $L$
for $t = 1, \dots, n-1$ do
    $i, j \leftarrow \arg\max_{i,j \in V, i \neq j} \sigma(i,j)$
    append $i, j$ to $L$
    merge $i, j$ into node $n + t$
    update $\sigma$
```

Algorithm 1: Paris algorithm.

Paris algorithm

Paris is a hierarchical clustering algorithm that successively merges the two nodes with the strongest link in terms of (1).

[1] Paris = Pairwise AgglomeRation Induced by Sampling

For convenience, we denote by $i \cup j$ the node resulting from the merge of nodes i and j (indexed by $n+t$ in the algorithm). After the merge, the sampling distribution becomes:

$$p(i \cup j, k) = p(i,k) + p(j,k), \quad \forall k \in V \setminus \{i,j\},$$

and

$$p(i \cup j) = p(i) + p(j).$$

We deduce that the new link strengths are the weighted averages of previous link strengths:

Proposition 1 (Update formula)

$$\forall k \neq i,j, \quad \sigma(i \cup j, k) = \frac{p(i)}{p(i)+p(j)}\sigma(i,k) + \frac{p(j)}{p(i)+p(j)}\sigma(j,k).$$

4 Notion of dendrogram

The merge of any two nodes i,j is natural if the link strength $\sigma(i,j)$ is high, equivalently if the "distance" between these two nodes $d(i,j) = \sigma(i,j)^{-1}$ is low. It is convenient in practice to represent the hierarchical clustering of a graph not only through the successive merges (the binary tree) but through the successive distances of these merges (the height of each branching point of the binary tree). This is illustrated by Figure 1, where several levels of hierarchy appear clearly.

The distance associated with link strength (1) is:

$$d(i,j) = \frac{d_i d_j}{v A_{ij}}.$$

Observe that the distance between two nodes is infinite in the absence of edge between these nodes. In particular, if the graph has K connected components, then the last $K-1$ merges are at infinite distance.

Proposition 2 (Reducible distance) *We have:*

$$\forall k \neq i,j, \quad d(i \cup j, k) \geq \min(d(i,k), d(j,k)).$$

Proof. In view of Proposition 1, we have $\sigma(i \cup j, k) \leq \max(\sigma(i,k), \sigma(j,k))$. Taking the inverse gives the desired result. □

A consequence of Proposition 2 is that the sequence of distances resulting for the agglomerative algorithm is non-decreasing:

$$\forall k \neq i,j, \quad d(i \cup j, k) \geq \min(d(i,k), d(j,k)) \geq d(i,j),$$

where the last inequality follows from the fact that the distance is minimized for the node pair i,j. In particular, there is no inversion in the dendrogram.

5 The nearest-neighbor chain

Finding the two closest nodes at each step of the algorithm requires $O(m)$ operations, hence an overall complexity in $O(nm)$. Fortunately, it is possible to reduce the complexity of the algorithm on observing that any nodes i,j that are nearest from each other, in the sense that

$$d(i,j) = \min_{k \neq i,j} d(i,k) = \min_{k \neq i,j} d(j,k), \tag{2}$$

can be merged. In particular, it is not necessary to merge those nodes i,j that attain the global minimum of $d(i,j)$; a *local* minimum is sufficient. Indeed, any nodes i,j that satisfy (2) can be merged at any step of the algorithm, because, in view of Proposition 2, any other merge will not change the fact that i, j are nearest from each other.

An efficient way to find two nodes that are nearest from each other is to build the *nearest-neighbor chain* [2]. The chain starts from an arbitrary node i_0. Then the second element of the chain is the nearest neighbor i_1 of i_0; the third element of the chain is the nearest neighbor i_2 of i_1; if $i_2 = i_1$, the chain is complete and i_1, i_2 are nearest neighbors from each other; otherwise, a fourth element is added to the chain, and so on until two nodes are nearest neighbors. The algorithm stops provided ties are broken with a fixed, pre-defined rule[2].

Nearest-neighbor chain

The nearest-neighbor chain is an agglomerative algorithm that relies on the search of local minima; the resulting dendrogram is the same as that returned by an agglomerative algorithm based on the search of a global minimum.

The algorithm consists in merging recursively nodes appearing at the end of the chain. When the chain is exhausted, a new chain is started whenever there are at least two nodes left. After each merge, the distances are updated using the formulas of Proposition 1.

```
Input: $i_0$ (initial node)
Output: $L$, list of merges
S ← empty stack
S.push($i_0$)
while S is not empty do
    $i$ ← S.pop
    $j$ ← nearest neighbor of $i$
    if $j$ is in S then
        $j$ ← S.pop
        append $i, j$ to $L$
        merge $i, j$
    else
        S.push($i$)
        S.push($j$)
```

Algorithm 2: Nearest-neighbor chain

6 Link with modularity

The proximity metric (1) used in the agglomerative algorithm can be interpreted in terms of resolution. Recall that the modularity of clustering C at resolution γ is given by:

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

For $\gamma \to 0$, the fit term dominates and the maximum is achieved with a single cluster; for $\gamma \to +\infty$, the diverstiy term dominates and the maximum is achieved with n clusters (one per node). Starting from the

[2] For instance, if $d(i, j_1) = d(i, j_2)$ then decide that j_1 is nearest from i than j_2 if and only if $j_1 < j_2$. Then the chain cannot form any triangle, which guarantees that the algorithm stops in finite time and outputs two nearest neighbors.

trivial clustering where each node is in its own cluster, the increase in modularity on merging nodes i, j is:

$$\Delta Q_\gamma = 2(\frac{A_{ij}}{v} - \frac{\gamma d_i d_j}{v^2}) = \frac{2}{v}\left(A_{ij} - \gamma \frac{d_i d_j}{v}\right),$$

which is positive whenever:

$$\sigma(i,j) > \gamma.$$

We deduce that the maximum value of resolution below which there is at least one cluster with 2 nodes is:

$$\gamma_1 = \max_{i \neq j} \sigma(i,j).$$

Below this resolution parameter, nodes i, j achieving this maximum must be merged. After this merge, the next value of the resolution parameter, say γ_2, below which two nodes must be merged is the maximum of the link strengths $\sigma(i,j)$ in the aggregate graph, resulting from the first merge. By construction, we have $\gamma_2 \leq \gamma_1$. After this second merge, we look for the next value of the resolution parameter, say γ_3, below which two nodes must be merged, and so on. So the agglomerative algorithm based on the link strength σ can be seen as a greedy algorithm for maximizing modularity at the highest resolution. The resulting sequence of resolutions is non-increasing; their inverses, corresponding to the heights of the successive merging points in the dendrogram, is non-decreasing.

7 Extensions

Weighted graphs. The algorithm extends naturally to weighted graphs, with A the weighted adjacency matrix and $d = A1$ the vector of node weights.

Directed graphs. The extension to directed graphs is less obvious. Let $d^+ = A1$ and $d^- = A^T1$ be the vectors of out-degrees and in-degrees. We use the following definition of link strength:

$$\sigma(i,j) = v\frac{A_{ij} + A_{ji}}{d_i^+ d_j^- + d_j^+ d_i^-}.$$

Equivalently,

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

with p the (asymmetric) probability distribution under edge sampling:

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

and p^+, p^- the marginal distributions:

$$p^+(i) = \sum_{j \in V} p(i,j) = \frac{d_i^+}{v}, \quad p^-(i) = \sum_{j \in V} p(j,i) = \frac{d_i^-}{v}.$$

Observe that the probability distribution

$$\hat{p}(i,j) = p^+(i)p^-(j)$$

is that of edge sampling in the graph of adjacency matrix[3]:

$$\hat{A} = \frac{d^+ {d^-}^T}{v}.$$

[3] This is the reference graph used in the diversity term of modularity for directed graphs.

The algorithm is similar, with the update formula:

$$\forall k \neq i, j, \quad \sigma(i \cup j, k) = \frac{q(i,k)}{q(i,k) + q(j,k)}\sigma(i,k) + \frac{q(j,k)}{q(i,k) + q(j,k)}\sigma(j,k).$$

using the notation $q(i,j) = \hat{p}(i,j) + \hat{p}(j,i) = p^+(i)p^-(j) + p^+(j)p^-(i)$. Proposition 2 holds.

Bipartite graphs. The algorithms also apply to bipartite graphs, seen as undirected graphs. This technique provides a single hierarchical structure mixing nodes of both parts of the bipartite graph (this is hierarchical co-clustering). Two distinct hierarchical structures, one for each part, can be recovered on identifying the successive merges of nodes of each part of the graph in the original dendrogram.

References

[1] T. Bonald, B. Charpentier, A. Galland, and A. Hollocou. Hierarchical graph clustering based on node pair sampling. In *Proceedings of the 14th International Workshop on Mining and Learning with Graphs (MLG)*, 2018.

[2] F. Murtagh and P. Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2012.