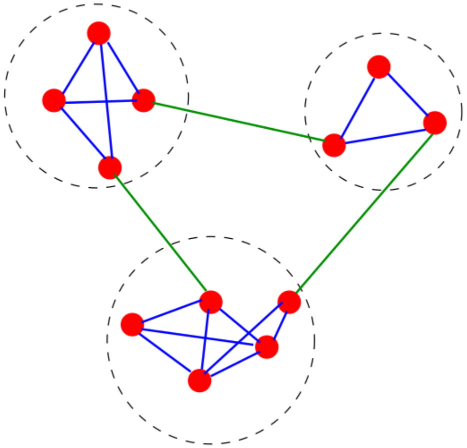
ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

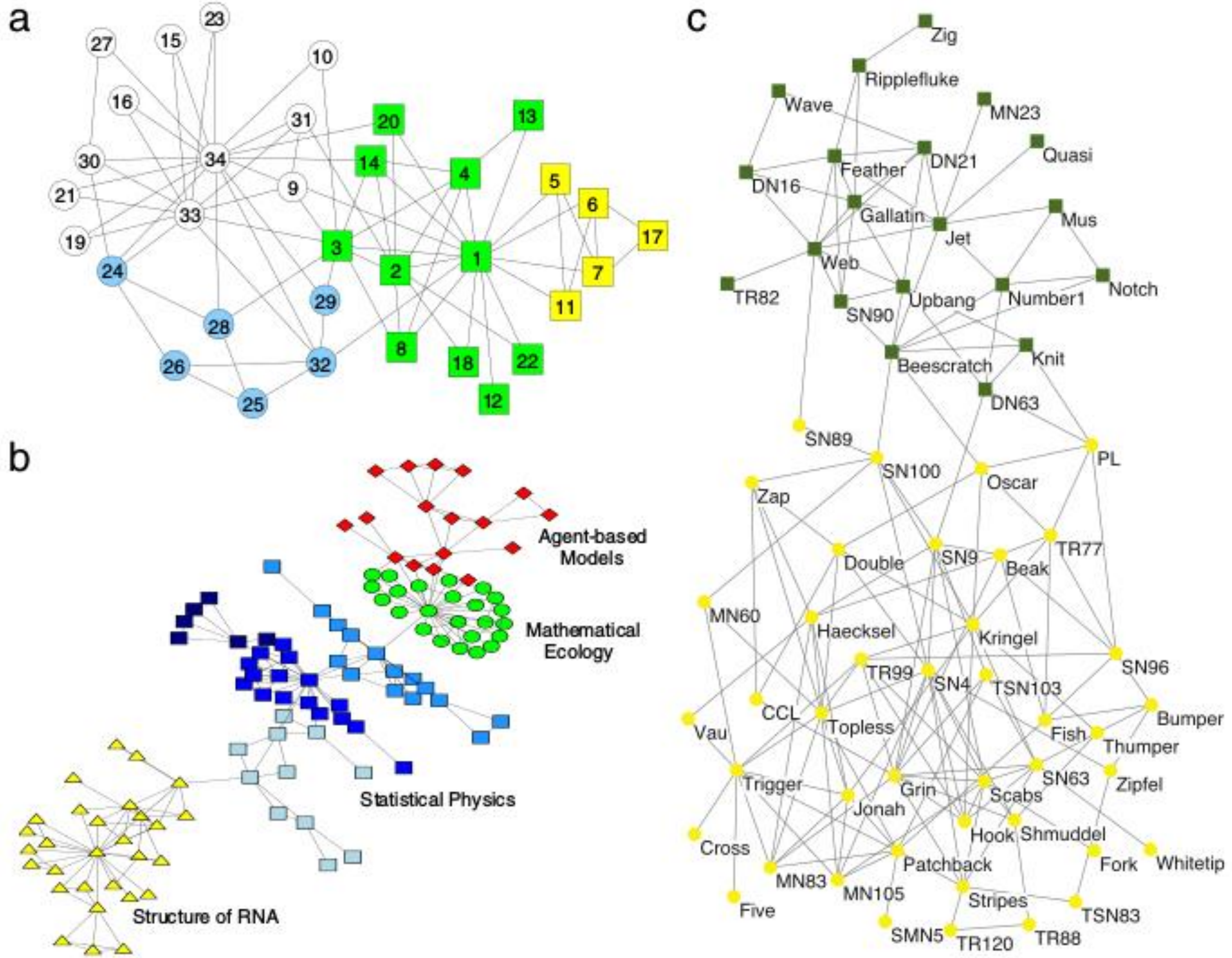
BÀI 5: PHÂN TÍCH LIÊN KẾT (TIẾP)

2. Nhận diện cộng đồng
2.1 Nhận diện cộng đồng

- Phát hiện các cộng đồng trong mạng lưới
- Các thành viên trong cộng đồng có tính chất tương tự nhau
- Các cộng đồng có thể có mối liên hệ với nhau
- Số lượng cộng đồng phụ thuộc vào thuật toán



from Fortunato (2015)

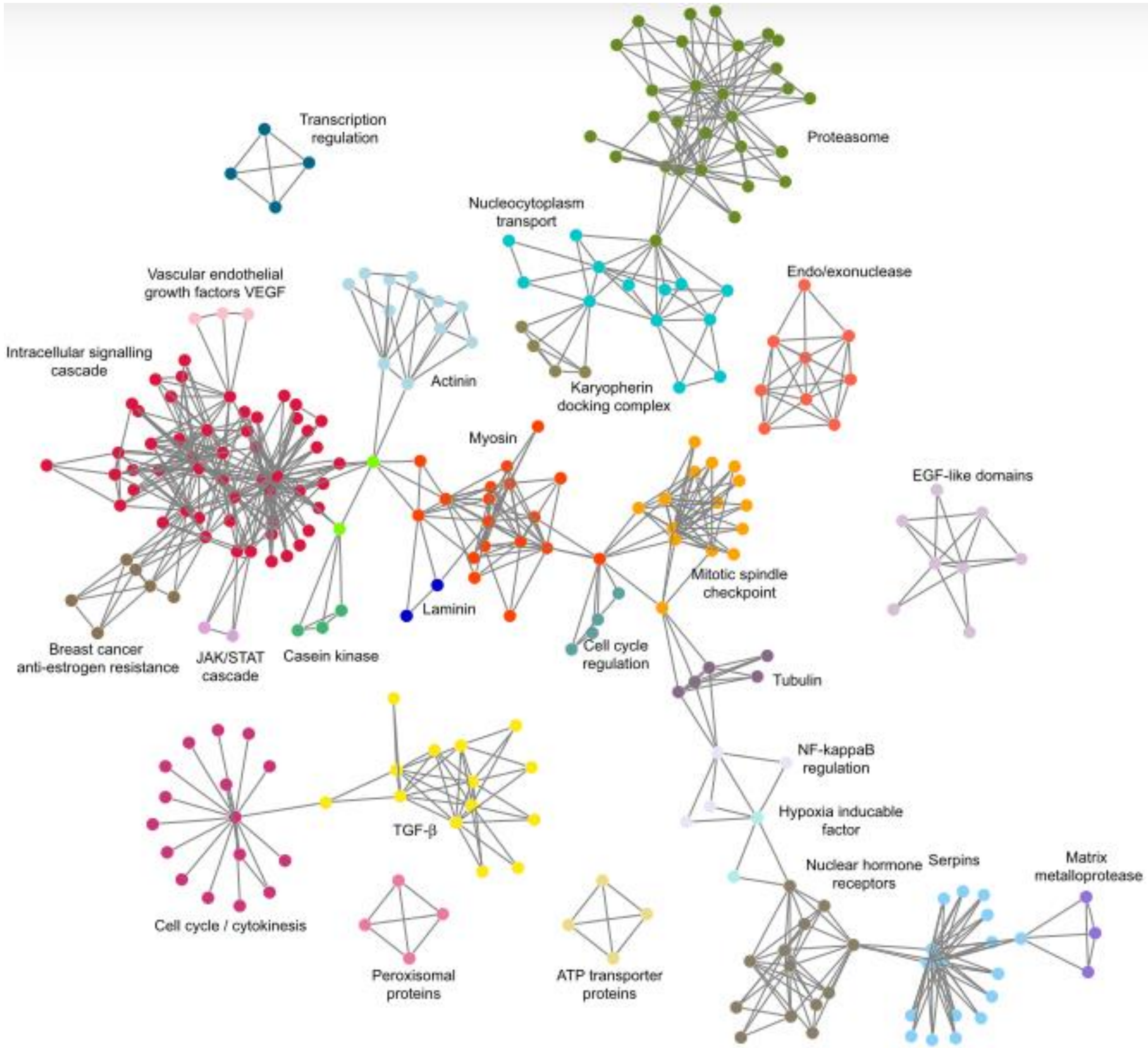


a) Zachary’s karate club

b) Collaboration network between scientists working at the Santa Fe Institute

c) Lusseau’s network of bottlenose dolphins

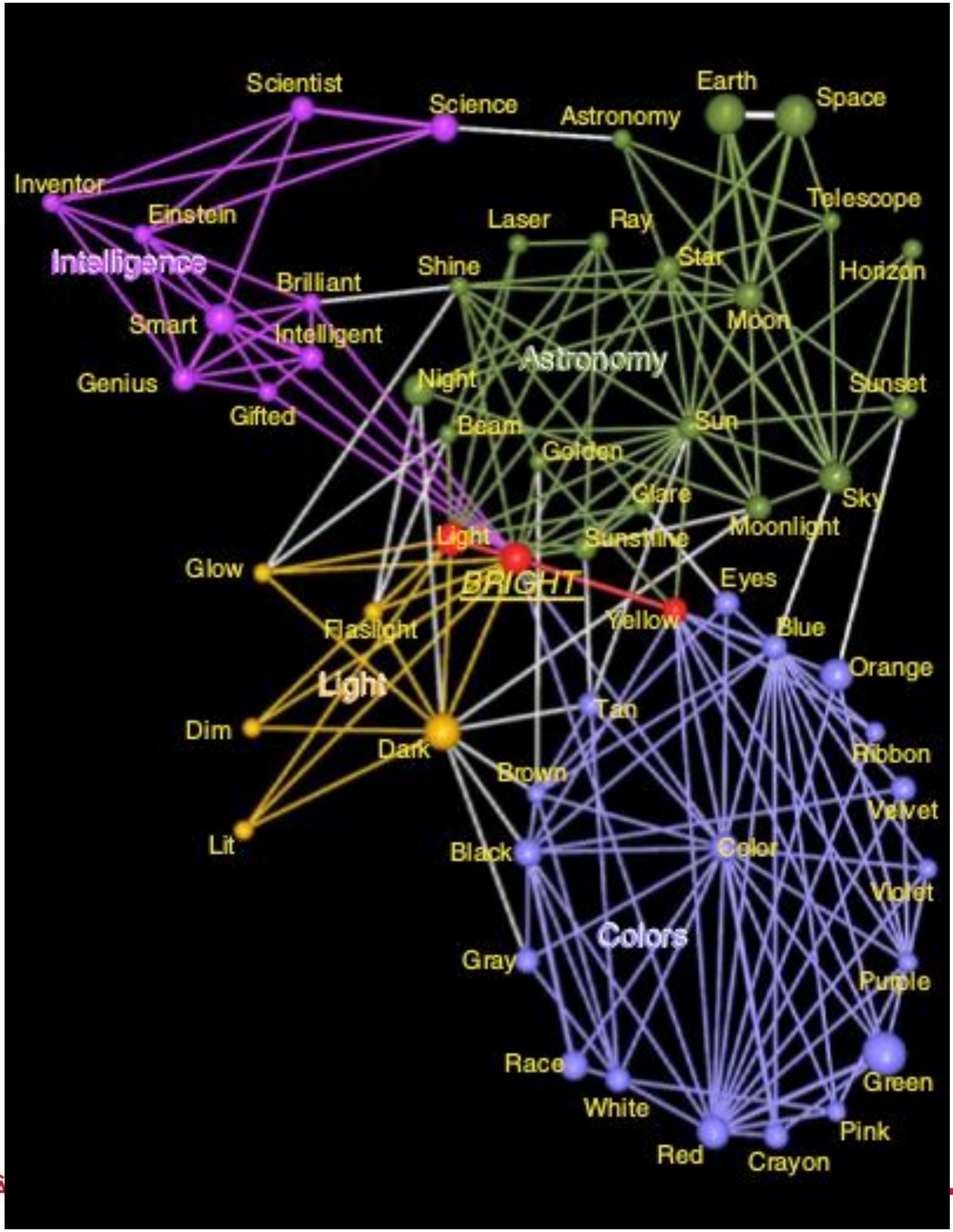
from Fortunato (2015)



Community structure in protein–protein interaction networks

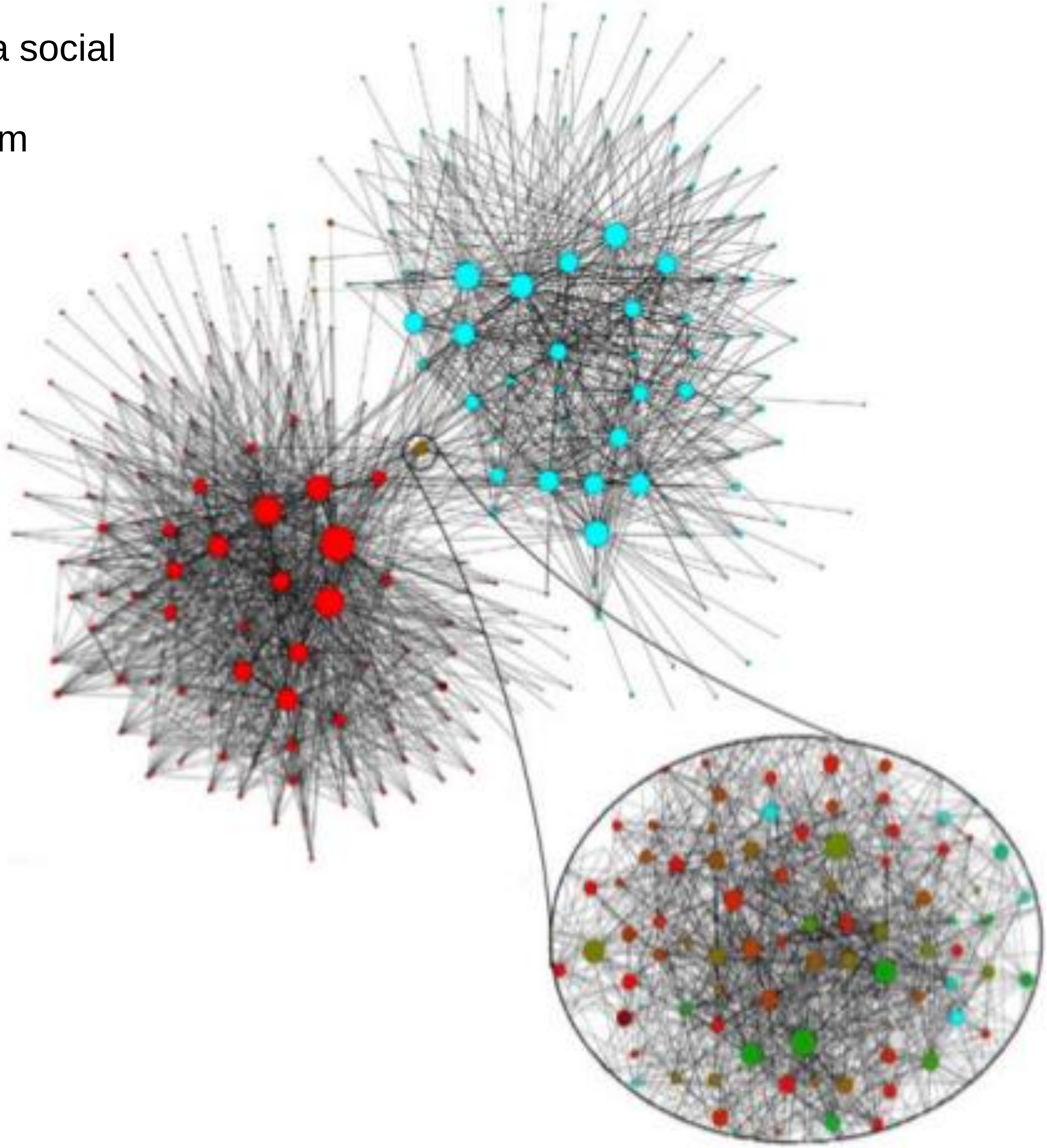
from Fortunato (2015)

Overlapping communities in a network of word association



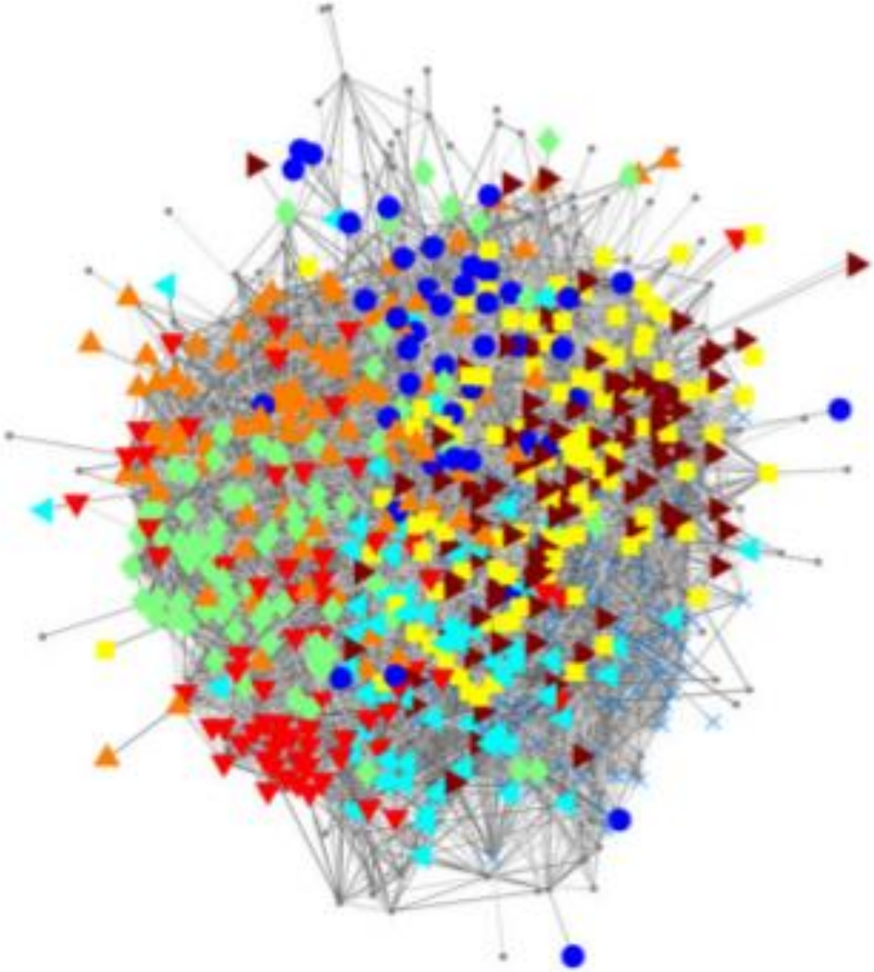
from Fortunato (2015)

Community structure of a social network of mobile phone communication in Belgium

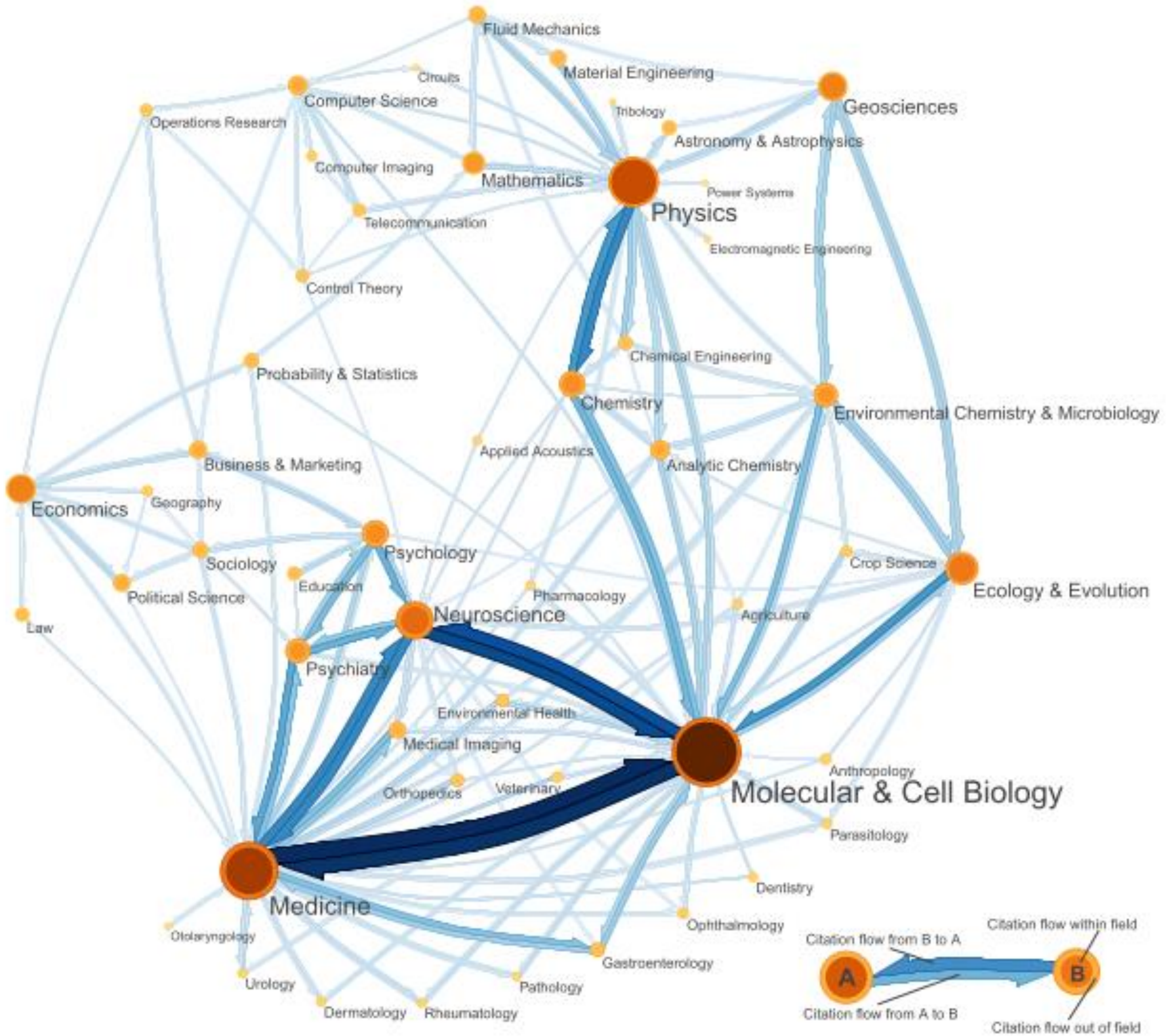


from Fortunato (2015)

Network of friendships
between students at Caltech



from Fortunato (2015)

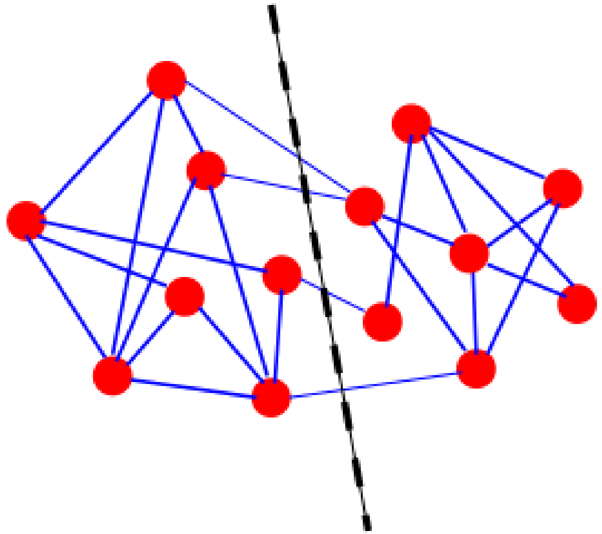


Map of science derived from a clustering analysis of a citation network

from Fortunato (2015)

2.2 Thuật toán Kernighan–Lin

Bài toán lắt cắt nhỏ nhất: Phân miền đồ thị vô hướng thành hai miền có số đỉnh tương đương sao cho tổng trọng số của các cạnh nối hai cụm là nhỏ nhất



Thuật toán

- $G = (V, E)$
- Chia các đỉnh vào hai cụm A và B không trùng lặp
- a ∈ A:

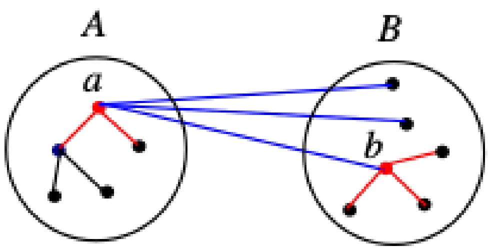
  Chi phí bên trong $I_a = \Sigma_{u \in A}\, c_{a,u}$

  Chi phí bên ngoài $E_a = \Sigma_{\in B}\, c_{a,v}$

  $D_a = E_a - I_a$
- b ∈ B, chi phí giảm nếu đổi chỗ a và b

  $T_{old} - T_{new} = D_a + D_b - 2c_{a,b}$
- Lặp lại việc tìm các cặp tối ưu (a,b) để giảm chi phí trong khi tổng chi phí (của lát cắt) tiếp tục giảm

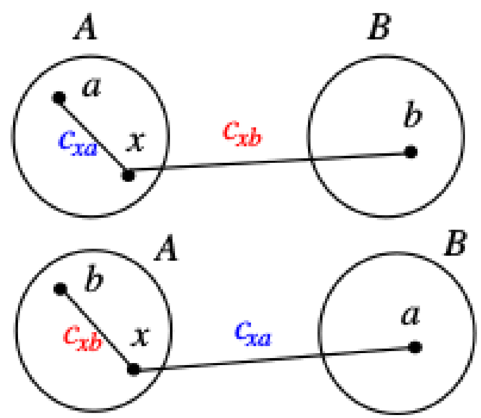
Cập nhật chi phí



$Gain_{a \Rightarrow B}: D_a - c_{ab}$

$Gain_{b \Rightarrow A}: D_b - c_{ab}$

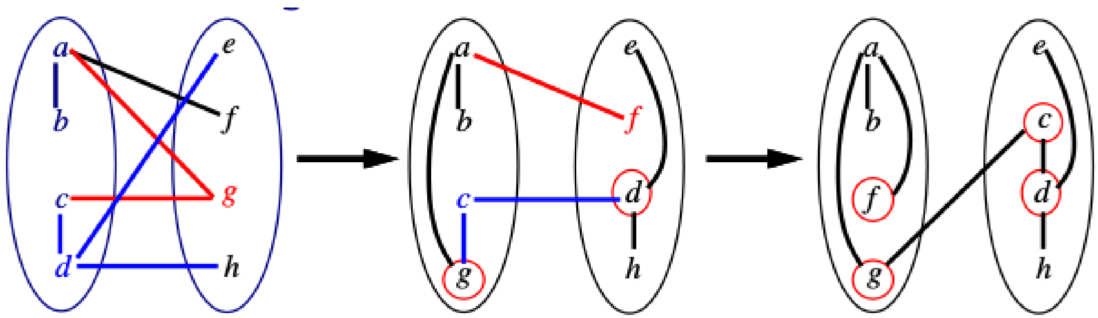
Internal cost vs. External cost



before swap	after swap	$\triangle C$
$-c_{xa}$	$+c_{xa}$	$+2c_{xa}$
$+c_{xb}$	$-c_{xb}$	$-2c_{xb}$

updating D–values

VD



Thuật toán

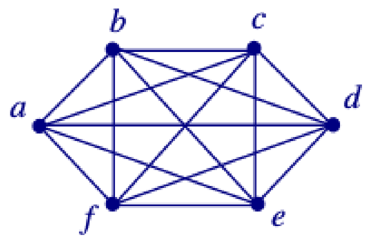
Algorithm: Kernighan-Lin(G)
Input: $G = (V, E), |V| = 2n$.
Output: Balanced bi-partition A and B with ''small'' cut cost.

begin
Bipartition G into A and B such that $|V_A| = |V_B|$, $V_A \cap V_B = \emptyset$,
  and $V_A \cup V_B = V$.
repeat
  Compute D_v, $\forall v \in V$.
  for $i = 1$ **to** n **do**
    Find a pair of unlocked vertices $v_{ai} \in V_A$ and $v_{bi} \in V_B$ whose exchange makes the largest decrease or smallest increase in cut cost;
    Mark v_{ai} and v_{bi} as locked, store the gain $\hat{g}_i$, and compute the new D_v, for all unlocked $v \in V$;
  Find k, such that $G_k = \sum_{i=1}^{k} \hat{g}_i$ is maximized;
  if $G_k > 0$ **then**
    Move $v_{a1}, \ldots, v_{ak}$ from V_A to V_B and $v_{b1}, \ldots, v_{bk}$ from V_B to V_A;
  Unlock v, $\forall v \in V$.
until $G_k \leq 0$;
end

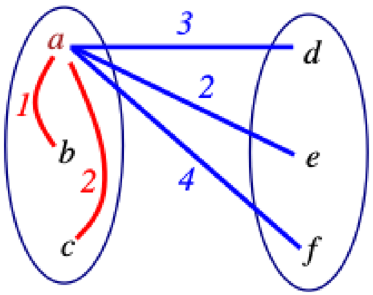
Độ phức tạp tính toán

- Khởi tạo tính toán D: O(n^2) (line 4)
- Vòng lặp: O(n) (line 5)
- Thân vòng lặp: O(n^2)
  - Bước i cần $(n - i + 1)^2$ thời gian
- Mỗi vòng lặp: O(n^3) (line 4-11)
- Giả sử thuật toán kết thúc sau r vòng lặp
- Tổng thời gian: O(rn^3)

VD



	a	b	c	d	e	f
a	0	1	2	3	2	4
b	1	0	1	4	2	1
c	2	1	0	3	2	1
d	3	4	3	0	4	3
e	2	2	2	4	0	2
f	4	1	1	3	2	0



costs associated with a

Initial cut cost = (3+2+4)+(4+2+1)+(3+2+1) = 22

- Iteration 1:

$$
\begin{array}{lll}
I_a = 1+2 = 3; & E_a = 3+2+4 = 9; & D_a = E_a - I_a = 9-3 = 6 \\
I_b = 1+1 = 2; & E_b = 4+2+1 = 7; & D_b = E_b - I_b = 7-2 = 5 \\
I_c = 2+1 = 3; & E_c = 3+2+1 = 6; & D_c = E_c - I_c = 6-3 = 3 \\
I_d = 4+3 = 7; & E_d = 3+4+3 = 10; & D_d = E_d - I_d = 10-7 = 3 \\
I_e = 4+2 = 6; & E_e = 2+2+2 = 6; & D_e = E_e - I_e = 6-6 = 0 \\
I_f = 3+2 = 5; & E_f = 4+1+1 = 6; & D_f = E_f - I_f = 6-5 = 1
\end{array}
$$

VD (tiếp)

- Iteration 1:

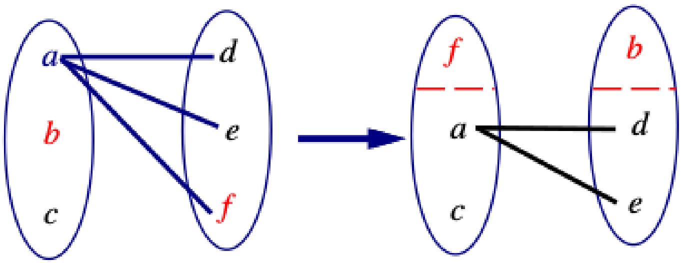
$$\begin{array}{lll} I_a = 1+2=3; & E_a = 3+2+4=9; & D_a = E_a - I_a = 9-3=6 \\ I_b = 1+1=2; & E_b = 4+2+1=7; & D_b = E_b - I_b = 7-2=5 \\ I_c = 2+1=3; & E_c = 3+2+1=6; & D_c = E_c - I_c = 6-3=3 \\ I_d = 4+3=7; & E_d = 3+4+3=10; & D_d = E_d - I_d = 10-7=3 \\ I_e = 4+2=6; & E_e = 2+2+2=6; & D_e = E_e - I_e = 6-6=0 \\ I_f = 3+2=5; & E_f = 4+1+1=6; & D_f = E_f - I_f = 6-5=1 \end{array}$$

- $g_{xy} = D_x + D_y - 2c_{xy}$.

$$\begin{array}{lll} g_{ad} & = & D_a + D_d - 2c_{ad} = 6+3-2\times 3 = 3 \\ g_{ae} & = & 6+0-2\times 2 = 2 \\ g_{af} & = & 6+1-2\times 4 = -1 \\ g_{bd} & = & 5+3-2\times 4 = 0 \\ g_{be} & = & 5+0-2\times 2 = 1 \\ g_{bf} & = & 5+1-2\times 1 = 4 \ (maximum) \\ g_{cd} & = & 3+3-2\times 3 = 0 \\ g_{ce} & = & 3+0-2\times 2 = -1 \\ g_{cf} & = & 3+1-2\times 1 = 2 \end{array}$$

- Swap b and f! ($\hat{g}_1 = 4$)

VD (tiếp)



- $D'_x = D_x + 2c_{xp} - 2c_{xq}, \forall x \in A - \{p\}$ (swap p and q, $p \in A$, $q \in B$)

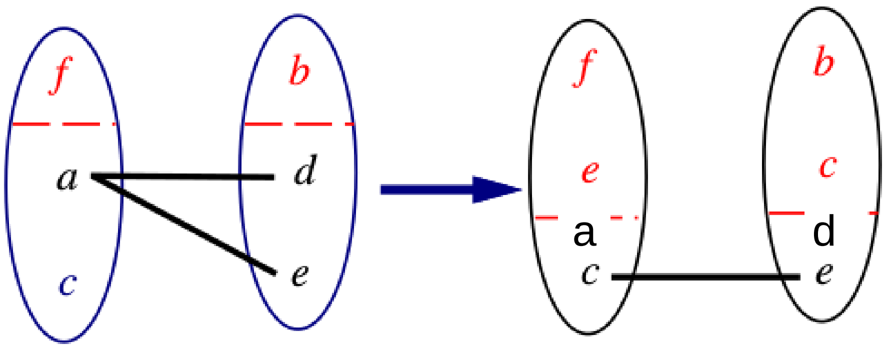
$$\begin{aligned} D'_a &= D_a + 2c_{ab} - 2c_{af} = 6 + 2 \times 1 - 2 \times 4 = 0 \\ D'_c &= D_c + 2c_{cb} - 2c_{cf} = 3 + 2 \times 1 - 2 \times 1 = 3 \\ D'_d &= D_d + 2c_{df} - 2c_{db} = 3 + 2 \times 3 - 2 \times 4 = 1 \\ D'_e &= D_e + 2c_{ef} - 2c_{eb} = 0 + 2 \times 2 - 2 \times 2 = 0 \end{aligned}$$

- $g_{xy} = D'_x + D'_y - 2c_{xy}$.

$$\begin{aligned} g_{ad} &= D'_a + D'_d - 2c_{ad} = 0 + 1 - 2 \times 3 = -5 \\ g_{ae} &= D'_a + D'_e - 2c_{ae} = 0 + 0 - 2 \times 2 = -4 \\ g_{cd} &= D'_c + D'_d - 2c_{cd} = 3 + 1 - 2 \times 3 = -2 \\ g_{ce} &= D'_c + D'_e - 2c_{ce} = 3 + 0 - 2 \times 2 = -1 \ (maximum) \end{aligned}$$

- Swap c and e! ($\widehat{g_2} = -1$)

VD (tiếp)



- $D''_x = D'_x + 2c_{xp} - 2c_{xq}, \forall x \in A - \{p\}$

$$\begin{aligned} D''_a &= D'_a + 2c_{ac} - 2c_{ae} = 0 + 2 \times 2 - 2 \times 2 = 0 \\ D''_d &= D'_d + 2c_{de} - 2c_{dc} = 1 + 2 \times 4 - 2 \times 3 = 3 \end{aligned}$$

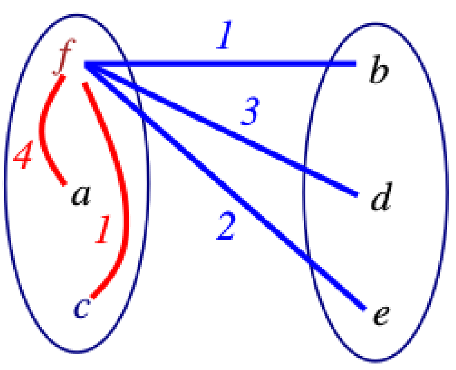
- $g_{xy} = D''_x + D''_y - 2c_{xy}$.

$$g_{ad} = D''_a + D''_d - 2c_{ad} = 0 + 3 - 2 \times 3 = -3 (\hat{g}_3 = -3)$$

- Note that this step is redundant ($\sum_{i=1}^{n} \hat{g}_i = 0$).
- Summary: $\hat{g}_1 = g_{bf} = 4$, $\hat{g}_2 = g_{ce} = -1$, $\hat{g}_3 = g_{ad} = -3$.
- Largest partial sum $\max \sum_{i=1}^{k} \hat{g}_i = 4$ $(k = 1) \Rightarrow$ Swap b and f.

VD (tiếp)

	a	b	c	d	e	f
a	0	1	2	3	2	4
b	1	0	1	4	2	1
c	2	1	0	3	2	1
d	3	4	3	0	4	3
e	2	2	2	4	0	2
f	4	1	1	3	2	0



Initial cut cost = (1+3+2)+(1+3+2)+(1+3+2) = 18 (22−4)

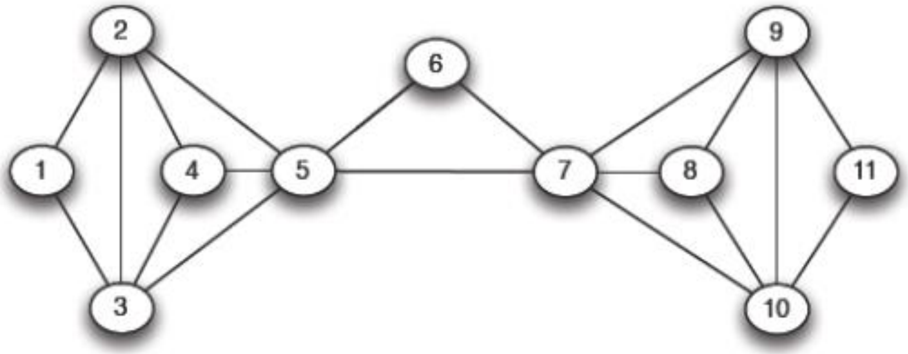
- Iteration 2: Repeat what we did at Iteration 1 (Initial cost$= 22 - 4 = 18$).
- Summary: $\hat{g_1} = g_{ce} = -1$, $\hat{g_2} = g_{ab} = -3$, $\hat{g_3} = g_{fd} = 4$.
- Largest partial sum $=$ max $\sum_{i=1}^{k} \hat{g_i} = 0$ $(k = 3) \Rightarrow$ Stop!

2.3 Thuật toán Girvan-Newman

- Tìm các cầu nối giữa các cộng đồng dựa trên khả năng thông qua
- Lặp lại với các cộng đồng để tìm các cộng đồng con
- Kết quả là các cây phân cấp với gốc là toàn bộ đồ thị, lá là các đỉnh của đồ thị

Cầu

- Cầu: kết nối các cộng đồng trong đồ thị
- Nếu giữa hai đỉnh có k đường đi ngắn nhất, mỗi đường đi tương ứng với 1/*k* đơn vị luồng
- VD: 1 đến 5 có hai đường đi ngắn nhất tương ứng với ½ đơn vị luồng



Khả năng thông qua của cạnh

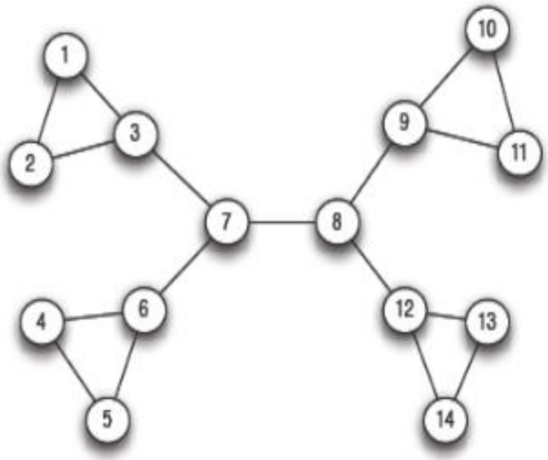
- Khả năng thông qua của cạnh *i* = số luồng tải bởi cạnh *i*
- VD:

  7-8: 49

  3-7: 33

  1-3: 12

  1-2: 1



Thuật toán

Thuật toán:

1) Tính khả năng thông qua của tất cả các cạnh

2) Xóa các cạnh có khả năng thông qua cao nhất

3) Tính lại khả năng thông qua của các cạnh bị ảnh hưởng

4) Quay lại 2), dừng khi không còn cạnh nào

Tính khả năng thông qua

- Tính khả năng thông qua dựa trên BFS
- Với mỗi đỉnh *u*

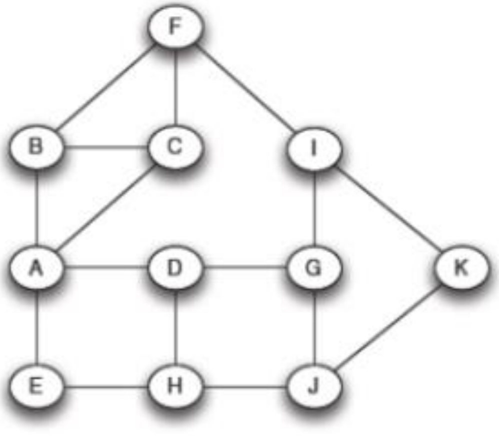
  1) Thực hiện BFS từ *u*
  2) Tính số đường đi ngắn nhất từ *u* tới mỗi đỉnh còn lại trong đồ thị
  3) Dựa trên đó tính tổng số luồng từ *u* tới mỗi đỉnh còn lại

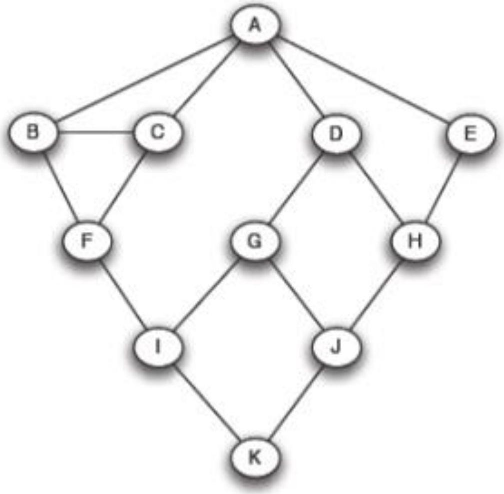
Tính khả năng thông qua (tiếp)

- Áp dụng BFS cho mỗi đỉnh
- Dựa trên đó tính khả năng thông qua cho mỗi cạnh
- Chia đôi kết quả (do mỗi cạnh tính hai lần)

VD

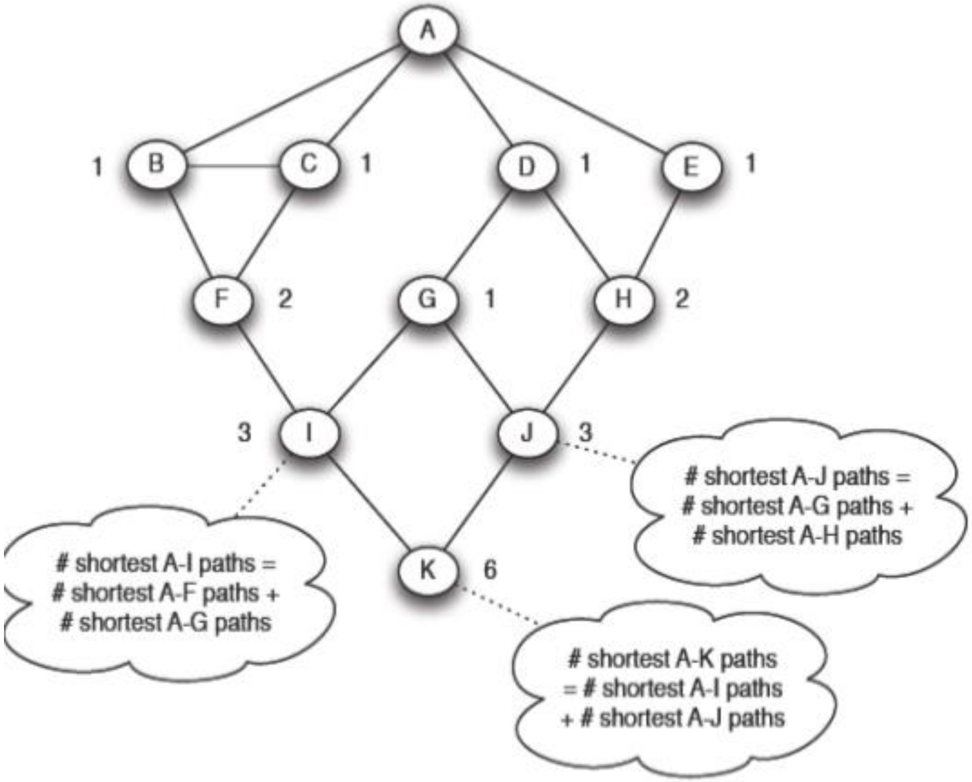
- Bước 1:





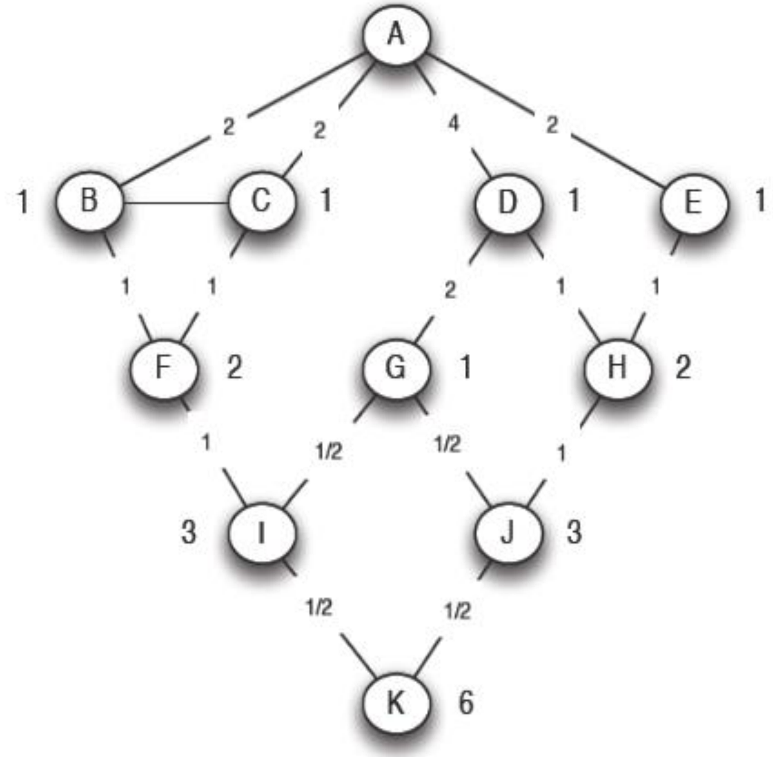
VD (tiếp)

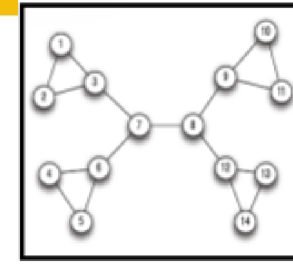
- Bước 2:



VD (tiếp)

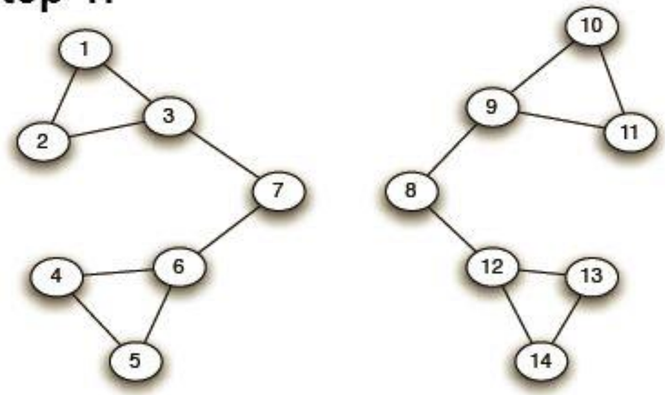
- Bước 3:



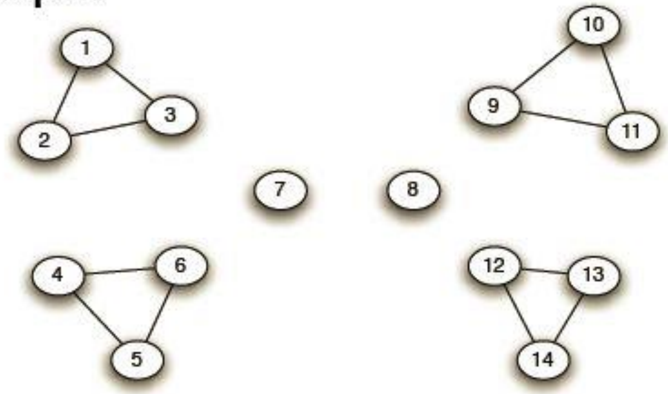


Girvan-Newman: Example

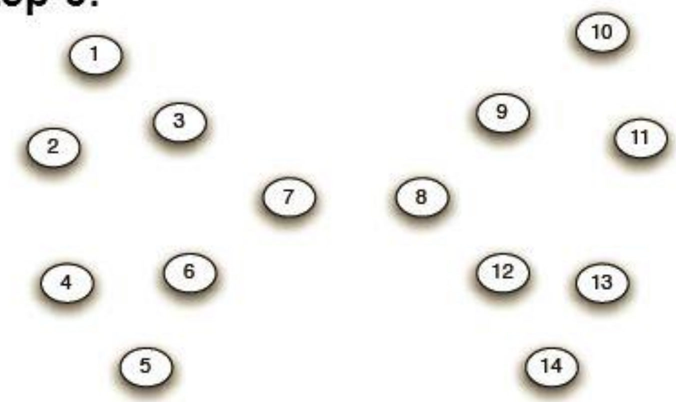
Step 1:



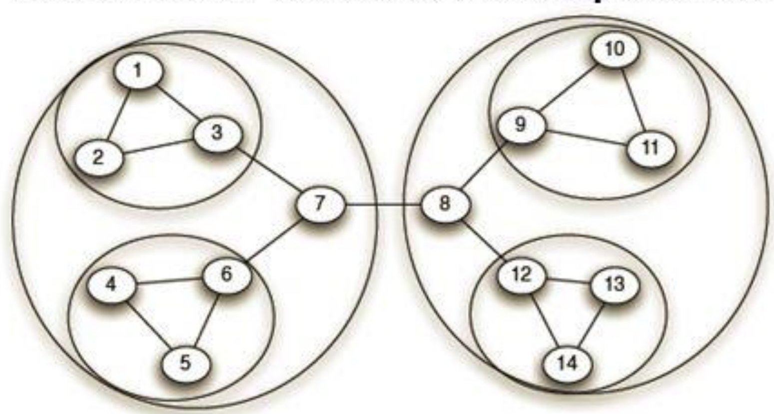
Step 2:



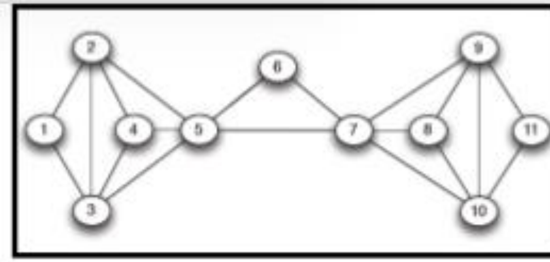
Step 3:

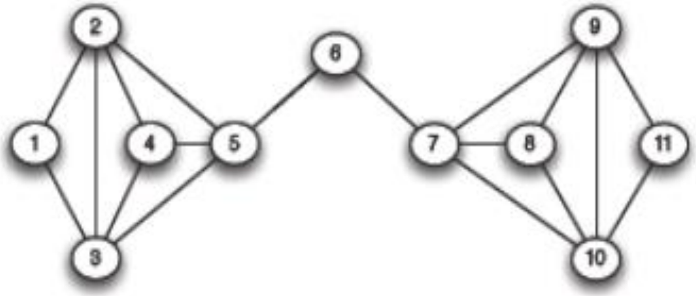


Hierarchical network decomposition:

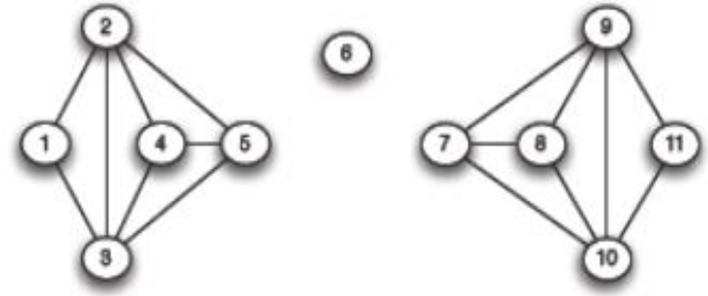


Example 2

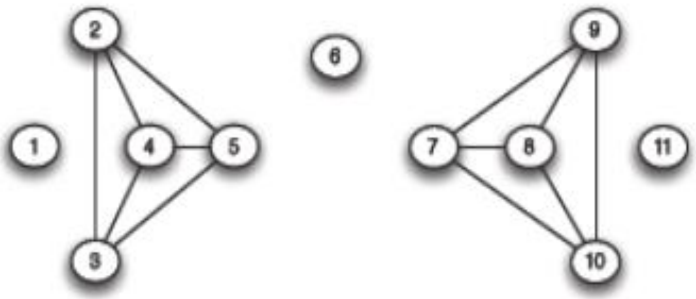




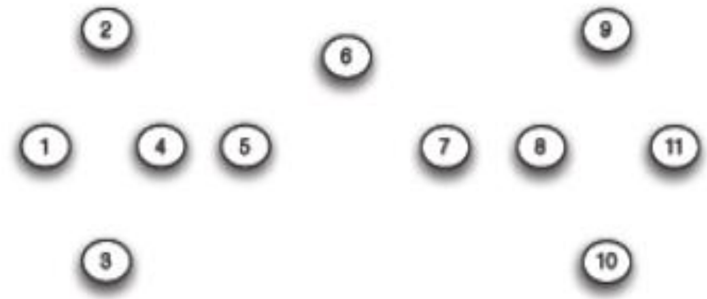
(a) *Step 1*



(b) *Step 2*



(c) *Step 3*

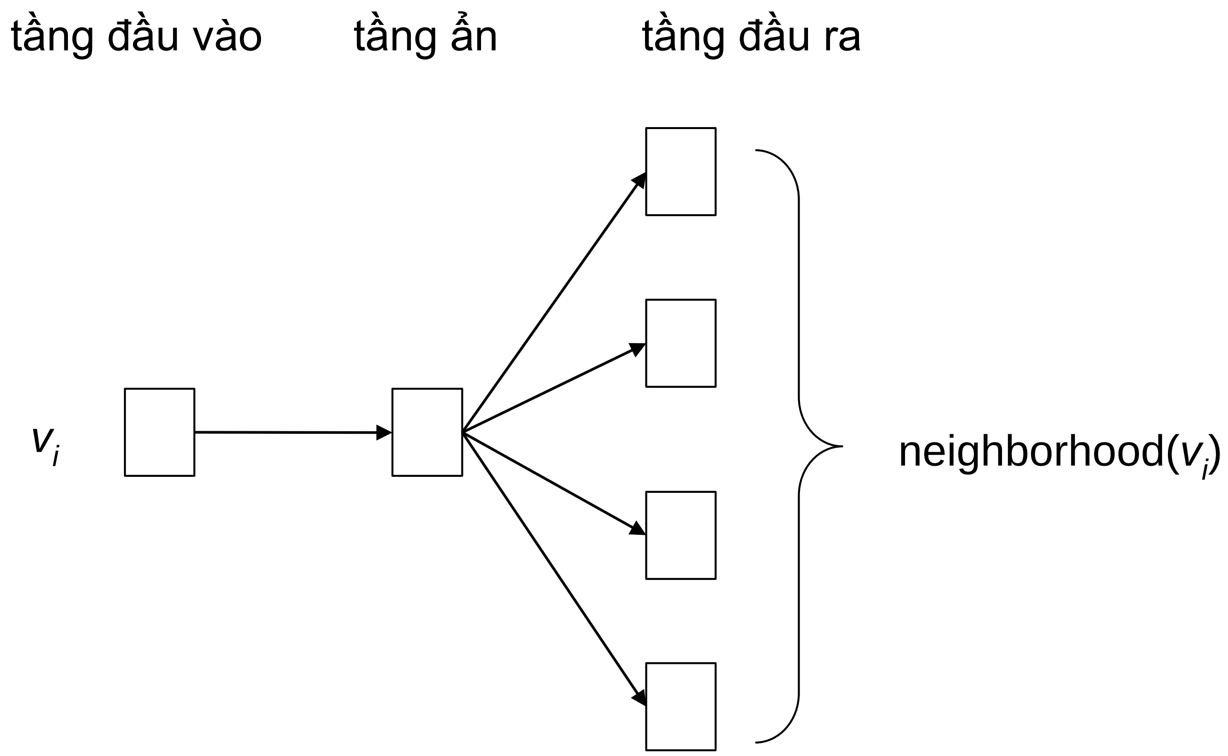


(d) *Step 4*

3. Học biểu diễn đồ thị

- Ma trận kề thường thưa, có số chiều lớn
- Cần học ra biểu diễn của các nút với số chiều thấp
- Ứng dụng vào các bài toán khác trong phân tích đồ thị, đặc biệt là các bài toán dự đoán và phân loại

node2vec



MÔ HÌNH SKIP-GRAM

Tầng đầu vào

- Biểu diễn các nút dưới dạng one-hot
  - Giá trị 1 ứng với nút hiện tại, giá trị 0 ở các vị trí khác
  - Số chiều *V* - số nút trong đồ thị

Tầng ẩn

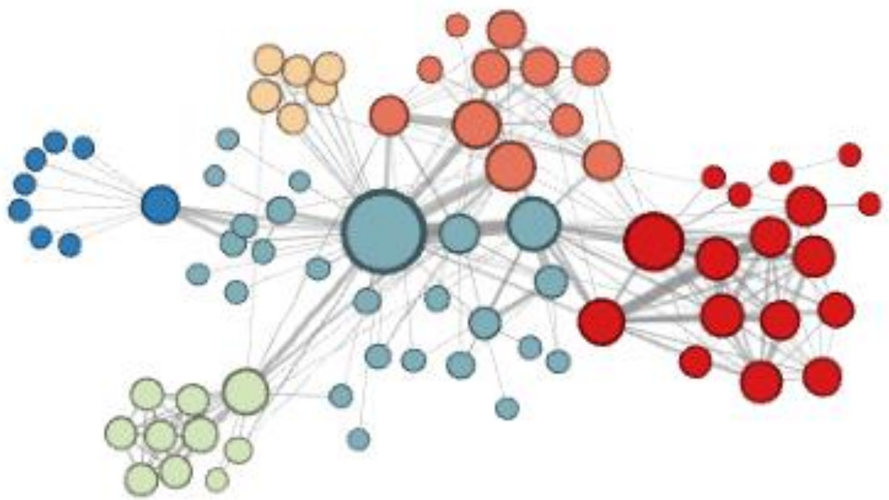
- Có số chiều *K*
- Số liên kết giữa tầng đầu vào và tầng ẩn *V* x *K*
- Trọng số liên kết giữa tầng đầu vào và tầng ẩn được dùng làm biểu diễn "học trước" của nút và được tinh chỉnh trong các tác vụ (có giám sát) khác

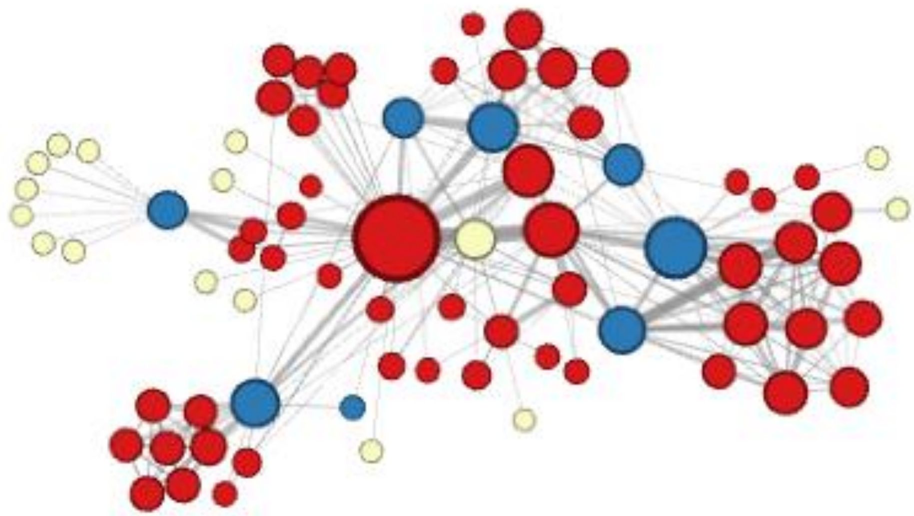
Tầng đầu ra

- Có số chiều *V* - số lượng nút trong đồ thị
- Mô hình skip-gram dùng nút hiện tại v_i để dự đoán ra các nút hàng xóm neighborhood(v_i)
- Hàm kích hoạt *softmax*
- Hàm lỗi *log-likelihood*

Neighborhood(v_i)

- BFS:
  - Lấy mẫu từ các nút liền kề với nút v_i
  - Các nút trong cùng một cộng đồng có biểu diễn tương tự nhau
- DFS:
  - Lấy mẫu trong quá trình duyệt theo chiều sâu
  - Các nút có vai trò giống nhau trong đồ thị có biểu diễn tương tự nhau (nút lá, nút trung tâm, nút cầu nối)
- Random walk: Cân bằng giữa BFS và DFS
- Lấy mẫu với số lượng k ($k = 3$)





from Aditya Grover and Jure Leskovec. “*node2vec: Scalable Feature Learning for Networks*”. KDD2016

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attentions!