

Minería de Medios Sociales

Máster en Ciencia de Datos e Ingeniería de Computadores

Bloque I: Redes Sociales y Ciencia de Datos en Redes

Sesión I.4: Detección de Comunidades

Oscar Cordón García

Dpto. Ciencias de la Computación e Inteligencia Artificial. Universidad de Granada
ocordon@decsai.ugr.es

ESTRUCTURA DE COMUNIDADES

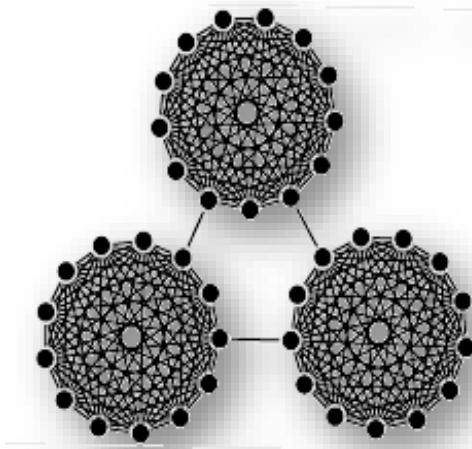
Las redes complejas tienden a mostrar una **estructura de comunidades**

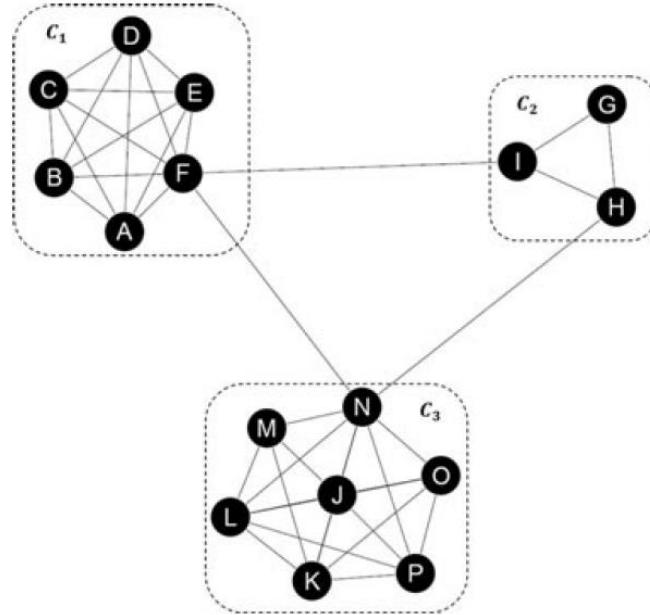
Esta propiedad suele darse como consecuencia de la heterogeneidad global y local de la distribución de los enlaces en un grafo

A menudo encontramos una **alta concentración de enlaces en ciertas regiones del grafo**, denominadas **comunidades**, y una **baja concentración de enlaces entre esas regiones (modularidad)**

Las comunidades, también conocidas como módulos o clusters, se definen de forma sencilla como grupos de nodos similares

A partir del concepto de densidad de la red, **las comunidades pueden definirse como grupos de nodos densamente conectados internamente que presentan conexiones dispersas entre sí**





- En esta red, hay **tres comunidades**: C_1 , C_2 y C_3
- Cada comunidad está formada por un grafo completo (un **clique**) de tamaño variable ($C_1= K_6$, $C_2= K_3$ y $C_3= K_7$)
- La densidad de enlaces entre las comunidades es muy baja. Los pocos enlaces que existen son **puentes**

La modularidad Q es una función que mide la calidad de una partición concreta de una red en comunidades. Toma valor máximo (teórico) cuando la red presenta todos los enlaces dentro de cada comunidad y ninguno entre comunidades

Se define como la diferencia entre el número de enlaces existentes en los grupos y el número de enlaces esperado en una red aleatoria equivalente:

$$Q = \frac{1}{2 \cdot L} \sum_{ij} \left[A_{ij} - \frac{k_i \cdot k_j}{2 \cdot L} \right] \delta(c_i, c_j)$$

número de
enlaces de la red → A_{ij}
 matriz de adyacencia → $\frac{k_i \cdot k_j}{2 \cdot L}$
 la probabilidad de un enlace entre dos
nodos es proporcional a sus grados → $\delta(c_i, c_j)$
 vale 1 si los nodos son
de la misma comunidad

Sumamos los enlaces internos a todas las comunidades de la red y les restamos el valor que tendría esa suma si los enlaces estuvieran dispuestos aleatoriamente

Si un nodo i tiene grado k_i y un nodo j grado k_j , la probabilidad de que estén conectados de forma aleatoria es $\frac{k_i \cdot k_j}{2 \cdot L}$

La idea es que la red muestra una estructura modular coherente si el número de enlaces internos de las comunidades es mayor que el esperado en una red aleatoria equivalente

$Q \in [-1, 1]$. Cuanto mayor es su valor, mejor es la partición, es decir, las comunidades encontradas están densamente conectadas internamente (hay más enlaces de los que cabría esperar aleatoriamente) y dispersamente conectadas entre sí

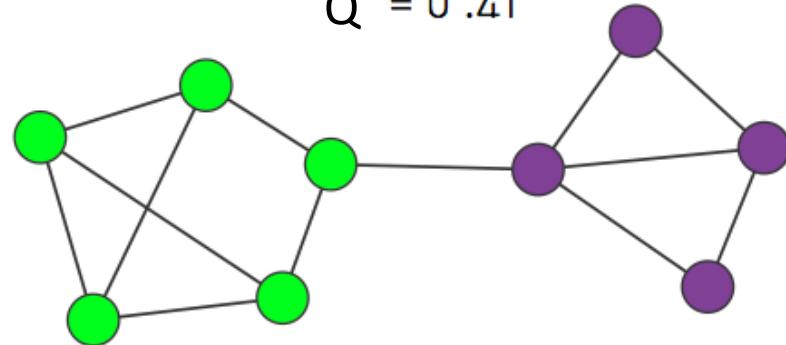
Un valor $Q=0$ corresponde a un agrupamiento trivial de una sola comunidad en una red aleatoria. **En la práctica, una modularidad de 0.3 es un buen valor**

Q se usa tanto para comparar la calidad de distintas particiones como diseñar métodos de descubrimiento de comunidades que traten de maximizar su valor

(a)

OPTIMAL PARTITION

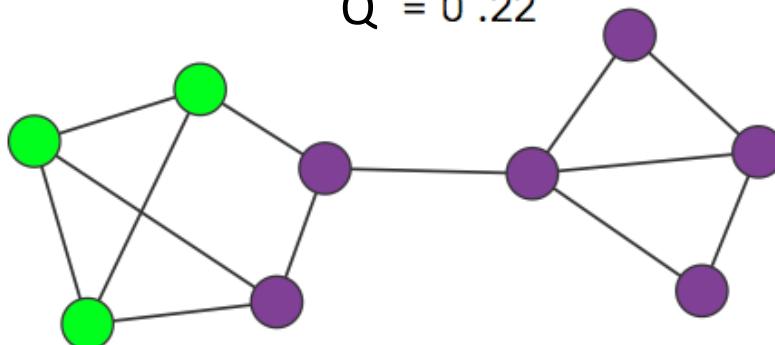
$$Q = 0.41$$



(b)

SUBOPTIMAL PARTITION

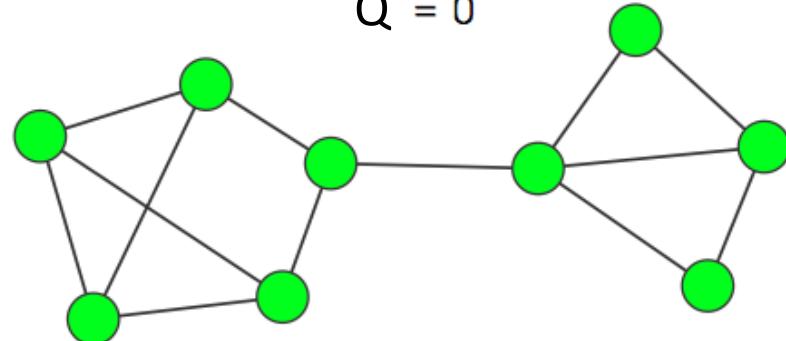
$$Q = 0.22$$



(c)

SINGLE COMMUNITY

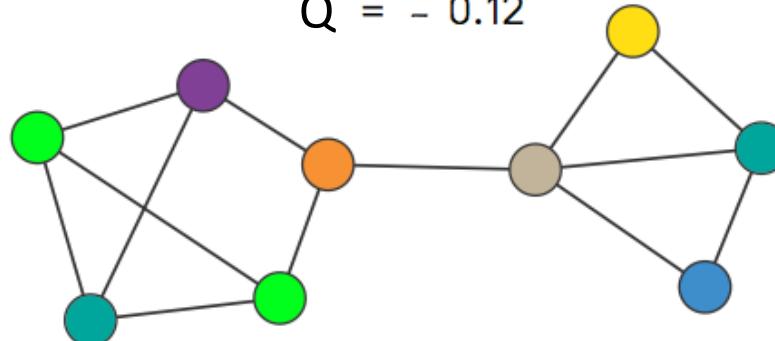
$$Q = 0$$



(d)

NEGATIVE MODULARITY

$$Q = -0.12$$



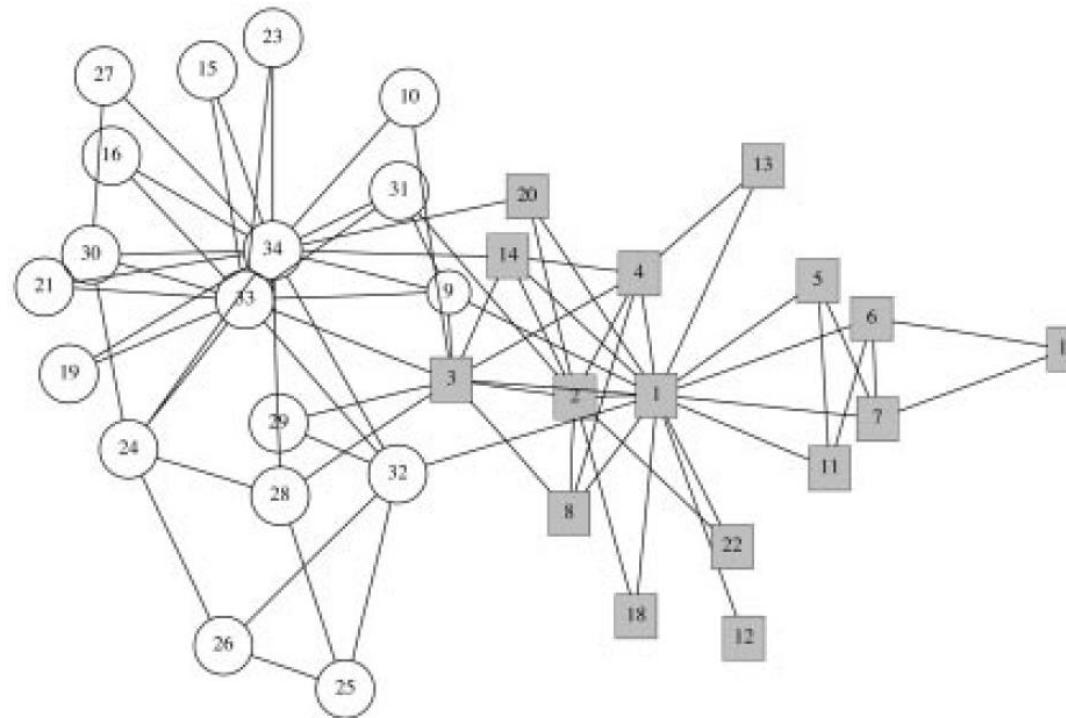
**¿PARA QUÉ DETECTAR
COMUNIDADES?**

COMUNIDADES EN EL MUNDO REAL (1)

En la vida real existen muchos ejemplos de grupos compactos en redes complejas:

- Sociedades: las personas tienen una tendencia natural a formar grupos (familias, círculos de amigos, grupos profesionales o religiosos, ciudades, naciones, etc.)
- Empresariales/Económicas: compañías, clientes, etc.
- Biología: p.ej. redes metabólicas. En redes de interacción de proteínas encontramos grupos de proteínas con funciones similares dentro de la célula
- **Medios sociales**: comunidades virtuales (Facebook, Twitter, etc.), grupos de páginas web relacionadas, usuarios con preferencias comunes, etc. (útiles para sistemas de recomendaciones – *social media mining*), ...

Los nodos de una comunidad tienen propiedades comunes

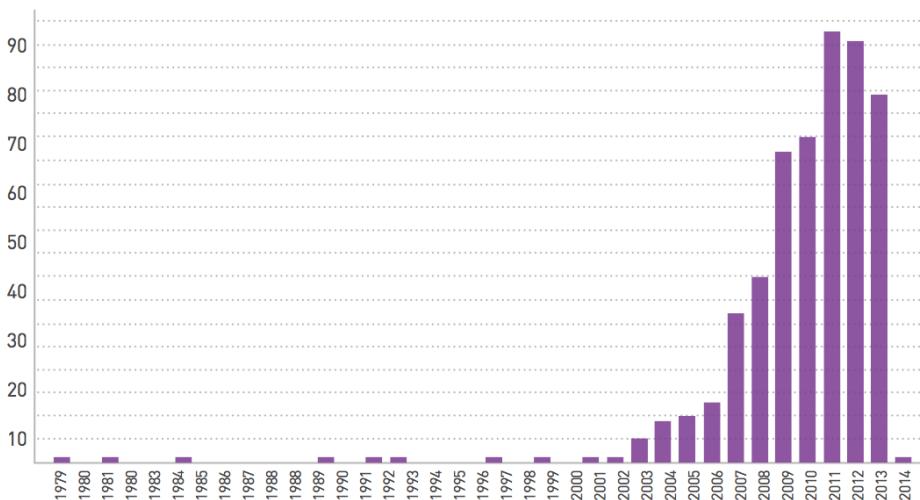


El club de karate de Zachary: Red clásica en la literatura. Club de karate en el que había un grupo cercano al administrador (nodo 34, nodos redondos blancos) y otro cercano al instructor (nodo 1, nodos cuadrados grises) y se separaron en dos grupos

Zachary. An information flow model for conflict and fission in small groups. J. Anthropol. Res. 33: 452-473 (1977)

COMUNIDADES EN EL MUNDO REAL (3)

Ejemplo clásico (2)



El artículo del club de karate de Zachary ha experimentado un auge tremendo desde que se inició el área de investigación en detección de comunidades. Sus citas han aumentado de forma exponencial

Su popularidad es tal que existe un premio para el primer científico que use ese conjunto de datos en cada conferencia de Redes Complejas, pasando a formar parte del *Club del Club de Karate de Zachary*

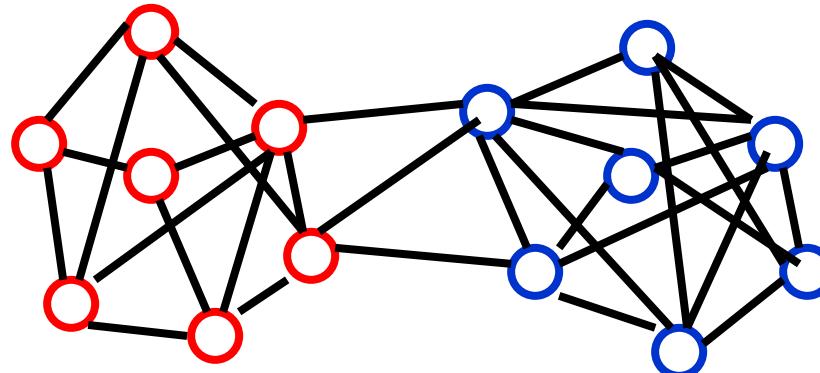
In social media mining, analyzing communities is essential. Studying communities in social media is important for many reasons. First, individuals often form groups based on their interests, and when studying individuals, we are interested in identifying these groups. Consider the importance of finding groups with similar reading tastes by an on-line book seller for recommendation purposes. Second, groups provide a clear global view of user interactions, whereas a local-view of individual behavior is often noisy and ad hoc. Finally, some behaviors are only observable in a group setting and not on an individual level. This is because the individual's behavior can fluctuate, but group collective behavior is more robust to change. Consider the interactions between two opposing political groups on social media. Two individuals, one from each group, can hold similar opinions on a subject, but what is important is that their communities can exhibit opposing views on the same subject.

Communities in social media are more or less representatives of communities in the real world. The geographical location becomes less important in social media, and many communities on social media consist of highly diverse people from all around the planet. Similar to real-world communities, communities in social media can be labeled as explicit or implicit. Examples of explicit communities include the following:

- **Facebook.** In Facebook, there exist a variety of explicit communities, such as *groups* and *communities*. In these communities, users can post messages and images, comment on other messages, like posts, and view activities of others.
- **Yahoo! Groups.** In Yahoo! groups, individuals join a group mailing list where they can receive emails from all or a selection of group members (administrators) directly.
- **LinkedIn.** LinkedIn provides its users with a feature called *Groups and Associations*. Users can join professional groups where they can post and share information related to the group.

Because these sites represent explicit communities, individuals have an understanding of when they are joining them. However, there exist implicit communities in social media as well. For instance, consider individuals with the same taste for certain movies on a movie rental site. These individuals are rarely all members of the same explicit community. However, the movie rental site is particularly interested in finding these implicit communities so it can better market to them by recommending movies similar to their tastes. We discuss techniques to find these implicit communities next.

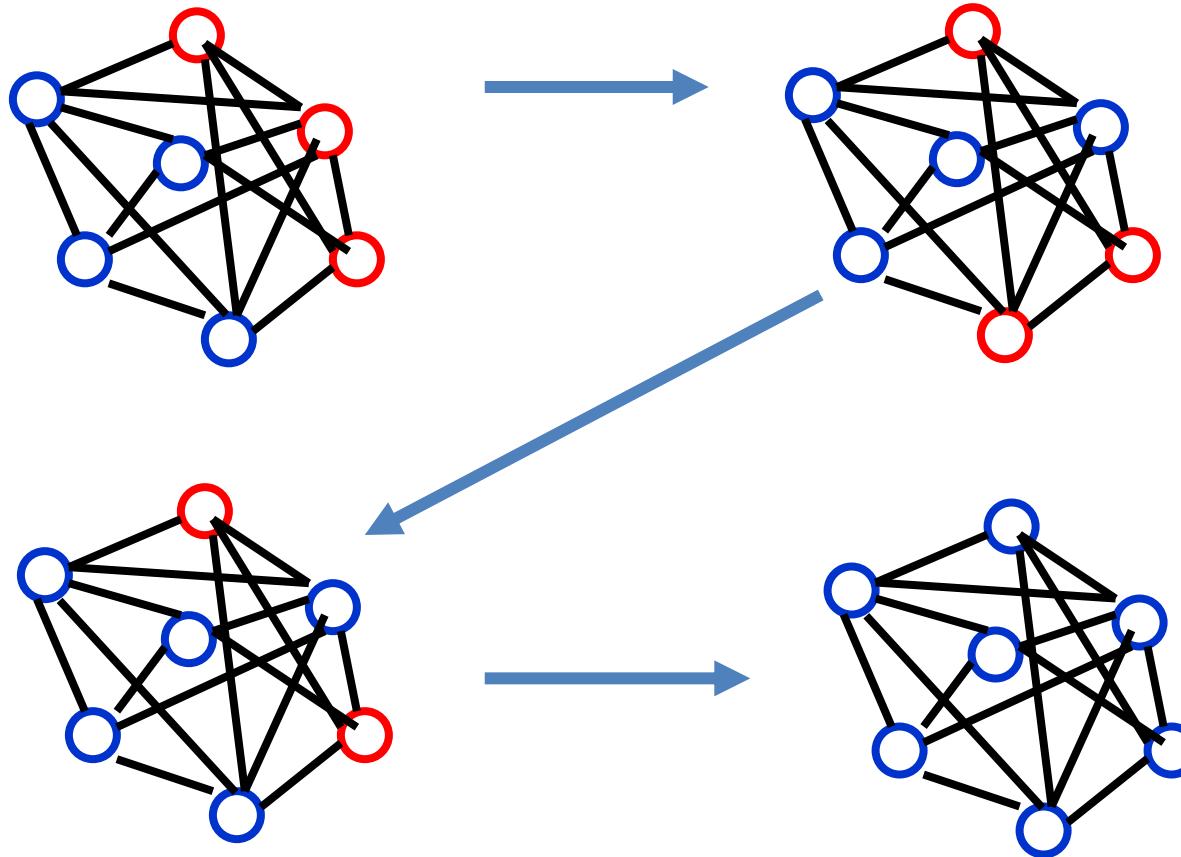
FORMACIÓN DE OPINIONES Y UNIFORMIDAD:



Si cada nodo adopta la opinión de la mayoría de sus vecinos, es posible formar opiniones distintas en subgrupos cohesivos distintos

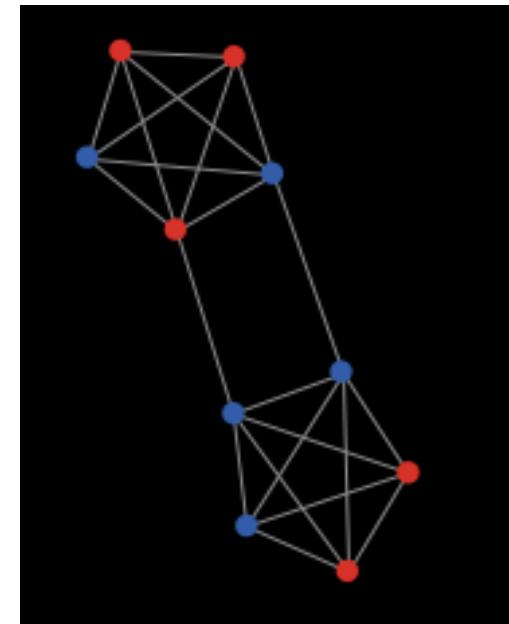
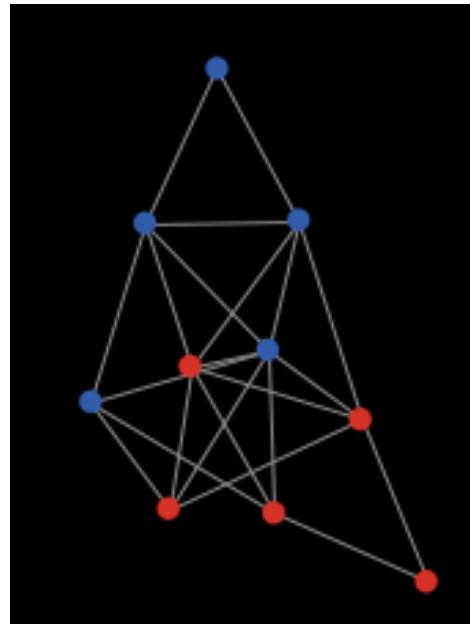
Hay más uniformidad dentro de un grupo cohesivo

Hay más uniformidad dentro de un grupo cohesivo:



<http://www.ladamic.com/netlearn/NetLogo502/OpinionFormationModelToy.html>

- Cada nodo adopta la opinión mayoritaria de sus vecinos (una opinión aleatoria, en caso de empates)
- Si la red no tiene comunidades todos los nodos tienden a la misma opinión
- Si tiene comunidades, cada comunidad puede tener una opinión distinta



MÉTODOS DE DETECCIÓN DE COMUNIDADES

Fortunato. Community detection in graphs. Phys Rep 486: 75–174 (2010)

Lista actualizada de referencias en:

<http://bactra.org/notebooks/community-discovery.html>

TAXONOMÍA DE LOS MÉTODOS DE DETECCIÓN DE COMUNIDADES

Se pueden clasificar en función del criterio considerado para la detección de las comunidades. Se consideran cuatro familias, no exclusivas:

1 y 2. Comunidades centradas en nodos y en grupos de nodos:

- *Cada nodo* del grupo satisface ciertas propiedades (criterio estructural)
- Se consideran las conexiones *del grupo* globalmente. El grupo completo tiene que satisfacer ciertas propiedades, no los nodos individuales (cr. estructural)

Wasserman y Faust. Social Network Analysis. Cambridge University Press; 1994

3. Comunidades centradas en jerarquía:

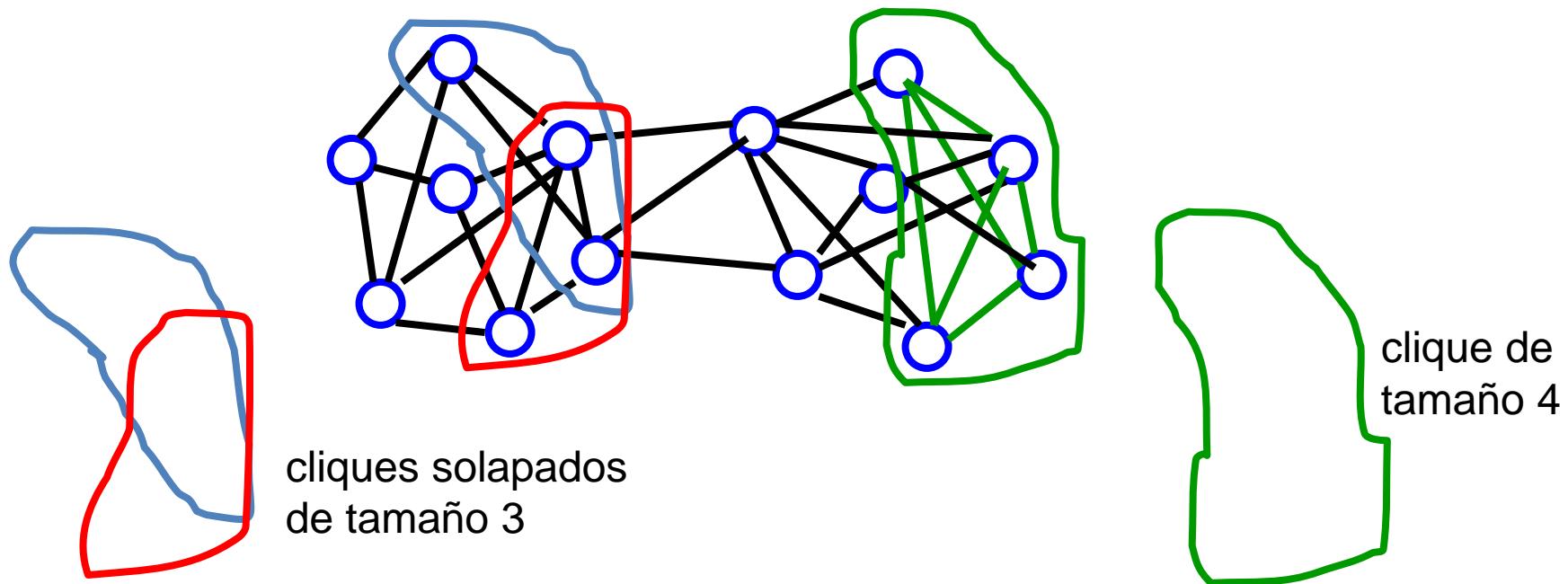
- Se construye una *estructura jerárquica* de comunidades (clustering jerárquico)

4. Comunidades centradas en la red:

- Se divide *la red completa* en conjuntos disjuntos (particionamiento de grafos)

MÉTODOS DE DETECCIÓN DE COMUNIDADES CENTRADOS EN NODOS Y GRUPOS DE NODOS

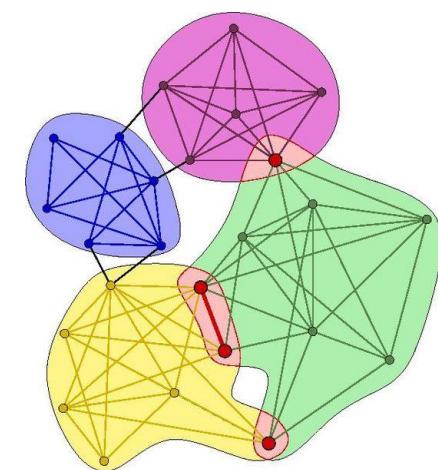
- Todos los miembros del grupo tienen enlaces al resto
- Los triángulos son los cliques más básicos, los de mayor dimensión son menos frecuentes
- Los cliques pueden estar solapados



- Es un problema NP-completo
- No son robustos:
 - Un solo enlace faltante hace que el grupo no sea considerado una comunidad
- No son interesantes:
 - Todo el mundo está conectado entre sí. No hay estructura centro-periferia, no hay jerarquía entre los enlaces. Las medidas de centralidad no dan información
- El solapamiento de los cliques puede ser más relevante que su propia existencia... Existen métodos que consideran solapamiento, como el **Clique percolation method (CPM)**

Palla y otros. Uncovering the overlapping community structure of complex networks in nature and society. Nature 435: 814–818 (2005).

<http://www.cfinder.org>



Sea un subgrafo conexo G_s con N_s nodos

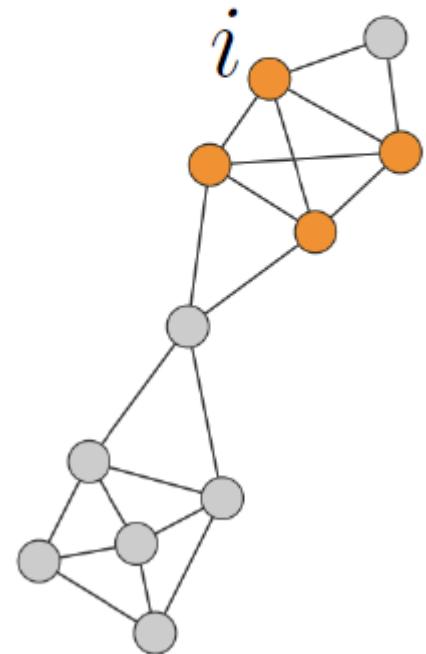
- Se define el *grado interno* de un nodo i , k_i^{int} , como el conjunto de enlaces que lo conectan con otro nodo del mismo subgrafo
- Se define el *grado externo* de un nodo i , k_i^{ext} , como el conjunto de enlaces que lo conectan al resto de la red

Si $k_i^{\text{ext}} = 0$ todos los vecinos de i pertenecen al subgrafo y G_s es una buena comunidad

Si $k_i^{\text{int}} = 0$ todos los vecinos de i pertenecen a otras comunidades e i debe asignarse a una comunidad distinta

$$k_i^{\text{int}} = 3$$

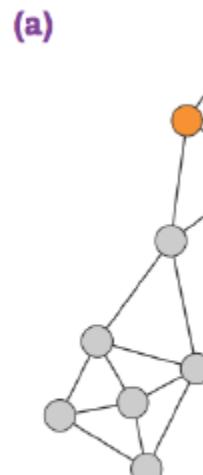
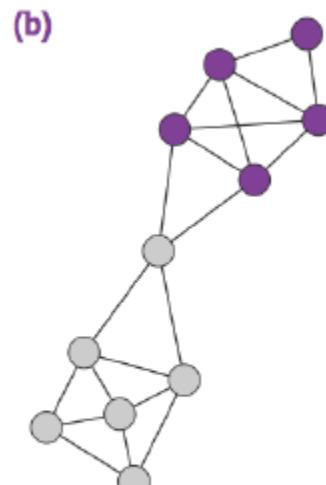
$$k_i^{\text{ext}} = 1$$



Comunidad fuerte (nodos):

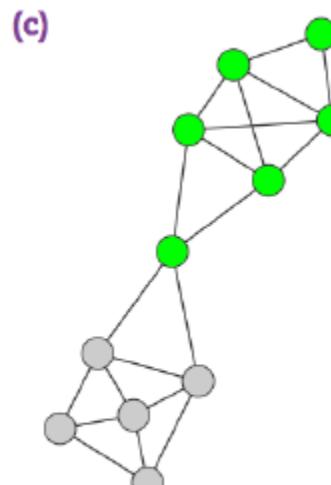
Cada nodo de G_s tiene más enlaces dentro de la comunidad que con el resto del grafo

$$k_i^{\text{int}}(G_s) > k_i^{\text{ext}}(G_s)$$

*Clique**Fuerte***Comunidad débil** (grupo):

El grado interno total de G_s es mayor que su grado externo total

$$\sum_{i \in G_s} k_i^{\text{int}}(G_s) > \sum_{i \in G_s} k_i^{\text{ext}}(G_s)$$

*Débil*

MÉTODOS DE DETECCIÓN DE COMUNIDADES CENTRADOS EN RED Y JERARQUÍA

3-4. MÉTODOS CENTRADOS EN LA RED Y EN JERARQUÍA

Según Newman y Girvan (2004), existen dos líneas de investigación principales para el descubrimiento de comunidades **a nivel global** en redes complejas:

1. **Clustering jerárquico** (también llamado **detección de la estructura de comunidades**): se origina en Sociología. Está motivado por el descubrimiento de grupos en una sociedad para facilitar el análisis de fenómenos sociales
2. **Particionamiento de grafos**: tiene su origen en Informática, en el campo de la computación distribuida. Busca la mejor forma de asignar tareas a procesadores para minimizar las comunicaciones entre ellos

En cualquier caso, el propósito de ambos enfoques es descubrir grupos de nodos relacionados y, si es posible, la estructura jerárquica correspondiente, a partir de la información proporcionada por la topología de la red

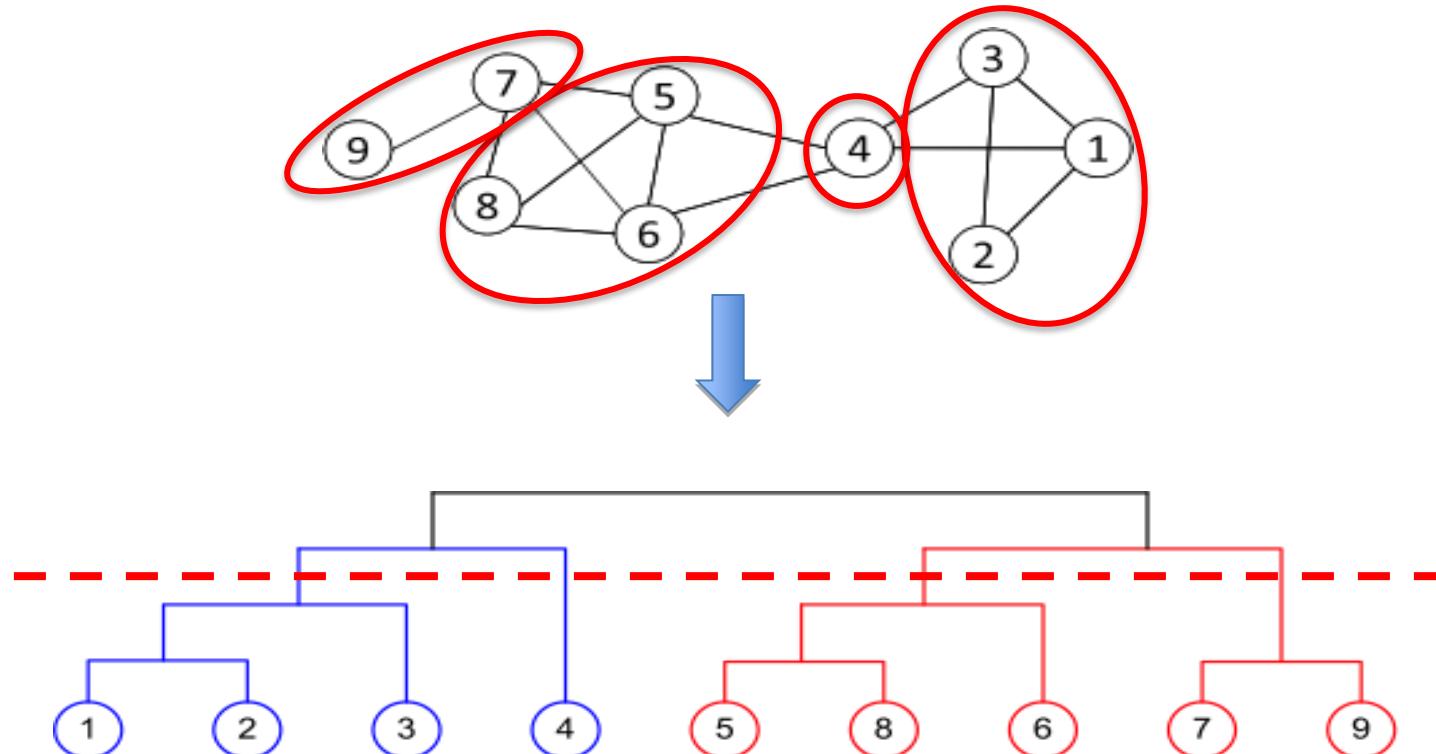
Estos métodos devuelven **particiones disjuntas** del conjunto de nodos, cada nodo pertenece a una única comunidad (no permiten el solapamiento de comunidades)

3. MÉTODOS CENTRADOS EN JERARQUÍA

- Su objetivo es construir una **estructura jerárquica** de comunidades basada en la topología de la red
- Permite analizar la red a **distintas resoluciones**
- Enfoques representativos:
 - Clustering jerárquico aglomerativo (*bottom-up*)
 - Clustering jerárquico divisivo (*top-down*)

- Se comienza con **todos los nodos desconectados**, considerando cada uno de ellos como una **comunidad independiente**
- Las similitudes entre pares de nodos (i,j) se calculan como un peso w_{ij} usando distintas medidas: coseno, Jaccard, distancia Euclídea, Manhattan, Hamming, número total de caminos entre i y j , número de caminos disjuntos entre i y j , etc.
- La similitud debe ser alta para pares de nodos que tengan una probabilidad alta de pertenecer a la misma comunidad y baja para los que probablemente sean de comunidades distintas
- Se van enlazando los grupos por pares, un par o varios en cada paso, mezclando comunidades, en orden de similitud decreciente

- **Resultado:** Componentes anidados formando un **dendrograma**, del que se pueden obtener distintas particiones cortando a varios niveles del árbol

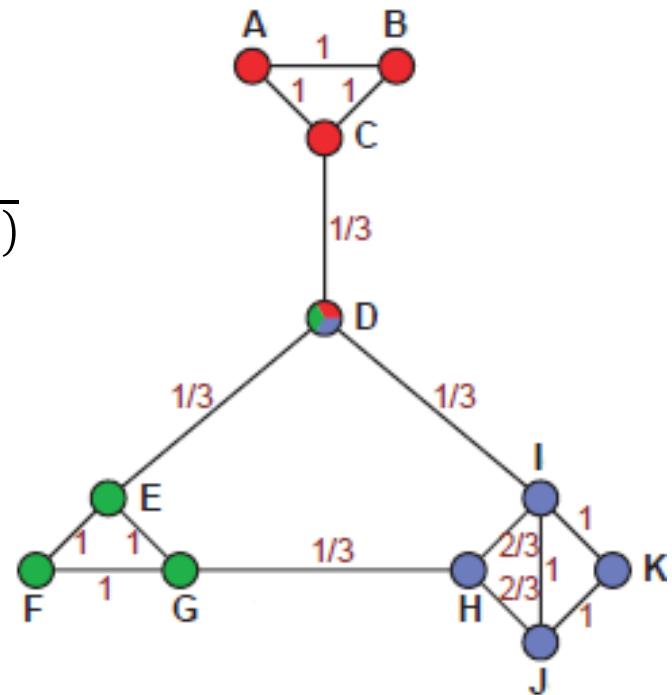
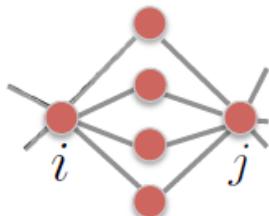


Dendrograma resultante de un clustering aglomerativo basado en modularidad

Ravasz y otros. Hierarchical Organization of Modularity in Metabolic Networks. Science 297: 1551-1555 (2002)

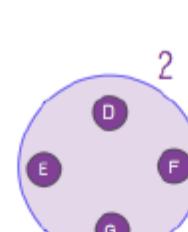
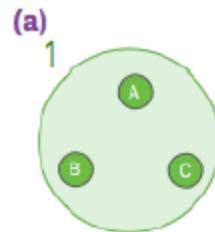
Paso 1: Definición de la matriz de similitud:

- Los nodos que están conectados entre sí o que comparten muchos vecinos tienen más probabilidad de pertenecer al mismo grupo (vecindario local denso)
- Por tanto, su similitud $s(i,j)$ debe ser más alta
- Matriz de solapamiento topológico: $s(i,j) = \frac{J_N(i,j)}{\min(k_i, k_j)}$
- $J_N(i,j)$ es el número de vecinos comunes entre i y j .
Vale 1 si existe el enlace directo (i,j)



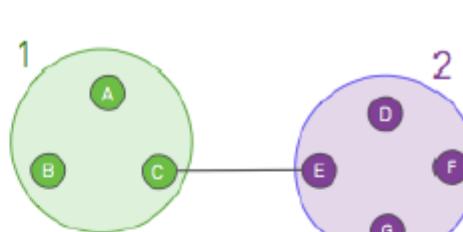
Paso 2: Decisión del mecanismo de agregación de los grupos:

- Los grupos se mezclan en función de su distancia D, calculada a partir de la similitud s_{ij} / distancia d_{ij} de los objetos que los forman. Existen distintas posibilidades: **fusión de grupos** enlace simple, enlace completo o promedio

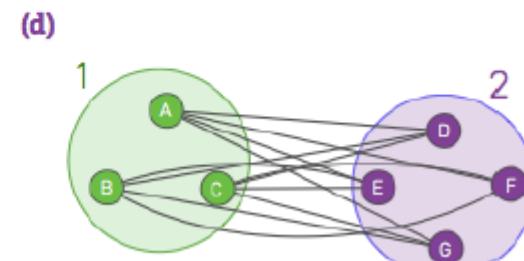
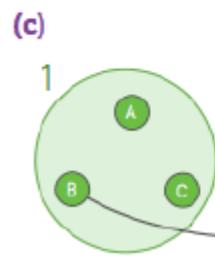


$$d(i,j) = \frac{1}{s(i,j)} =$$

	D	E	F	G
A	2.75	2.22	3.46	3.08
B	3.38	2.68	3.97	3.40
C	2.31	1.59	2.88	2.34



Single Linkage $D(1,2)= 1.59$



Complete Linkage $D(1,2)= 3.97$

Average Linkage: $D(1,2)= 2.84$

Paso 3: Aplicación del clustering jerárquico aglomerativo:

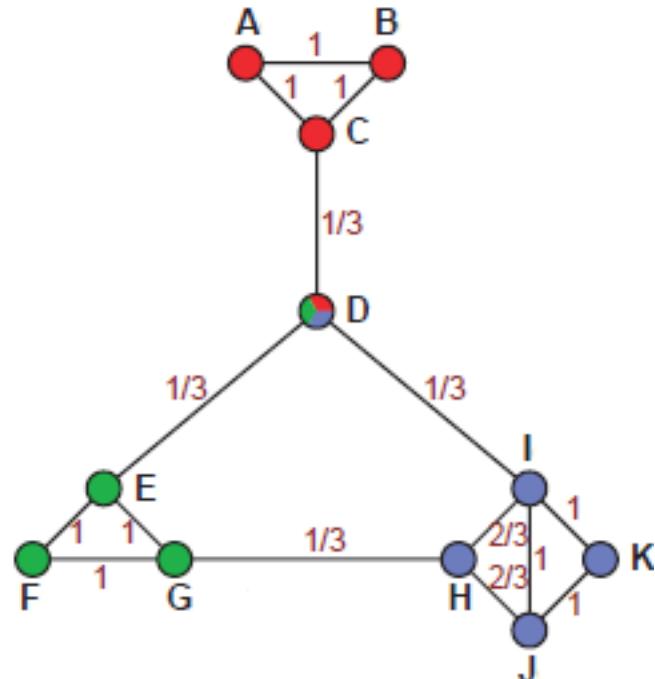
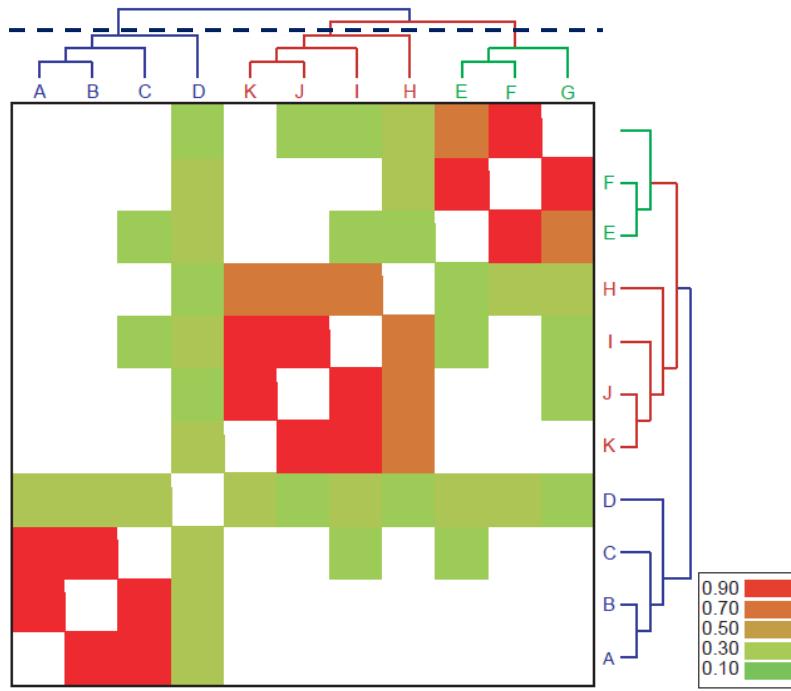
- Asignar cada nodo a una comunidad individual y evaluar la similitud (distancia) entre ellas. Las similitudes (distancias) iniciales entre esas comunidades son directamente las similitudes (distancias) entre los pares de nodos $s(i,j)$ ($d(i,j)$)
- Escoger el par de comunidades con el mayor valor de similitud $S(i,j)$ (el menor valor de distancia $D(i,j)$) y mezclarlas para formar una única comunidad
- Recalcular la similitud entre la nueva comunidad y el resto
- Repetir hasta que todos los nodos queden integrados en una única comunidad

Paso 4: Construcción del dendrograma:

- Representar el orden concreto en que los nodos se asignan a las distintas comunidades generadas

CLUSTERING AGLOMERATIVO

Algoritmo de Ravasz (4)



Orden de complejidad:

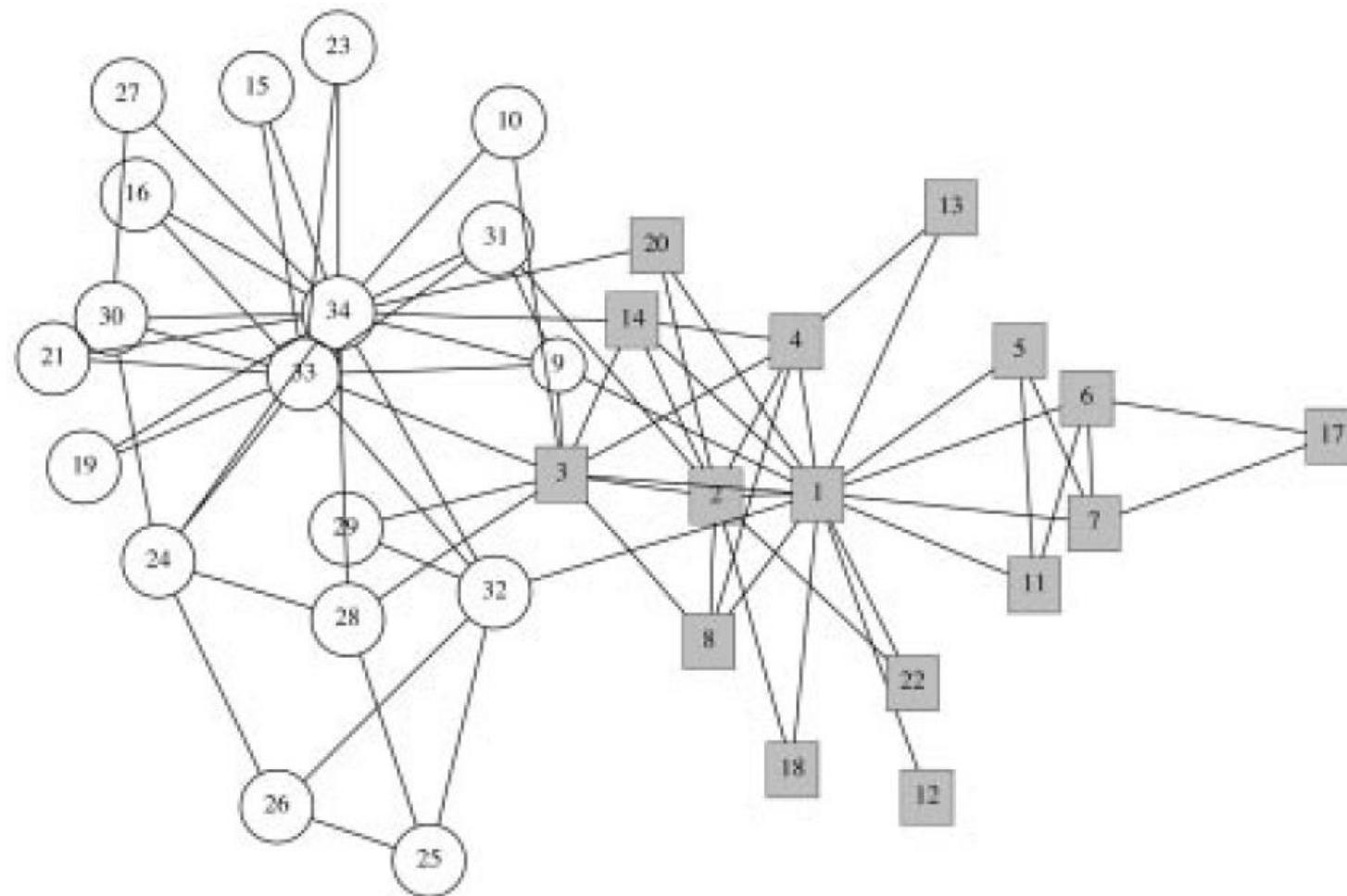
- Paso 1 (cálculo de la matriz de similitud): $O(N^2)$
- Pasos 2 y 3: (cálculo de las similitudes entre grupos): $O(N^3)$
- Paso 4 (dendrograma): $O(N \log N)$



$O(N^3)$

CLUSTERING AGLOMERATIVO

Ejemplo: club de karate de Zachary (1)

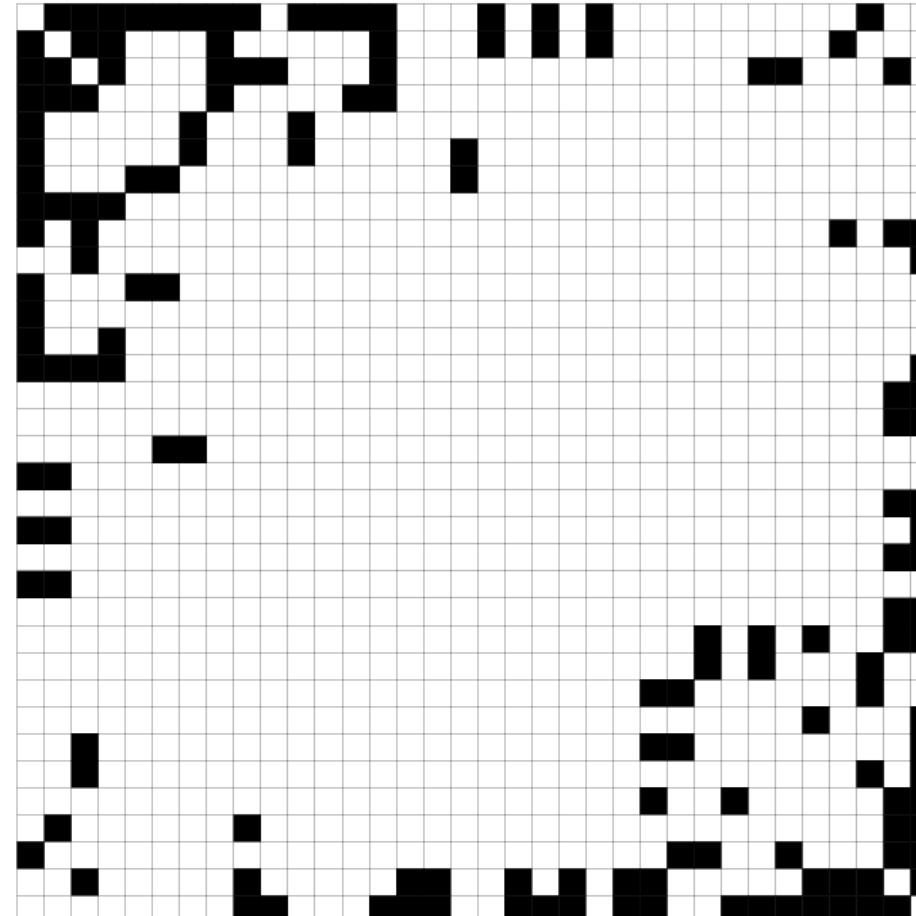


Zachary. An information flow model for conflict and fission in small groups. J. Anthropol. Res. 33: 452-473 (1977)

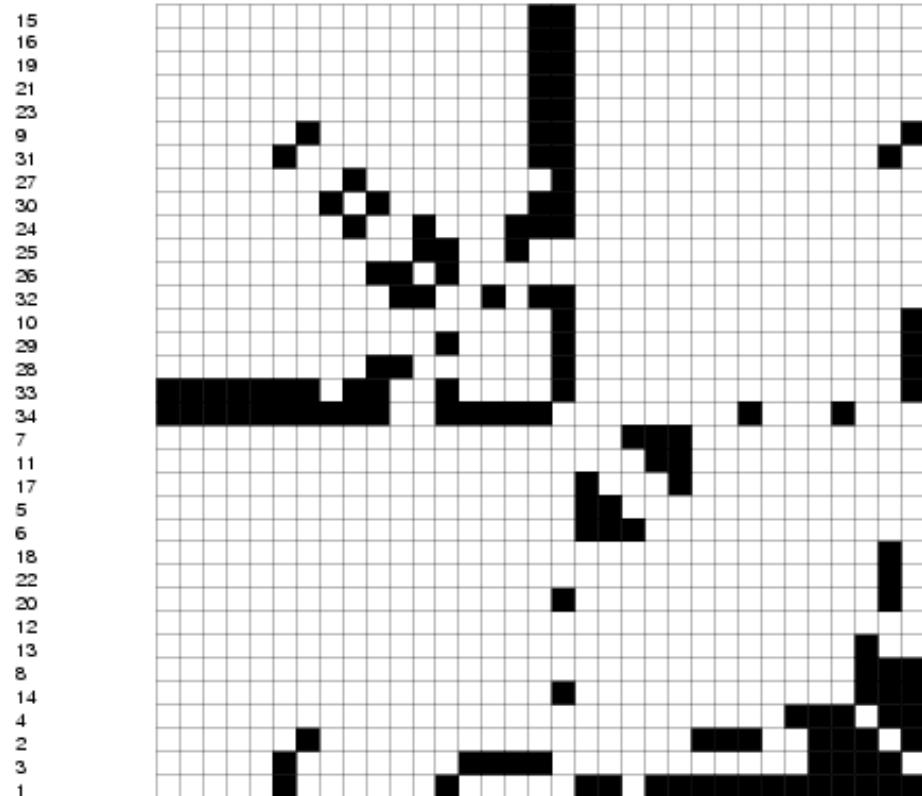
CLUSTERING AGLOMERATIVO

Ejemplo: club de karate de Zachary (2)

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34



Matriz de adyacencia original

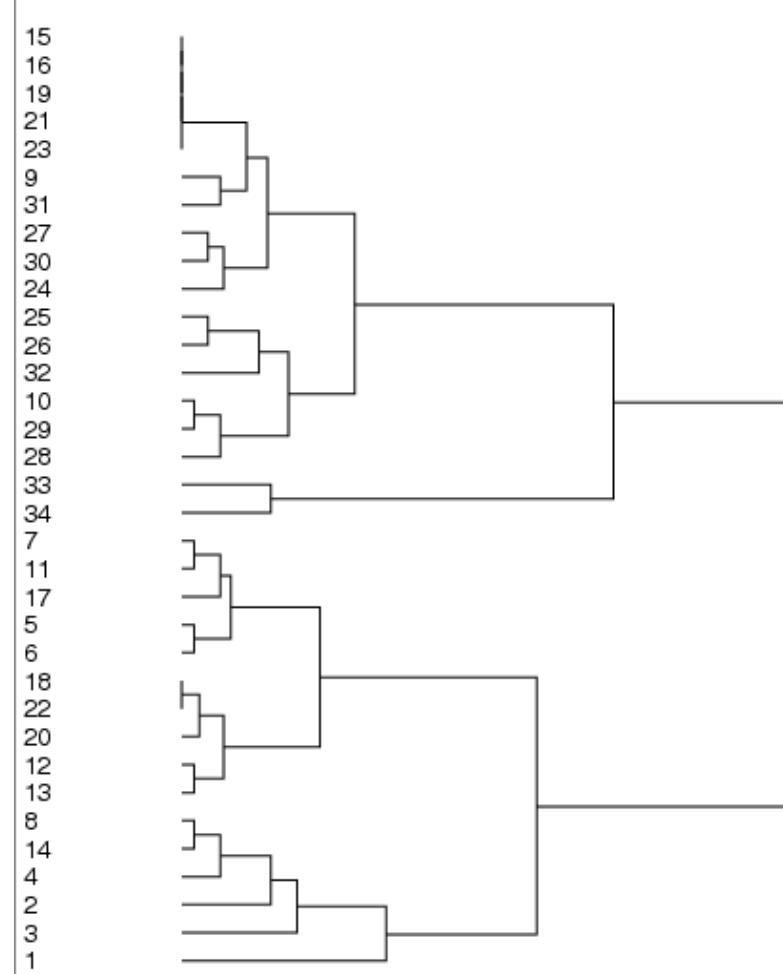


© 2014 Luis M. Alfonso - Facultad de Informática - Universidad de Valencia

Matriz ordenada en función del agrupamiento obtenido

CLUSTERING AGLOMERATIVO

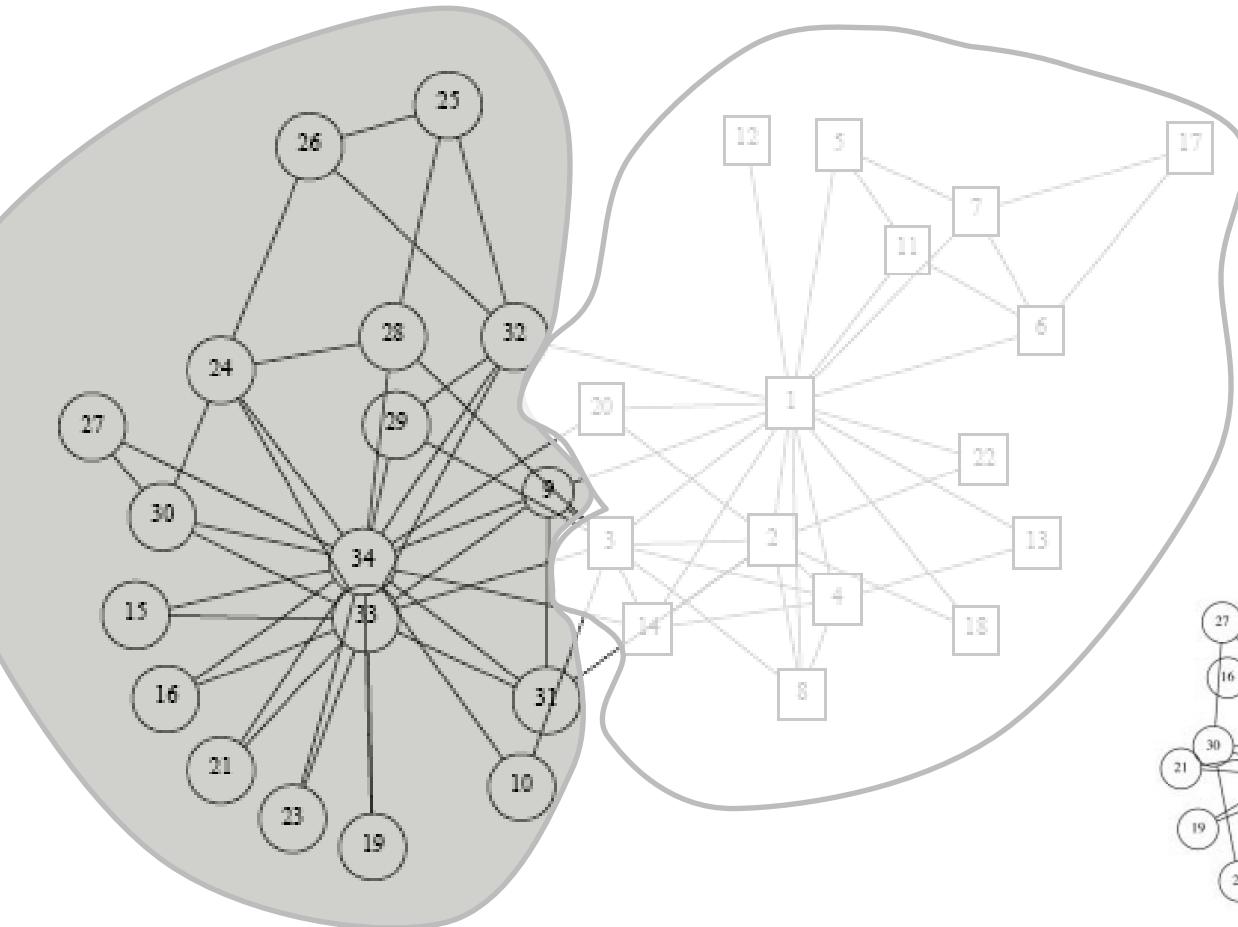
Ejemplo: club de karate de Zachary (4)



Dendrograma

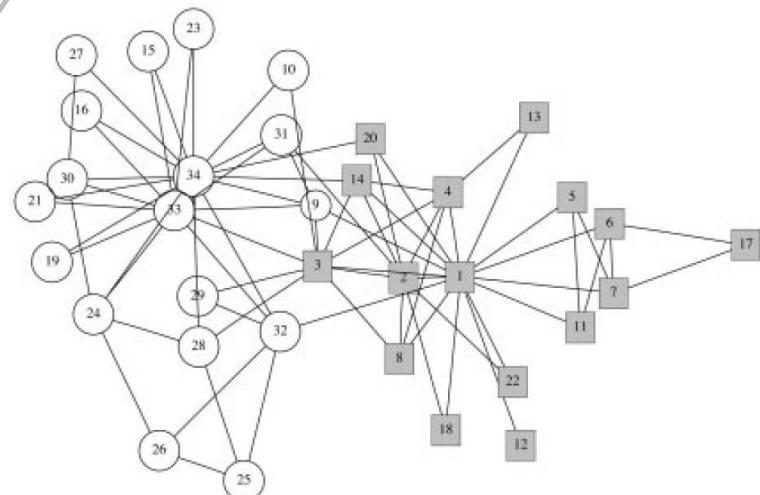
CLUSTERING AGLOMERATIVO

Ejemplo: club de karate de Zachary (5)



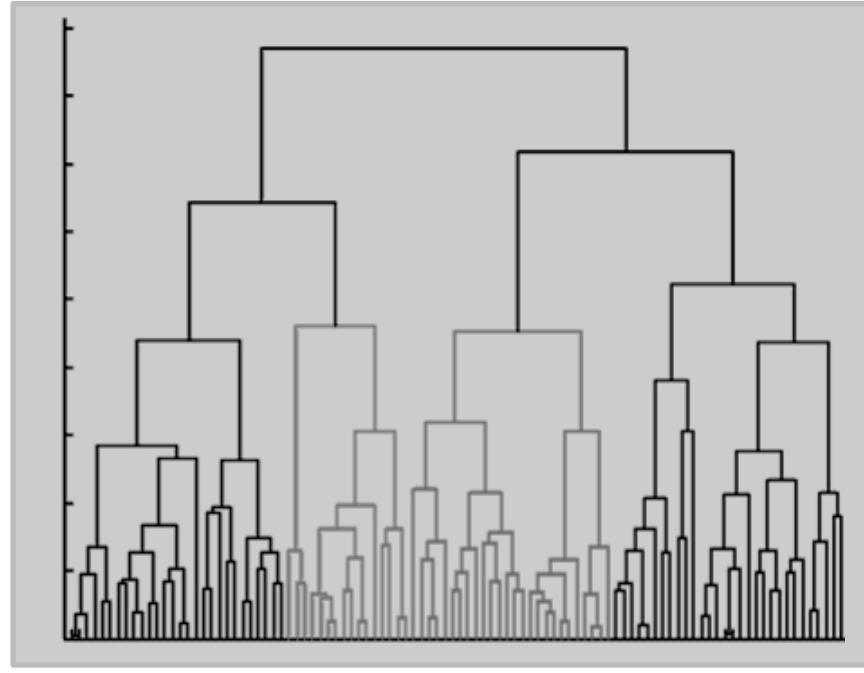
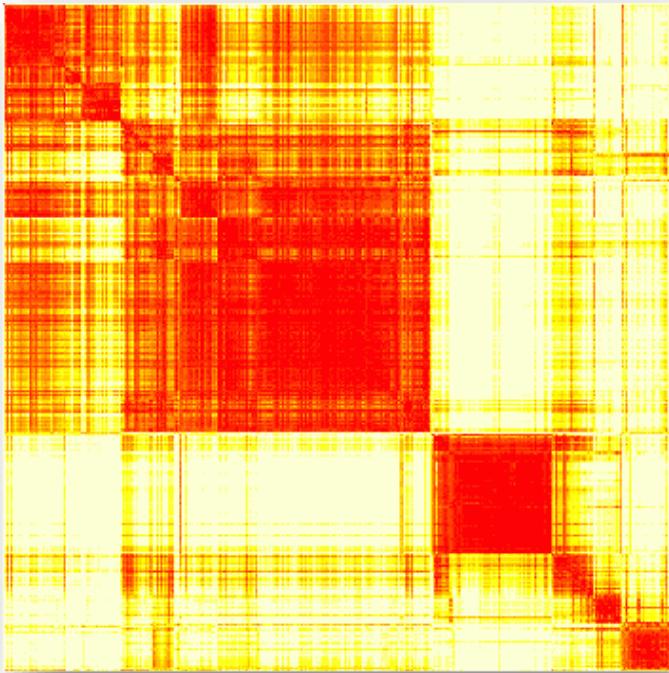
círculos → grupo del administrador (nodo 34)

cuadrados → grupo del instructor (nodo 1)



CLUSTERING AGLOMERATIVO

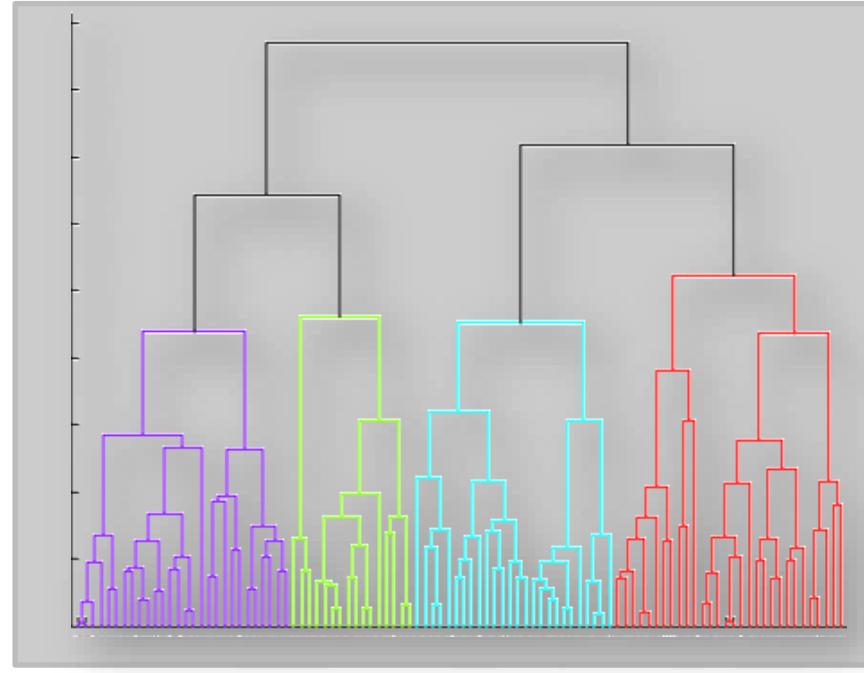
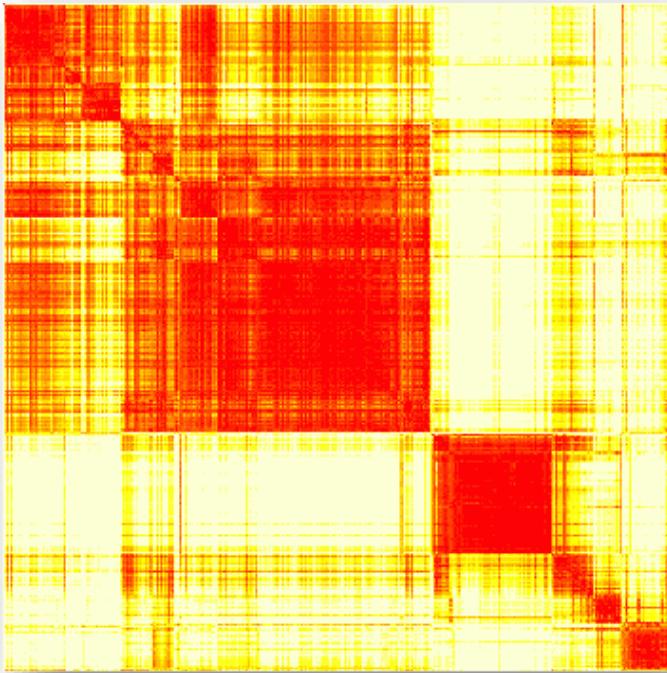
Selección de la partición óptima



Una buena estrategia para escoger la mejor partición, i.e. el mejor número de comunidades, es calcular el valor de **modularidad** para cada partición posible y seleccionar la que maximice el valor de la función

CLUSTERING AGLOMERATIVO

Selección de la partición óptima



Una buena estrategia para escoger la mejor partición, i.e. el mejor número de comunidades, es calcular el valor de **modularidad** para cada partición posible y seleccionar la que maximice el valor de la función

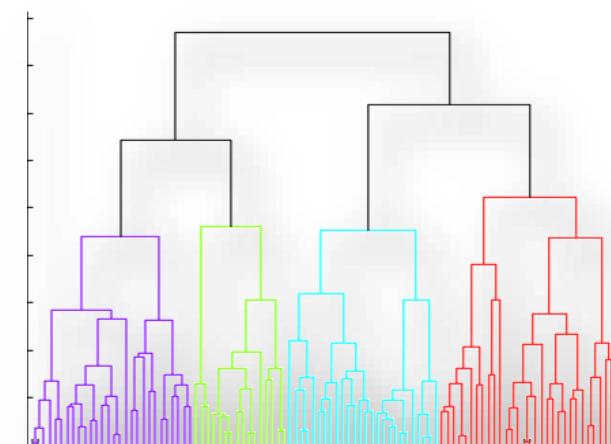
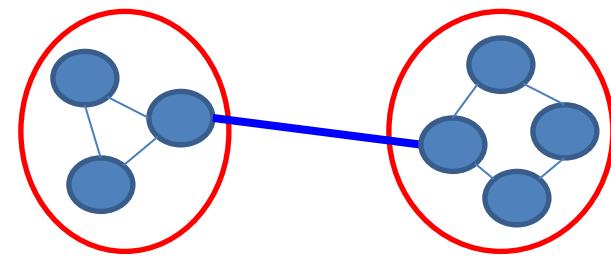
- Se comienza con los nodos agrupados en subconjuntos iniciales. El caso extremo es considerar toda la red como una única comunidad
- Cada subconjunto se va dividiendo en otros más pequeños. Se puede considerar un método basado en particionamiento de grafos para ello
- Un caso particular es la “eliminación recursiva del enlace más débil”:
 - Encontrar el enlace de menor peso
 - Eliminarlo y actualizar el peso de los enlaces restantes
- Se repiten los pasos anteriores hasta que la red se divide en el número de componentes deseado
- Cada componente forma una comunidad

CLUSTERING DIVISIVO

El método de Girvan-Newman (1)



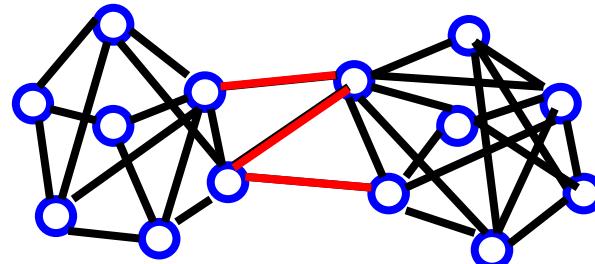
La medida de similitud ([el peso del enlace](#)) se basa en la topología de la red:
intermediación de los enlaces (número de caminos geodésicos que pasan por el enlace en cuestión)



Girvan y Newman. Community structure in social and biological networks. Proc Natl Acad Sci USA 99:7821–7826 (2002)

Técnica de clustering jerárquico divisiva: divide la red original en partes conectadas más pequeñas de forma progresiva hasta que no haya más enlaces que eliminar y cada nodo representa una comunidad por sí solo o hasta un cierto umbral:

1. Calcular el valor de intermediación de todos los enlaces de la red
2. Eliminar el enlace de mayor intermediación. Este paso puede provocar que la red se divida en partes desconectadas, formando el primer nivel de nivel de regiones en la división de la red
3. Repetir los pasos anteriores hasta que no haya enlaces que eliminar en la red o hasta que el valor máximo de intermediación sea menor que un umbral



Este método de descubrimiento de comunidades es muy popular y existen un gran número de implementaciones. P.ej. la biblioteca *igraph* de *R* incorpora la función *edge.betweenness.community*

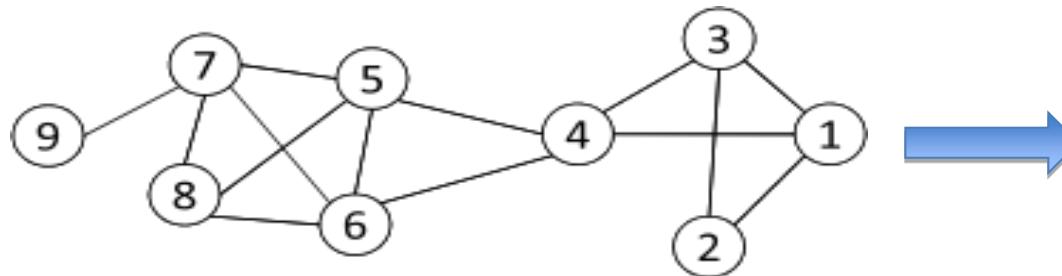
Su problema es que es necesario **recalcular la intermediación en cada paso**:

- Calcular la intermediación para todos los enlaces es muy costoso, $O(L \cdot N)$, donde L es el número de enlaces y N el de nodos
- La eliminación de un enlace suele impactar en la intermediación del resto. Es necesario recalcularla cada vez: $O(L^2 \cdot N) \rightarrow O(N^3)$ en redes densas
- El método no escala bien para redes con más de unos cientos de nodos, incluso en las implementaciones más rápidas

Hay aproximaciones alternativas al método de GN que reemplazan la intermediación por una medida local para reducir el tiempo de ejecución en redes grandes

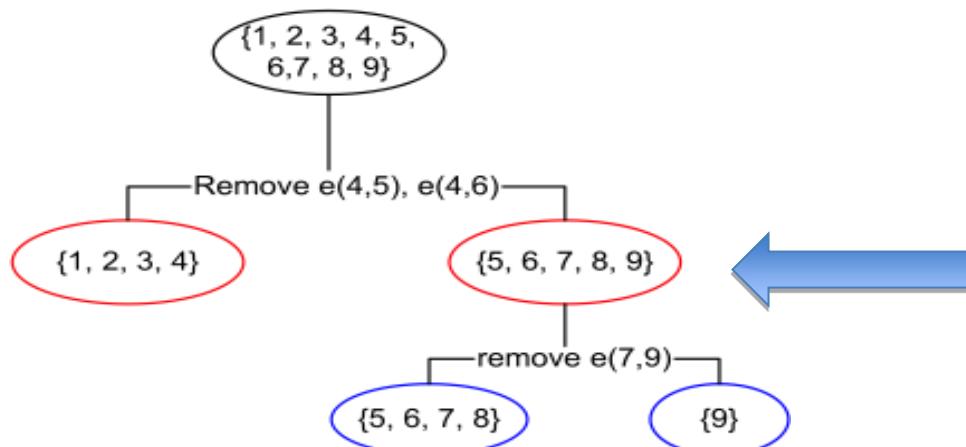
CLUSTERING DIVISIVO

Ejemplo Ilustrativo (1)



Valor inicial de intermediación

		Table 3.3: Edge Betweenness								
		1	2	3	4	5	6	7	8	9
1	1	0	4	1	9	0	0	0	0	0
2	2	4	0	4	0	0	0	0	0	0
3	3	1	4	0	9	0	0	0	0	0
4	4	9	0	9	0	10	10	0	0	0
5	5	0	0	0	10	0	1	6	3	0
6	6	0	0	0	10	1	0	6	3	0
7	7	0	0	0	0	6	6	0	2	8
8	8	0	0	0	0	3	3	2	0	0
9	9	0	0	0	0	0	0	8	0	0

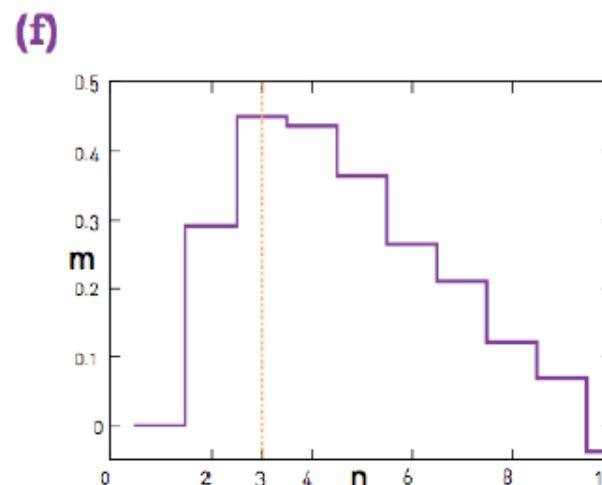
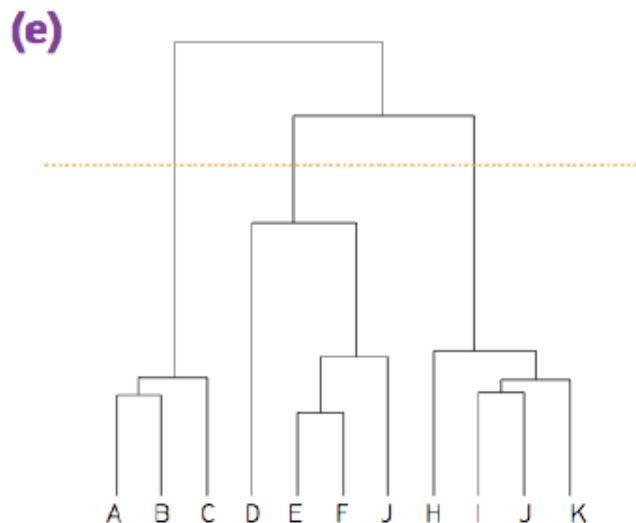
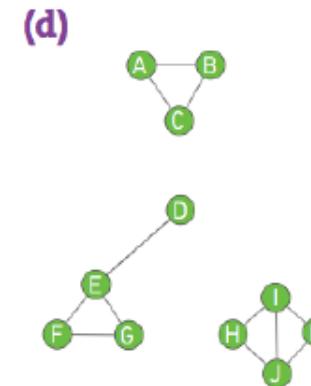
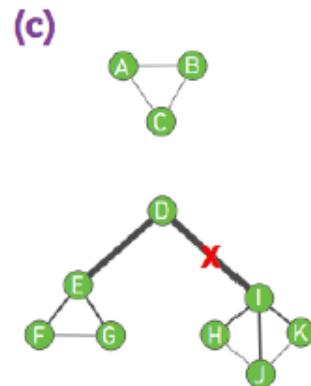
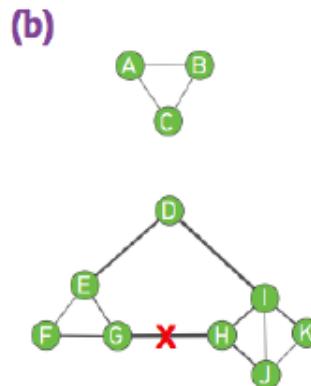
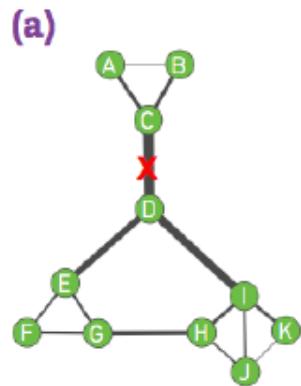


Después de eliminar el enlace $e(4,5)$, la intermediación de $e(4, 6)$ pasa a valer 20, el nuevo máximo

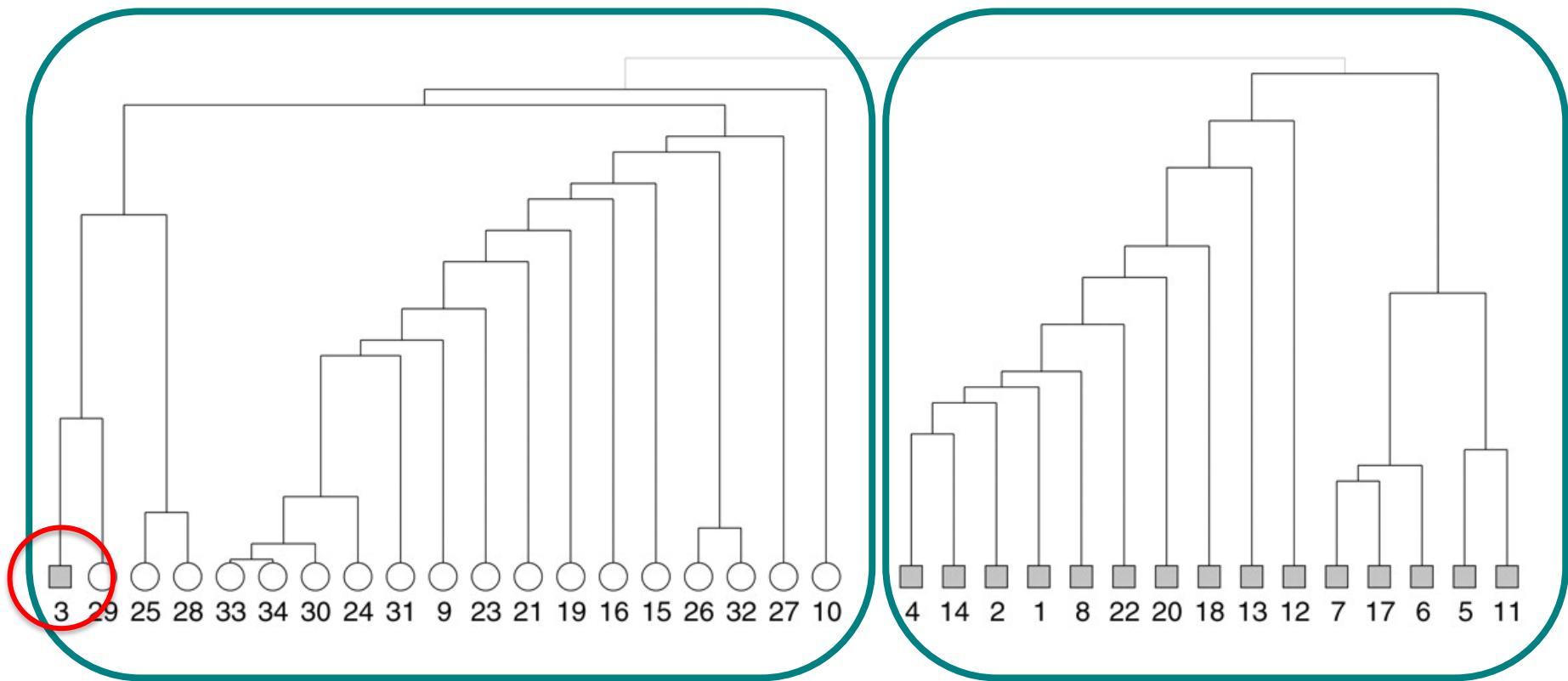
Tras eliminar $e(4,6)$, el enlace $e(7,9)$ tiene el mayor valor de intermediación (4) y debe ser eliminado

CLUSTERING DIVISIVO

Ejemplo Ilustrativo (2)



Determinación del número óptimo de comunidades calculando la modularidad para cada nivel del dendrograma

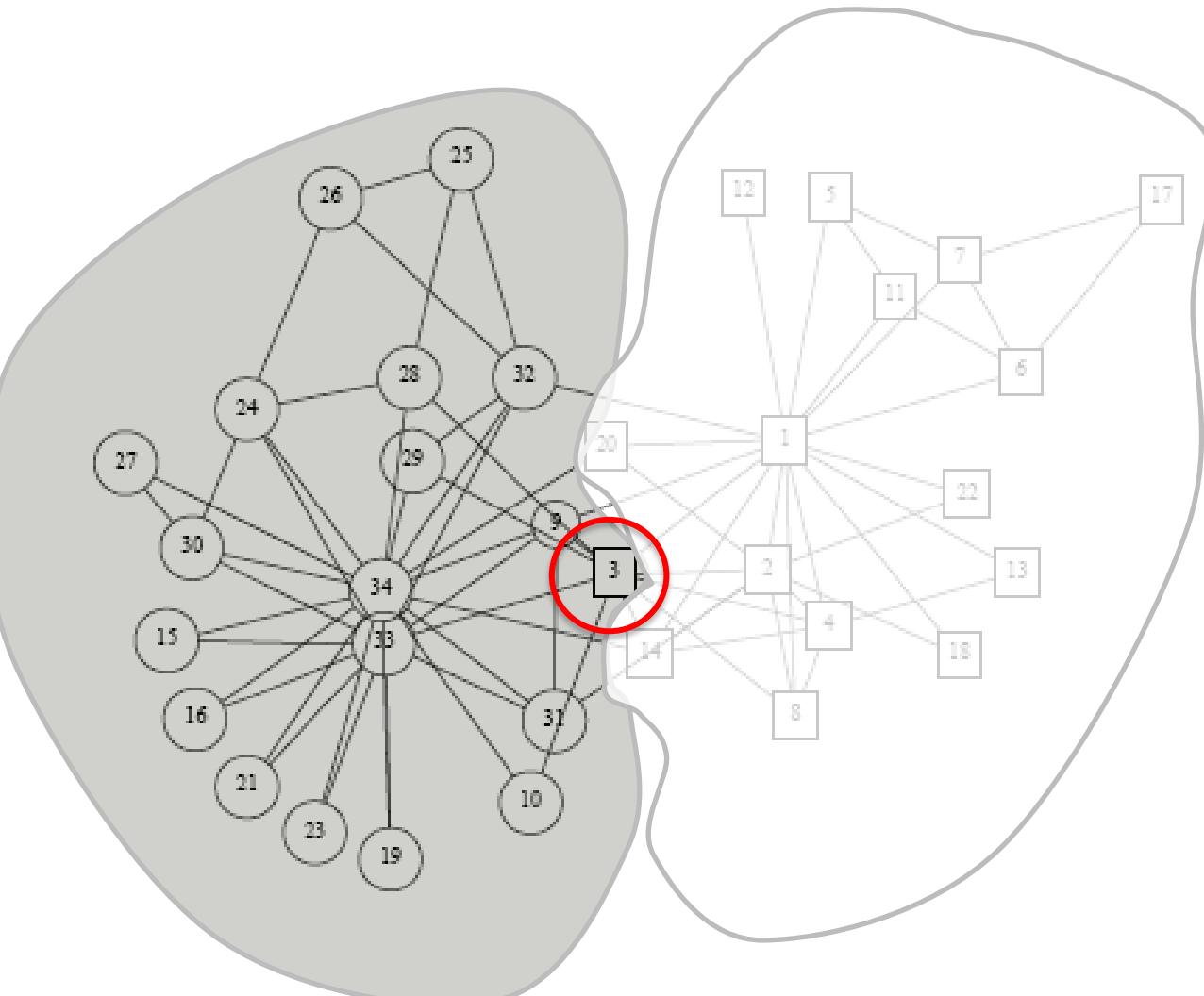


El club de karate de Zachary:

El método GN recupera lo observado sociológicamente salvo el nodo 3

CLUSTERING DIVISIVO

Ejemplo clásico (2)



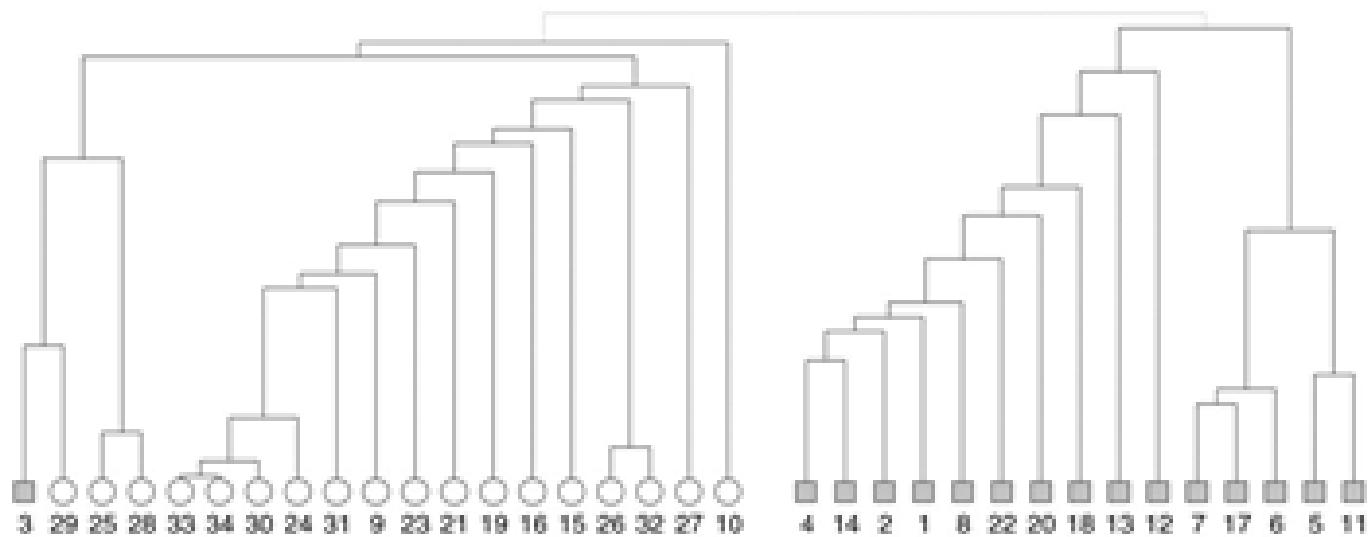
círculos → grupo del administrador (nodo 34)

cuadrados → grupo del instructor (nodo 1)

CLUSTERING DIVISIVO

Ejemplo clásico (3)

Dendrograma del
método GN



Esto no quiere decir que los métodos aglomerativos sean siempre mejores que los divisivos, depende de la medida de similitud, del tipo de fusión, de la red concreta, etc.

Dendrograma del método de clustering jerárquico aglomerativo con pesos de caminos “arista-disjuntos”



4. PARTICIONAMIENTO DE REDES COMPLEJAS

Maximización de la modularidad: Este enfoque comprende métodos heurísticos que tratan de maximizar directamente la medida de modularidad Q , incluyendo a su vez el aprendizaje del número óptimo de comunidades

Existen variantes basadas en programación lineal entera, algoritmos *greedy*, *simulated annealing*, algoritmos genéticos, etc.

Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E* 69: 066133 (2004)

Blondel y otros. Fast unfolding of communities in large networks. *J Stat Mech: Theory Exp* 10: P10008 (2008)

Guimera y otros. Modularity from fluctuations in random graphs and complex networks. *Phys. Rev. E* 70: 025101 (2004)

Tasgin y Bingol. Community Detection in Complex Networks using Genetic Algorithm. *arXiv:cond-mat/0604419v1* (2006)

Newman. Fast algorithm for detecting community structure in networks. Phys. Rev. E 69, 066133 (2004)

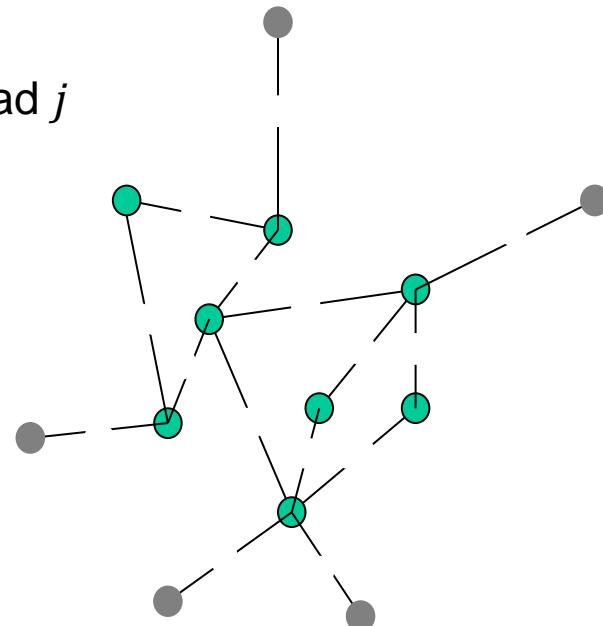
En una red $G=(V,E)$ se espera que cualquier comunidad $U_i \subset V$ tenga más conexiones internas que hacia el exterior, es decir, que:

$$2 \cdot |E(U_i, U_i)| > |E(U_i, N - U_i)|, \forall i$$

$E(U_i, U_j)$ es la mitad de la fracción de enlaces de la red que conectan nodos de la comunidad i con nodos de la comunidad j

En la misma comunidad, el número total de enlaces internos es $E(U_i, U_i) + E(U_i, U_i) = 2 \cdot E(U_i, U_i)$

$\sum_i 2 \cdot E(U_i, U_j)$ es la fracción total de enlaces internos de la red. El resto son externos. **Cuanto más se acerque a 1, mejor es la estructura de comunidades**



Si $k(U_i) = \sum_{l \in U_i} k_l$ es la suma de los grados de los nodos de una comunidad U_i , la probabilidad de que un extremo de un enlace esté en U_i es $\frac{k(U_i)}{2 \cdot L}$ y la de que ambos extremos estén en U_i (enlace interno) es $\left(\frac{k(U_i)}{2 \cdot L}\right)^2$

La modularidad compara el ratio de enlaces de cada comunidad U_i con el esperable en una red aleatoria con la misma suma de grados. Se puede reformular como:

$$Q(U_1, \dots, U_m) = \sum_{i=1}^m \left[\frac{|E(U_i, U_i)|}{L} - \left(\frac{k(U_i)}{2 \cdot L} \right)^2 \right]$$

ratio real de
enlaces de U_i

ratio
esperable

Si U_i fuera realmente una comunidad, debería haber un exceso. Se puede demostrar que si se cumple $2 \cdot |E(U_i, U_i)| > |E(U_i, N - U_i)|, \forall i$, entonces $Q > 0$

La ventaja de esta formulación es que está factorizada, se calcula a nivel de cada comunidad individual, acelerando el cálculo en métodos de optimización local

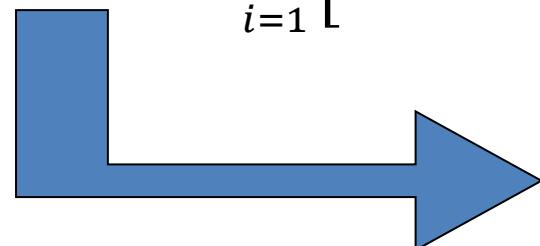
La medida se extiende fácilmente a redes ponderadas. Se define la “fuerza” (*strength*) de un nodo como la suma de los pesos de sus enlaces:

$$s_i = \sum_j w_{ij}$$

Esa fuerza generaliza la noción de grado. De este modo, se considera la suma de los pesos de las aristas entre dos comunidades en lugar de considerar la cantidad:

$$f(U_i, U_j) = \sum_{e \in E(U_i, U_j)} w_e \quad s(U_i) = \sum_{j \in U_i} s_j \quad S = \frac{1}{2} \sum_{j \in V} s_j$$

$$Q'(U_1, \dots, U_m) = \sum_{i=1}^m \left[\frac{|E(U_i, U_i)|}{L} - \left(\frac{k(U_i)}{2L} \right)^2 \right]$$



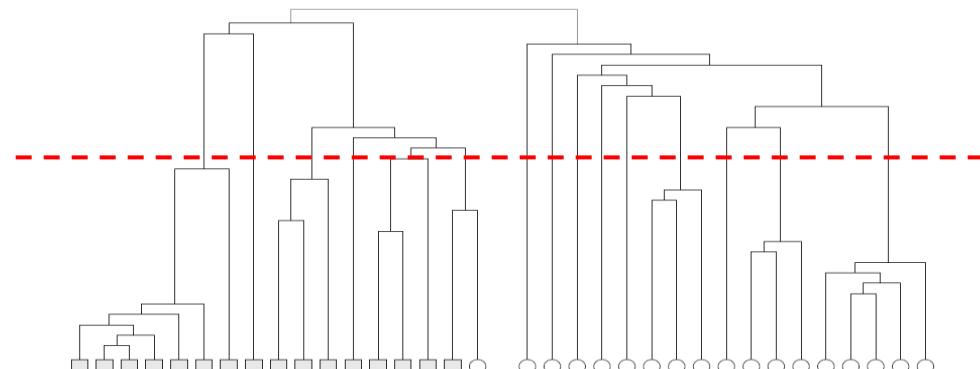
$$Q^w(U_1, \dots, U_m) = \sum_{i=1}^m \left[\frac{f(U_i, U_i)}{S} - \left(\frac{s(U_i)}{2S} \right)^2 \right]$$

Newman. Fast algorithm for detecting community structure in networks. Phys. Rev. E 69: 066133 (2004)

Es un método rápido que sigue una estrategia *greedy*:

1. Comenzar con todos los nodos sueltos (N comunidades)
2. Evaluar todos las fusiones posibles de pares de comunidades y realizar la mejor posible (la que provoque un mayor incremento o un menor decremento de Q)
3. Parar cuando todos los nodos estén incluidos en una sola comunidad

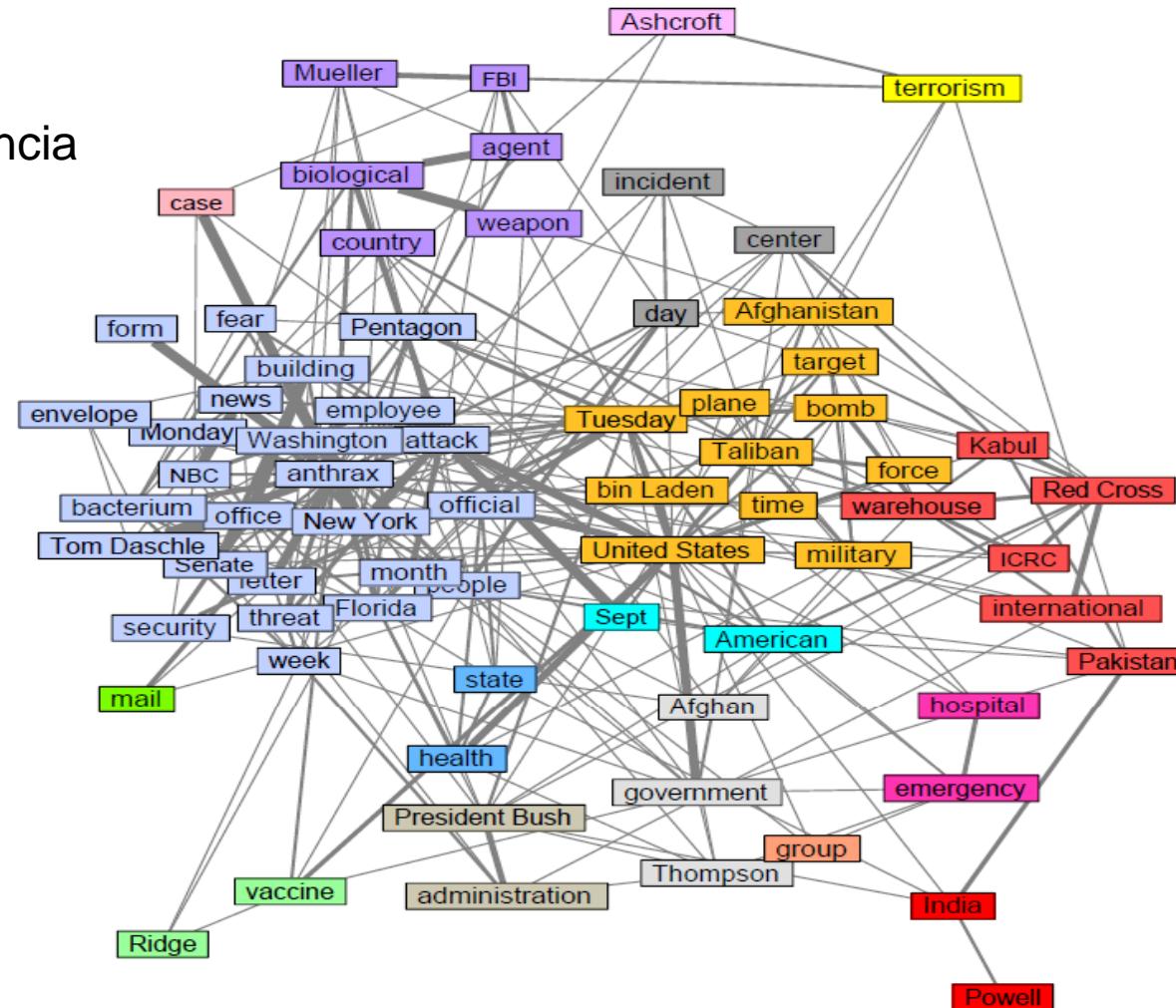
El método genera una jerarquía de comunidades. Al final se devuelve la partición con mejor valor de Q



- El algoritmo requiere $N-1$ pasos. En cada uno se busca el par que proporciona la mejor fusión de entre todos los posibles
- Sólo se consideran aquellas comunidades que comparten enlaces, el resto no provocan mejora en Q . El número de pares posibles está acotado por $|E|$
- El cambio de Q (ΔQ) sólo implica a dos comunidades y se calcula en un tiempo constante. La actualización de la matriz de adyacencia se hace en orden $O(N)$
- Por tanto, el algoritmo es de orden $O(|E|+N)$, $O(N^2)$ para redes dispersas. Clauset y otros usan *heaps* y otros trucos para obtener una implementación $O(N \cdot \log N)$.
- Además, la elección de la mejor partición se hace directamente ya que el valor de Q se calcula en cada paso
- En general este algoritmo no es especialmente bueno. Su ventaja es la eficiencia y sirve como primer indicador de presencia o ausencia de modularidad

EJEMPLO DE APLICACIÓN:

- Red ponderada de coocurrencia de palabras en noticias de Reuters, Octubre de 2001.
- El peso es la frecuencia de coocurrencias



- Otro algoritmo que incorpora características de métodos aglomerativos (devuelve una jerarquía de comunidades) y que obtiene buenos resultados en la práctica
- Es muy eficiente, puede manejar redes de hasta **100 millones de nodos**. El análisis de una red de **2 millones de nodos** sólo requiere **2 minutos** en un PC estándar
- También funciona en redes ponderadas y no ponderadas
- Se basa en un enfoque *greedy* que considera la optimización local de la medida de modularidad **hasta que no se produce ninguna mejora**
- Está implementado en softwares de redes complejas como **Gephi** y **NetworkX**

Blondel y otros. Fast unfolding of communities in large networks. J Stat Mech: Theory Exp 10: P10008 (2008)

<http://perso.uclouvain.be/vincent.blondel/research/louvain.html>

- Cada iteración incluye dos pasos: uno de **optimización**, donde se mejora localmente la modularidad, y otro de **agregación**, en el que se “**colapsan**” las comunidades a mega-nodos
- La etapa de optimización es una ascensión de colinas que intenta mover nodos entre comunidades mientras que Q vaya aumentando
- El algoritmo comienza con n comunidades, una por nodo, y aplica el proceso iterativo:

REPETIR

Paso de optimización

Paso de agregación

MIENTRAS (haya mejora en Q)

PASO DE OPTIMIZACIÓN:

REPETIR

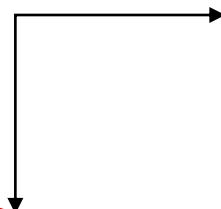
\forall nodo i

\forall nodo j vecino de i

 Calcular $b_j = \Delta Q$ tras cambiar i a la comunidad de j

 Si $\max b_j > 0$, cambiar i a la comunidad del j que maximiza b_j

MIENTRAS (haya mejora en Q)



Es rápido por los mismos motivos del método *greedy* de Newman

- En cada paso del algoritmo, intentamos cambiar la asignación de cada nodo a la comunidad de sus nodos vecinos y aplicamos el mejor cambio posible

PASO DE OPTIMIZACIÓN:

- Se codifica la solución al problema (partición de comunidades) en forma de un vector (lista de nodos) en el que el índice (**la posición**) corresponde al nodo y **el contenido** a la comunidad a la que se asigna. Ej:

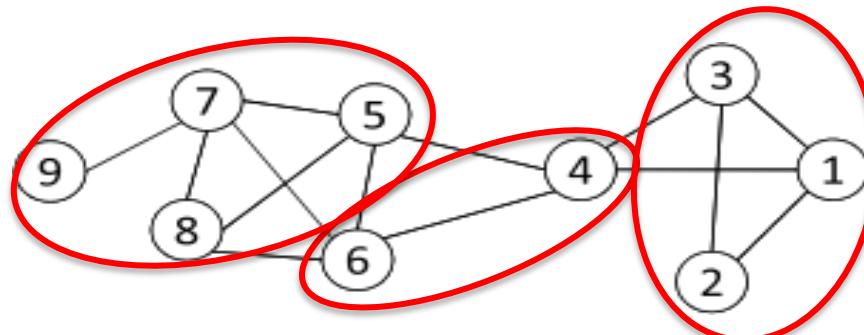
$$P_1 = [1 \text{ } 1 \text{ } 1 \text{ } 2 \text{ } 3 \text{ } 2 \text{ } 3 \text{ } 3 \text{ } 3]$$

- Este vector corresponde a una red de 9 nodos particionada en 3 comunidades:

$$C_1 = \{1,2,3\}$$

$$C_2 = \{4,6\}$$

$$C_3 = \{5,7,8,9\}$$



PASO DE OPTIMIZACIÓN:

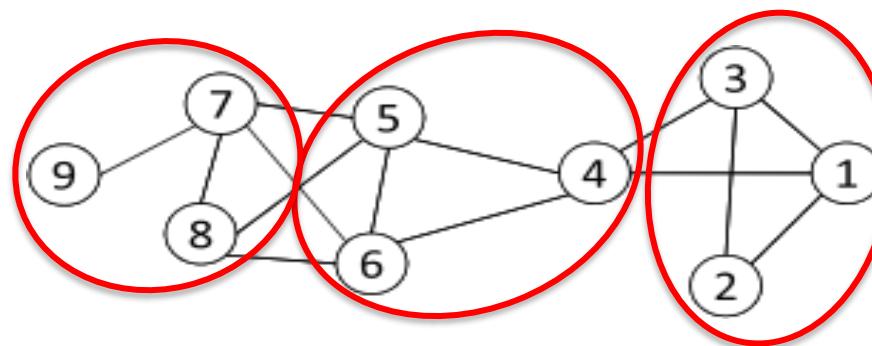
- A partir de una partición concreta, se puede generar otra alternativa cambiando la asignación de un nodo a otra comunidad. Ej: nodo 5 a la comunidad 2:

$$P_2 = [\begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & 1 & 1 & 2 & 2 & 2 & 3 & 3 & 3 \end{matrix}]$$

$$C_1 = \{1, 2, 3\}$$

$$C_2 = \{4, 5, 6\}$$

$$C_3 = \{7, 8, 9\}$$

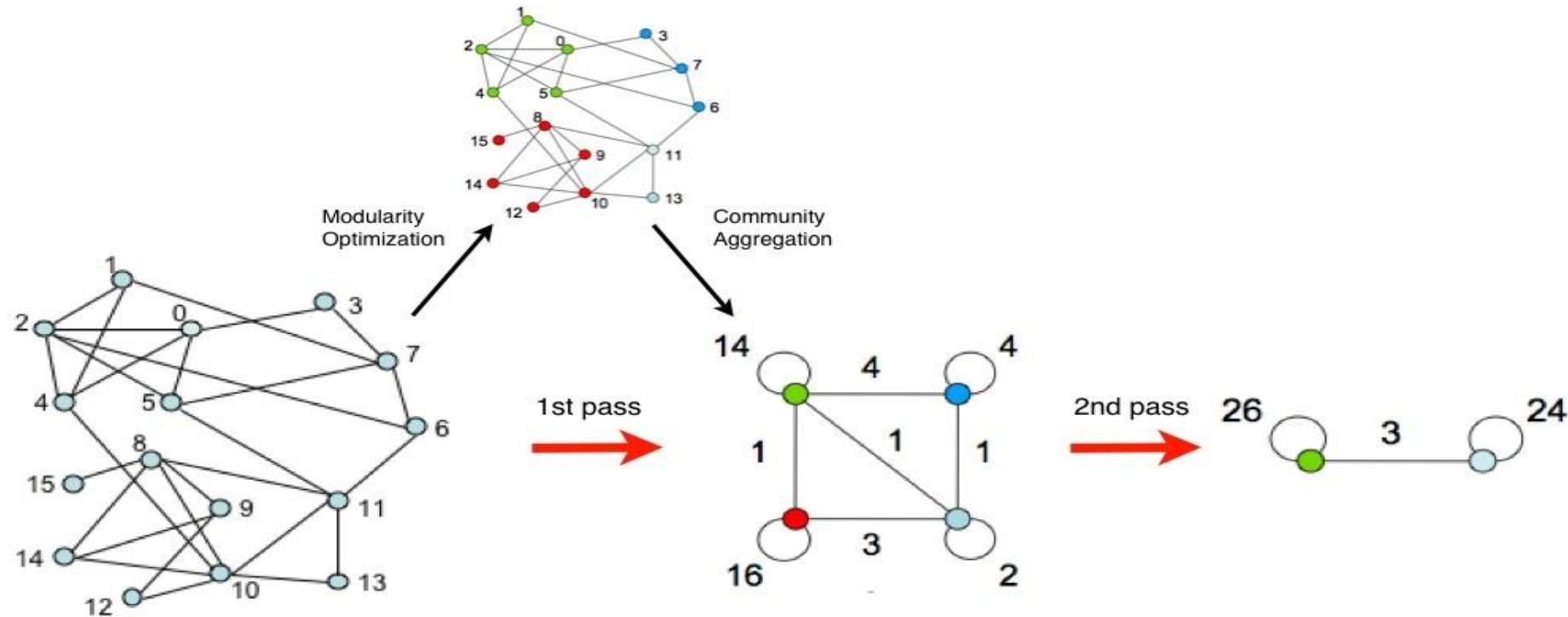


- ¿Cuál es la mejor partición de las dos? La que tenga mayor valor de modularidad Q . Si la P_1 tiene un valor $Q = 0.32$ y la P_2 tiene $Q = 0.38$, la mejor partición sería la segunda

PASO DE AGREGACIÓN:

- Crea una red ponderada con un nodo i por cada comunidad U_i de la red previa
- Incluye un enlace para cada par de comunidades U_i, U_j que contengan al menos un par de nodos que estuviesen enlazados en la red previa
- Pondera dicho enlace con un peso w_{ij} equivalente a la suma de los pesos de aristas entre U_i y U_j en la red previa

EJEMPLO ILUSTRATIVO:



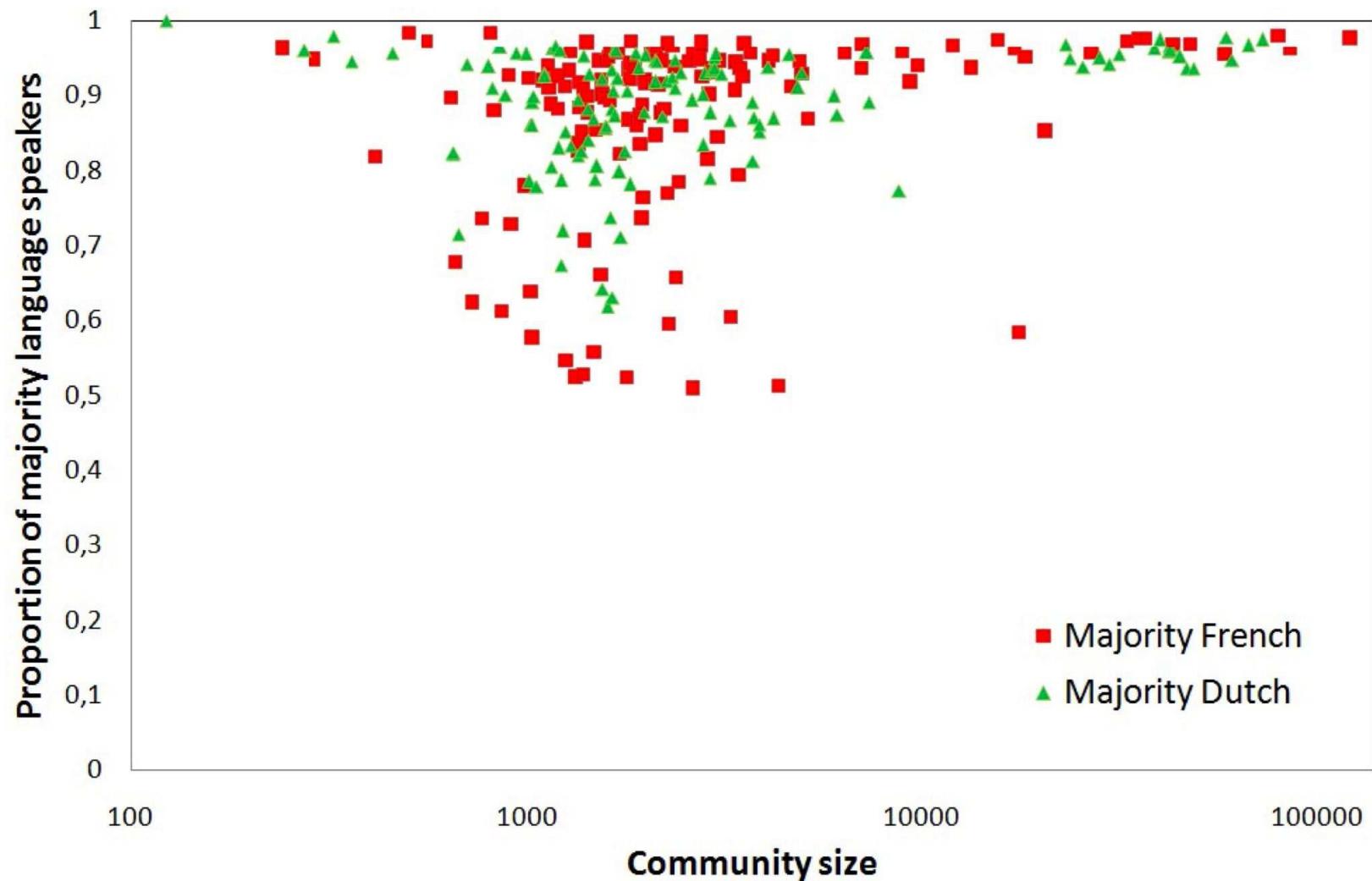
- El método tiene un tiempo de ejecución “empírico” $O(N \cdot \log N)$
- Obtiene buenos valores de Q . En redes pequeñas, donde la ejecución de otros algoritmos más lentos es viable, proporciona valores competitivos
- La partición obtenida en el primer paso suele incluir muchas comunidades de pequeño tamaño. En pasos sucesivos, se van generando comunidades cada vez más grandes gracias al mecanismo de agregación
- Así se obtiene una descomposición jerárquica de la red de forma natural. **El algoritmo no termina en una única comunidad**, como los métodos de clustering aglomerativo, sino que para cuando no hay mejora en Q
- Por tanto, **es capaz de aprender el número de comunidades**

EJEMPLO DE APLICACIÓN:

- Red social ponderada de 2.6 millones de teléfonos móviles en Bélgica
- El peso es la cantidad de llamadas entre los números durante 6 meses. Los datos de los suscriptores incluían su idioma (principalmente, **francés** y **flamenco**, aunque también hay inglés y alemán)
- Por tanto, la partición obtenida se puede validar estudiando la homogeneidad lingüística de las comunidades
- El método de Lovaina obtiene una partición de 6 niveles. En el más alto, hay 261 comunidades de más de 100 usuarios (un 75% del total)
- Hay 36 comunidades con más de 10,000 usuarios. Salvo una comunidad que hace de interfaz, todas tienen al menos una homogeneidad del 85%

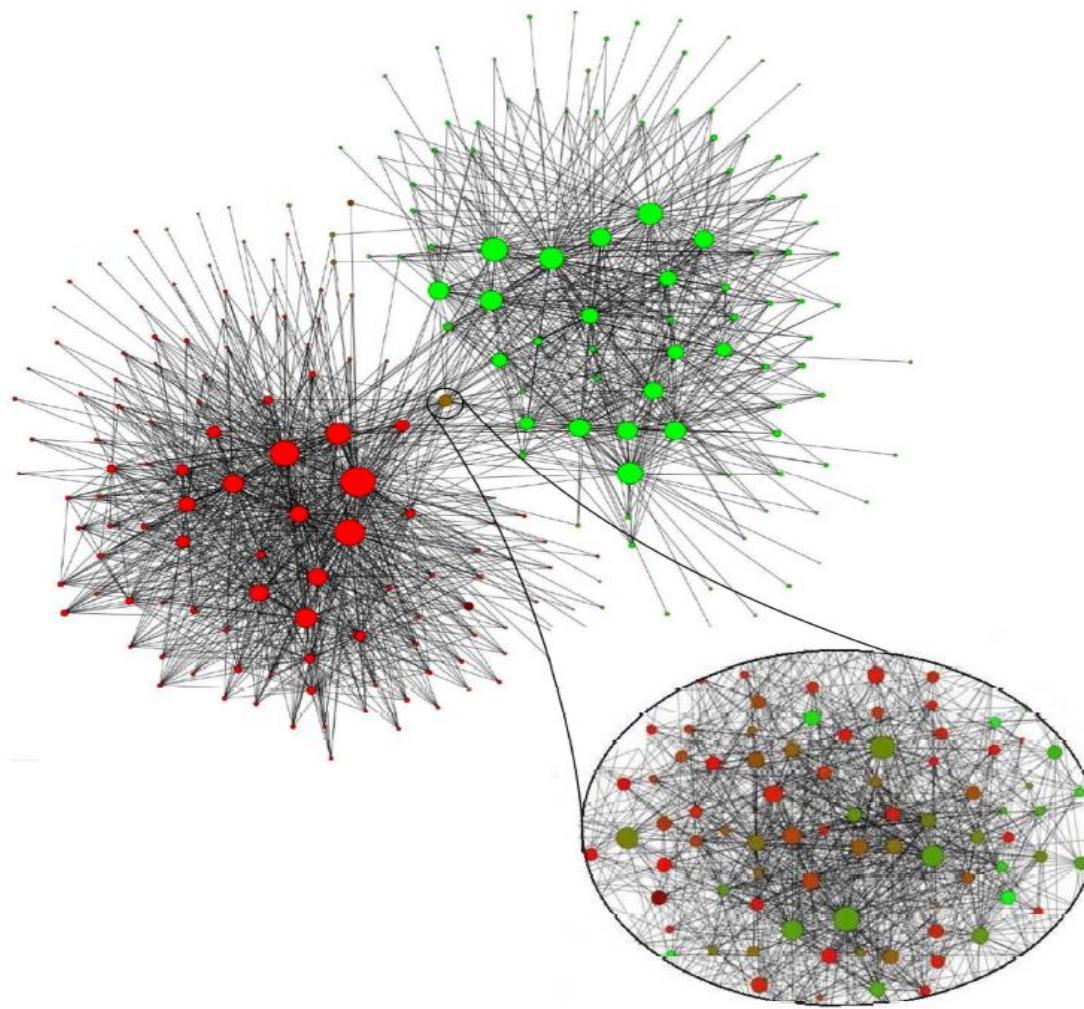
MAXIMIZACIÓN DE LA MODULARIDAD

Ejemplo real (2)



MAXIMIZACIÓN DE LA MODULARIDAD

Ejemplo real (3)



- Las 261 comunidades de primer nivel (**rojo = francés** y **verde = flamenco**)
- El tamaño de los nodos es proporcional al nº de usuarios
- La gradación de color muestra el idioma mayoritario
- La comunidad intermedia está compuesta por otras más pequeñas con una separación lingüística menos clara
- El método es **multi-resolución** y permite analizarla fácilmente

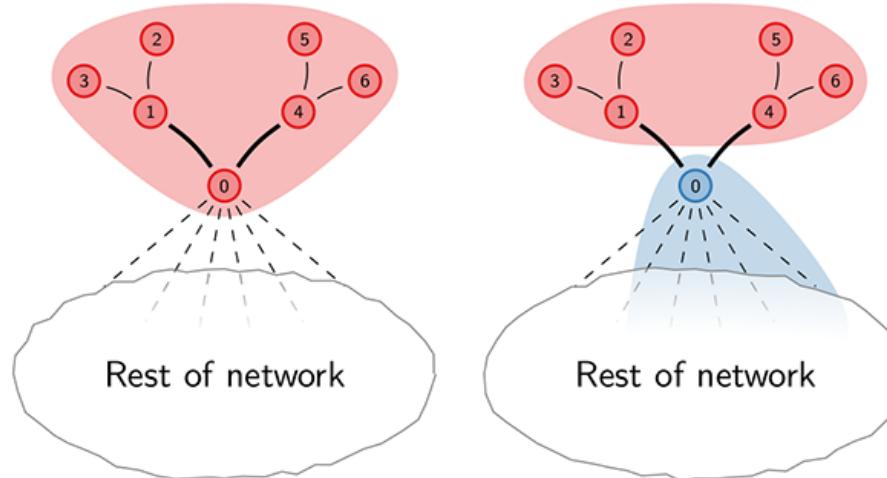
- El método de Leiden es una modificación reciente del de Lovaina
- Se basa en la corrección de un fallo de Lovaina, que puede generar comunidades erróneas en las que se incluyan nodos que no tienen enlaces entre sí
- Además, la pequeña modificación realizada hace que el algoritmo de Leiden sea más rápido que el de Lovaina

Traag, Waltman y van Eck. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports* 9: 5233 (2019)

<http://ars-uns.blogspot.com/2019/11/comunidades-el-algoritmo-de-leiden.html>

DEFECTO DEL MÉTODO DE LOVAINA:

- El problema se produce con nodos que hacen de porteros para el resto de nodos de su comunidad. Sin ellos, los demás no estarían conectados



- Lovaina mueve nodos de un grupo a otro de forma continua, con lo que puede mover el nodo clave a un grupo diferente, rompiendo la conectividad de la comunidad original y no siendo capaz de restaurarla

SOLUCIÓN DEL MÉTODO DE LEIDEN:

- El algoritmo de Leiden puede dividir grupos en lugar de solo fusionarlos, como hace el algoritmo de Lovaina
- Al dividir los grupos de una manera específica, el algoritmo de Leiden garantiza que las comunidades estén bien conectadas
- En vez de tratar de mover todos los nodos de la red como Lovaina, Leiden solo trata de cambiar de grupo los llamados *nodos inestables*
- Como resultado, el método de Leiden no solo encuentra clústeres de mayor calidad que el de Lovaina, sino que además lo hace en mucho menos tiempo
- De hecho, se supone que garantiza el “óptimo” si se ejecuta varias veces

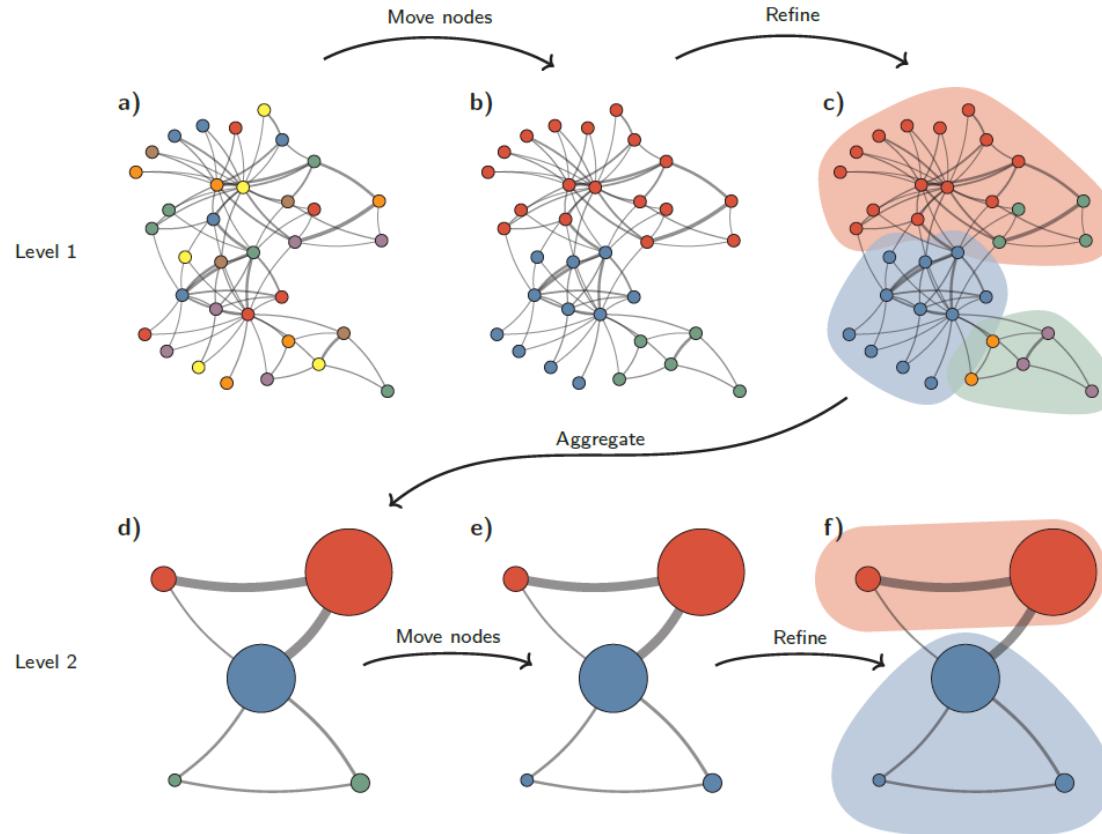
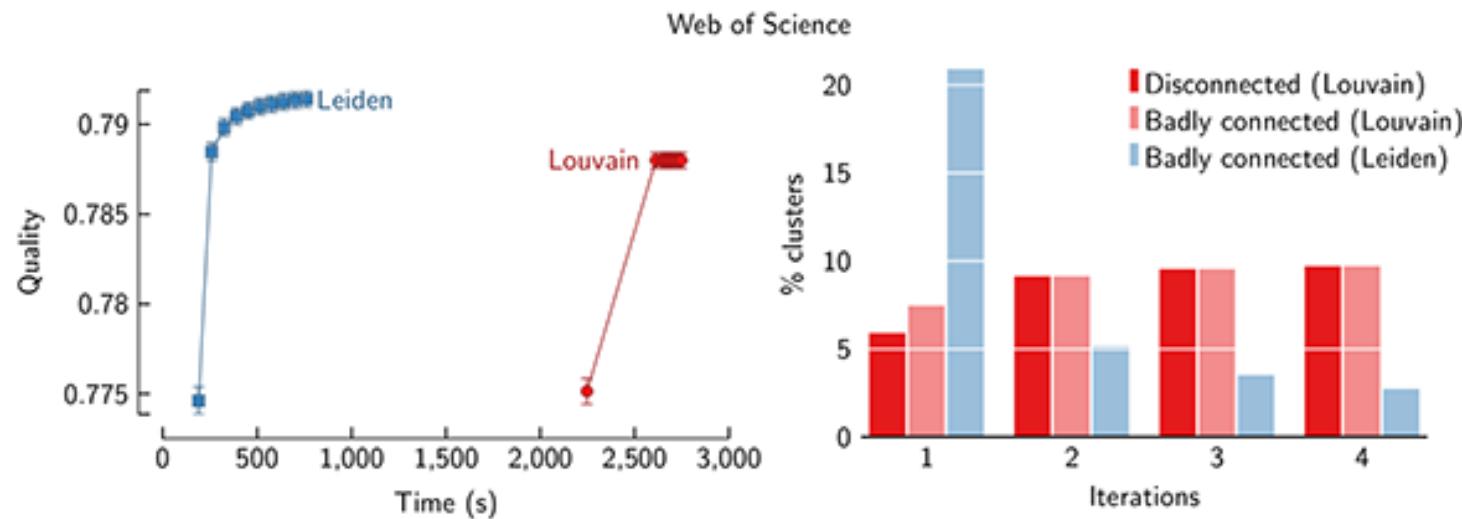


FIG. 3. Leiden algorithm. The Leiden algorithm starts from a singleton partition (a). The algorithm moves individual nodes from one community to another to find a partition (b), which is then refined (c). An aggregate network (d) is created based on the refined partition, using the non-refined partition to create an initial partition for the aggregate network. For example, the red community in (b) is refined into two subcommunities in (c), which after aggregation become two separate nodes in (d), both belonging to the same community. The algorithm then moves individual nodes in the aggregate network (e). In this case, refinement does not change the partition (f). These steps are repeated until no further improvements can be made.

COMPARATIVA DE LOS MÉTODOS DE LOVAINA Y LEIDEN:

- Red de citación de artículos científicos: 10 millones de artículos y 200 millones de enlaces de citas
- Tiempo de ejecución de Leiden: poco más de tres minutos

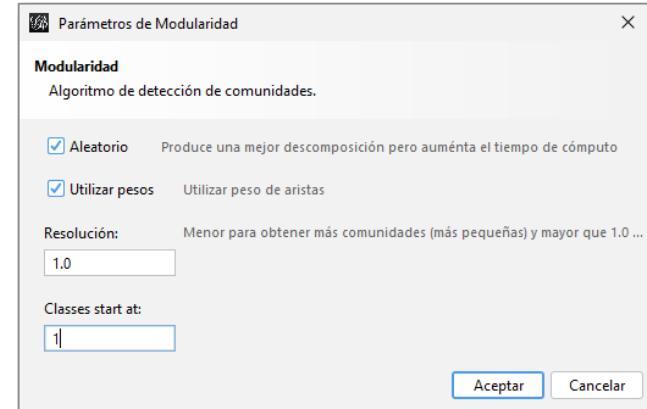


IMPLEMENTACIONES DE MÉTODOS DE DETECCIÓN DE COMUNIDADES

MÉTODO DE LOVAINA

- **Dos implementaciones de los autores en C++** están disponibles en <http://sites.google.com/site/findcommunities/>. También hay código en Matlab

- **Gephi** incluye una implementación (<https://github.com/gephi/gephi/wiki/Modularity>). Permite pesos y la asignación de auto-enlaces para incrementar o reducir la aversión de los nodos a formar comunidades



- **NetworkX** (<http://perso.crans.org/aynaud/communities/>):

```
community.best_partition(graph, partition=None)
```

Compute the partition of the graph nodes which maximises the modularity (or try..) using the Louvain heuristics

This is the partition of highest modularity, i.e. the highest partition of the dendrogram generated by the Louvain algorithm.

```
community.modularity(partition, graph)
```

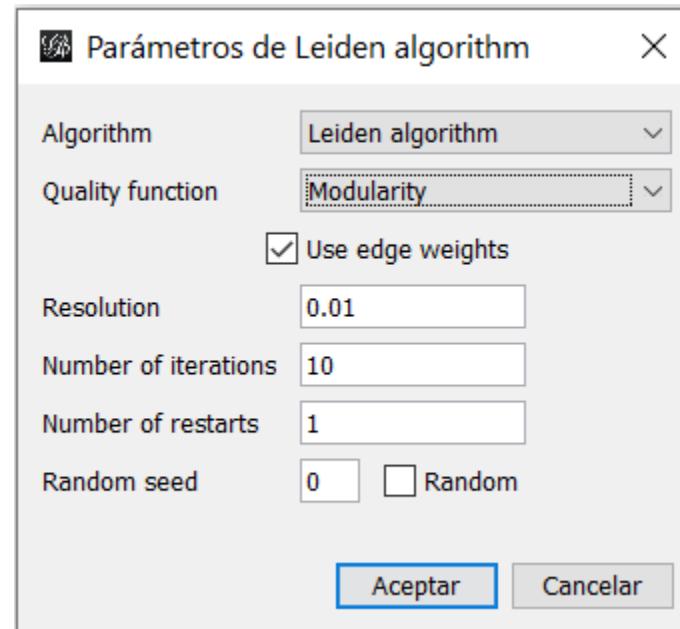
Compute the modularity of a partition of a graph

Devuelve la partición que maximiza la modularidad.
Considera pesos y dirección en los enlaces

Calcula la modularidad de una partición

MÉTODO DE LEIDEN

- **Implementación de los autores en Java** disponible en GitHub:
<https://github.com/CWTSLeiden/networkanalysis>
- Existe también un plugin de la comunidad de **Gephi**:



MÉTODO GREEDY DE NEWMAN

- **Implementación de los autores en C++** para redes ponderadas y no ponderadas disponible en <http://cs.unm.edu/~aaron/research/fastmodularity.htm>
- Hace uso de estructuras de datos eficientes basadas en la dispersión de la matriz de adyacencia para acelerar la ejecución

The screenshot shows the homepage of the website [Structure * STRANGENESS](http://cs.unm.edu/~aaron/research/fastmodularity.htm). The page features a red header bar with the title. Below it is a large logo with the word "Structure" in red script and "STRANGENESS" in black sans-serif font, separated by a blue asterisk (*). A horizontal bar with three segments is positioned between the logo and the main content area. The main content area has a white background with a thin red border. It contains several sections of text and links. On the right side, there are three vertical columns of links under headings: "creative", "persona", and "website". At the bottom right, there is a copyright notice.

Structure *
STRANGENESS

"Fast Modularity" Community Structure Inference Algorithm

This page documents and supports the fast modularity maximization algorithm I developed jointly with [Mark Newman](#) and [Cristopher Moore](#). This algorithm is being widely used in the community of complex network researchers, and was originally designed for the express purpose of analyzing the community structure of extremely large networks (i.e., hundreds of thousand or millions of vertices). The original version worked only with unweighted, undirected networks. I've recently posted a version that works on weighted, undirected networks.

Update February 2007: Please see [my recent blog entry](#) about this algorithm.

Update May 2007: See also [the igraph library](#), a pretty comprehensive library of network utility functions (generation and analysis), including an implementation of the **fast modularity** algorithm described here (along with a few other nice clustering heuristics).

Update October 2008: I've finally gotten around to posting a version that works with weighted networks, which is available [here](#). This version wants a .wpairs file as input, which is an edge list with integer weights, e.g., "54\t91\t3\n" would be an edge with weight 3. Otherwise, it should work just the same as the unweighted version.

creative

[Photography](#) ::
 :: Artistic ::
 :: Blog ::
 :: Thinking ::
 :: Research ::

persona

[About](#) ::
 :: .plan ::
 :: Vitae ::

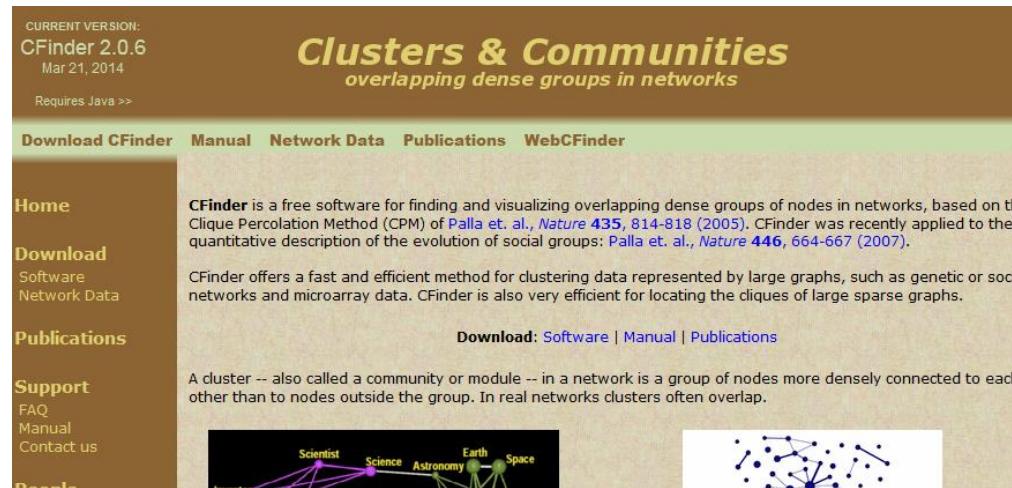
website

[Search](#) ::
 :: Copyright ::
 :: Sitemap ::
 :: Links ::

© Aaron Clauset

MÉTODO CLIQUE PERCOLATION METHOD

- **CFinder: software de los autores** para redes ponderadas y no ponderadas, dirigidas y no dirigidas, estáticas y dinámicas disponible en <http://www.cfinder.org/>
- Multiplataforma: Windows, Linux y Mac. Se pueden descargar varias redes reales: Wikipedia, Red científica de coautoría y red de páginas web de Google



- **NetworkX** también tiene una implementación:

`k_clique_communities (G, k[, cliques])`

Find k-clique communities in graph using the percolation method.

IMPLEMENTACIONES DE DISTINTOS MÉTODOS

<https://sites.google.com/site/andrealancichinetti/codes>

- Códigos *Python* de Andrea Lancichinetti. Incluyen varios métodos conocidos junto con su método de clustering por consenso y otro suyo reciente (OSLOM), con variantes para redes dirigidas y no dirigidas:

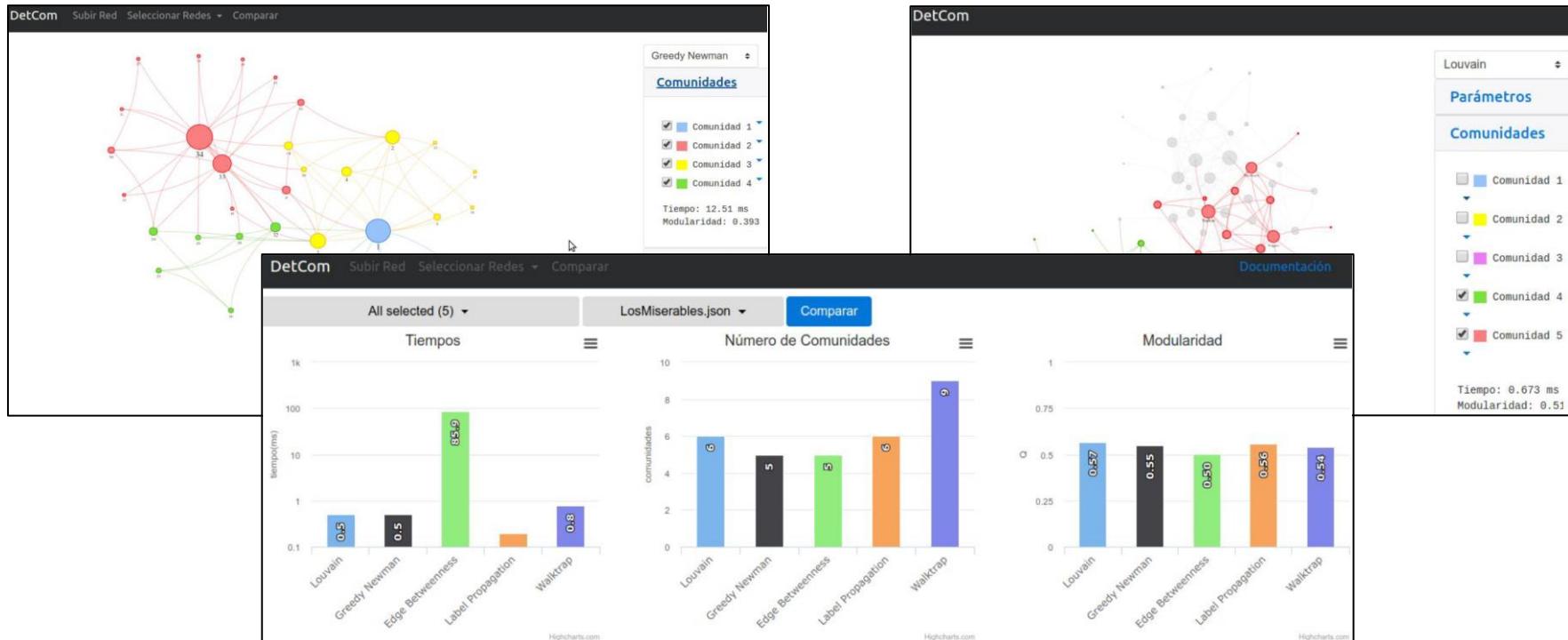
Lancichinetti y Fortunato. Consensus clustering in complex networks. *Scientific Reports* 2: 336 (2012)

- Los métodos considerados son: una implementación de Simulated Annealing para optimización de la modularidad, *Infomap* (método jerárquico dinámico basado en *random walks*), OSLOM, el método de Lovaina, y el método *Label Propagation*
- Incluye código para visualizar comunidades separadas y para generar redes de *benchmark*
- Todos soportan el mismo formato de entrada (lista de enlaces). Genera salidas para Gephi

HERRAMIENTA CON IMPLEMENTACIONES DE DISTINTOS MÉTODOS

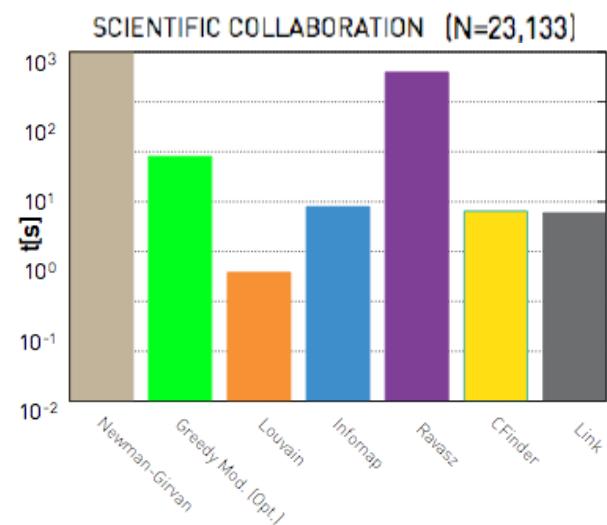
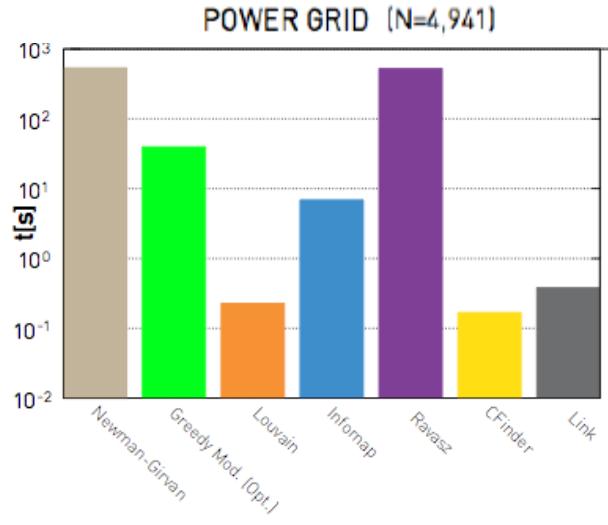
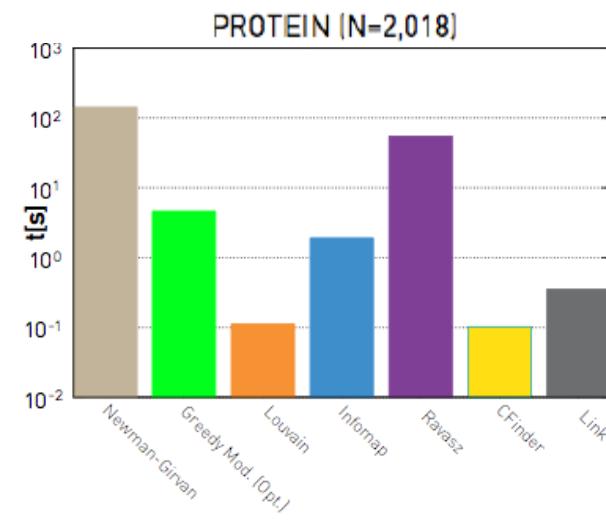
<https://github.com/fnavales/TFG-community-detection-in-graphs>

- Herramienta ***DetCom*** desarrollada por Fran Navarro, UGR. Incluye cinco métodos distintos. Permite comparar algoritmos, visualizar las redes e interactuar con las comunidades, maneja ficheros en formato .json y Gephi:



ESTUDIO COMPARATIVO DE EFICIENCIA

NAME	NATURE	COMP.
Ravasz	Hierarchical Agglomerative	$O(N^3)$
Girvan-Newman	Hierarchical Divisive	$O(N^3)$
Greedy Modularity	Modularity Optimization	$O(N^2)$
Greedy Modularity	Modularity Optimization	$O(N \log^2 N)$
Louvain	Modularity Optimization	$O(N \log N)$
Infomap	Flow Optimization	$O(N \log N)$
Clique Percolation	Overlapping Communities	$Exp(N)$
Link Clustering CFinder	Hierarchical Agglomerative; Overlapping Communities	$O(N^2)$



Referencias y Agradecimientos

Para diseñar los materiales de este tema, he hecho uso de material desarrollado por expertos en el área disponible en Internet:

- “Network Science Interactive Book Project” del Laszlo Barabasi Lab:
Roberta Sinatra: <http://barabasilab.com/networksciencebook>
- Curso on-line “Social Network Analysis” de Lada Adamic, Coursera,
Universidad de Michigan: <https://www.coursera.org/course/sna>
- Curso “Social Networks, Media and Transfer Learning” de Qiang
Yang, Hong Kong University of Science and Technology
- Curso “Computational Molecular Biology” de My T. Thai,
Universidad de Florida

