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

Patrones en grafos

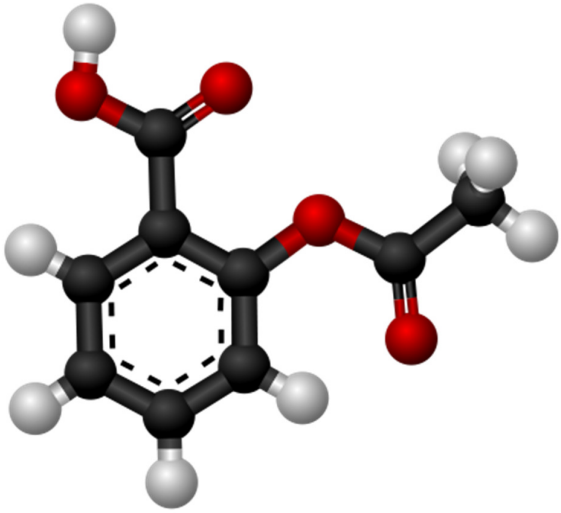
© Fernando Berzal, berzal@acm.org

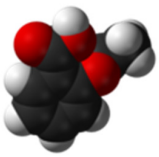
Patrones en grafos

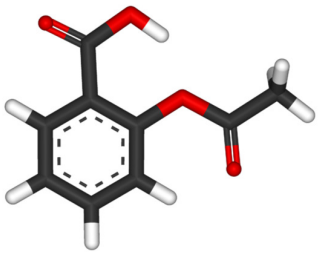
- Graph Mining
- Isomorfismo de (sub)grafos
- Subgrafos frecuentes
  - Algoritmos derivados de Apriori
  - Algoritmos derivados FP-Growth
  - Métodos escalables
- Motif Discovery
  - Algoritmos exactos [exact census algorithms]
  - Algoritmos aproximados (con muestreo / probabilísticos)
- Aplicaciones
  - Compresión de datos
  - Indexación & sistemas de recuperación de información
  - Redes de interacción de proteínas, e.g. NeMoFinder

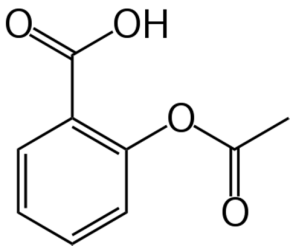
Patrones en grafos

El compuesto químico de la aspirina



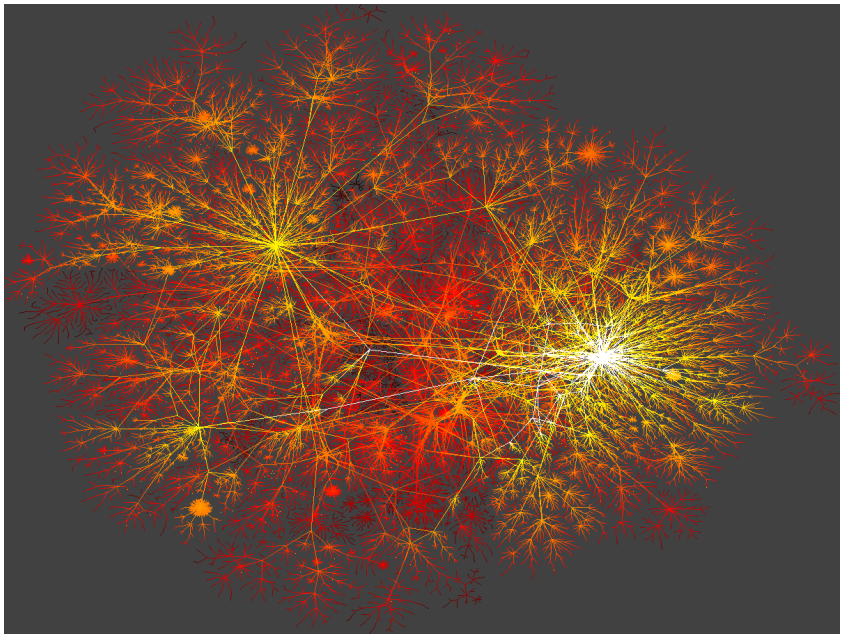






Patrones en grafos

Internet

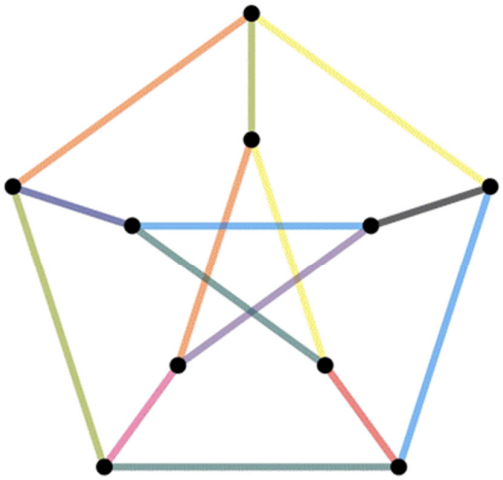


Patrones en grafos

¿Qué interés tiene encontrar patrones en grafos?

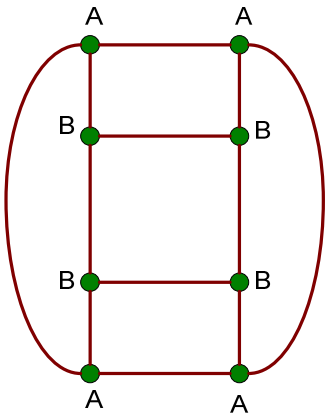
- Caracterizar conjuntos de grafos.
- Crear modelos de clasificación de grafos.
- Diseñar métodos de agrupamiento en grafos.
- Construir índices en bases de datos de grafos.

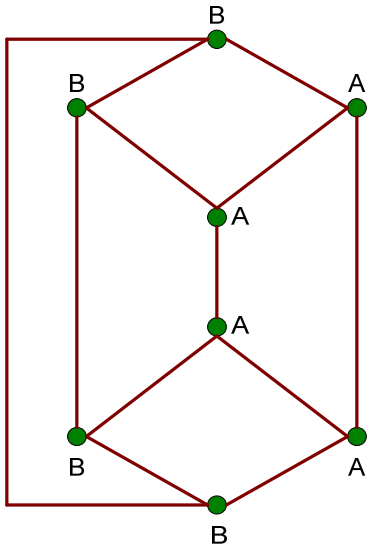
Isomorfismo de (sub)grafos



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

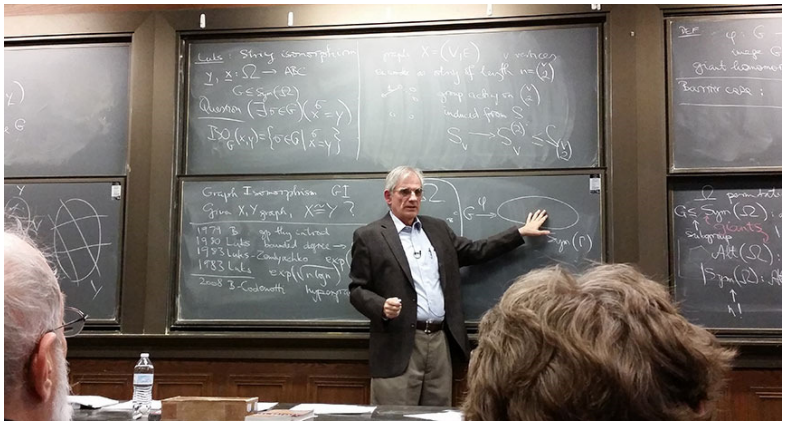
P

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

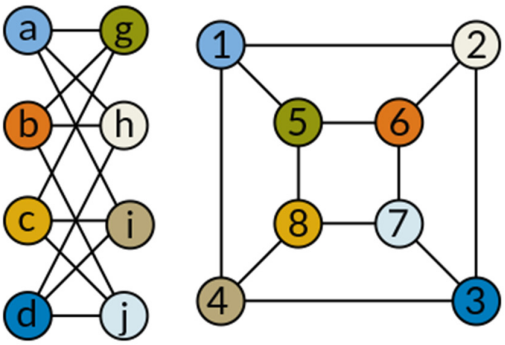
Isomorfismo de (sub)grafos

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

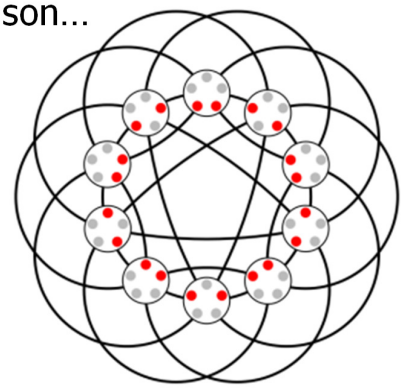
O (exp (polylog (n))) = O (exp ($(\log n)^k$))

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

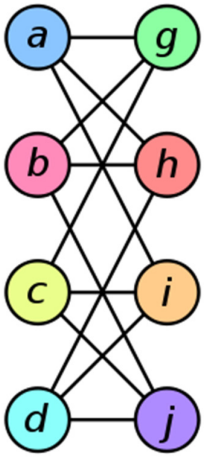
P

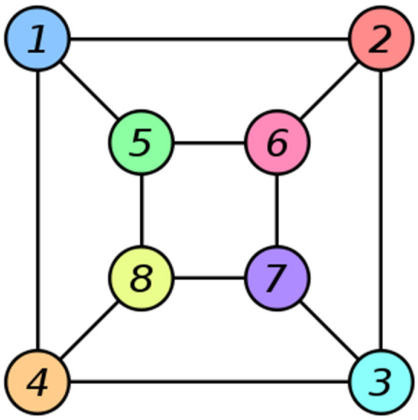
O (exp (polylog (n)) = O (exp ($(\log n)^k$))
http://people.cs.uchicago.edu/~laci/quasipoly.html

Isomorfismo de (sub)grafos

Ejemplo

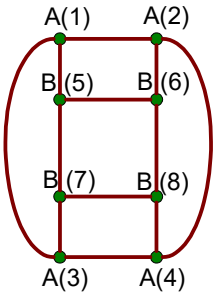
f(a) = 1
f(b) = 6
f(c) = 8
f(d) = 3
f(g) = 5
f(h) = 2
f(i) = 4
f(j) = 7



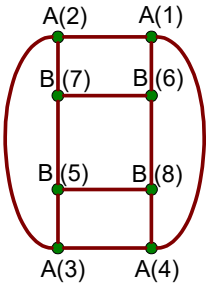


Isomorfismo de (sub)grafos

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

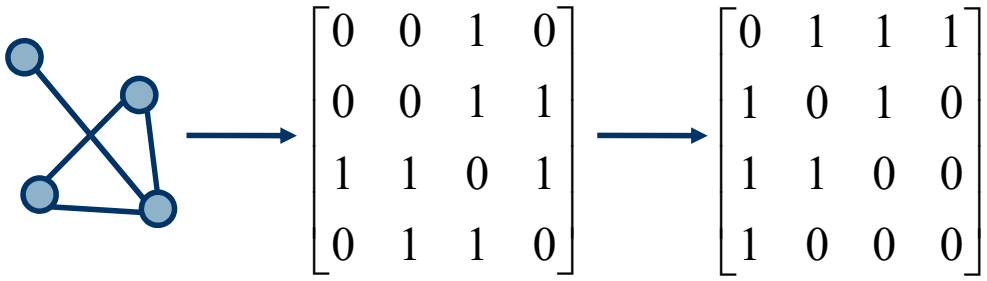
- Cuando manejamos bases de datos de grafos,
  ¡¡¡ tenemos que comparar con conjuntos de grafos !!!
- Se hace imprescindible en la práctica el uso de técnicas de preprocesamiento e indexación.

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)

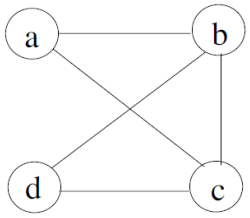
Propiedad de las matrices de adyacencia de dos grafos isomorfos: $A_2 = P\, A_1\, P^T$, donde P es una matriz que representa una permutación (la que define el isomorfismo).

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

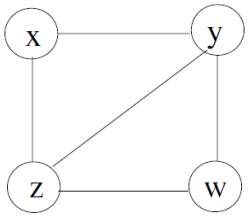
$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} g & h & i \\ a & b & c \\ d & e & f \end{bmatrix}$$

Isomorfismo de (sub)grafos

Algoritmo de Ullman (1976)



G_1



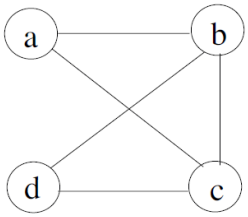
G_2

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

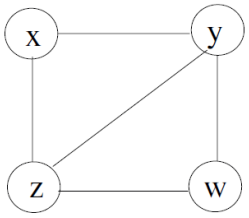
Isomorfismo de (sub)grafos

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

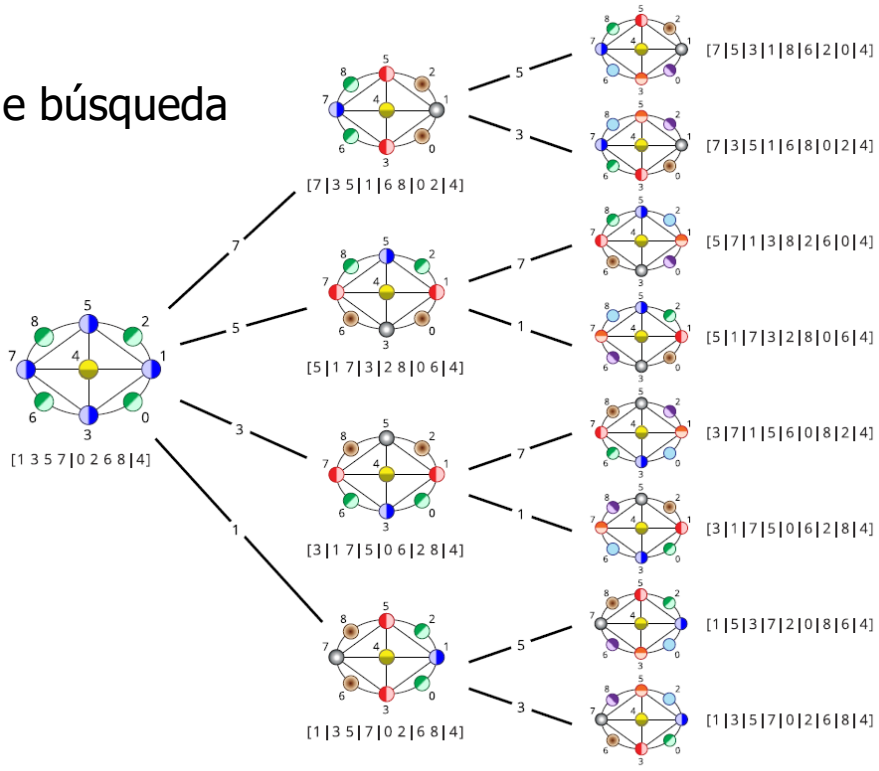
Input : $G_1(V_1, E_1)$
Output : $G_2(V_2, E_2)$: a canonical graph

$\mathcal{P} \leftarrow$ 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

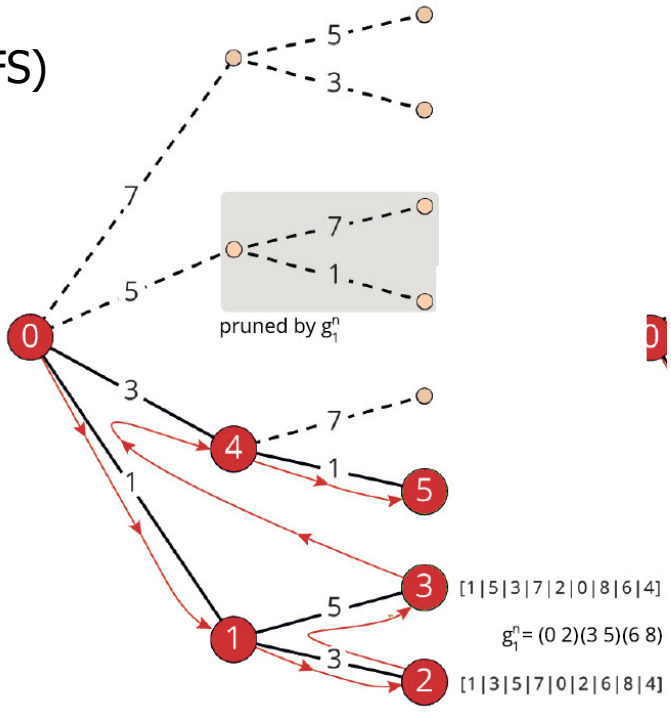


Isomorfismo de (sub)grafos

nauty

Exploración (DFS)

y poda…



Isomorfismo de (sub)grafos

VF2

[Cordella et al., TPAMI 2004]

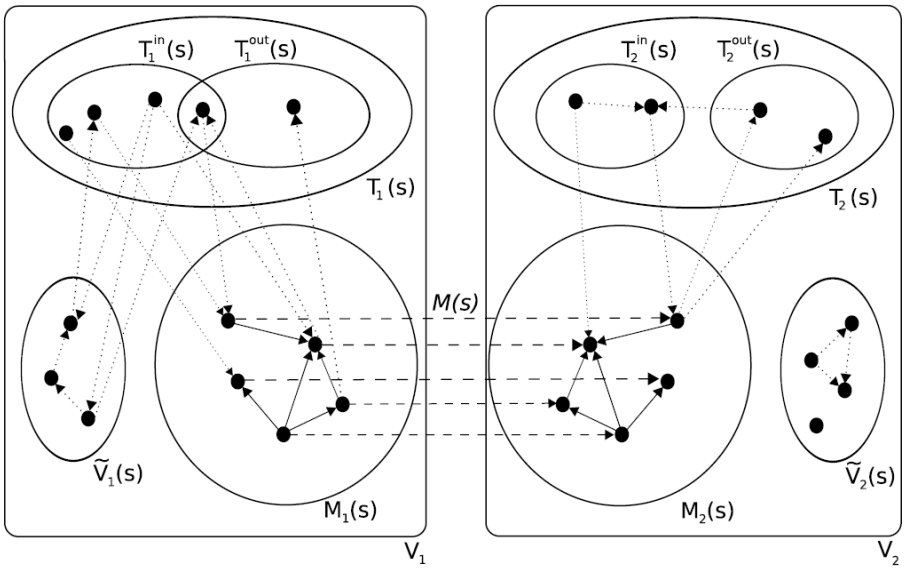
- DFS
- Reglas para podar el árbol de búsqueda que comprueban, para cada pareja de nodos candidatos, si el emparejamiento parcial es compatible.

	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

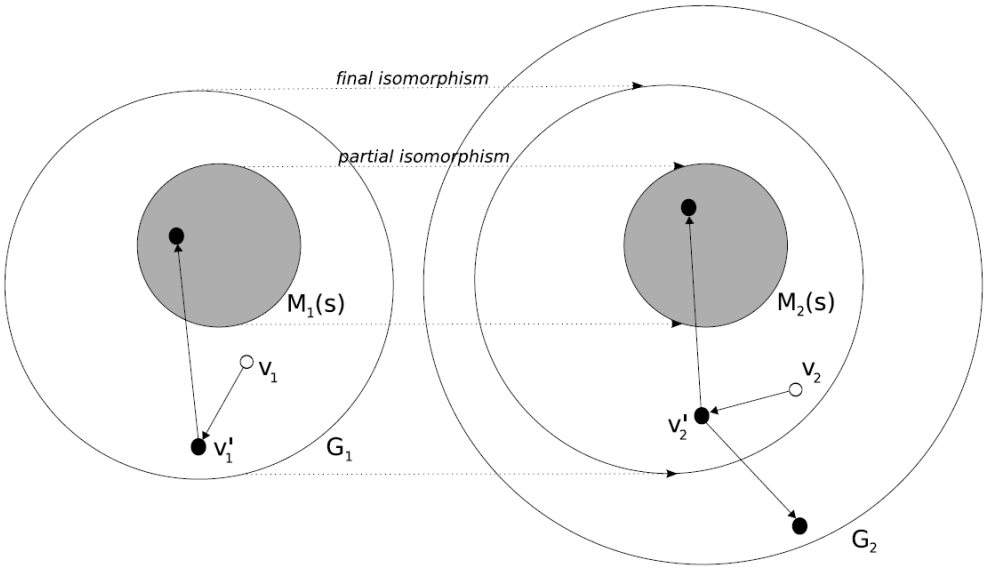
Correspondencia parcial



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]

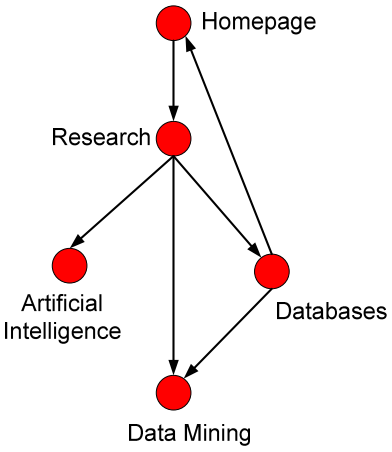
```
procedure CompatiblePaths(v1, v2, d)
    Output : the mappings between G1 and G2
    for all x in 1,...,d do
        for all y in 1,...,2^d do
            if PathsDS[v1][x][y] > PathsDS[v2][x][y] then
                return false
            end if
        end for
    end for
end procedure
```

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

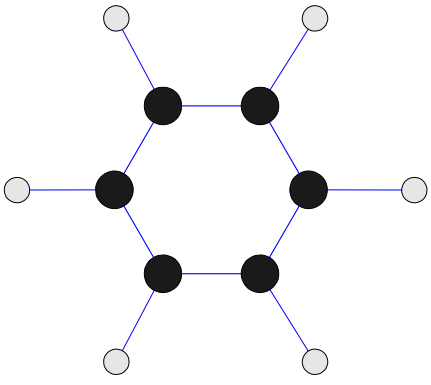
Subgrafos frecuentes

Identificación de patrones frecuentes en grafos

Aplicaciones: Web Mining, Bioinformática, redes sociales…



Grafo dirigido



Grafo no dirigido

Subgrafos frecuentes

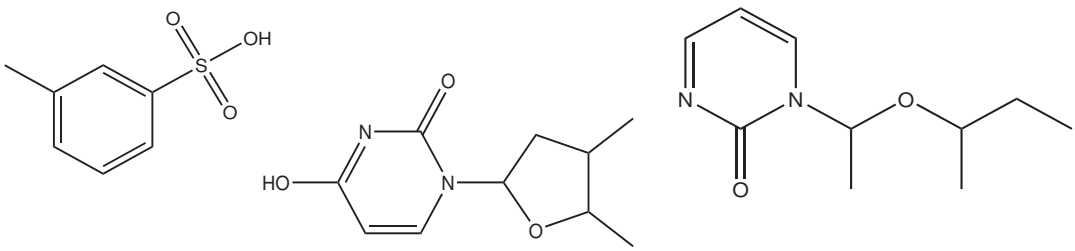
- **Subgrafo**
  Un grafo g es un subgrafo de otro grafo G ($g \subseteq G$) si existe un isomorfismo de subgrafos de g a G.

- **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

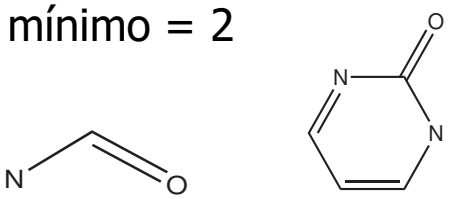
Ejemplo

Conjunto de datos (grafos no dirigidos)



Patrones frecuentes

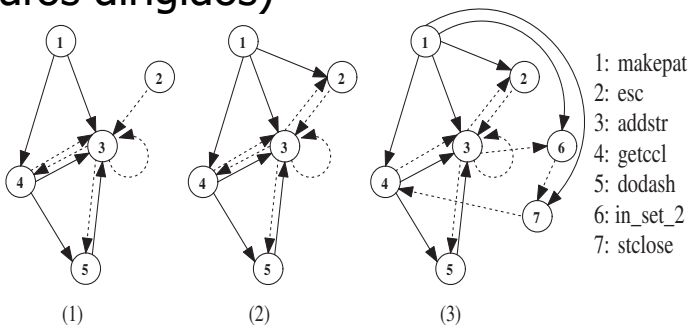
- Umbral de soporte mínimo = 2



Subgrafos frecuentes

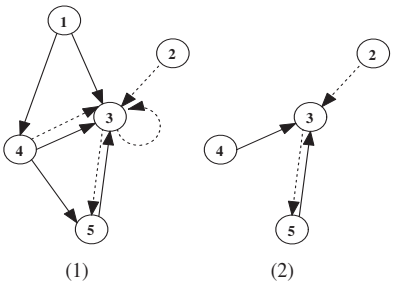
Ejemplo

Conjunto de datos (grafos dirigidos)



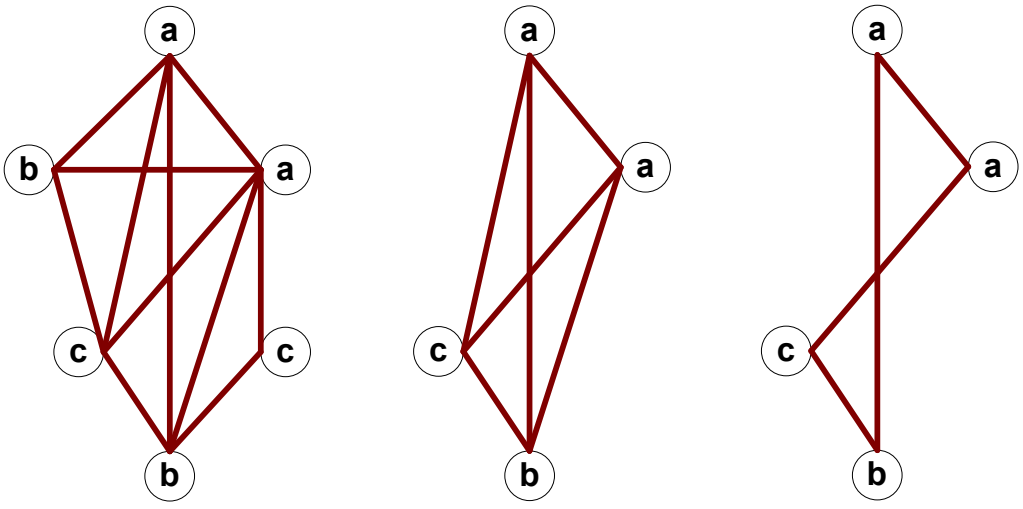
Patrones frecuentes

- Umbral de soporte mínimo = 2



Subgrafos frecuentes

Tipos de subgrafos frecuentes



Grafo original

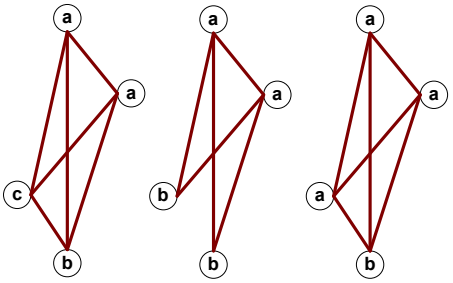
Subgrafo inducido

Subgrafo empotrado

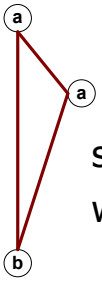
Subgrafos frecuentes

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

support = 3

weighted support = 6

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

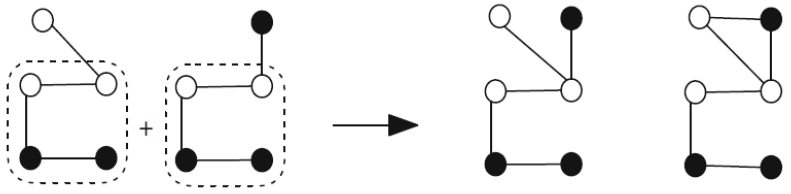
- Tipo de grafos (dirigidos/no-dirigidos, etiquetados…)
- Tipo de patrones identificados (inducidos/empotrados)
- Cálculo del soporte
- Orden de búsqueda (anchura vs. profundidad)
- Generación de candidatos (Apriori vs FP-Growth)
- Eliminación de duplicados
- Orden de identificación de patrones (p.ej.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

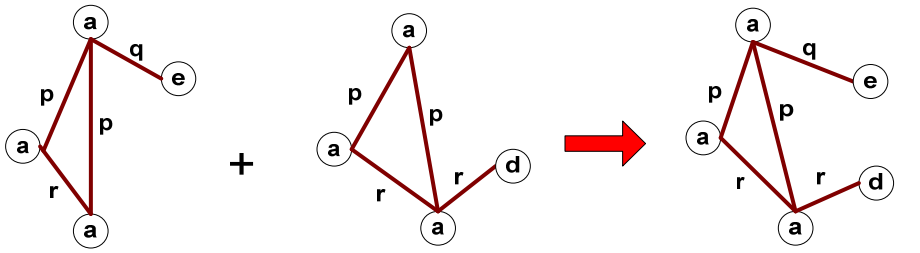
Input: Graph dataset D, minimum support threshold min_sup,
size-k frequent subgraphs S_k
Output: The set of size-$(k+1)$ frequent subgraphs S_{k+1}

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]

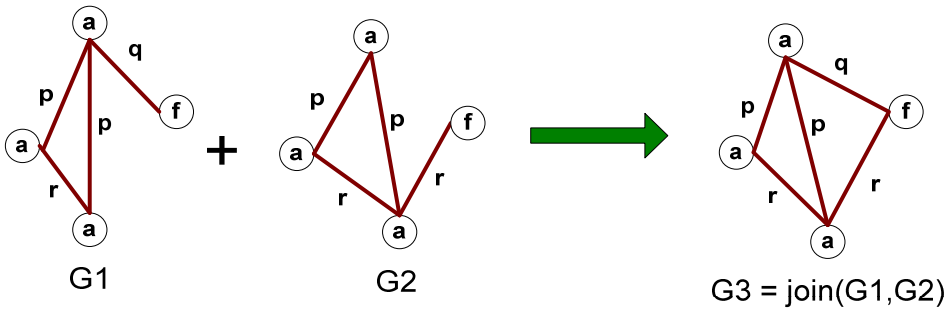
Vertex growing



Subgrafos frecuentes: Apriori

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

Edge growing

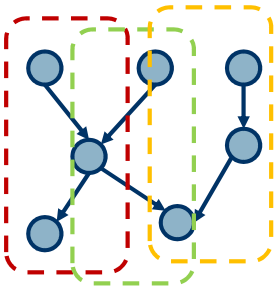


Subgrafos frecuentes: Apriori

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.

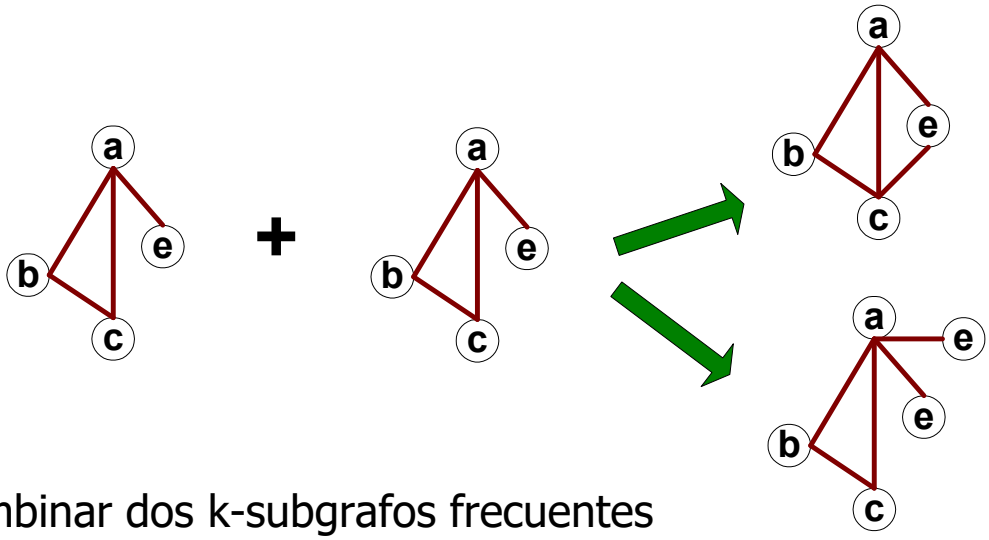


Grafo con 3 EDPs

Subgrafos frecuentes: Apriori

Generación de candidatos [edge growing]

p.ej. Mismas etiquetas en distintos nodos

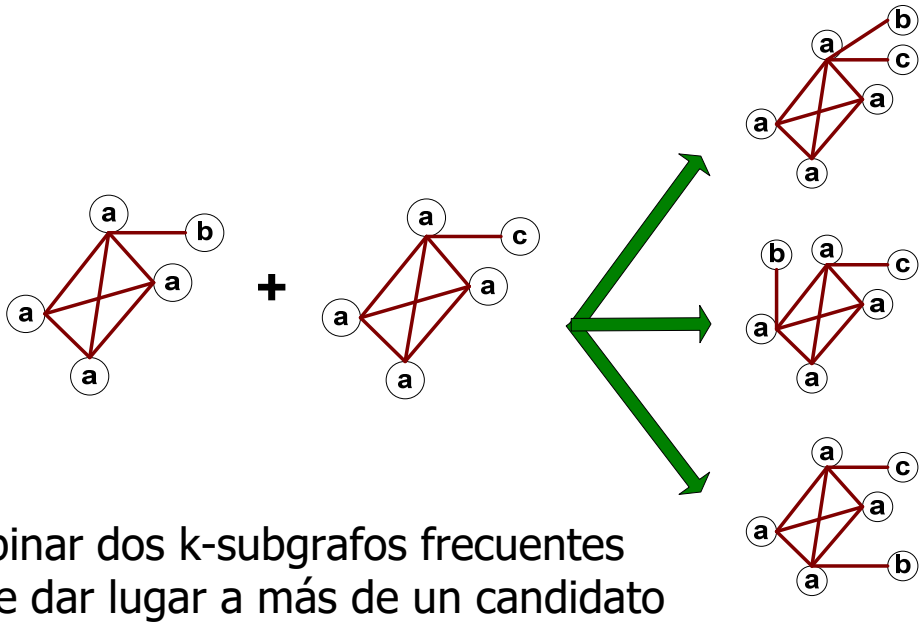


Combinar dos k-subgrafos frecuentes puede dar lugar a más de un candidato

Subgrafos frecuentes: Apriori

Generación de candidatos [edge growing]

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

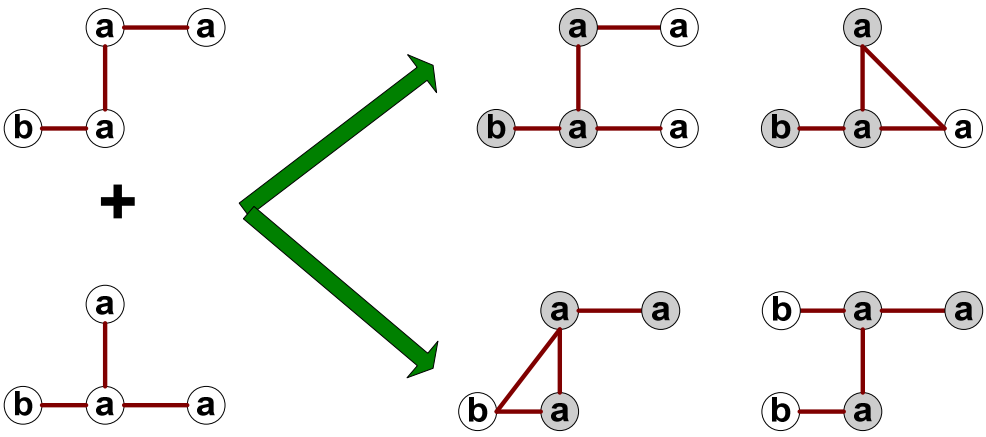


Combinar dos k-subgrafos frecuentes
puede dar lugar a más de un candidato

Subgrafos frecuentes: Apriori

Generación de candidatos [edge growing]

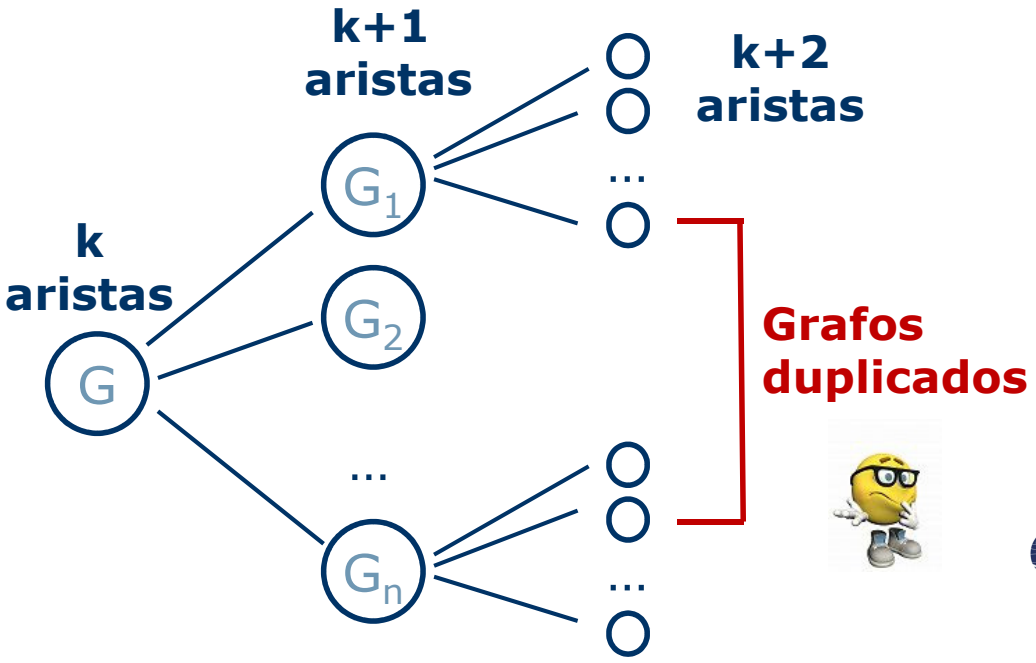
p.ej. Múltiples núcleos



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

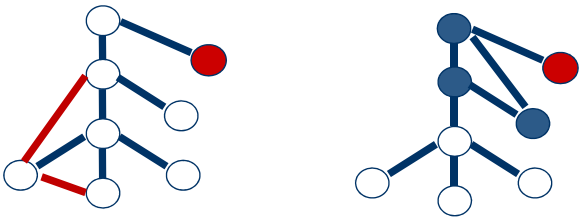
Algoritmos derivados de FP-Growth

- 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]

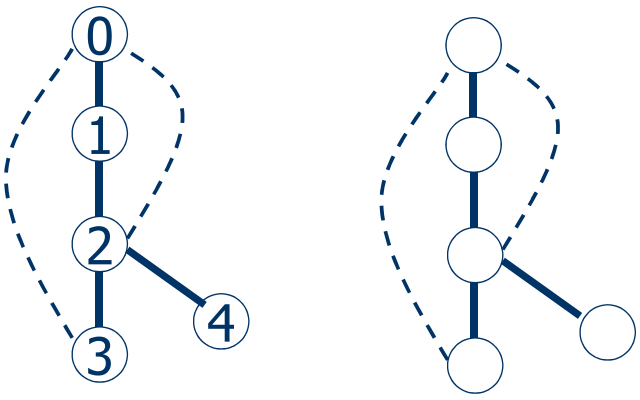
"Right-most extension"



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

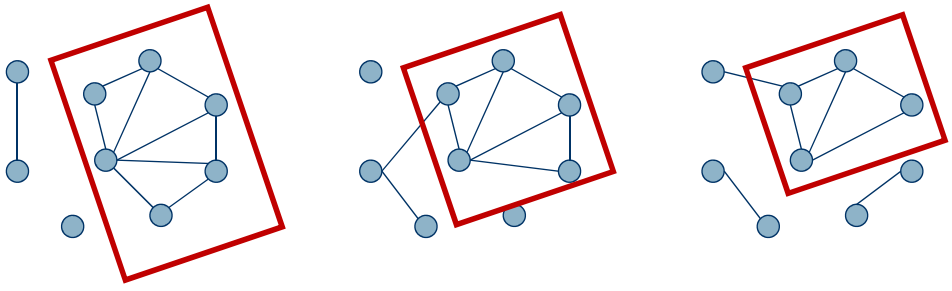
CloseGraph [Yan & Han, KDD'2003]

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

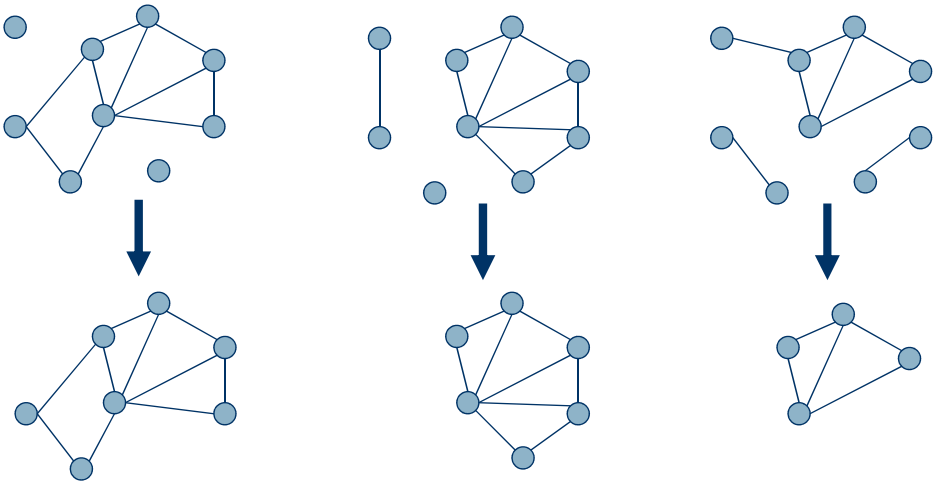
CloseCut & Splat [Yan, Zhou & Han, KDD'2005]
Subestructuras densas



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

Subgrafos frecuentes: FP-Growth

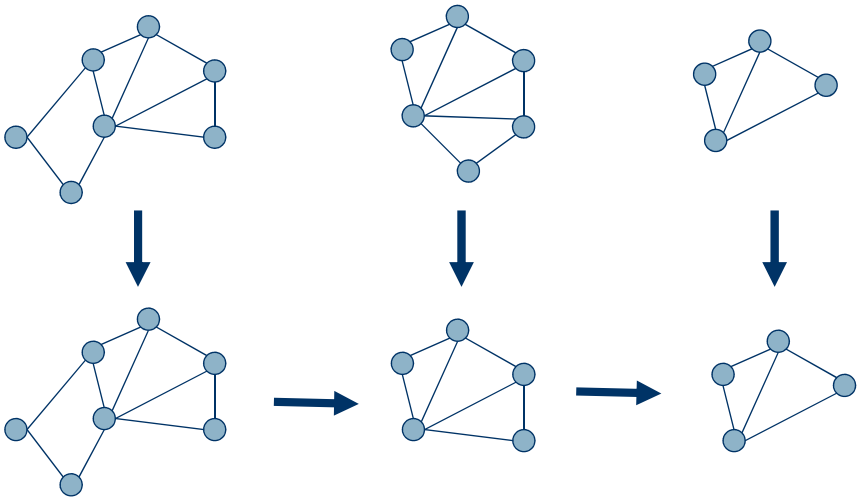
CloseCut & Splat [Yan, Zhou & Han, KDD'2005]
Subestructuras densas: Reducción de patrones (1/2)



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

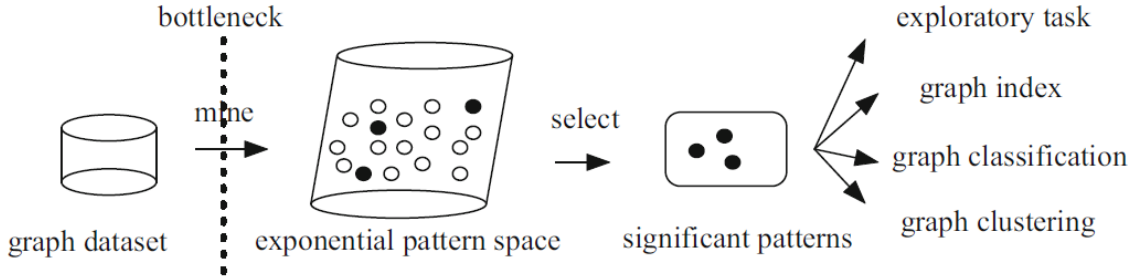
- MoFa [Borgelt & Berthold, ICDM'2002]
- FFSM [Han et al., ICDM'2003]
- SPIN [Huan et al., KDD'2004]
- GREW [Kuramochi & Karypis, ICDM'2004]
- SiGram [Kuramochi & Karypis, DMKD'2005]
- TSMiner [Jin et al., KDD'2005]
- MARGIN [Thomas et al., KDD'2006]

...

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]

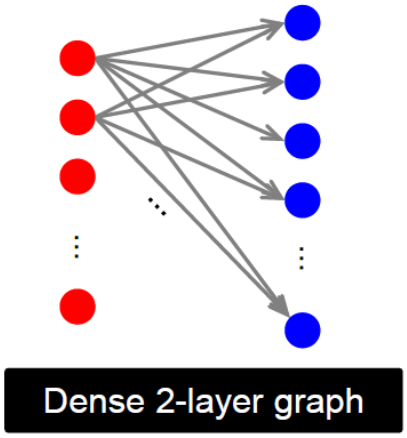
- K-trusses = Triangle k-cores: $O(m^{1.5})$
  [Wang & Cheng, PVLDB'2012]
  [Zhang & Partharasany, ICDE'2012]

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 $K_{s,t}$

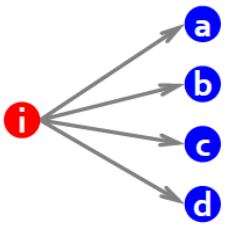
¿Cómo?

Encontrando itemsets frecuentes…

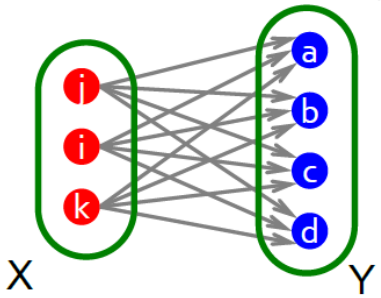
Subgrafos frecuentes

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

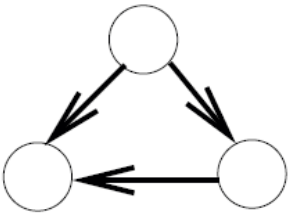
Modelos escalables: Subgrafos pequeños

"Motif" de tamaño k

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

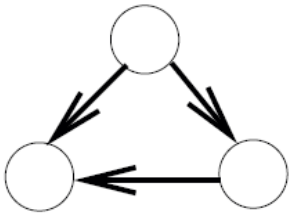
Ejemplo

Feed-forward loop

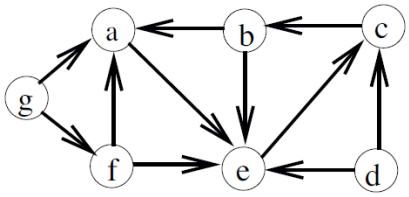


Motif Discovery

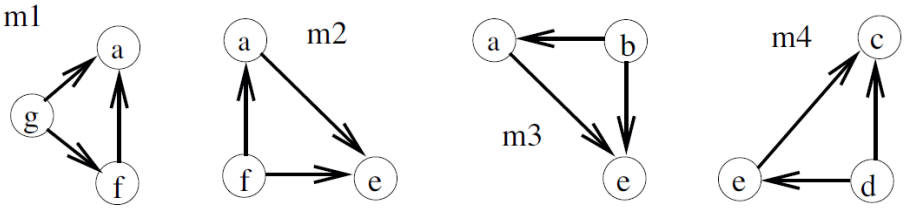
Ejemplo: Feed-forward loop



- Grafo de entrada

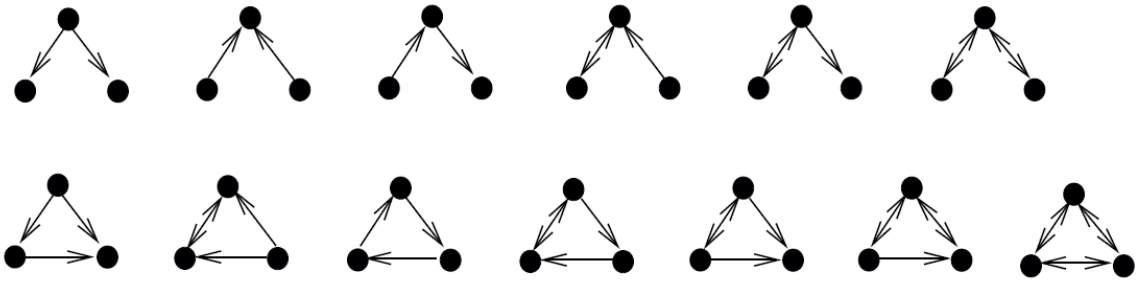


- Ocurrencias del patrón



Motif Discovery

Los 13 "motifs" dirigidos de tamaño 3:

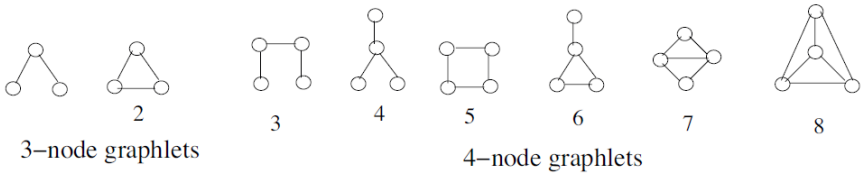


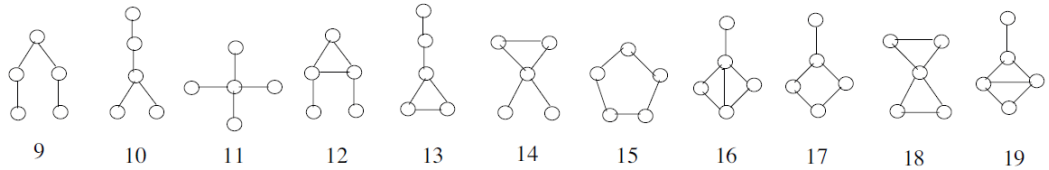
Número exponencial de posibles "motifs": **$O(2^{n(n-1)})$**

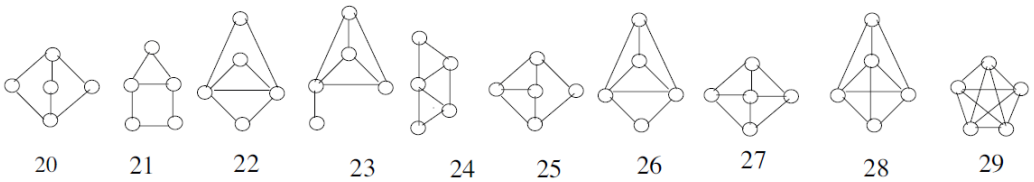
Motif size	3	4	5	6	7	8	9	10
Undirected subgraphs	2	6	21	112	853	$\approx 10^4$	$\approx 10^5$	$\approx 10^7$
Directed subgraphs	13	199	9364	10^6	$\approx 10^9$	$\approx 10^{12}$	$\approx 10^{16}$	$\approx 10^{20}$

Motif Discovery

"Graphlets" de tamaños 3, 4 y 5



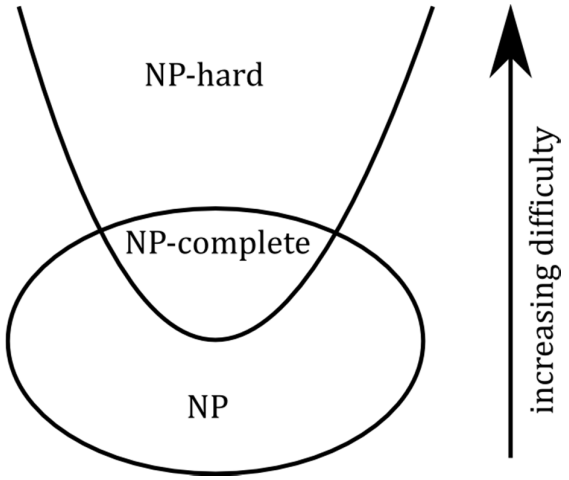




5–node graphlets

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

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

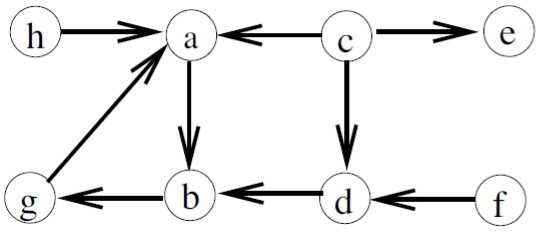
- **Significance profile:**
  (comparación de redes con respecto a los motifs que contienen)

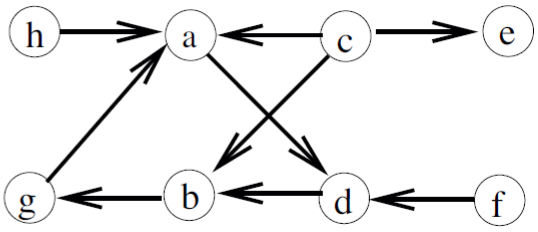
$$SP(m_i) = \frac{Z(m_i)}{\sqrt{\sum_{i=1}^{n} Z(m_i)^2}}$$

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

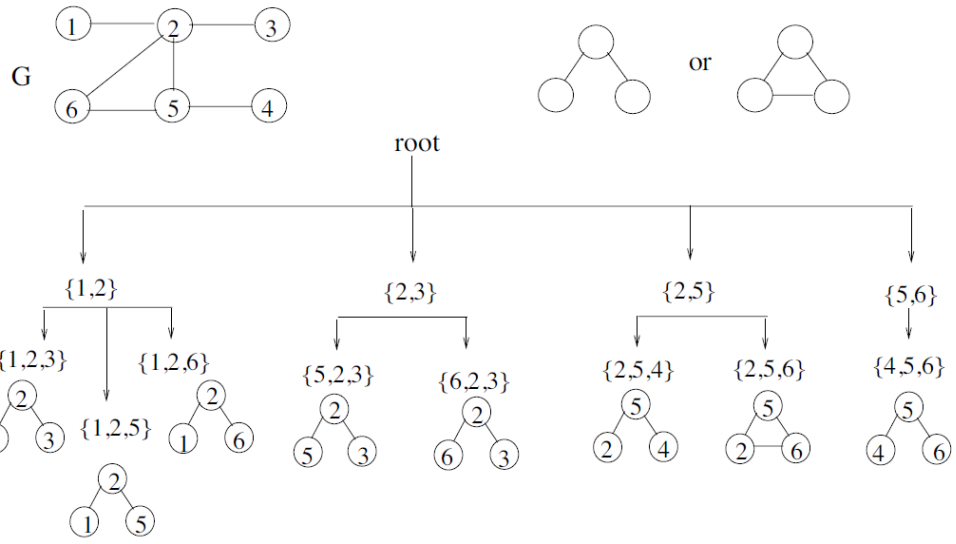
Mfinder

```
Input : G(V,E)                                          ▷ weighted or unweighted
        int 2 ≤ k ≤ n
Output : Subgraphs of size k
for all (u,v) ∈ 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 ∈ G' do
                for all (u,w) ∈ E do                    ▷ check neighbors
                    if w ∉ G' then
                        if G' ∪ {w} is not in Hash then
                            Extend(G' ∪ {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.

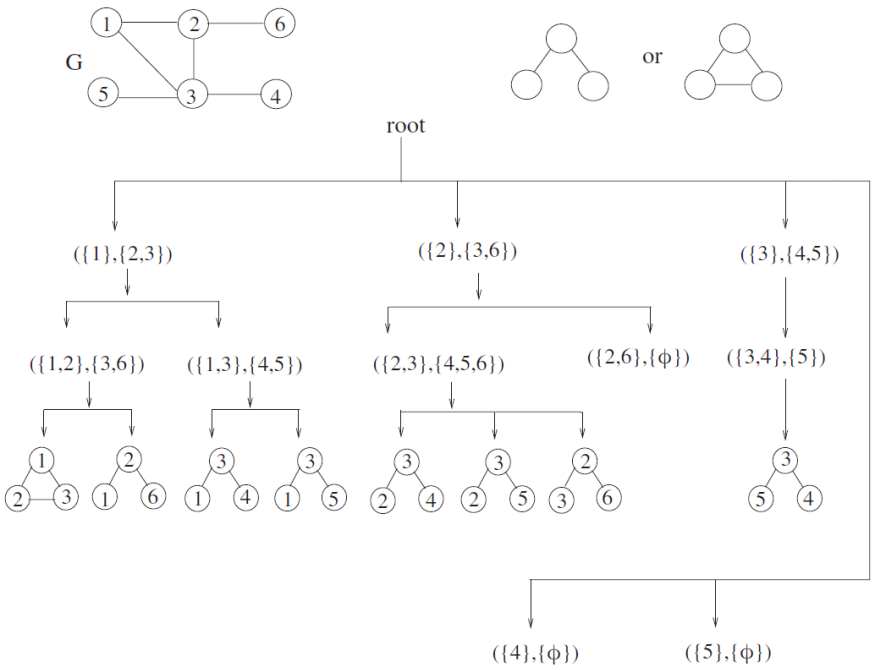
Motif Discovery

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

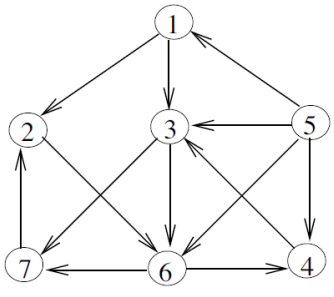
Kavosh

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

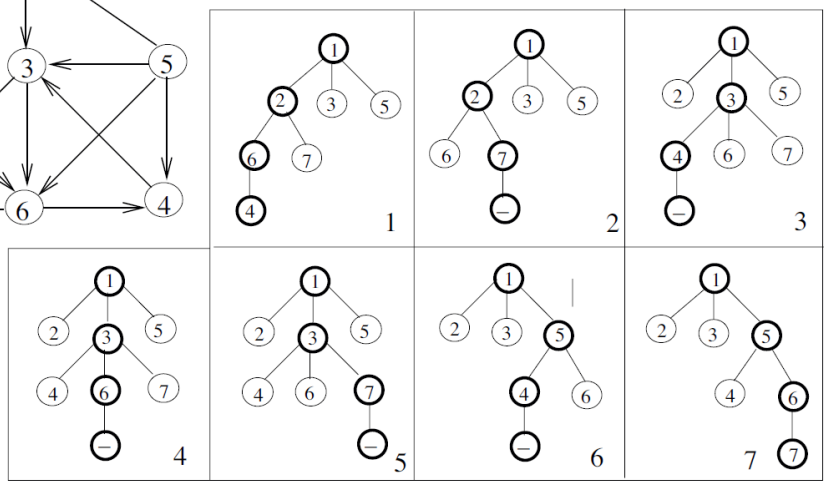
Input : $G(V, E)$ undirected or directed
Output : L : List of all k-size subgraphs of G
for all $v \in V$ **do**
  $visited[v] \leftarrow true$; $S_0 \leftarrow v$
  $EnumVertex(G, v, S, k-1, 1)$
  $visited[v] \leftarrow true$
end for

Motif Discovery

Kavosh



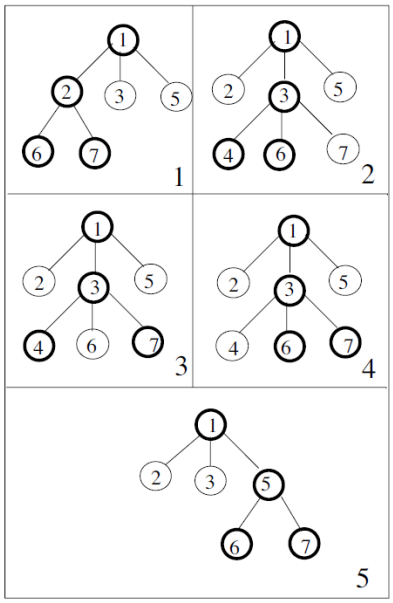
Motifs de tamaño 4 partiendo del nodo 1 (1,1,1)



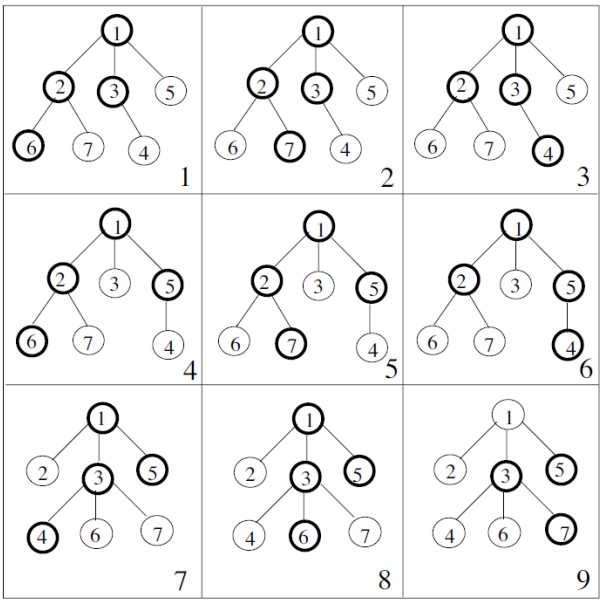
Motif Discovery

Kavosh

(1,2)

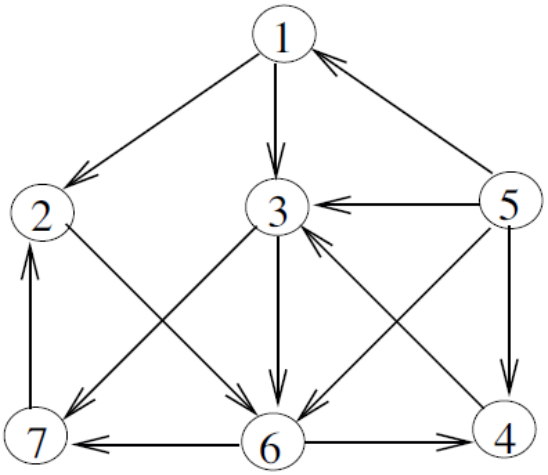


(2,1)

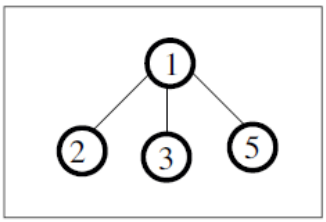


Motif Discovery

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

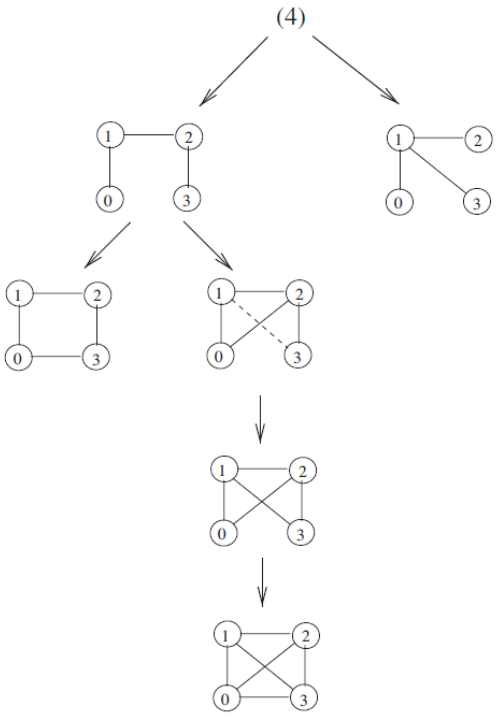
Input : $G(V,E)$, **int** $1 \leq k \leq n$, Δ : threshold value
Output : L : List of frequent k-size subgraphs of G
repeat
  $G'(V',E') \leftarrow Get\_Next\_BFS(T_k)$
  if $|E'| = k-1$ **then**
    $MappingModule(G',G)$
  else
    $EnumeratingModule(G',G,T_k)$
  end if
  save F_2
  if $|F_G| > \Delta$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

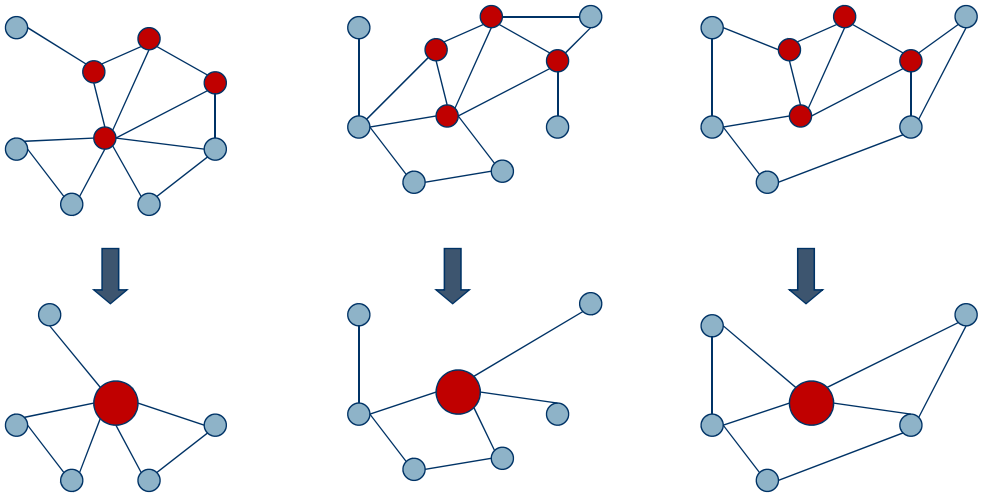
Aplicaciones

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

Aplicaciones: Compresión

Compresión de datos

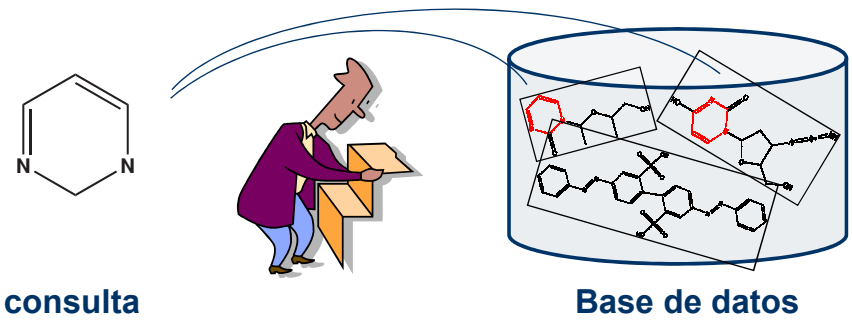
Extraer subgrafos comunes
y condensar éstos en un solo nodo



Aplicaciones: Indexación

Indexación

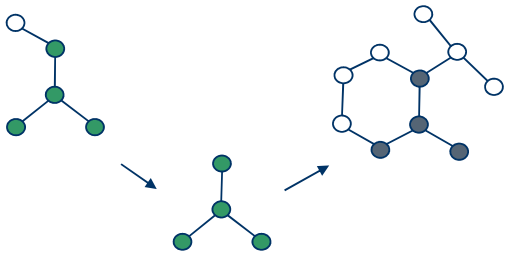
En consultas sobre bases de datos de grafos, recorrer secuencialmente toda la base de datos sería demasiado ineficiente tanto por las operaciones de E/S como por las comprobaciones de isomorfismo entre grafos



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.

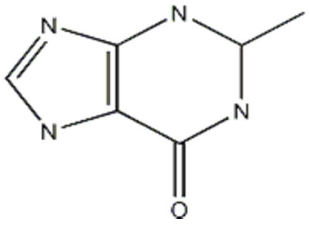
IDEA:
Cuanto más se reduzca el número de falsos positivos, menor será el tiempo de respuesta

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

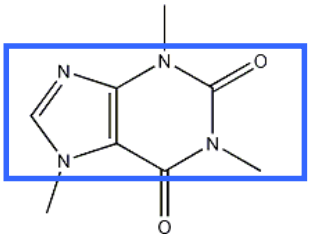
Aplicaciones: SRI

Recuperación de información

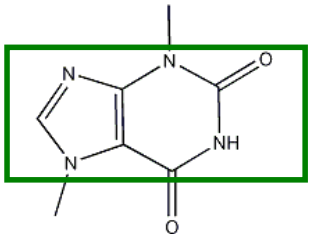
Grafo de la consulta



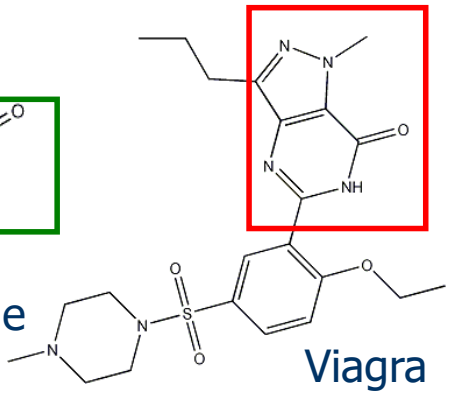
Resultado de la consulta



cafeína



diurobromine



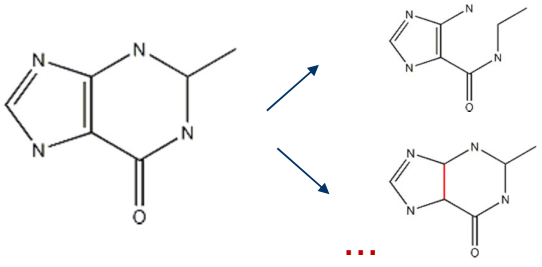
Viagra

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).

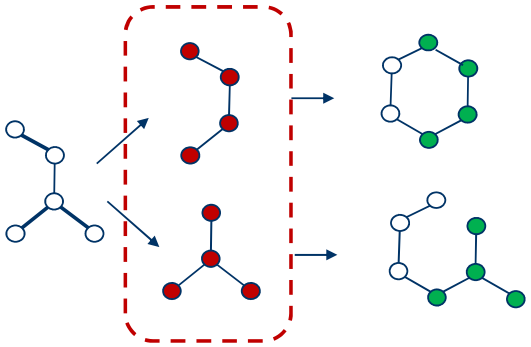


Aplicaciones: SRI

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.

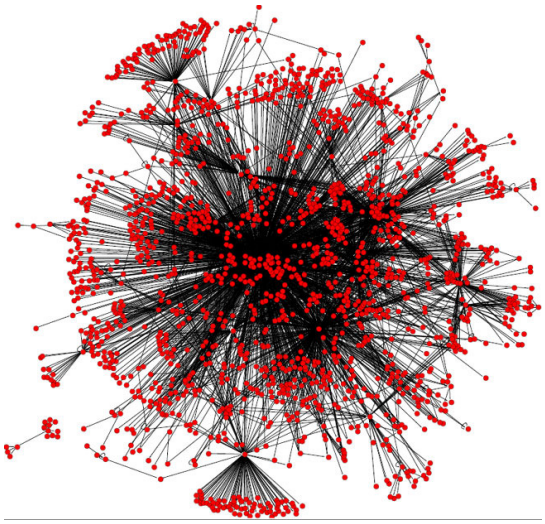
Aplicaciones: PPI

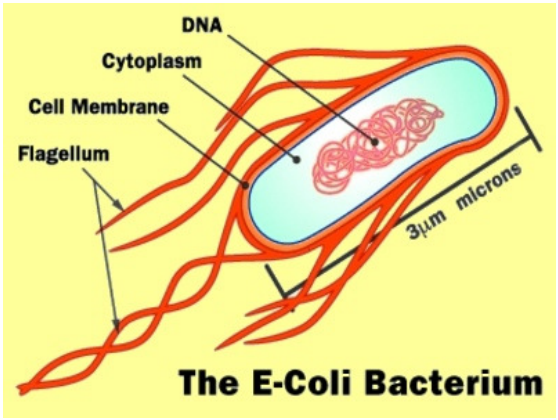
Interacciones de la proteína de la levadura

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

Aplicaciones: PPI

E-coli transcriptional regulatory network

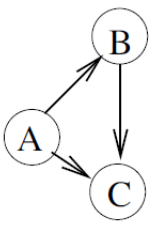




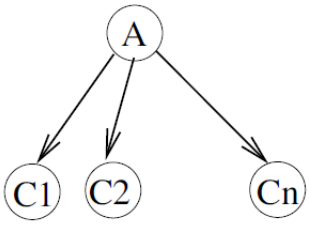
Aplicaciones: PPI

E-coli transcriptional regulatory network

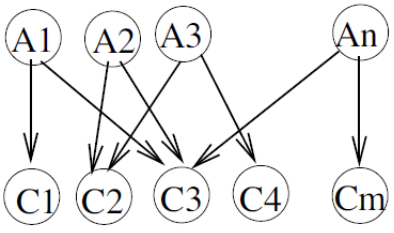
Motifs frecuentes



Feedforward Loop



Single input module (SIM)



Dense overlapping regions (DOR)

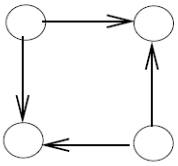
Aplicaciones

Motifs frecuentes en distintos tipos de redes

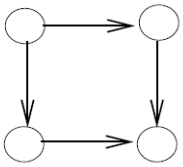
Redes regulatorias de genes

Redes neuronales

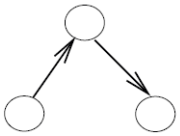
Redes tróficas [food webs]



Bifan



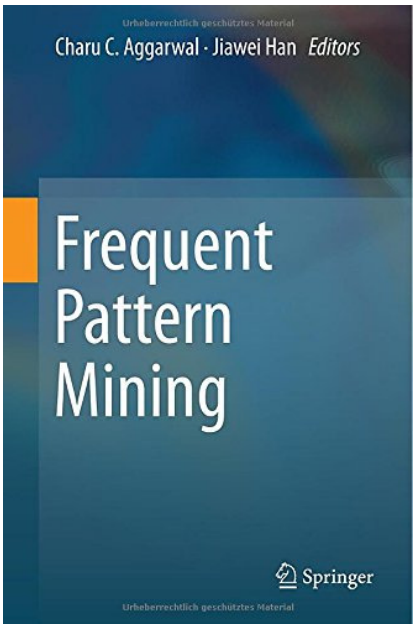
Biparallel



Three-chain

Bibliografía

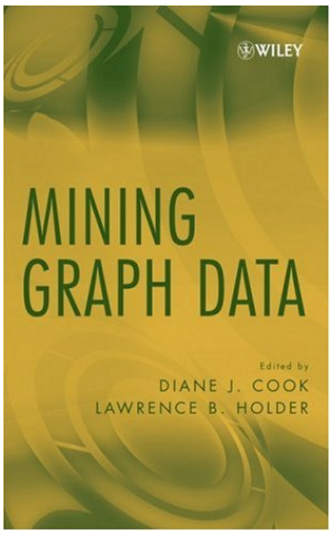
- Charu C. Aggarwal & Jiawei Han (editors): **Frequent Pattern Mining**. Springer, 2014. ISBN 3319078208.



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

Bibliografía

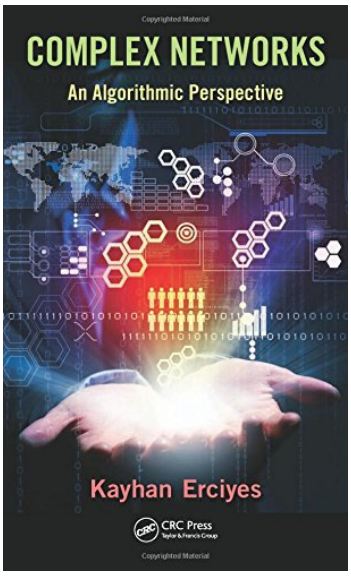
- Diane J. Cooke & Lawrence B. Holder (editors):
  Mining Graph Data.
  Wiley, 2007.
  ISBN 0-471-73190-0



Chapter 5
Discovery of frequent substructures
Xifeng Yang & Jiawei Han

Bibliografía

- 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: Graph Isomorphism

- J.R. Ullman: **An Algorithm for Subgraph Isomorphism**. Journal of the ACM, 23(1):31-42, 1976.
- B.D. McKay, **Practical Graph Isomorphism**. In Manitoba Conference on Numerical Mathematics and Computing Winnipeg 1980, Congressus Numerantium, 30:45-87, 1981.
- B.D. McKay & A. Piperno: **Practical Graph Isomorphism, II**, J. Symbolic Computation, 60:94-112, 2014. DOI 10.1016/j.jsc.2013.09.003. http://pallini.di.uniroma1.it/
- L. Babai and E. M. Luks. **Canonical Labeling of Graphs**. In Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing, STOC'83, pages 171-183, 1983.
- Eugene M. Luks. **Isomorphism of graphs of bounded valence can be tested in polynomial time**. J. Comput. Syst. Sci., 25(1):42-65, 1982.
- S. Fortin. **The Graph Isomorphism Problem**. Technical Report TR 96-20, July 1996, Dept. of Computer Science, The University of Alberta.
- B.T. Messmer, H. Bunke. **Subgraph isomorphism detection in polynomial time on preprocessed model graphs**. In *Recent Developments in Computer Vision,* pages 373-382, Springer, Berlin, 1996.
- L.P. Cordella, P. Foggia, C. Sansone, M. Vento: **A (sub) graph isomorphism algorithm for matching large graphs**. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(10):1367-1372, 2004.
- R. Battiti and F. Mascia. **An algorithm portfolio for the subgraph isomorphism problem**. In Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics, pages 106-120. Springer, Berlin, 2007.

Bibliografía: Graph Patterns

- C. Borgelt & M. R. Berthold, **Mining molecular fragments: Finding relevant substructures of molecules**, ICDM'2002
- Diane J. Cook & Lawrence B. Holder: **Graph-Based Data Mining**. IEEE Intelligent Systems 15(2):32-41, 2000
- L. Dehaspe, H. Toivonen & R. King. **Finding frequent substructures in chemical compounds**, KDD'1998
- L. Dehaspe & H. Toivonen, **Discovery of frequent Datalog patterns**, DMKD'1999
- E. Gudes, S. E. Shimony & N. Vanetik: **Discovering Frequent Graph Patterns Using Disjoint Paths**. IEEE TKDE 18(11):1441-1456, 2006
- L. Holder, D. Cook & S. Djoko. **Substructure discovery in the SUBDUE system**, KDD'1994
- J. Huan, W. Wang & J. Prins. **Efficient mining of frequent subgraph in the presence of isomorphism**, ICDM'2003
- J. Huan, W. Wang & J. Prins, **SPIN: Mining Maximal Frequent Subgraphs from Graph Databases**. KDD'2004
- A. Inokuchi, T. Washio & H. Motoda. **An Apriori-based algorithm for mining frequent substructures from graph data**, PKDD'2000
- A. Inokuchi, T. Washio & H. Motoda. **Complete Mining of Frequent Patterns from Graphs: Mining Graph Data**. Machine Learning, 2003.

Bibliografía: Graph Patterns

- M. Kuramochi and G. Karypis. **Frequent subgraph discovery**, ICDM'2001
- M. Kuramochi and G. Karypis, **GREW: A Scalable Frequent Subgraph Discovery Algorithm**, ICDM'2004
- M. Kuramochi and G. Karypis, **An Efficient Algorithm for Discovering Frequent Subgraphs**. IEEE TKDE 16(9):1038-1051, 2004
- M. Kuramochi and G. Karypis, **Finding Frequent Patterns in a Large Sparse Graph**, Data Mining and Knowledge Discovery, 11(3):243-271, 2005
- S. Nijssen and J. Kok. **A quickstart in frequent structure mining can make a difference**. KDD'2004
- N. Vanetik, E. Gudes, and S. E. Shimony. **Computing frequent graph patterns from semistructured data**, ICDM'2002
- N. Vanetik, E. Gudes. **Mining Frequent Labeled and Partially Labeled Graph Patterns**. ICDE'2004
- C. Wang, W. Wang, J. Pei, Y. Zhu, and B. Shi. **Scalable mining of large disk-base graph**
- **databases**. KDD'2004.
- T.Washio & H. Motoda: **State of the art of graph-based data mining**. SIGKDD Explorations, 5:59–68, 2003.
- X. Yan and J. Han, **gSpan: Graph-Based Substructure Pattern Mining**, ICDM'2002
- X. Yan and J. Han**, CloseGraph: Mining Closed Frequent Graph Patterns**, KDD'2003
- X. Yan, X. J. Zhou, and J. Han, **Mining Closed Relational Graphs with Connectivity Constraints**, KDD'2005

Bibliografía: Graph Patterns

- M. Al Hasan,V. Chaoji, S. Salem, J. Besson, and M. J. Zaki. **ORIGAMI: Mining representative orthogonal graph patterns**. ICDM'2007
- T. Asai, K. Abe, S. Kawasoe, H. Arimura, H. Satamoto, and S. Arikawa. **Efficient substructure discovery from large semi-structured data**. SDM'2002.
- B. Bringmann and S. Nijssen. **What is frequent in a single graph?** PAKDD'2008.
- M. Fiedler and C. Borgelt. **Support computation for mining frequent subgraphs in a single graph**. MLG'2007.
- H. Cheng, X.Yan, J. Han, and C.-W. Hsu. **Discriminative frequent pattern analysis for effective classification**. ICDE'2007.
- J. Cheng,Y. Ke, A. Fu, J. X.Yu, and L. Zhu. **Finding maximal cliques in massive networks by H*-graph**. SIGMOD'2010.
- J. Cheng, Y. Ke, S. Chu, and M. T. Ozsu. **Efficient core decomposition in massive networks**. ICDE'2011.
- J. Cheng, L. Zhu, Y. Ke, and S. Chu. **Fast algorithms for Maximal Clique Enumeration with Limited Memory**. KDD'2012.
- D. Gibson, R. Kumar, and A. Tomkins. **Discovering large dense subgraphs in massive graphs**. VLDB'2005.
- Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, Andrew Tomkins: **Trawling the Web for emerging cyber-communities**. WWW'1999 & Computer Networks '1999

Bibliografía: Graph Patterns

- H. He and A. K. Singh. **Efficient algorithms for mining significant substructures in graphs with quality guarantees**. ICDM'2007.
- R. Jin, C.Wang, D. Polshakov, S. Parthasarathy, and G. Agrawal. **Discovering frequent topological structures from graph datasets**. KDD'2005.
- T. Kudo, E. Maeda, and Y. Matsumoto. **An application of boosting to graph classification**. In Advances in Neural Information Processing Systems 18 (NIPS'04), 2004.
- S. Ranu and A. K. Singh. **GraphSig: A scalable approach to mining significant subgraphs in large graph databases**. ICDE'2009.
- H. Saigo, N. Krämer, and K. Tsuda. **Partial least squares regression for graph mining**. KDD'2008.
- L. Thomas, S. Valluri, and K. Karlapalem. **MARGIN: Maximal frequent subgraph mining**. ICDM'2006
- C. E. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. A. Tsiarli. **Denser than the densest subgraph: Extracting optimal quasi-cliques with quality guarantees**. KDD'2013.
- K. Tsuda. **Entire regularization paths for graph data**. ICDML'2007.
- J.Wang and J. Cheng. **Truss decomposition in massive networks**. PVLDB, 5(9):812–823, 2012.
- N. Wang, J. Zhang, K. L. Tan, A. K. H. Tung. **On Triangulation-based Dense Neighborhood Graphs Discovery**. PVLDB, 4(2):58–68, 2010.
- J. Wang, J. Cheng, and A. Fu. **Redundancy-aware maximal cliques**. KDD'2013.

Bibliografía: Graph Patterns

- J. Xiang, C. Guo, and A.Aboulnaga. **Scalable maximum clique computation using MapReduce**. ICDE'2013.
- X. Yan, H. Cheng, J. Han, and P. S. Yu. **Mining significant graph patterns by scalable leap search**. SIDMOD'2008.
- Y. Zhang and S. Parthasarathy. **Extracting analyzing and visualizing triangle k-core motifs within networks**. ICDE'2012.
- M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis. **Frequent substructure-based approaches for classifying chemical compounds**. IEEE TKDE 17:1036–1050, 2005.
- Rosalba Giugno and Dennis Shasha. **GraphGrep: A Fast and Universal Method for Substructure Searches**. ICPR'2002.
- J. Huan, W. Wang, D. Bandyopadhyay, J. Snoeyink, J. Prins, and A. Tropsha. **Mining spatial motifs from protein structure graphs**. RECOMB'2004
- M. Koyuturk, A. Grama, and W. Szpankowski. **An efficient algorithm for detecting frequent subgraphs in biological networks**. Bioinformatics, 20:I200–I207, 2004.
- X. Yan, P. S. Yu, and J. Han. **Graph indexing: A frequent structure-based approach**. SIGMOD'2004.
- X. Yan, P. S. Yu, and J. Han, **Substructure Similarity Search in Graph Databases**, SIGMOD'2005
- Xifeng Yan, Feida Zhu, Philip S. Yu & Jiawei Han: **Feature-based Similarity Search in Graph Structures**. ACM Transactions on Database Systems, December 2006.
- Hanghang Tong, Brian Gallagher, Christos Faloutsos & Tina Eliassi-Rad: **Fast Best-Effort Pattern Matching in Large Attributed Graphs**. KDD'2007

Bibliografía: Motif Discovery

Mfinder

http://www.weizmann.ac.il/mcb/UriAlon/index.html

- N. Kashtan, S. Itzkovitz, R. Milo & U. Alon. **Mfinder tool guide**. Technical Report, Department of Molecular Cell Biology and Computer Science and Applied Mathematics, Weizmann Institute of Science, 2002.
- R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii & U. Alon. **Network motifs: simple building blocks of complex networks**. *Science,* 298(5594):824-827, 2002.
- N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. **Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs**. *Bioinformatics,* 20:1746-1758, 2004.

FANMOD (ESU algorithm)

http://theinf1.informatik.uni-jena.de/motifs/

- S. Wernicke: **Efficient Detection of Network Motifs**. IEEE/ACM Trans. Comput. Biology Bioinform. 3(4):347-359, 2006
- S. Wernicke: **A Faster Algorithm for Detecting Network Motifs**. In Proceedings of the 5th WABI-05, Volume 3692, pages 165-177, Springer, 2005.
- S.Wernicke & F. Rasche: **FANMOD: A tool for fast network motif detection**. Bioinformatics, 22(9):1152-1153, 2006.

Bibliografía: Motif Discovery

Otros algoritmos

- S. Omidi, F. Schreiber, A. Masoudi-Nejad. **MODA: an efficient algorithm for network motif discovery in biological networks**. *Genes and Genetic Systems,* 84:385-395, 2009.
- J. Grochow & M. Kellis: **Network motif discovery using subgraph enumeration and symmetry-breaking**. Proceedings of the 11th annual international Conference on Research in Computational Molecular Biology, RECOMB'07, pages 92-106, 2007.
- Z.R. Kashani, H. Ahrabian, E. Elahi, A. Nowzari-Dalini, E.S. Ansari, S. Asadi, S. Mohammadi, F. Schreiber & A. Masoudi-Nejad. **Kavosh: a new algorithm for finding network motifs**. BMC Bioinformatics, 10(318), 2009.

FPF

- V. Spirin & L.A. Mirny: **Protein complexes and functional modules in molecular networks**. PNAS, 100(21):12123-12128, 2003

NeMoFinder

- J. Chen, W. Hsu, M.L. Lee & S-K Ng: **NeMoFinder: dissecting genome-wide protein-protein interactions with meso-scale network motifs**. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 106-115, 2006.