TEMA 3: Redes y grafos, algoritmos

Lourdes Araujo y Juan Martinez-Romo
Dpto. Lenguajes y Sistemas Informáticos
UNED

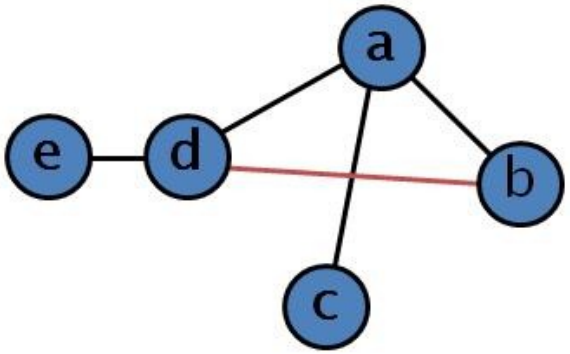
Representación de grafos

- Matrices de adyacencia
- Listas enlazadas

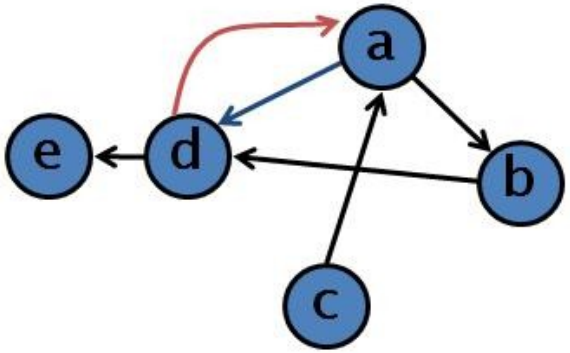
Matriz de adyacencia (1)

- A_{ij} = 1 si hay un enlace del nodo i al nodo j
- A_{ij} = 0 en otro caso
- Son simétricas en los grafos no dirigidos
- A_{ij} = peso del enlace en los grafos con pesos

Matriz de adyacencia (2)



	a	b	c	d	e
a	0	1	1	1	0
b	1	0	0	1	0
c	1	0	0	0	0
d	1	1	0	0	1
e	0	0	0	1	0



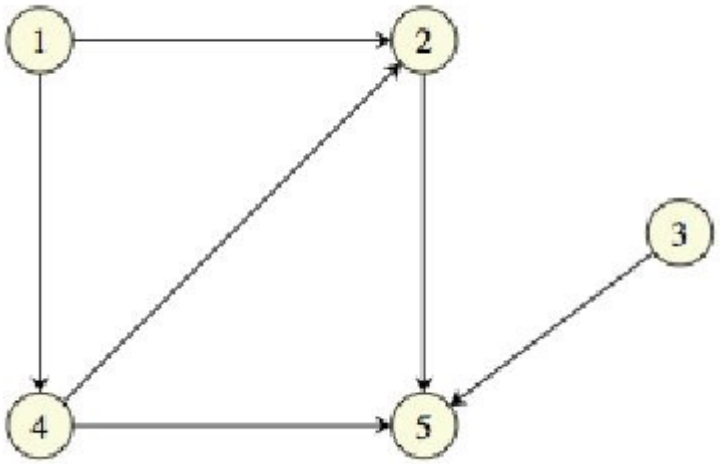
	a	b	c	d	e
a	0	1	0	1	0
b	0	0	0	1	0
c	1	0	0	0	0
d	1	0	0	0	1
e	0	0	0	0	0

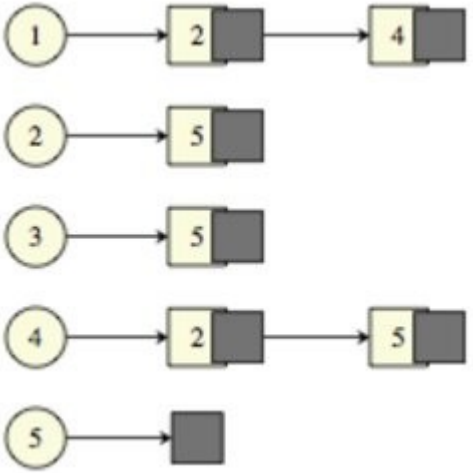
Matrices dispersas

- Muchas redes → grafos dispersos (Número de enlaces mucho más pequeño que el número de enlaces máximo o Grado medio mucho más pequeño que N-1)(Leskovec et al., 2009):
  - WWW(Berkeley): N=319,717 ‹k›=9.65
  - Red social (LinkedIn): N=6,946,668 ‹k›=8.87
  - Comun. (MSN IM): N=242,720,596 ‹k›=11.1
  - Coautoria (DBLP): N=317,080 ‹k›=6.62
  - Calles (California): N=1,957,027 ‹k›=2.82
- La mayor parte de la matriz de adyacencia son ceros
- Algoritmos eficientes para matrices dispersas.

Listas de adyacencia

- Representación alternativa





Tamaño de las redes

- Varian en órdenes de magnitud.
- Ejemplos reales descritos en artículos científicos:
  - Red de intercambio de emails en un instituto de investigación: 436 nodos
  - Red de intercambio de emails en una universidad: 43553 nodos
  - Red de amistades declaradas en una comunidad de blogs: 4.4 millones de nodos

Tamaño de las redes

- Ejemplos reales descritos en artículos científicos (cont.):
  - Red de comunicación en Microsoft messenger: 240 millones de nodos
  - Red de Facebook: 800 millones de nodos

Tamaño de las redes

- Las redes pequeñas (decenas de vértices) puede representarse graficamente y analizarse por muchos algoritmos
- Las redes medianas (cientos de v.) aún pueden visualizarse pero no todos los algoritmos son aplicables.
- Las redes grandes no pueden visualizarse en detalle y requieren algoritmos especiales que tengan en cuenta la dispersión de los enlaces.

Cálculo de las propiedades de los grafos a partir de la matriz de adyacencia

- Número de enlaces E

$$E = \frac{1}{2} \sum_{i,\, j=1}^{N} A_{ij}$$

- Grado medio

$$\bar{K} = 2\frac{E}{N}$$

Número de caminos H entre nodos u y v

- Long h=1: Si hay un enlace entre u y v $A_{uv} = 1$ sino $A_{uv} = 0$
- Long h=2: Si hay un camino de longitud 2 entre u y v ent, $A_{uk}A_{kv} = 1$ sino $A_{uk}A_{kv} = 0$

$$H_{uv}^{(2)} = \sum_{k=1}^{N} A_{uk}\, A_{kv} = [A^2]_{uv}$$

- Long h: Si hay un camino de long. h entre u y v ent. $A_{uk} \ldots A_{kv}=1$ sino $A_{uk} \ldots A_{kv}=0$

$$H_{uv}^{(h)} = [A^h]_{uv}$$

Algoritmos para grafos

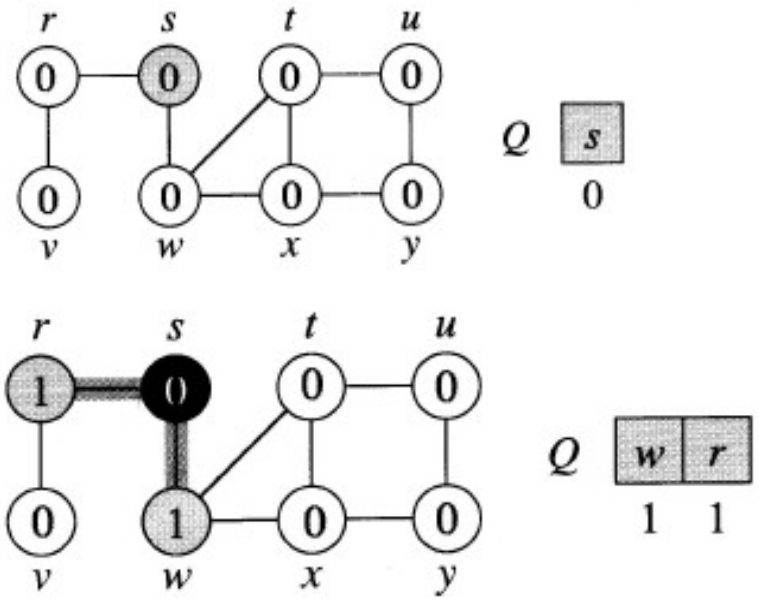
- Recorrido en anchura
- Recorrido por profundidad
- Árboles de recubrimento: alg. de Prim y Kruskal
- Búsqueda de camino mánimo: alg de Dijkstra y Floyd

Recorrido en anchura

- El recorrido por niveles o en anchura (breadth-first search - BF), basa el orden de visita de los nodos del grafo en una E.D. Cola, incorporándole en cada paso los adyacentes al nodo actual
- Esto implica que se visitarán todos los hijos de un nodo antes de proceder con sus demás descendientes
- Puede aplicarse tanto a grafos dirigidos y como a no dirigidos

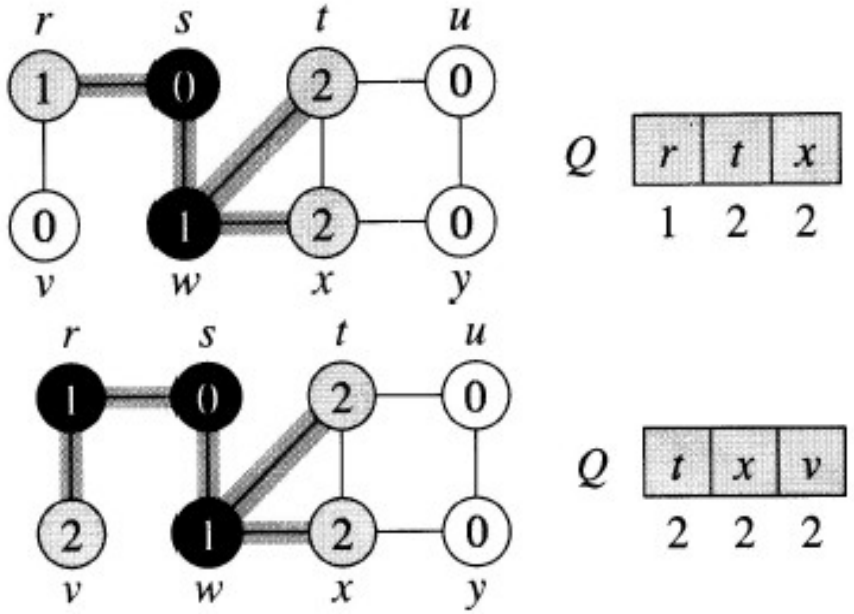
Recorrido en anchura

- Ejemplo de recorrido en anchura
  - s (nodo inicial)



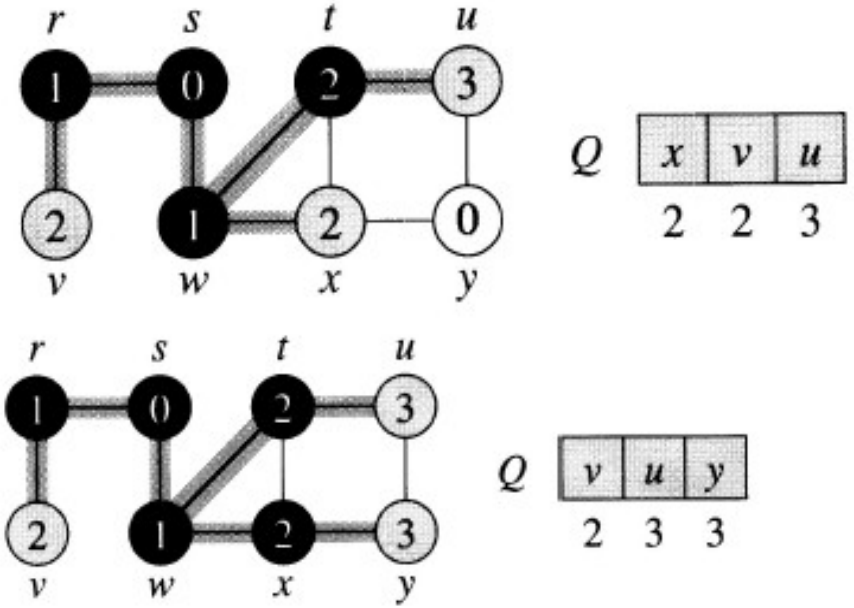
Recorrido en anchura

- Ejemplo de recorrido en anchura



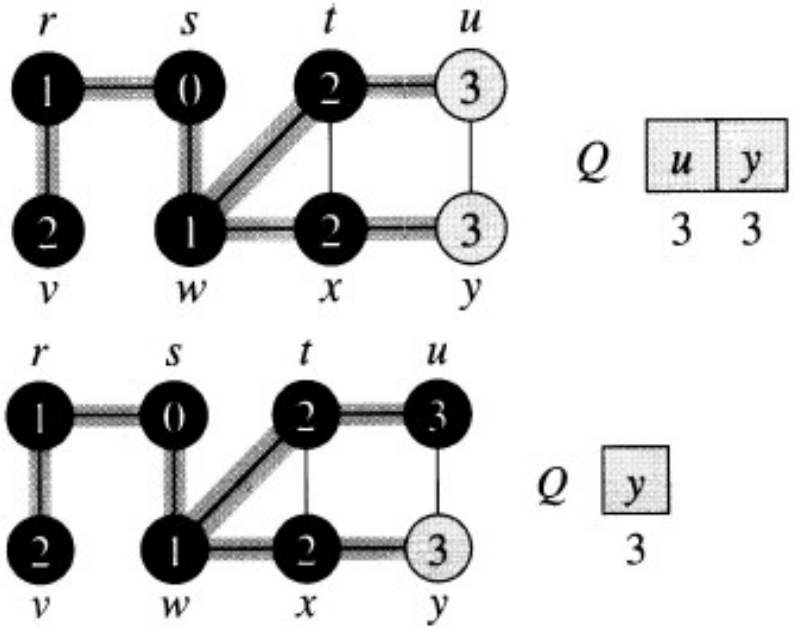
Recorrido en anchura

- Ejemplo de recorrido en anchura



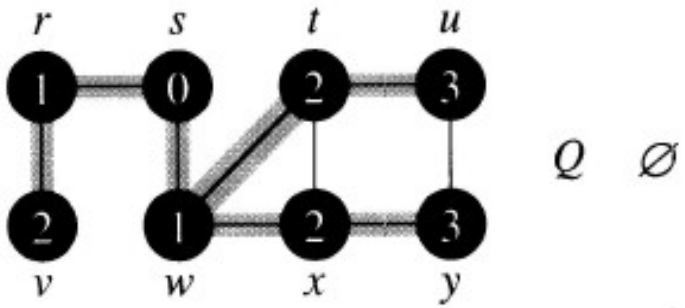
Recorrido en anchura

- Ejemplo de recorrido en anchura



Recorrido en anchura

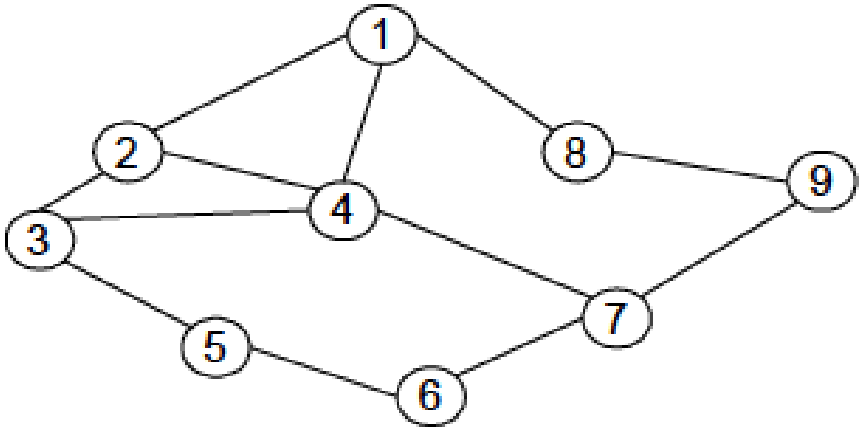
- Ejemplo de recorrido en anchura

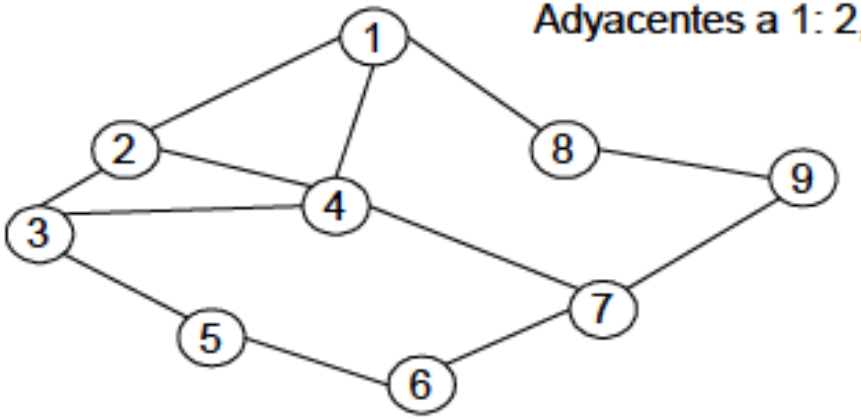


Recorrido por profundidad

- El recorrido por profundidad, o depth-first search (DF), basa el orden de visita de los nodos del grafo en una E.D. Pila, agregando en cada paso los adyacentes al nodo actual
- Esto hace que agote los nodos accesibles desde un hijo antes de proceder con sus hermanos
- Al igual que el recorrido por anchura, se presenta un algoritmo no determinista que puede modificarse para incluir un orden en la elección de los nodos
- Puede aplicarse tanto a grafos dirigidos como a grafos no dirigidos

Recorrido por Profundidad

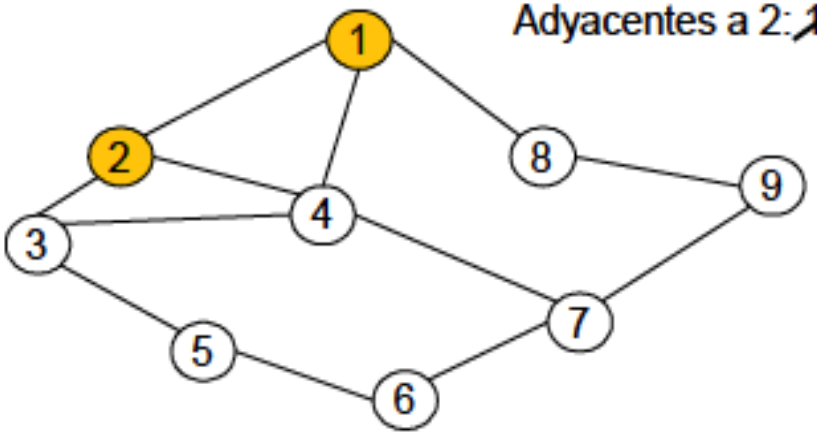




Adyacentes a 1: 2, 4, 8

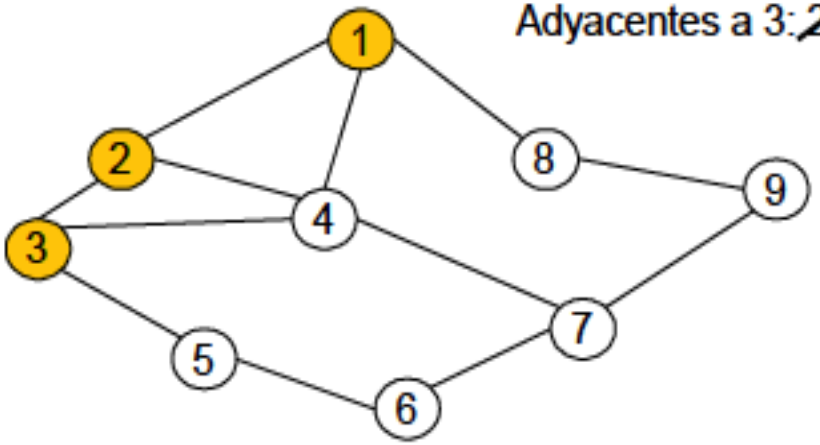
Recorrido: 1, 2

Recorrido por Profundidad



Adyacentes a 2: ~~1~~, 3, 4

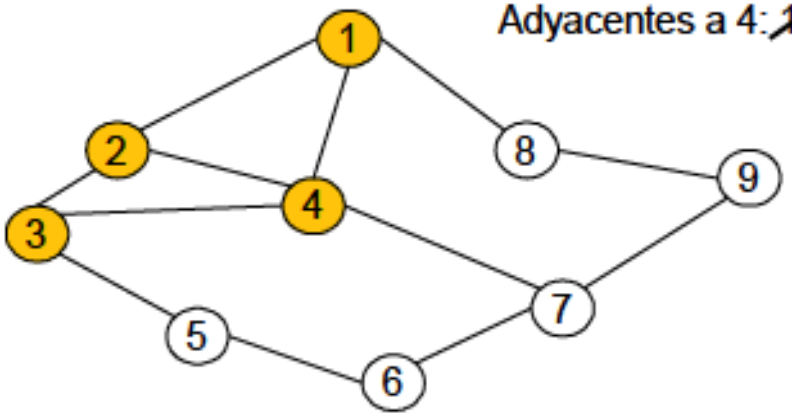
Recorrido: 1, 2, 3



Adyacentes a 3: ~~2~~, 4, 5

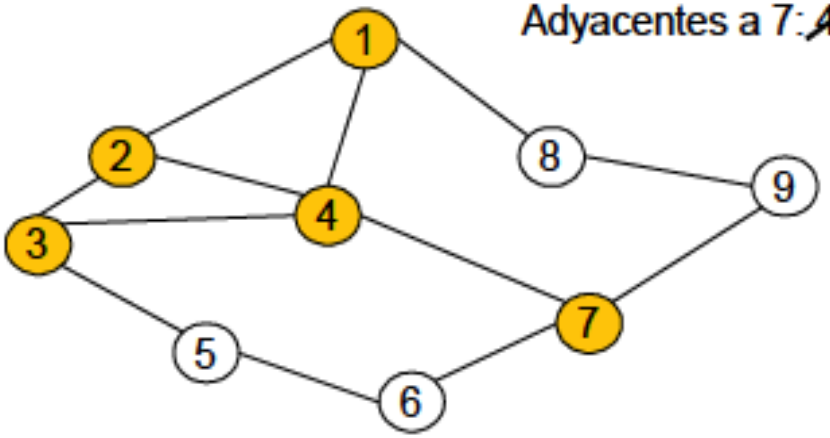
Recorrido: 1, 2, 3, 4

Recorrido por Profundidad



Adyacentes a 4: ~~1~~, ~~2~~, ~~3~~, 7

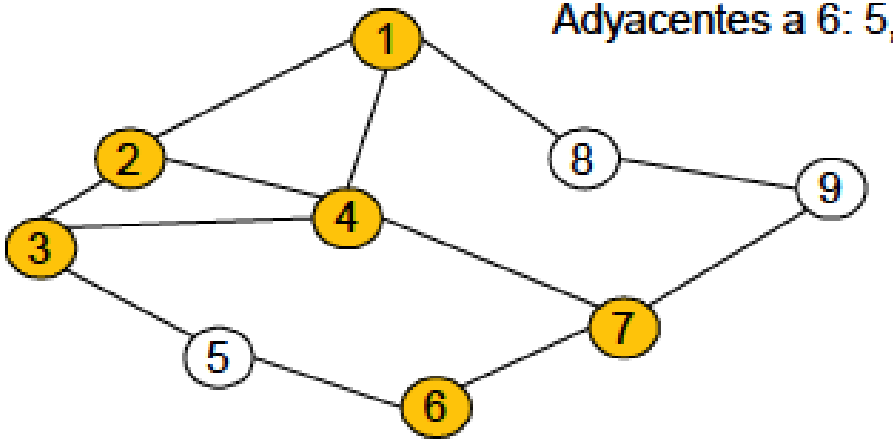
Recorrido: 1, 2, 3, 4 , 7



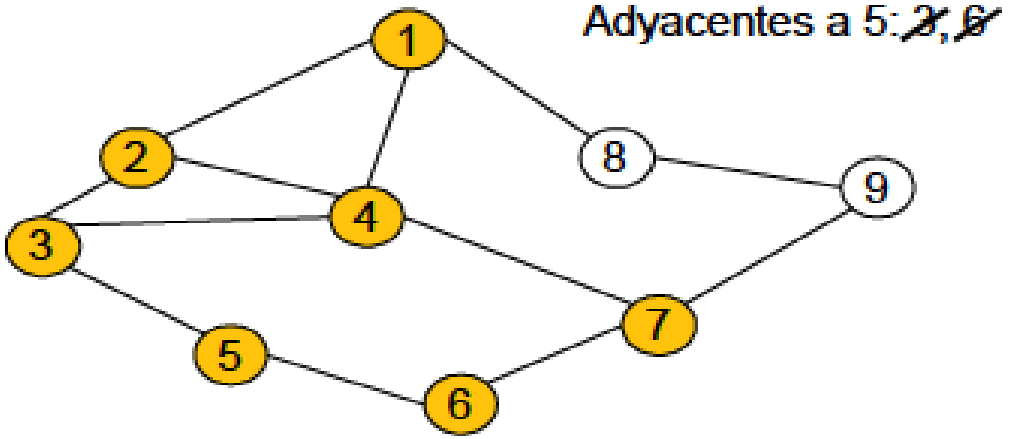
Adyacentes a 7: ~~4~~, 6, 9

Recorrido: 1, 2, 3, 4, 7, 6

Recorrido por Profundidad

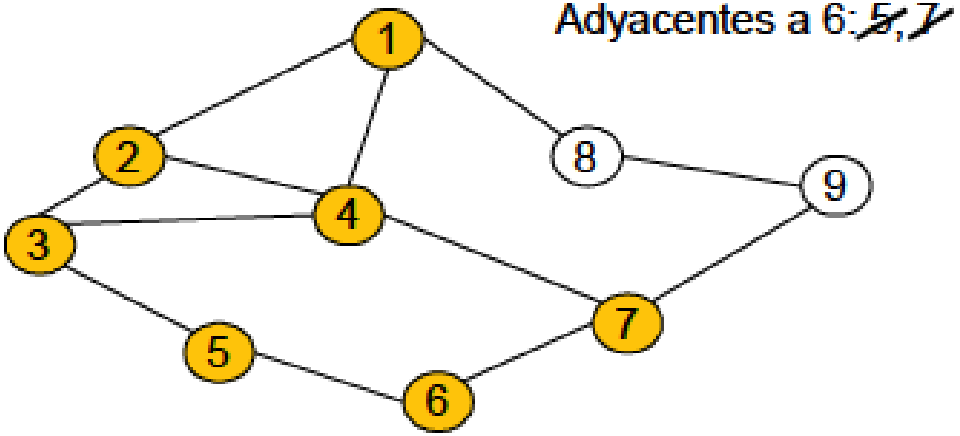


Recorrido: 1, 2, 3, 4, 7, 6, 5

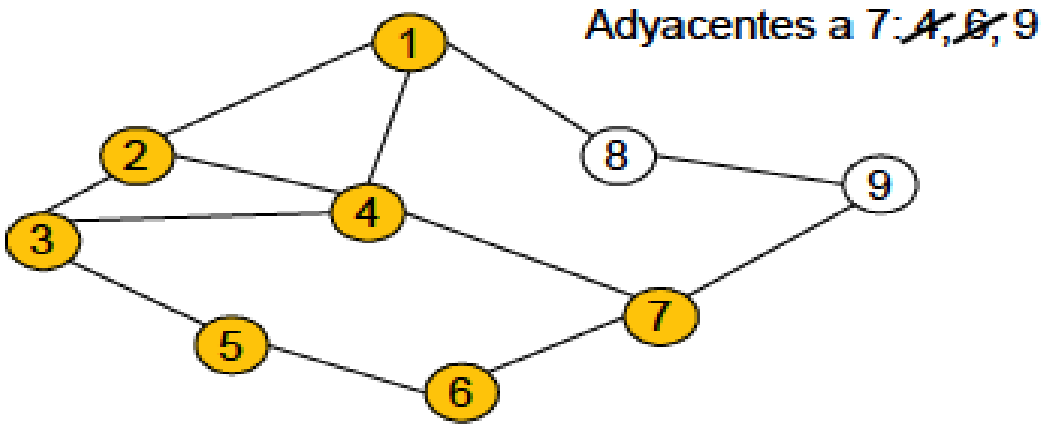


Recorrido: 1, 2, 3, 4, 7, 6, 5

Recorrido por Profundidad

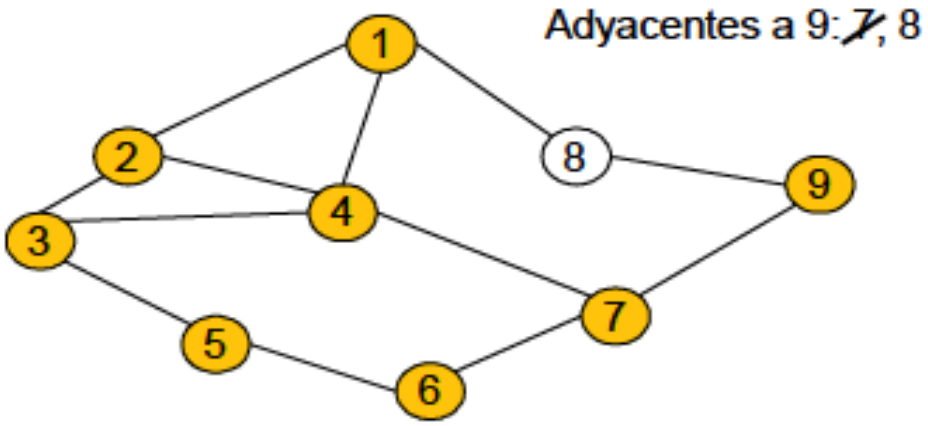


Recorrido: 1, 2, 3, 4, 7, 6, 5



Recorrido: 1, 2, 3, 4, 7, 6, 5 , 9

Recorrido por Profundidad



Recorrido: 1, 2, 3, 4, 7, 6, 5, 9 , 8

Recorrido por Profundidad

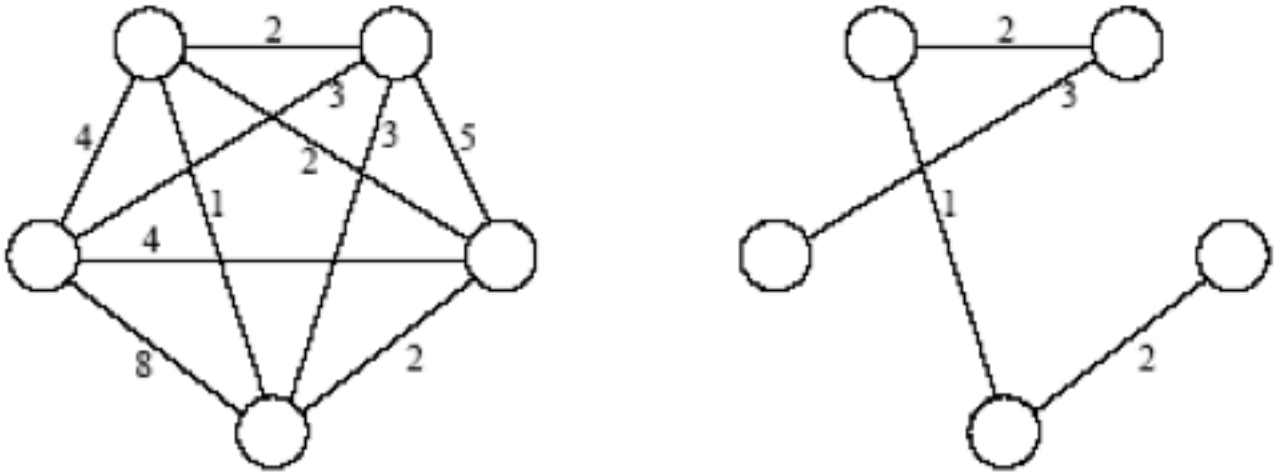
```
procedimiento recorrido_en_profundidad (G: grafo)
   para cada v ∈ N hacer
    marca[v] ← no visitado
   fpara
   para cada v ∈ N hacer
    si marca[v] ≠ visitado entonces
        rp(G, marca, v)
    fsi
   fpara
fprocedimiento

procedimiento rp (G: grafo; marca:vector[1..n]; v: nodo)
{El nodo v no ha sido visitado anteriormente}
   marca[v] ← visitado
   para cada nodo adyacente a v hacer
    si marca[w] ≠ visitado entonces
        rp(G, marca, w)
    fsi
   fpara
fprocedimiento
```

Arbol de Expansión Mínimo

- Un *spanning tree* de un grafo no dirigido *G* es un subgrafo de *G* que es un árbol que contiene todos los vértices de *G*.
- En un grafo ponderado, el peso de un subgrafo es la suma de los pesos de las aristas del subgrafo.
- Un *minimum spanning tree* (MST) sería el spanning tree con peso mínimo.

Arbol de Expansión Mínimo

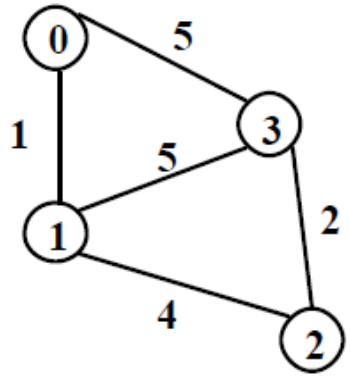
- Un grafo no dirigido y su arbol de expansión mínimo (MST)

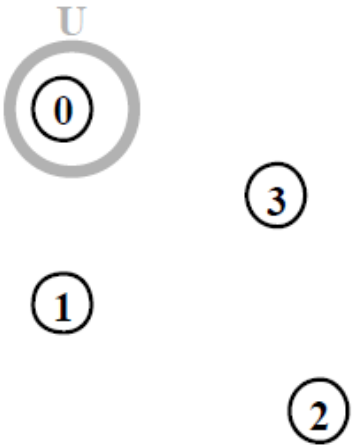


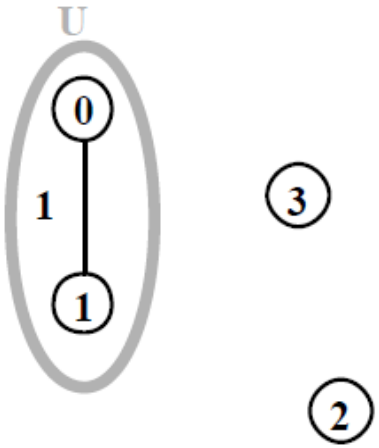
Algoritmo de Prim

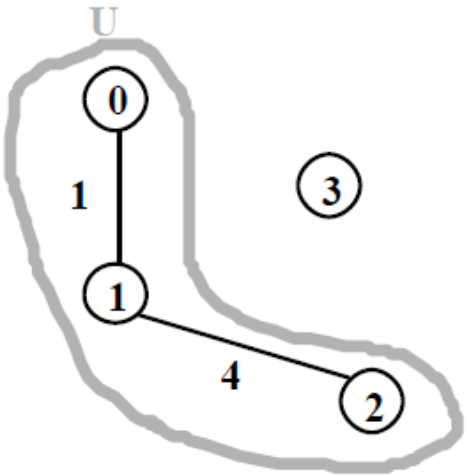
- El algoritmo de Prim es tal vez el algoritmo de Minimum Spanning Tree MST (arbol de expansión mínimo) más sencillo de implementar y el mejor método para grafos densos.
- Este algoritmo puede encontrar el MST de cualquier grafo conexo pesado.
- El siguiente ejemplo ilustra el funcionamiento del algoritmo. La secuencia de ilustraciones va de izquierda a derecha y de arriba hacia abajo.
- La primera imagen muestra el grafo pesado y las siguientes muestran el funcionamiento del algoritmo de Prim y como va cambiando el conjunto U durante la ejecución.

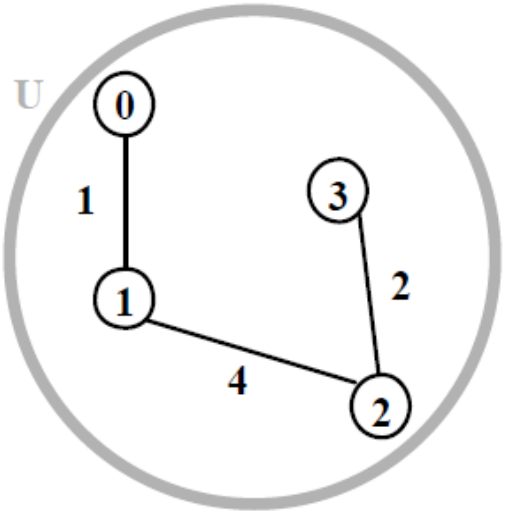
Algoritmo de Prim











Algoritmo de Prim

- Video de Ejemplo
  - http://www.youtube.com/watch?v=O8XEOz8FCDQ

Algoritmo de Dijkstra

- El algoritmo de Dijkstra resuelve el problema de encontrar los caminos más cortos a partir de un origen, en grafos pesados que no tengan pesos negativos.
- El algoritmo de Dijkstra es un algoritmo voraz que opera a partir de un conjunto S de nodos cuya distancia más corta desde el origen ya es conocida.
- En principio, S contiene sólo el nodo origen.
- En cada paso, se agrega algún nodo v a S, cuya distancia desde el origen es la más corta posible.

Algoritmo de Dijkstra

- Pseudocódigo del algoritmo de Dijkstra
  - Uso de colas de prioridad

```
DIJKSTRA (Grafo G, nodo_fuente s)
    para u ∈ V[G] hacer
        distancia[u] = INFINITO
        padre[u] = NULL
    distancia[s] = 0
    añadir (cola, (s,distancia[s]))
    mientras que cola no es vacía hacer
        u = extraer_minimo(cola)
        para todos v ∈ adyacencia[u] hacer
            si distancia[v] > distancia[u] + peso (u, v) hacer
                distancia[v] = distancia[u] + peso (u, v)
                padre[v] = u
                añadir(cola,(v,distancia[v]))
```

Algoritmo de Dijkstra

- Video de ejemplo
  - http://www.youtube.com/watch?v=LLx0QVMZVkk

Otros algoritmos

- Partición de grafos
- Algoritmos de Comunidades
- Ramdom walks

Partición de grafos

- La partición de grafos consiste en dividir un grafo en subgrafos (cluster) del tal forma que los nodos de cada subgrafo tengan una relación “más fuerte” entre los nodos del mismo subgrafo que con los demás vértices del resto del grafo.
- Las medidas del grado de relación pueden ser de multiple naturaleza como por ejemplo el peso de los enlaces que los unen.
- No hay una definición única de clúster en un grafo

Partición de Grafos

- Definición de mediciones de similaridad
  - Hay dos enfoques principales para la identificación de un clúster:
    - Calculando algunos valores de los nodos y clasificar los nodos en grupos sobre la base de los valores obtenidos (algoritmos locales)
    - Calcular una medida de la idoneidad en el conjunto de posibles clústeres y luego elegir entre el conjunto de clúster candidatos que optimizan la medida utilizada (algoritmos globales)

Partición de Grafos

- Tipos de Agrupamiento
  - Jerárquico (Hierarchical) : dendrogramas, Grafos (Arboles)
  - De partición: División en grupos (SOM, LVQ, etc.)

Partición de Grafos

- Dendogramas:
    1. El primer paso es calcular las distancias entre todos los pares de objetos. Esto es lo mismo que asumir que cada objeto constituye un cluster: $\{C_1, ..., C_N\}$.
    2. Se buscan los dos clusters más cercanos (C_i, C_j), éstos se juntan y constituyen uno solo C_{ij}.
    3. Se repite el paso 2 hasta que no quedan pares de comparación.

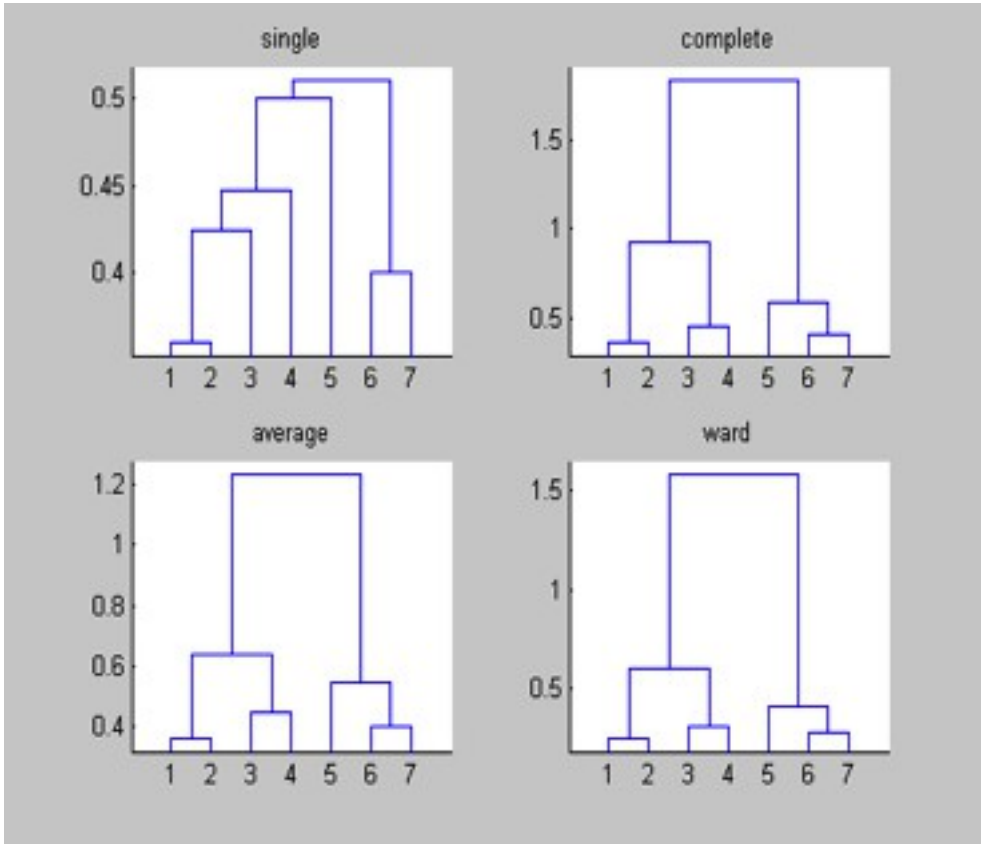
En general se representan como árboles binarios.

Partición de Grafos

- Dendrogramas: Ejemplo en matlab

```
Datos=[ 0.8 1.8;...
        1.1 1.6;...
        0.8 1.3;...
        1.0 0.9;...
        1.4 0.6;...
        1.5 0.1;...
        1.1 0.1];
```

En formato Matlab



Partición de Grafos

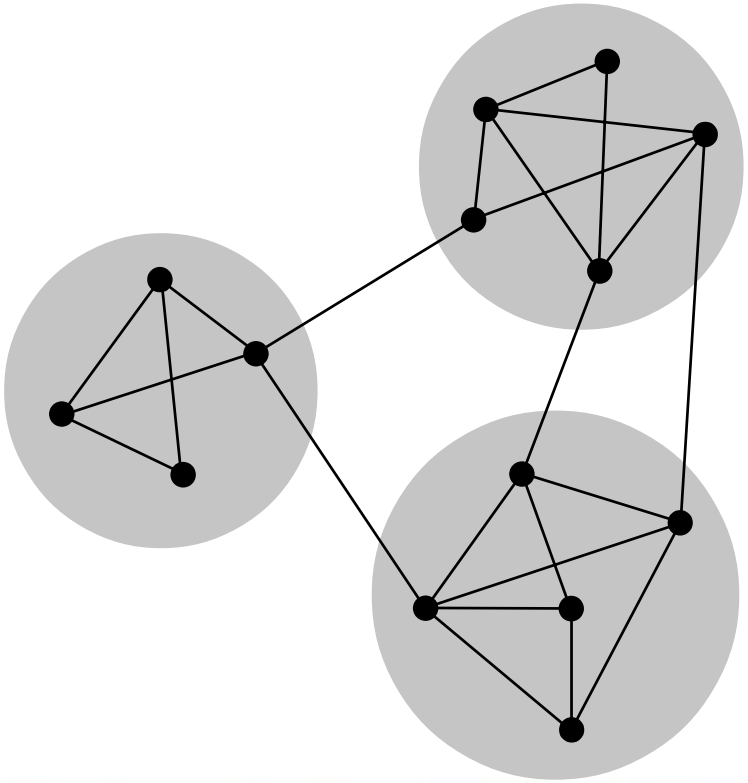
- El algoritmo de las K-medias es otro algoritmo de partición.
  - Toma el parámetro K como entrada.
  - Es un método de agrupamiento, que tiene como objetivo la partición de un conjunto n en k grupos en el que cada observación pertenece al grupo más cercano a la media.

Partición de Grafos

1. Inicialmente se seleccionan K objetos del conjunto de entrada. Estos K objetos serán los centroides iniciales de los K-grupos.
2. Un centroide podría definirse como el promedio de todos los puntos de un conjunto.
3. Se calculan las **distancias** de los objetos (datos) a cada uno de los centroides. Los datos (objetos) se asignan a aquellos grupos cuya distancia es mínima con respecto a todos los centroides.
4. Se actualizan los centroides como el valor medio de todos los objetos asignados a ese grupo.
5. Se repite el paso 2 y 3 hasta que se satisface algún criterio de convergencia.

Algoritmos de Comunidades

- En la siguiente figura se puede apreciar una representación esquemática de una red con una estructura de comunidades
- Existen tres comunidades con nodos densamente conectados y una mucho menos densidad de conexión entre dichas comunidades



Algoritmos de Comunidades

- Comunidad: Está formada por individuos de tal manera que los que están dentro de un grupo interactúan entre sí con más frecuencia que con los que están fuera del grupo
- Existen comunidades extrínsecas de las que un individuo decide formar parte (seguidores de los Yankees).
- También existen comunidades intrínsecas en las que un individuo debido a sus propiedades forma parte de una comunidad (gusto por la música Jazz).
- Detección de comunidades: focaliza en el descubrimiento de comunidades intrínsecas.

Algoritmos de Comunidades

- Ejemplo: Comunidades en Redes Sociales
- ¿Por qué se generan comunidades en redes sociales?
  - Los seres humanos son sociales
  - Su facilidad de uso permite a las personas ampliar su vida social en una forma sin precedentes
  - Resulta difícil reunirse con amigos en el mundo físico, pero muy fácil encontrar amigos en línea con intereses similares
  - Las interacciones entre los nodos pueden ayudar a determinar las comunidades

Algoritmos de Comunidades

- Taxonomía de los criterios de una comunidad
  - Los métodos de detección de comunidades pueden ser divididos en 4 categorías:
    - **Comunidad centrada en el Nodo**
      - Cada nodo en un grupo satisface ciertas propiedades (e.j. cliqués)
    - **Comunidad centrada en el grupo**
      - Considera las conexiones dentro de un grupo como un conjunto (e.j. densidad)
      - El grupo tiene que satisfacer ciertas propiedades globales
    - **Comunidad centrada en la Red**
      - Partición de la red completa en varios conjuntos disjuntos (e.j. clustering basado en similitud de vértices, clustering espectral)
    - **Comunidad centrada en la Jerarquía**
      - Construcción de una estructura jerárquica de comunidades (e.j. clustering jerárquico aglomerativo)

Random Walks

- Random Walk (“paseo aleatorio”) es un sistema teórico usado en estadística que consiste básicamente en ir tomando decisiones aleatorias en nuestro problema cada vez que se requiera.
- Supongamos un caminante que sale desde un nodo cualquiera.
- Aleatoriamente en cada paso va escogiendo un camino. Y al mismo tiempo, en cada paso, con una probabilidad dada puede volver al origen o a otro nodo destino establecido al inicio.
- El resultado final será una probabilidad para cada nodo de los del grafo de que el andador haya pasado por dicho nodo.
- Uno de los algoritmos más populares que emplean Random walks es el PageRank.

Random Walks

- A nivel matemático la fórmula del “Random Walk With Restart” es la siguiente:
- $$\mathbf{p}^{t+1} = (1 - r)\mathbf{W}\mathbf{p}^{t} + r\mathbf{p}^{0}$$

- Donde p(t+1) es el vector de probabilidades de estar en un nodo determinado en el paso t+1.
- r es la probabilidad de reiniciar el algoritmo.
- W es la matriz de adyacencia del grafo.
- p(t) es el vector de probabilidades en el momento actual.
- Y p(0) es el vector de probabilidades inicial.

Random Walks

- Algunas aplicaciones del camino aleatorio son:
  - En genética de poblaciones, el camino aleatorio describe las propiedades estadísticas de la deriva genética.
  - En física, los caminos aleatorios son utilizados como modelos simplificados del movimiento browniano y difusión tales como el movimiento aleatorio de las moléculas en líquidos y gases.
  - En biología matemática, los caminos aleatorios son utilizados para describir los movimientos individuales de los animales, para apoyar empíricamente los procesos de biodifusión, y en ocasiones para desarrollar la dinámica de poblaciones.
  - En otros campos de las matemáticas, el camino aleatorio se utiliza para calcular las soluciones de la ecuación de Laplace, para estimar la media armónica, y para varias construcciones en el análisis y la combinatoria.
  - En informática, los caminos aleatorios son utilizados para estimar el tamaño de la Web. En la World Wide Web conference-2006, Bar-Yossef et al. publicó sus descubrimientos y algoritmos para lo mismo.
  - En el procesamiento de imágenes, los caminos aleatorios son utilizados para determinar las etiquetas (es decir, "objeto" o "fondo") para asociarlas con cada píxel. Este algoritmo se suele denominar como algoritmo de segmentación del camino aleatorio.

Referencias

- Networks, Crowds, and Markets:
  Reasoning About a Highly Connected World
  David Easley and Jon Kleinberg

  http://www.cs.cornell.edu/home/kleinber/networks-book/
  Capítulo 1