DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

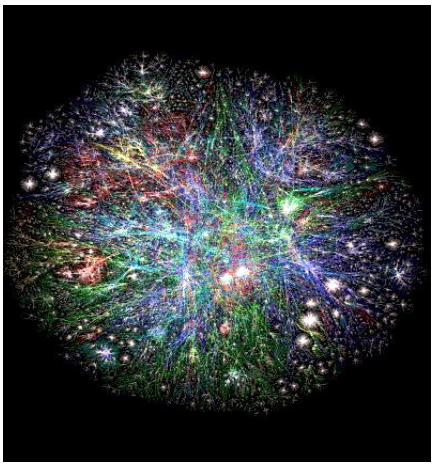
Minería de grafos

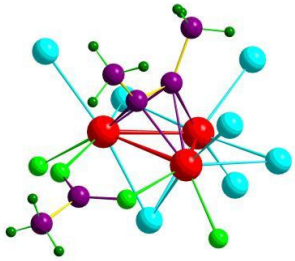
© Fernando Berzal, berzal@acm.org

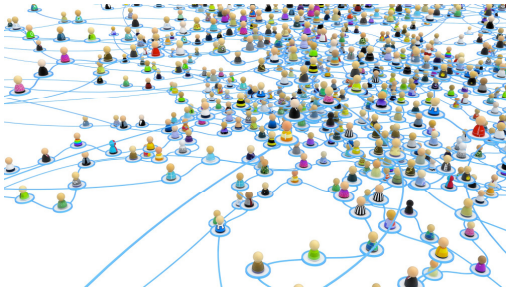
Minería de grafos

- Detección de patrones en grafos
  - Isomorfismo de (sub)grafos
  - Subgrafos frecuentes
  - Motif discovery
- Agrupamiento en grafos
  - El problema de la detección de comunidades
  - Métodos de detección de comunidades
  - Detección de comunidades con solapamiento
- Clasificación en grafos
  - El problema de la predicción de enlaces
  - Técnicas de predicción de enlaces

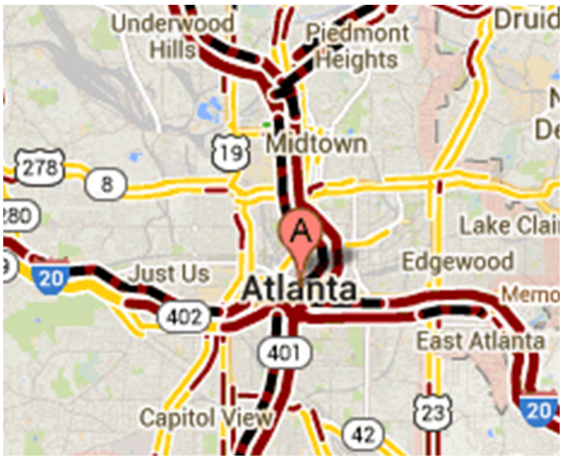
Datos

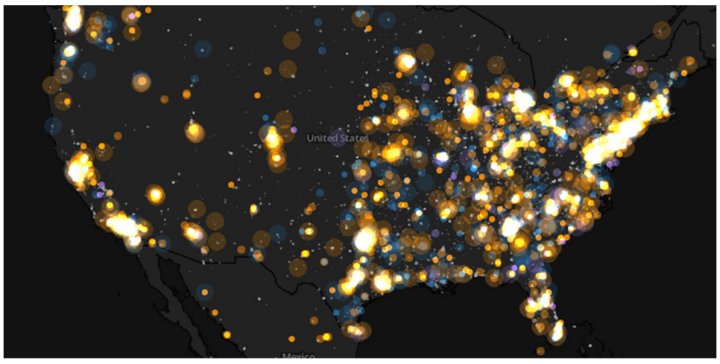






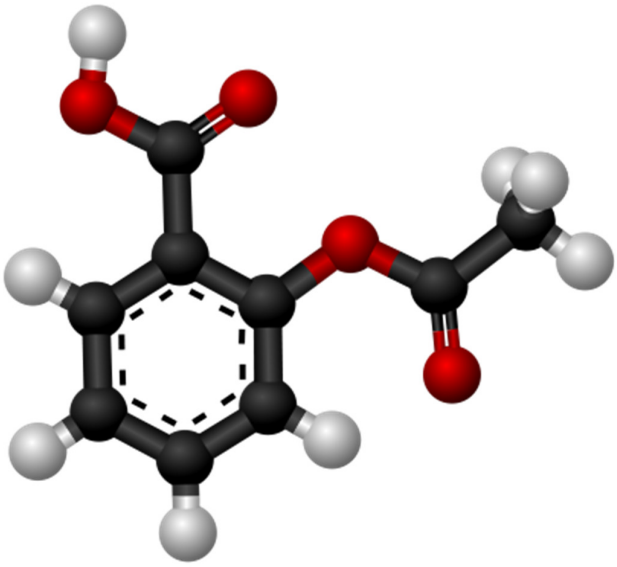
Datos

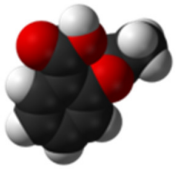

Underwood Hills
Piedmont Heights
Druid
Midtown
278
19
8
280
20
Just Us
Atlanta
Edgewood
Lake Clai
402
401
East Atlanta
Capitol View
42
23
20

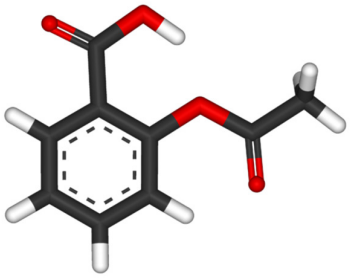


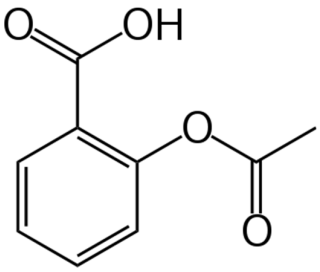
Patrones en grafos

El compuesto químico de la aspirina



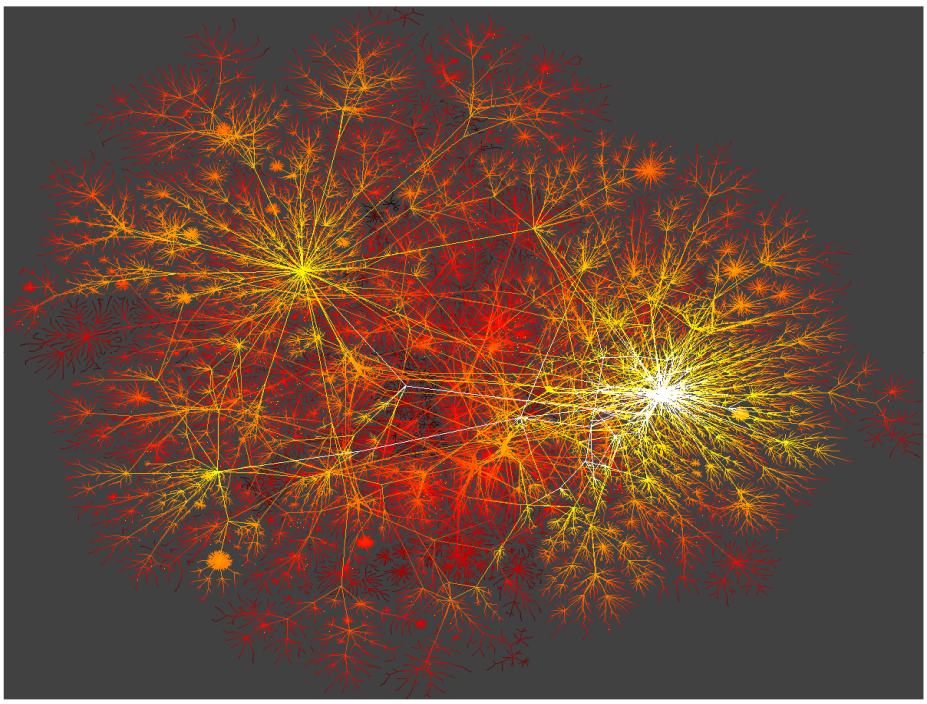






Patrones en grafos

Internet

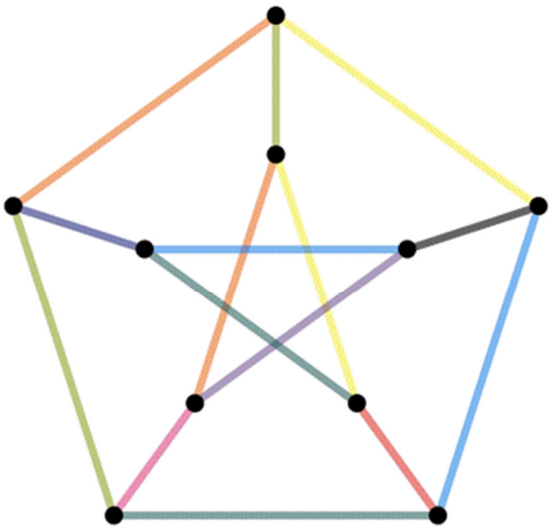


Patrones en grafos

¿Qué interés tiene encontrar patrones en grafos?

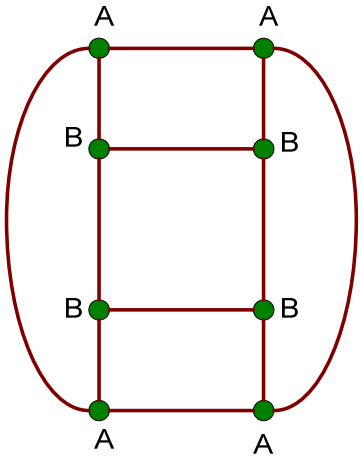
- Caracterizar conjuntos de grafos.
- Crear modelos de clasificación de grafos.
- Diseñar métodos de agrupamiento en grafos.
- Construir índices en bases de datos de grafos.

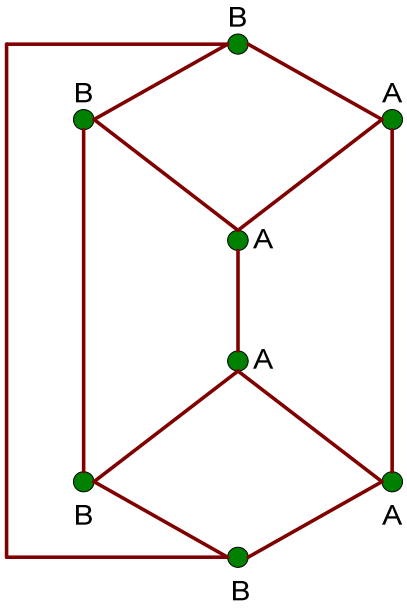
Isomorfismo de (sub)grafos



Isomorfismo de (sub)grafos

Comparar grafos implica medir la similitud entre ellos: ver hasta qué punto son isomorfos.





Dos grafos son isomorfos
si son topológicamente equivalentes

Isomorfismo de (sub)grafos

La detección de isomorfismo entre grafos es un problema NP peculiar.

- No se sabe si está en P.
- No se sabe si es NP-completo.
- ¿Clase de complejidad intermedia [GI]?

NP completo
Clique

Isomorfismo de grafos

Factorización

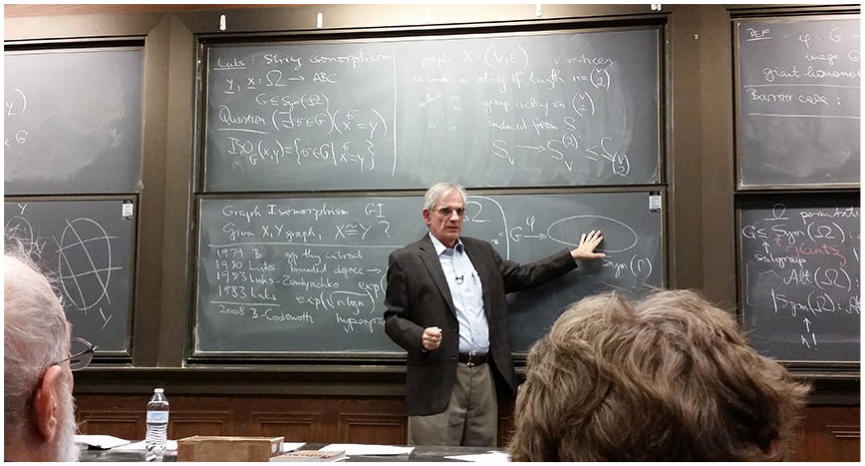
P

NOTA: Su generalización, el isomorfismo de subgrafos, sí es un problema NP-completo (reducción: clique).

Isomorfismo de (sub)grafos

Landmark Algorithm Breaks 30-Year Impasse

Chicago, 10 de noviembre de 2015 ➔ STOC'2016



NP completo
Clique

Isomorfismo de grafos

Factorización

P

Laszlo Babái: Graph isomorphism in quasipolynomial time,

O (exp (polylog (n))) = O(exp ($\log^k$ (n)))

http://people.cs.uchicago.edu/~laci/quasipoly.html

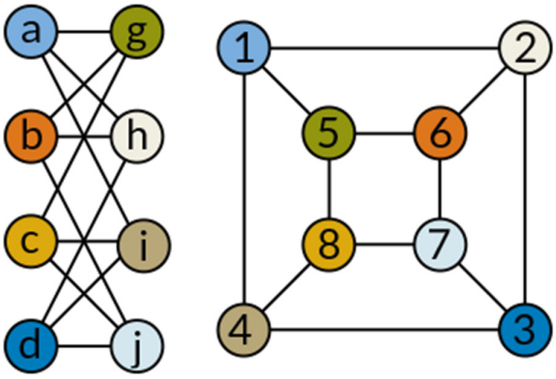
Isomorfismo de (sub)grafos

Landmark Algorithm Breaks 30-Year Impasse

Chicago, 10 de noviembre de 2015

Algoritmo divide y vencerás:

Coloreado de posibles correspondencias…



NP completo
Clique

Isomorfismo de grafos

Factorización

P

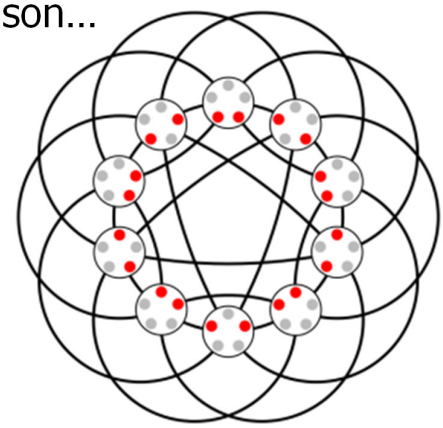
O (exp (polylog (n))) = O (exp ($(\log n)^k$))

http://people.cs.uchicago.edu/~laci/quasipoly.html

Isomorfismo de (sub)grafos

Landmark Algorithm Breaks 30-Year Impasse
Chicago, 10 de noviembre de 2015

Teniendo cuidado con un caso particular:
Grafos simétricos de Johnson...



NP completo
Clique

Isomorfismo de grafos

Factorización

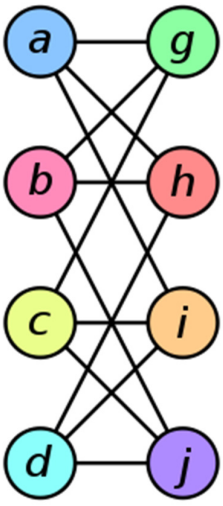
P

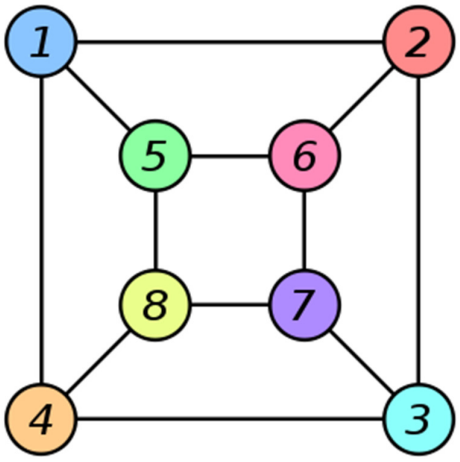
O (exp (polylog (n)) = O (exp ($(\log n)^k$))
http://people.cs.uchicago.edu/~laci/quasipoly.html

Isomorfismo de (sub)grafos

Ejemplo

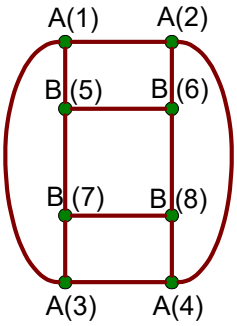
f(a) = 1
f(b) = 6
f(c) = 8
f(d) = 3
f(g) = 5
f(h) = 2
f(i) = 4
f(j) = 7



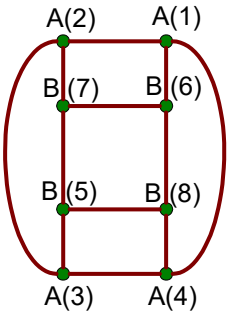


Isomorfismo de (sub)grafos

Un mismo grafo se puede representar de muchas formas:



	A(1)	A(2)	A(3)	A(4)	B(5)	B(6)	B(7)	B(8)
A(1)	1	1	1	0	1	0	0	0
A(2)	1	1	0	1	0	1	0	0
A(3)	1	0	1	1	0	0	1	0
A(4)	0	1	1	1	0	0	0	1
B(5)	1	0	0	0	1	1	1	0
B(6)	0	1	0	0	1	1	0	1
B(7)	0	0	1	0	1	0	1	1
B(8)	0	0	0	1	0	1	1	1



	A(1)	A(2)	A(3)	A(4)	B(5)	B(6)	B(7)	B(8)
A(1)	1	1	0	1	0	1	0	0
A(2)	1	1	1	0	0	0	1	0
A(3)	0	1	1	1	1	0	0	0
A(4)	1	0	1	1	0	0	0	1
B(5)	0	0	1	0	1	0	1	1
B(6)	1	0	0	0	0	1	1	1
B(7)	0	1	0	0	1	1	1	0
B(8)	0	0	0	1	1	1	0	1

Isomorfismo de (sub)grafos

- Cuando manejamos bases de datos de grafos,
  ¡¡¡ tenemos que comparar con conjuntos de grafos !!!

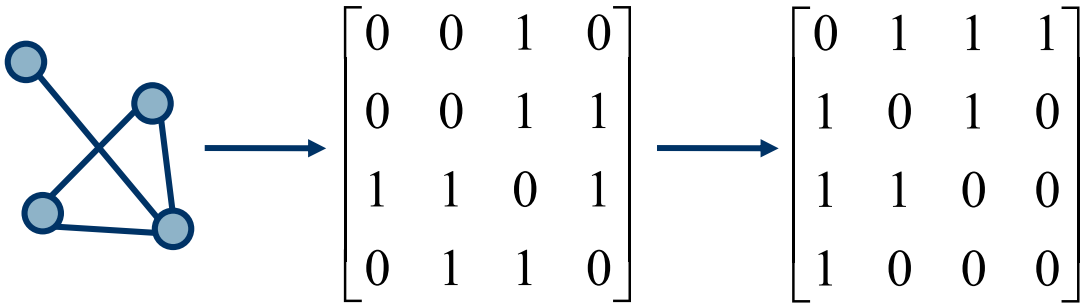
- Se hace imprescindible en la práctica el uso de técnicas de preprocesamiento e indexación.

Isomorfismo de (sub)grafos

Canonicalización

Cada grafo se convierte en una cadena ordenada (su "código") de forma que dos grafos isomorfos tendrán la misma codificación canónica.

Ejemplo: "Lexicographically largest adjacency matrix"



String: 0010001111010110
Canonical: 0111101011001000

Isomorfismo de (sub)grafos

Invariantes

Dados dos grafos $G_1(V_1,E_1)$ y $G_2(V_2,E_2)$, un invariante es una etiqueta $l(v_1)$ asignada a un vértice $v_1 \in G_1$ tal que, si existe un isomorfismo que asigna v_1 al vértice $v_2 \in G2$, entonces $l(v_1)=l(v_2)$.

EJEMPLOS

- Grado de un vértice
- Número de vecinos a distancia dos [two path]
- Triángulos adyacentes
- K-cliques (cliques de tamaño k que incluyen v)
- Conjuntos independientes de tamaño k que incluyen v

Isomorfismo de (sub)grafos

Algoritmos

$G_1(k)$
Subgrafo inducido de G_1 sobre los k primeros vértices

ESQUEMA GENERAL: BACKTRACKING

Para descubrir un isomorfismo entre dos grafos G_1 y G_2, construimos un isomorfismo sobre $G_1(k)$ y lo extendemos a $G_1(k+1)$ añadiendo un nuevo vértice (los invariantes nos permitirán podar la búsqueda).

Isomorfismo de (sub)grafos

Algoritmos

ESQUEMA GENERAL: BACKTRACKING

```
MATCH (G1, G2, s)
  if Ms covers all the vertices of G1 then
      return Ms
  else
    foreach (v1, v2) ∈ P(s) do
      if COMPATIBLE(v1, v2) then
        s' ← s ∪ (v1, v2)
        MATCH (G1, G2, s')
    done
  return no match found
```

Isomorfismo de (sub)grafos

Algoritmo de Ullman (1976)

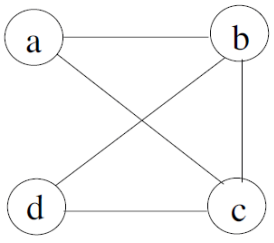
Propiedad de las matrices de adyacencia de dos grafos isomorfos: $A_2 = P\, A_1\, P^T$, donde P es una matriz que representa una permutación (la que define el isomorfismo).

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} c & a & b \\ f & d & e \\ i & g & h \end{bmatrix}$$

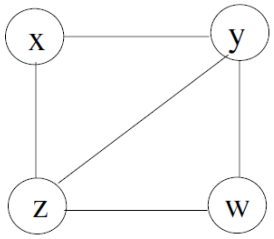
$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} g & h & i \\ a & b & c \\ d & e & f \end{bmatrix}$$

Isomorfismo de (sub)grafos

Algoritmo de Ullman (1976)



G_1



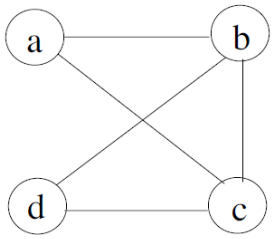
G_2

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

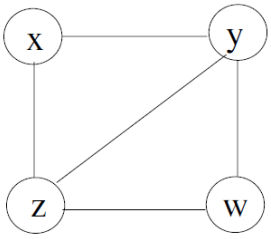
Isomorfismo de (sub)grafos

Algoritmo de Ullman (1976)

a → x
b → y
c → z
d → w



G_1



G_2

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Isomorfismo de (sub)grafos

Algoritmo de Ullman (1976)

Input : $G_1(V_1,E_1)$, $G_2(V_2,E_2)$
$P[n_2,n_2]$: a permutation matrix
Output : $G' \subset G_2$ such that $G' \cong G_1$ if exists
$n_1 \leftarrow |V_1|$; $n_2 \leftarrow |V_2|$
BackTrack $(A_1,A_2, P,1)$

procedure BACKTRACK(A_1,A_2,P,k)
  if $k > n_1$ **then**
    P represents a subgraph isomorphism from G_1 to G_2
    return(P)
  end if
  for $i = 1$ to n_2 **do**
    $p_{ki} \leftarrow 1$
    for all $j = 1$ to n_1, $j = i$ **do**
      $p_{kj} \leftarrow 0$
    end for
    if $S_{k,k}(A_1) = S_{k,n}(P)A_2(S_{k,n}(P)^T)$ **then**
      BackTrack $(A_1,A_2, P,k+1)$
    end if
  end for
end procedure

Tiempo
$O(m^n n^2)$

Espacio
$O(n^2 m)$

Isomorfismo de (sub)grafos

nauty
[McKay, 1981]

- Representa el grafo de **forma canónica**.
- Utiliza **invariantes** en la búsqueda de isomorfismos.
- Define **particiones** sobre los vértices
  (va dividiendo los vértices en conjuntos disjuntos).
  - Partición inicial (invariantes sobre el grafo completo).
  - Refinamiento (invariantes sobre cada partición).
  - Partición hoja (conjuntos de un nodo).

Isomorfismo de (sub)grafos

nauty

- Algoritmo:
  Búsqueda en profundidad en el espacio de particiones.
- Refinamiento de las particiones:
  Dada una partición $P=\{V_1..V_m\}$
  en la que $\forall v,w \in V_i$, $d(v,V_i)=d(w,V_i)$
  - Seleccionar $Vi \in P$ con más de un elemento.
  - $\forall v,w \in V_i$, calcular $d(v,V_i)$
  - Dividir V_i en subconjuntos que tengan el mismo valor para $d(v,V_i)$.

Isomorfismo de (sub)grafos

nauty

- El refinamiento de las particiones se repite para todos los V_i hasta que ningún conjunto puede dividirse más.
- Los hijos de una partición se generan seleccionando, para cada vértice $v \in V_i$, una partición hija con los conjuntos $\{V_1, .., V_{i-1}, \{v\}, V_i/\{v\}, V_{i+1}, .., V_m\}$.
- Nauty devuelve la representación canónica del grafo correspondiente a la matriz de adyacencia del menor automorfismo (isomorfismo del grado consigo mismo).

Isomorfismo de (sub)grafos

nauty

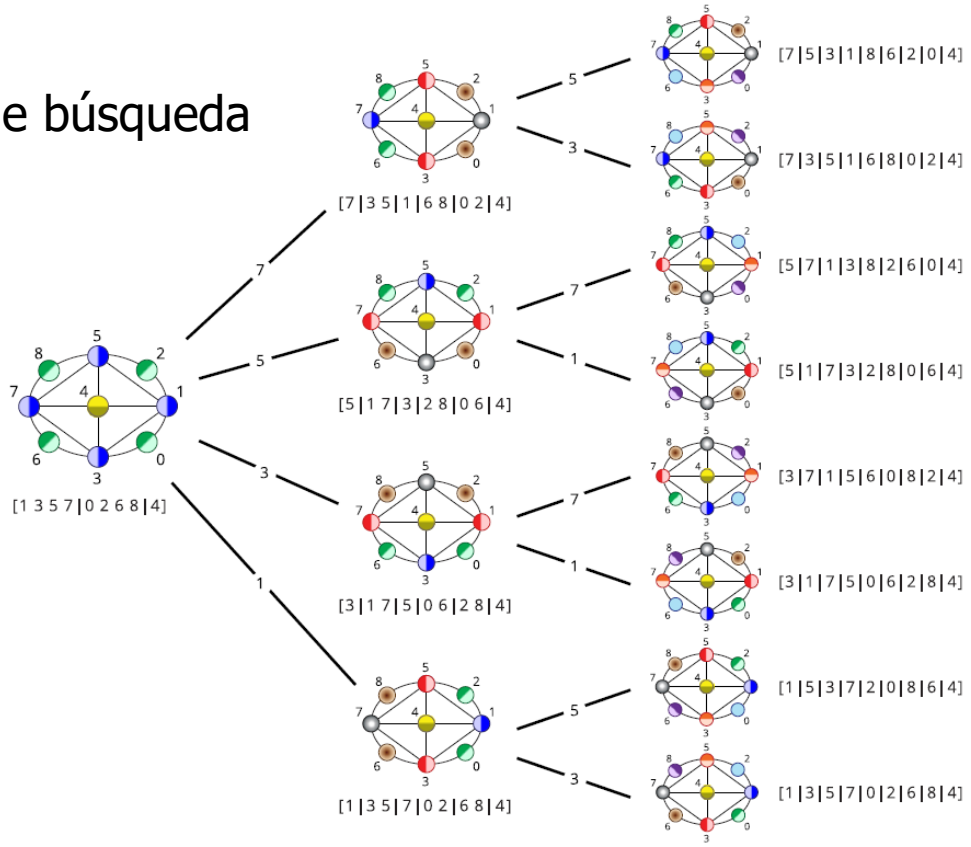
Input : $G_1(V_1, E_1)$
Output : $G_2(V_2, E_2)$: a canonical graph

$\mathcal{P} \leftarrow$ partition of a single cell V
$S \leftarrow$ stack containing $\mathcal{P}$
while $S \neq \emptyset$ **do**
  $u \leftarrow pop(S)$
  if $u = leaf\ partition$ **then**
    $update(G_2, u)$
  else
    $refine(u)$
    append children of u to S
  end if
end while

Isomorfismo de (sub)grafos

nauty

Árbol de búsqueda

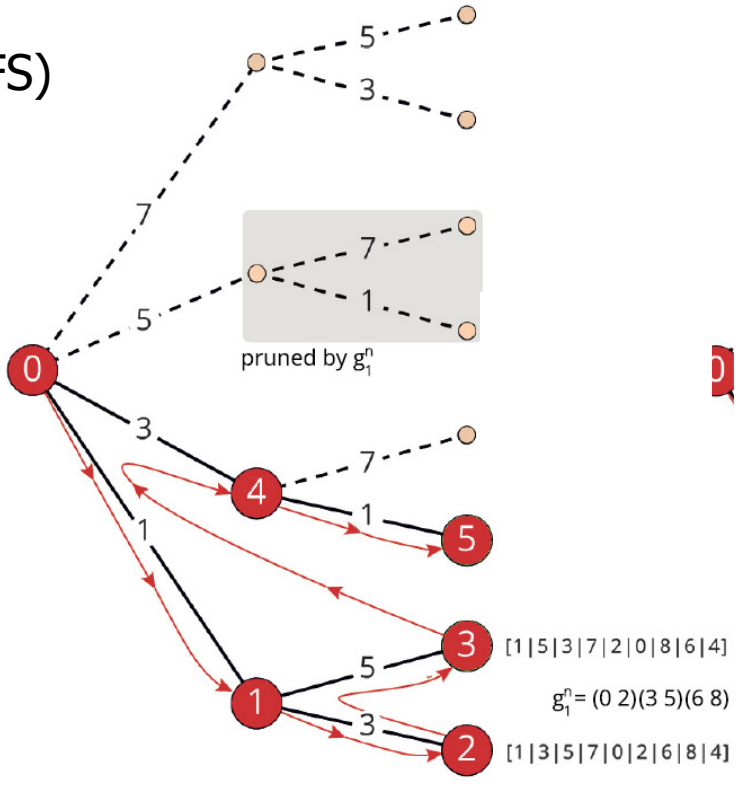


Isomorfismo de (sub)grafos

nauty

Exploración (DFS)

y poda…



Isomorfismo de (sub)grafos

VF2

[Cordella et al., TPAMI 2004]

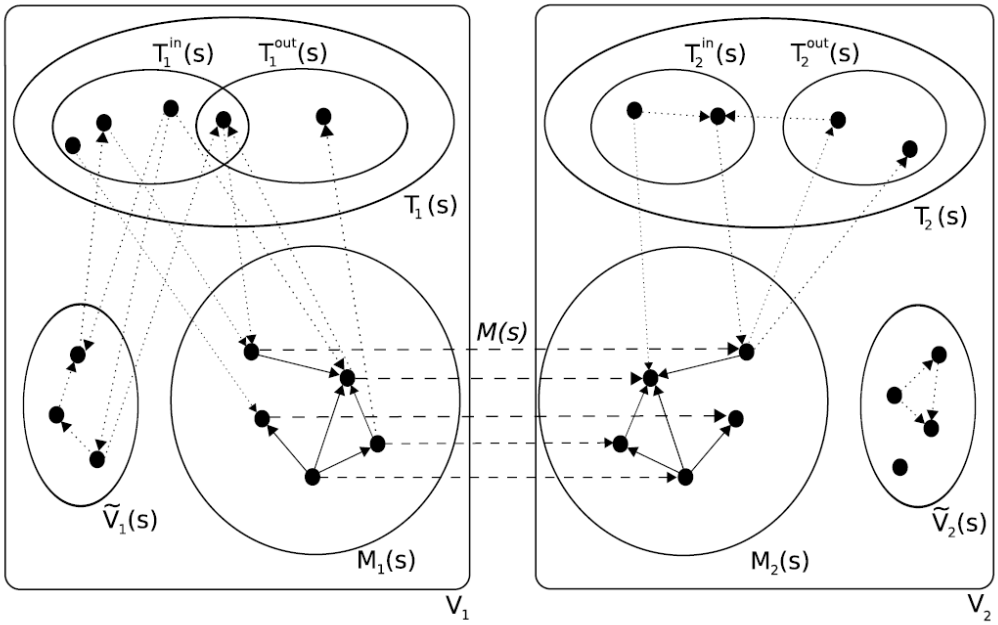
- DFS
- Reglas para podar el árbol de búsqueda que comprueban, para cada pareja de nodos candidatos, si el emparejamiento parcial es compatible.

	VF2 Mejor caso	Peor caso	**Ullman** Mejor caso	Peor caso
Tiempo	$\Theta(n^2)$	$\Theta(n!n)$	$\Theta(n^3)$	$\Theta(n!n^2)$
Espacio	$\Theta(n)$	$\Theta(n)$	$\Theta(n^3)$	$\Theta(n^3)$

Isomorfismo de (sub)grafos

VF2

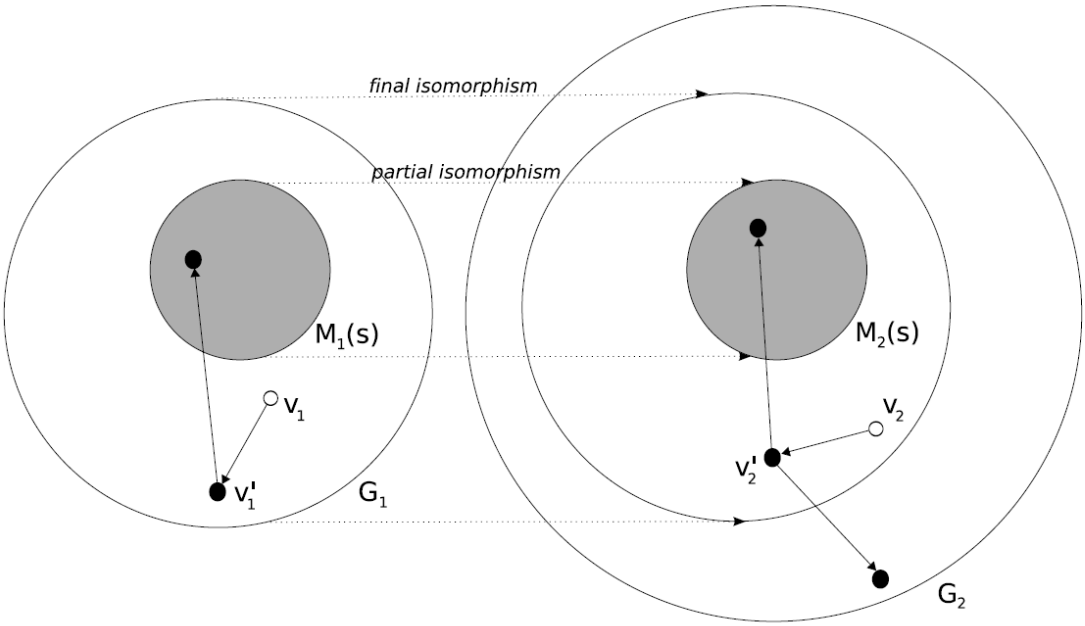
Correspondencia parcial



Isomorfismo de (sub)grafos

VF2

"look-ahead"



Isomorfismo de (sub)grafos

VF2

procedure MATCH(G_1, G_2, s)
  Input : two graphs G_1 and G_2, an intermediate state s
  initial state s_0 has $M(s_0) = \emptyset$
  Output : the mappings between G_1 and G_2
  if M_s covers all nodes of G_1 **then**
    return(M_s)
  else
    compute $P(s)$ of candidate pairs to be included in $M(s)$
    for all $p \in P(s)$ **do**
      if p is compatible with $M(s)$ **then**
        $s' \leftarrow p \cup M(s)$
        compute state s' obtained by adding p to $M(s)$
        $Match(G_1, G_2, s')$
      end if
    end for
    restore data structures
  end if
end procedure

Isomorfismo de (sub)grafos

VF2

	Randomly Connected			2D (regular and irregular) Mesh				Bounded valence		
Nodes	η=0.01	η=0.05	η=0.1	regular	ρ=0.2	ρ=0.4	ρ=0.6	ν=3	ν=6	ν=9
20	VF2	VF2	Nauty	VF2	Nauty	Nauty	Nauty	Nauty	Nauty	Nauty
40	VF2	Nauty	Nauty	VF2	VF2	Nauty	Nauty	Nauty	Nauty	Nauty
60	VF2	Nauty	Nauty	VF2	VF2	VF2	Nauty	VF2	Nauty	Nauty
80	VF2	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	Nauty	Nauty
100	VF2	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	Nauty	Nauty
200	VF2	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	Nauty
400	Nauty	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	Nauty
600	Nauty	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	Nauty
800	Nauty	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	VF2
1000	Nauty	Nauty	Nauty	VF2	VF2	VF2	VF2	VF2	VF2	VF2

Isomorfismo de (sub)grafos

BM1

[Battiti & Mascia, SLS'2007]

- Mejora de VF2 en tiempo de CPU.
- IDEA: "local-path-based pruning"

  Si existe un camino de longitud d desde v_1 en G_1, debe existir el mismo camino desde v_2 en G_2 para que el par (v_1,v_2) sea compatible.

Isomorfismo de (sub)grafos

BM1

[Battiti & Mascia, SLS'2007]

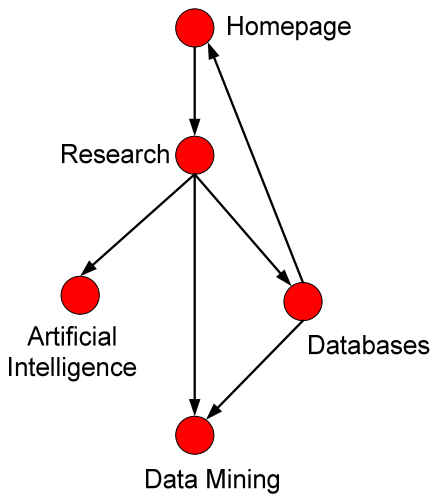
```
procedure CompatiblePaths(v1, v2, d)
    Output : the mappings between G1 and G2
    for all x in 1,...,d do
        for all y in 1,...,2^d do
            if PathsDS[v1][x][y] > PathsDS[v2][x][y] then
                return false
            end if
        end for
    end for
end procedure
```

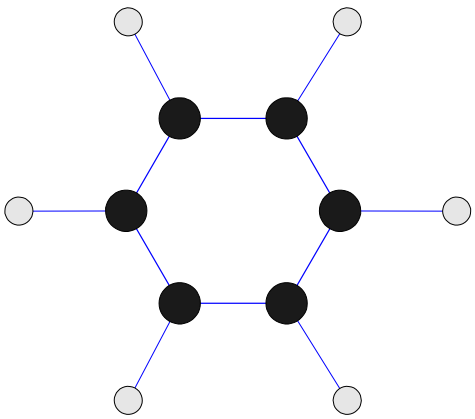
	BM1	VF2
Tiempo	$O(n^d+n!n)$	$O(n!n)$
Espacio	$\Theta(n2^{d+1})$	$O(n)$

Subgrafos frecuentes

Identificación de patrones frecuentes en grafos

Aplicaciones: Web Mining, Bioinformática, redes sociales…





Grafo dirigido

Grafo no dirigido

Subgrafos frecuentes

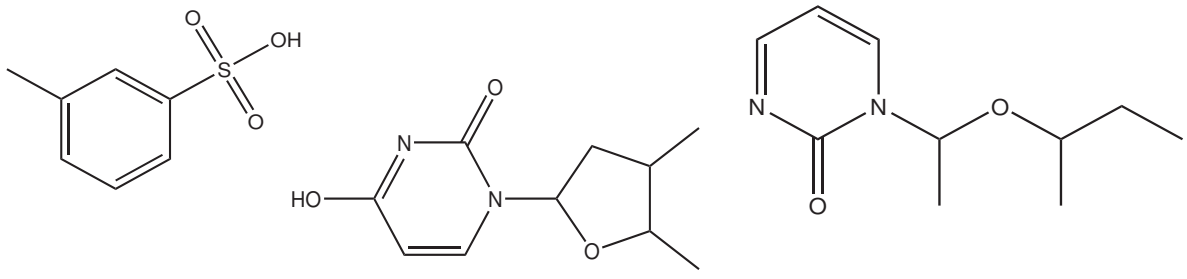
- **Subgrafo**
  Un grafo g es un subgrafo de otro grafo G ($g \subseteq G$) si existe un isomorfismo de subgrafos de g a G.

- **Subgrafo frecuente**
  Dada una base de datos de grafos (o un único grafo enorme), un subgrafo es frecuente si su soporte (frecuencia de ocurrencia) no es menor que un umbral de soporte mínimo preestablecido.

Subgrafos frecuentes

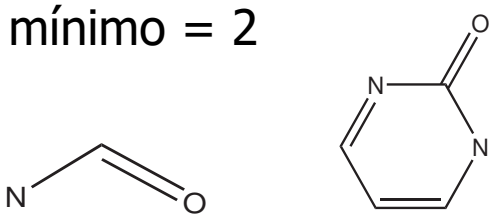
Ejemplo

Conjunto de datos (grafos no dirigidos)



Patrones frecuentes

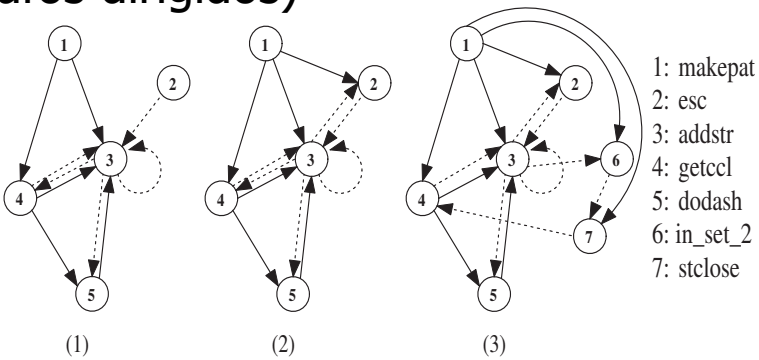
- Umbral de soporte mínimo = 2



Subgrafos frecuentes

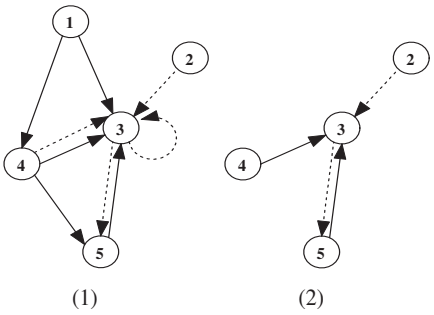
Ejemplo

Conjunto de datos (grafos dirigidos)



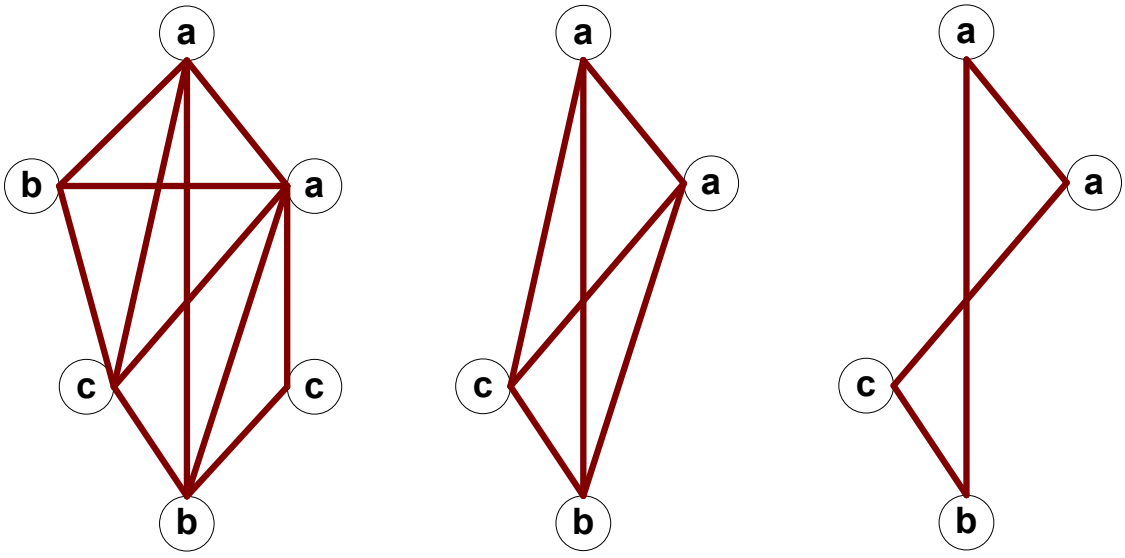
Patrones frecuentes

- Umbral de soporte mínimo = 2



Subgrafos frecuentes

Tipos de subgrafos frecuentes



Grafo original

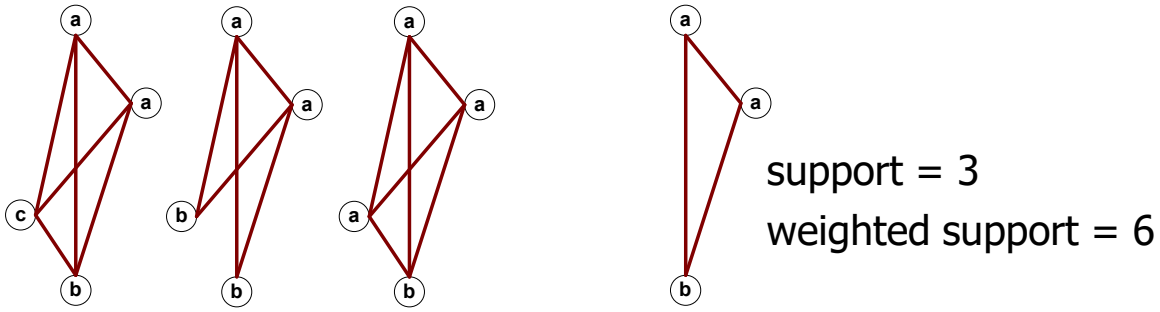
Subgrafo inducido

Subgrafo empotrado

Subgrafos frecuentes

Conteo del soporte (número de ocurrencias)

- Soporte [support]:
  Número de grafos en la base de datos que contienen al menos una ocurrencia del subgrafo.
- Soporte ponderado [weighted support]
  Número total de ocurrencias del subgrafo en todos los grafos de la base de datos.



Base de datos Subgrafo

Subgrafos frecuentes

Algoritmos

Búsqueda dirigida [beam search]

- **SUBDUE** [Holder et al., KDD'1994]
  [Cook & Holder, IEEE Intelligent Systems, 2000]

Inductive Logic Programming (ILP): Datalog

- **WARMR** [Dehaspe et al., KDD'1998 & DMKD'1999]

Patrones frequentes

- Tipo Apriori:
  AGM/AcGM, FSG, "disjoint paths", SiGram
- Tipo FP-Growth:
  MoFa, gSpan, FFSM, Gaston, CloseGraph, Spin

Subgrafos frecuentes: SUBDUE

Búsqueda dirigida

Se limita el número de "mejores" subestructuras.

MDL [Minimum Description Length]

- Las subestructuras se evalúan en función de su capacidad para "comprimir" los grafos de entrada.
- La mejor subestructura S de un grafo G minimiza DL(S)+DL(G\S)

Algoritmo greedy: Comenzando con vértices individuales, se añaden nuevas aristas a las mejores subestructuras encontradas hasta que no se puedan encontrar nuevas subestructuras.

Subgrafos frecuentes

Propiedad clave

ANTI-MONOTONICIDAD

a.k.a. propiedad Apriori

> Un subgrafo de tamaño k es frecuente sólo si todos sus subgrafos son frecuentes.

NOTA: Un grafo con n aristas tiene 2^n subgrafos…

Subgrafos frecuentes: WARMR

Algoritmo basado en ILP [Inductive Logic Programming] para extraer patrones de datos estructurados.

- Uso de Datalog para representar tanto los datos como los patrones descubiertos.

> *?- six_ring(C,S), atomel(C,A1,h), atomel(C,A2,c), bond(C,A1,A2,X), occurs_in(A2,S)* (frequency: 157 compounds (70%), i.e., "a hydrogen atom bound to a carbon atom in a six ring".

- Primer sistema que aprovecha la propiedad Apriori.

Artículo original (KDD'1998):
Aplicación a la predicción de químicos carcinógenos.

Subgrafos frecuentes

Caracterización de los algoritmos de identificación de subgrafos frecuentes

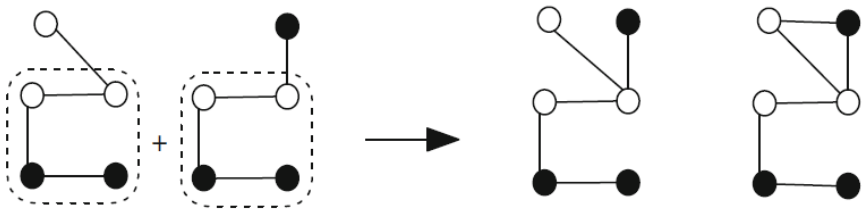
- Tipo de grafos (dirigidos/no-dirigidos, etiquetados…)
- Tipo de patrones identificados (inducidos/empotrados)
- Cálculo del soporte
- Orden de búsqueda (anchura vs. profundidad)
- Generación de candidatos (Apriori vs FP-Growth)
- Eliminación de duplicados
- Orden de identificación de patrones (p.ej. camino → árbol → grafo)

Subgrafos frecuentes: Apriori

Algoritmos basados en Apriori

Búsqueda en anchura

Grafos con k elementos → Grafos con k+1 elementos



- AGM [Inokuchi et al., PKDD'2000 & Machine Learning '2003] genera grafos candidatos con un nuevo nodo.
- FSG [Kuramochi & Karypis, ICDM'2001 & TKDE'2004] genera grafos candidatos con una nueva arista.

Subgrafos frecuentes: Apriori

Algoritmos basados en Apriori

Apriori (D, min_sup, S_k)

Input: Graph dataset D, minimum support threshold *min_sup*, size-k frequent subgraphs S_k

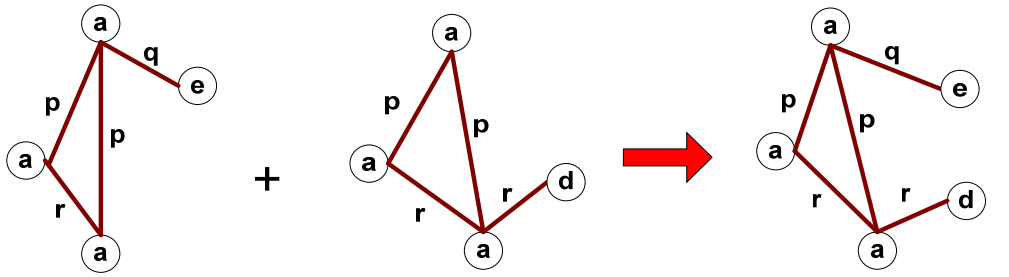
Output: The set of size-$(k+1)$ frequent subgraphs S_{k+1}

1: $S_{k+1} \leftarrow \varnothing$;
2: **for** each frequent subgraph $g_i \in S_k$ **do**
3: **for** each frequent subgraph $g_j \in S_k$ **do**
4: **for** each size-$(k+1)$ graph g formed by joining g_i and g_j **do**
5: **if** g is frequent in D and $g \notin S_{k+1}$ **then**
6: insert g to S_{k+1};
7: **if** $S_{k+1} \neq \varnothing$ **then**
8: call Apriori(D, *min_sup*, S_{k+1});
9: **return**;

Subgrafos frecuentes: Apriori

AGM: Apriori-based Graph Mining
[Inokuchi et al., PKDD'2000 & Machine Learning'2003]

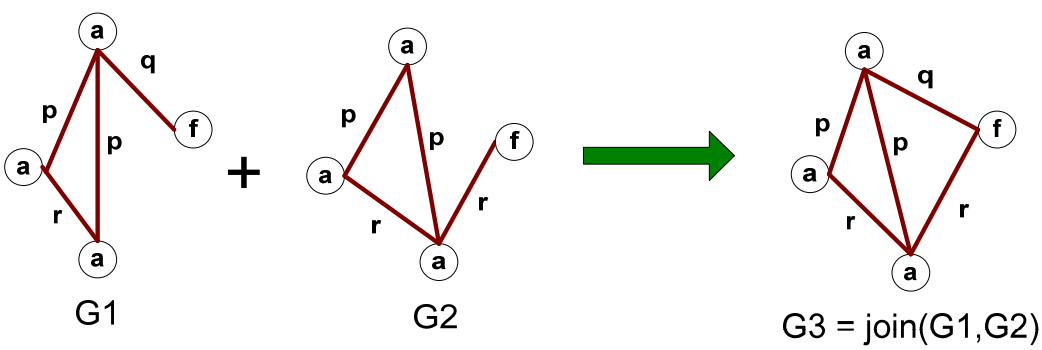
Vertex growing



Subgrafos frecuentes: Apriori

FSG: Frequent Sub-Graph discovery
[Kuramochi and Karypis, ICDM'2001 & IEEE TKDE'2004]

Edge growing

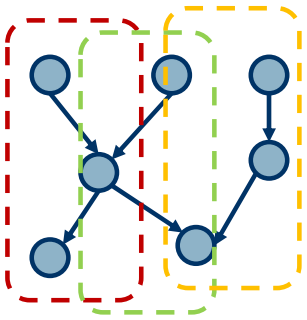


Subgrafos frecuentes: Apriori

JoinPath [Vanetik et al., ICDM'2002 & ICDE'2004]
[Gudes et al., IEEE TKDE'2006]

EDPs = Edge-disjoint paths
(caminos sin aristas comunes)

1. Identificar caminos frecuentes
2. Identificar grafos frecuentes con 2 "edge-disjoint paths"
3. Iterativamente, construir grafos con k+1 EDPs a partir de grafos con k EDPs.

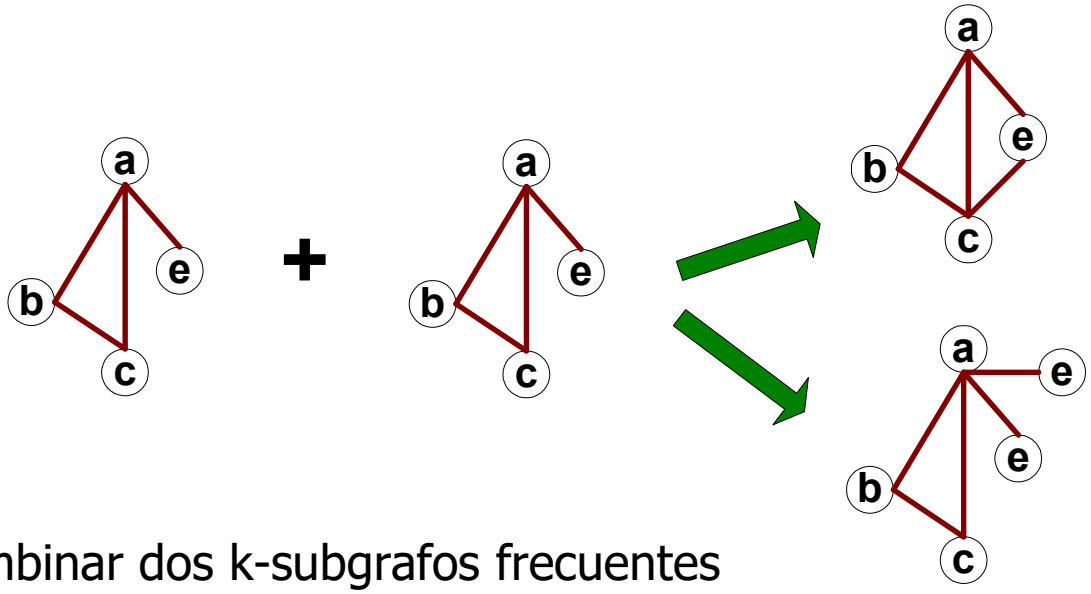


Grafo con 3 EDPs

Subgrafos frecuentes: Apriori

Generación de candidatos [edge growing]

p.ej. Mismas etiquetas en distintos nodos

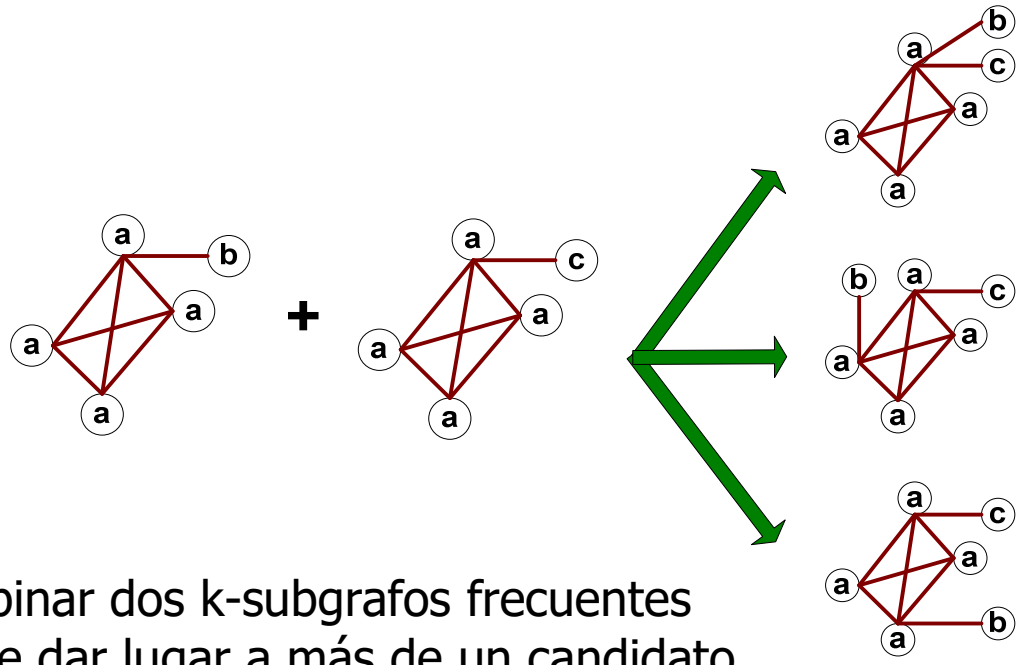


Combinar dos k-subgrafos frecuentes
puede dar lugar a más de un candidato

Subgrafos frecuentes: Apriori

Generación de candidatos [edge growing]

p.ej. "Núcleo" [core] con las mismas etiquetas

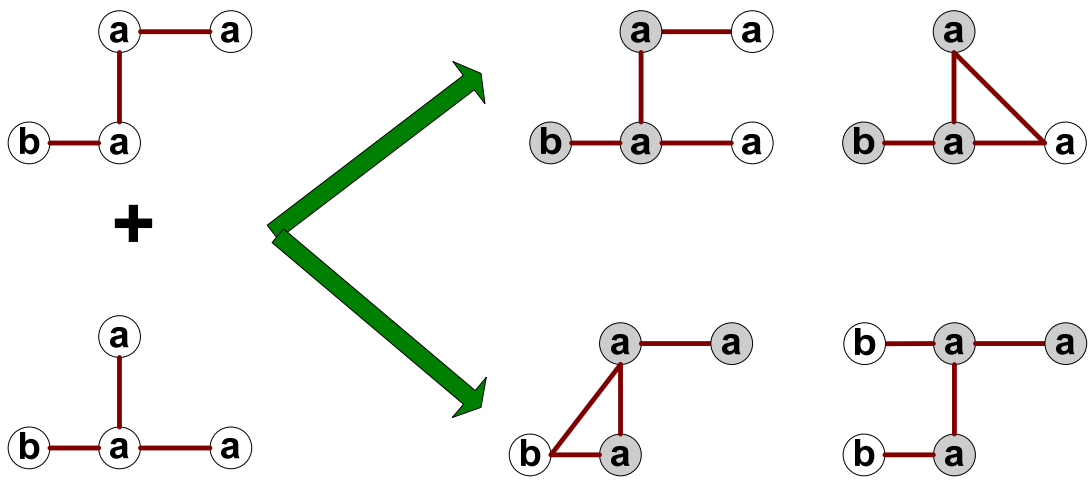


Combinar dos k-subgrafos frecuentes
puede dar lugar a más de un candidato

Subgrafos frecuentes: Apriori

Generación de candidatos [edge growing]

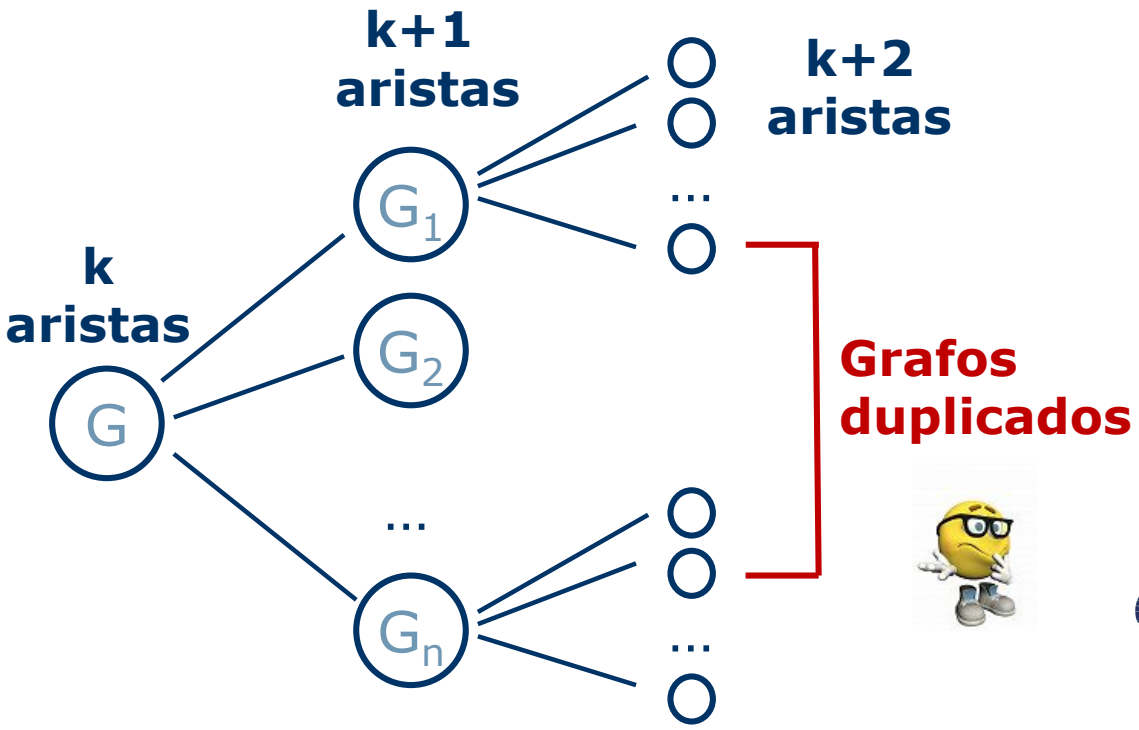
p.ej. Múltiples núcleos



Combinar dos k-subgrafos frecuentes
puede dar lugar a más de un candidato

Subgrafos frecuentes: FP-Growth

Problema de los algoritmos basados en Apriori



Subgrafos frecuentes: FP-Growth

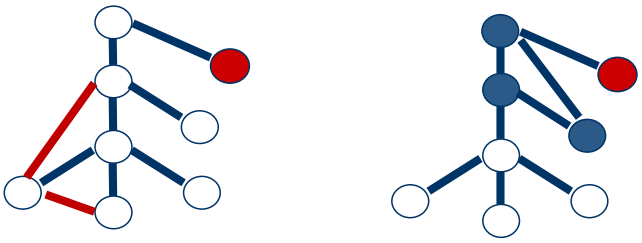
Algoritmos derivados de FP-Growth

- Inspirados en PrefixSpan (secuencias), TreeMinerV y FREQT (árboles).
- Extienden los patrones frecuentes directamente (añadiendo una arista, en todas las posiciones posibles).
- Evitan la reunión de dos patrones para generar nuevos candidatos (operación más costosa en los algoritmos derivados de Apriori).

Subgrafos frecuentes: FP-Growth

gSpan
[Yan & Han, ICDM'2002]

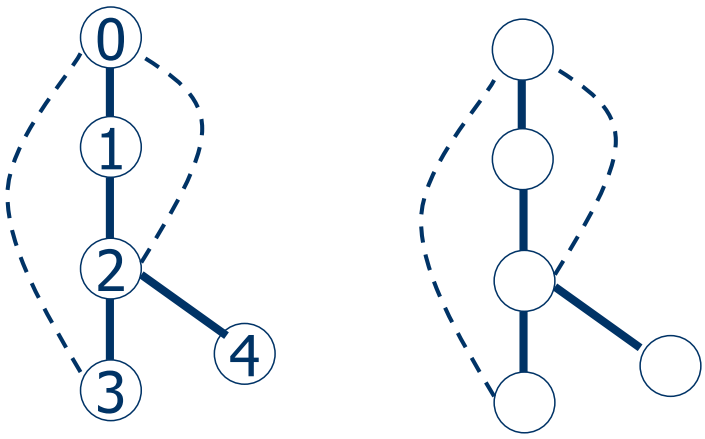
"Right-most extension"



La enumeración de grafos usando su "extensión más a la derecha" es completa.

Subgrafos frecuentes: FP-Growth

gSpan
[Yan & Han, ICDM'2002]



e0: (0,1)

e1: (1,2)

e2: (2,0)

e3: (2,3)

e4: (3,1)

e5: (2,4)

Búsqueda en profundidad (DFS)
Grafo → Secuencia de aristas

Subgrafos frecuentes: FP-Growth

Gaston [Nijssen and Kok, KDD'2004]
GrAph, Sequences and Tree extractiON algorithm

Separa la identificación de distintos tipos de patrones, ya que la identificación de estructuras más simples es mucho más eficiente (así como la eliminación de duplicados):

caminos → árboles → grafos

Subgrafos frecuentes: FP-Growth

CloseGraph [Yan & Han, KDD'2003]

- Grafo cerrado:
  Un grafo G se dice cerrado si no existe ningún supergrafo de G que tenga el mismo soporte que G.

- IDEA: Compresión sin pérdidas

  Si hay subgrafos de G con exactamente su mismo soporte, no es necesario identificarlos (grafos no cerrados).

Subgrafos frecuentes: FP-Growth

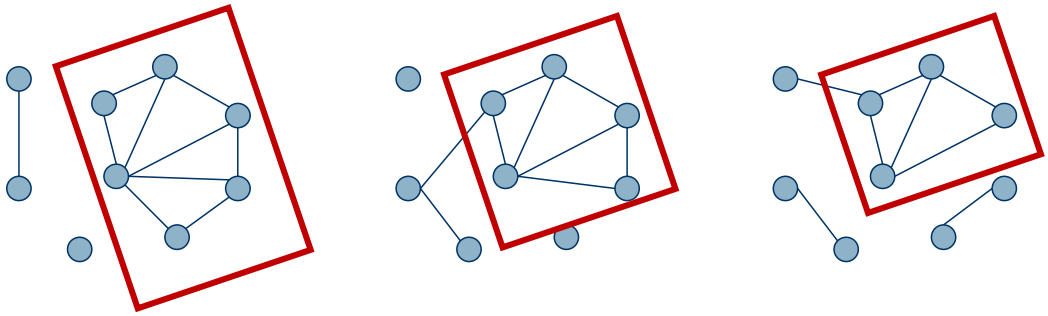
CloseGraph [Yan & Han, KDD'2003]

Dados dos grafos frecuentes G y G', con G subgrafo de G', si siempre que encontramos G en nuestros datos también aparece G'

- Sólo serán cerrados los descendientes de G que sean también descendientes de G'.
- No es necesario que sigamos expandiendo G para encontrar nuevos patrones, salvo en situaciones muy puntuales ["tricky exception cases"]…

Subgrafos frecuentes: FP-Growth

CloseCut & Splat [Yan, Zhou & Han, KDD'2005]
Subestructuras densas

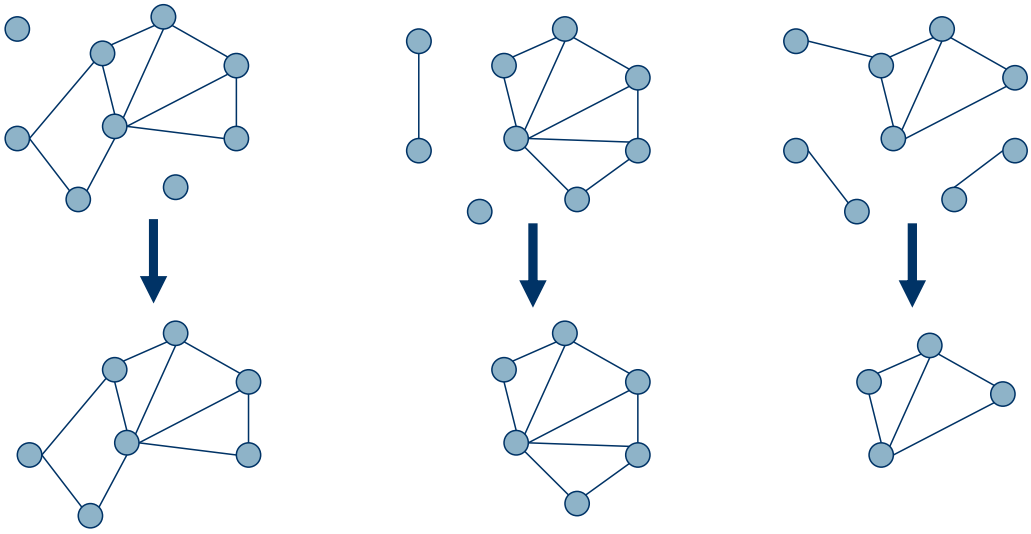


Restricciones adicionales nos permiten optimizar los algoritmos de identificación de subgrafos frecuentes (conectividad, grado, diámetro, densidad…)

Subgrafos frecuentes: FP-Growth

CloseCut & Splat [Yan, Zhou & Han, KDD'2005]

Subestructuras densas: Reducción de patrones (1/2)

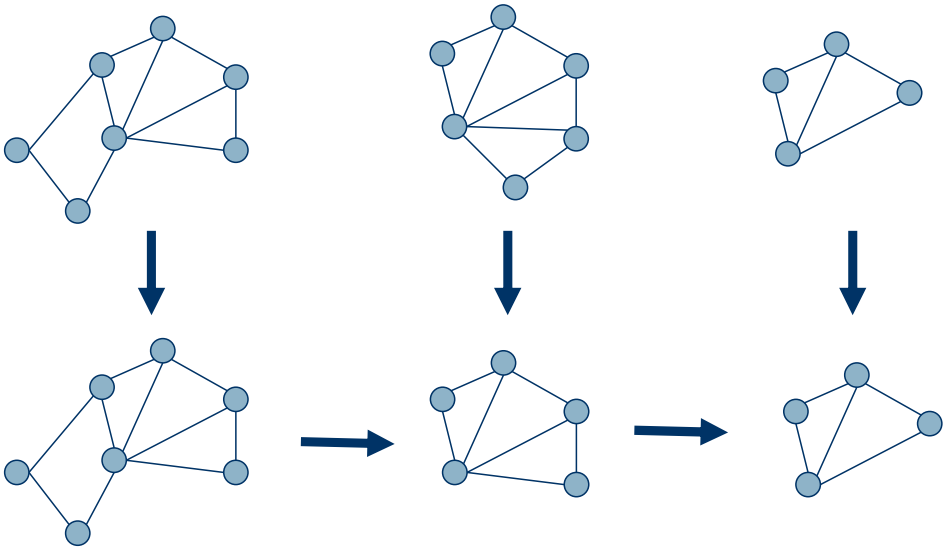


Descomposición de grafos en función de su conectividad.

Subgrafos frecuentes: FP-Growth

CloseCut & Splat [Yan, Zhou & Han, KDD'2005]

Subestructuras densas: Reducción de patrones (2/2)



Intersección y descomposición de subgrafos (**Splat**)

Subgrafos frecuentes

Otros algoritmos

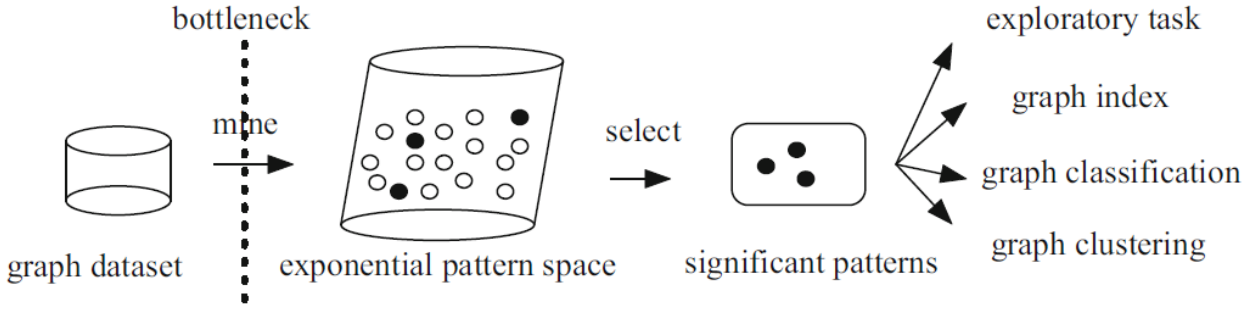
- MoFa [Borgelt & Berthold, ICDM'2002]
- FFSM [Han et al., ICDM'2003]
- SPIN [Huan et al., KDD'2004]
- GREW [Kuramochi & Karypis, ICDM'2004]
- SiGram [Kuramochi & Karypis, DMKD'2005]
- TSMiner [Jin et al., KDD'2005]
- MARGIN [Thomas et al., KDD'2006]

...

Subgrafos frecuentes

PROBLEMA

Para encontrar todos los grafos frecuentes, tenemos que analizar un número exponencial de subgrafos…



¿Cómo evitar la explosión combinatoria en la práctica?

Subgrafos frecuentes

Modelos escalables

Evitan la explosión combinatoria
(la generación de un número exponencial de grafos)

- Subgrafos significativos [He & Singh, ICDM'2007]:
  gboost [Kudo et al., NIPS'2004]
  gPLS [Saigo et al., KDD'2008]
  LEAP [Yan et al., SIGMOD'2008]
  GraphSig [Ranu & Singh, ICDE'2009]

- Subgrafos representativos:
  ORIGAMI [Hasan… & Zaki, ICDM'2007]

Subgrafos frecuentes

Modelos escalables

Subgrafos densos

- Cliques & cuasi-cliques (problema NP)
  H*Graph [Cheng et al., SIGMOD'2010+KDD'2012&13]

- K-cores: O(m)
  EMcore [Cheng et al., ICDE'2011]

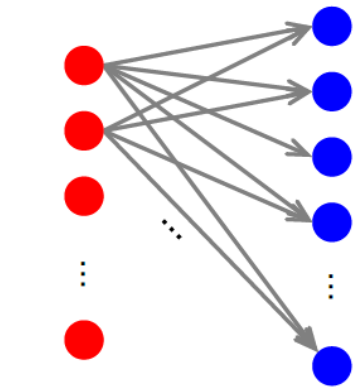
- K-trusses = Triangle k-cores: $O(m^{1.5})$
  [Wang & Cheng, PVLDB'2012]
  [Zhang & Partharasany, ICDE'2012]

Subgrafos frecuentes

Modelos escalables: Trawling

Subgrafos bipartidos frecuentes

p.ej. Pequeñas comunidades en la Web



Dense 2-layer graph

Problema formal:

Enumerar todos los grafos bipartidos completos $K_{s,t}$

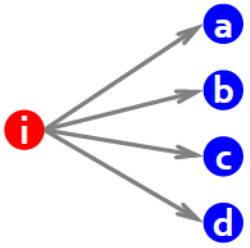
¿Cómo?

Encontrando itemsets frecuentes…

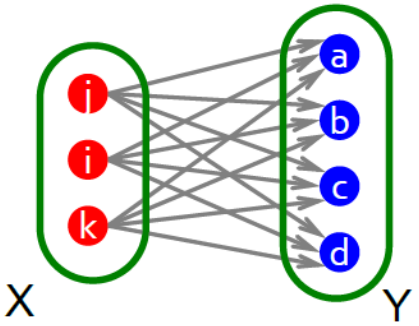
Subgrafos frecuentes

Modelos escalables: Trawling

Itemsets frecuentes = Grafos bipartidos completos $K_{s,t}$



S_i={a,b,c,d}



- Cada nodo i puede verse como el conjunto de nodos a los que apunta S_i.
- $K_{s,t}$ es el conjunto de tamaño t que ocurre en s conjuntos S_i.

Buscar los grafos bipartidos completos $K_{s,t}$ es buscar t-itemsets frecuentes usando s como umbral :-)

Motif Discovery

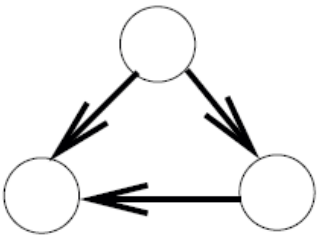
Modelos escalables: Subgrafos pequeños

"Motif" de tamaño k

Pequeño grafo conectado con k vértices/nodos
que aparece en una red con más frecuencia de la esperada.

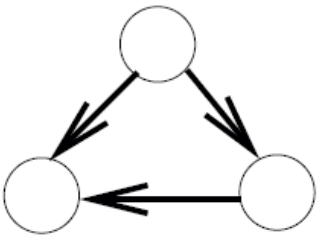
Ejemplo

Feed-forward loop

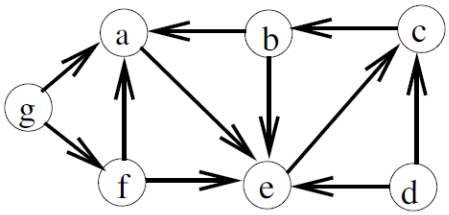


Motif Discovery

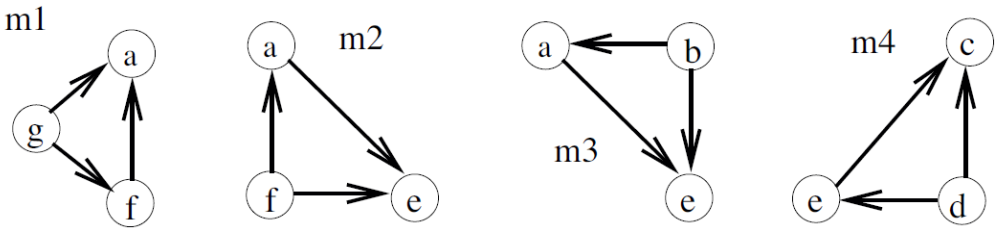
Ejemplo: Feed-forward loop



- Grafo de entrada

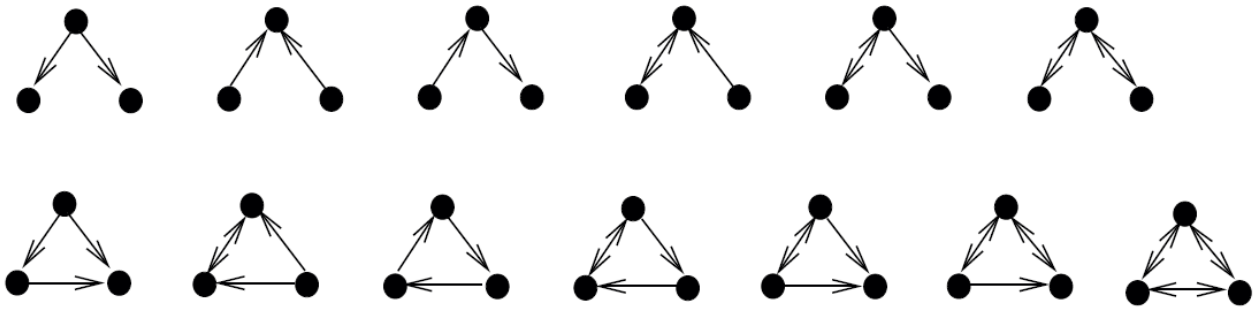


- Ocurrencias del patrón



Motif Discovery

Los 13 "motifs" dirigidos de tamaño 3:

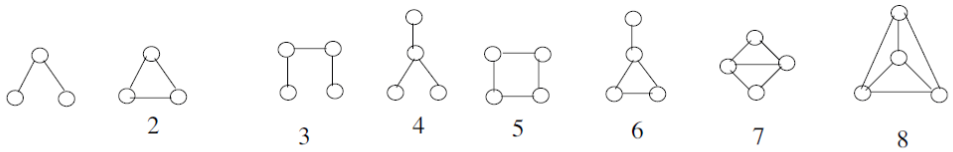


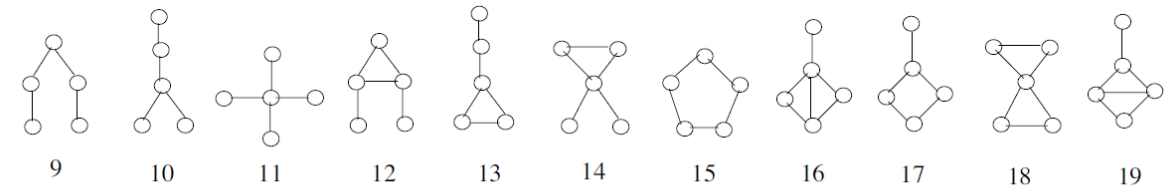
Número exponencial de posibles "motifs": **$O(2^{n(n-1)})$**

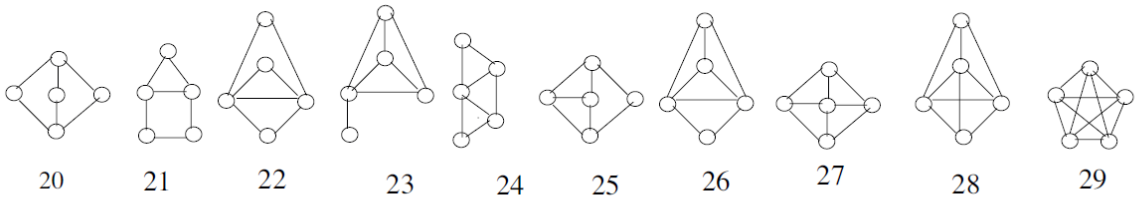
Motif size	3	4	5	6	7	8	9	10
Undirected subgraphs	2	6	21	112	853	$\approx 10^4$	$\approx 10^5$	$\approx 10^7$
Directed subgraphs	13	199	9364	10^6	$\approx 10^9$	$\approx 10^{12}$	$\approx 10^{16}$	$\approx 10^{20}$

Motif Discovery

"Graphlets" de tamaños 3, 4 y 5



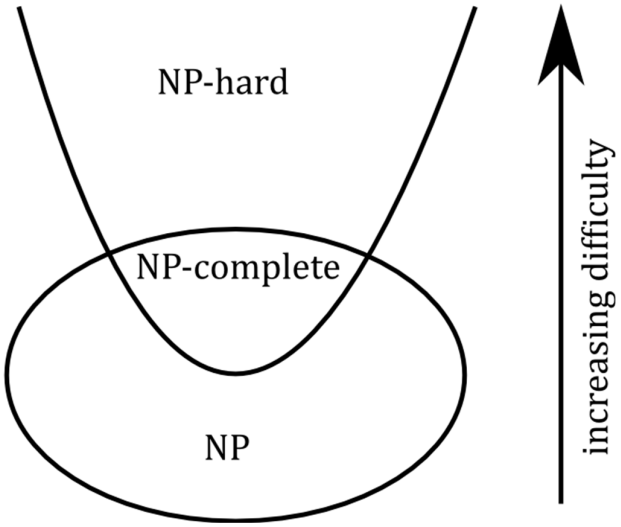




5–node graphlets

Motif Discovery

- Encontrar el número de apariciones de un "motif" es un problema NP-hard (encontrar una aparición concreta de un "motif" en un grafo también lo es), por lo que se hace necesario el uso de heurísticas.



- Encontrar y agrupar "motifs" isomórficos es un problema equivalente a encontrar los subgrafos isomórficos del patrón buscado (NP-completo).

Motif Discovery

Para evaluar la importancia de la aparición de un "motif" en un grafo, es necesario identificar sus ocurrencias en el grafo **y también en redes aleatorias**.

Una versión aleatoria R de un grafo G, similar en su estructura a G, se conoce como modelo nulo [null model].

Existe una familia $\Psi(G)$ de grafos aleatorios con propiedades similares a G (tamaño, secuencia de grados…), por lo que se genera una muestra S de n grafos de $\Psi(G)$ y se calcula la frecuencia de un "motif" tanto en el grafo G como en la muestra S.

Motif Discovery

Si la frecuencia de un patrón m es significativamente mayor en G que su frecuencia media en la muestra S de $\Psi(G)$, entonces se acepta como "motif":

- **Z-score**:
  e.g. Z(m)>2.0

$$Z(m) = \frac{F_1(m) - \overline{F_{1,r}(m)}}{\sigma_r(m)}$$

- **P-value:**
  e.g. P(m)<0.05

$$P(m) = \frac{1}{n}\sum_{i=1}^{n} \sigma_{R_i}(m)$$

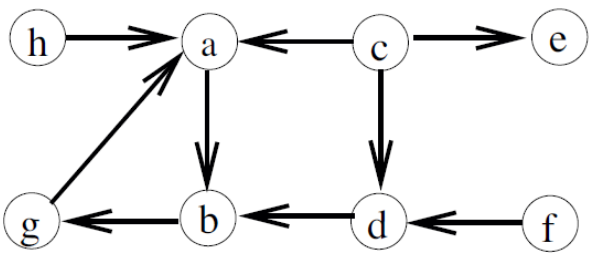
- **Significance profile:**
  (comparación de redes con respecto a los motifs que contienen)

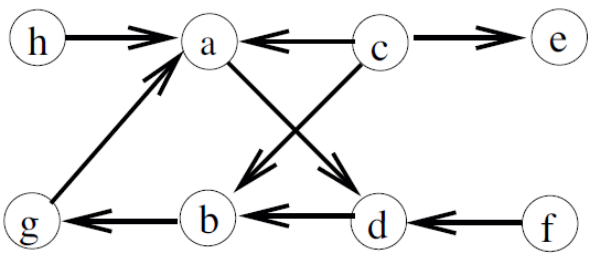
$$SP(m_i) = \frac{Z(m_i)}{\sqrt{\sum_{i=1}^{n} Z(m_i)^2}}$$

Motif Discovery

Generación de modelos nulos

Grafos del mismo tamaño pero distinta topología que preserven su secuencia de grados (y los grados de entrada y salida para cada nodo).





- Se escogen dos arcos al azar: (a,b), (c,d)
- Se cruzan sustituyéndolos por (a,d), (c,b)

Motif Discovery

Generación de modelos nulos

Método basado en cadenas de Markov

Input : $G(V,E)$
Output : Random graph $G'(V',E')$ similar to G
$G' \leftarrow G$
while G' is not as randomized as required **do**
  pick two random edges $(a,b),(c,d) \in E'$
  if $(a,d) \notin E' \wedge (c,b) \notin E'$ **then**
    $E' \leftarrow E' \setminus \{(a,b),(c,d)\}$
    $E' \leftarrow E' \cup \{(a,d),(c,b)\}$
  end if
end while

Motif Discovery

Fases en la identificación de motifs

- Descubrir los motifs $m_1..m_n$ que ocurren en G con más frecuencia de la esperada.
- Agrupar los motifs $m_1..m_n$ en clases isomórficas $C_1..C_k$ de forma que todos los motifs topológicamente equivalentes estén en la misma clase.
- Determinar cuáles de las clases identificadas ocurren en G con mucha más frecuencia que en grafos aleatorios topológicamente similares a G.

Motif Discovery

Algoritmos de identificación de motifs

- EXACT CENSUS ALGORITHMS
  Los algoritmos exactos intentan encontrar todas las ocurrencias de un motif en una red.

- APPROXIMATE CENSUS ALGORITHMS
  Los algoritmos aproximados, por cuestiones de eficiencia, muestrean subgrafos [subgraph sampling].

Motif Discovery

Mfinder

Enumeración completa

- Para cada arista del grafo, se construye un subgrafo G' alrededor de la arista.

- Se van añadiendo a este grafo vértices vecinos a cualesquiera de los vértices del grafo parcial G' hasta que el tamaño de G' coincida con el de los motids buscados.

Motif Discovery

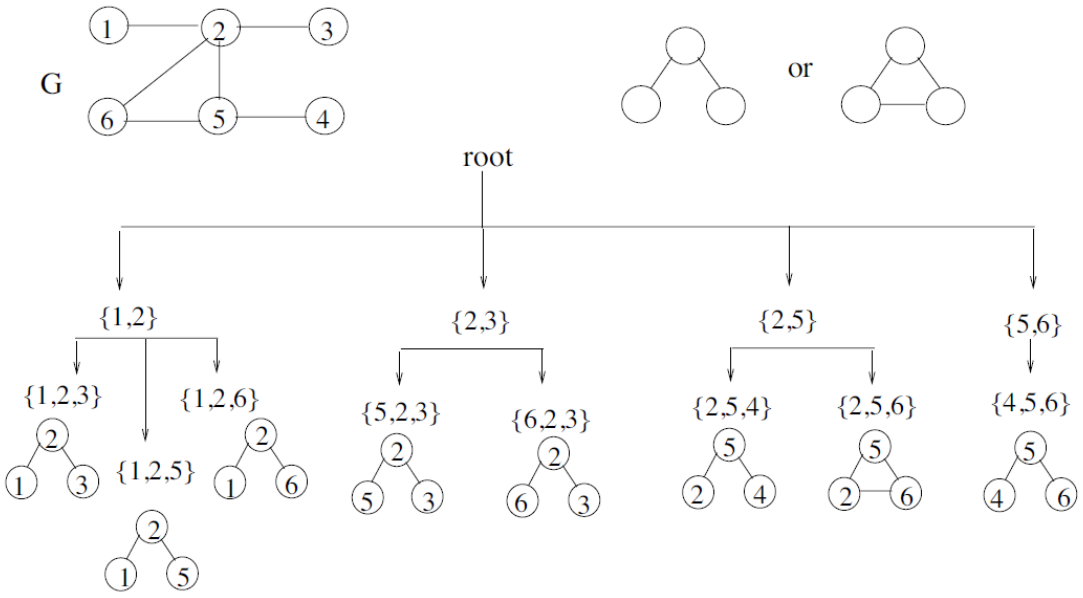
Mfinder

Input : $G(V,E)$ ▷ weighted or unweighted
int $2 \leq k \leq n$
Output : Subgraphs of size k
for all $(u,v) \in E$ **do** ▷ do it for all edges
$Extend(\{u,v\})$
end for
procedure EXTEND (G')
if $|G'| = k$ **then** ▷ if required size is reached
if G' is unique **then**
increment frequency of isomorphic class of G'
else
insert G' in Hash
for all $u \in G'$ **do**
for all $(u,w) \in E$ **do** ▷ check neighbors
if $w \notin G'$ **then**
if $G' \cup \{w\}$ is not in Hash **then**
$Extend(G' \cup \{w\})$
end if
end if
end for
end for
end if
end if
end procedure

Motif Discovery

Mfinder

Enumeración completa



Motif Discovery

ESU [Enumerate SUBgraphs]

- Genera sistemáticamente todos los subgrafos de tamaño k: empezando con un vértice v, añade nuevos vértices ordenadamente para asegurar que cada subgrafo sólo se enumera una vez.
- Algoritmo incorporado en la herramienta FANMOD.
- Utiliza Nauty para calcular isomorfismos.

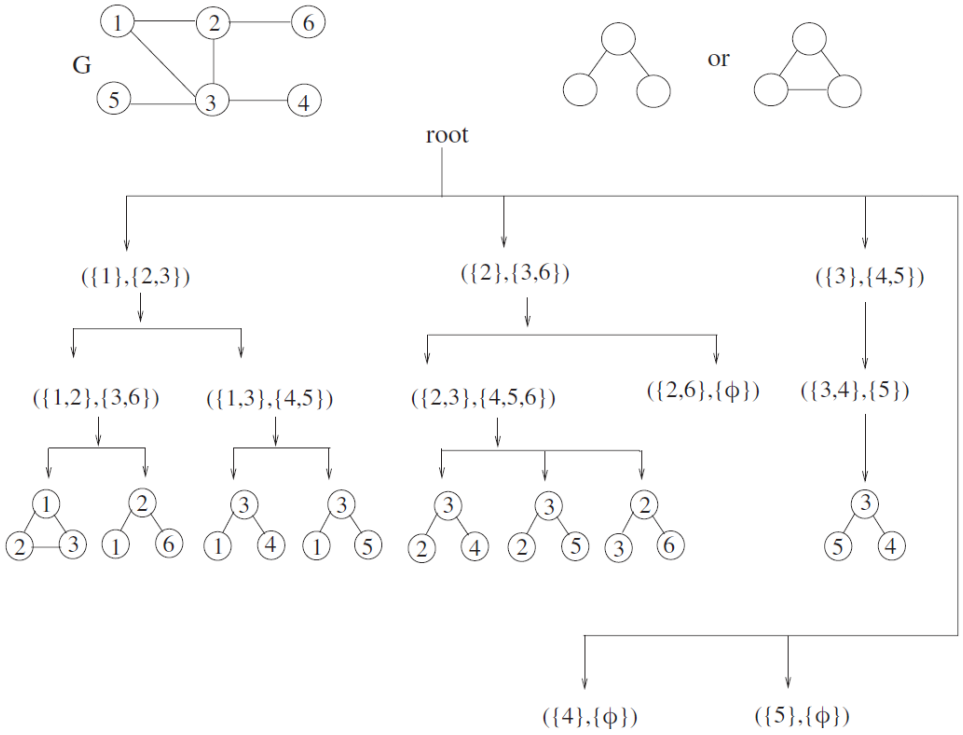
Motif Discovery

ESU [Enumerate SUBgraphs]

Input : $G(V,E)$, **int** $1 \leq k \leq n$
Output : All k-size subgraphs of G
for all $v \in V$ **do**
    $V_{ext} \leftarrow \{u \in N(v) : u > v\}$
    $ExtSubgraph(\{v\}, V_{ext}, v)$
end for
return
procedure EXTSUBGRAPH(V_s, V_{ext}, v)
    if $|V_s| = k$ **then output** $G[V_s]$
        return
    end if
    while $V_{ext} \neq \emptyset$ **do**
        $V_{ext} \leftarrow V_{ext} \setminus \{\text{an arbitrary vertex } w \in V_{ext}\}$
        $V'_{ext} \leftarrow V_{ext} \cup \{u \in N_{excl}(w, V_s) : u > v\}$
        $ExtSubgraph((V_s \cup \{w\}, V'_{ext}, v))$
    end while
    return
end procedure

Motif Discovery

ESU [Enumerate SUBgraphs]



Motif Discovery

Kavosh

Para que cada subgrafo sólo se enumere una vez:

- Se encuentran todos los subgrafos de tamaño k que incluyen un vértice v (explorando un árbol de profundidad k con él vértice v como raíz).
- Se elimina el vértice v del grafo.

NOTA:
También utiliza Nauty para comprobar isomorfismos.

Motif Discovery

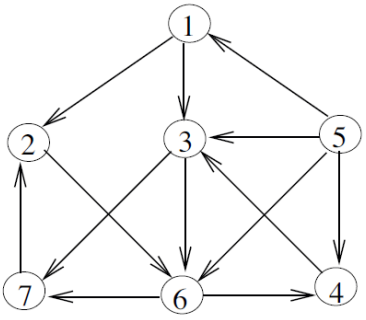
Kavosh

procedure ENUMVERTEX(G, v, S, rem)
  if $rem = 0$ **then**
    return
  else
    $List \leftarrow Validate(G, S_{i-1}, v)$
    $n_i \leftarrow min(|List|, rem)$
    for $k_i = 1$ to n_i **do**
      $C \leftarrow InitialComb(List, k_i)$
      repeat
        $S_i \leftarrow C$
        $EnumVertex(G, v, S, rem - k_i, i+1)$
        $NextComb(List, k_i)$
      until $C = \emptyset$
    end for
    for all $u \in List$ **do**
      $visited[u] \leftarrow false$
    end for
  end if
end procedure

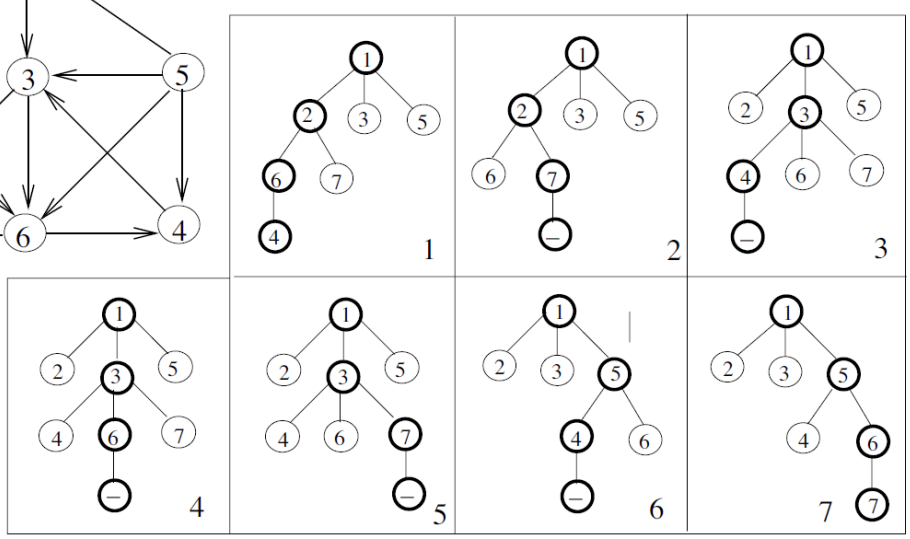
Input : $G(V,E)$ undirected or directed
Output : L : List of all k-size subgraphs of G
for all $v \in V$ **do**
  $visited[v] \leftarrow true$; $S_0 \leftarrow v$
  $EnumVertex(G, v, S, k-1, 1)$
  $visited[v] \leftarrow true$
end for

Motif Discovery

Kavosh



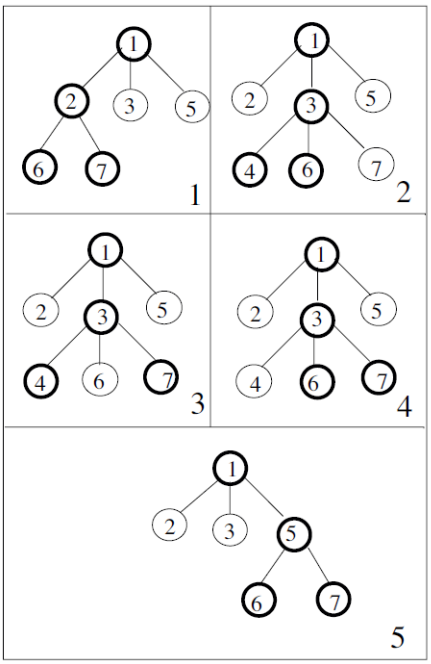
Motifs de tamaño 4 partiendo del nodo 1 (1,1,1)



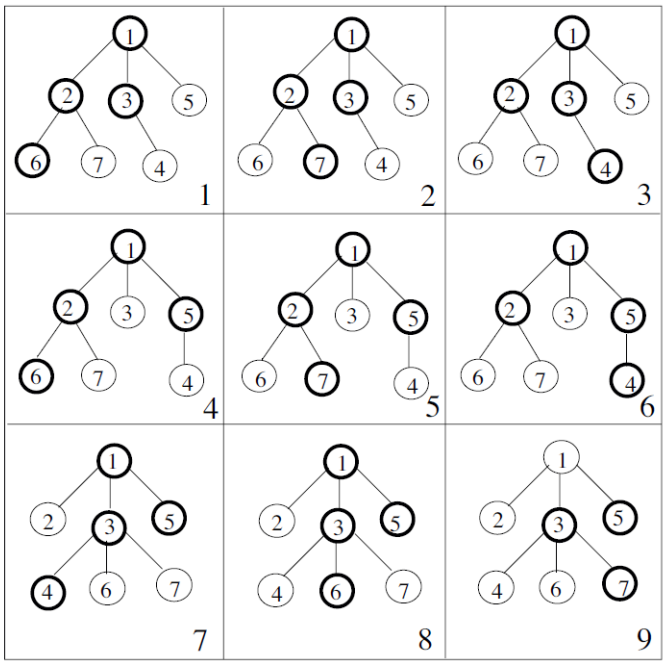
Motif Discovery

Kavosh

(1,2)

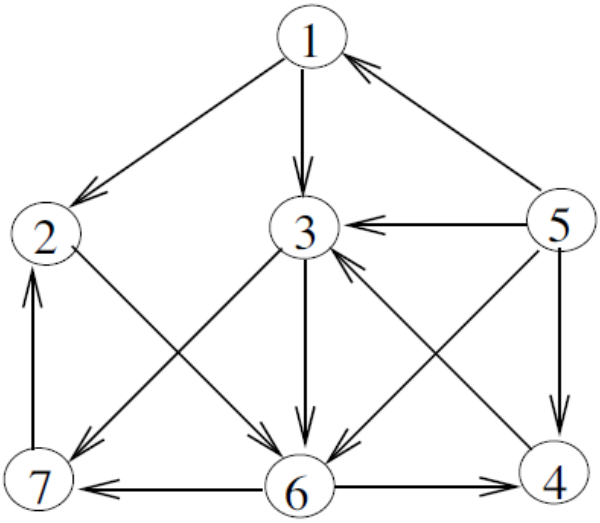


(2,1)

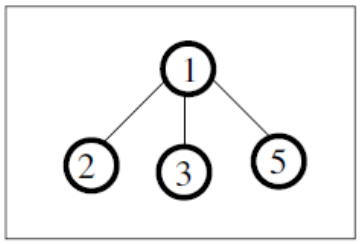


Motif Discovery

Kavosh



(3)



Motif Discovery

MODA

Pattern growth

Árboles de expansión

- Un nodo a nivel k corresponde a un grafo de tamaño k con k-1 aristas (la raíz está a nivel 0).
- El número de nodos del primer nivel corresponde al número de árboles no isomórficos de tamaño k.
- Cada nodo es un subgrafo de sus hijos.
- La única hoja a nivel $(k^2-3k+4)/2$ corresponde al grafo completo K_k.
- El camino más largo de la raíz a la hoja tiene longitud $(k^2-3k+4)/2$.

Motif Discovery

MODA

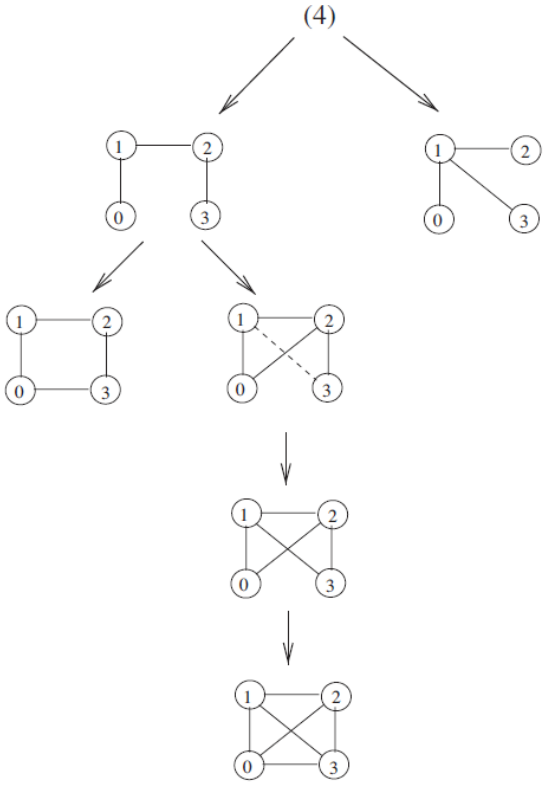
Input : $G(V,E)$, **int** $1 \leq k \leq n$, Δ : threshold value
Output : L : List of frequent k-size subgraphs of G
repeat
  $G'(V',E') \leftarrow Get\_Next\_BFS(T_k)$
  if $|E'| = k-1$ **then**
    $MappingModule(G',G)$
  else
    $EnumeratingModule(G',G,T_k)$
  end if
  save F_2
  if $|F_G| > \Delta$ **then**
    $L \leftarrow L \cup G'$
  end if
until $|E'| = (k-1)/2$

procedure ENUMERATINGMODULE(G',G,T_k)
  $F_G \leftarrow \emptyset$
  $H \leftarrow Get\_Parent(G',T_k)$
  get F_H from memory
  $(u,v) \leftarrow E(G') - E(H)$
  for all $f \in F_H$ **do**
    if $f(u), f(v) \in G$ **and** $< f(u), f(v) >$ violates the corresponding conditions **then**
      add f to F'_G
    end if
  end for
  return F_G
end procedure
procedure MAPPINGMODULE(G,G')
  $Grochow\_Kellis(G',G)$
end procedure

Motif Discovery

MODA

Pattern growth



Motif Discovery

Algoritmos aproximados

PROBLEMA:

El número de subgrafos crece exponencialmente
tanto con el tamaño de la red
como con el tamaño del patrón investigado.

SOLUCIÓN:

Utilización de algoritmos probabilísticos aproximados.

- Se muestrean subgrafos del tamaño requerido a partir del grafo original.
- La precisión (y el coste computacional) aumenta cuantas más muestras utilicemos.

Motif Discovery

Mfinder con muestreo

Selección aleatoria de aristas

procedure EDGESAMPLE(G,k)
  Input : $G(V,E)$, ▷
    int $2 \leq k \leq n$
  Output : A subgraph of size k
  $E_s \leftarrow \emptyset$; $V_s \leftarrow \emptyset$
  pick a random edge $(u,v) \in E$
  $E_s \leftarrow (u,v)$; $V_s \leftarrow \{u,v\}$
  while $|V_s| \neq k$ **do**
    $L \leftarrow$ neighbor edges of $\{u,v\}$
    $L \leftarrow L \setminus$ { all edges between the vertices in V_s}
    if $L = \emptyset$ **then exit**
    end if
    pick a random edge $(w,z) \in L$
    $V_s \leftarrow V_s \cup \{w,z\}$
    $E_s \leftarrow E_s \cup (w,z)$
  end while
  return V_s
end procedure

Motif Discovery

ESU probabilístico

Exploración probabilística del árbol ESU

Input : $G(V,E)$, **int** $1 \leq k \leq n$
Output : All k-size subgraphs of G
for all $v \in V$ **do**
  $V_{ext} \leftarrow \{u \in N(v) : u > v\}$
  With probability P_d
    $Ext\_Subgraph(\{v\}, V_{ext}, v)$
end for

procedure $Ext\_Subgraph(V_s, V_{ext}, v)$
  if $|V_s| = k$ **then output** $G[V_s]$
    return
  end if
  while $V_{ext} \neq \emptyset$ **do**
    $V_{ext} \leftarrow V_{ext} \setminus$ {an arbitrary vertex $w \in V_{ext}$}
    $V_{ext} \leftarrow V_{ext} \cup \{u \in N_{excl}(w, V_s) : u > v\}$
    $V'_s \leftarrow V_s \cup \{w\}$
    With probability $P_{|V'_s|}$
      $Ext\_Subgraph((V_s \cup \{w\}, V'_{ext}, v))$
  end while
  return
end procedure

Motif Discovery

MODA con muestreo

Estimación de la frecuencia de un subgrafo

Input : network graph $G(V,E)$, query graph $G'(V',E')$
Output : approximate frequency and mapping of G'
procedure MAPSAMPLE(G')
  for $i = 1$ to $n_samples$ **do**
    select $u \in V$ with probability $deg(v)$
    for all $v \in V'$ **do**
      if $deg(u) \geq deg(v)$ **then**
        $Grochow(G')$ with $f(v) = u$
      end if
      $V \leftarrow V \setminus \{u\}$
    end for
  end for
end procedure

Motif discovery

¿De qué tamaño podemos encontrar motifs?

Sistema utilizado	Tamaño máximo de los motifs descubiertos
Mfinder	5
Mfinder con muestreo	6
FPF	9
NeMoFinder	12

Graph500

Supercomputación sobre grafos

http://www.graph500.org/ Noviembre 2015

Sistema	País	Nodos	Cores	GTEPS
Fujitsu K	Japón	82944	663552	38621
IBM BlueGene/Q	USA	98304	1572864	23751
Tianhe-2	China	8192	196608	2061
SGI UV 2000	USA	1	1280	174
i5 + NVIDIA GPU	USA	1	28	132
i7	USA	1	4	1.28
Apple MacBook Air	USA	1	2	1.22
Apple iPad	USA	1	2	0.03

Benchmark basado en algoritmos sobre grafos: 3 kernels

- Búsqueda (concurrente),
- Optimización (caminos mínimos)
- Procesamiento de aristas [maximal independent set]

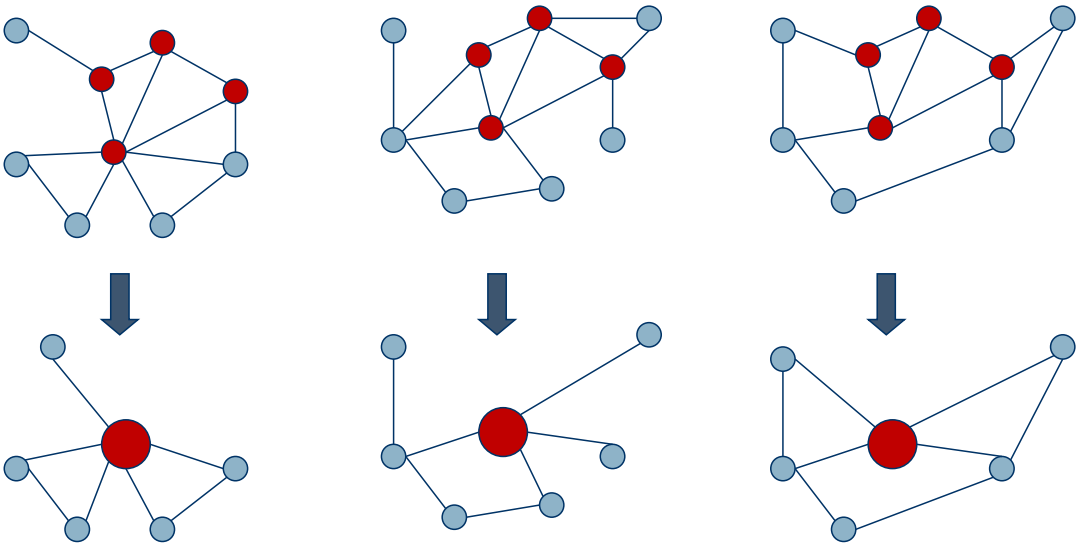
Aplicaciones

- Compresión de datos
- Indexación
- Recuperación de información
- Redes de interacción de proteínas (PPI)

Aplicaciones: Compresión

Compresión de datos

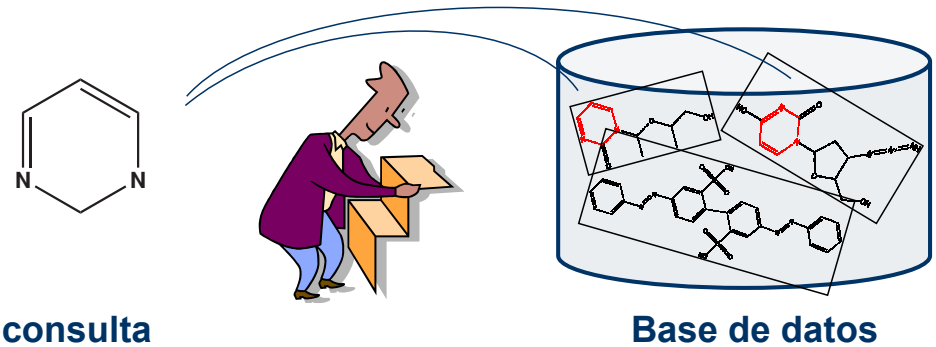
Extraer subgrafos comunes
y condensar éstos en un solo nodo



Aplicaciones: Indexación

Indexación

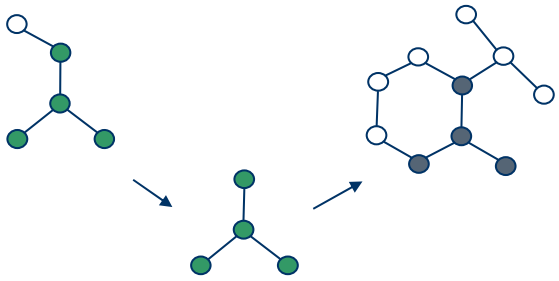
En consultas sobre bases de datos de grafos, recorrer secuencialmente toda la base de datos sería demasiado ineficiente tanto por las operaciones de E/S como por las comprobaciones de isomorfismo entre grafos



p.ej. GraphGrep, Grace, gIndex

Aplicaciones: Indexación

Si un grafo G contiene el grafo Q, G debe contener cualquier subestructura de Q:



Indexar las subestructuras del grafo Q para podar los grafos que no contienen esas subestructuras:

- Construcción del índice: Enumerar estructuras para construir un índice invertido (estructuras → grafos).
- Consulta: Obtener candidatos (grafos que contienen las subestructuras encontradas en el grafo de consulta) y podar los falsos positivos (mediante un test de isomorfismo entre grafos).

Aplicaciones: Indexación

¿Qué estructuras se incluyen en el índice?

- Caminos.
- Estructuras de interés
- Estructuras frecuentes.
- Estructuras discriminantes.

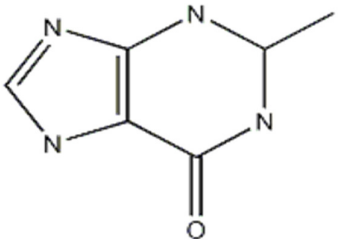
IDEA:
Cuanto más se reduzca el número de falsos positivos, menor será el tiempo de respuesta

$$T_{index} + |C_q| \times \left(T_{io} + T_{isomorphism\_testing} \right)$$

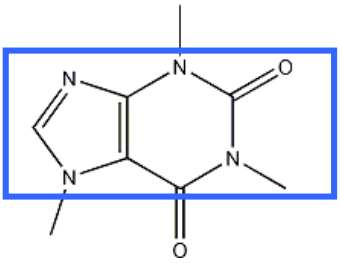
Aplicaciones: SRI

Recuperación de información

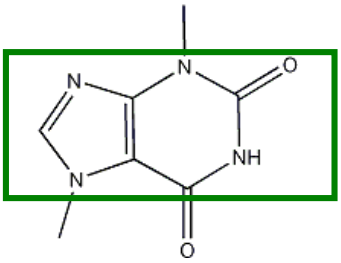
Grafo de la consulta



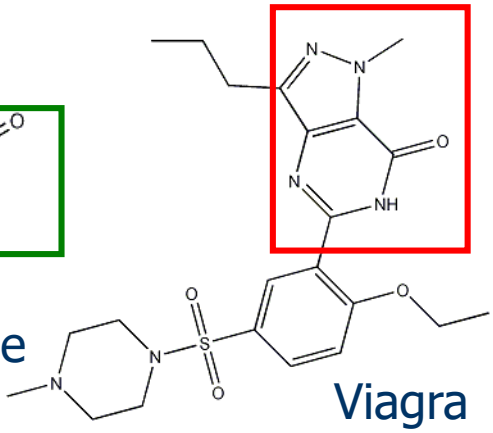
Resultado de la consulta



cafeína



diurobromine



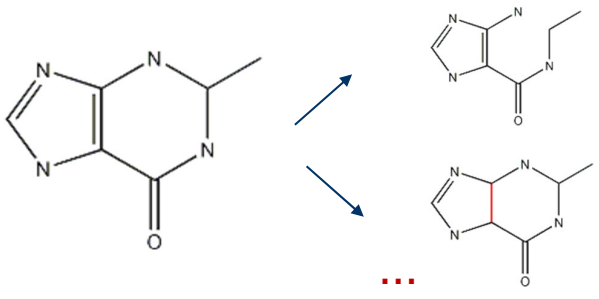
Viagra

Aplicaciones: SRI

Alternativas de diseño:

Soluciones exactas (problema NP-completo)

1. Calcular la similitud entre los grafos de la base de datos y el grafo de consulta (recorrido secuencial)
2. Crear subgrafos del grafo de consulta y hacer una búsqueda exacta (tendremos que probar multitud de subgrafos si queremos encontrar todos los grafos que sean "aproximadamente" iguales al grafo de consulta).

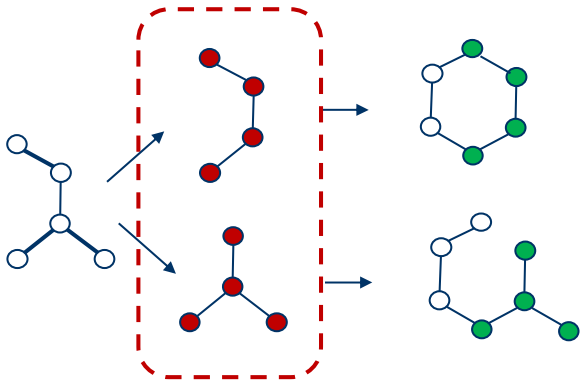


Aplicaciones: SRI

Alternativas de diseño:

Soluciones aproximadas (heurísticas P)

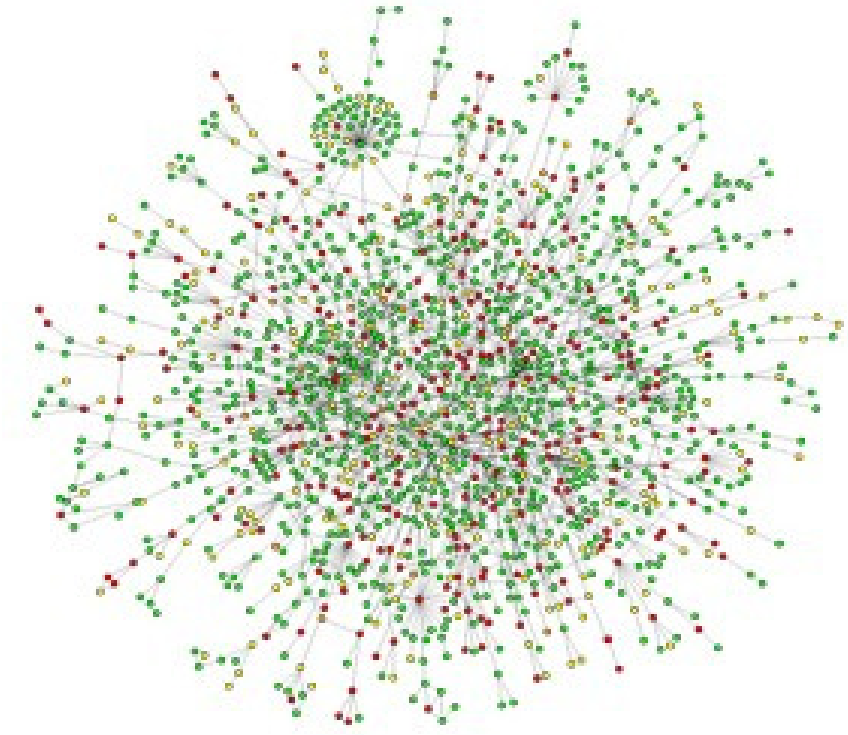
3. Similitud "subestructural"
   Selección de características e indexación
   p.ej. **Grafil**



IDEA: Si un grafo G contiene al grafo de consulta Q, G debería compartir características con Q.

Aplicaciones: PPI

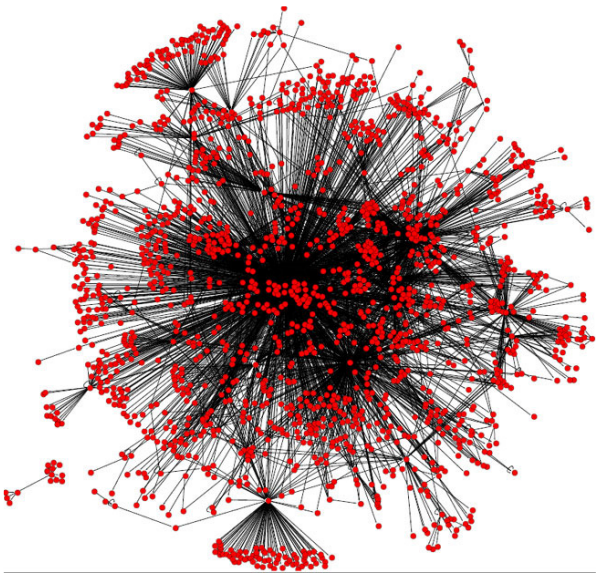
Interacciones de la proteína de la levadura

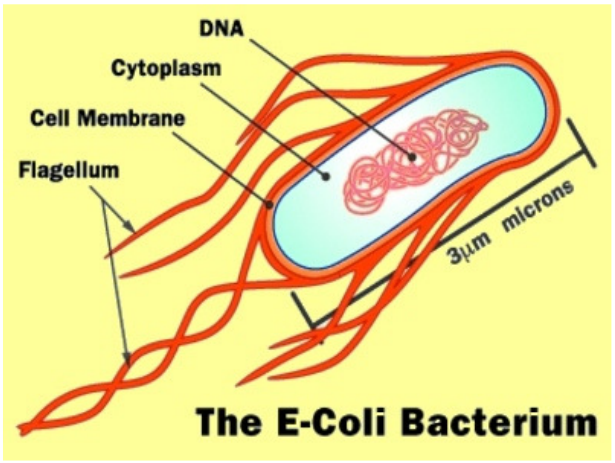


from H. Jeong et al Nature 411, 41 (2001)

Aplicaciones: PPI

E-coli transcriptional regulatory network

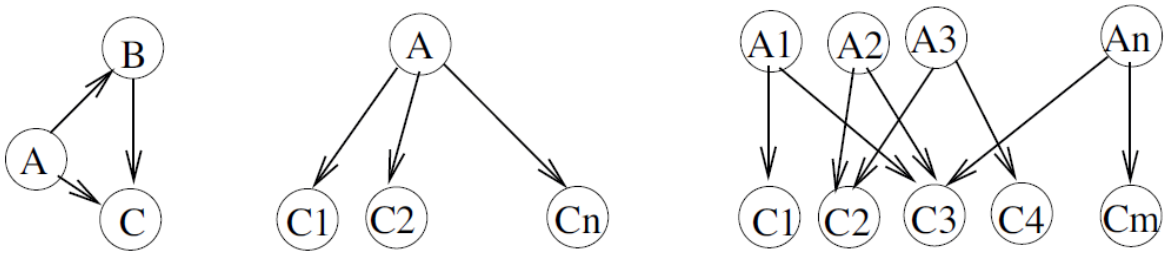




Aplicaciones: PPI

E-coli transcriptional regulatory network

Motifs frecuentes



Feedforward Loop

Single input module (SIM)

Dense overlapping regions (DOR)

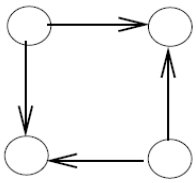
Aplicaciones

Motifs frecuentes en distintos tipos de redes

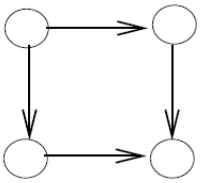
Redes regulatorias de genes

Redes neuronales

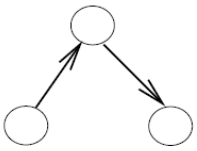
Redes tróficas [food webs]



Bifan



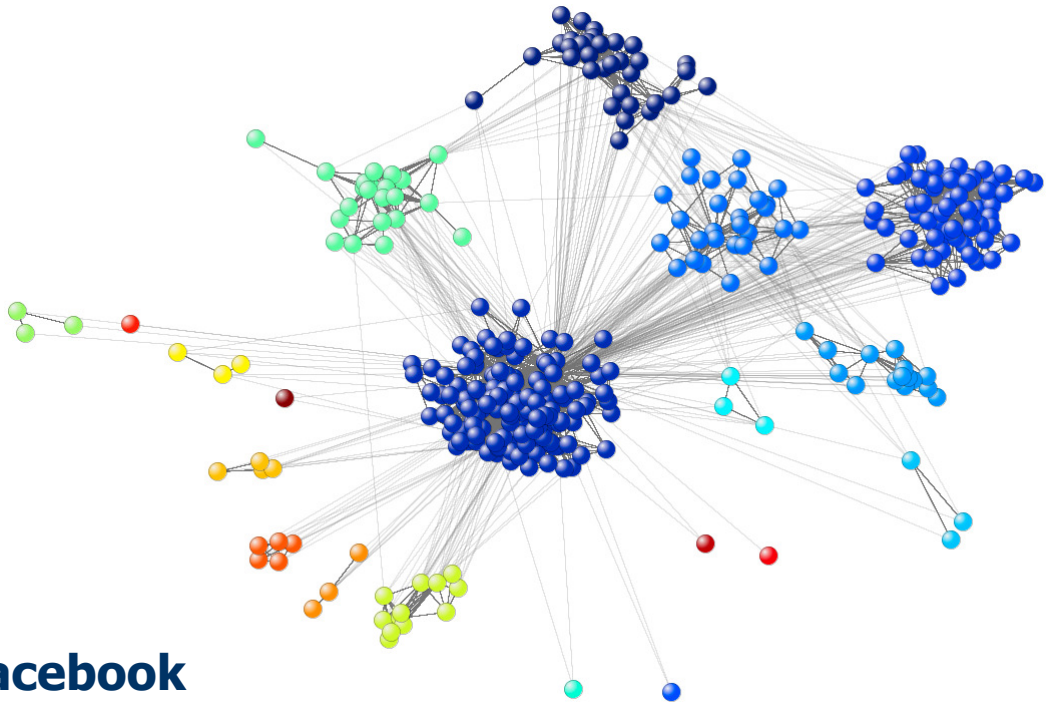
Biparallel



Three-chain

Detección de comunidades

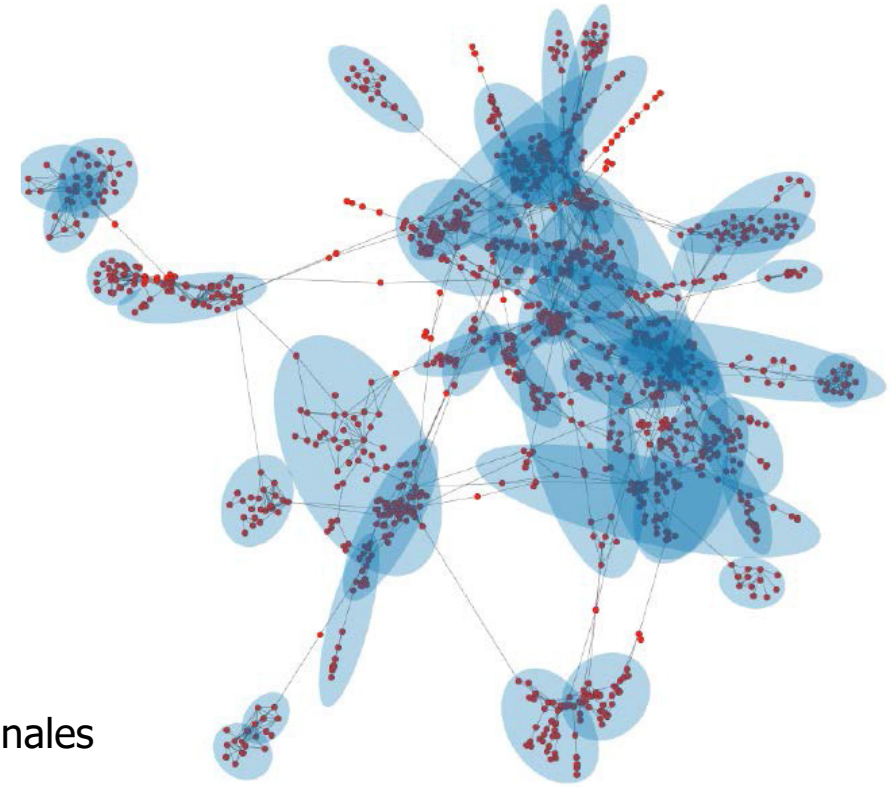
Redes sociales



Facebook

Detección de comunidades

Redes de interacción de proteínas

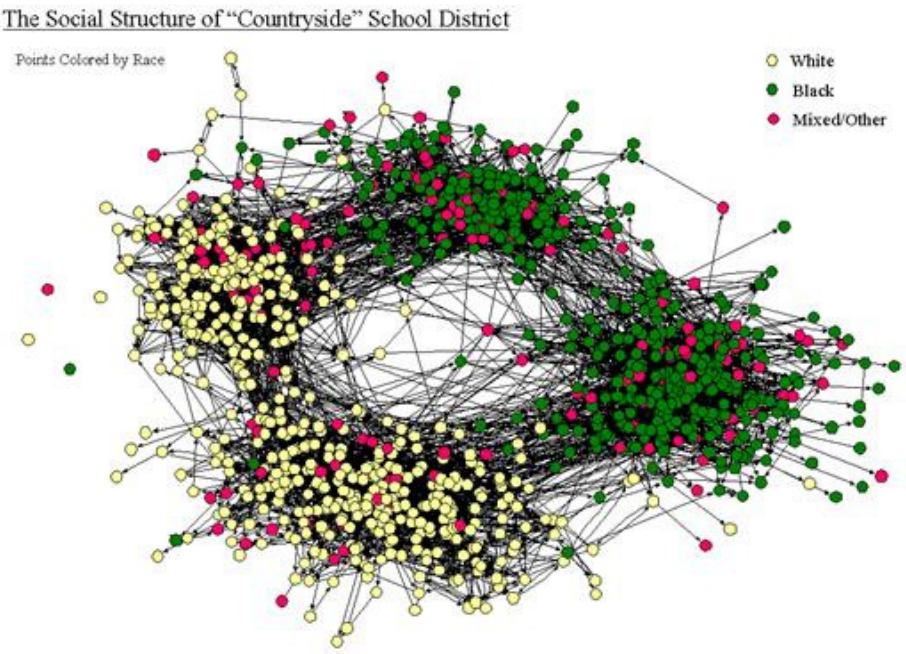


Nodos
Proteínas

Enlaces
Interacciones

Comunidades
Módulos funcionales

Detección de comunidades

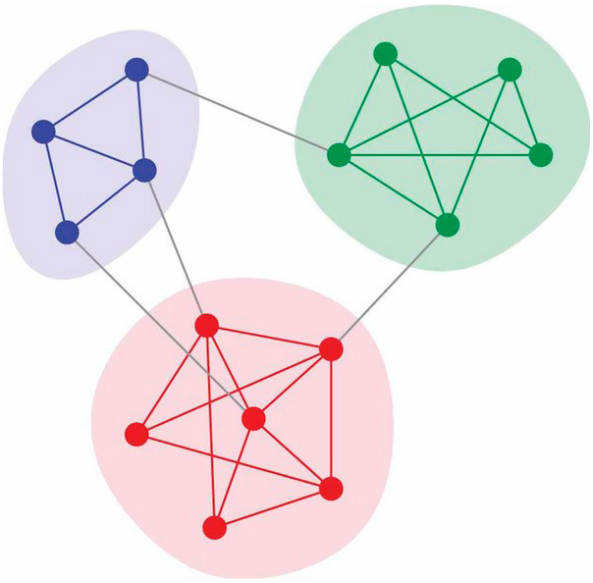


Red social FOAF [Friend of a Friend]

Detección de comunidades

El problema

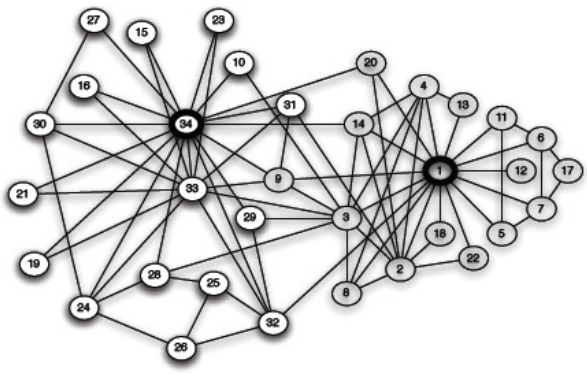
Agrupamiento [clustering] en redes



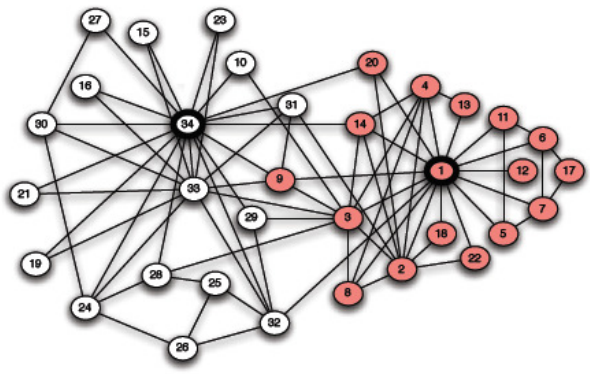
Detección de comunidades

Ejemplo

Club de kárate



(a) *Karate club network*



(b) *After a split into two clubs*

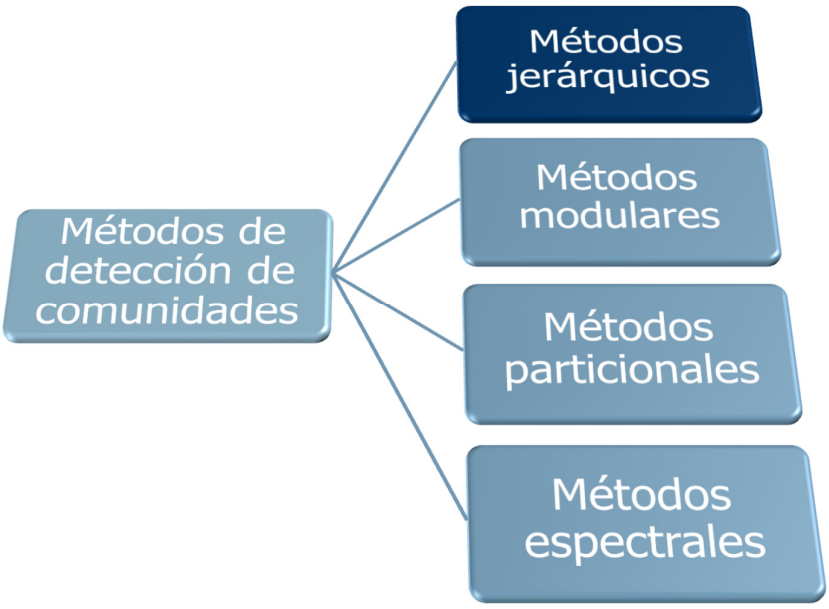
W. W. Zachary:
An information flow model for conflict and fission in small groups,
Journal of Anthropological Research 33:452-473 (1977)

Detección de comunidades

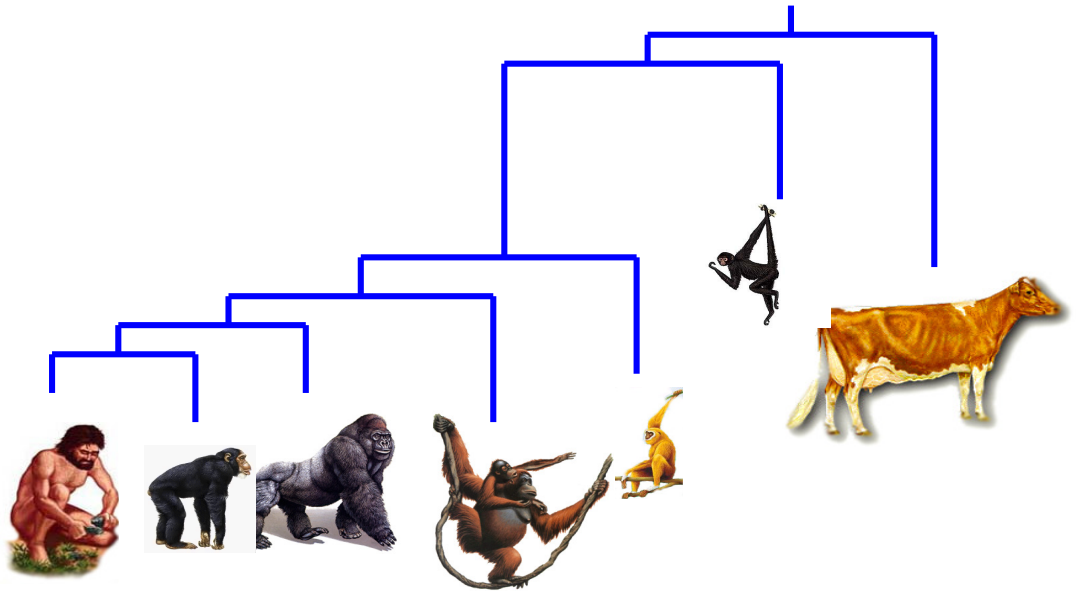
Heurísticas

- Enlaces mutuos & vecinos compartidos
- Frecuencia de enlaces dentro de una comunidad (cliques & k-cores)
- Cercanía [closeness] de los miembros de una comunidad (n-cliques)
- Frecuencia relativa de los enlaces comunidad (enlaces entre miembros de una comunidad frente a enlaces con "no-miembros")

Detección de comunidades



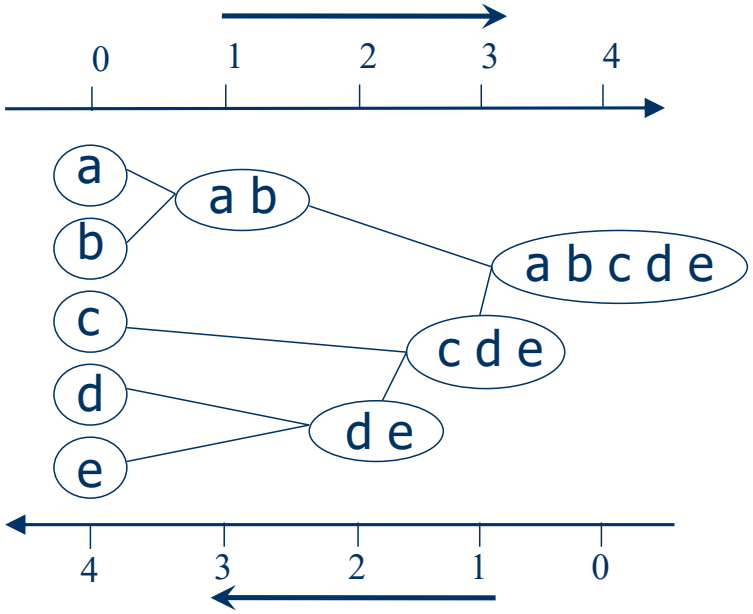
Métodos jerárquicos



DENDROGRAMA: La similitud entre dos objetos viene dada por la "altura" del nodo común más cercano.

Métodos jerárquicos

Métodos aglomerativos
(AGNES: AGglomerative NESting)



Métodos divisivos
(DIANA: Divisive ANAlysis)

Métodos jerárquicos

Métodos jerárquicos aglomerativos

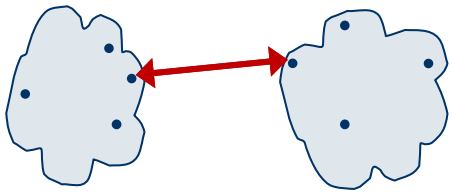
Calcular la matriz de similitud/distancias
Inicialización: Cada caso, un cluster
Repetir
    Combinar los dos clusters más cercanos
    Actualizar la matriz de similitud/distancias
hasta que sólo quede un cluster

- Estrategia de control irrevocable (greedy):
  Cada vez que se unen dos clusters,
  no se reconsidera otra posible unión.

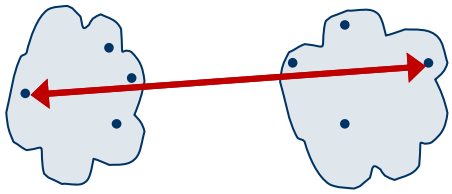
Métodos jerárquicos

¿Cómo se mide la distancia entre clusters?

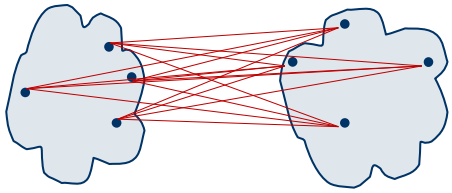
- Mínimo
  [single-link]



- Máximo (diámetro)
  [complete-link]



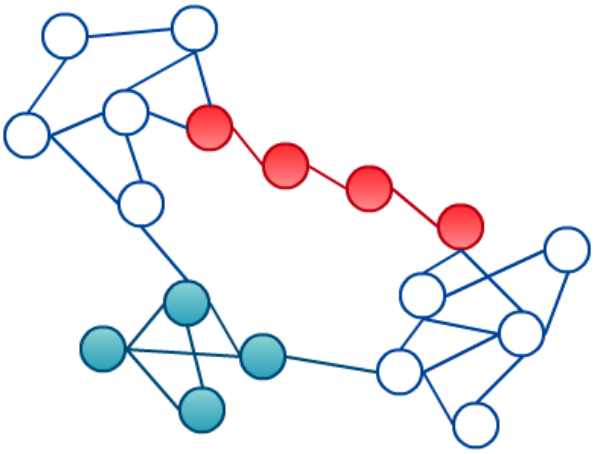
- Promedio
  [averaga-link]



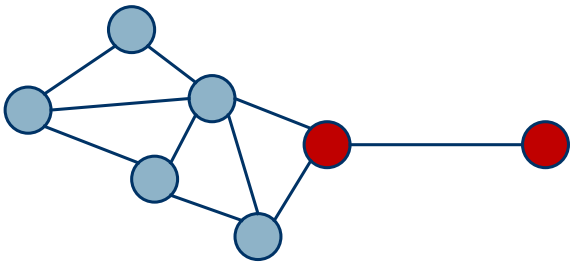
Métodos jerárquicos

Problemas

- Simple-link:
  Encadenamiento



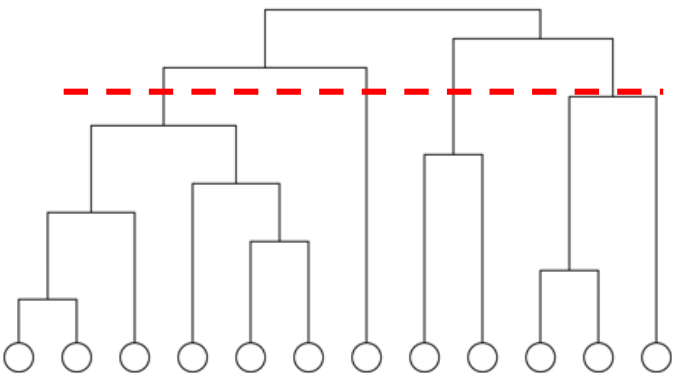
- Complete-link:
  Existencia de outliers



Métodos jerárquicos

Método de Newman & Girvan

Algoritmo jerárquico divisivo

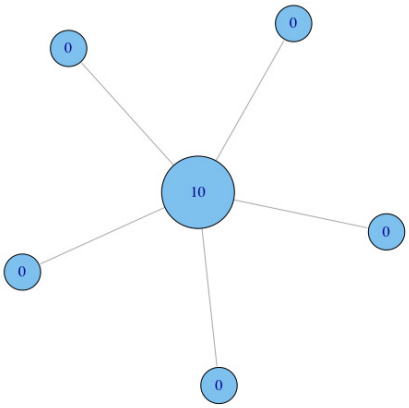


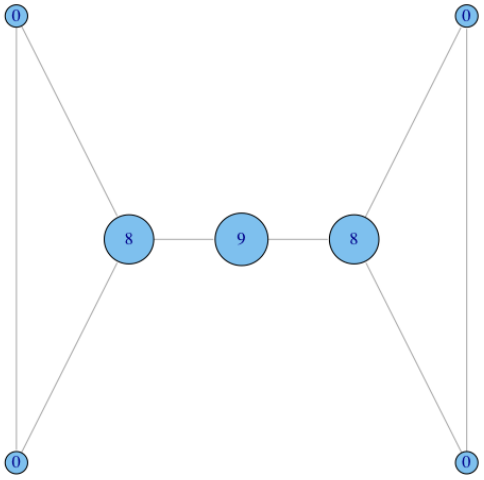
Michelle Girvan & Mark E.J. Newman:
"Community structure in social and biological networks"
PNAS **99**(12):7821–7826, 2002
doi:10.1073/pnas.122653799

Métodos jerárquicos

Método de Newman & Girvan

Betweenness [intermediación]



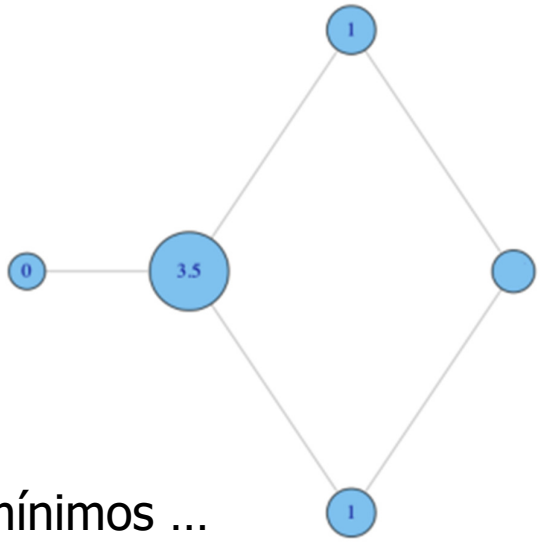


IDEA:
Número de caminos mínimos que pasan por cada nodo como medida de la importancia de ese nodo.

Métodos jerárquicos

Método de Newman & Girvan

Betweenness [intermediación]



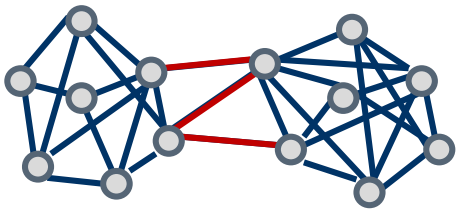
Asignación parcial de crédito
cuando existen varios caminos mínimos ...

La misma idea se puede extender para evaluar la importancia de los enlaces en función del número de caminos mínimos de los que forman parte.

Métodos jerárquicos

Método de Newman & Girvan

Clústering jerárquico utilizando "edge betweenness"



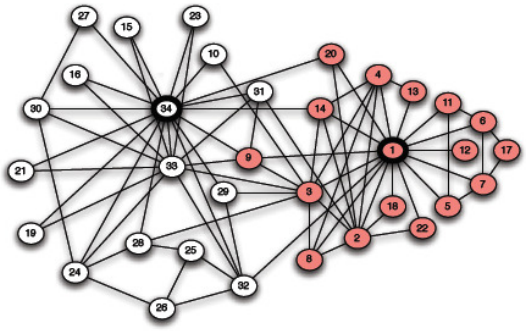
compute the betweenness of all edges
while (betweenness of any edge > threshold)
    remove edge with highest betweenness
    recalculate betweenness

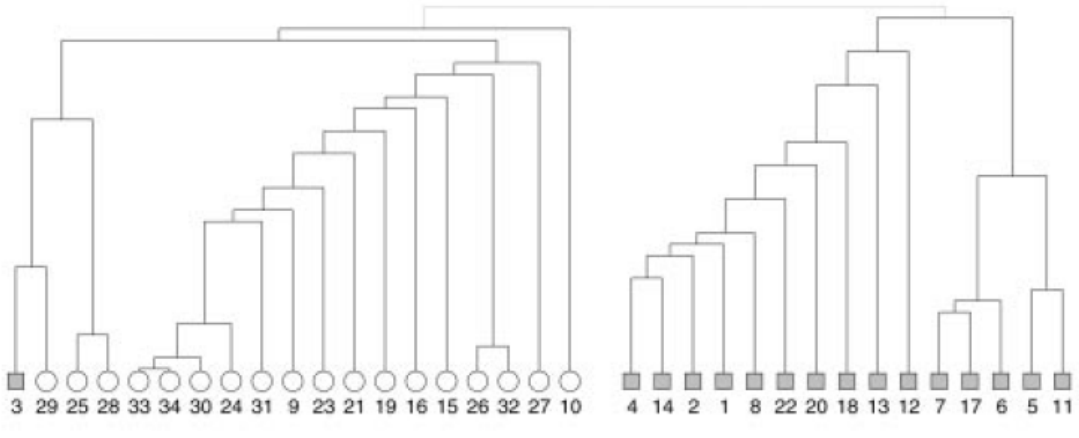
Ineficiente debido a la necesidad de recalcular el "edge betweenness" de todos los enlaces en cada iteración.

Métodos jerárquicos

Método de Newman & Girvan

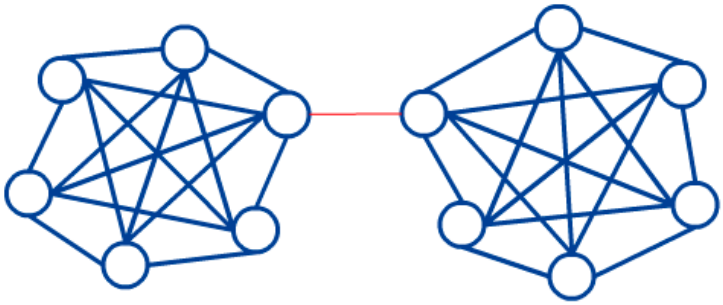
Red del club de kárate





Métodos jerárquicos

Método de Radicchi



IDEA:

Una comunidad contiene nodos muy interconectados entre sí, con muchos ciclos; sin embargo, los enlaces que conectan unas comunidades con otras se ven involucrados en menos ciclos.

Métodos jerárquicos

Método de Radicchi

Coeficiente de agrupamiento

nbr(n)	Vecinos del nodo n en la red.
k	Número de vecinos de u, i.e. |nbr(n)|.
max(n)	Número máximo de enlaces entre los vecinos de n, e.g. k*(k-1)/2.

Coeficiente de clustering para el nodo n:

CC(n) = (#enlaces entre vecinos de n) / max(n)

Métodos jerárquicos

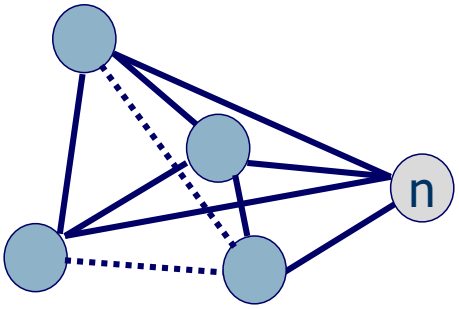
Método de Radicchi

Coeficiente de agrupamiento

k = 4

m = 6

CC(n) = 4/6 = 0.66



0 <= CC(n) <= 1

Similitud del conjunto de vecinos de n a un clique.

Métodos jerárquicos

Método de Radicchi

Coeficiente de agrupamiento de los enlaces

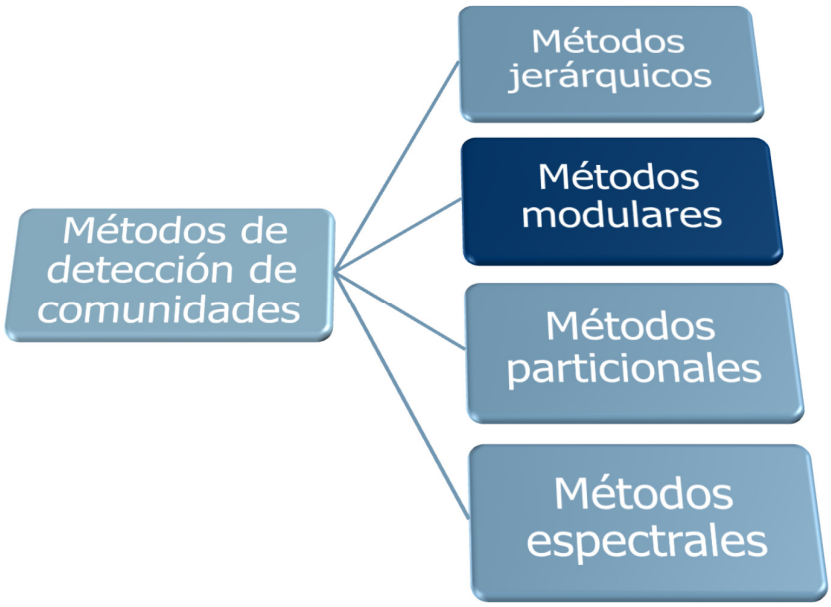
$$C_{ij} = \frac{z_{ij} + 1}{\min(k_i - 1, k_j - 1)}$$

k_i Grado del nodo i

z_{ij} Número de triángulos en los que participan los nodos i y j

Más eficiente que el método de Newman & Girvan.

Detección de comunidades



Métodos modulares

ORIGEN

Medida de modularidad Q

IDEA

Uso del término de "modularidad" como cualquier medida numérica que resulte adecuada para determinar y encontrar comunidades.

La detección de comunidades se convierte en un problema de optimización numérica…

Métodos modulares

Modularidad Q

- Métrica que compara los enlaces internos de una comunidad frente a los enlaces que connectan la comunidad con el resto de la red.

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_v, c_w)$$

Vértices en la misma comunidad

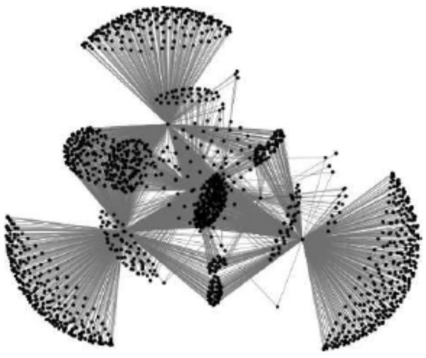
Matriz de adyacencia

Probabilidad de un enlace entre dos vértices (propocional a sus grados)

NOTA: En una red completamente aleatoria, Q=0

Métodos modulares

Algoritmo greedy



start with all vertices as isolates
do
  join clusters with the greatest increase in modularity (ΔQ)
while (ΔQ > 0)

Aaron Clauset, Mark E. J. Newman, Cristopher Moore:
"Finding community structure in very large networks"
Physical Review E 70(6):066111, 2004
doi:10.1103/physreve.70.066111

Métodos modulares

Algoritmo Fast Greedy

FASE 1: Inicialización

Formar pequeños grupos con algún método particional sencillo (tipo K-Means), p.ej. K=n/2

FASE 2: Algoritmo greedy aleatorio

Mientras queden enlaces que mejoren Q:

Seleccionar (de forma aleatoria) un enlace que mejore la modularidad de la red y añadirlo.

Métodos modulares

Algoritmo MultiStep Greedy

IDEA

Se selecciona el enlace que más incrementa la modularidad (ΔQ).

Si no están en contacto, se pueden añadir varios enlaces en la misma iteración del algoritmo greedy.

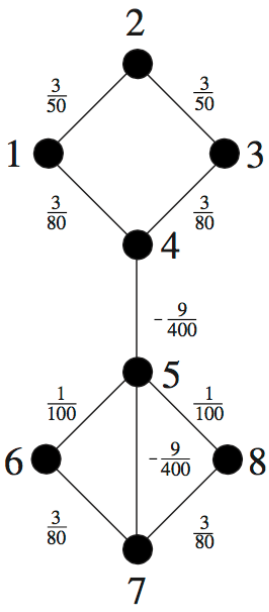
P. Schuetz & A. Caflisch: "Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement" Physical Review E, 77(4):046112, 2008

Métodos modulares

Algoritmo MultiStep Greedy

ESTRUCTURA DE DATOS QMatrix

Mejoras de modularidad asociadas a cada arista (ΔQ_{ij}).



ΔQ_{ij}	i j
3/50	1 2
3/50	2 3
3/80	1 4
3/80	3 4
3/80	6 7
3/80	7 8
1/100	5 6
1/100	5 8
~~-9/400~~	~~4 5~~
~~-9/400~~	~~5 7~~

ALGORITMO

start with all vertices as isolates

do

compute QMatrix

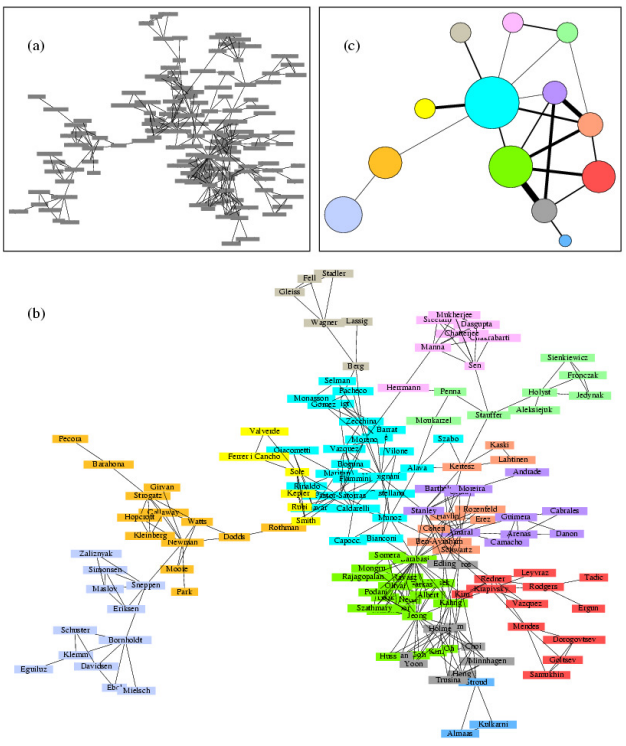
sort QMatrix (descending ΔQ, ascending link)

add non-touching links with greatest increase in modularity (ΔQ_{ij})

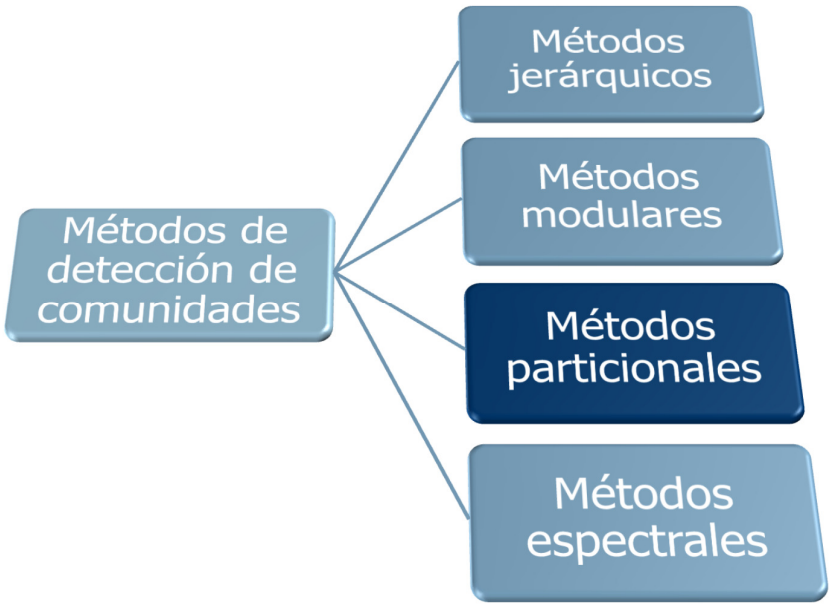
while ($\Delta Q_{ij} > 0$ for some link to be added)

Métodos modulares

Aplicación: Visualización de grandes redes (Gephi)



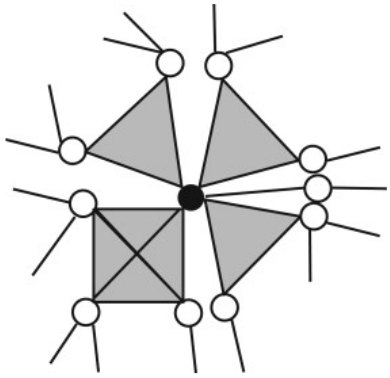
Detección de comunidades



Métodos particionales

Cliques & k-cores

- Cliques (subgrafos completos)
  - La ausencia de un simple enlace descalifica al clique completo
  - Los cliques se solapan.



- K-cores
  (cada nodo, conectado al menos con otros k nodos)

Métodos particionales

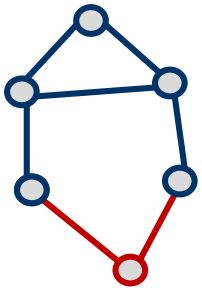
n-cliques

Cualquier pareja de nodos a distancia máxima n

IDEA: Flujo de información a través de intermediarios.

Problemas:

- Diámetro > n
- n-cliques desconectados



2 – clique
diámetro = 3

Camino fuera del 2-clique

Solución: **n-clubs** (subgrafos máximos de diámetro n)

Métodos particionales

Particionamiento sobre un espacio métrico

Técnicas clásicas de clustering basadas en agrupar un conjunto de puntos de un espacio métrico

- **Minimum k-Clustering**
  (intenta minimizar el diámetro de los clusters)
- **Min-Sum k-Clustering** (intenta maximizar la cohesión dentro de los clusters, i.e. la distancia media entre cada par de nodos dentro de cada clúster).
- **K-Center** (intenta minimizar la distancia máxima del centroide a los demás puntos del clúster).
- **K-Means** (intenta minimizar la distancia media del centroide a los demás puntos del clúster)

Métodos particionales

Particionamiento sobre un espacio métrico

K-Means

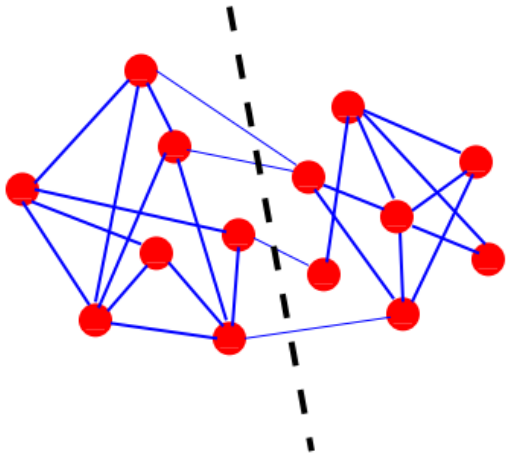
IDEA

- Se transforma la red en un conjunto de puntos de un espacio métrico (p.ej. usando el algoritmo de visualización de redes de Fruchterman-Reingold).
- Se aplica el algoritmo de las k medias.

Métodos particionales

Particionamiento de grafos

Se divide el grafo
en k componentes conexas
intentando minimizar
una **función de corte**.



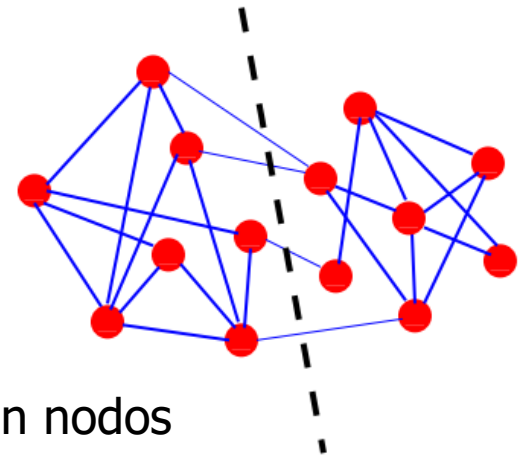
p.ej. Corte mínimo

$$Cut(C_1, C_2, \dots, C_k) = \frac{1}{2}\sum_{i=1}^{k} W(C_i, \overline{C_i}) \qquad W(C_r, C_t) = \sum_{i \in C_r j \in C_t} a_{ij}$$

Métodos particionales

Particionamiento de grafos

Algoritmo de Kernighan-Lin

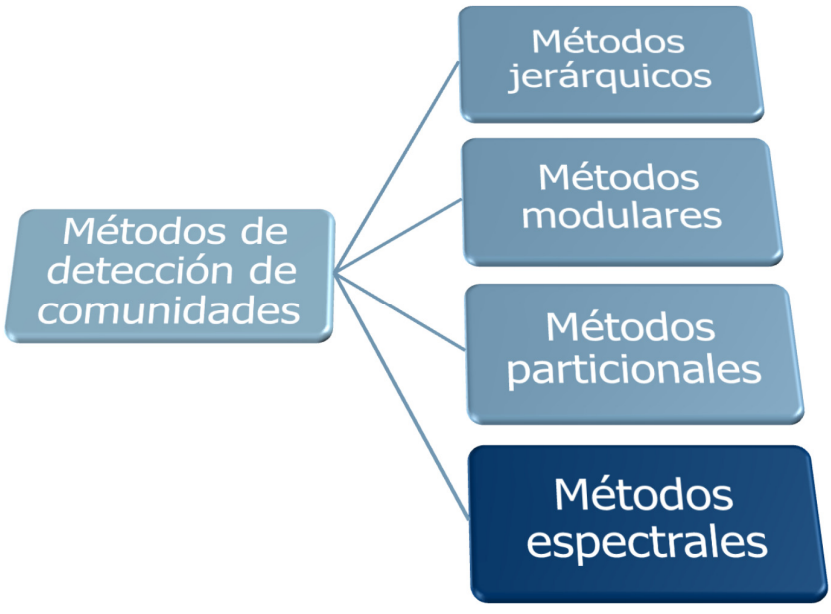


- Bisección mínima (k=2)
- Algoritmo greedy heurístico: Iterativamente, se intercambian nodos para minimizar el corte.
- Selección de parejas de nodos de acuerdo a una función de coste asociado al intercambio.

$$g(i,j) = D_i + D_j - 2w_{ij} \qquad I_i = \sum_{j \in A} w_{ij} \qquad E_i = \sum_{j \in B} w_{ij}$$

$$D_i = E_i - I_i$$

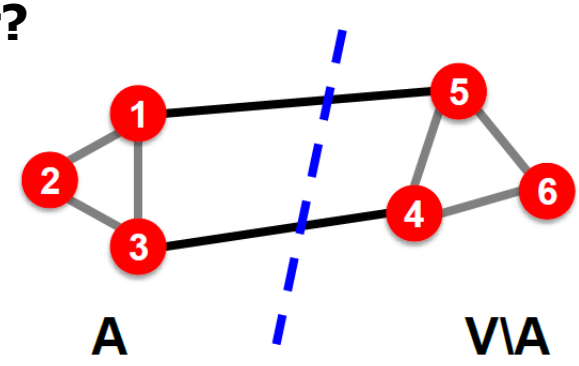
Detección de comunidades



Métodos espectrales

¿Qué hace bueno a un cluster?

- Se maximiza el número de conexiones dentro del cluster.
- Se minimiza el número de conexiones con otros clusters.



$$cut(A) = 2$$

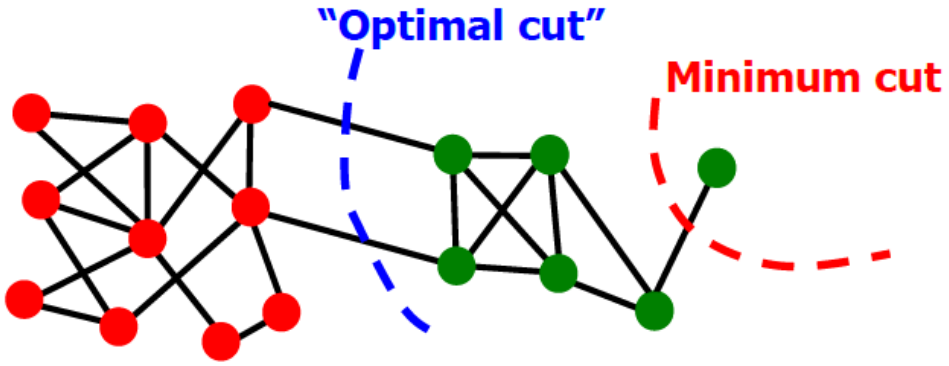
IDEA

Expresar la calidad del cluster como una función del "corte" que separa al cluster del resto de la red.

Métodos espectrales

PROBLEMA

El corte sólo tiene en cuenta conexiones entre clusters.



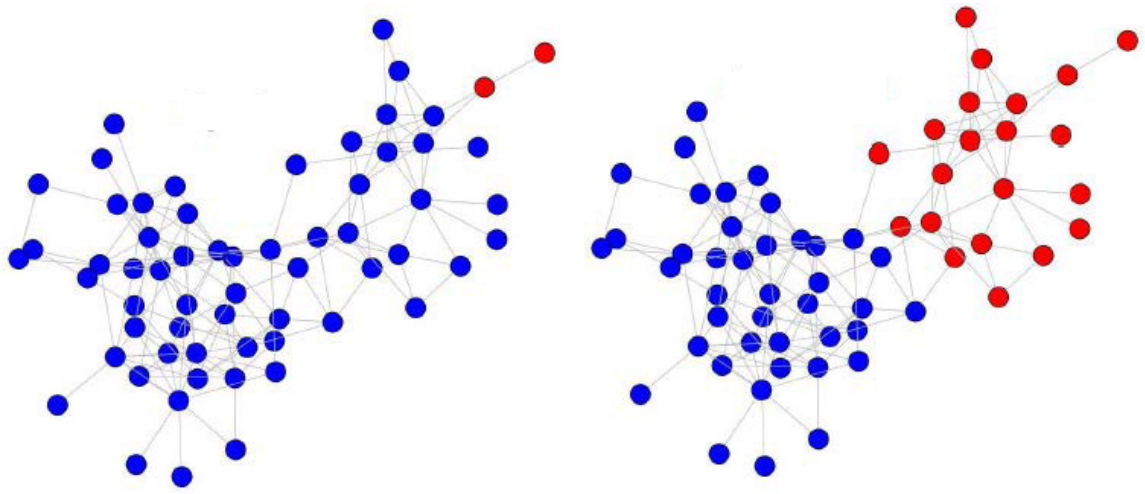
SOLUCIÓN

La **conductancia** (conectividad del grupo con el resto de la red, con respecto a la densidad del grupo) ofrece particiones más balanceadas…

Métodos espectrales

Conductancia

$$\phi(A) = \frac{|\{(i,j) \in E; i \in A, j \notin A\}|}{\min(vol(A),\ 2m - vol(A))}$$

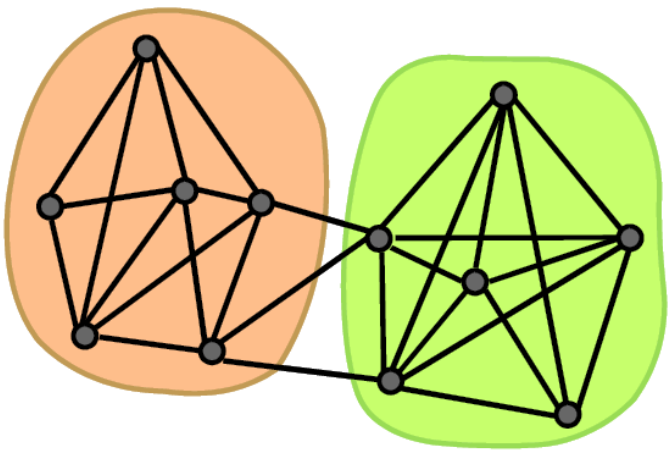


ϕ = 2/4 = 0.5

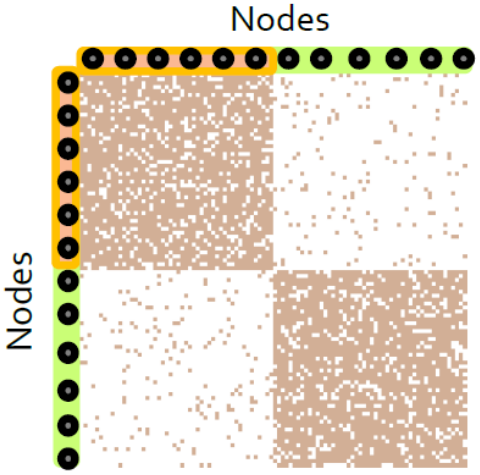
ϕ = 6/92 = 0.065

Métodos espectrales

Encontrar un corte óptimo es un problema NP-duro…



Red



Matriz de adyacencia

Métodos espectrales

A Matriz de adyacencia del grafo G

x Vector de valores asociados a cada nodo de G

Ax Para cada nodo,
suma de los valores asociados a sus vecinos.

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

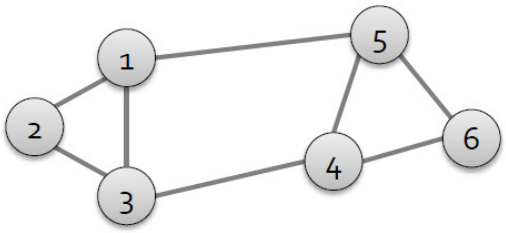
Teoría espectral de grafos: $Ax = \lambda x$

Análisis del "espectro" de la matriz que representa G.

Métodos espectrales

Matriz de adyacencia A

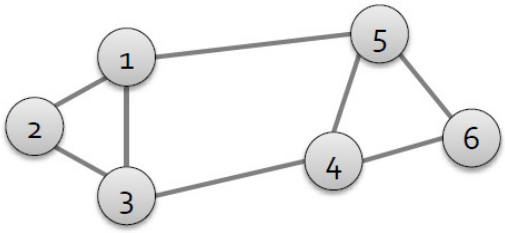
	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0



Matriz simétrica, con eigenvectors reales y ortogonales.

Métodos espectrales

Matriz de grados D

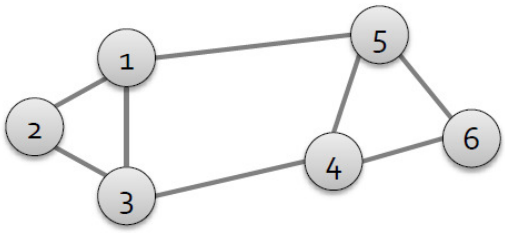


	1	2	3	4	5	6
1	3	0	0	0	0	0
2	0	2	0	0	0	0
3	0	0	3	0	0	0
4	0	0	0	3	0	0
5	0	0	0	0	3	0
6	0	0	0	0	0	2

Matriz diagonal

Métodos espectrales

Matriz laplaciana L = D - A



	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

- Eigenvalues: Números reales no negativos.
- Eigenvectors: Reales y ortogonales.

Métodos espectrales

En un grafo conexo…

- **Primer eigenvalue**
  Eigenvector trivial $x_1=(1,..,1)$

$$\lambda_1 = 0$$

- **Segundo eigenvalue**
  (al ser una matriz simétrica)

$$\lambda_2 = \min_x \frac{x^T M\, x}{x^T x}$$

$$x^T L\, x = \sum_{i,j=1}^{n} L_{ij}\, x_i x_j = \sum_{i,j=1}^{n} (D_{ij} - A_{ij})\, x_i x_j$$

$$= \sum_i D_{ii} x_i^2 - \sum_{(i,j)\in E} 2 x_i x_j$$

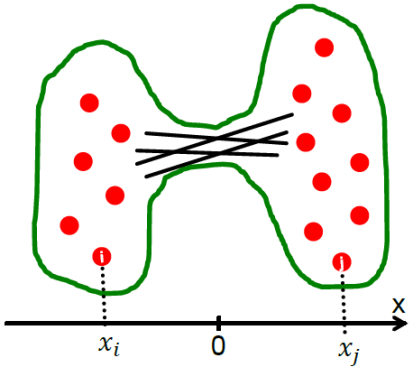
$$= \sum_{(i,j)\in E} (x_i^2 + x_j^2 - 2 x_i x_j) = \sum_{(i,j)\in E} (x_i - x_j)^2$$

Métodos espectrales

¿Qué más sabemos del segundo eigenvector?

- Vector unitario $\sum_i x_i^2 = 1$
- Ortogonal al primer eigenvector $\sum_i x_i \cdot 1 = \sum_i x_i = 0$

$$\lambda_2 = \min_{\substack{\text{All labelings} \\ \text{of nodes } i \text{ so} \\ \text{that } \sum x_i = 0}} \frac{\sum_{(i,j)\in E} (x_i - x_j)^2}{\sum_i x_i^2}$$



Queremos minimizar, por lo que asignaremos los valores x_i de forma que pocas aristas crucen 0 (queremos que x_i y x_j se compensen)

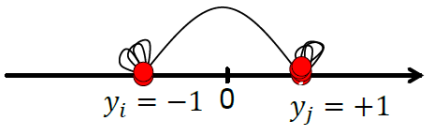
Métodos espectrales

PROBLEMA DE OPTIMIZACIÓN

- Partición (A,B) como vector $$y_i = \begin{cases} +1 & if\ i \in A \\ -1 & if\ i \in B \end{cases}$$

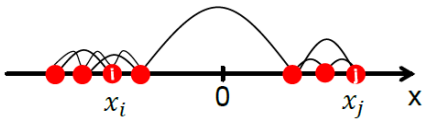
- Minimización del corte

$$\arg\min_{y\in[-1,+1]^n} f(y) = \sum_{(i,j)\in E} (y_i - y_j)^2$$



- Relajación del problema: Teorema de Rayleigh

$$\min_{y\in\Re^n} f(y) = \sum_{(i,j)\in E} (y_i - y_j)^2 = y^T L y$$



Métodos espectrales

Bisección espectral (EIG1)

- Basado en el **autovector de Fiedler** F (el correspondiente al segundo autovalor más pequeño de la matriz laplaciana).

- Para cada valor x_i correspondiente al nodo n_i, si $x_i > \sigma$ lo asociamos al primer cluster; si no, al segundo.

L. Hagen & A.B. Kahng: "New spectral methods for ratio cut partitioning and clustering". IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 11(9):1074-1085, 1992.

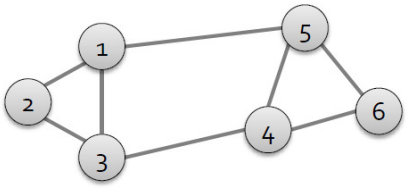
Métodos espectrales

$\lambda =$

0.0
1.0
3.0
3.0
4.0
5.0

$X =$

0.4	0.3	-0.5	-0.2	-0.4	-0.5
0.4	0.6	0.4	-0.4	0.4	0.0
0.4	0.3	0.1	0.6	-0.4	0.5
0.4	-0.3	0.1	0.6	0.4	-0.5
0.4	-0.3	-0.5	-0.2	0.4	0.5
0.4	-0.6	0.4	-0.4	-0.4	0.0



1	0.3
2	0.6
3	0.3
4	-0.3
5	-0.3
6	-0.6

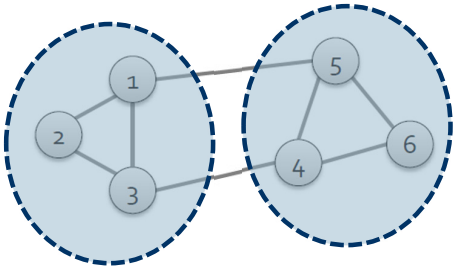
Split at 0:

Cluster A: Positive points

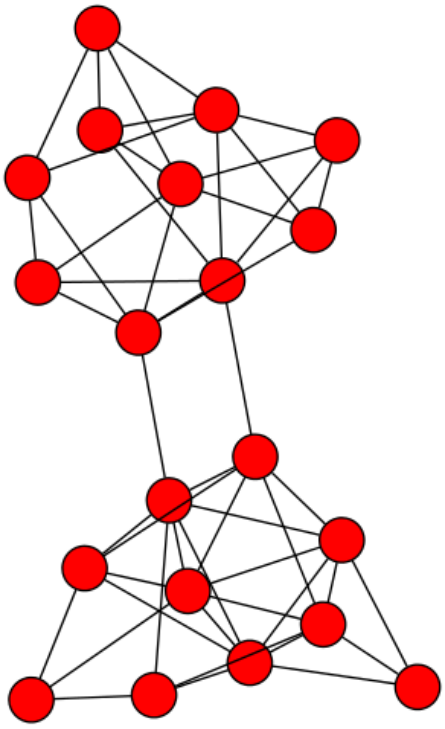
Cluster B: Negative points

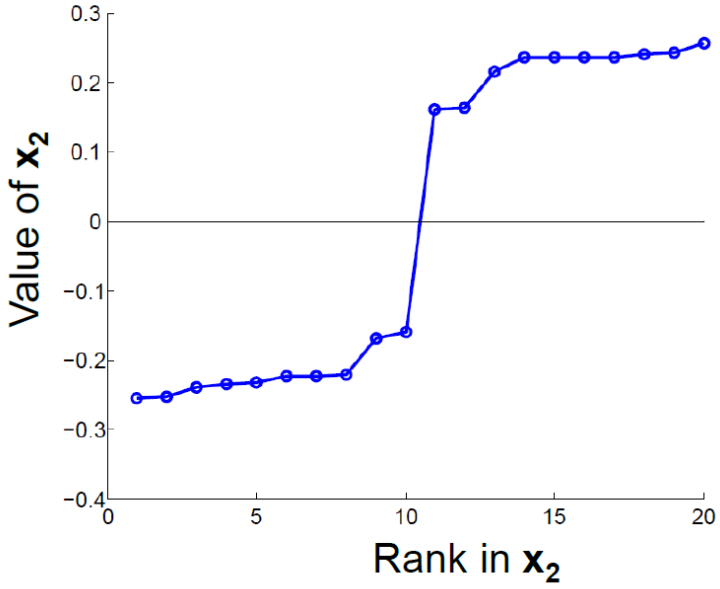
1	0.3
2	0.6
3	0.3

4	-0.3
5	-0.3
6	-0.6



Métodos espectrales

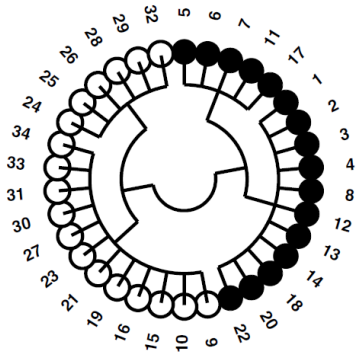


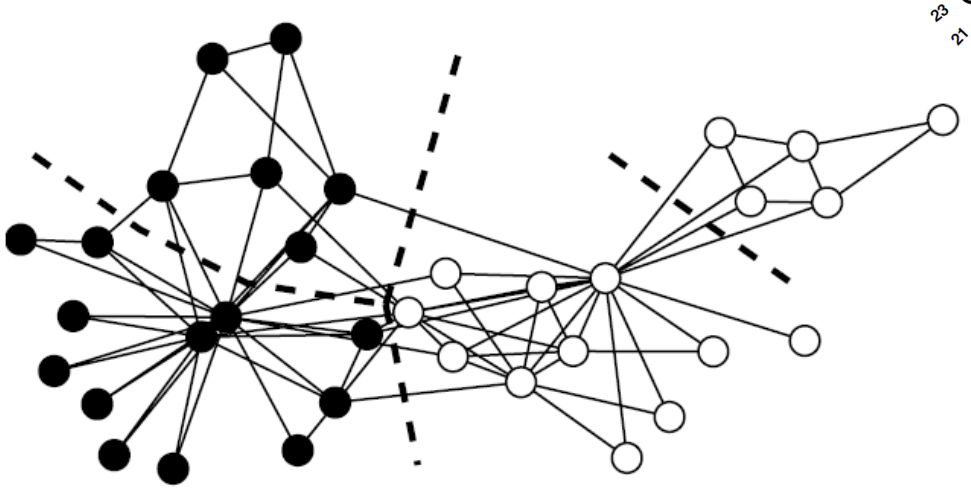


Métodos espectrales

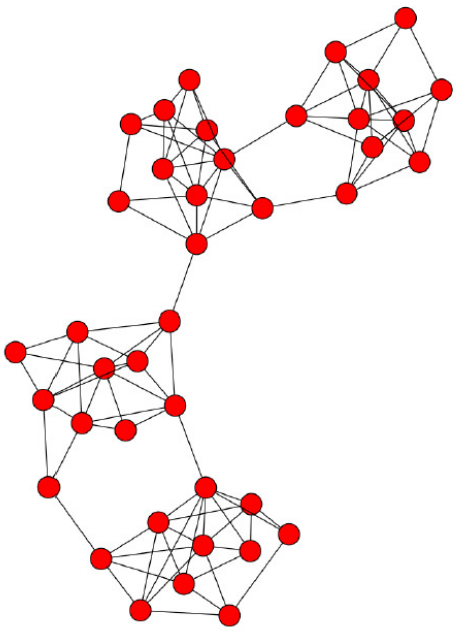
Ejemplo

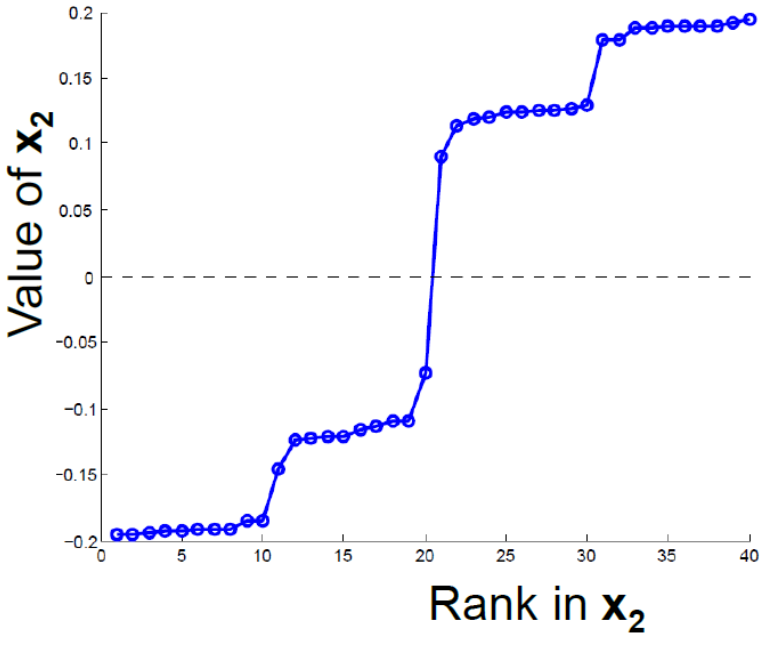
Zachary karate club



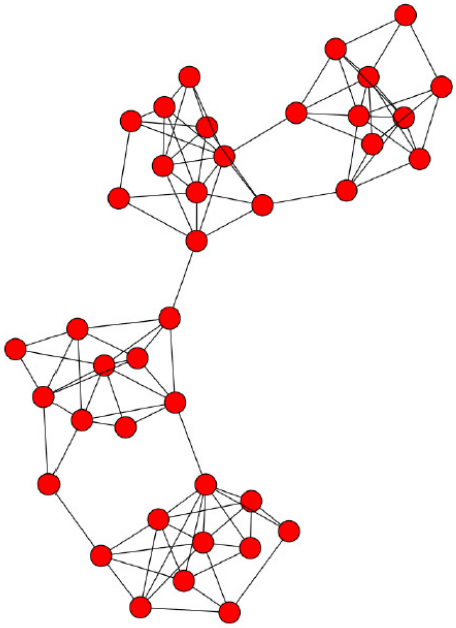


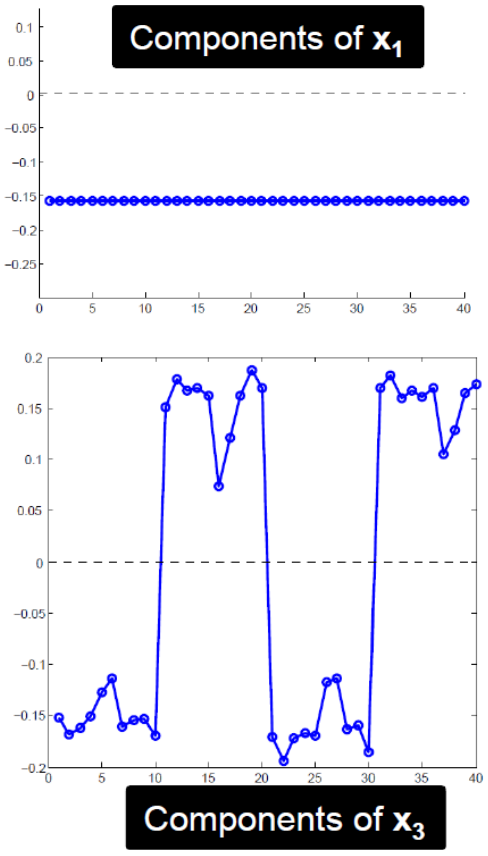
Métodos espectrales





Métodos espectrales





Métodos espectrales

Laplaciano de un grafo

$$L = D - A$$

PROPIEDADES

- El número de autovalores de L iguales a 0 coincide con el número de componentes conexas del grafo.
- Si tenemos k grupos bien definidos en nuestra red, los k primeros autovalores serán cercanos a 0 y sus autovectores asociados nos ayudarán a diferenciar claramente los grupos en un espacio k-dimensional.

Métodos espectrales

Laplaciano de un grafo

Estimación del número de clusters

Si tenemos k particiones bien definidas, los primeros k autovalores de la matriz laplaciana serán cercanos a 0, por lo que es de esperar que el autovalor k + 1 difiera bastante del autovalor k.

NOTA: Para que este método funcione medianamente bien, la red debe tener comunidades claramente definidas.

Métodos espectrales

Laplaciano de un grafo

Normalización

- Normalización simétrica

$$L_{sym} = D^{-1/2} L D^{-1/2}$$

- Normalización asimétrica (por caminos aleatorios)

$$L_{rw} = D^{-1} L$$

Métodos espectrales

IDEA GENERAL

1. Transformar el conjunto de nodos en un conjunto de puntos en un espacio métrico cuyas coordenadas se corresponderán a los k vectores propios más relevantes de la matriz laplaciana del grafo.
2. Agrupar dichos puntos mediante alguna técnica de particionamiento en el espacio métrico.

U. von Luxburg:
"A tutorial on spectral clustering"
Statistics and Computing, 17(4):395-416, 2007.

Métodos espectrales

Algoritmo genérico

1. Calculamos la matriz laplaciana L de nuestra red (normalizada o no).
2. Calculamos los autovalores y autovectores de L.
3. Formamos una matriz U con los k primeros autovectores de L como columnas.
4. Interpretamos las filas de U como vectores de un espacio métrico k-dimensional.
5. Agrupamos los vectores usando cualquier técnica de particionamiento en espacios métricos (p.ej. k-means).

Métodos espectrales

UKMeans

1. Calculamos la matriz laplaciana L sin normalizar.
2. Calculamos autovalores y autovectores de L.
3. Formamos una matriz U con los k primeros autovectores de L como columnas.
4. Interpretamos las filas de U como vectores en un espacio métrico k-dimensional.
5. Agrupamos los vectores usando K-Means.

Métodos espectrales

Algoritmo NJW (a.k.a. KNSC1)

1. Calculamos la matriz laplaciana normalizada simétrica.
2. Calculamos autovalores y autovectores de L_{sym}.
3. Formamos una matriz U con los k primeros autovectores de L como columnas.
4. Realizamos una nueva normalización U' de U.
5. Interpretamos las filas de U' como vectores en un espacio métrico k-dimensional.
6. Agrupamos los vectores de U' usando K-Means.

A.Y. Ng, M.I. Jordan & Y. Weiss: "On spectral clustering: Analysis and an algorithm". Advances in Neural Information Processing Systems, 2:849-856, 2002.

Detección de comunidades

Limitaciones de los métodos descritos

- **Escalabilidad**: Identificación de grandes comunidades.
- Existencia de **solapamiento** entre comunidades.
- Modelos **poco realistas**
  (los algoritmos realizan suposiciones demasiado simplificadas sobre las comunidades de una red, por lo que no funcionan bien con conjuntos de datos reales).
- Técnicas heurísticas **sin garantías**
  (incluso para los algoritmos que funcionan bien en la práctica, no existen garantías sobre la calidad de sus resultados).

Evaluación de resultados

Métricas de evaluación no supervisada

Evaluación global

- **Cohesión**

$$\text{cohesión}(C_i) = \sum_{u,v \in C_i} \text{proximidad}(u, v)$$

- **Separación**

$$\text{separación}(C_i, C_j) = \sum_{\substack{u \in C_i \\ v \in C_j}} \text{proximidad}(u, v)$$

Evaluación de resultados

Métricas de evaluación no supervisada

Evaluación individual de nodos y clusters

- **Coeficiente de silueta**

$$s(v_i) = \frac{b(v_i) - a(v_i)}{\max\left(a(v_i), b(v_i)\right)}$$

a(v)
Distancia media del nodo
a los demás nodos de su cluster.

b(v)
Distancia mínima entre el nodo
y un cluster al que no pertenece.

$$s(C_j) = \frac{1}{m}\sum_{i=1}^{m} s(v_i)$$

$$s(G) = \frac{1}{c}\sum_{j=1}^{c} s(C_j)$$

Evaluación de resultados

Métricas de evaluación no supervisada

Evaluación individual de nodos y clusters

- **Conductancia**

$$\varphi(C_i) = \frac{\mathrm{cut}(C_i)}{\min\left(\mathrm{vol}(C_i), \mathrm{vol}(\bar{C}_i)\right)}$$

$$\varphi(G) = \min\left(\varphi(C_i)\right), C_i \subseteq V$$

  - **… intra-cluster**

$$\alpha(C) = \min \varphi(G[C_i]), \quad i \in \{1, \ldots, k\}$$

  - **… inter-cluster**

$$\sigma(C) = 1 - \max \varphi(C_i), \quad i \in \{1, \ldots, k\}$$

Evaluación de resultados

Métricas de evaluación no supervisada

Evaluación individual de nodos y clusters

- **Cobertura**

$$cov(C_i) = \frac{w(C_i)}{w(G)}$$

- **Rendimiento**

$$perf(C) = 1 - \frac{2m(1 - 2cov(C)) + \sum_{i=1}^{k} |C_i|(|C_i| - 1)}{n(n-1)}$$

Evaluación de resultados

Modularidad Q

- Métrica de evaluación no supervisada que compara los enlaces internos de una comunidad frente a los enlaces que connectan la comunidad con el resto de la red.

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_v, c_w)$$

Vértices en la misma comunidad

Matriz de adyacencia

Probabilidad de un enlace entre dos vértices (propocional a sus grados)

NOTA: En una red completamente aleatoria, Q=0

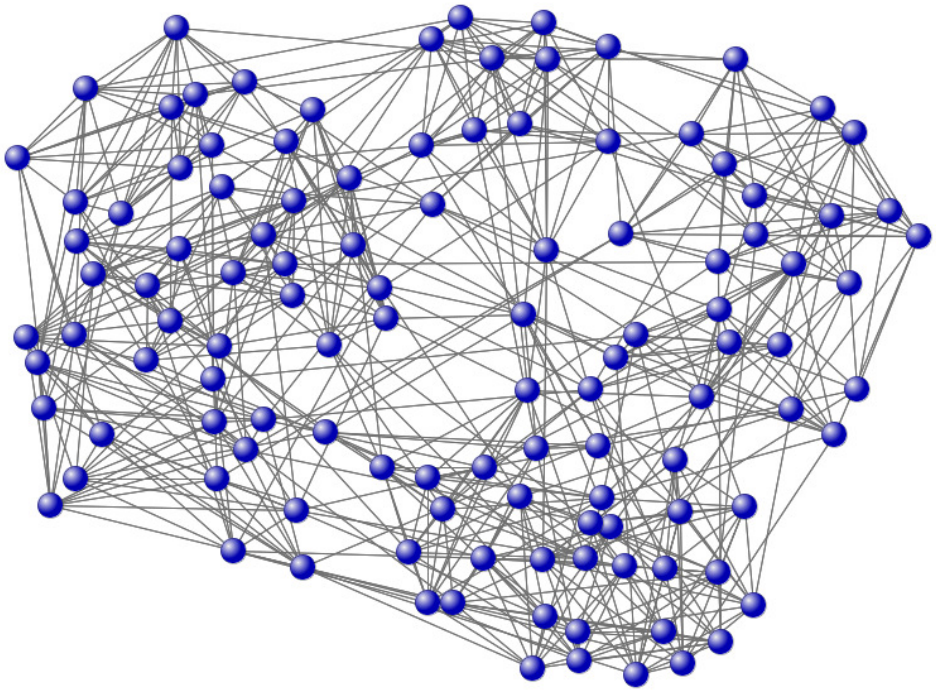
Evaluación de resultados

Métricas de evaluación no supervisada

Enfoque	Medida	Ref.	Rango	Características
Análisis Global	Cohesión	[79]	$[0, \infty]$	Mide las distancias entre nodos dentro de un cluster, se buscan valores pequeños, varía dependiendo de la medida de proximidad.
	Separación	[79]	$[0, \infty]$	Mide las distancias de los nodos del *cluster* con respecto a aquellos que no pertenece, se busca el máximo posible, varía dependiendo de la medida de proximidad.
Análisis individual	Coeficiente de silueta	[68]	$[-1,1]$	Adecuada para comunidades altamente conectadas. Alta complejidad y fallos con nodos hoja.
	Conductancia	[32]	[0,1]	Medición de cuellos de botella, adecuado para *clusters* de gran tamaño; fallos en la evaluación de *clusters* con pocos nodos, pequeños y/o muy grandes.
	Cobertura	[5]	[0,1]	Peso del *cluster*, basado en los cortes mínimos; fallos en la evaluación de *clusters* con pocos nodos, pequeños y/o muy grandes.
	Rendimiento	[5]	[0,1]	Número de nodos adyacentes, densidad; falla en redes de gran tamaño con numero alto de *clusters*.
	Coeficiente de agrupamiento	[64]	[0,1]	Búsqueda de estructuras conexas (triángulos).
	Modularidad	[52]	[0,1]	Comparación del cluster con estructura aleatoria.

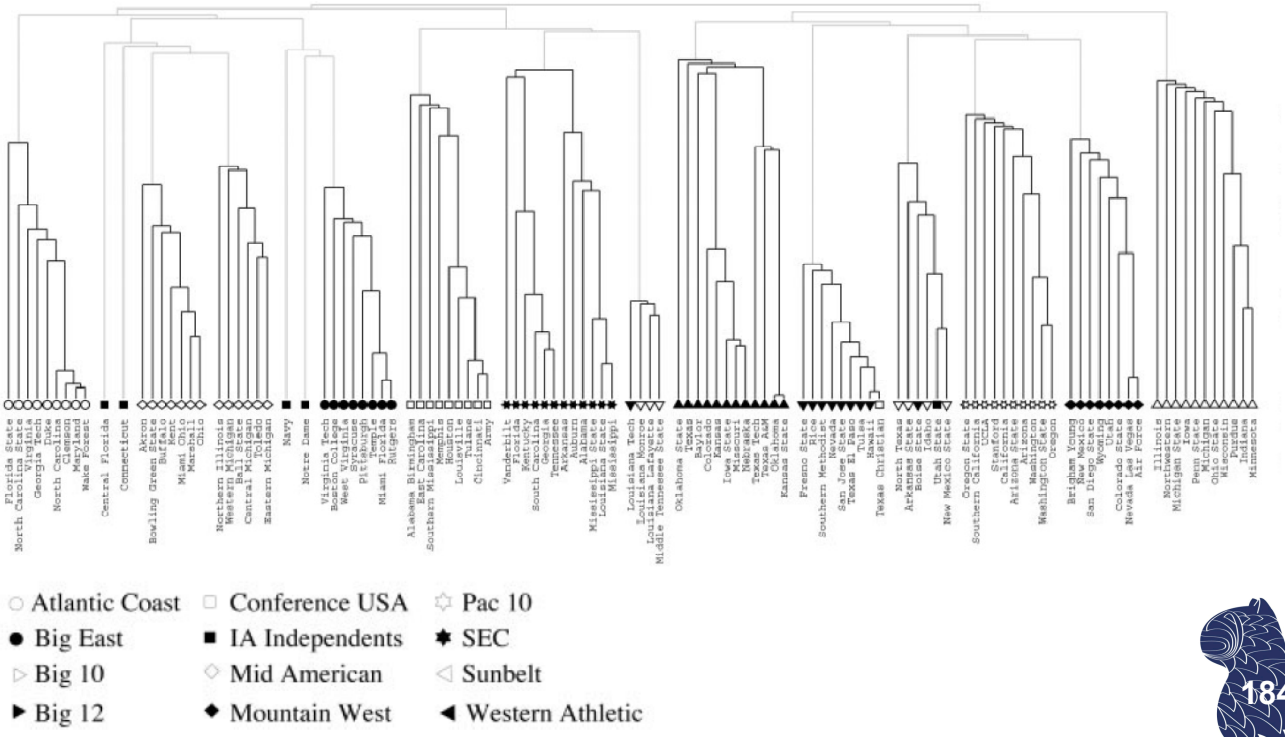
Evaluación de resultados

Red de fútbol americano (115 nodos, 613 enlaces)



Evaluación de resultados

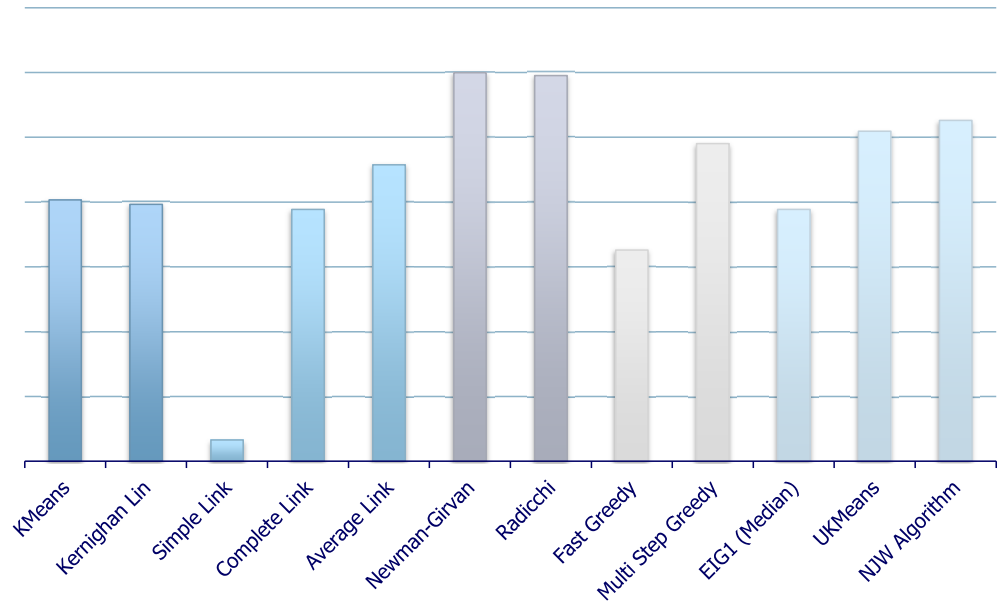
Red de fútbol americano (115 nodos, 613 enlaces)



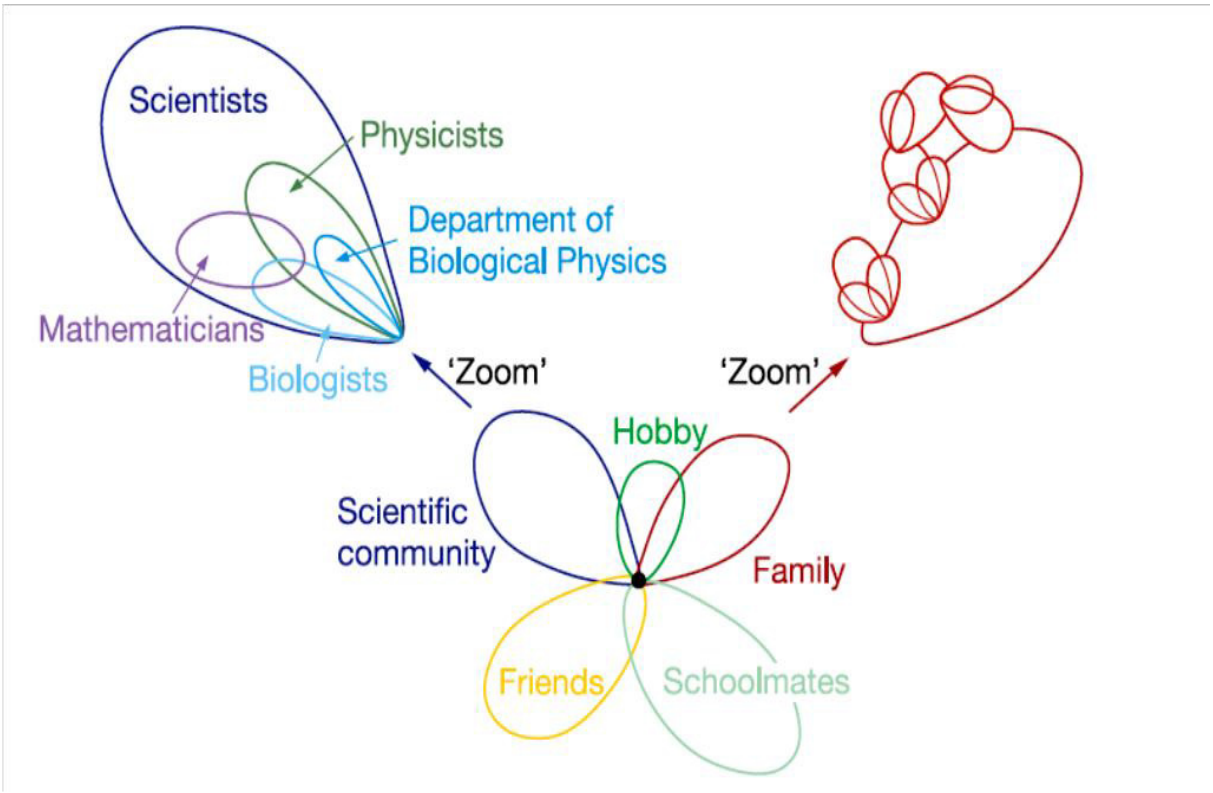
Evaluación de resultados

Red fútbol americano (115 nodos, 613 enlaces)

Modularidad

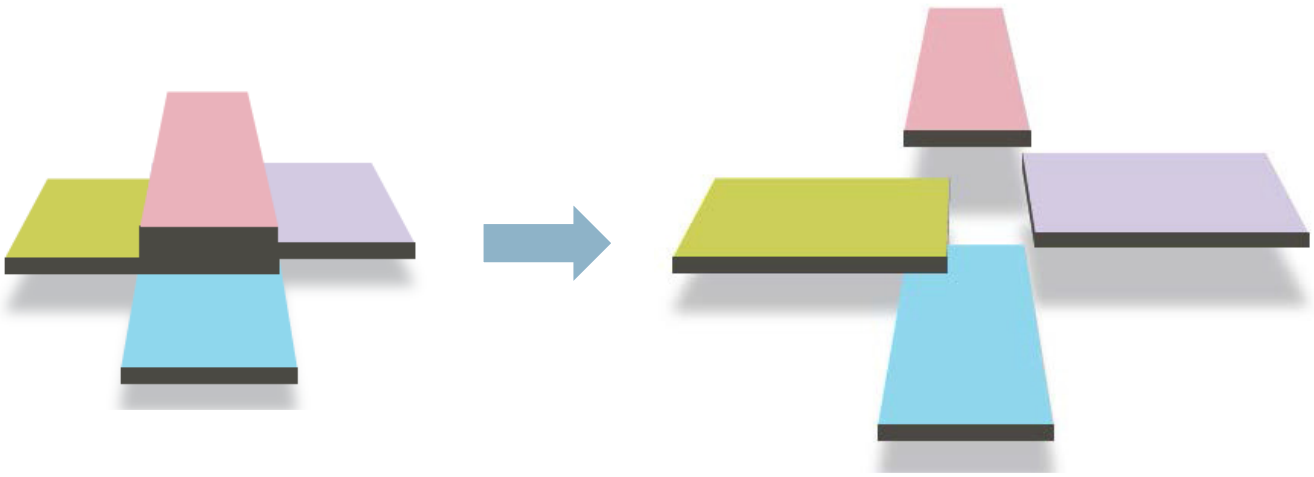


Comunidades solapadas



Comunidades solapadas

Comunidades en una red real

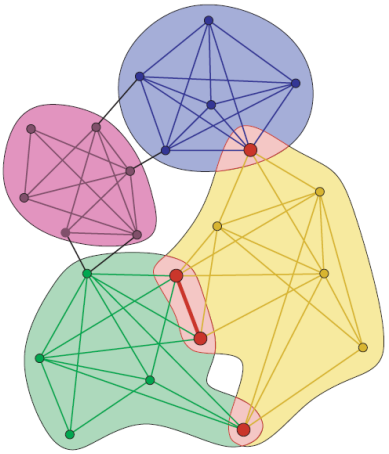


Leskovec, Rajamaran & Ullman:
"Mining of Massive Datasets"
Stanford University

CPM

Clique Percolation Method

[Palla et al., Nature'2005]

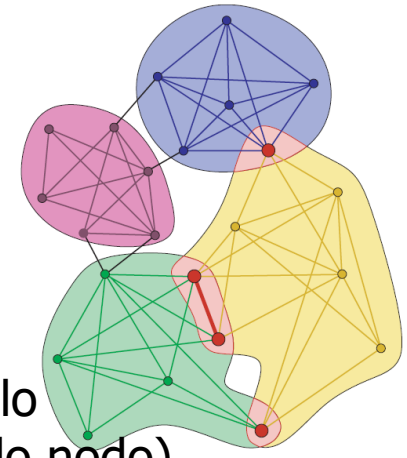


- Si de un k-clique eliminamos un enlace, se obtienen dos (k-1)-cliques solapados que comparten k-2 nodos.
- La unión de estos conjuntos de nodos solapados forma una cadena de cliques.
- IDEA (similar a Radicchi): Las aristas existentes dentro de una comunidad tienden a formar cliques; las aristas que conectan nodos de distintas comunidades, no.

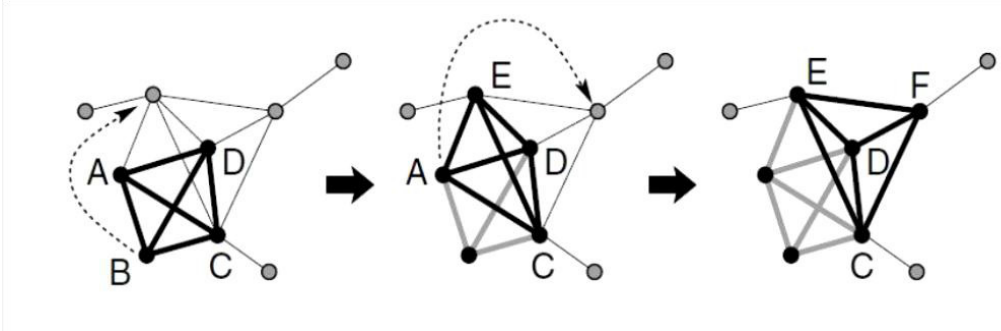
CPM

Clique Percolation Method

[Palla et al., Nature'2005]

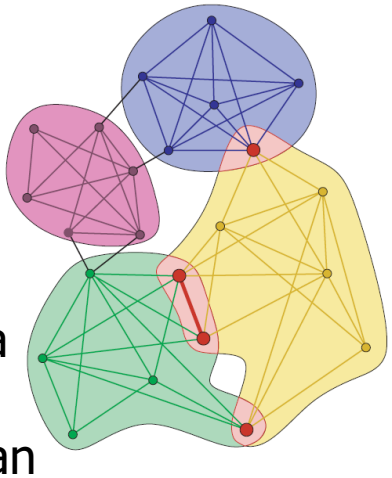


- ALGORITMO
  Encontrar cliques adyacentes
  para formar una cadena de cliques
  (es posible rotar/pivotar los k-cliques a lo largo de la cadena reemplazando un solo nodo).



CPM

Clique Percolation Method
[Palla et al., Nature'2005]

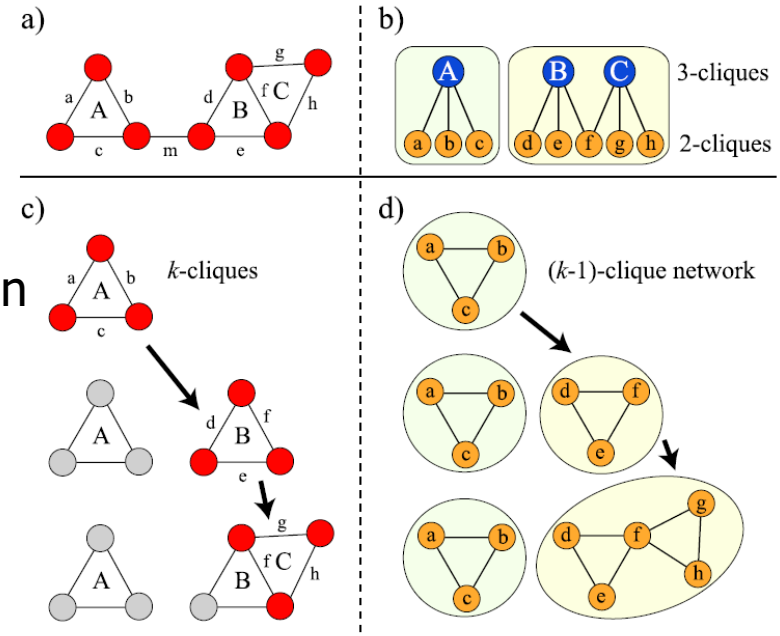


- IMPLEMENTACIÓN
  Matriz de adyacencia de k-cliques (número de nodos compartidos por cada par de cliques) filtrada (a 0 para valores <= k-1), a partir de la cual se determinan fácilmente las comunidades solapadas (conectividad).

- CPM es de orden exponencial (búsqueda de cliques), si bien CFinder (http://www.cfinder.org/) ofrece una versión aproximada más eficiente, $O(n_{cliques}^2)$

CPM Secuencial

[Kumpula et al., Physical Review E 2008]



- IDEA:
  Comprobar la formación de k-cliques conforme se añaden aristas.

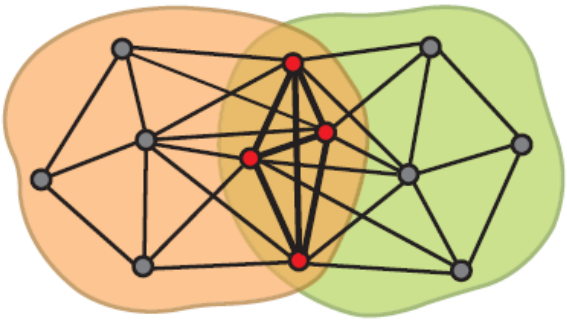
- Algoritmo escalable, prácticamente lineal.

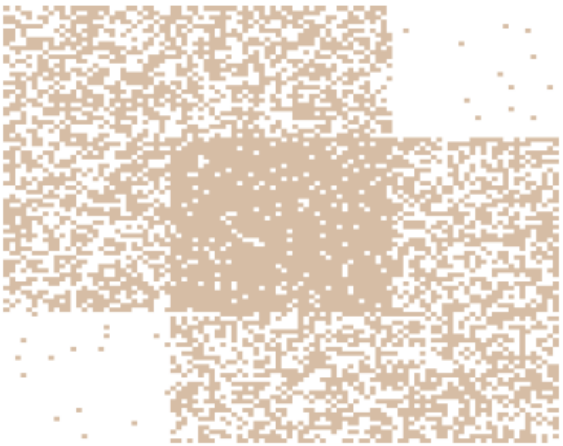
Alternativas

- CPMw para redes con pesos [Farkas, NJP'2007]
- "Maximal cliques" como núcleos de comunidades que luego se fusionan. [Shang et al., CPL'2010]
- MOSES basado en modelos estadísticos [McDaid & Hurley, ASONAM'2010]
- Algoritmo de fuzzy clustering basado en una relación difusa [Sun et al., Information Sciences 2011]
- CONA en dos etapas: primero se buscan comunidades no solapadas y luego se buscan vínculos entre ellas [Wu et al., Physica A 2012]

BigCLAM

IDEA:
La densidad de las aristas en las zonas solapadas es mayor…



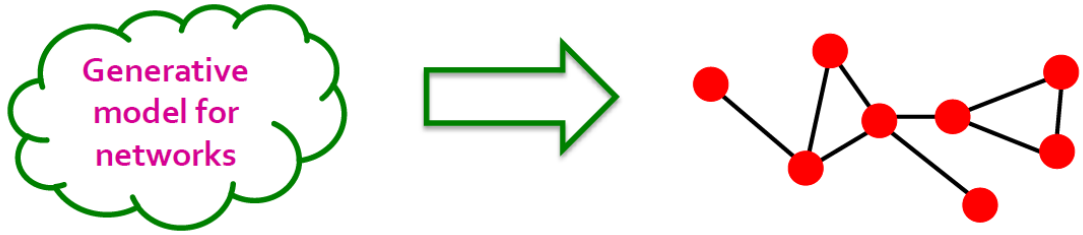


Yang & Leskovec: "Overlapping Community Detection at Scale: A Nonnegative Matrix Factorization Approach". ACM International Conference on Web Search and Data Mining (WSDM), 2013.

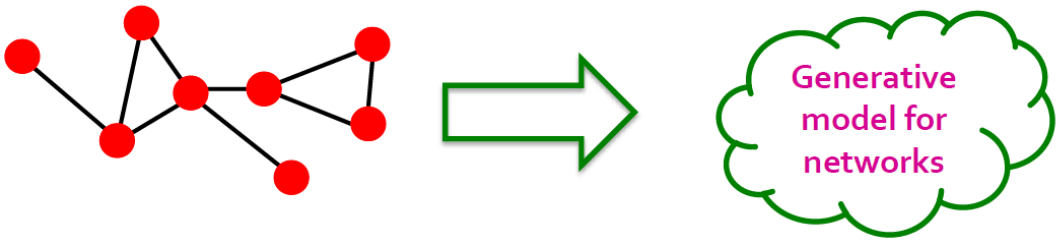
BigCLAM

PLAN

- Dado un modelo, podemos generar una red

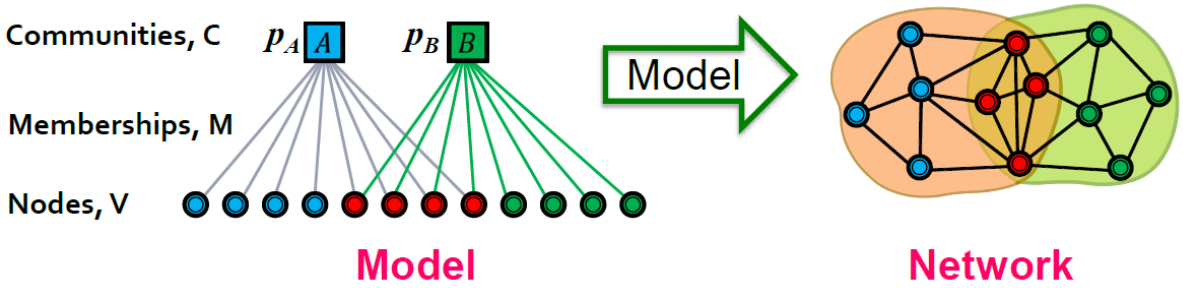


- Dada una red, podemos encontrar el "mejor" modelo



BigCLAM

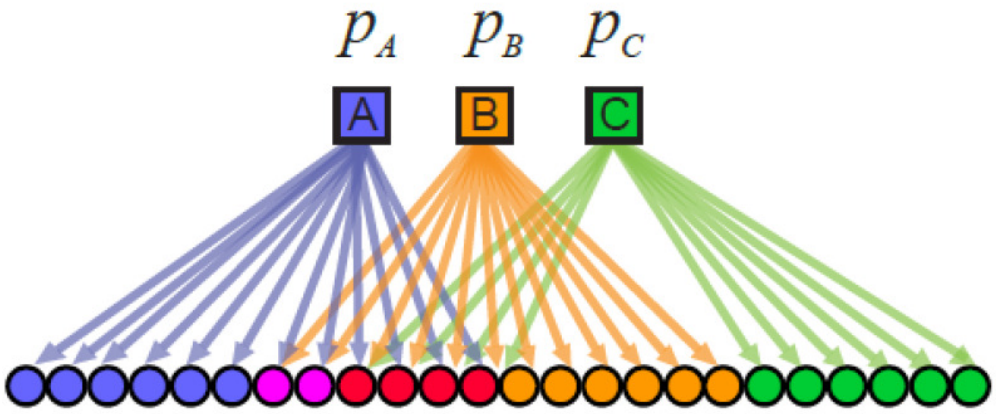
AGM [Affiliation Graph Model]



- Generación de enlaces: Para cada par de nodos de una misma comunidad A, creamos un enlace entre ellos con probabilidad p_A.
- Modelo cuyos parámetros estimaremos para detectar las comunidades.

BigCLAM

AGM [Affiliation Graph Model]

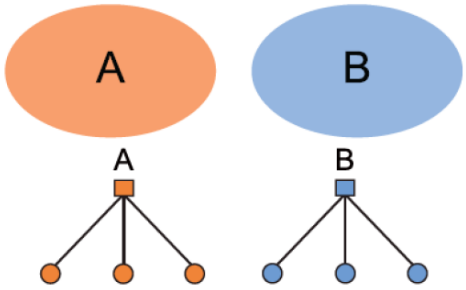


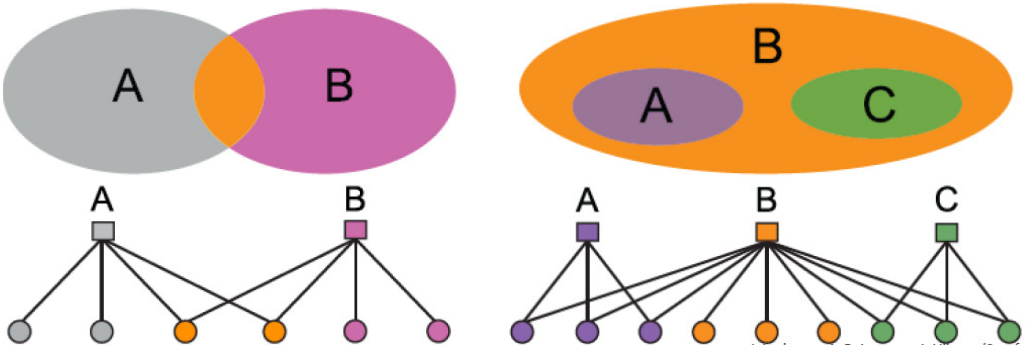
$$P(u,v) = 1 - \prod_{c \in M_u \cap M_v} (1 - p_c)$$

If $\boldsymbol{u}, \boldsymbol{v}$ share no communities: $\boldsymbol{P(u,v) = \varepsilon}$

BigCLAM

AGM [Affiliation Graph Model]

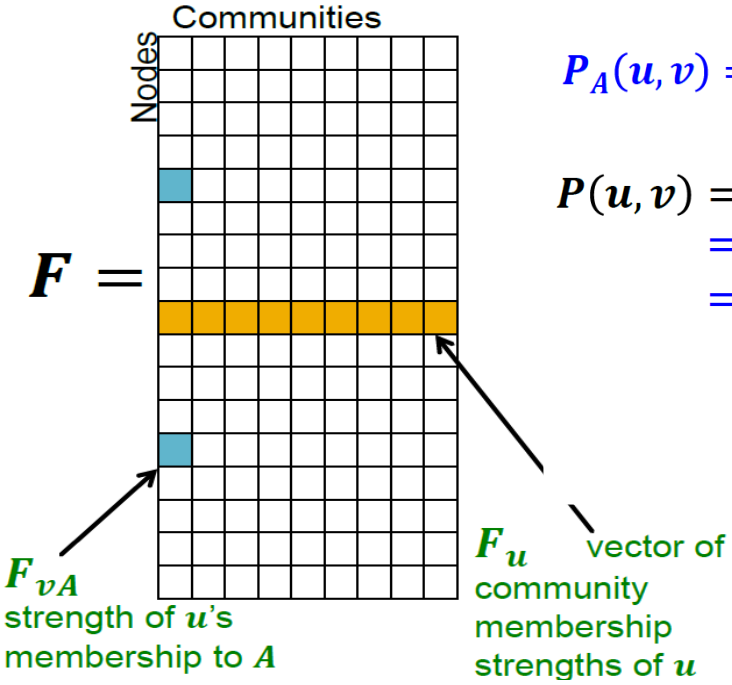




Modelo versátil:
Comunidades no solapadas, solapadas y anidadas

BigCLAM

- Parámetros del modelo F_{nA}: Fuerza con la que un nodo n pertenece a una comunidad A.



$$P_A(u,v) = 1 - \exp(-F_{uA} \cdot F_{vA})$$

$$P(u,v) = 1 - \prod_C (1 - P_C(u,v)) = 1 - \exp(-\sum_C F_{uC} \cdot F_{vC}) = 1 - \exp(-F_u \cdot F_v^T)$$

BigCLAM

PROBLEMA

Dada una red, estimar F

- Maximizar likelihood P(G|F)

$$arg\ max_F \prod_{(u,v)\in E} p(u,v) \prod_{(u,v)\notin E} (1 - p(u,v))$$

- Log-likelihood l(F) = log P(G|F)

$$l(F) = \sum_{(u,v)\in E} \log(1 - \exp(-F_u F_v^T)) - \sum_{(u,v)\notin E} F_u F_v^T$$

BigCLAM 1.0

ALGORITMO

- Fila de F

$$l(F_u) = \sum_{v \in \mathcal{N}(u)} \log(1 - \exp(-F_u F_v^T)) - \sum_{v \notin \mathcal{N}(u)} F_u F_v^T$$

- Gradiente de una fila de F

$$\nabla l(F_u) = \sum_{v \in \mathcal{N}(u)} F_v \frac{\exp(-F_u F_v^T)}{1 - \exp(-F_u F_v^T)} - \sum_{v \notin \mathcal{N}(u)} F_v$$

- Maximización: **Gradiente ascendente coordinado**
  Algoritmo iterativo

$$\boldsymbol{F_u \leftarrow F_u + \eta \nabla l(F_u)}$$

BigCLAM 2.0

- Problema:
  Calcular el gradiente $\nabla l(F_u)$ requiere tiempo lineal con respecto al tamaño de la red.

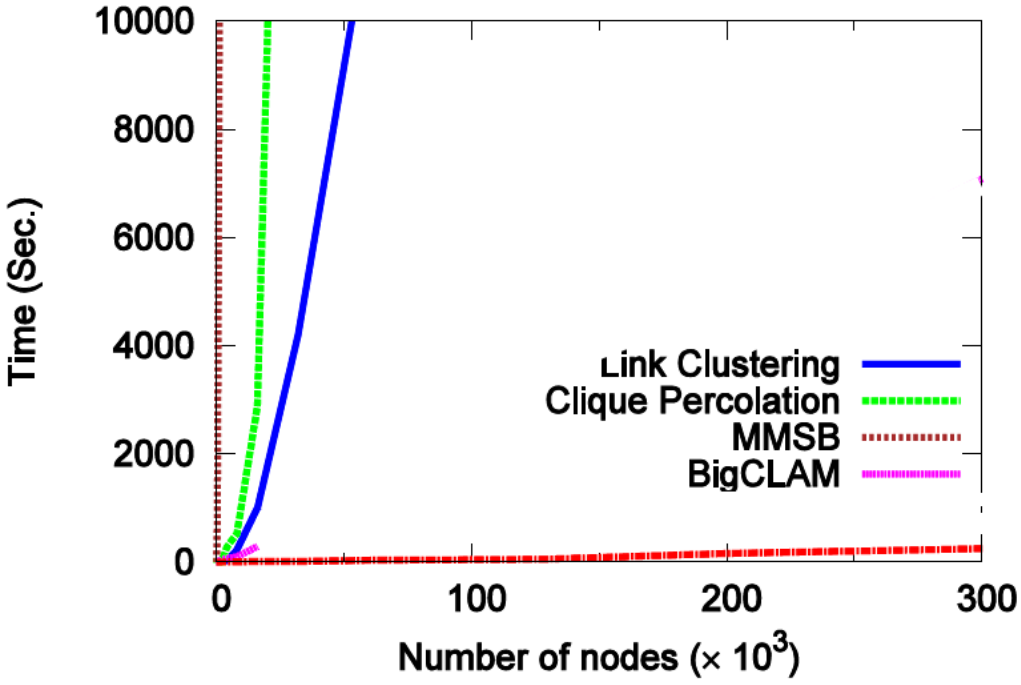
- Solución:

$$\sum_{v \notin \mathcal{N}(u)} F_v = (\sum_{v} F_v - F_u - \sum_{v \in \mathcal{N}(u)} F_v)$$

  Podemos precalcular $\Sigma_v F_v$ para obtener en tiempo lineal con respecto al grado de los nodos |N(u)|

BigCLAM

Resultado: Algoritmo escalable



Aplicaciones

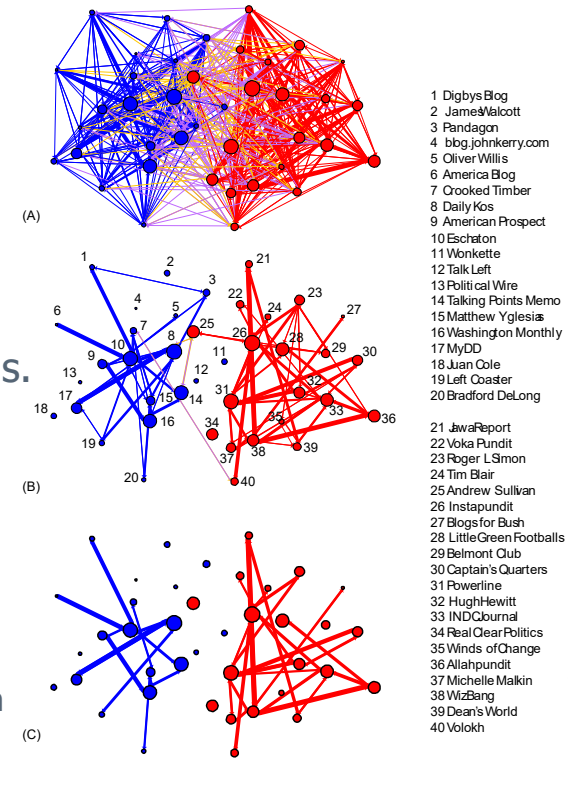
Blogs políticos

A) All citations between blogs.

B) Blogs with at least 5 citations in both directions.

C) Edges further limited to those exceeding 25 combined citations.

only 15% of the citations bridge communities

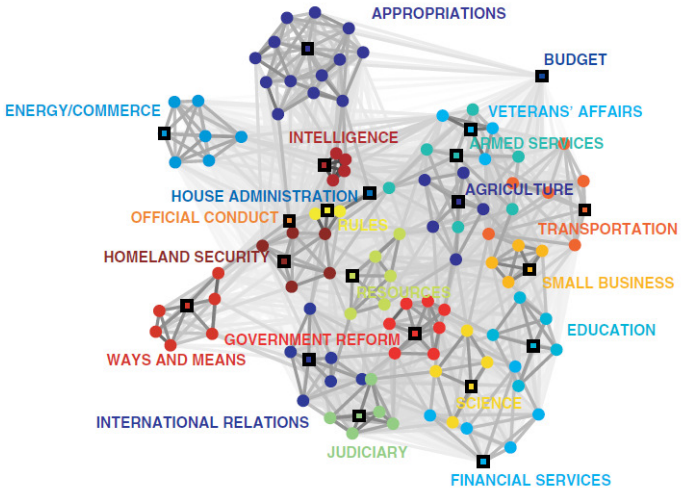


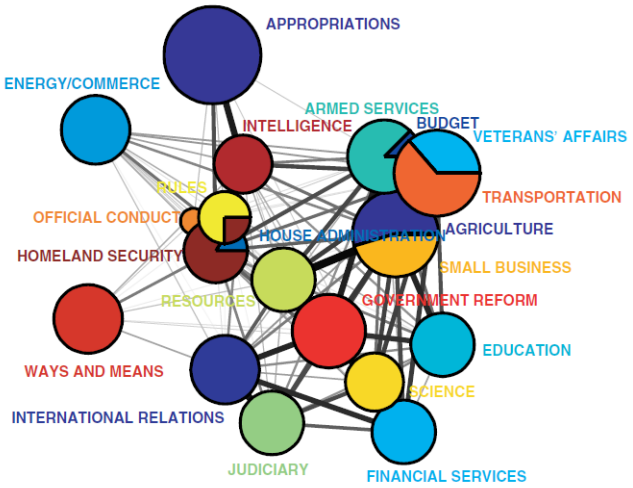
L. A. Adamic & N. Glance: **The political blogosphere and the 2004 US Election** LinkKDD'2005

Aplicaciones

Comités y subcomités

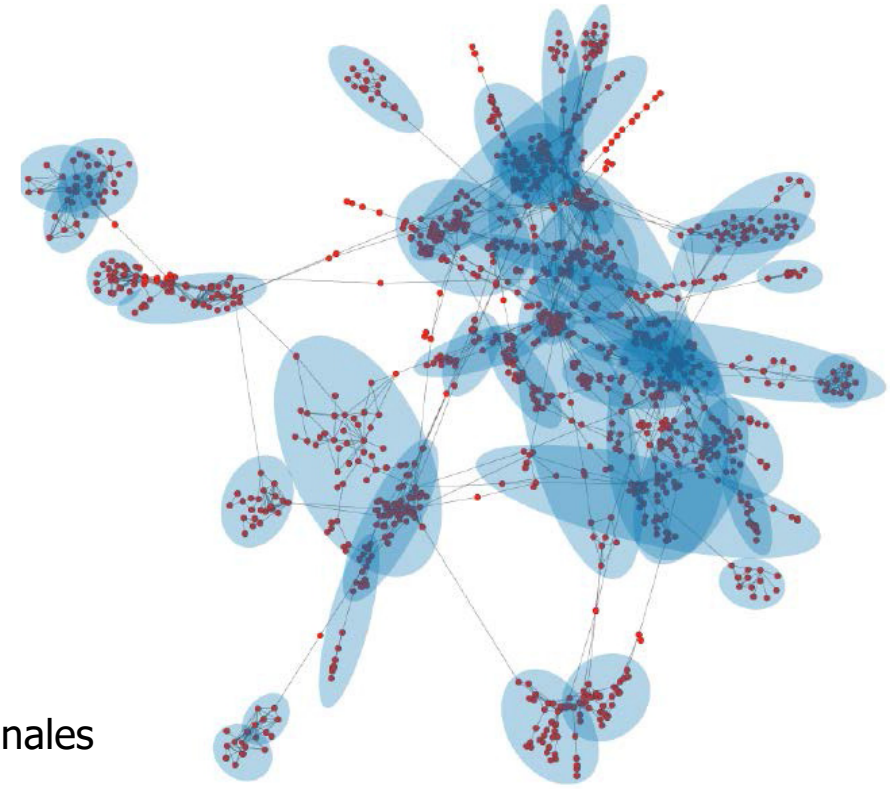
U.S. House of Representatives 2003-2004





Aplicaciones

Redes de interacción de proteínas (PPI)



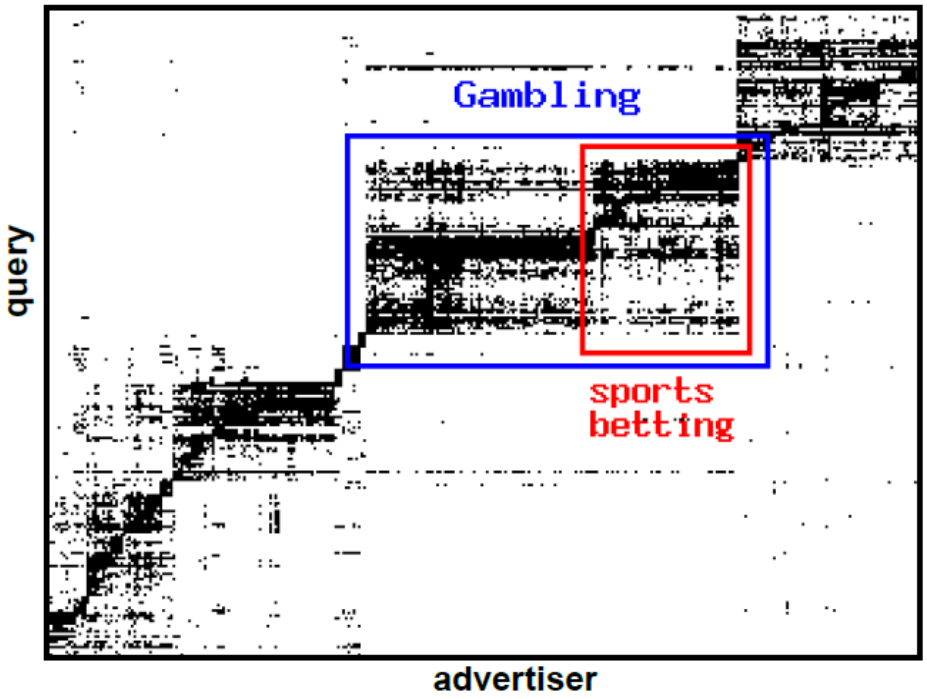
Nodos
Proteínas

Enlaces
Interacciones

Comunidades
Módulos funcionales

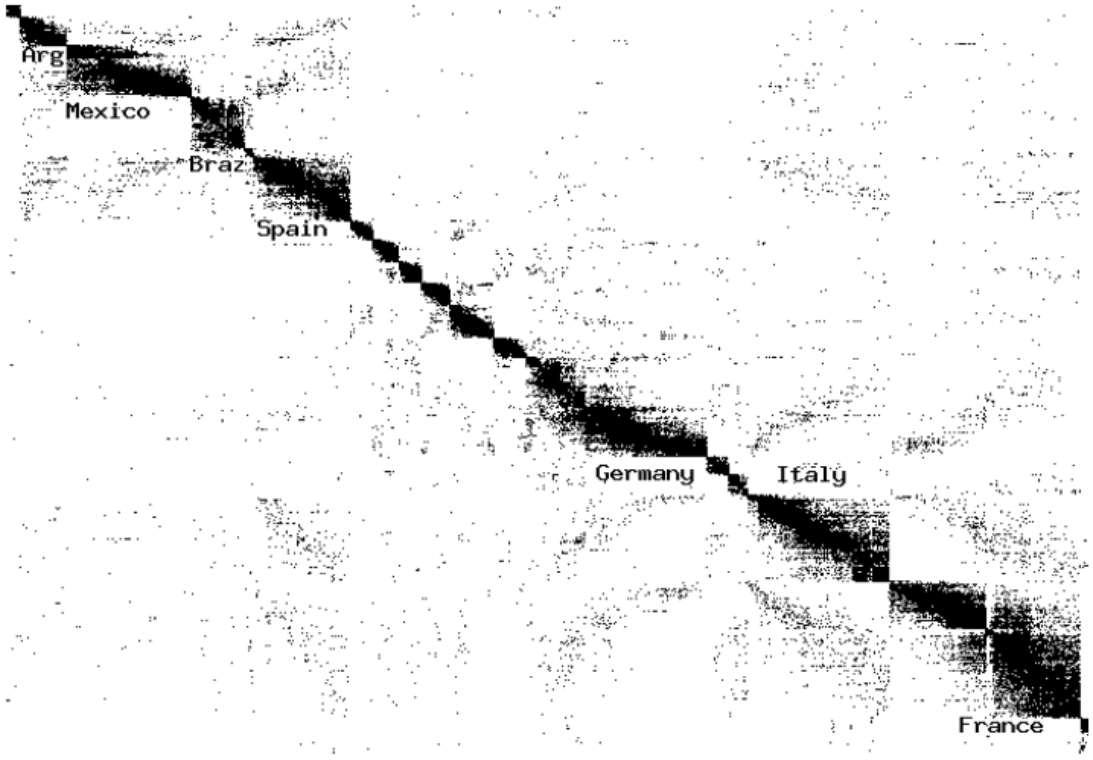
Aplicaciones

Query-advertiser graphs (micromarkets)



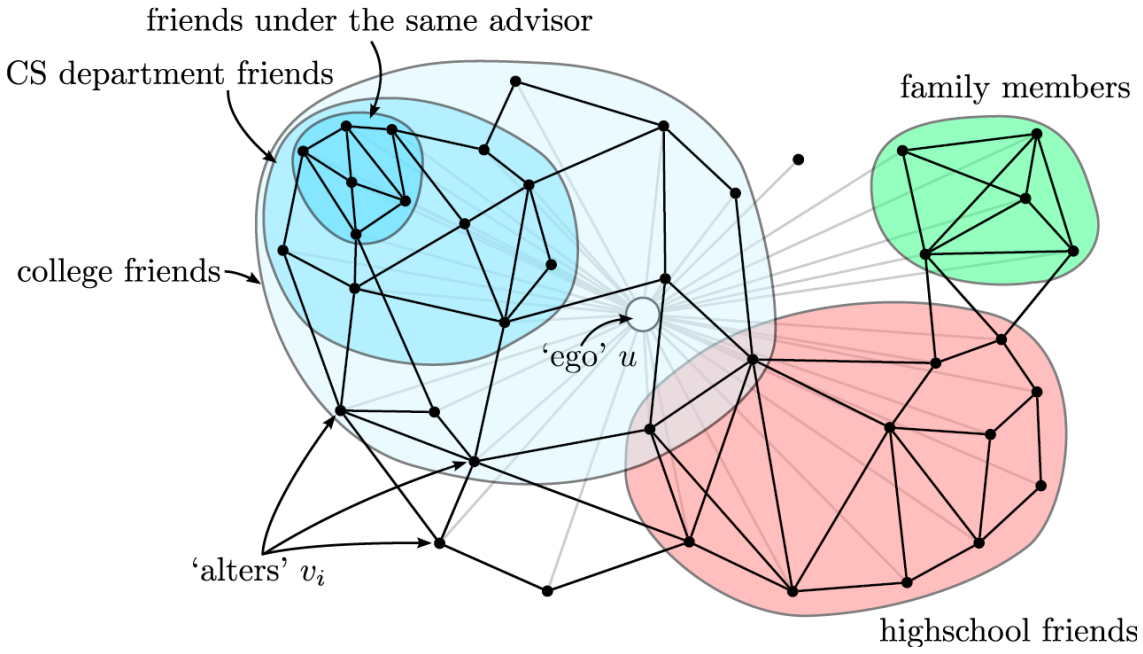
Aplicaciones

Movie-actor network



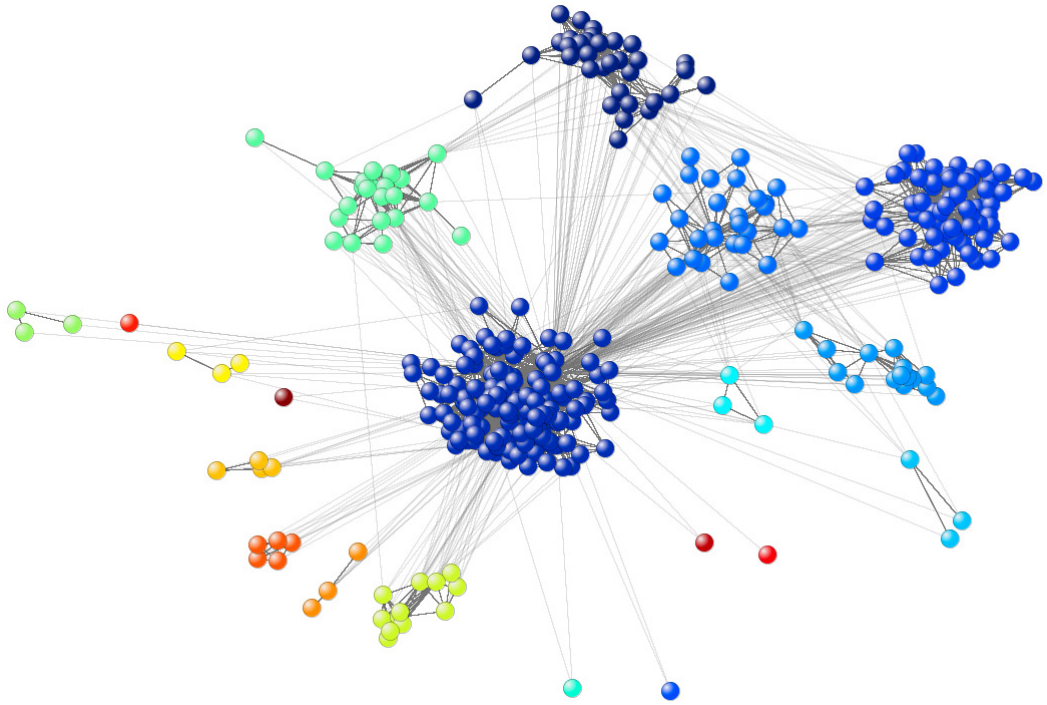
Aplicaciones

Círculos sociales



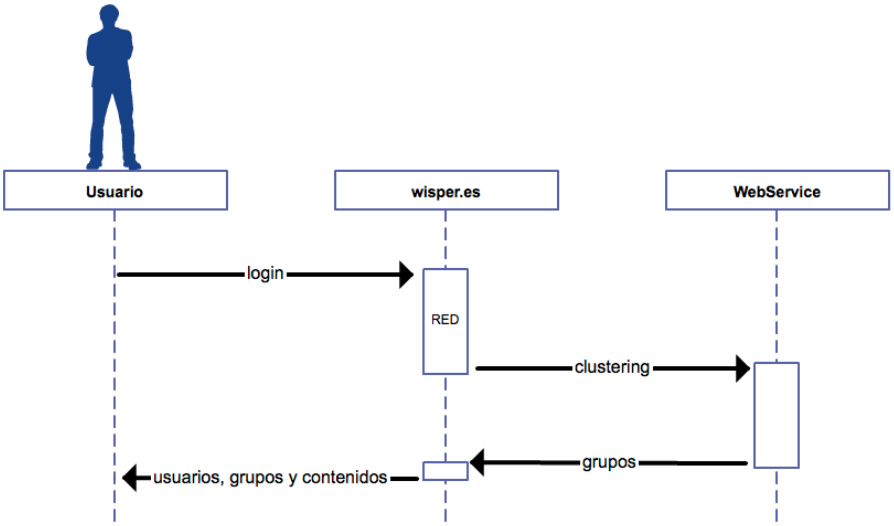
Aplicaciones

Comunidades en una red de amigos en Facebook



Aplicaciones

wisper.es

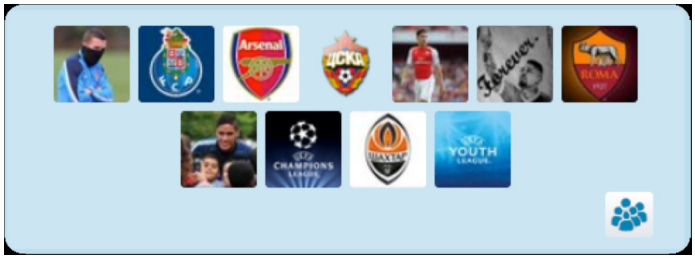


Aplicaciones

wisper.es

Aplicaciones

wisper.es



Agradecimientos

Julio Omar Palacio Niño

Detección de comunidades en redes:

Algoritmos y aplicaciones

MSc Thesis, September 2013

Department of Computer Science and Artificial Intelligence

University of Granada (Spain)

Aarón Rosas Rodríguez & Francisco Javier Gijón Moleón

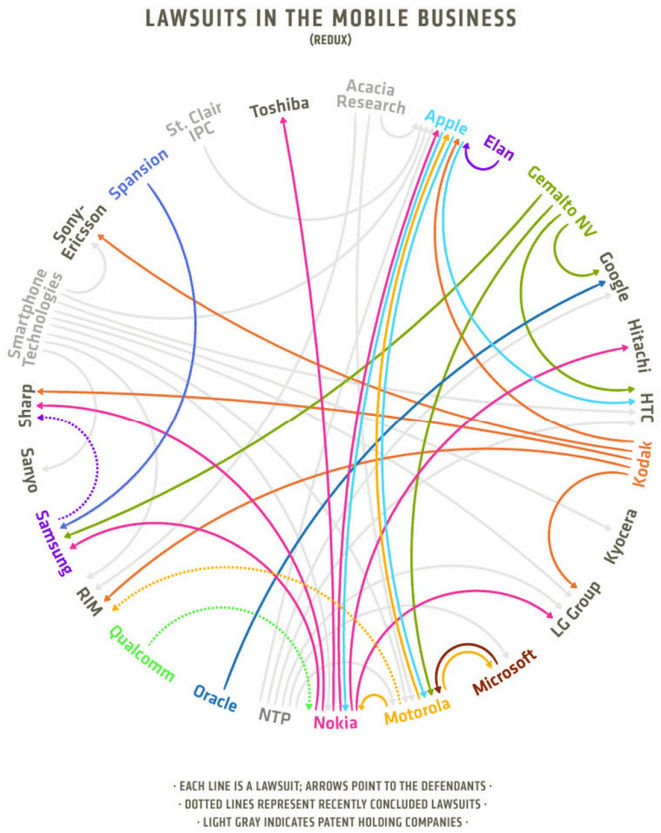
Algoritmos paralelos

para la detección de comunidades en redes

Proyecto de Fin de Carrera, septiembre de 2014

ETSIIT, Universidad de Granada

Predicción de enlaces



Predicción de enlaces

El problema

Dada una instantánea de una red
en el instante de tiempo t, G(t) = (V(t), E(t)),
¿cuál será el conjunto de enlaces
que se formará en el instante t+Δ?

$E(t) \rightarrow E(t+\Delta)$?

Predicción de enlaces

Aplicaciones

- Sistemas de recomendación
  - "Collaborative filtering" (vs. content-based filtering)
  - Redes sociales
- Integración de datos
  - Resolución de entidades (a.k.a. record linkage)
- Bioinformática
  - Predicción de interacciones entre proteínas

Evaluación: Métricas

Matriz de confusión
(confusion matrix)

		Predicción	
		C_P	C_N
Clase real	C_P	TP: True positive	FN: False negative
	C_N	FP: False positive	TN: True negative

Precisión del clasificador

accuracy = (TP+TN)/(TP+TN+FP+FN)

Evaluación: Métricas

Medidas "cost-sensitive"

		Predicción	
		C_P	C_N
Clase real	C_P	TP: True positive	FN: False negative
	C_N	FP: False positive	TN: True negative

precision = TP/(TP+FP)

True positive recognition rate

recall = sensitivity = hit-rate = TP/P = TP/(TP+FN)

True negative recognition rate

specificity = TN/N = TN/(TN+FP)

Evaluación: Métricas

Medidas "cost-sensitive"

		Predicción	
		C_P	C_N
Clase real	C_P	TP: True positive	FN: False negative
	C_N	FP: False positive	TN: True negative

F-score

Media armónica de precision y recall:

F = 2*precision*recall / (precision+recall)

F = 2TP / (2TP+FP+FN)

Evaluación: Métricas

Medidas "cost-sensitive"

		Predicción	
		C_P	C_N
Clase real	C_P	TP: True positive	FN: False negative
	C_N	FP: False positive	TN: True negative

F-score (β)

Media armónica ponderada entre precision y recall:

$$F = (1+\beta^2)*precision*recall / (\beta^2*precision+recall)$$

Evaluación: Métricas

Medidas "cost-sensitive"

		Predicción	
		C_P	C_N
Real	C_P	**TP**	FN
	C_N	FP	**TN**

Accuracy

		Predicción	
		C_P	C_N
Real	C_P	**TP**	**FN**
	C_N	FP	TN

Recall

		Predicción	
		C_P	C_N
Real	C_P	**TP**	FN
	C_N	**FP**	TN

Precision

		Predicción	
		C_P	C_N
Real	C_P	**TP**	**FN**
	C_N	**FP**	TN

F-measure

Evaluación: Métricas

En el caso de la predicción de enlaces…

Normalmente, sólo nos interesarán aquellos enlaces candidatos que es más probable que se formen [top k]:

- La precisión [precision] nos indica el porcentaje de acierto dentro de los k enlaces más probables:

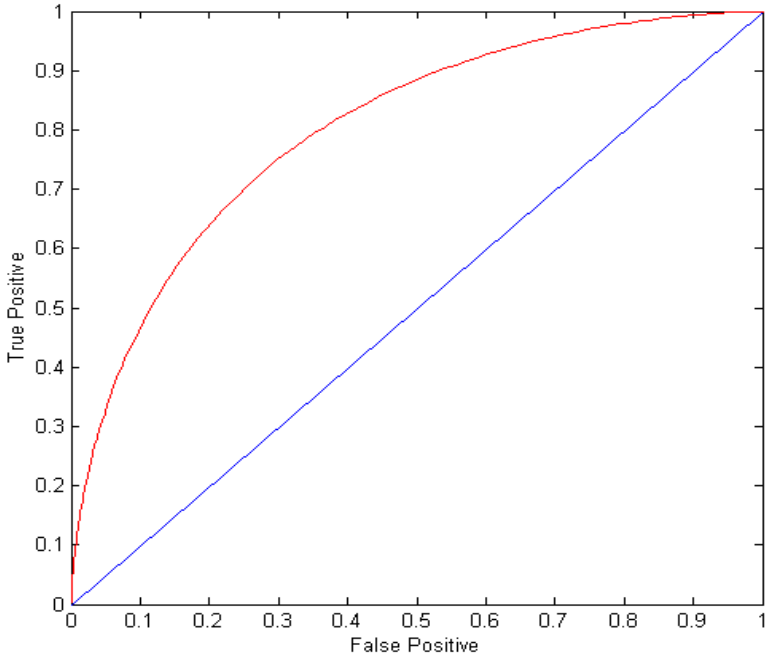
$$\textbf{precision(k) = TP(k) / k}$$

- Accuracy, recall (sensitivity), specificity y F-score **no** aportan información adicional en este contexto.

Evaluación: Comparación

Curvas ROC

Receiver Operating Characteristics



TPR = TP/(TP+FN) Eje vertical: "true positive rate"

FPR = FP/(FP+TN) Eje horizontal: "false positive rate"

Evaluación: Comparación

Curvas ROC

- Desarrolladas en los años 50 para analizar señales con ruido: caracterizar el compromiso entre aciertos y falsas alarmas.
- Permiten comparar visualmente distintos modelos de clasificación.
- **AUC**: El área que queda bajo la curva es una medida de la precisión [accuracy] del clasificador:
  - Cuanto más cerca estemos de la diagonal (área cercana a 0.5), menos preciso será el modelo.
  - Un modelo "perfecto" tendrá área 1.

Evaluación: Comparación

Curvas ROC

Cálculo del área bajo la curva AUC

Se puede aproximar muestreando pares de enlaces del conjunto de validación y enlaces no existentes:

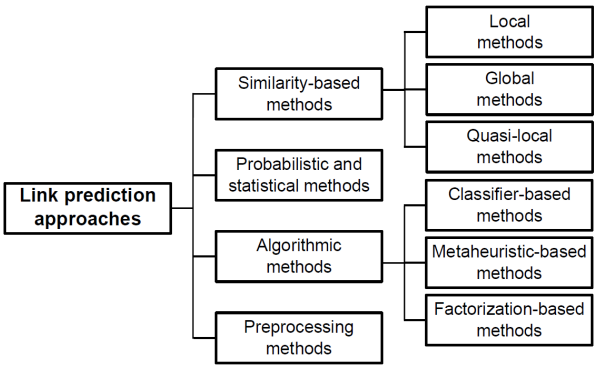
$$AUC = (n' + 0.5n'') / n$$

donde **n** es el número pares muestreados, **n'** es el número de pares en los que enlace del conjunto de validación recibió una probabilidad de existencia mayor que el enlace no existente y **n''** es el número de enlaces en los que hubo un empate.

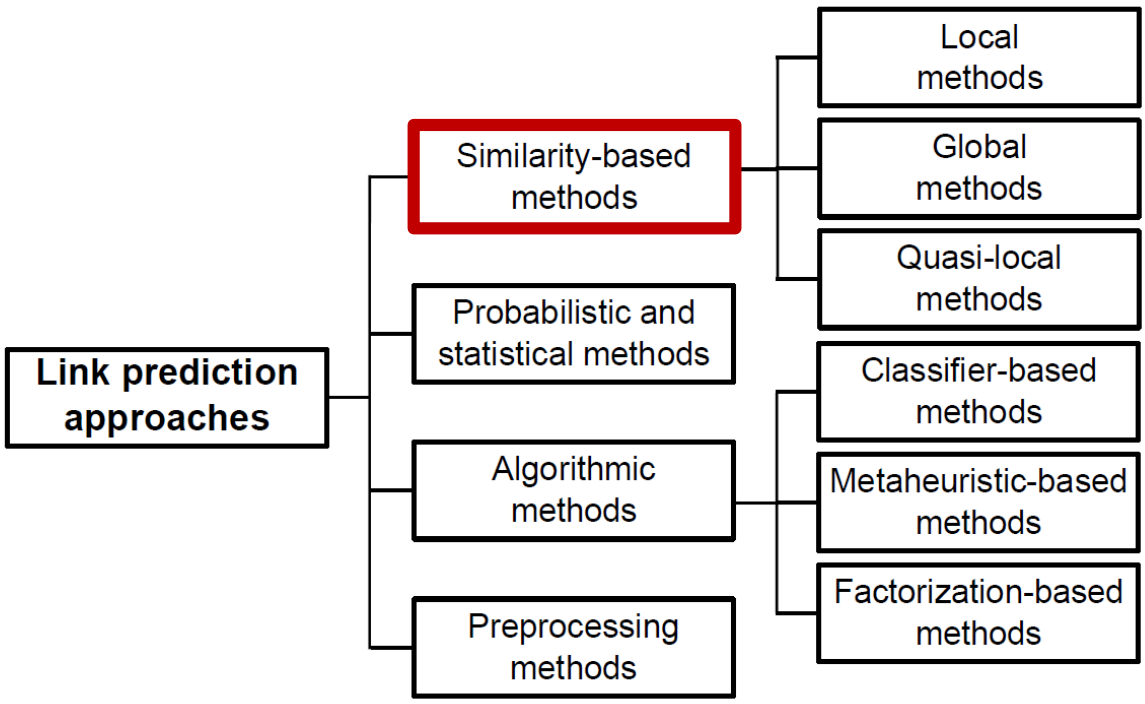
Predicción de enlaces

Métodos de predicción de enlaces

- Métodos basados en similitud
  - Métodos locales
  - Métodos globales
  - Métodos cuasi-locales
- Métodos probabilísticos
- Métodos algorítmicos
  - Métodos basados en clasificadores
  - Métodos basados en metaheurísticas
  - Métodos basados en factorizaciones



Métodos basados en similitud



Métodos basados en similitud

Hipótesis

Los nodos de una red tienden a formar enlaces con otros nodos similares

Idea básica

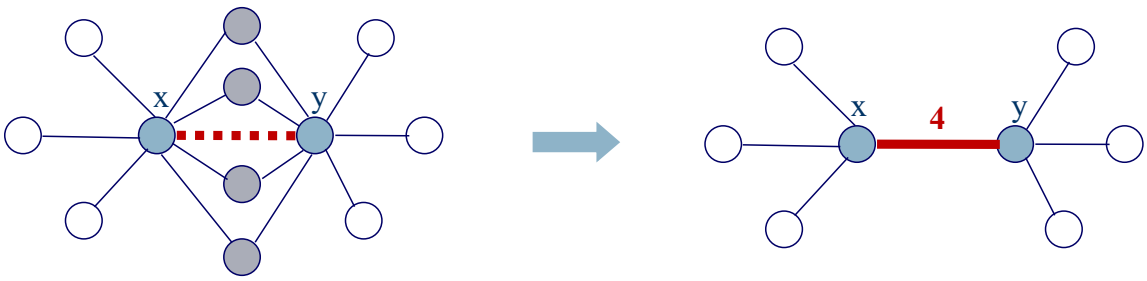
Si definimos una función de similitud s(x,y) entre parejas de nodos, podemos utilizar dicha función para establecer un ranking que nos indique qué enlaces es más probable que se formen en el futuro.

Métodos basados en similitud

Métodos locales

Dos nodos se consideran similares

si tienen vecinos compartidos…



Métodos basados en similitud

Métodos locales (1/5)

- **CN** [Common Neighbors]

$$s(x,y) = |\Gamma_x \cap \Gamma_y|$$

- **AA** [Adamic-Adar index]

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{\log |\Gamma_z|}$$

- **RA** [Resource Allocation index]

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{|\Gamma_z|}$$

Métodos basados en similitud

Métodos locales (2/5)

- **RA-CNI** [Resource Allocation index based on Common Neighbor Interaction]

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{|\Gamma_z|} + \sum_{e_{i,j} \in E, |\Gamma_i| < |\Gamma_j|, i \in \Gamma_x, j \in \Gamma_y} \left(\frac{1}{|\Gamma_i|} - \frac{1}{|\Gamma_j|} \right)$$

- **PA** [Preferential Attachment index]: Modelo de Barabasi- Albert

$$s(x,y) = |\Gamma_x||\Gamma_y|$$

- **J** [Jaccard index]

$$s(x,y) = \frac{|\Gamma_x \cap \Gamma_y|}{|\Gamma_x \cup \Gamma_y|}$$

Métodos basados en similitud

Métodos locales (3/5)

- **SA** [Salton index] = Cosine similarity

$$s(x,y) = \frac{|\Gamma_x \cap \Gamma_y|}{\sqrt{|\Gamma_x||\Gamma_y|}}$$

- **SO** [Sorensen index]

$$s(x,y) = \frac{2|\Gamma_x \cap \Gamma_y|}{|\Gamma_x| + |\Gamma_y|}$$

- **LLHN** [Local Leicht-Holme-Newman index]

$$s(x,y) = \frac{|\Gamma_x \cap \Gamma_y|}{|\Gamma_x||\Gamma_y|}$$

Métodos basados en similitud

Métodos locales (4/5)

- **HPI** [Hub-Promoted Index] $$s(x,y) = \frac{|\Gamma_x \cap \Gamma_y|}{\min(|\Gamma_x|, |\Gamma_y|)}$$

- **HDI** [Hub-Depressed Index] $$s(x,y) = \frac{|\Gamma_x \cap \Gamma_y|}{\max(|\Gamma_x|, |\Gamma_y|)}$$

- **IA** [Individual Attraction index]

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{|e_{z,\Gamma_x \cap \Gamma_y}| + 2}{|\Gamma_z|}$$

- **SIA** [Simple IA]

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{|e_{\Gamma_x \cap \Gamma_y}| + 2}{|\Gamma_z||\Gamma_x \cap \Gamma_y|}$$

Métodos basados en similitud

Métodos locales (5/5)

- **MI** [Mutual Information]

$$s(x,y) = -I(e_{x,y}|\Gamma_x \cap \Gamma_y) = -I(e_{x,y}) + \sum_{z \in \Gamma_x \cap \Gamma_y} I(e_{x,y}; z)$$

- **LNB** [Local Naive Bayes]

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} f(z) \log (oR_z)$$

- **CAR** [CAR-based indices]: Local communities

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} 1 + \frac{|\Gamma_x \cap \Gamma_y \cap \Gamma_z|}{2}$$

Métodos basados en similitud

Métodos locales

VENTAJAS

- Eficientes
- Paralelizables

DESVENTAJAS

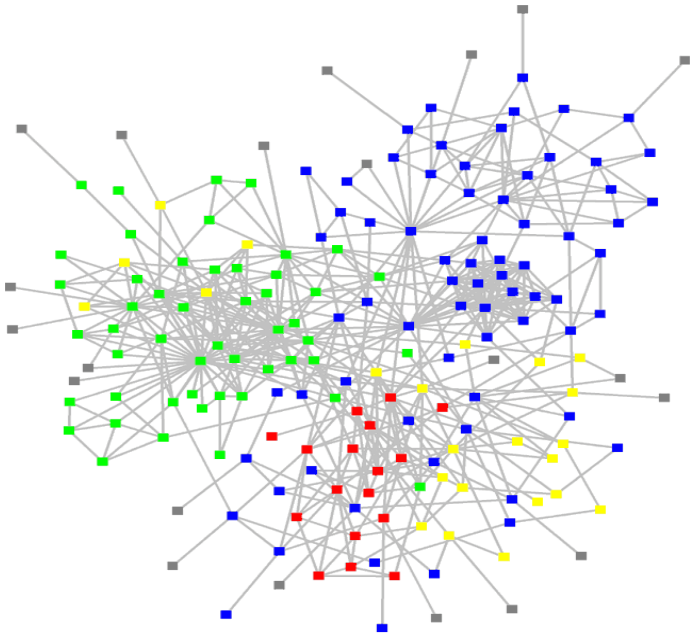
- Sólo consideran información local
  (de hecho, sólo se calcula la similitud entre pares de nodos con vecinos compartidos, i.e. a distancia 2)

  p.ej. Redes de mundo pequeño [small-world networks]

Métodos basados en similitud

Métodos globales

Se utiliza toda la información de la topología de la red



Métodos basados en similitud

Métodos globales: Caminos en la red

- **NSP** [Negated Shortest Path]

$$s(x,y) = -|shortest\ path_{x,y}|$$

- **KI** [Katz Index]

$$s(x,y) = \sum_{l=1}^{\infty} \beta^l |paths^l_{x,y}| = \sum_{l=1}^{\infty} \beta^l (A^l)_{x,y}$$

- **GLHN** [Global Leicht-Holme-Newman index]

$$S = I + \sum_{l=1}^{\infty} \phi^l A^l$$

Métodos basados en similitud

Métodos globales: Caminos aleatorios

- **RW** [Random Walks]

$$\overrightarrow{p^x}(t) = M^T \overrightarrow{p^x}(t-1)$$

- **RWR** [Random Walks with Restart]

$$\overrightarrow{p^x}(t) = \alpha M^T \overrightarrow{p^x}(t-1) + (1-\alpha)s^x$$

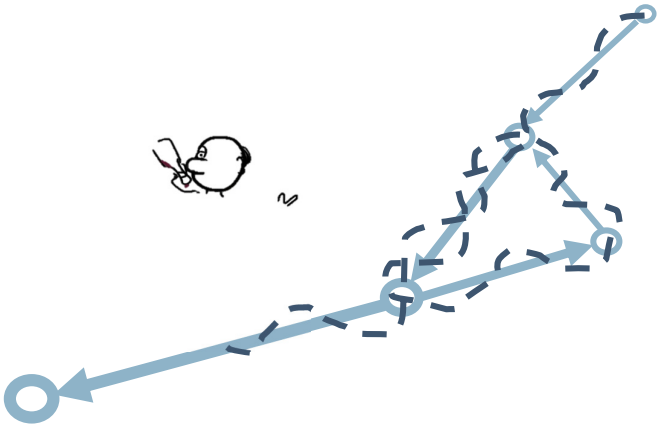
- **FP** [Flow Propagation]: Usando la matriz Laplaciana en vez de la matriz de adyacencia (L=D-A).
- **MERW** [Maximal Entropy Random Walk], teniendo en cuenta la tendencia a conectarse con nodos centrales.

Métodos basados en similitud

Métodos globales: Caminos aleatorios

Relación con PageRank

Google



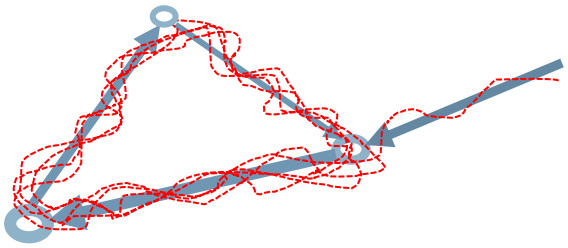
Lada Adamic, "Social Network Analysis"
https://www.coursera.org/course/sna

El PageRank de Google mide la importancia de un nodo en la red en proporción a la fracción de tiempo que un caminante aleatorio pasaría en él.

Métodos basados en similitud

Métodos globales: Caminos aleatorios

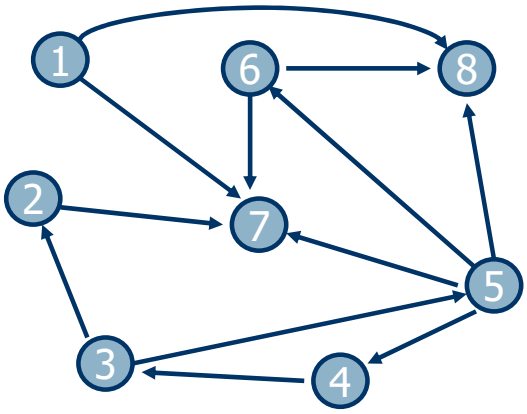
Relación con PageRank

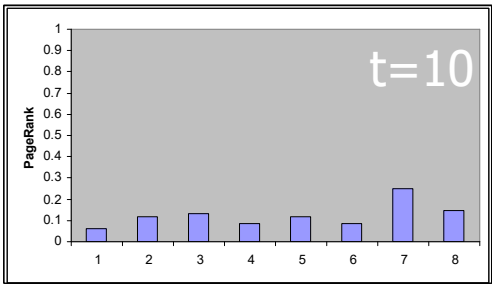


Problema: Atrapado en la red

Solución: Teletransporte

Salto aleatorio con una probabilidad dada.





Métodos basados en similitud

Métodos globales

- **SimRank** (cómo de pronto se encontrarán dos caminantes que empiezan en nodos diferentes y siguen un camino aleatorio).

- **PLM** [Pseudoinverse of the Laplacian Matrix]

$$s(x,y) = \frac{L^+_{x,y}}{\sqrt{L^+_{x,x} L^+_{y,y}}}$$

Métodos basados en similitud

Métodos globales

- **ACT** [Average Commute Time]: Número medio de pasos que hay que dar para llegar de x a y.

$$n(x,y) = |E|(L^+_{x,x} + L^+_{y,y} - 2L^+_{x,y})$$

$$s(x,y) = \frac{1}{L^+_{x,x} + L^+_{y,y} - 2L^+_{x,y}}$$

- **RFK** [Random Forest Kernel]

$$S = (I + L)^{-1}$$

- **BI** [Blondel Index]

$$S(t) = \frac{AS(t-1)A^T + A^T S(t-1)A}{\|AS(t-1)A^T + A^T S(t-1)A\|_F}$$

Métodos basados en similitud

Métodos globales

VENTAJAS

- Utilizan toda la información topológica de la red
- No están limitados a medir similitudes entre nodos que tengan vecinos compartidos (i.e. a distancia 2)

DESVENTAJAS

- Complejidad computacional.
- Paralelización compleja.

Métodos basados en similitud

Métodos cuasi-locales

Balance entre medidas locales y globales

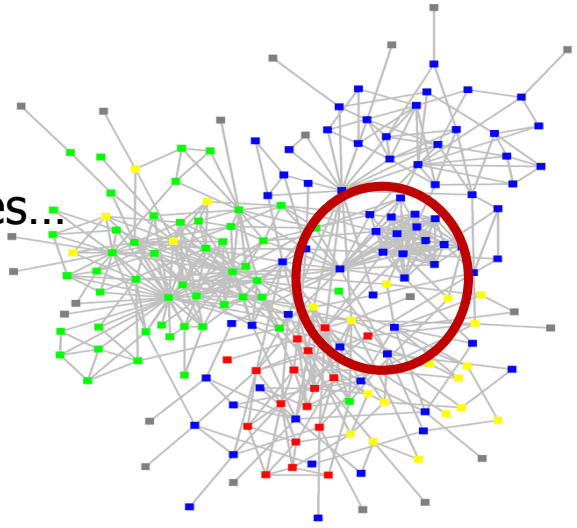
- Casi tan eficientes como los métodos locales.
- Consideran más información topológica que los métodos locales…

Métodos basados en similitud

Métodos cuasi-locales

Equilibro intermedio entre métodos locales y globales

- Casi tan eficientes como los métodos locales.
- Consideran más información topológica que los métodos locales…



Métodos basados en similitud

Métodos cuasi-locales

- **LPI** [Local Path Index]: Extensión del índice de Katz

$$S = \sum_{i=2}^{l} \beta^{i-2} A^i$$

- **LRW** [Local Random Walks]

$$s^{x,y}(t) = \frac{|\Gamma_x|}{2|E|} \overrightarrow{p_y^x}(t) + \frac{|\Gamma_y|}{2|E|} \overrightarrow{p_x^y}(t)$$

- **SRW** [Superposed Random Walks]

$$s^{x,y}(t) = \sum_{i=1}^{t} \left(\frac{|\Gamma_x|}{2|E|} \overrightarrow{p_y^x}(i) + \frac{|\Gamma_y|}{2|E|} \overrightarrow{p_x^y}(i) \right)$$

Métodos basados en similitud

Métodos cuasi-locales

- **ORA-CNI** [3rd Order Resource Allocation based on Common Neighbor Interactions]

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{|\Gamma_z|} + \sum_{e_{i,j} \in E, |\Gamma_i| < |\Gamma_j|, i \in \Gamma_x, j \in \Gamma_y} \left(\frac{1}{|\Gamma_i|} - \frac{1}{|\Gamma_j|} \right) + \beta \sum_{[x,p,q,y] \in paths^3_{x,y}} \frac{1}{|\Gamma_p||\Gamma_q|}$$

- **FL** [Friend Link], similar a LPI

$$s(x,y) = \sum_{i=2}^{l} \frac{1}{i-1} \frac{(A^i)_{x,y}}{\prod_{j=2}^{i}(|V|-j)}$$

- **PFP** [PropFlow Predictor], similar a RWR

Métodos basados en similitud

Métodos cuasi-locales

PFP [PropFlow Predictor]

Input: Network $G = (V, E)$, node x and max path length l.
Output: Score $S_{x,y}$ for all $n \leq l$-degree neighbors of y from x.

$Found = \{x\}$;
$NewSearch = \{x\}$;
$S_{x,x}$ = 1;
for ***each*** z ***in*** $V - \{x\}$ **do**
    $S_{x,z}$ = 0;
end
for $CurrentDegree$ ***from*** 0 ***to*** l **do**
    $OldSearch$ = $NewSearch$;
    $NewSearch = \emptyset$;
    for ***each*** i ***in*** $OldSearch$ **do**
        for ***each*** j ***in*** Γ_i **do**
            $S_{x,j} \leftarrow S_{x,j} + \frac{S_{x,i}}{|\Gamma_i|}$;
            if j ***is not in*** $Found$ **then**
                $Found = Found \cup \{j\}$;
                $NewSearch = NewSearch \cup \{j\}$;
            end
        end
    end
end

Métodos basados en similitud

Tabla resumen

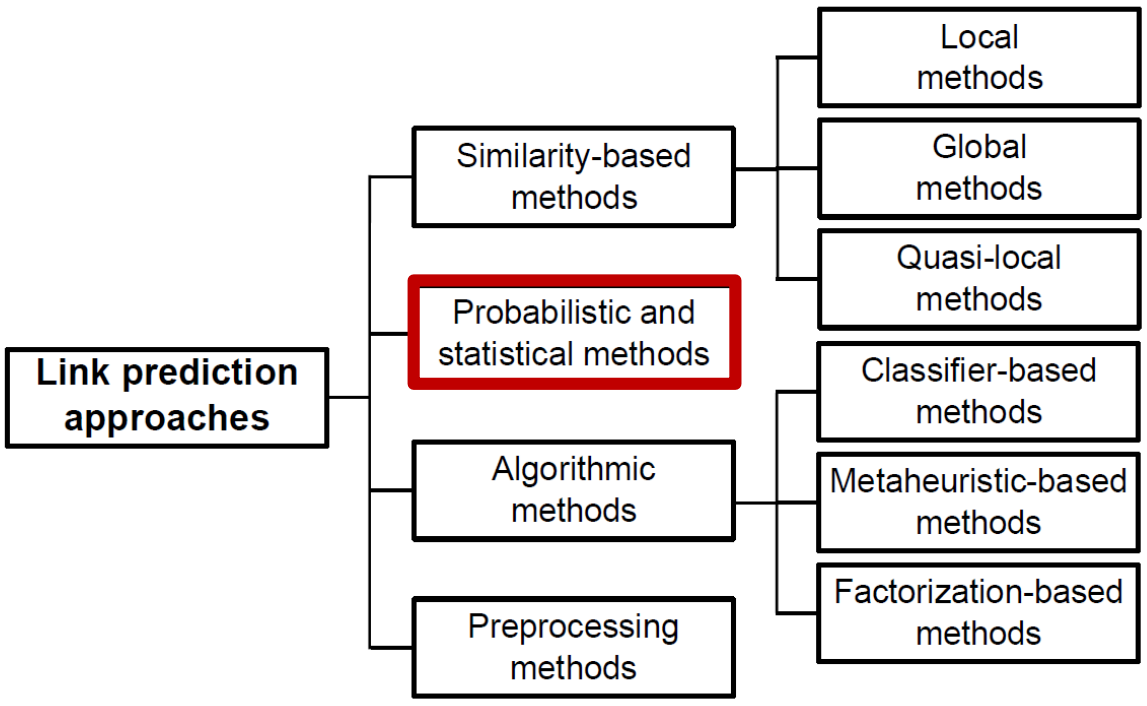
Local	CN	$O(vk^3)$	[Liben-Nowell and Kleinberg 2007]
	AA	$O(vk^3)$	[Adamic and Adar 2003]
	RA	$O(vk^3)$	[Zhou et al. 2009]
	RA-CNI	$O(vk^4)$	[Zhang et al. 2014]
	PA	$O(vk^2)$	[Barabási and Albert 1999]
	JA	$O(vk^3)$	[Jaccard 1901]
	SA	$O(vk^3)$	[Salton and McGill 1983]
	SO	$O(vk^3)$	[Sørensen 1948]
	HPI	$O(vk^3)$	[Ravasz et al. 2002]
	HDI	$O(vk^3)$	[Ravasz et al. 2002]
	LLHN	$O(vk^3)$	[Leicht et al. 2006]
	IA1	$O(vk^4)$	[Dong et al. 2011]
	IA2	$O(vk^3)$	[Dong et al. 2011]
	MI	$O(nk^6)$	[Tan et al. 2014]
	LNB	$O(O(f(z)) + vk^3)$	[Liu et al. 2011]
	CAR	$O(vk^4)$	[Cannistraci et al. 2013]

Métodos basados en similitud

Tabla resumen

Global	NSP	$O(ev \log v)$	[Liben-Nowell 2005]
	KI	$O(v^3)$	[Katz 1953]
	GLHN	$O(cv^2k)$	[Leicht et al. 2006]
	RW	$O(cv^2k)$	[Pearson 1905]
	RWR	$O(cv^2k)$	[Tong et al. 2006]
	FP	$O(cv^2k)$	[Vanunu and Sharan 2008]
	MERW	$O(cv^2k)$	[Li et al. 2011]
	SR	$O(v^2k^{2l+2})$	[Jeh and Widom 2002]
	PLM	$O(v^3)$	[Fouss et al. 2007]
	ACT	$O(v^3)$	[Fouss et al. 2007]
	RFK	$O(v^3)$	[Chebotarev and Shamis 2006]
	BI	$O(cv^2k)$	[Blondel et al. 2004]
Quasi local	LPI	$O(lv^2k)$	[Lü et al. 2009]
	LRW	$O(lv^2k)$	[Liu and Lü 2010]
	SRW	$O(lv^2k)$	[Liu and Lü 2010]
	ORA-CNI	$O(vk^6)$	[Zhang et al. 2014]
	FL	$O(lv^2k)$	[Papadimitriou et al. 2012]
	PFP	$O(vlk^l)$	[Lichtenwalter et al. 2010]

Métodos probabilísticos



Métodos probabilísticos

Hipótesis

La formación de la red se produce de acuerdo a algún modelo formal (de tipo estadístico).

Idea básica

Asumiendo que la red se ajusta a un modelo concreto, se estiman los parámetros de dicho modelo y se calcula la probabilidad de formación de cada posible enlace…

Métodos probabilísticos

Hierarchical structure model

Red organizada jerárquicamente

$$\mathcal{L}(D, \{p_n\}) = \prod p_n^{e_n}(1-p_n)^{l_n r_n - e_n}$$

Input: Network $G = (V, E)$, number n of dendrograms to sample.
Output: Probability $P_{x,y}$ for all unconnected pairs of nodes.
$Samples = \emptyset$;
for i ***from*** 1 ***to*** n **do**
    Initialize the Markov chain with a random dendrogram;
    Run Monte Carlo algorithm until equilibrium is reached;
    Insert resulting dendrogram D into $Samples$;
end
for ***each*** $e_{x,y}$ ***in*** $U_G - E$ **do**
    $avg\_prob = 0$;
    for ***each*** $sample$ ***in*** $Samples$ **do**
        $n \leftarrow$ lower common ancestor of x and y in $sample$;
        $avg\_prob \leftarrow$ avg_prob $+ \frac{\overline{p}_n}{|Samples|}$;
    end
    $P_{x,y} = avg\_prob$;
end

Métodos probabilísticos

Stochastic block model

Red organizada en torno a comunidades…

$$\mathcal{L}(G|\mathcal{M}) = \prod_{a \leq b; a,b \in \mathcal{M}} p_{a,b}^{l_{a,b}} (1 - p_{a,b})^{r_{a,b} - l_{a,b}}$$

$$P_{x,y} = \frac{\sum_{\mathcal{M} \in \omega} \mathcal{L}(e_{x,y} \in E|\mathcal{M}) \mathcal{L}(G|\mathcal{M}) p(\mathcal{M})}{\sum_{\mathcal{M}' \in \omega} \mathcal{L}(G|\mathcal{M}') p(\mathcal{M}')}$$

Métodos probabilísticos

Cycle formation model

Red con tendencia a cerrar ciclos…

"Los amigos de mis amigos son mis amigos"

$$p_{x,y}(c_1, ..., c_k) = \frac{c_1 \prod_{i=2}^{k} c_i^{|paths_{x,y}^i|}}{c_1 \prod_{i=2}^{k} c_i^{|paths_{x,y}^i|} + (1 - c_1) \prod_{i=2}^{k} (1 - c_i)^{|paths_{x,y}^i|}}$$

Input: Network $G = (V, E)$, model degree k.
Output: Probability $P_{x,y}$ for all unconnected pairs of nodes.
Compute Generalized Clustering Coefficients $C(2), ..., C(k)$;
c_1 = Connecting probability in random graph with same degree distribution that G;
c_2 = $\frac{(1-c_1)C(2)}{c_1 - 2c_1C(2) + C(2)}$;
for i ***from*** 3 ***to*** k **do**
    $c_i \leftarrow 0.5$;
end
for i ***from*** 3 ***to*** k **do**
    $c_i \leftarrow \arg\min_{c_i} |C(i) - f(c_1, ..., c_k)|$;
end
for ***each*** $e_{x,y}$ ***in*** $U_G - E$ **do**
    $P_{x,y} \leftarrow p_{x,y}(c_1, ..., c_k)$;
end

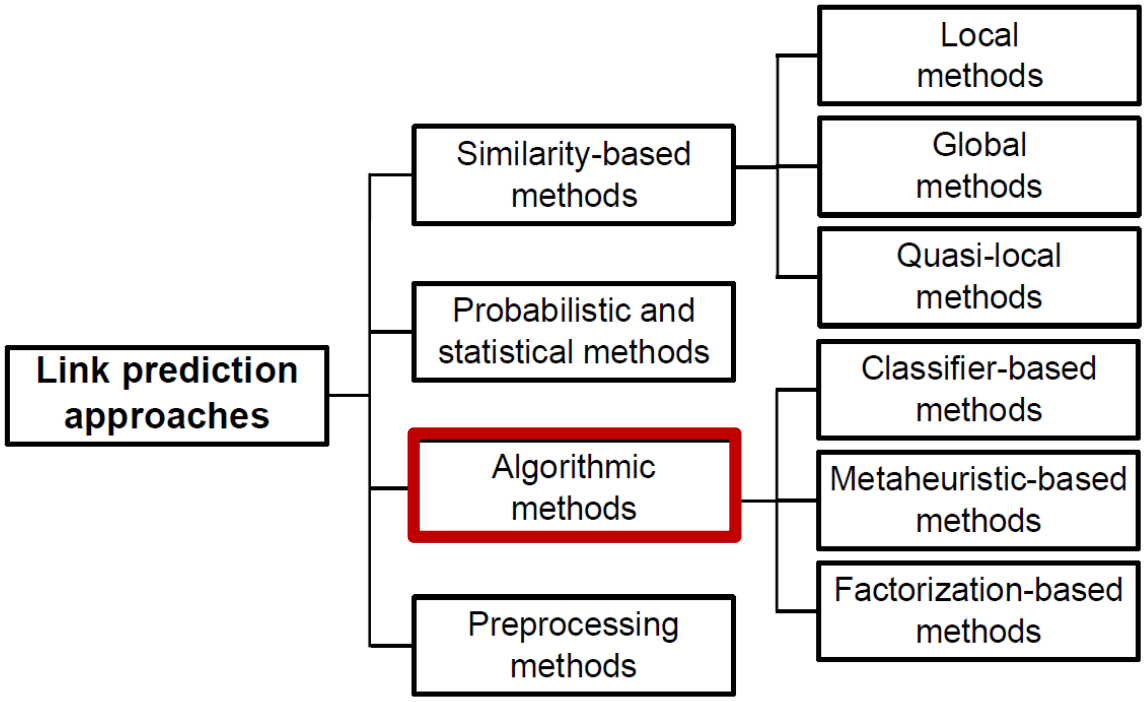
Métodos probabilísticos

Local co-occurrence model

Basado en propiedades topológicas locales ("escalable")

Input: Network $G = (V, E)$, central neighborhood set max size t, max path length k.
Output: Probability $P_{x,y}$ for all unconnected pairs of nodes.
for ***each*** $e_{x,y}$ ***in*** $U - E$ **do**
  $C_{x,y} = \emptyset$;
  for l ***from*** 2 ***to*** k **do**
    $p_i \leftarrow$ Compute and sort by length and frequency $paths^{l}_{x,y}$;
    for ***each*** p ***in*** p_i **do**
      if $|C_{x,y}| < t$ **then**
        Insert all nodes in p into $C_{x,y}$;
      end
    end
  end
  NDI = Compute non-derivable itemsets from $C_{x,y}$;
  $R_{x,y} = \emptyset$;
  for ***each*** *ndi* ***in*** *NDI* **do**
    if *ndi* ***in*** $C_{x,y}$ **then**
      Insert ndi into $R_{x,y}$;
    end
  end
  M = Initialize Markov Random Fields using $C_{x,y}$ and $R_{x,y}$;
  while ***not*** *M satisfies all constrains in* $R_{x,y}$ **do**
    for ***each*** r ***in*** $R_{x,y}$ **do**
      Update M to force satisfying r;
    end
  end
  $P_{x,y}$ = Infer probability of $e_{x,y}$ from M;
end

Métodos algorítmicos



Métodos algorítmicos

Métodos basados en clasificadores

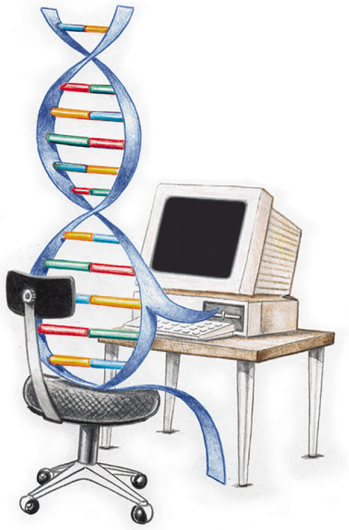
Consideran la predicción de enlaces como un problema clásico de aprendizaje supervisado (no balanceado).

- Árboles de decisión
- k-NN (vecinos más cercanos)
- SVMs [Support Vector Machines]
- Redes neuronales: perceptrones multicapa, RBFs…
- Naive Bayes
- Ensembles, p.ej. random forests

Métodos algorítmicos

Métodos basados en metaheurísticas

Algoritmos evolutivos
(permiten modelar la coexistencia de varios mecanismos de formación de enlaces).



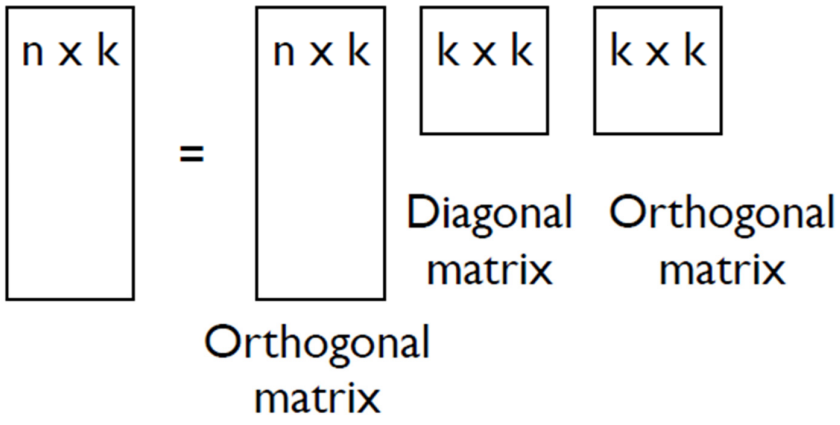
p.ej.
CMA-ES [Covariance Matrix Adaptation Evolution Strategy]

Métodos algorítmicos

Factorización de matrices

Muy utilizada en sistemas de recomendación

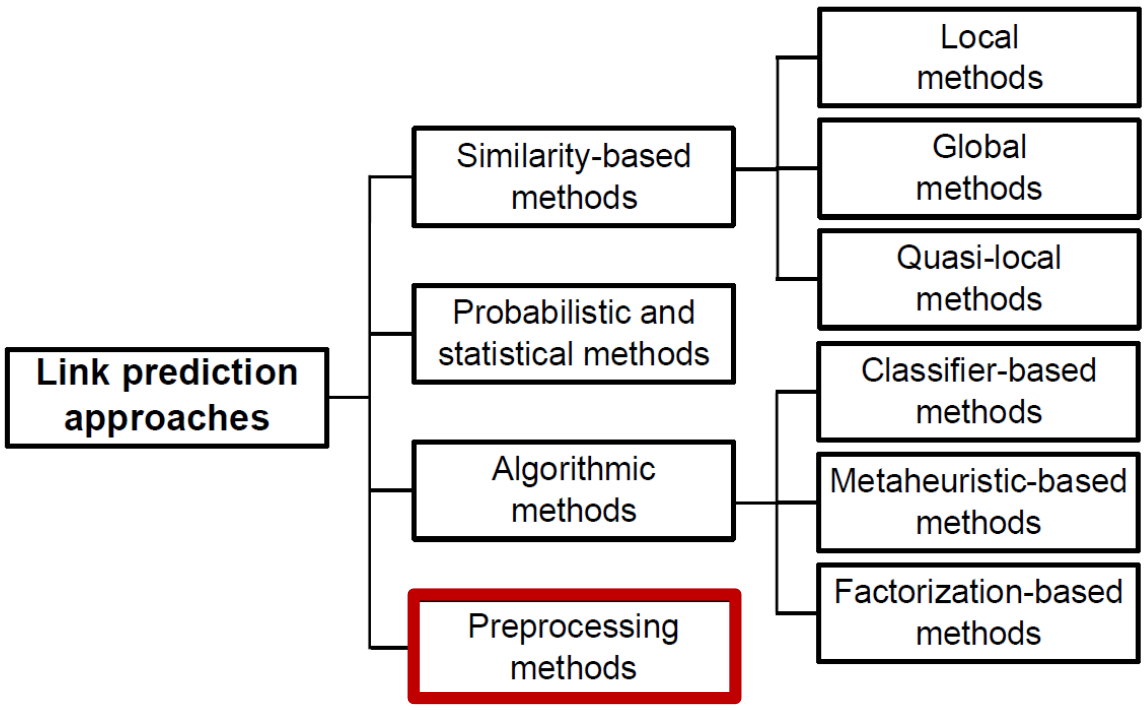
$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}}$$



p.ej.

SVD [Singular Value Decomposition]

Técnicas de preprocesamiento



Técnicas de preprocesamiento

Utillizadas en combinación con otros métodos, pretenden reducir el ruido presente en las redes en forma de enlaces falsos o "débiles".

- **Low-rank approximation** (SVD)
  ~ Extracción de características
- **Unseen bigrams**
  (se reemplaza un nodo por sus nodos más similares)
- **Filtering**
  (eliminación de los enlaces más débiles, determinados con la ayuda de una método de predicción de enlaces).

Agradecimientos

Víctor Martínez

Supervised Data Mining in Networks: Methods and Applications

PhD Thesis, 2018

Department of Computer Science and Artificial Intelligence

University of Granada (Spain)

Víctor Martínez, Fernando Berzal & Juan-Carlos Cubero:
A survey of link prediction in complex networks.
ACM Computing Surveys 49(4):69:1-69:33, 2017.

Víctor Martínez, Fernando Berzal & Juan-Carlos Cubero:
Adaptive degree penalization for link prediction.
Journal of Computational Science, 13:1-9, March 2016

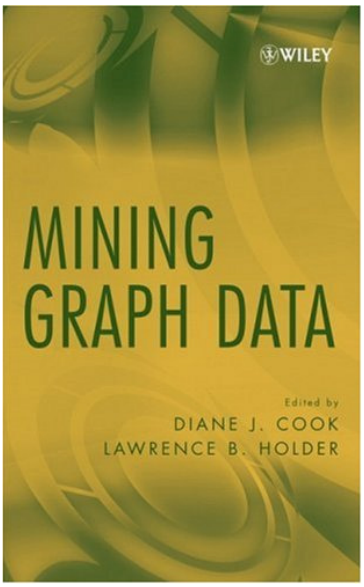
NOESIS

Network-Oriented Exploration,
Simulation, and Induction System
https://noesis.ikor.org

Víctor Martínez, Fernando Berzal & Juan-Carlos Cubero:
NOESIS: A Framework for Complex Network Data Analysis.
Complexity, 2019. https://doi.org/10.1155/2019/1439415

Bibliografía

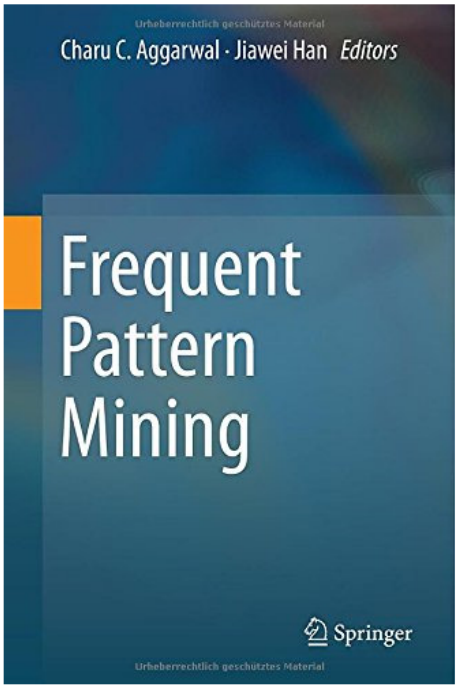
- Diane J. Cooke & Lawrence B. Holder (editors):
  Mining Graph Data.
  Wiley, 2007.
  ISBN 0-471-73190-0



Chapter 5
Discovery of frequent substructures
Xifeng Yang & Jiawei Han

Bibliografía

- Charu C. Aggarwal & Jiawei Han (editors): **Frequent Pattern Mining**. Springer, 2014. ISBN 3319078208.



Chapter 13
Mining Graph Patterns
Hong Chen, Xifeng Yang & Jiawei Han

Bibliografía

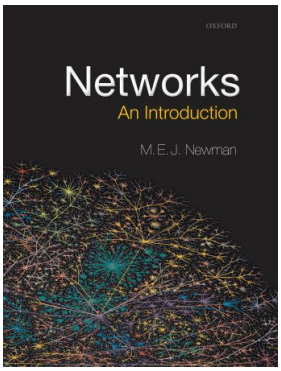
- Kayhan Erciyes: **Complex Networks: An Algorithmic Perspective**. CRC Press, 2014. ISBN 1466571667.

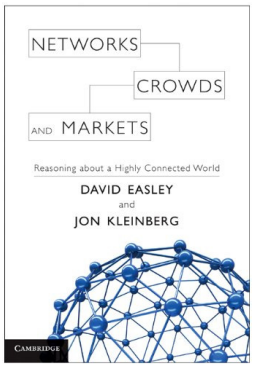
Chapter 9
Network Motif Discovery

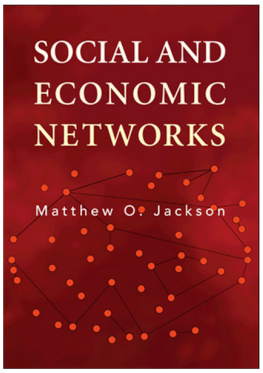
Chapter 10
Protein Interaction Networks
10.4 Network Motifs in PPI Networks

Bibliografía

- David Easley & Jon Kleinberg: **Networks, Crowds, and Markets: Reasoning About a Highly Connected World**. Cambridge University Press, 2010. ISBN 0521195330 http://www.cs.cornell.edu/home/kleinber/networks-book/
- Mark Newman: **Networks: An Introduction**. Oxford University Press, 2010. ISBN 0-19-920665-1
- Matthew O. Jackson: **Social and Economic Networks**, Princeton University Press, 2008. ISBN 0-691-13440-5

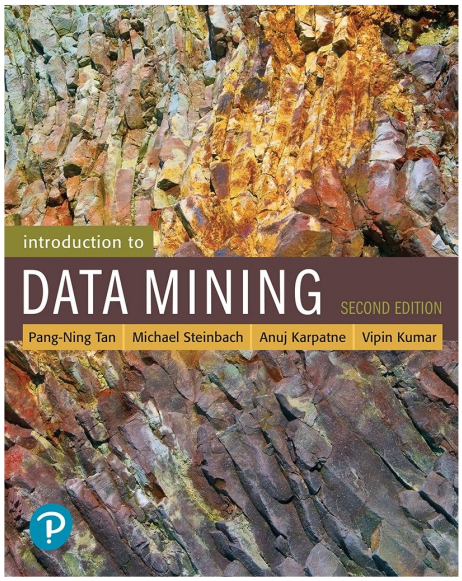






Bibliografía

Pang-Ning Tan,
Michael Steinbach,
Vipin Kumar &
Anuj Karpatne:
Introduction to Data Mining,
2nd edition, Addison Wesley, 2018.
ISBN 0133128903



Bibliografía

Jiawei Han,
Jian Pei &
Hanghang Tong:
Data Mining:
Concepts and Techniques,
4th edition, Morgan Kaufmann, 2022.
ISBN 0128117605