CỘNG ĐỒNG TRONG MẠNG XÃ HỘI

Biên soạn: **ThS. Nguyễn Thị Anh Thư**

Email: thunta@uit.edu.vn

NỘI DUNG

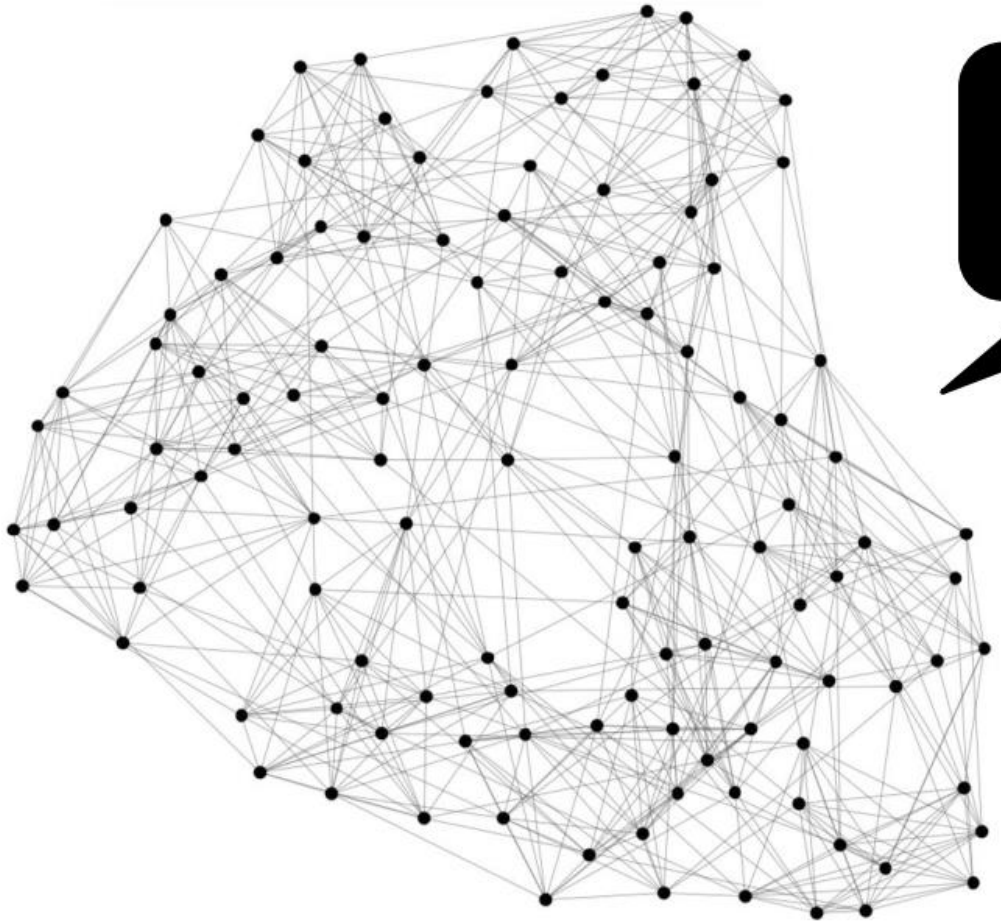
1. Tổng quan
2. Các loại cộng đồng
3. Phát hiện cộng đồng
4. Thuật toán Girvan Newman
5. Thuật toán Louvain

1. TỔNG QUAN

- **Đồ thị trong đời thực không phải là ngẫu nhiên.**
  - Ví dụ: Trong mạng xã hội, mọi người chọn bạn bè của họ dựa trên sở thích và hoạt động chung của họ.
- **Hy vọng rằng các nút trong biểu đồ sẽ được tổ chức theo các cộng đồng.**
  - Các nhóm đỉnh có thể chia sẻ các thuộc tính chung và / hoặc đóng các vai trò tương tự trong biểu đồ.
- **Làm thế nào để phát hiện cộng đồng?**
  - Các nút trong cùng một cộng đồng sẽ được kết nối chặc chẽ với nhau và kết nối thưa thớt với các cộng đồng khác.

1. TỔNG QUAN

- **NCAA Football network**

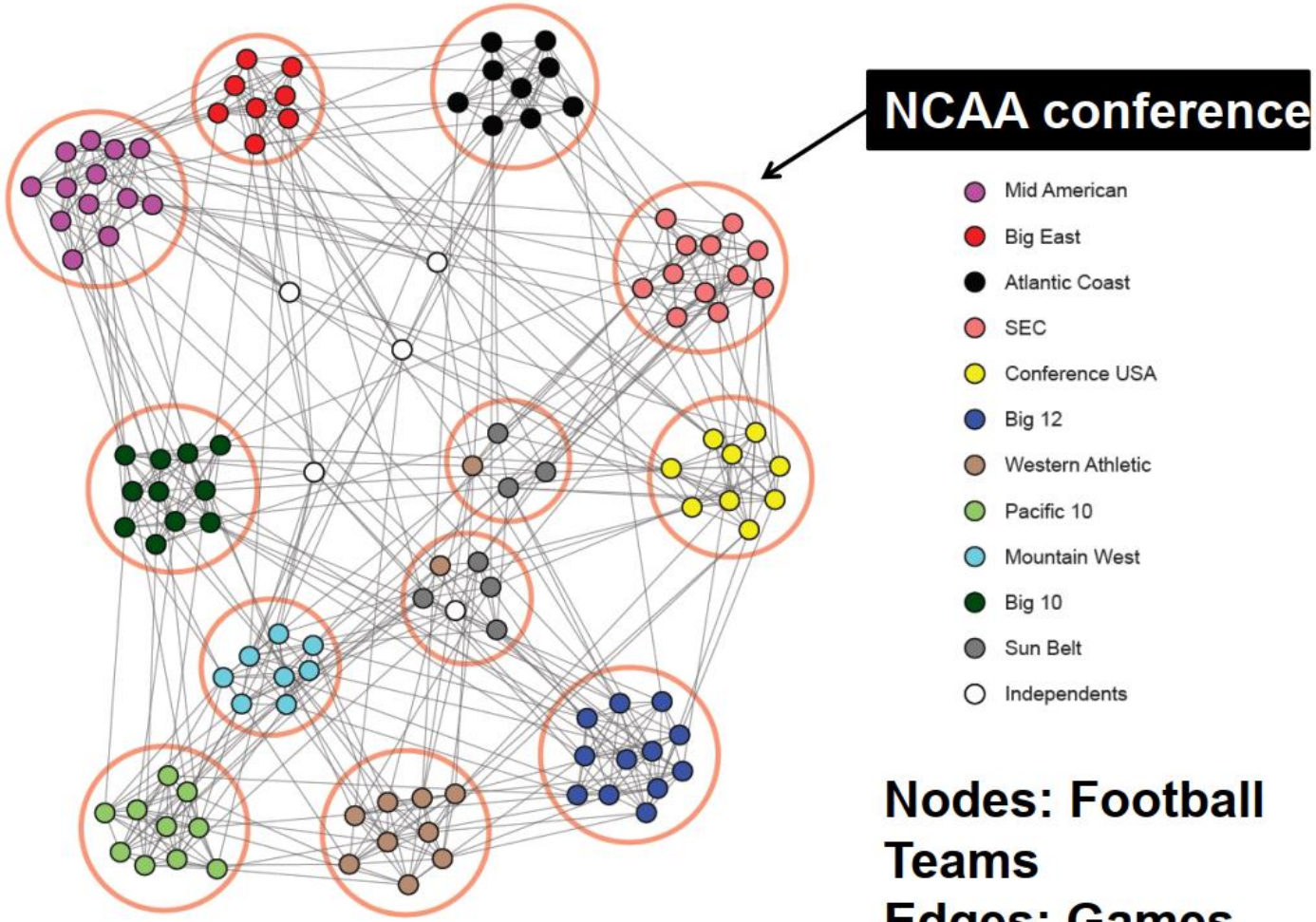


Can we identify node groups? (communities, modules, clusters)

Nodes: Football Teams
Edges: Games played

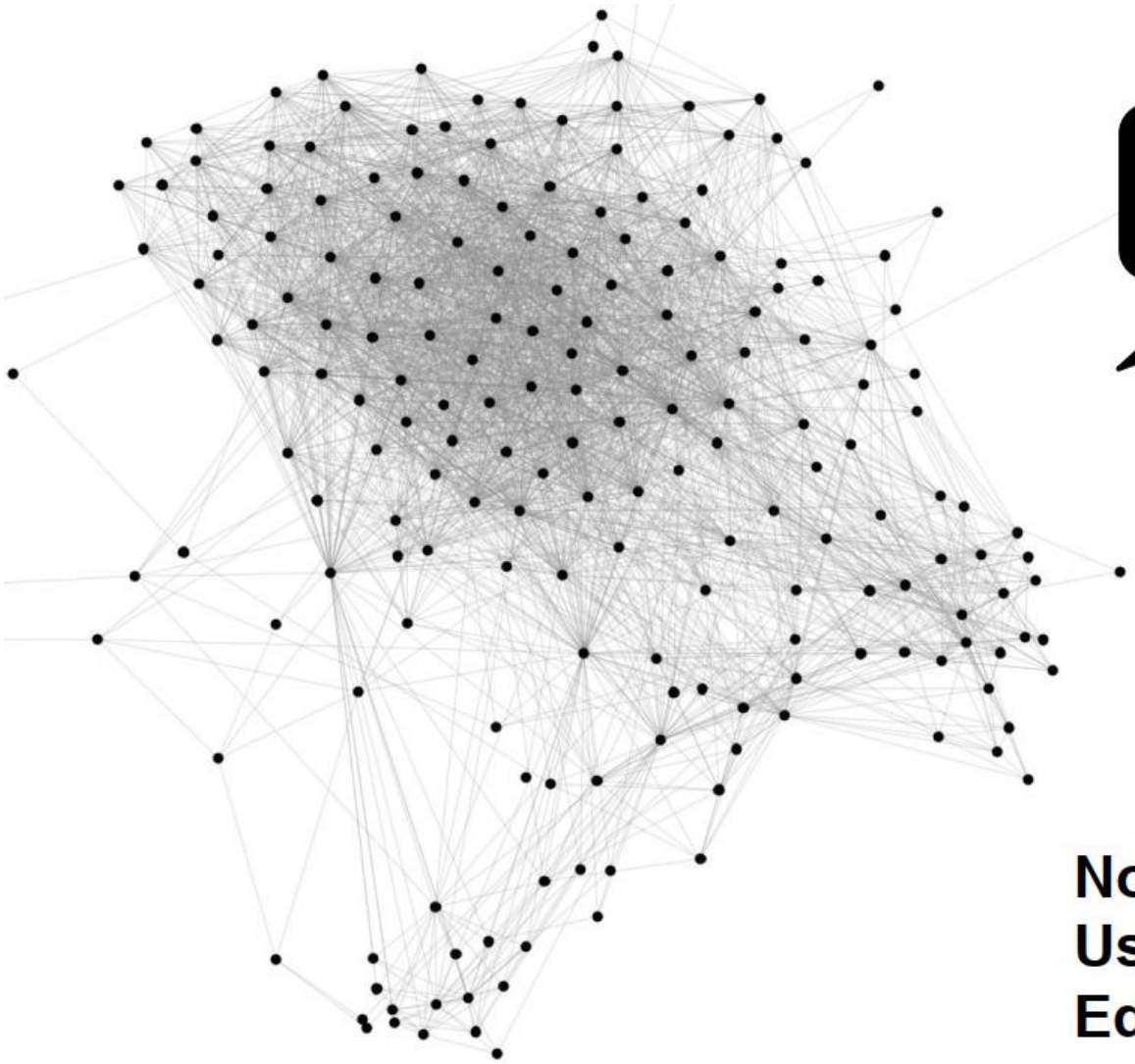
1. TỔNG QUAN

- **NCAA Football network**



1. TỔNG QUAN

- **Stanford Facebook network**

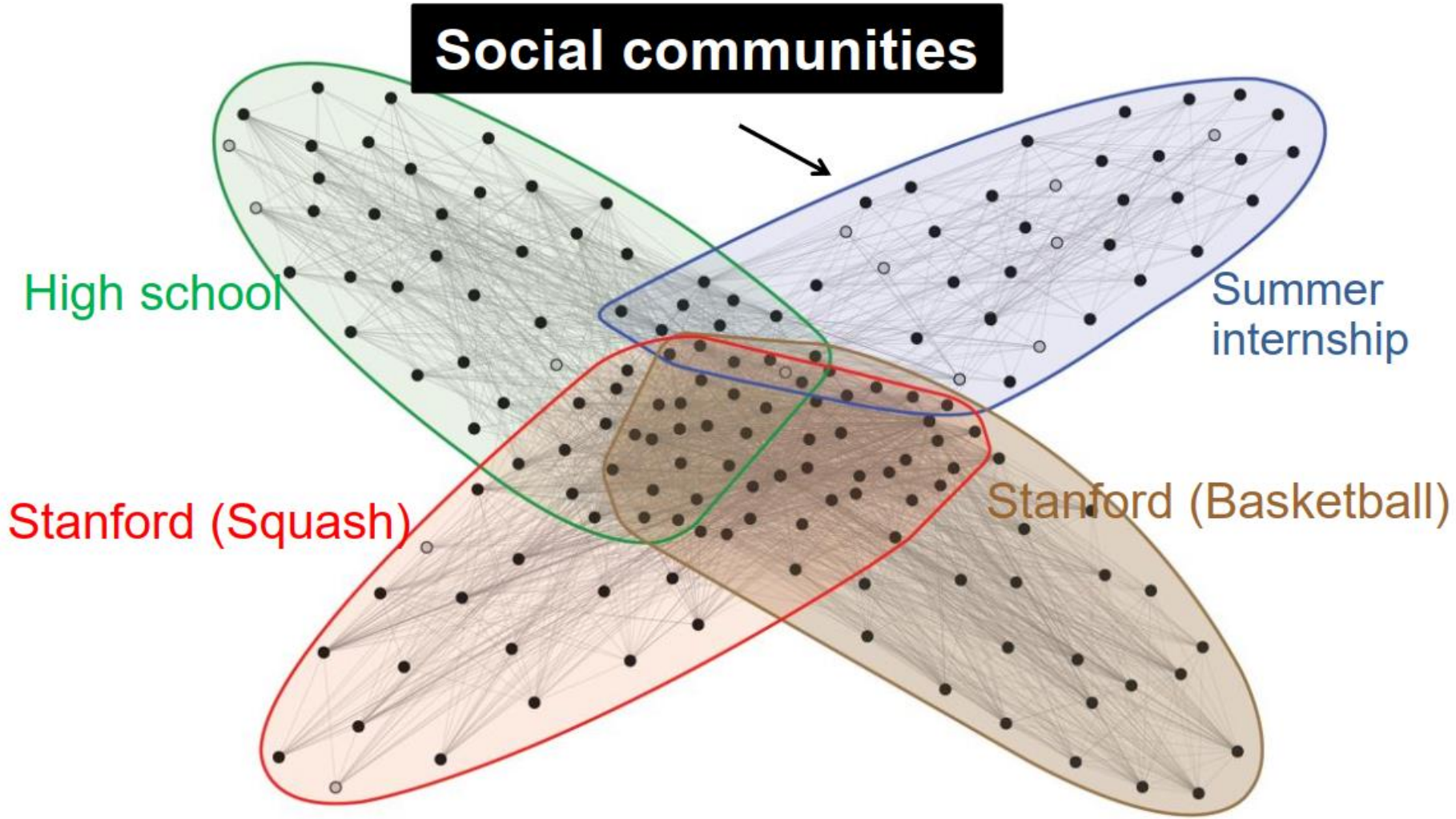


Can we identify social communities?

Nodes: Facebook Users
Edges: Friendships

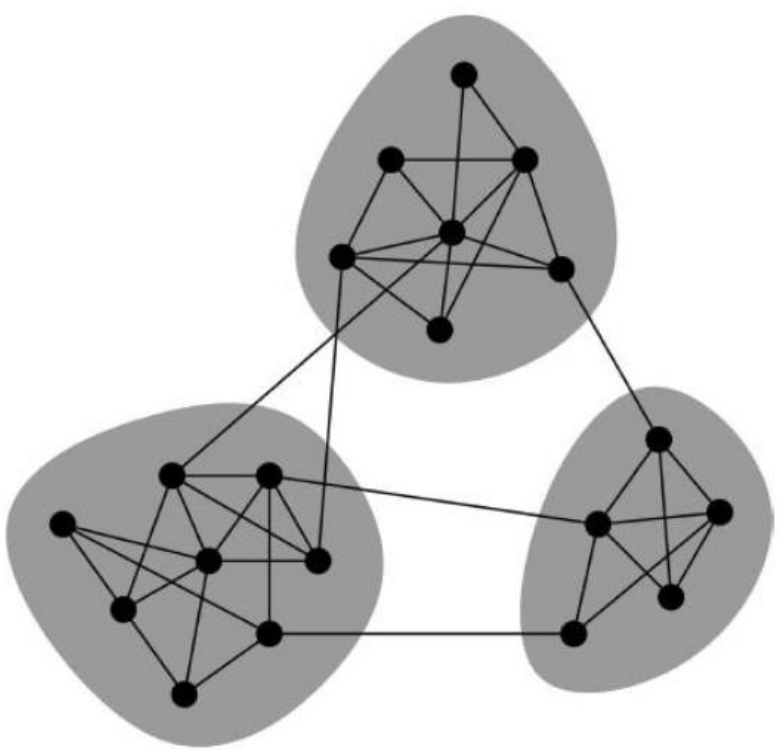
1. TỔNG QUAN

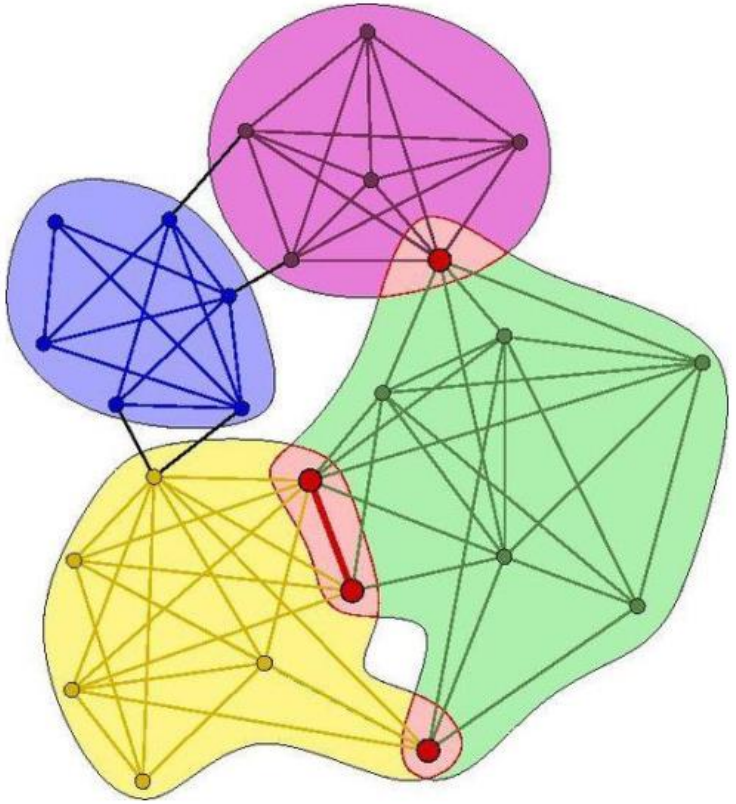
- **Stanford Facebook network**



2. CÁC LOẠI CỘNG ĐỒNG

- **Cộng đồng chồng chéo** (Overlapping communities)
- **Cộng đồng không chồng chéo** (Non-overlapping communities)

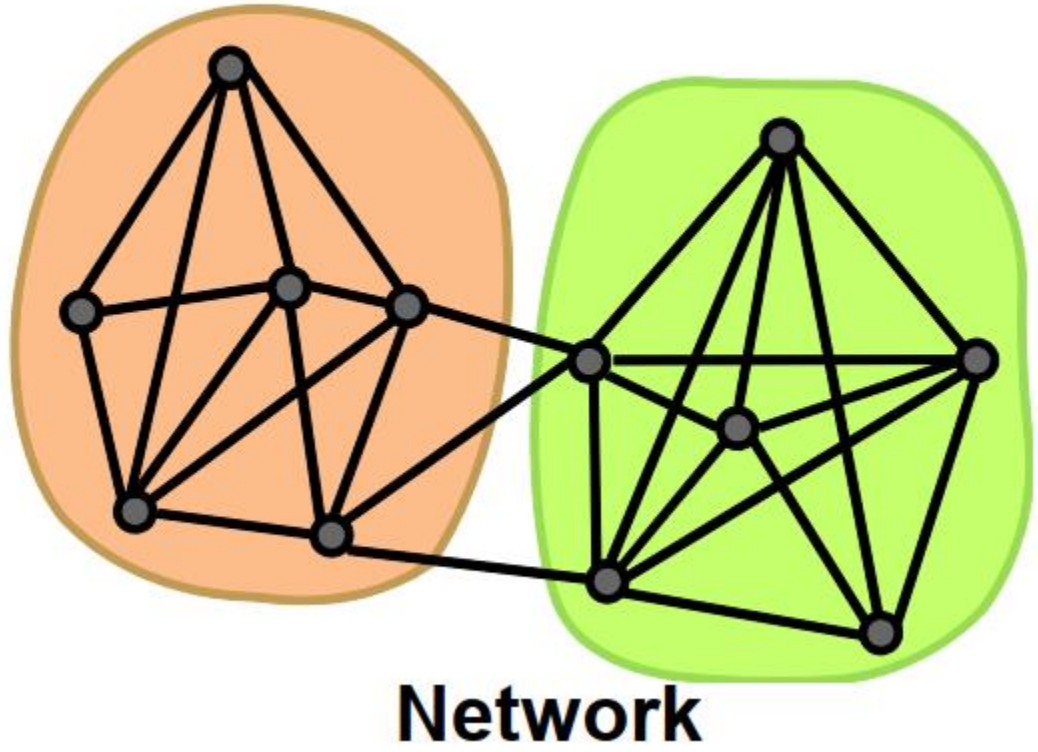


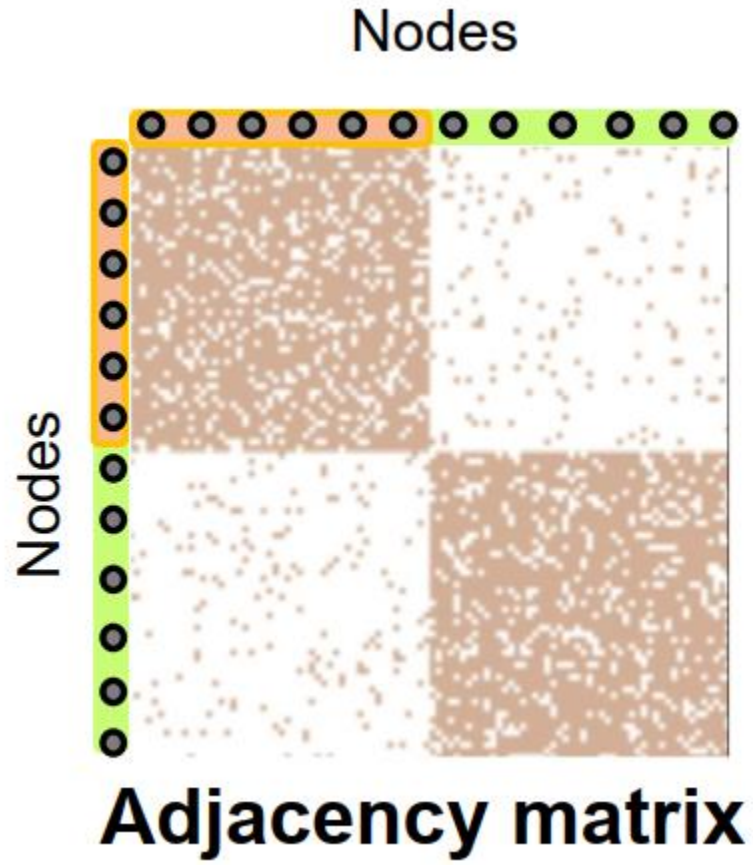


2. CÁC LOẠI CỘNG ĐỒNG

Cộng đồng không chồng chéo

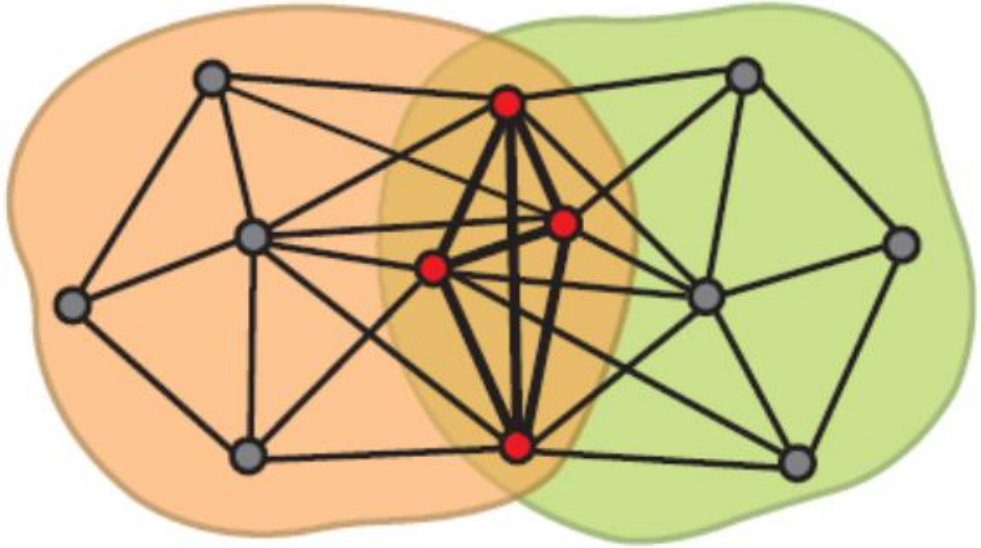
- Kết nối dày đặc trong cộng đồng, thưa thớt trên khắp các cộng đồng.

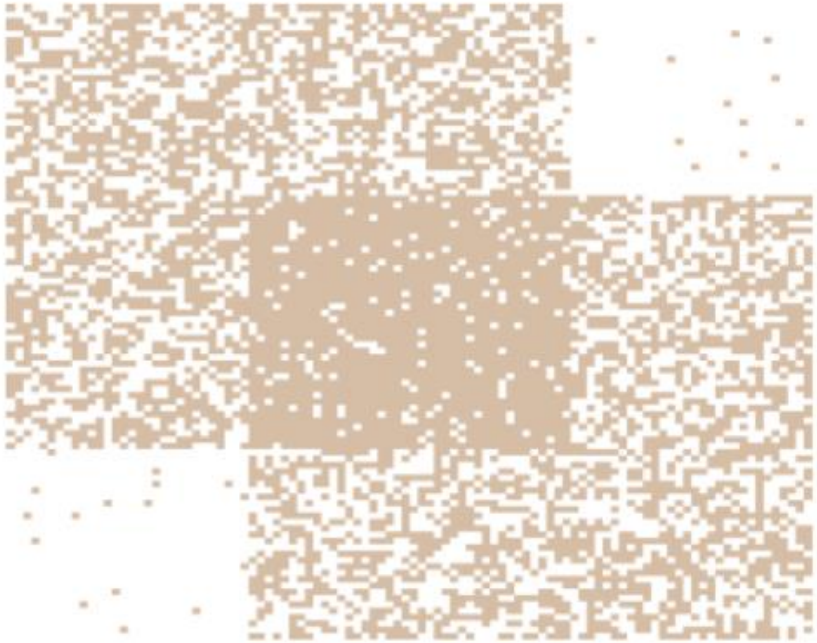




2. CÁC LOẠI CỘNG ĐỒNG

- **Cộng đồng chồng chéo**
- Kết nối dày đặc trong cộng đồng, có phần kết nối chung với các cộng đồng khác.





3. PHÁT HIỆN CỘNG ĐỒNG

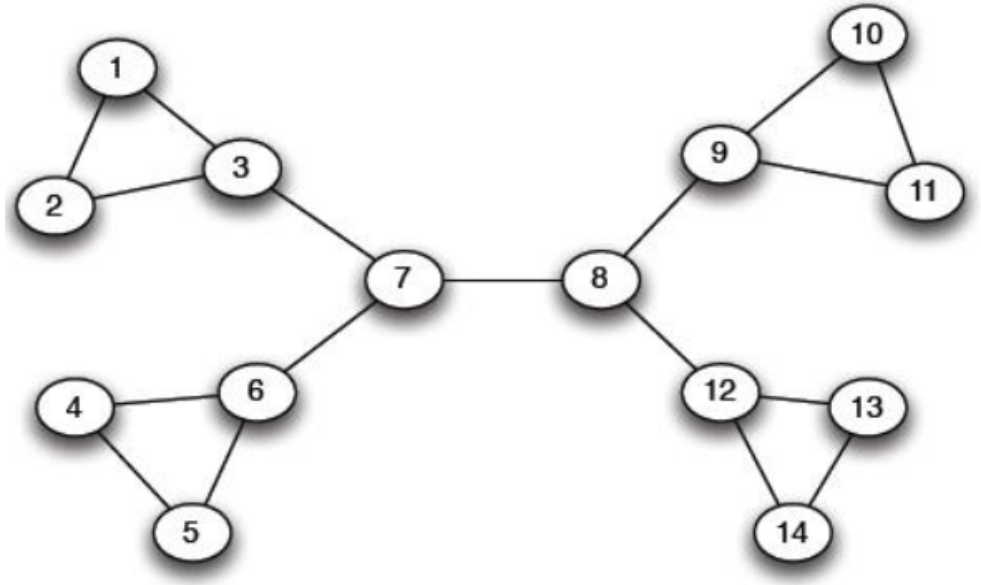
Phát hiện cộng đồng dưới dạng phân nhóm

- Theo nhiều cách, phát hiện cộng đồng chỉ là phân cụm trên đồ thị.
- Chúng ta có thể **áp dụng các thuật toán phân cụm trên ma trận kề**.
  - Ví dụ: k-mean.
- Có thể xác định **thước đo khoảng cách hoặc độ tương tự** giữa các nút trong biểu đồ và áp dụng các thuật toán khác (Ví dụ: phân nhóm phân cấp)
  - **Độ tương tự**: Bằng cách sử dụng độ tương tự jaccard trên các tập hợp hàng xóm.
  - **Khoảng cách**: Sử dụng đường đi ngắn nhất hoặc đi bộ ngẫu nhiên.
- **Ngoài ra, còn có các thuật toán dành riêng cho đồ thị**.

4. THUẬT TOÁN GIRVAN NEWMAN

Phương pháp chia thứ bậc

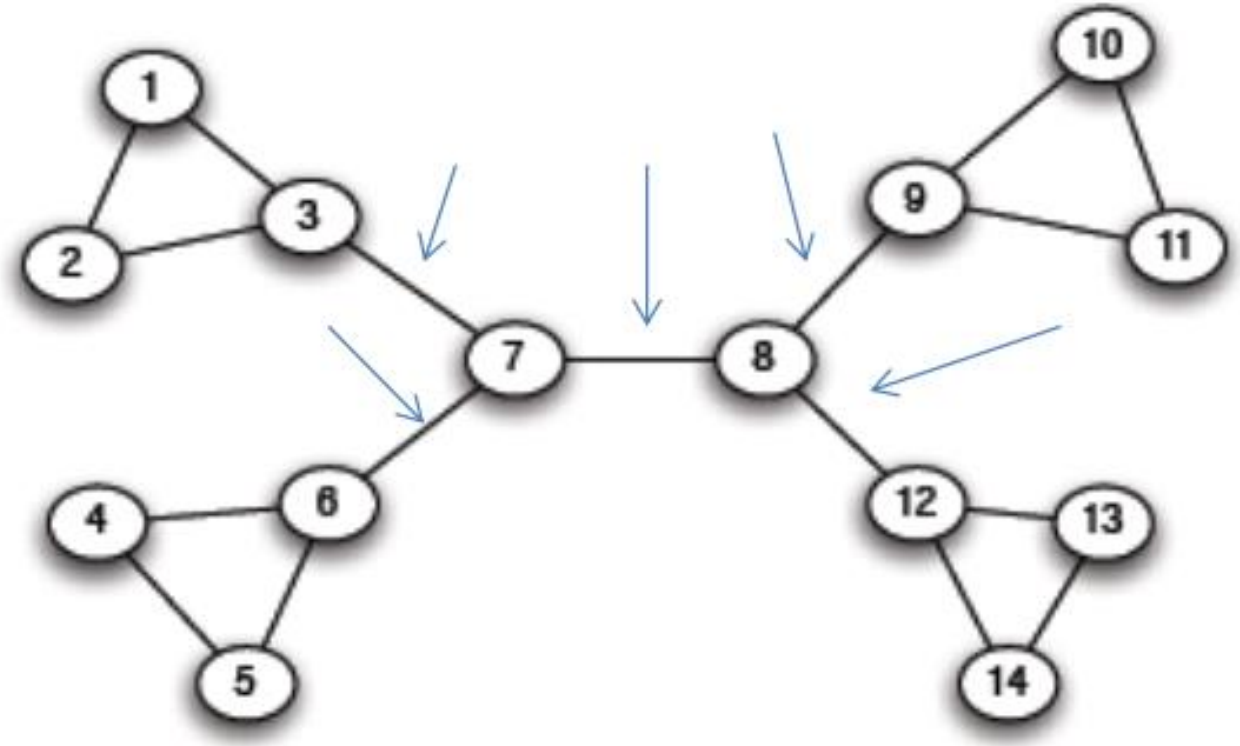
1. Bắt đầu với toàn bộ biểu đồ.
2. Tìm các cạnh cần loại bỏ, từ đó "phân vùng" biểu đồ.
3. Lặp lại với mỗi đồ thị con cho đến khi chỉ còn các đỉnh đơn.



Which edge to remove?

4. THUẬT TOÁN GIRVAN NEWMAN

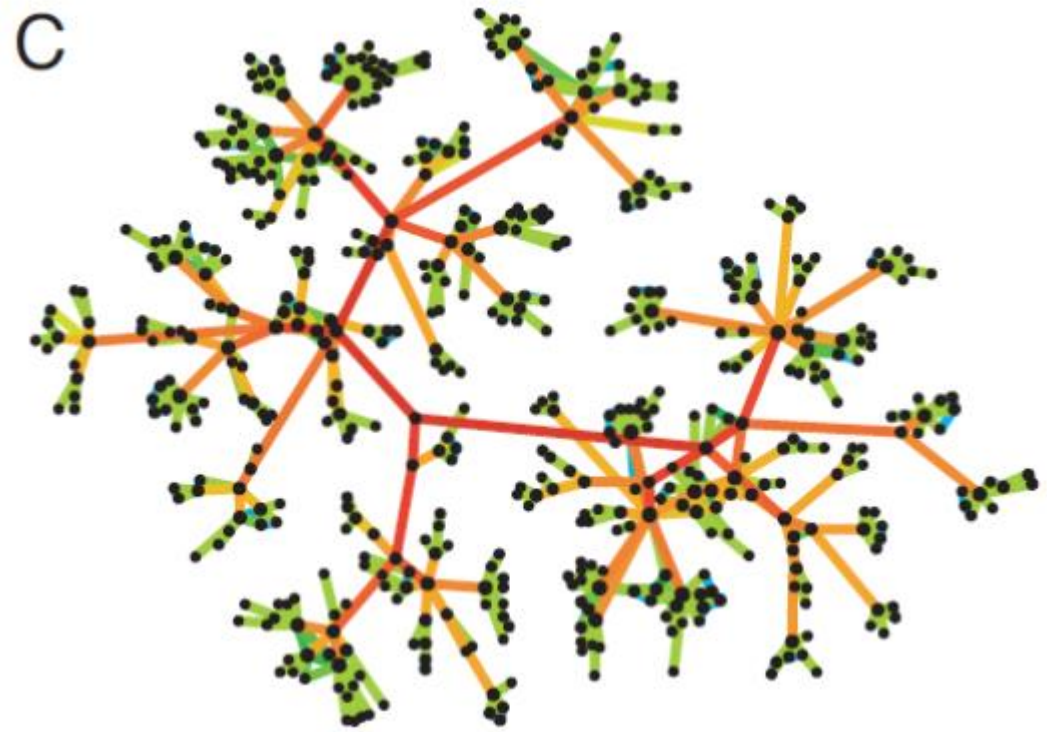
- Chọn các **cạnh cắt** (còn gọi là các **cạnh cầu**): các cạnh mà khi loại bỏ chúng sẽ ngắt kết nối biểu đồ.
- Có thể có nhiều.



4. THUẬT TOÁN GIRVAN NEWMAN

Tầm quan trọng của cạnh

- Chúng ta cần một thước đo về mức độ quan trọng của một cạnh trong việc giữ cho biểu đồ được kết nối.
- **Edge betweenness**: Số đường đi ngắn nhất đi qua cạnh.



4. THUẬT TOÁN GIRVAN NEWMAN

Edge Betweeness

Betweeness of edge (a, b) $(B(a, b))$

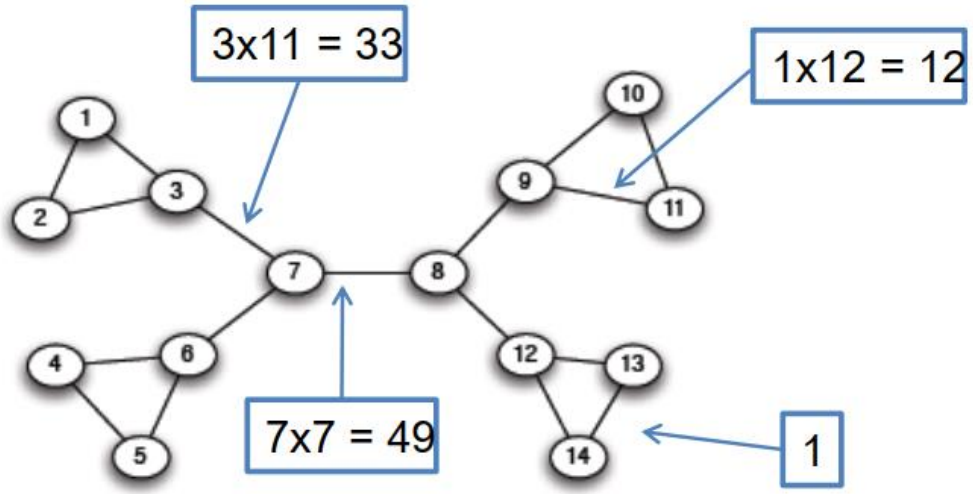
- Đối với mỗi cặp nút x, y tính số đường đi ngắn nhất bao gồm (a, b).
- Có thể có nhiều đường đi ngắn nhất giữa x, y $(SP\ (x, y))$. Tính phần xác xuất của đường đi ngắn nhất đi qua (a, b).

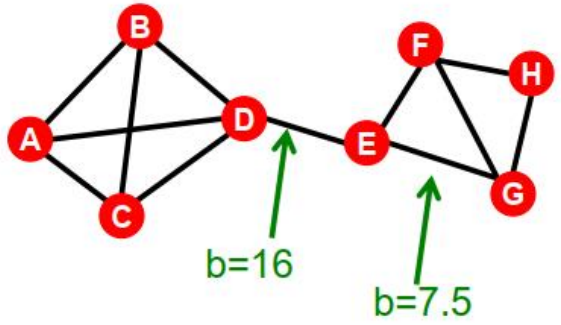
$$B(a,b) = \sum_{x,y \in V} \frac{|SP(x,y)\ that\ include\ (a,b)|}{|SP(x,y)|}$$

4. THUẬT TOÁN GIRVAN NEWMAN

Edge Betweeness

- **Ví dụ:**





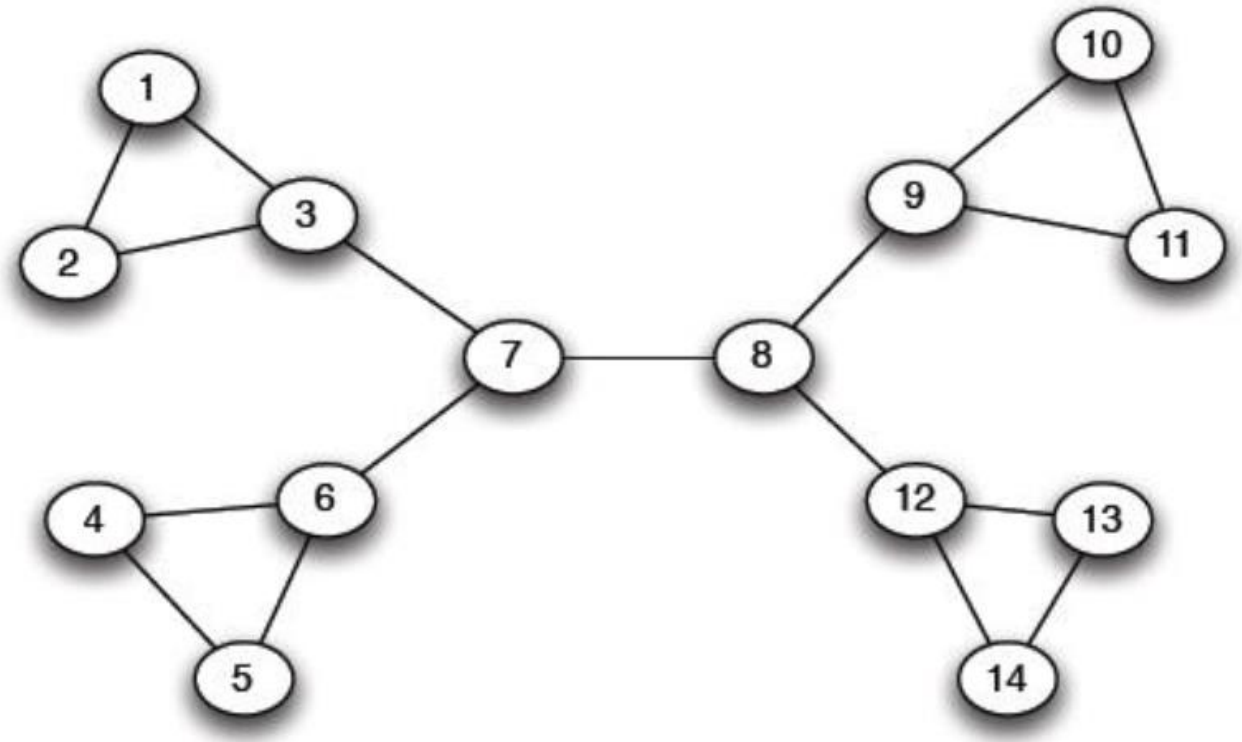
4. THUẬT TOÁN GIRVAN NEWMAN

- Một đồ thị không có trọng số vô hướng:
- Lặp lại cho đến khi không còn cạnh nào:
  - Tính edge betweenness cho tất cả các cạnh.
  - Xóa cạnh có edge betweenness cao nhất.

- Ở mỗi bước của thuật toán, **các thành phần được kết nối là các cộng đồng**.
- Cung cấp sự phân tách có thứ bậc của biểu đồ thành các cộng đồng.

4. THUẬT TOÁN GIRVAN NEWMAN

- **Ví dụ:**



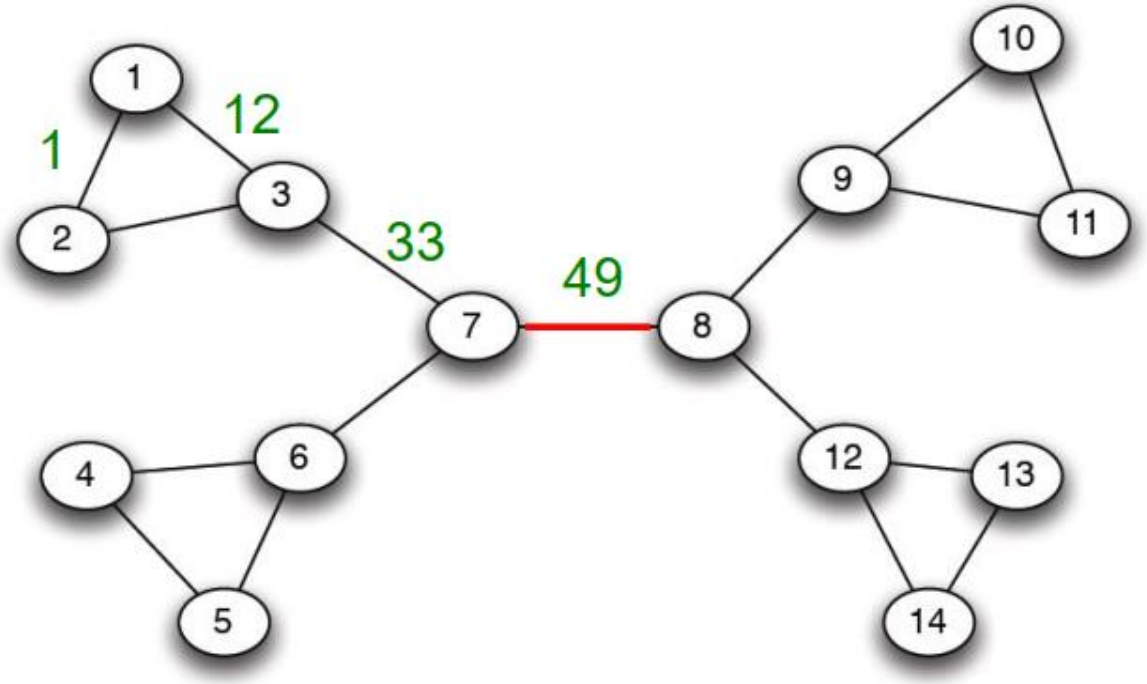
Betweenness(7, 8)= 7x7 = 49

Betweenness(1, 3) = 1X12=12

Betweenness(3, 7) = Betweenness(6, 7) =
Betweenness(8, 9) = Betweenness(8, 12)= 3X11=33

4. THUẬT TOÁN GIRVAN NEWMAN

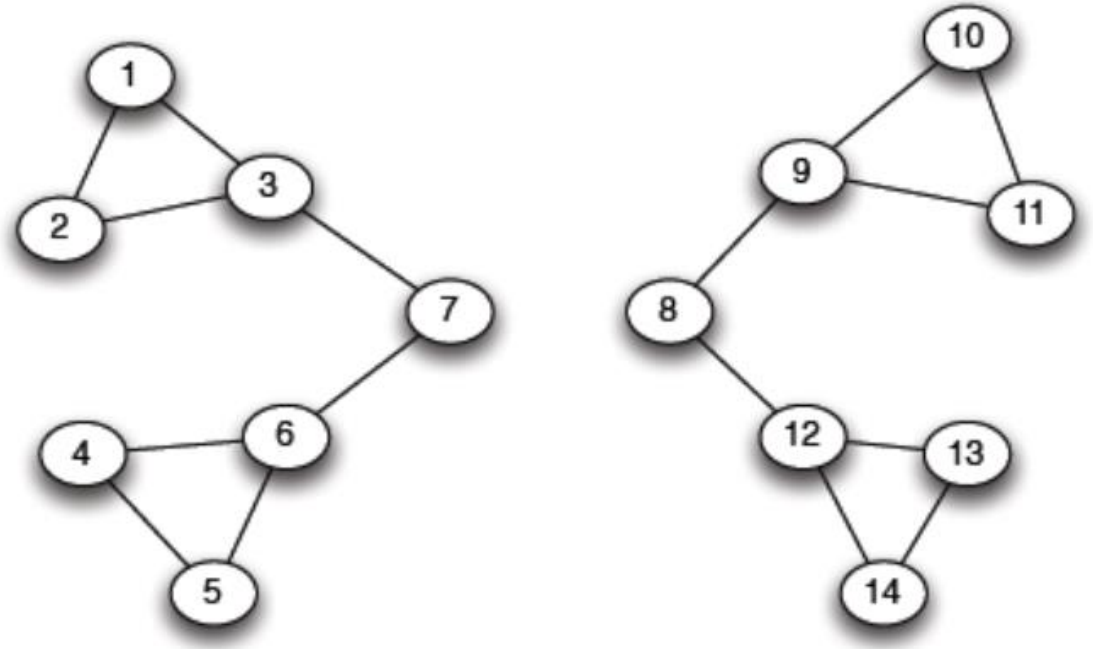
- **Ví dụ:**



Need to re-compute betweenness at every step

4. THUẬT TOÁN GIRVAN NEWMAN

- **Ví dụ:**



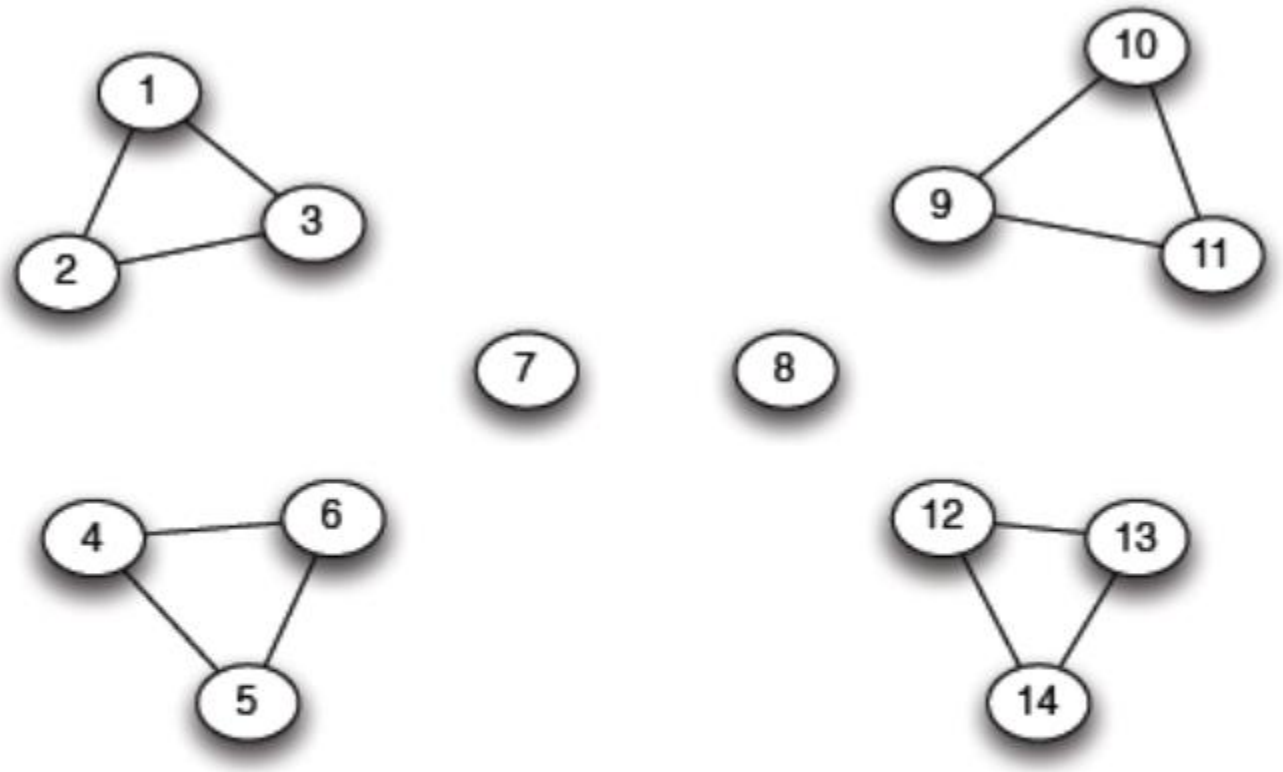
(a) *Step 1*

Betweenness(1, 3) = 1X5=5

Betweenness(3,7) = Betweenness(6,7) =
Betweenness(8,9) = Betweenness(8,12) = 3X4=12

4. THUẬT TOÁN GIRVAN NEWMAN

- **Ví dụ:**

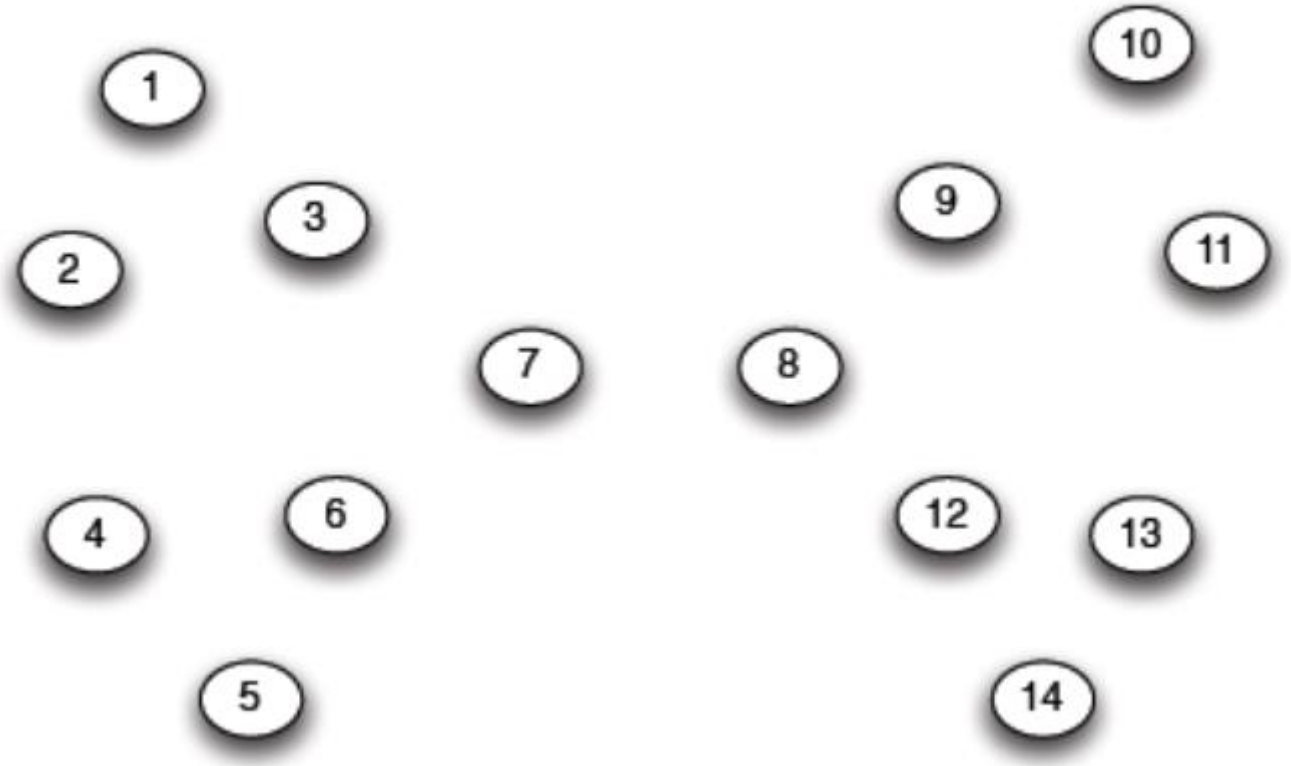


(b) *Step 2*

Betweenness of every edge = 1

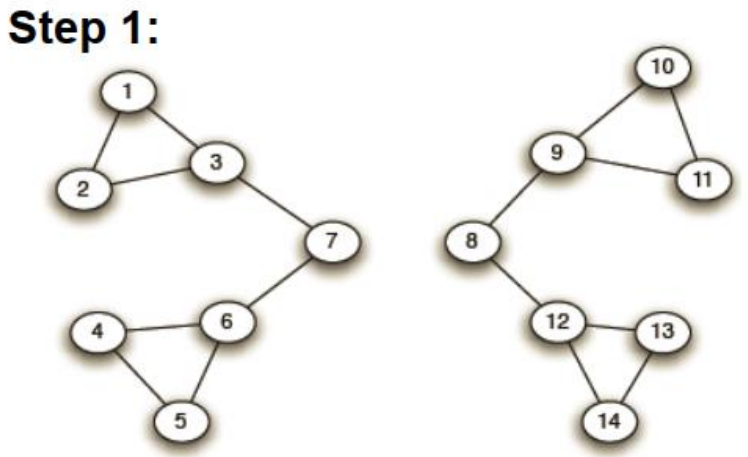
4. THUẬT TOÁN GIRVAN NEWMAN

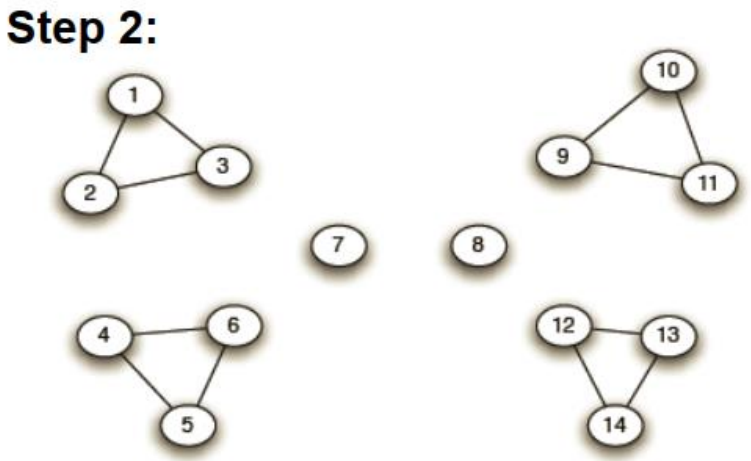
- **Ví dụ:**

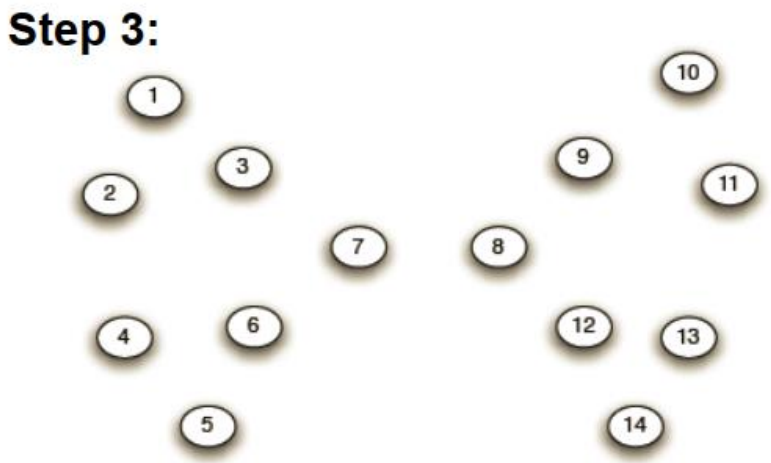


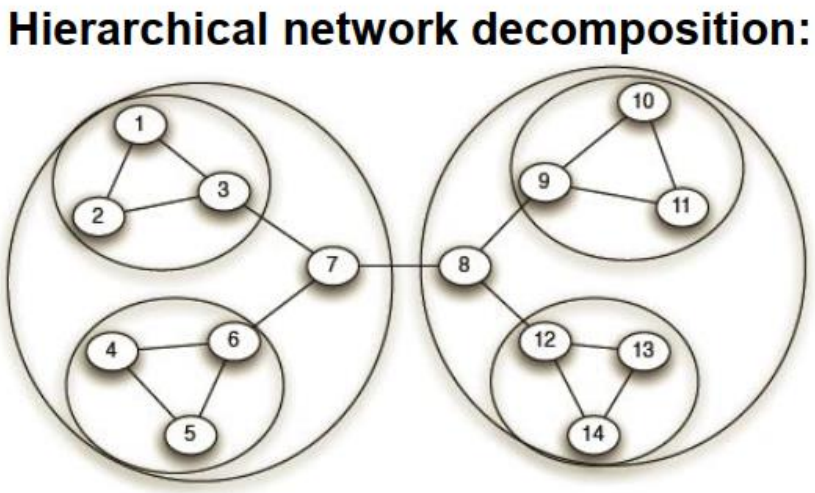
4. THUẬT TOÁN GIRVAN NEWMAN

- **Ví dụ:**



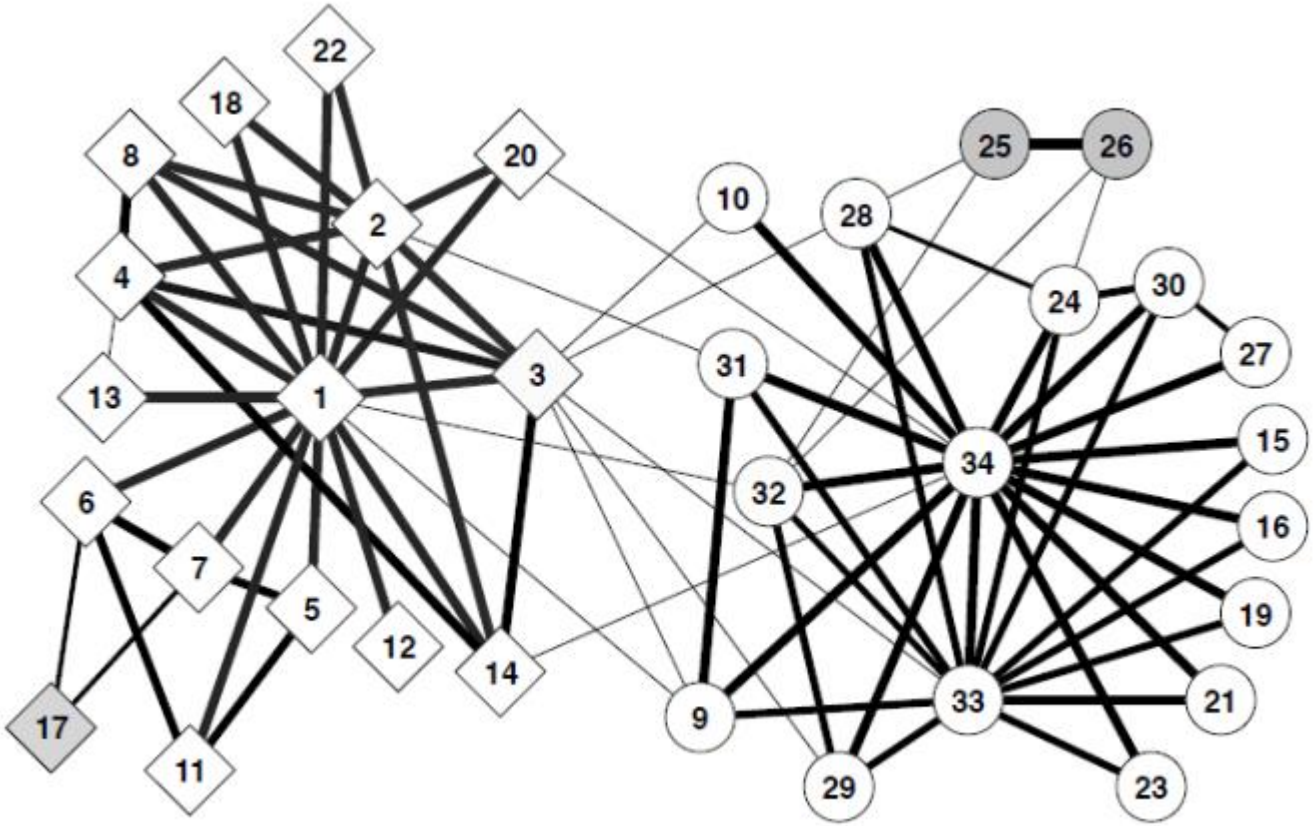


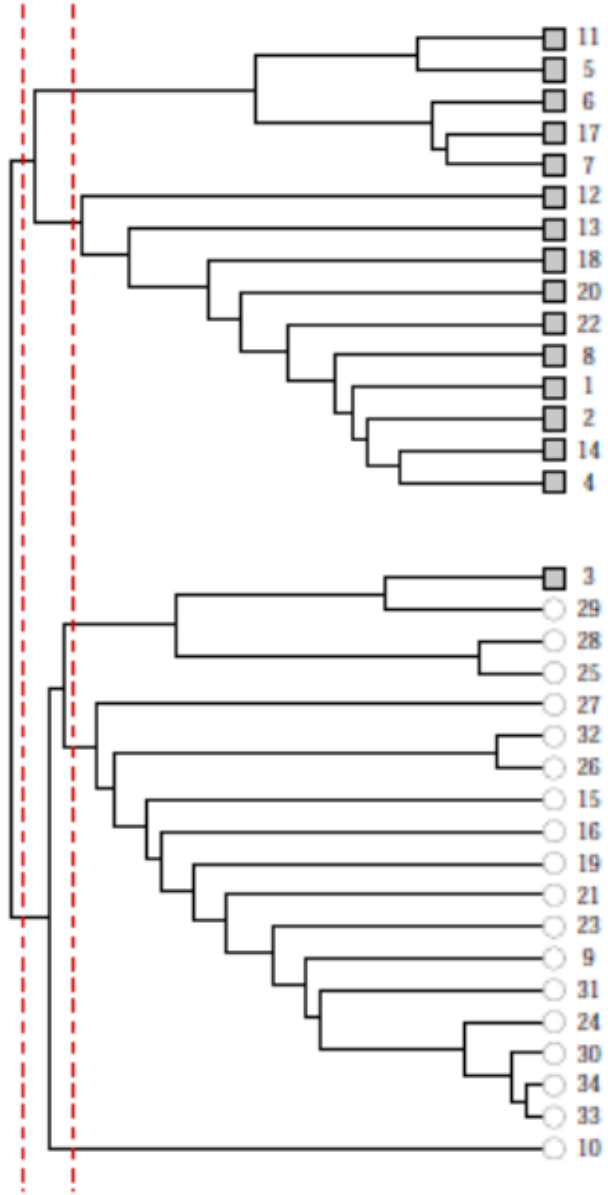




4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently





4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Đối với mỗi nút N trong biểu đồ:

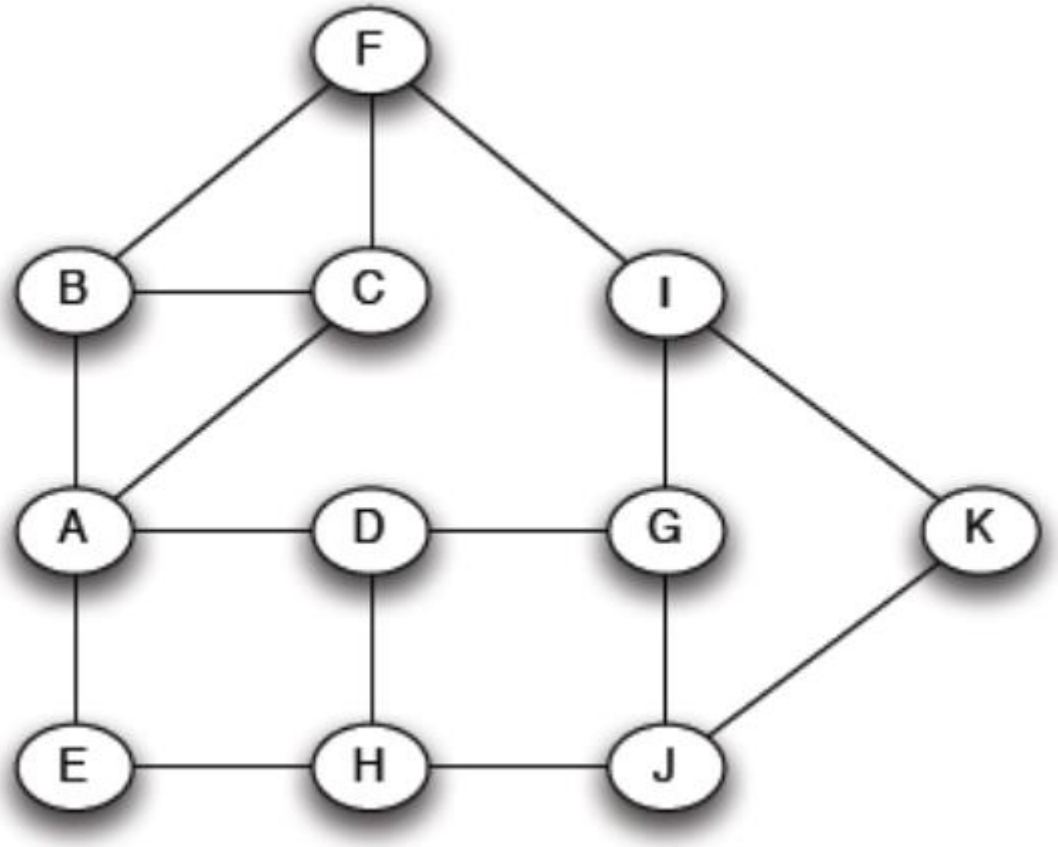
1. Thực hiện tìm kiếm theo **breadth-first search** của đồ thị bắt đầu từ nút N.
2. Xác định số đường đi ngắn nhất từ N đến mọi nút khác.
3. Dựa trên những con số này, hãy xác định lượng dòng chảy từ N đến tất cả các nút khác cho mỗi cạnh.

Chia tổng lưu lượng của tất cả các cạnh cho 2.

4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

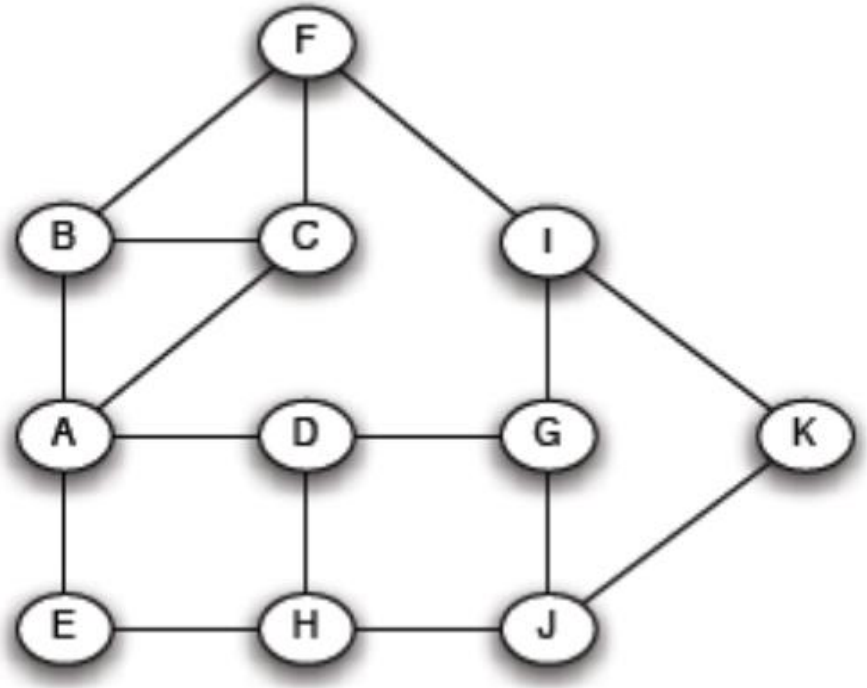
Bắt đầu từ nút *A*.



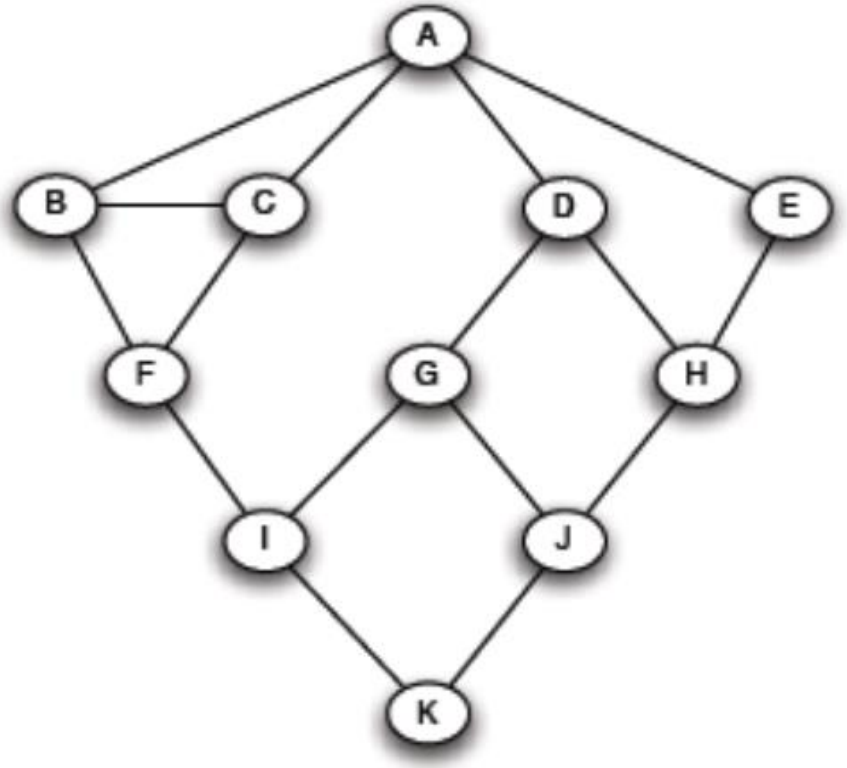
4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Bước 1



Initial network

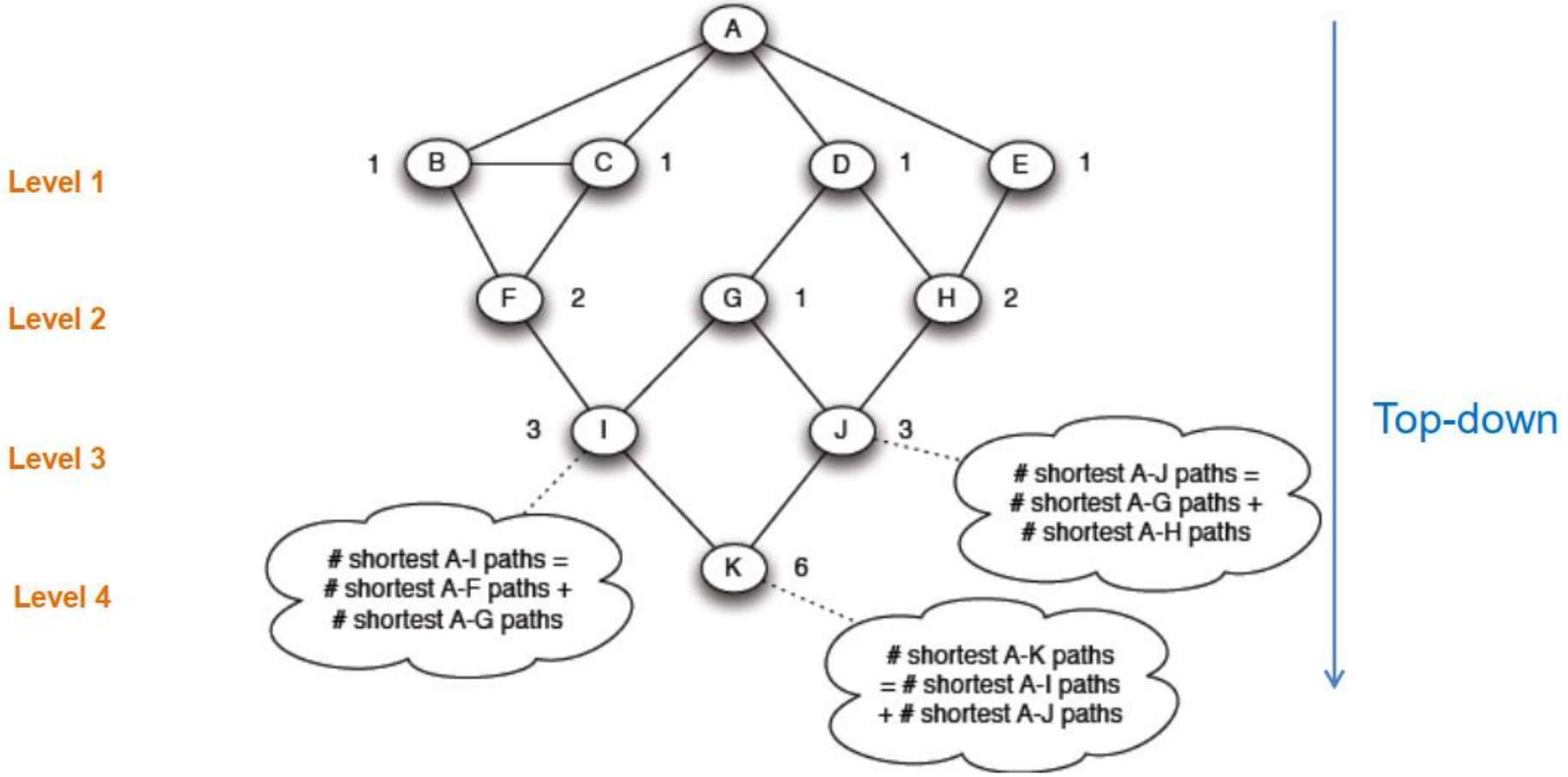


BFS from A

4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Bước 2: Đếm số đường đi ngắn nhất từ A đến một nút cụ thể.



4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Bước 3

- Tính toán **Betweenness** bằng cách làm việc trên cây:
  - Đối với mỗi nút, có một đơn vị lưu lượng dành cho nút đó và nó được chia thành từng phần cho các cạnh tiếp cận với nút đó.
    - Có một đơn vị của dòng chảy tới K, K qua các cạnh (I, K) và (J, K).
    - Vì có 3 con đường từ I đến K và 3 con đường từ J, mỗi cạnh nhận ½ dòng chảy: Betweeness ½.
  - Nếu nút có con cháu trong BFS, chúng ta cũng cần tính đến luồng đi từ nút đó tới nút con.

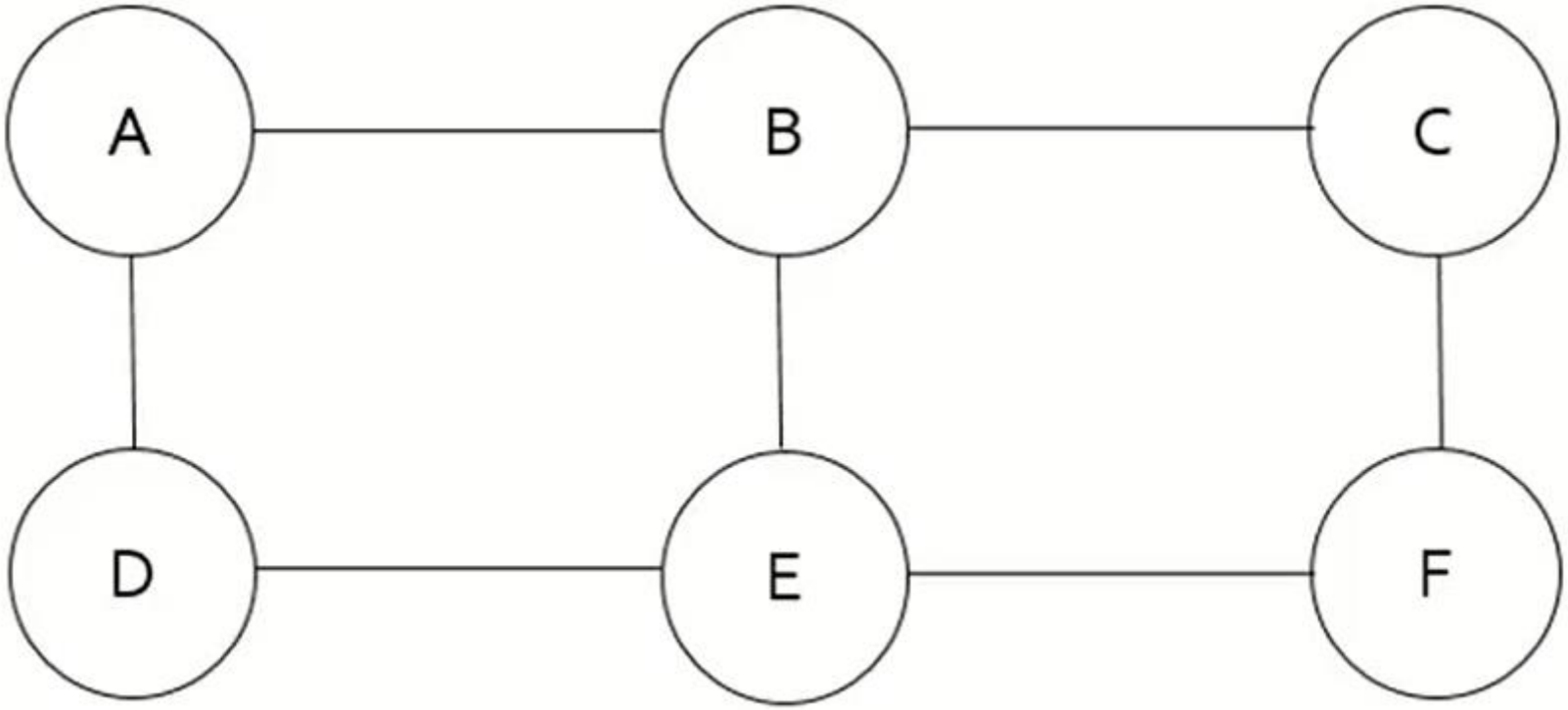
Lặp lại quy trình cho tất cả các nút và lấy tổng.

Chia tổng lưu lượng của tất cả các cạnh cho 2.

4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Ví dụ: Tính toán Edge Betweenness.

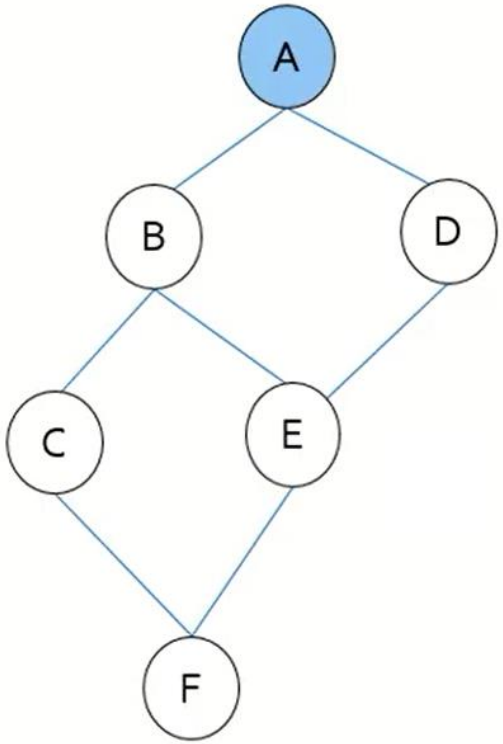


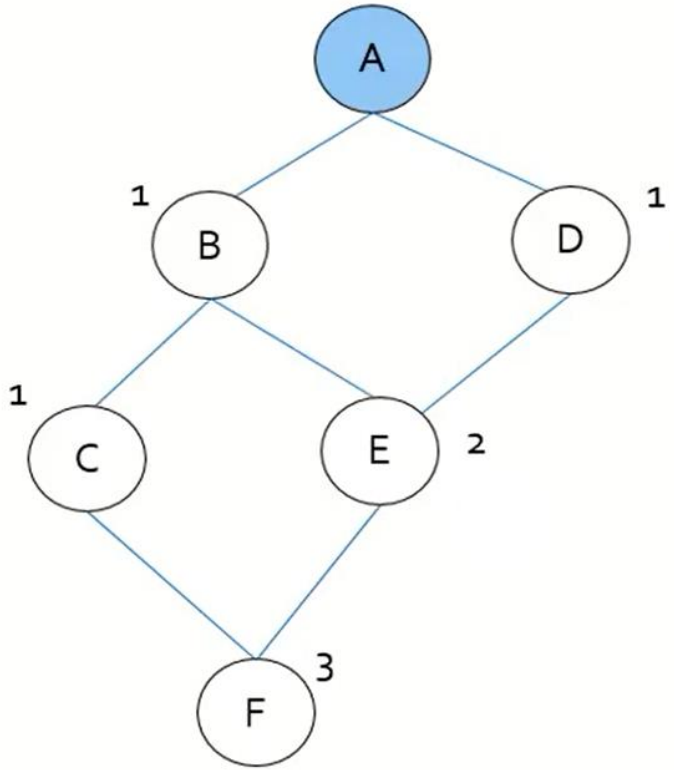
4. THUẬT TOÁN GIRVAN NEWMAN

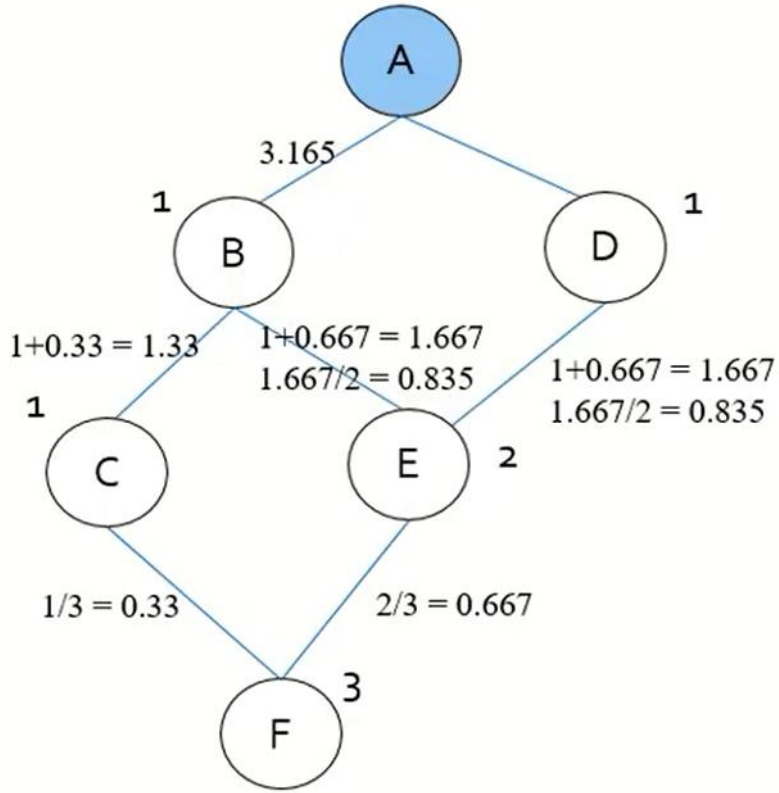
Method 2: Computing Edge Betweenness Efficiently

Ví dụ: Tính toán Edge Betweenness.

- Bắt đầu từ đỉnh A.





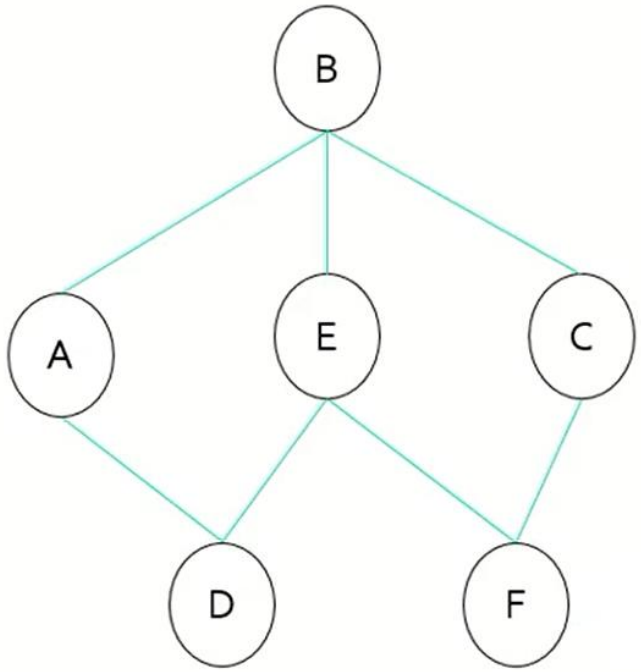


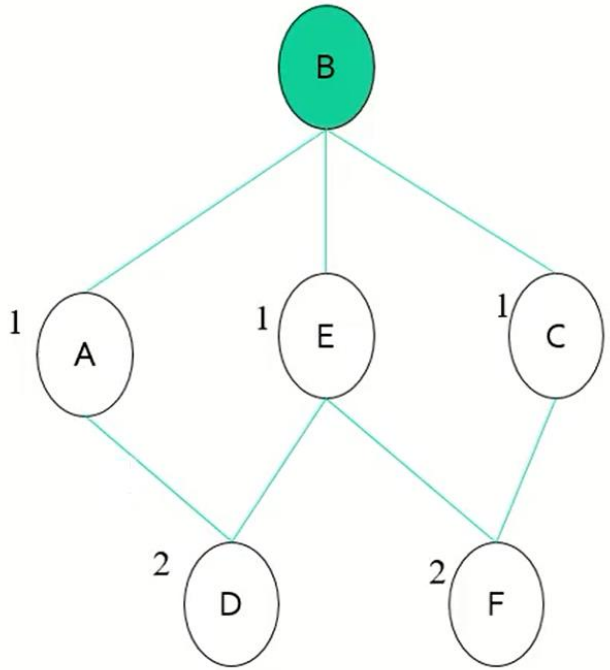
4. THUẬT TOÁN GIRVAN NEWMAN

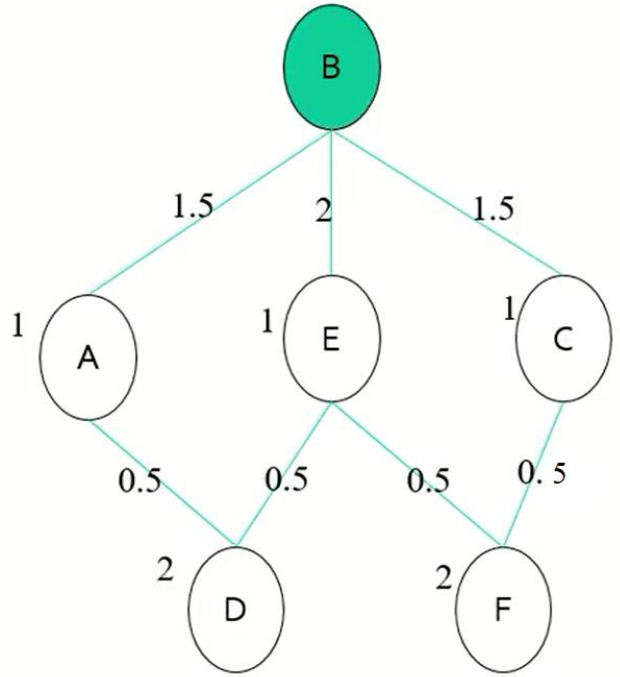
Method 2: Computing Edge Betweenness Efficiently

Ví dụ: Tính toán Edge Betweenness.

- Bắt đầu từ đỉnh B.





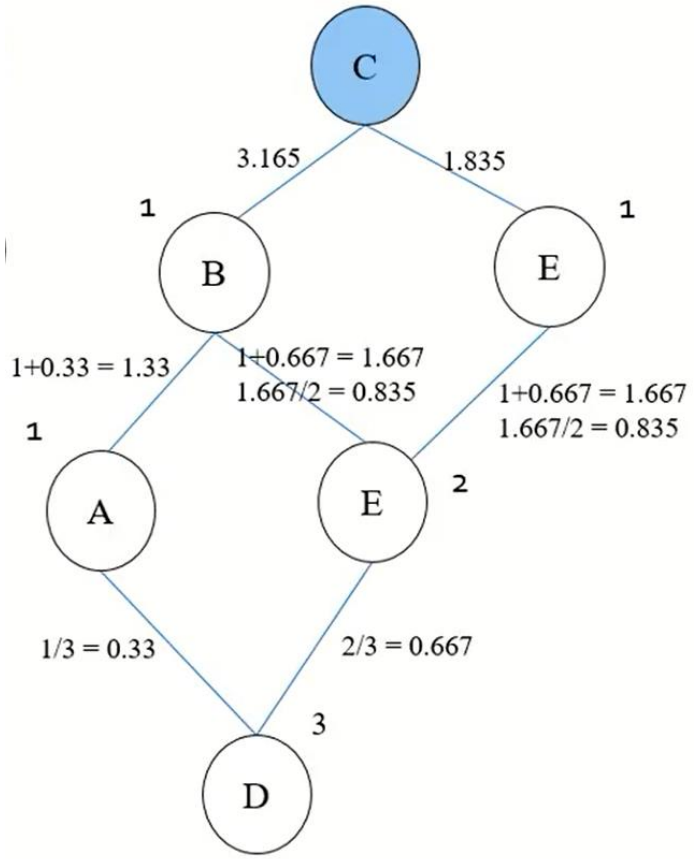


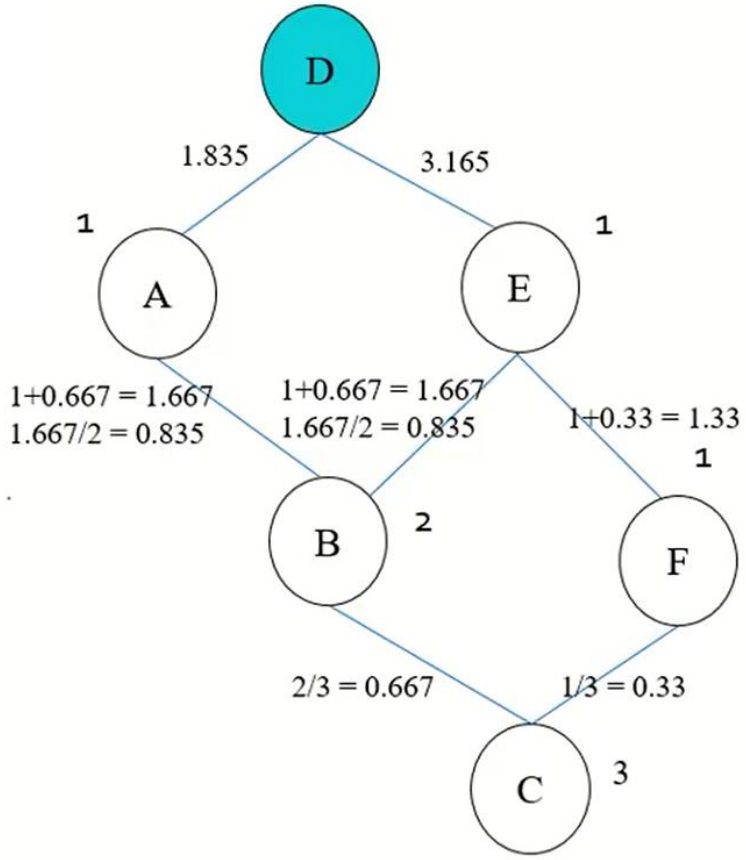
4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Ví dụ: Tính toán Edge Betweenness.

- Bắt đầu từ đỉnh C, D.



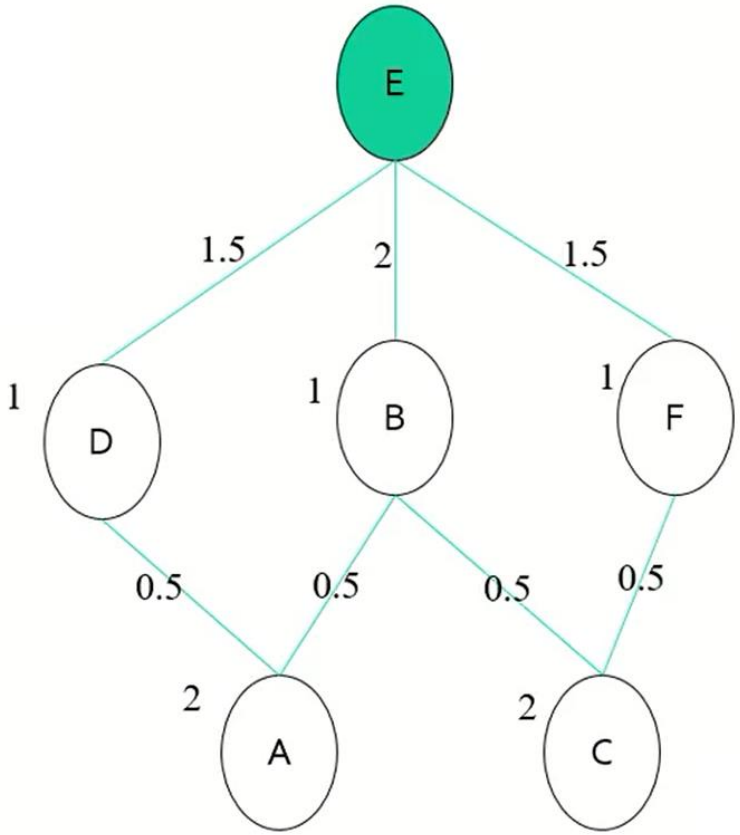


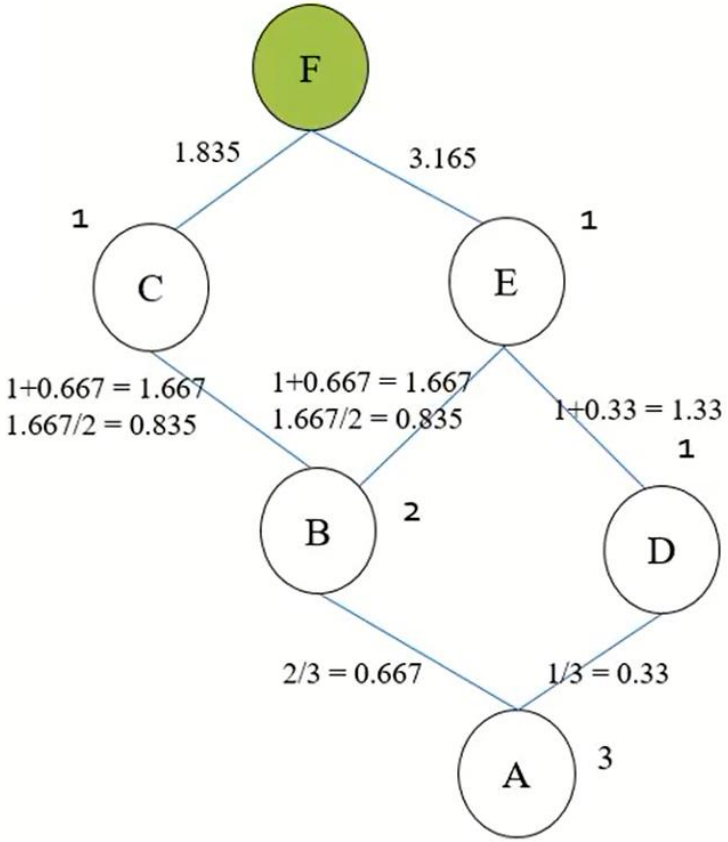
4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Ví dụ: Tính toán Edge Betweenness.

- Bắt đầu từ đỉnh E, F.

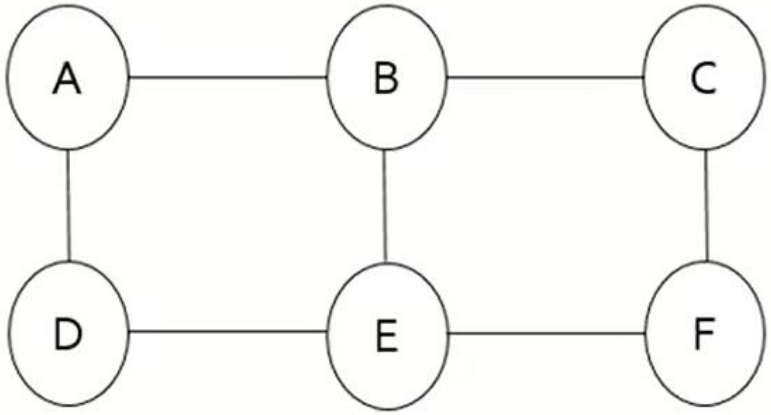




4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Ví dụ: Tính toán Edge Betweenness.

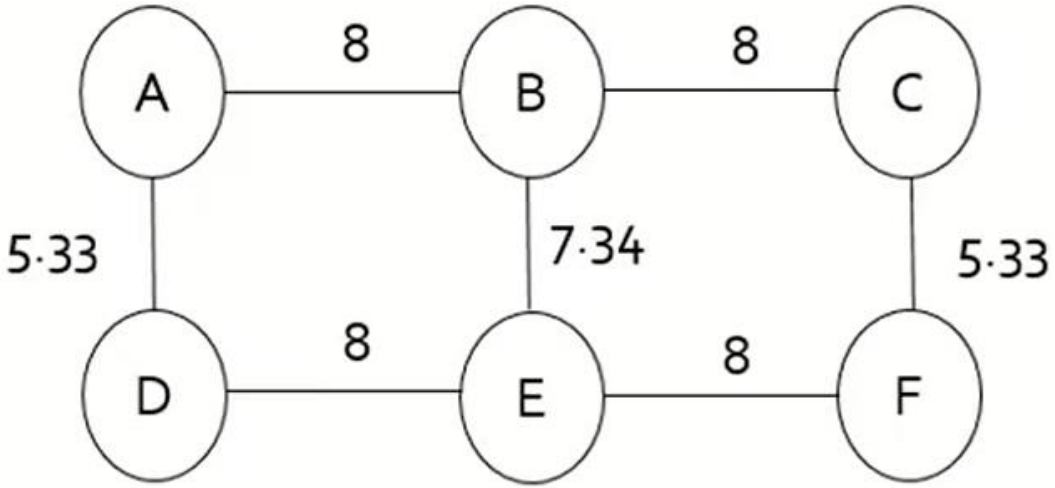


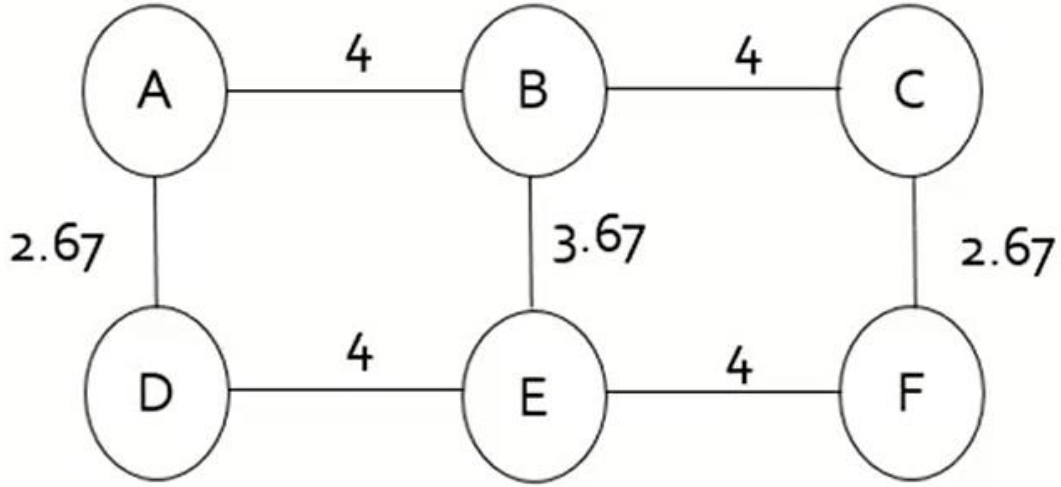
EDGES	EDGE BETWEENNESS
AB	3.165+1.5+1.33+0.835+0.5+0.667 = 8
AD	1.835+0.5+0.33+1.835+0.5+0.33 = 5.33
BC	3.165+1.5+1.33+0.835+0.5+0.667 = 8
BE	0.835+2+0.835++0.835+2+0.835 = 7.34
CF	1.835+0.5+0.33+1.835+0.5+0.33 = 5.33
DE	3.165+1.5+1.33+0.835+0.5+0.667 = 8
EF	3.165+1.5+1.33+0.835+0.5+0.667 = 8

4. THUẬT TOÁN GIRVAN NEWMAN

Method 2: Computing Edge Betweenness Efficiently

Ví dụ: Tính toán Edge Betweenness.

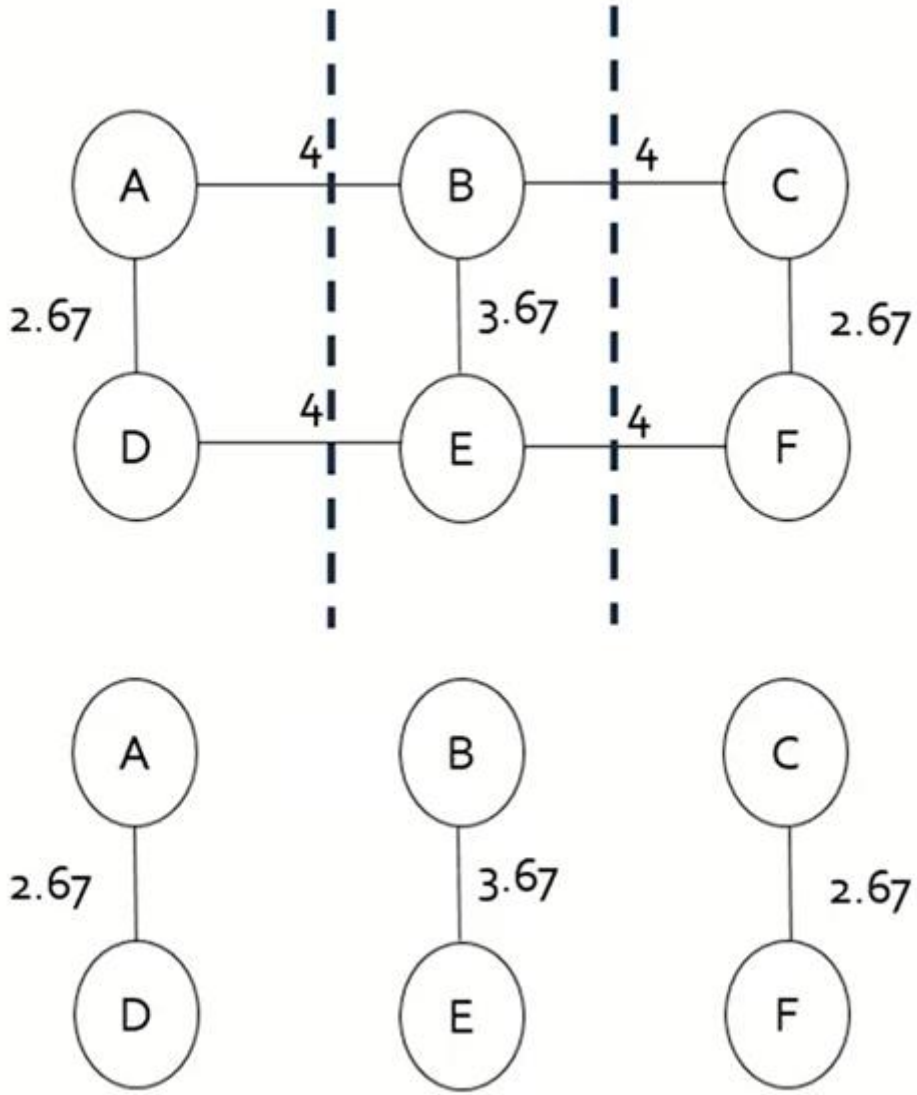




4. THUẬT TOÁN GIRVAN NEWMAN

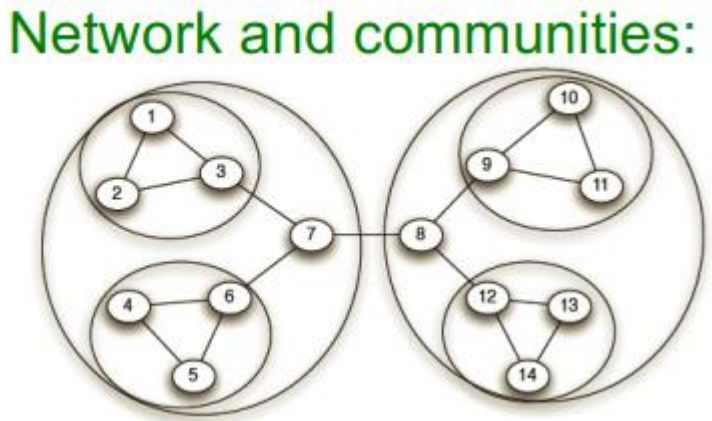
Method 2: Computing Edge Betweenness Efficiently

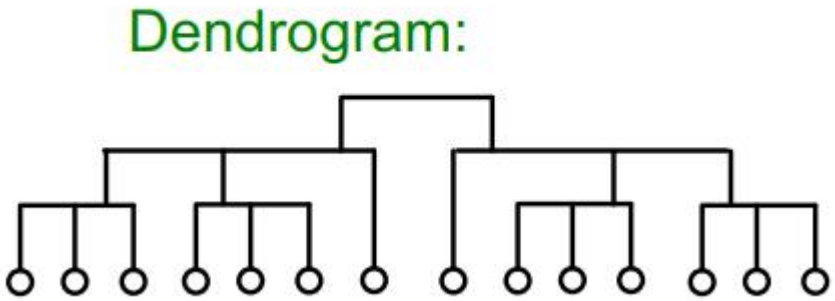
Ví dụ: Tính toán Edge Betweenness.



5. THUẬT TOÁN LOUVAIN

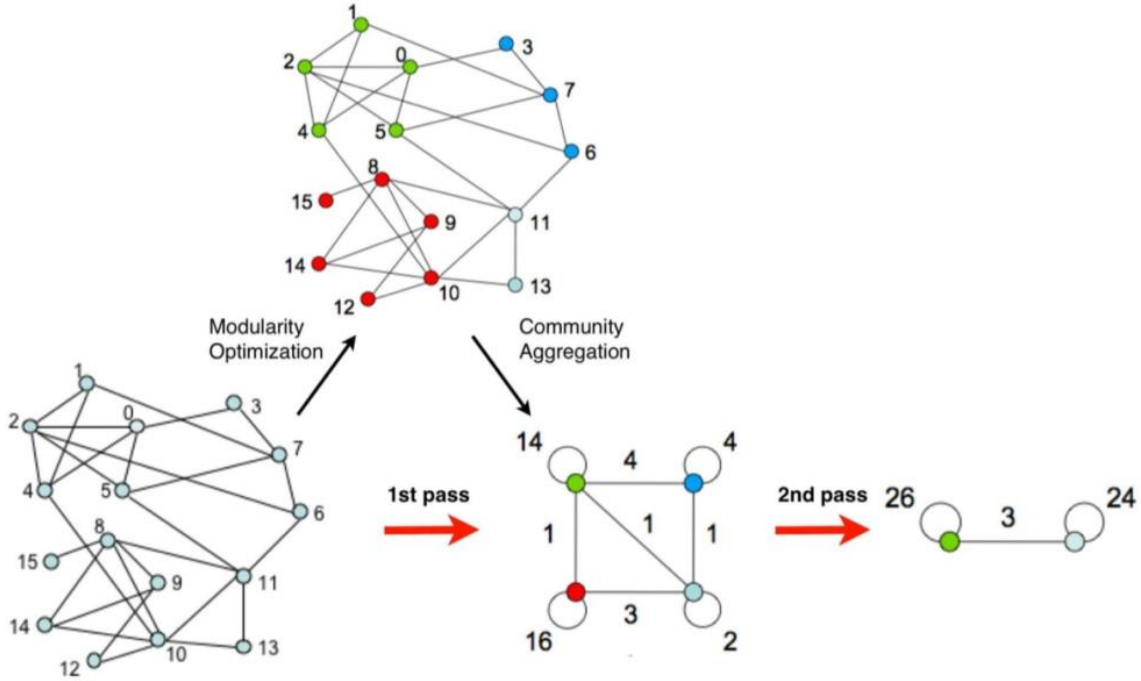
- **Thuật toán tham lam để phát hiện cộng đồng.**
  - $O\ (n \log n)$ thời gian chạy
- Hỗ trợ đồ thị có trọng số.
- Cung cấp các cộng đồng phân cấp.
- Được sử dụng rộng rãi để nghiên cứu các mạng lớn vì:
  - Nhanh.
  - Hội tụ nhanh chóng.
  - Kết quả đầu ra cao.





5. THUẬT TOÁN LOUVAIN

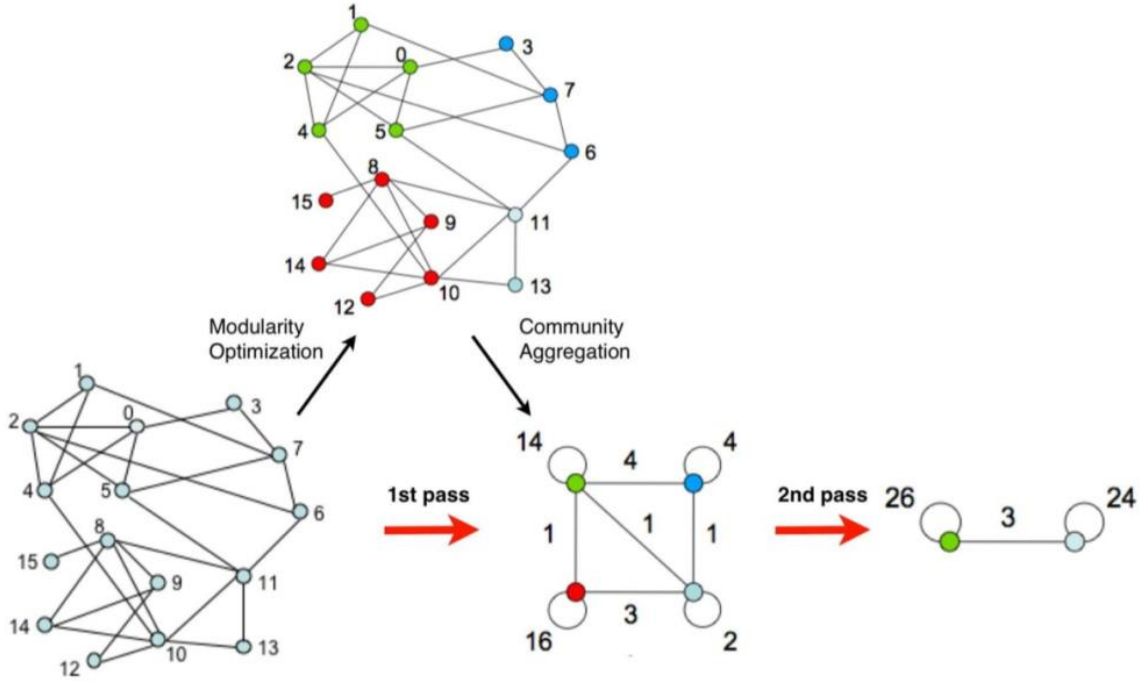
- **Thuật toán Louvain tối đa hóa module một cách tham lam.**
- Mỗi bước chuyển đổi được thực hiện bằng 2 giai đoạn:
  - **Giai đoạn 1**: Module được tối ưu hóa bằng cách chỉ cho phép các thay đổi cục bộ đối với tư cách thành viên của cộng đồng nút.
  - **Giai đoạn 2**: Các cộng đồng đã xác định được tổng hợp thành các siêu nút để xây dựng một mạng mới.
- **Quay lại giai đoạn 1**.
- **Các bước chuyển đổi được lặp đi lặp lại cho đến khi không thể tăng module.**



5. THUẬT TOÁN LOUVAIN

Thuật toán Louvain coi đồ thị là có trọng số.

- Đồ thị ban đầu có thể không có trọng số (*tức là, trọng số các cạnh đều là 1*).
- Khi các cộng đồng được xác định và tổng hợp thành các siêu nút, các đồ thị có trọng số được tạo ra (*các trọng số đếm số cạnh trong biểu đồ ban đầu*).
- Phiên bản có trọng số của module được áp dụng.



5. THUẬT TOÁN LOUVAIN

Giai đoạn 1: Phân vùng

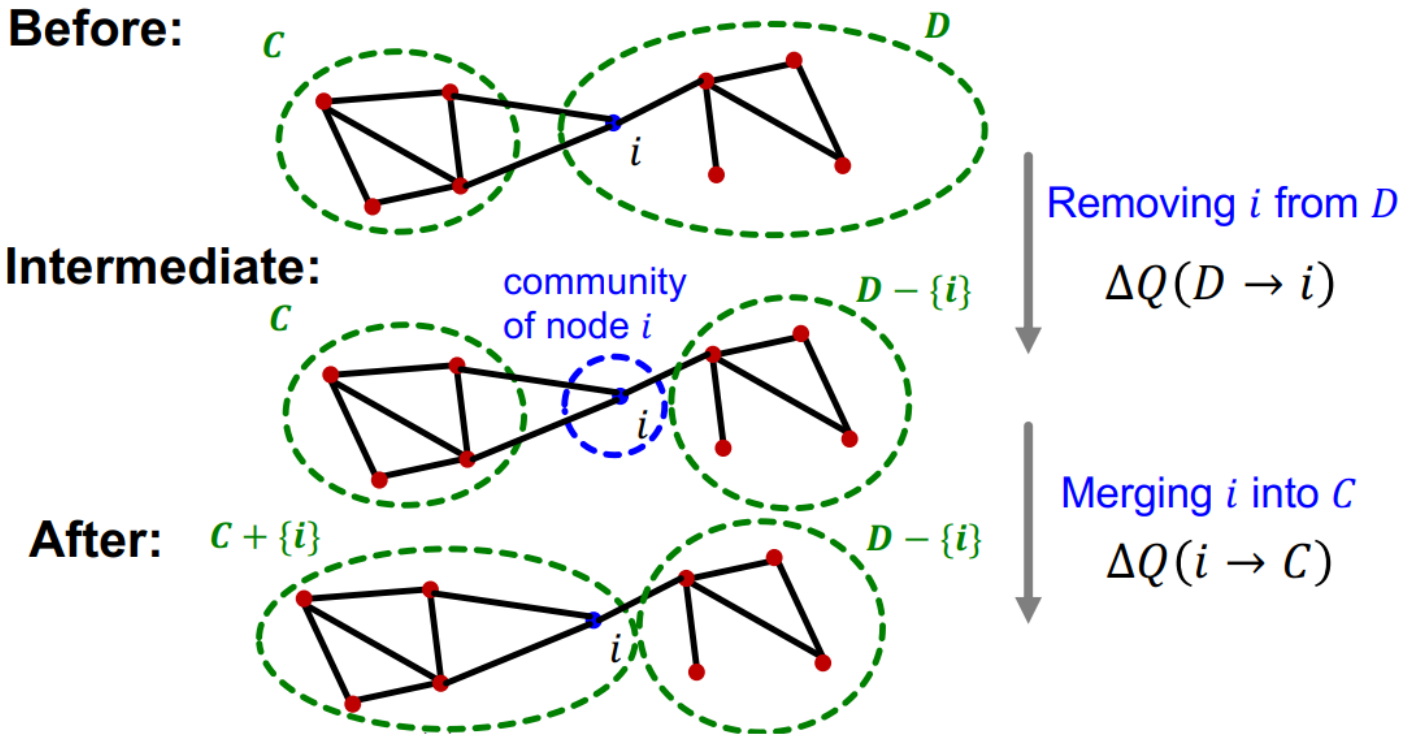
- Đặt mỗi nút trong biểu đồ thành một cộng đồng riêng biệt (một nút cho mỗi cộng đồng).
- Đối với mỗi nút i, thuật toán thực hiện hai phép tính:
  - Tính toán delta (ΔQ) khi đưa nút i vào cộng đồng của một số láng giềng j.
  - Di chuyển i đến một cộng đồng của nút j mang lại mức tăng lớn nhất trong ΔQ.
- Giai đoạn 1 chạy cho đến khi không có chuyển động nào mang lại hiệu quả hơn.

5. THUẬT TOÁN LOUVAIN

Giai đoạn 1: Phân vùng

- **Modularity Gain**
  - ΔQ là gì nếu nút i chuyển từ cộng đồng D sang C?

$$\Delta Q(D \to i \to C) = \Delta Q(D \to i) + \Delta Q(i \to C)$$



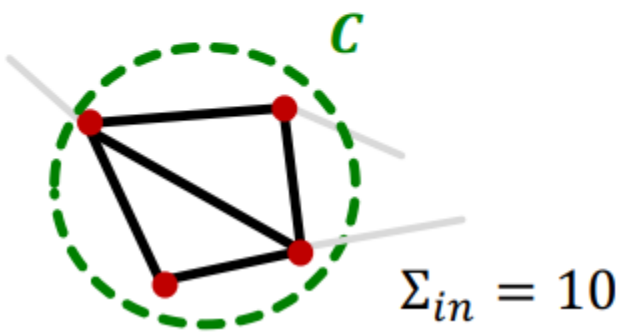
5. THUẬT TOÁN LOUVAIN

Giai đoạn 1: Phân vùng

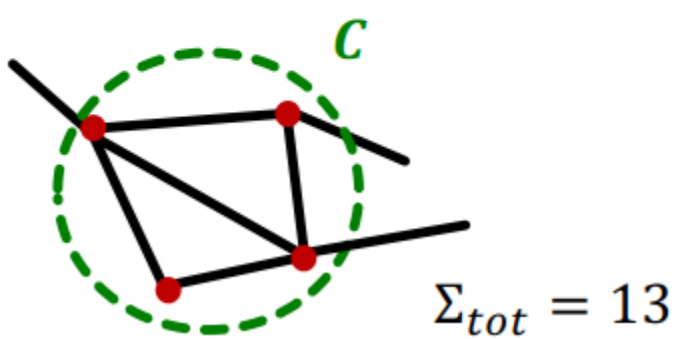
Define:

- $\Sigma_{in} \equiv \sum_{i,j \in C} A_{ij}$... sum of link weights between nodes in C
- $\Sigma_{tot} \equiv \sum_{i \in C} k_i$... sum of all link weights of nodes in C

Σ_{in}:



Σ_{tot}:

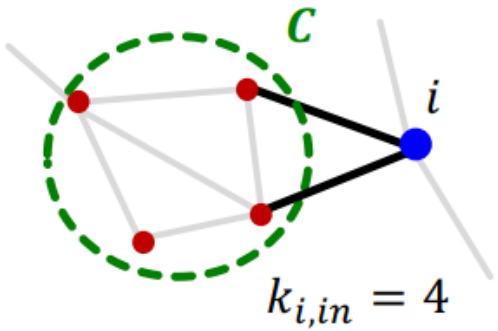


5. THUẬT TOÁN LOUVAIN

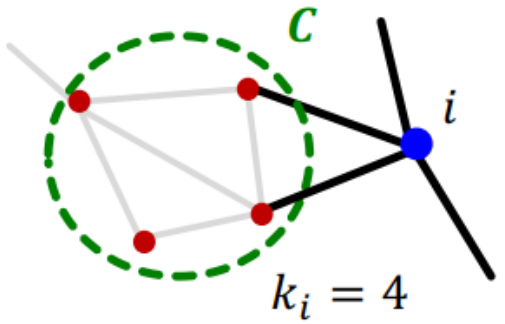
Giai đoạn 1: Phân vùng

- **Further define:**
  - $\boldsymbol{k_{i,in}} \equiv \sum_{j \in C} A_{ij} + \sum_{j \in C} A_{ji}$ … sum of link weights connecting node i and C
    - **(note that each edge gets counted twice, see formula)**
  - $\boldsymbol{k_i}$… sum of all link weights (i.e., degree) of node i

$\boldsymbol{k_{i,in}}$:



$\boldsymbol{k_i}$:



5. THUẬT TOÁN LOUVAIN

Giai đoạn 1: Phân vùng

- m: Trọng số của tất cả các cạnh trong biểu đồ (*Nếu đồ thị không có trọng số, số lượng các cạnh trong toàn bộ đồ thị*).

  - **Define:**
    - $\boldsymbol{\Sigma_{in}} \equiv \sum_{i,j \in C} A_{ij}$... sum of link weights between nodes in C
    - $\boldsymbol{\Sigma_{tot}} \equiv \sum_{i \in C} k_i$... sum of all link weights of nodes in C
  - Then, we have

$$Q(C) \equiv \frac{1}{2m} \sum_{i,j \in C} \left[A_{ij} - \frac{k_i k_j}{2m} \right] = \frac{\Sigma_{i,j \in C} A_{ij}}{2m} - \frac{(\Sigma_{i \in C} k_i)(\Sigma_{j \in C} k_j)}{(2m)^2}$$

$$= \frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m} \right)^2$$

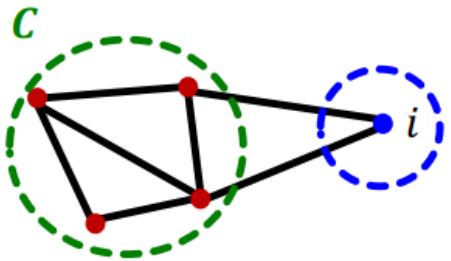
Links within the community — Total links

$Q(C)$ is large when most of the total links are within-community links

5. THUẬT TOÁN LOUVAIN

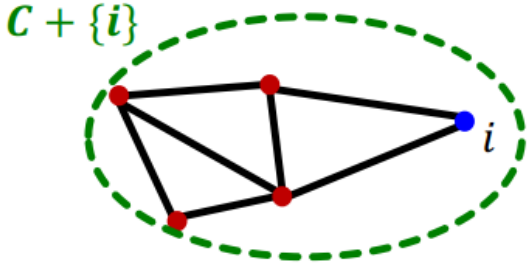
Giai đoạn 1: Phân vùng

Before merging



Isolated community of node i

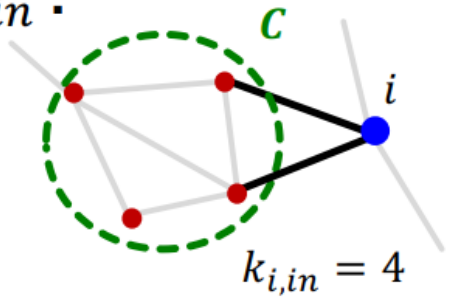
After merging



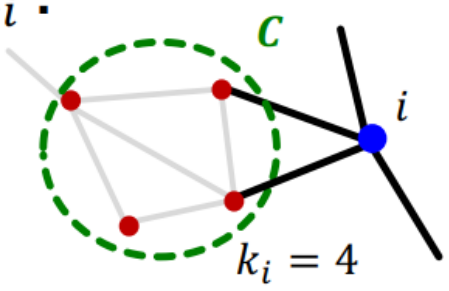
$$Q_{\text{before}} = Q(C) + Q(\{i\}) = \left[\frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2\right] + \left[0 - \left(\frac{k_i}{2m}\right)^2\right]$$

$$Q_{\text{after}} = Q(C + \{i\}) = \frac{\text{“}\Sigma_{in}\text{” of } C+\{i\}}{\boxed{\Sigma_{in} + k_{i,in}}} - \left(\frac{\text{“}\Sigma_{tot}\text{” of } C+\{i\}}{\boxed{\Sigma_{tot} + k_i}}\right)^2$$

Recall: $k_{i,in}$:



k_i :



5. THUẬT TOÁN LOUVAIN

Giai đoạn 1: Phân vùng

- **Modularity Gain**
  - $\Delta Q(i \to C) = Q_{\text{after}} - Q_{\text{before}}$

$$= \left[\frac{\Sigma_{in}+k_{i,in}}{2m} - \left(\frac{\Sigma_{tot}+k_i}{2m}\right)^2\right] - \left[\frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right]$$

  - $\Delta Q(D \to i)$ can be derived similarly.
  - **In summary, we can compute:**

$$\Delta Q(D \to i \to C) = \Delta Q(D \to i) + \Delta Q(i \to C)$$

5. THUẬT TOÁN LOUVAIN

Giai đoạn 1: Phân vùng

- Lặp lại cho đến khi không có nút nào chuyển sang một cộng đồng mới:
  - Đối với mỗi nút $i \in V$ hiện có trong cộng đồng C, hãy tính cộng đồng tốt nhất C'.
    - $C' = \text{argmax}_{C'} \Delta Q(C \to i \to C')$
    - If $\Delta Q(C \to i \to C') > 0$, then **update the community:**
      - $C \leftarrow C - \{i\}$
      - $C' \leftarrow C' + \{i\}$

5. THUẬT TOÁN LOUVAIN

Giai đoạn 2: Tái cấu trúc

Các cộng đồng thu được trong giai đoạn đầu tiên được hợp thành các siêu nút và mạng được tạo ra tương ứng:

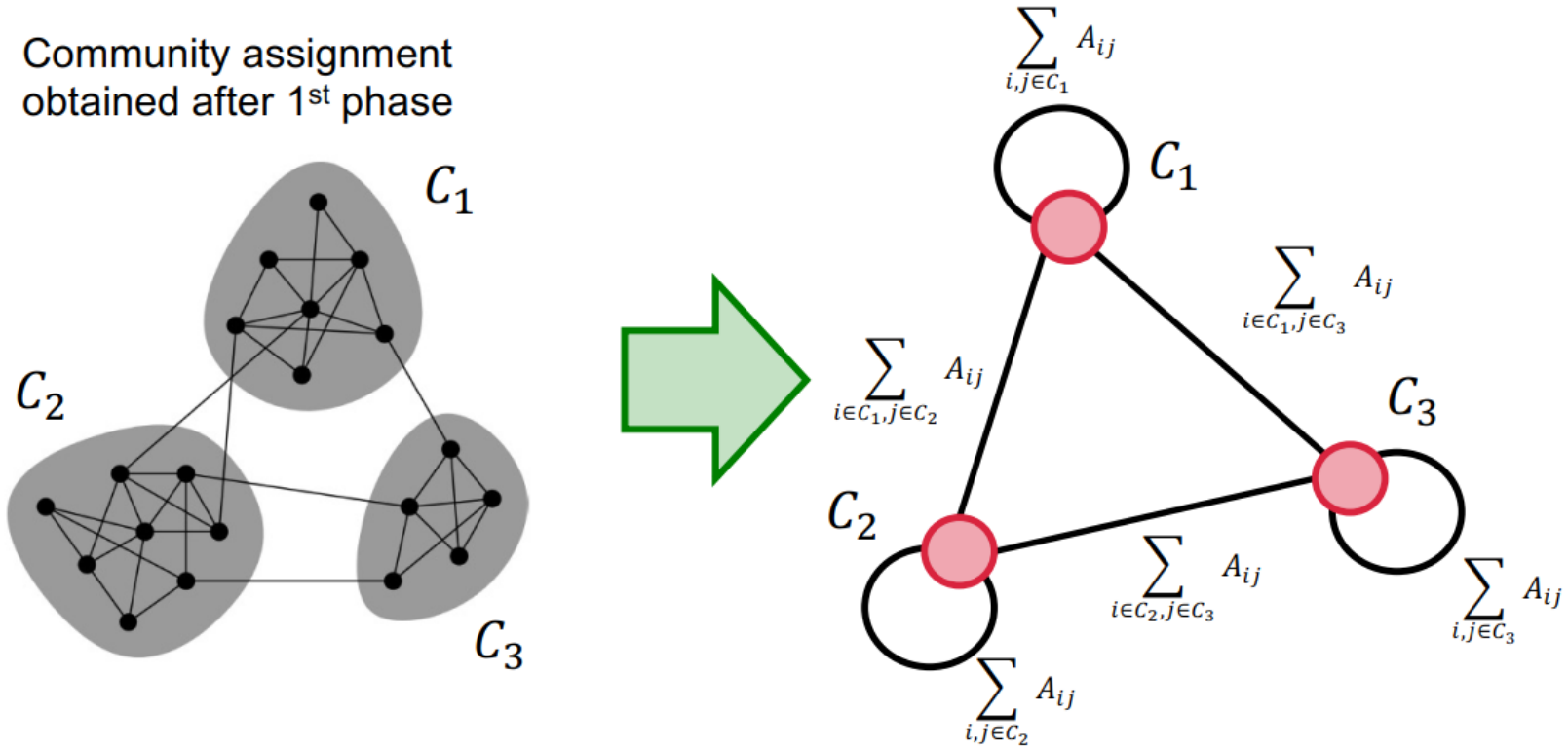
- Các siêu nút được kết nối nếu có ít nhất một cạnh giữa các nút của các cộng đồng tương ứng.
- Trọng số của cạnh giữa hai siêu nút là tổng trọng số của tất cả các cạnh giữa các cộng đồng tương ứng của chúng.

Giai đoạn 1 sau đó được chạy trên mạng siêu nút.

5. THUẬT TOÁN LOUVAIN

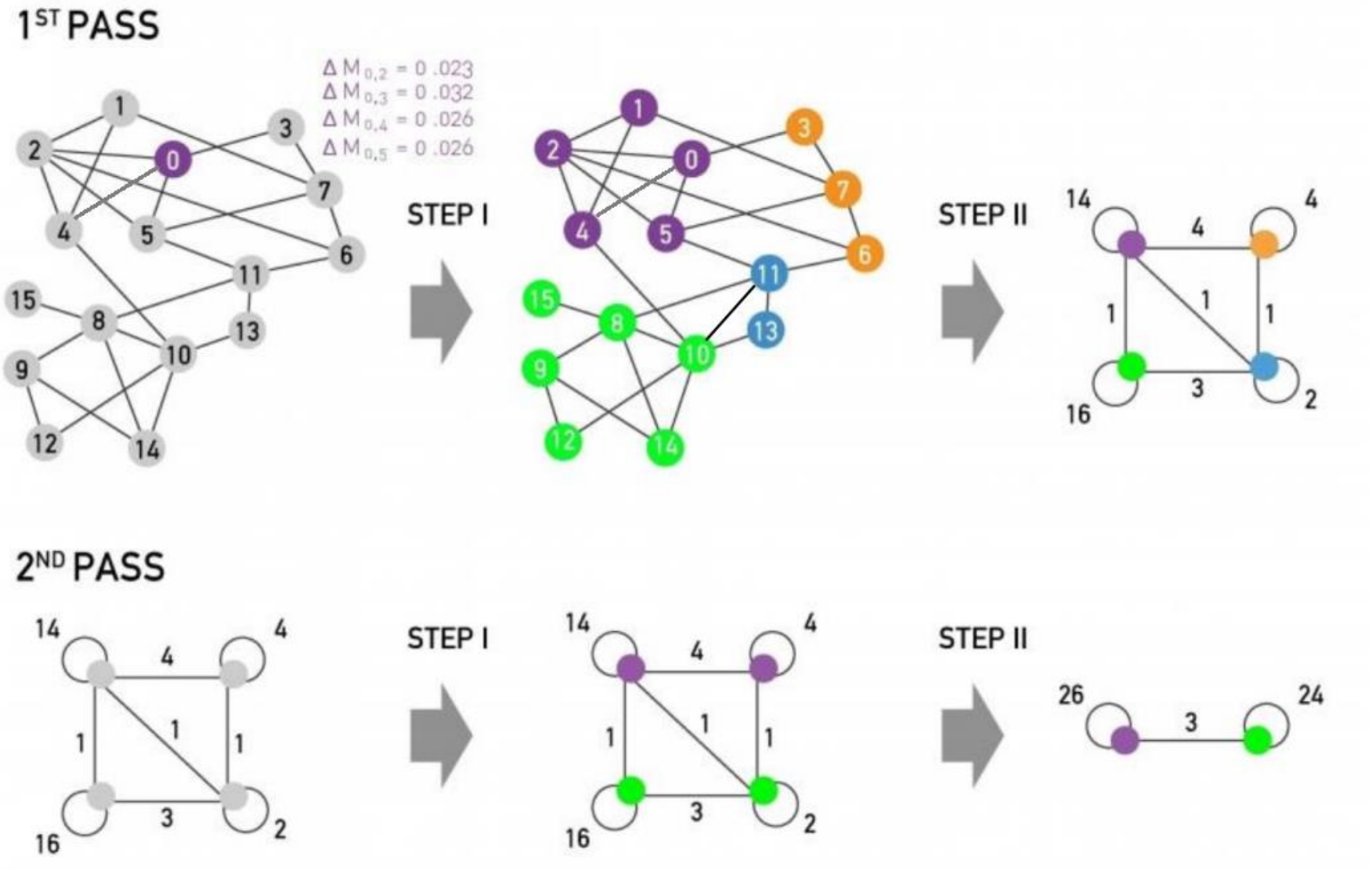
Giai đoạn 2: Tái cấu trúc

- Các siêu nút được xây dựng bằng cách hợp nhất các nút trong cùng một cộng đồng.



5. THUẬT TOÁN LOUVAIN

- **Ví dụ:**




Q & A