

# Redes Complexas

## Aula 16

### **Aula passada**

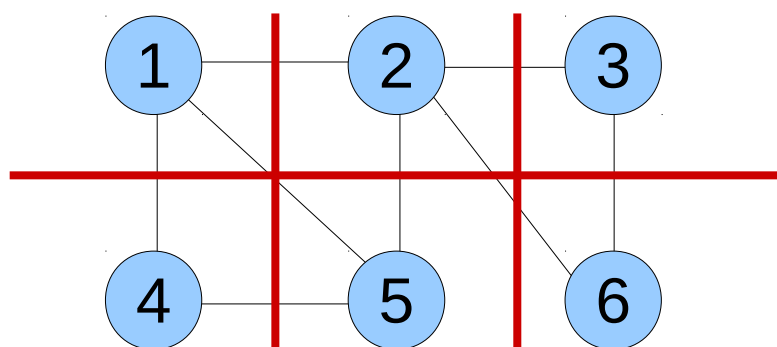
- Resilience (“robustez”)
- Tipo de falhas
- Influência da estrutura
- Análise do ponto crítico

### **Aula de hoje**

- Partição e bisseção em grafos
- Modularidade
- Algoritmos de Newman e Louvain
- Limitações

# Partição de Grafos

- **Problema:** Dividir um grafo em partes
  - partição do conjunto de vértices



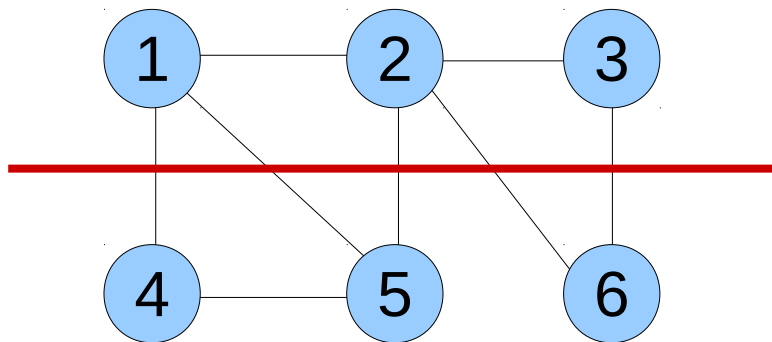
- Problema antigo, difícil e com muitas aplicações!
- Qual partição ao lado é melhor?

**Como definir uma boa partição?**

- Não há bala de prata, depende da aplicação

# Corte em Grafos

- **corte:** induz uma partição em duas partes
  - subconjuntos A e B de vértices

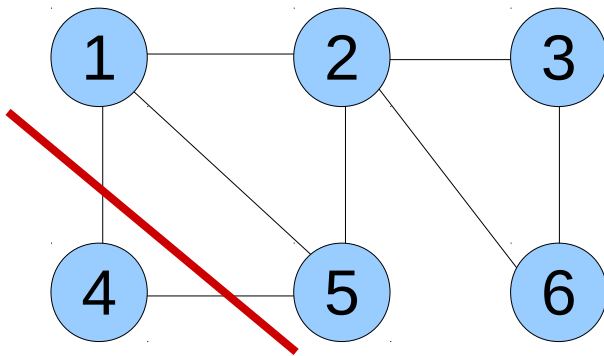


- $A = \{1, 2, 3\}$  ,  $B = \{4, 5, 6\}$

- $c(A, B)$  = custo do corte  $(A, B)$  = número de arestas entre vértices de A e B
  - soma dos pesos das arestas, em caso com peso
  - ex.  $c(A, B) = 5$

# Corte em Grafos

- **Problema:** encontrar o corte de custo mínimo de um grafo qualquer
  - problema clássico da computação, temos algoritmo polinomial



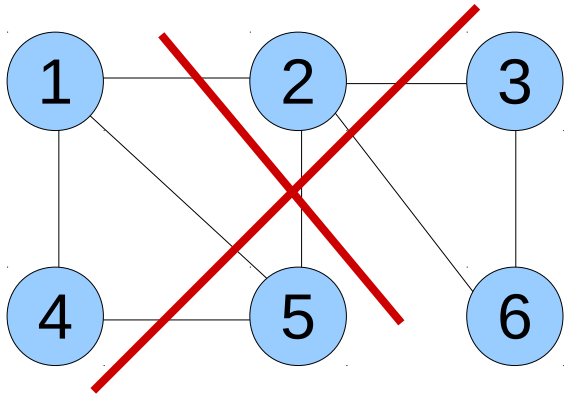
- $A = \{1, 2, 3, 5, 6\}$  ,  $B = \{4\}$

- $c(A, B) = 2$

- **Problema:** tamanho das duas partições encontradas não é considerado
  - em muitas aplicações, isto é importante

# Bisseção em Grafos

- **bisseção:** corte com duas partições do mesmo tamanho (ou 1 de diferença)
- restrição na definição de corte

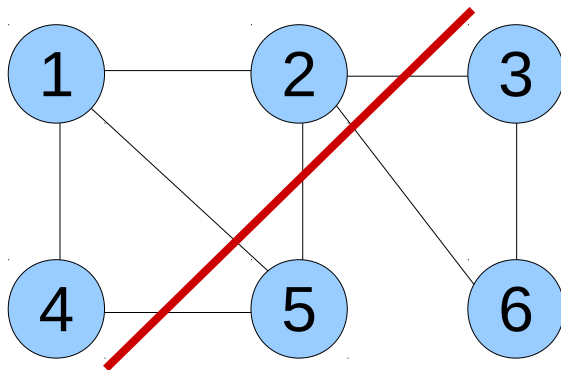


- $A = \{1, 2, 4\}$  ,  $B = \{3, 5, 6\}$
- $c(A, B) = 5$

- **Problema:** encontrar a bisseção de custo mínimo em um grafo qualquer
- ex.  $A = \{1, 4, 5\}$  ,  $B = \{2, 3, 6\}$  ,  $c(A, B) = 2$

# Bisseção em Grafos

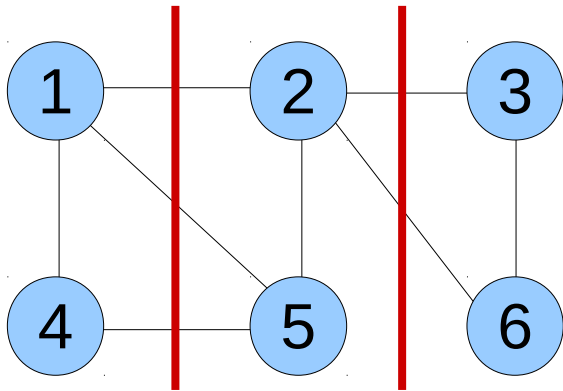
- Problema de bisseção é NP-Completo (não temos algoritmo polinomial)
  - exceções para classes de grafos, ex. árvores
- Algoritmo de Kernighan-Lin (1970)
  - algoritmo guloso (heurística) que troca pares de vértices até não melhorar mais
  - a cada passo, escolhe par que ao trocar de lado mais reduz custo do corte



- $A = \{1, 2, 4\}$ ,  $B = \{3, 5, 6\}$ ,  $c(A, B) = 5$
- Trocar 2 e 5
- $A = \{1, 4, 5\}$ ,  $B = \{2, 3, 6\}$ ,  $c(A, B) = 2$

# Partição Balanceada em Grafos

- Generalização da bisseção em grafos
- $k$  partes de tamanho igual (ou 1 de diferença)
  - $V_1, V_2, \dots, V_k$ , interseção é vazia, união é  $V$
- Custo da partição é o custo somado para todos os pares  $(V_i, V_j)$

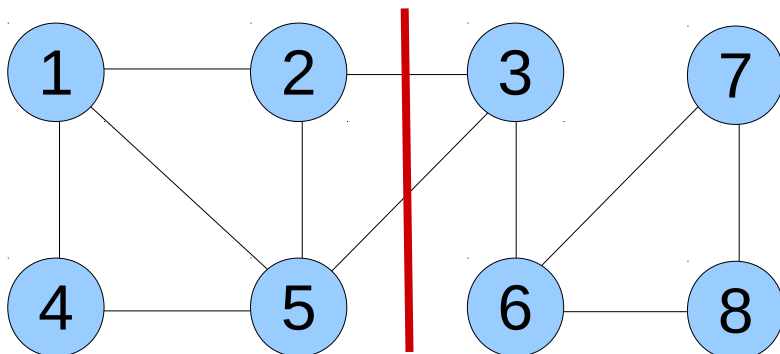


■  $k=3, V_1=\{1,4\}, V_2=\{2,5\}, V_3=\{3,6\}$

■  $\text{Custo} = c(V_1, V_2) + c(V_1, V_3) + c(V_2, V_3) = 3 + 0 + 2 = 5$

# Partição Balanceada em Grafos

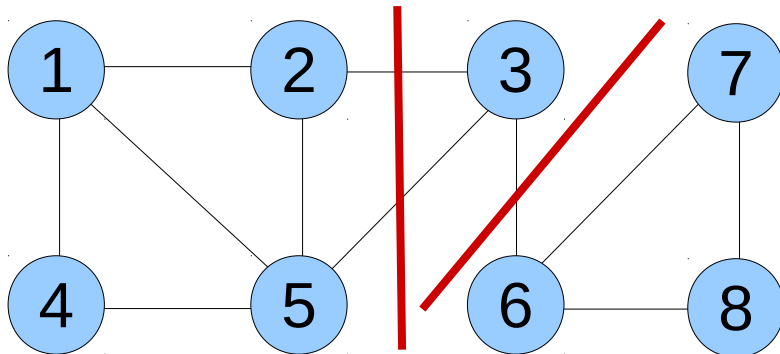
- **Problema 1:** número de partições tem que ser especificado
- **Problema 2:** tamanho das partições tem que ser igual (ou diferir em um)
- **Problema 3:** estrutura interna das partições não é considerado
  - podem não ser conexas, e ter baixa densidade



- Em muitas aplicações, isto é importante
- *Network Science* x Teoria dos grafos!

# Medindo Qualidade de Partição

- Como melhor aferir a qualidade de uma partição?
- **Ideia:** usar tamanho das partições para determinar “potencial” de arestas
  - normalização, se assim preferir
- $\text{Ratio-Cut}(A,B) = c(A,B) / |A||B|$ 
  - onde  $|A|$  é o número de vértices em A



- $\text{rc}(A,B) = 2 / (4*4) = 1/8$

- $\text{rc}(A,B) = 1 / (5*3) = 1/15$

# Partição com Ratio Cut

- **Problema:** encontrar partição (duas partes) que minimiza ratio cut
  - problema é NP-Completo
  - algoritmo aproximativo de Arora, Rao & Vazirani (2009), com fator  $O(\sqrt{\log n})$
- Ratio cut tem conexões com decomposição spectral do Laplaciano do grafo
  - método para encontrar corte, limitante do ratio cut mínimo
- Ratio Cut ainda não resolve problemas
  - número de partições
  - estrutura interna das partições

# Modularidade

- **Outra ideia:** considerar modelo aleatório para criação de arestas, aferir arestas internas
- Modularidade: fração de arestas dentro de cada partição menos o numero médio de arestas dentro de cada partição no modelo aleatório
- K partições, n vértices, m arestas

$$M = \frac{1}{2m} \sum_k \sum_{i,j \in V_k} (A_{ij} - p_{ij})$$

prob. de i e j serem vizinhos

constante de normalização

- Assumindo configuration model, com graus dos vértices da rede

$$M = \frac{1}{2m} \sum_k \sum_{i,j \in V_k} \left( A_{ij} - \frac{d_i d_j}{2m} \right)$$

# Modularidade

- Simplificando, temos

$$M = \sum_k \left( \frac{m_k}{m} - \left( \frac{d_k}{2m} \right)^2 \right)$$

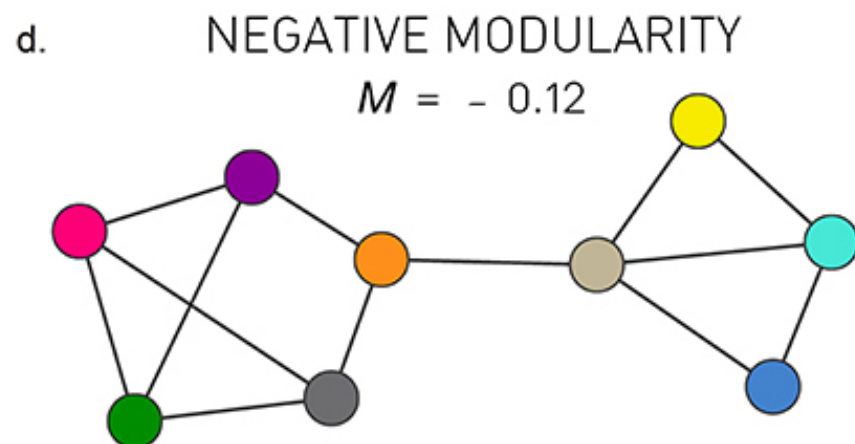
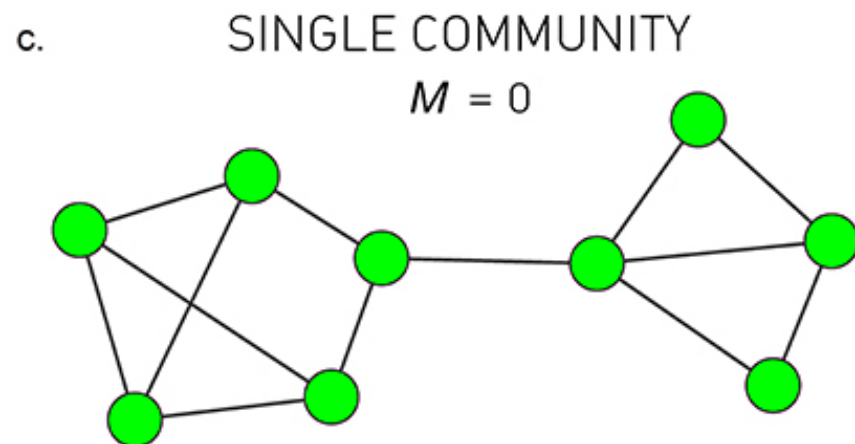
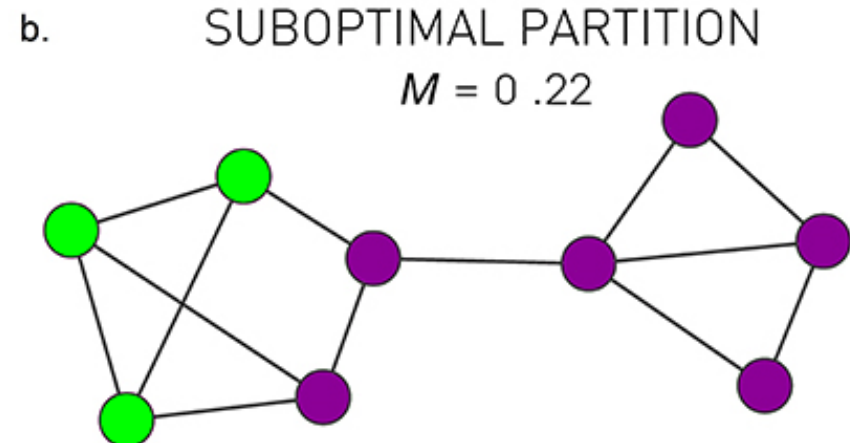
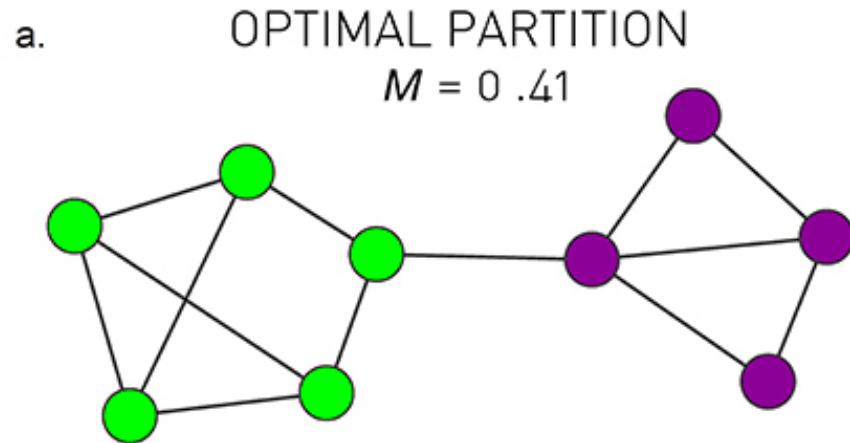
- onde  $m_k$  é o número de arestas dentro da partição  $k$ ,  $d_k$  é a soma dos graus dos vértices da partição  $k$

## Propriedades

- $M = 0$  se  $k=1$ : a rede inteira define um *baseline*
- $M < 0$  se  $k=n$ : muitas partições é ruim
- $\max M \leq 1$ : valor máximo da modularidade é 1
- Quanto maior, melhor!

**Partição é chamado de comunidade  
em *network science***

# Exemplo de Modularidade



# Calculando Modularidade

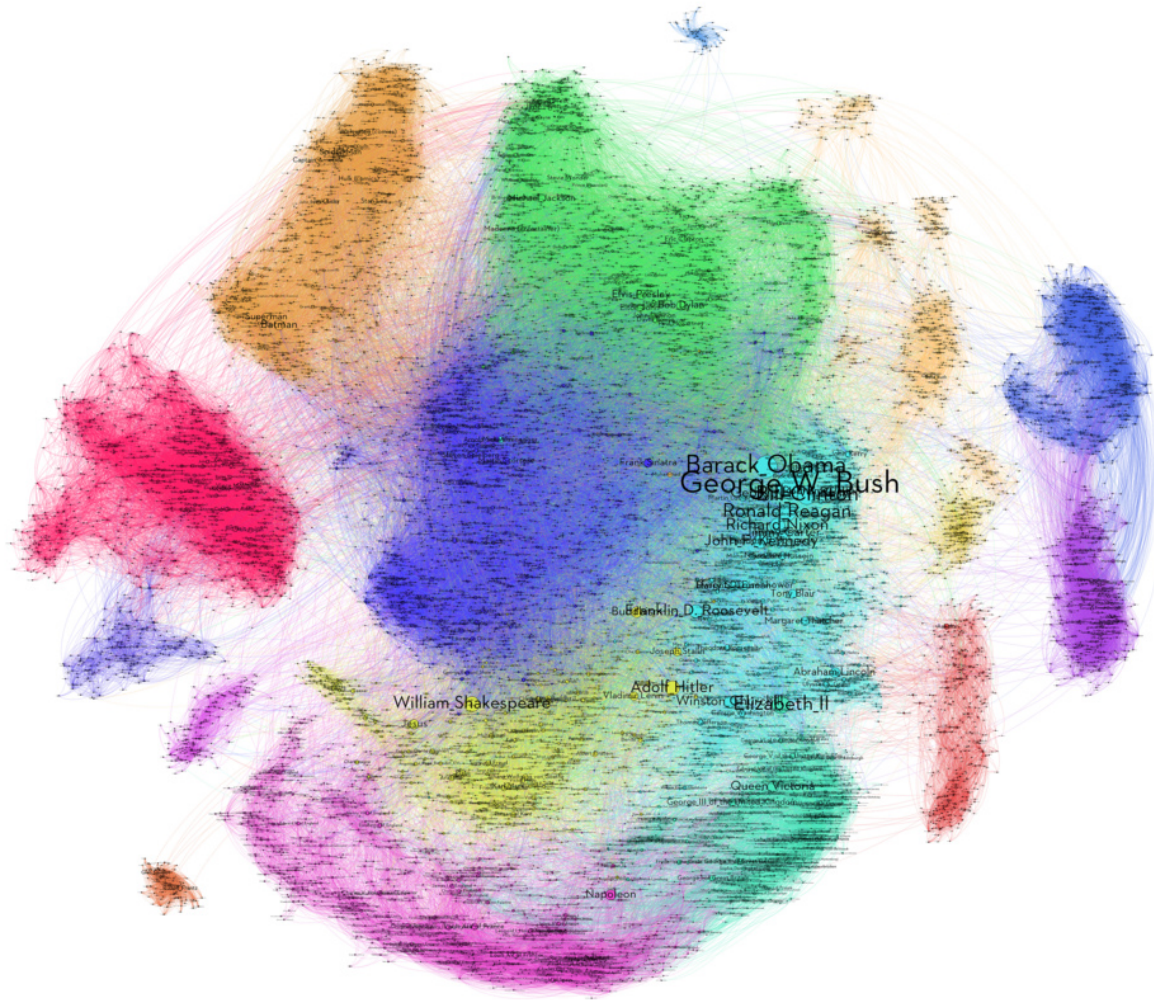
- **Problema:** determinar partição que maximiza modularidade em um grafo
  - não se conhece algoritmo eficiente
- Algoritmo guloso (heurística) de Newman
  - 1) Cada vértice em sua partição
  - 2) Determinar ganho de modularidade ao juntar um par de partições (para todos os pares)
  - 3) Fundir partição que possui maior ganho
  - 4) Voltar ao passo 2 até ter uma única partição
  - 5) Retornar partição de maior modularidade (guardar partição e modularidade a cada passo)
- Complexidade:  $O((n+m)n)$

# Algoritmo de Louvain

- Proposto por pesquisadores de Univ Catol de Louvain, Bélgica (Blondel et al.)
  - um guloso (heurística) ainda mais rápido
    - 1) Cada vértice em sua partição
    - 2) Determinar ganho de modularidade ao mover cada vértice para partição de um de seus vizinhos
    - 3) Mover vértices para partição de maior ganho (se houver ganho)
    - 4) Construir nova rede onde cada partição é um vértice
    - 5) Voltar ao passo 2 até não conseguir melhorar
- Complexidade:  $O(m)$

# Algoritmo de Louvain

- Funciona em redes com 100 milhões de vértices (bilhões de arestas)
- Algoritmo pode ser paralelizado



- Comunidades de uma rede social

# Limitações de Modularidade

- Métrica é enviesada para partições maiores
  - partições pequenas reduzem modularidade, quando rede cresce: *resolution limit*
- Diferença de modularidade ao juntar A e B

$$\delta M_{AB} = \frac{m_{AB}}{m} - \frac{d_A d_B}{2m^2}$$

- onde  $m_{AB}$  é o número de arestas entre A e B,  $d_A$  é a soma dos graus dos vértices da partição A

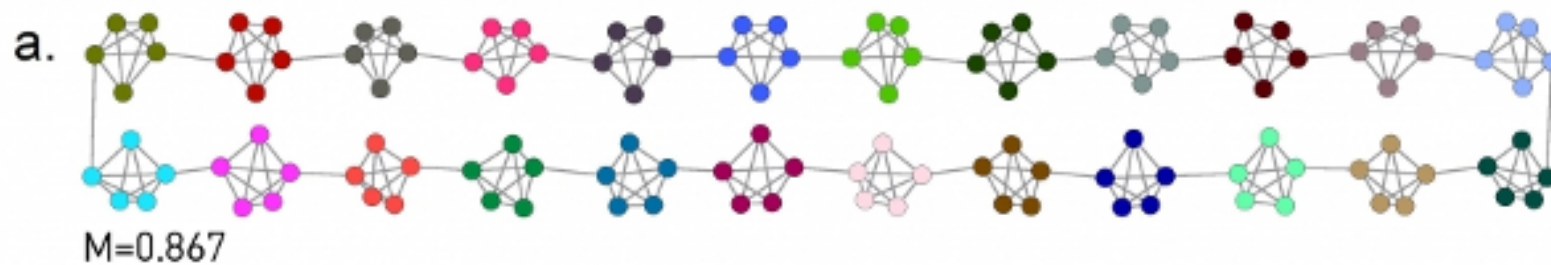
- Se  $d_A, d_B$  grande o suficiente, basta  $m_{AB} = 1$  para juntar as duas partições!

$$d_A \geq \sqrt{2m}$$

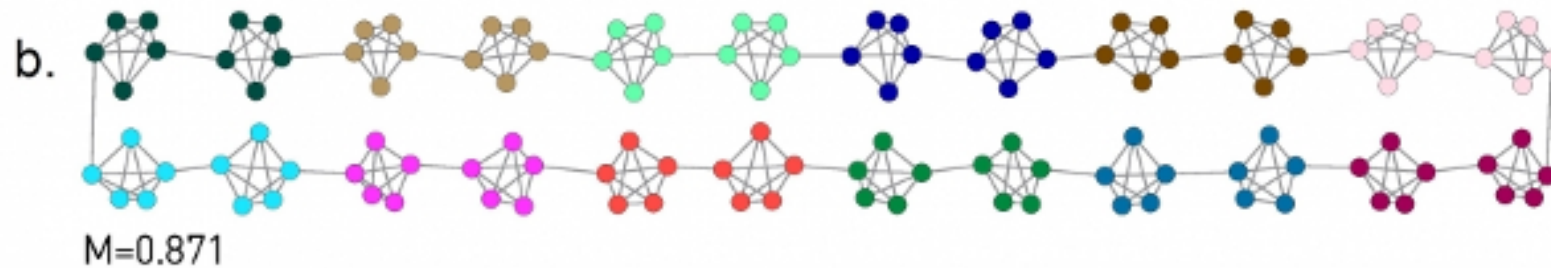
- Condição (aproximada) para tamanho mínimo de cada partição para maximizar modularidade da rede

# Limitações de Modularidade

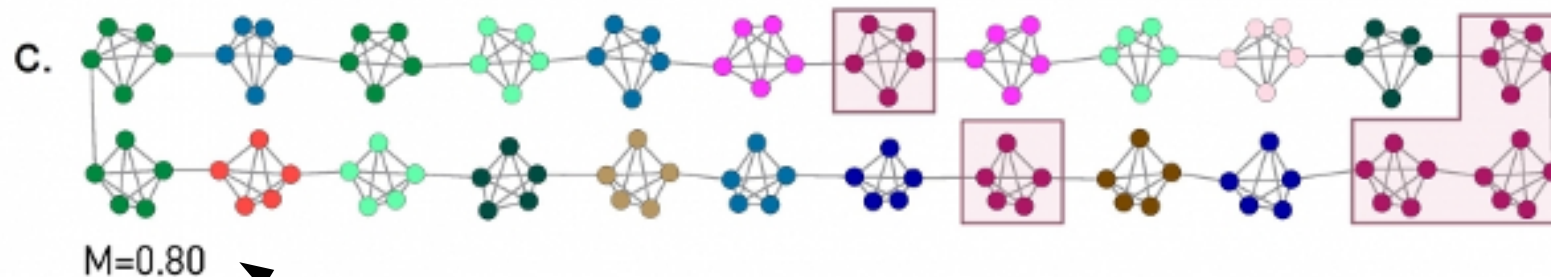
## ■ Anel com 24 cliques de tamanho 5



■ Partição intuitiva (cada clique uma partição)



■ Modularidade máxima (duas cliques por partição)



■ Partição aleatória (cliques em partições)

Valor relativamente alto!

# Partição com Sobreposição

- Vértices frequentemente pertencem a diferentes comunidades (partições)
  - redes sociais: família, trabalho, escola, hobby
  - redes de informação: conteúdo
- Partição com sobreposição
  - vértice pode estar em mais de uma partição (comunidade)
- Algoritmos
  - CFinder: baseado em percolação de cliques
  - Link Clustering: baseado em similaridade arestas

**Nada definitivo. Tema de pesquisa!**