Maximum Modularity

$$q^*(G) = \max\{ q_G(\mathcal{A}) : \underbrace{\mathcal{A} \text{ is a partition of } V(G)}_{\text{there are } B_n \approx \left(\frac{n}{e+o(\log n)}\right)^n \text{ of these!}} \}.$$

$q^*(G)$ is too hard to compute exactly, so algorithms try to approximate it using some local search.

Louvain Algorithm

Input: A partition $\mathcal{A} = \{A_1, \ldots, A_\ell\}$ of the vertices of a weighted graph G.

- ~~After each round~~, Do 1 round
- Make a contracted graph $G_{\mathcal{A}}$
  - Each A_i is a vertex.
  - self loop at A_i of weight $\sum_{e \in E(G[A_i])} \text{weight}(e)$
  - edge A_iA_j of weight

$$\sum_{v \in A_i} \sum_{w \in A_j} \underbrace{w(v,w)}_{=0 \text{ if } vw \notin E(G).}$$

Round:

Consider $v_1 \ldots v_n$ in random order.

for $p = 1$ to n.

- Suppose $v_p \in A_i$
- for each $j \neq i$ compute increase in $q(A)$ if we move v_p to A_j
- If at least one move increases $q(A)$ then make the move that maximizes increase.

If no move was done then stop.

Else make contracted graph G_A and repeat.

Louvain: Start with $A = \{\{v_1\}, \{v_2\}, \ldots, \{v_n\}\}$ repeatedly do 1 round and contract.

- Relatively efficient
- Seems to provide reasonable community structures.
- Sensitive to choice of permutation
  - Different runs can find quite different community structures.

- Lots of variants:

Problems with Modularity

- Resolution Limit
- Makes it impossible to detect small communities.

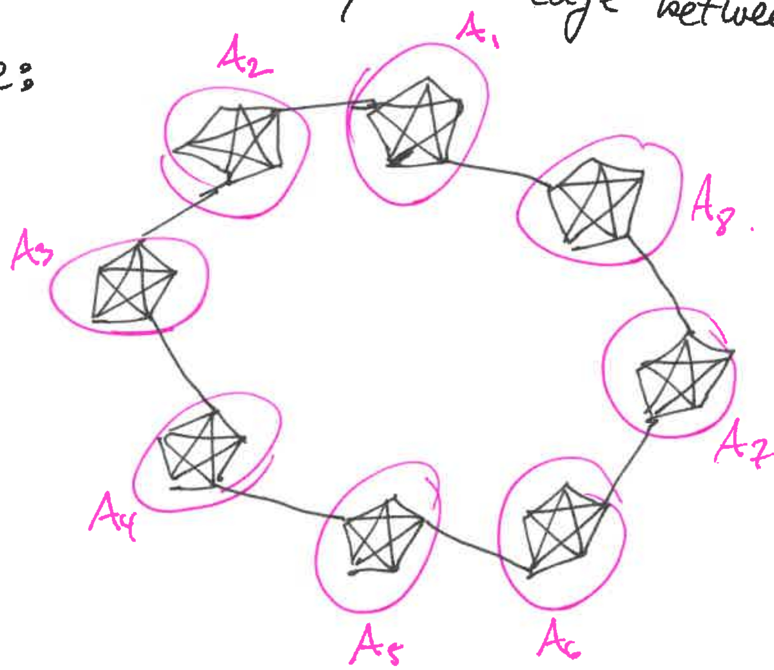
In the Chung-Lu Model, the expected number of edges between A_i and A_j is

$$\sum_{v \in A_i} \sum_{w \in A_j} \frac{\deg(v) \cdot \deg(w)}{2|E|} = \frac{\left(\sum_{v \in A_i} \deg(v)\right)\left(\sum_{w \in A_j} \deg(w)\right)}{2|E|} < 1$$

if $\left(\sum_{v \in A_i} \deg(v)\right) \leq \sum_{w \in A_j} \deg(w) < \sqrt{2|E|}$

In this case merging A_i and A_j will increase $q(A)$ even if there is only one edge between A_i and A_j

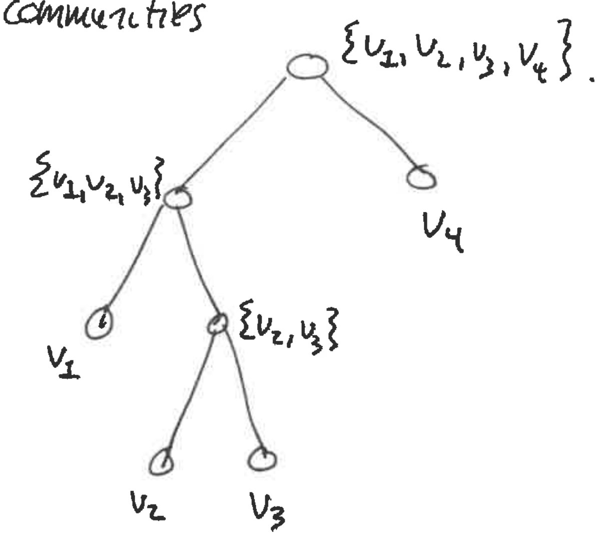
Example:



Merging A_i and A_{i+1} will increase $q(A)$.

Hierarchical Clustering

- Hierarchical clusterings provide a whole set of possible clusterings as a rooted ~~tree.~~ (usually binary) tree.
  - Leaves are vertices of G. (single ~~vertex~~ communities)
  - Each internal node represents a community obtained by merging the communities of its children.



Advantage: User gets to pick the level of detail.

Disadvantage: User has to pick the level of detail.

Ravasz (bottom up algorithm).

Assign score $s(v_i, v_j) = \dfrac{|N_G(v_i) \cap N_G(v_j)| + \mathbb{1}_{v_i v_j \in E(G)}}{\min\{|N_G(v_i)|, |N_G(v_j)|\} + \underbrace{\mathbb{1}_{v_i v_j \notin E(G)}}_{\text{Indicator variable.}}}$.

For two clusters, assign score

$$s(A_i, A_j) = \frac{1}{|A_i| \cdot |A_j|} \cdot \sum_{v \in A_i} \sum_{w \in A_j} s(v, w)$$

$$\mathbb{1}_{\text{Condition}} = \begin{cases} 1 & \text{if condition true.} \\ 0 & \text{if condition false.} \end{cases}$$

Start with trivial partition $A = \{\{v_1\}, \dots, \{v_n\}\}$.

Repeatedly choose A_i, A_j that maximizes $s(A_i, A_j)$ and merge A_i and A_j.