GRÁFICAS

Gráficas

- Muchas aplicaciones computacionales involucran no solo un conjunto de elementos sino también conexiones entre pares de elementos.
- Las relaciones que resultan de estas conexiones nos llevan a preguntas como:
  - ¿Hay forma de llegar de un elemento a otro siguiendo las conexiones?
  - ¿Cuáles y cuántos elementos se pueden alcanzar a partir de un elemento dado?
  - ¿Cuál es la mejor forma de llegar de un elemento a otro?
- Para modelar situaciones como estas se utilizan las gráficas (o grafos).

Definiciones

- Colección **V** de nodos y colección **E** de aristas (edges).
- Cada arista conecta a dos nodos.
- Representamos una arista **$e \in E$** como un subconjunto de dos elementos de **V**: **$e=\{u,v\}$** o **uv** para algún **$u,v \in V$**.
- Las aristas representan una relación **simétrica** entre dos nodos o vértices.
- Cuando queremos representar una relación **asimétrica**: gráfica **dirigida** o digráfica y una arista se representa como **(u,v)** o **$u \rightarrow v$**.

Definiciones

- Nos ocuparemos en general de **gráficas simples**: aquellas que no tienen ciclos o aristas paralelas.
- Gráficas no-simples se llaman también **multigráficas**.
- **V** representará el número de vértices o nodos en una gráfica.
- **E** representará el número de aristas en una gráfica.
- En una gráfica no dirigida tenemos: $0 \leq E \leq \binom{V}{2}$
- En una gráfica dirigida tenemos: $0 \leq E \leq V(V-1)$

Definiciones

- Para cada arista uv en un grafo no dirigido, llamamos a u **vecino** de v y viceversa.
- El **grado** de un nodo es su número de vecinos.
- En gráficas dirigidas tenemos dos tipos de vecinos:
  - Para la arista u→v llamamos a u el **predecesor** de v y a v el **sucesor** de u.
  - El grado de entrada (**in-degree**) de un nodo es el número de predecesores, igual que el número de aristas que entran al nodo.
  - El grado de salida (**out-degree**) de un nodo es el número de sucesores, igual al número de aristas que salen del nodo.
- Una gráfica G'=(V',E') es una **subgráfica** de G=(V,E) si V'⊆ V y E'⊆E.

Definiciones

- Una **caminata** en una gráfica es una secuencia de aristas, donde cada par sucesivo de aristas comparte un vértice.
- Una caminata es un **camino** si visita cada vértice a lo mas una vez.
- Una gráfica no-dirigida está **conectada** si hay una caminata (y por lo tanto un camino) entre cada dos vértices.
- Una gráfica **desconectada** consta de varios componentes, que son sus subgráficas conectadas máximas.
- Dos vértices están en la misma componente si y solo si hay un camino entre ellos.

Más definiciones

- Un **ciclo** es un camino que empieza y termina en el mismo vértice y tiene al menos una arista.
- Una gráfica no dirigida es **acíclica** si ninguna subgráfica es un ciclo.
- Las gráficas acíclicas también se llaman **bósques**.
- Un **árbol** es un componente del bosque.
- Un **árbol generador** de una gráfica G, es una subgráfica que es un árbol y que contiene a cada vértice de G.
- Una gráfica tiene un árbol generador si y solo si está conectada.

Más definiciones

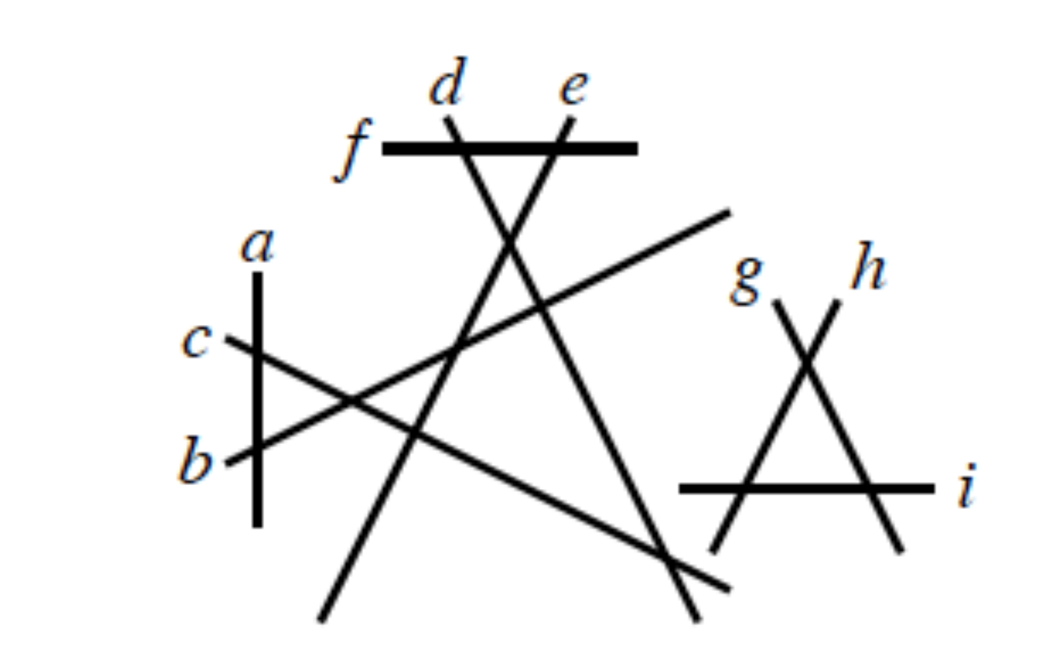
- Una gráfica no dirigida está **conectada** si, para cada par de nodos u y v, hay un camino de u a v.
- Una gráfica dirigida está **fuertemente conectada** si se cumple la misma condición.
- Una gráfica desconectada consta de varios **componentes**.
- Dos vértices están en la misma componente si y solo si hay un camino entre ellos.
- Una gráfica no dirigida es no cíclica si ninguna subgráfica es un ciclo: **bosque**.
- Un **árbol** es una gráfica acíclica conectada.
- Una gráfica acíclica dirigida se llama **dag**.

Representación abstracta

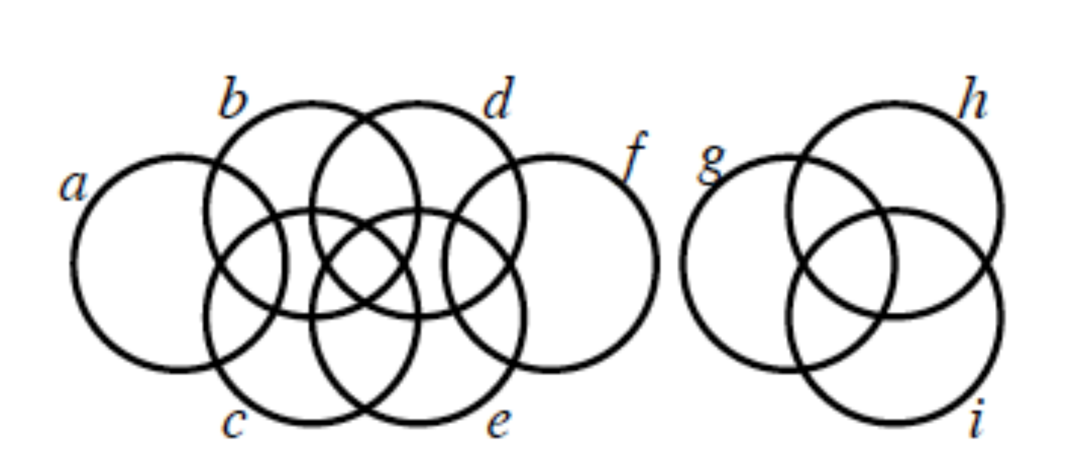
- La forma más común de representar una gráfica se llama **incrustación** (embedding).
- Una incrustación transforma vértices a puntos en el plano y aristas a curvas o segmentos de recta.
- Una curva es **plana** si tiene una incrustación donde no se crucen aristas.
- La misma gráfica puede tener varias incrustaciones.
- No confundir la incrustación con la gráfica misma.
- Gráficas planas pueden tener incrustaciones no planas.

Ejemplo (I)

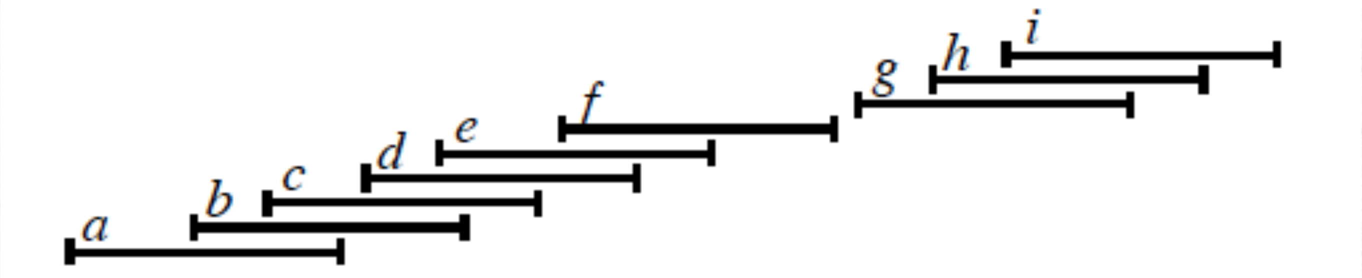
- Intersection graph.



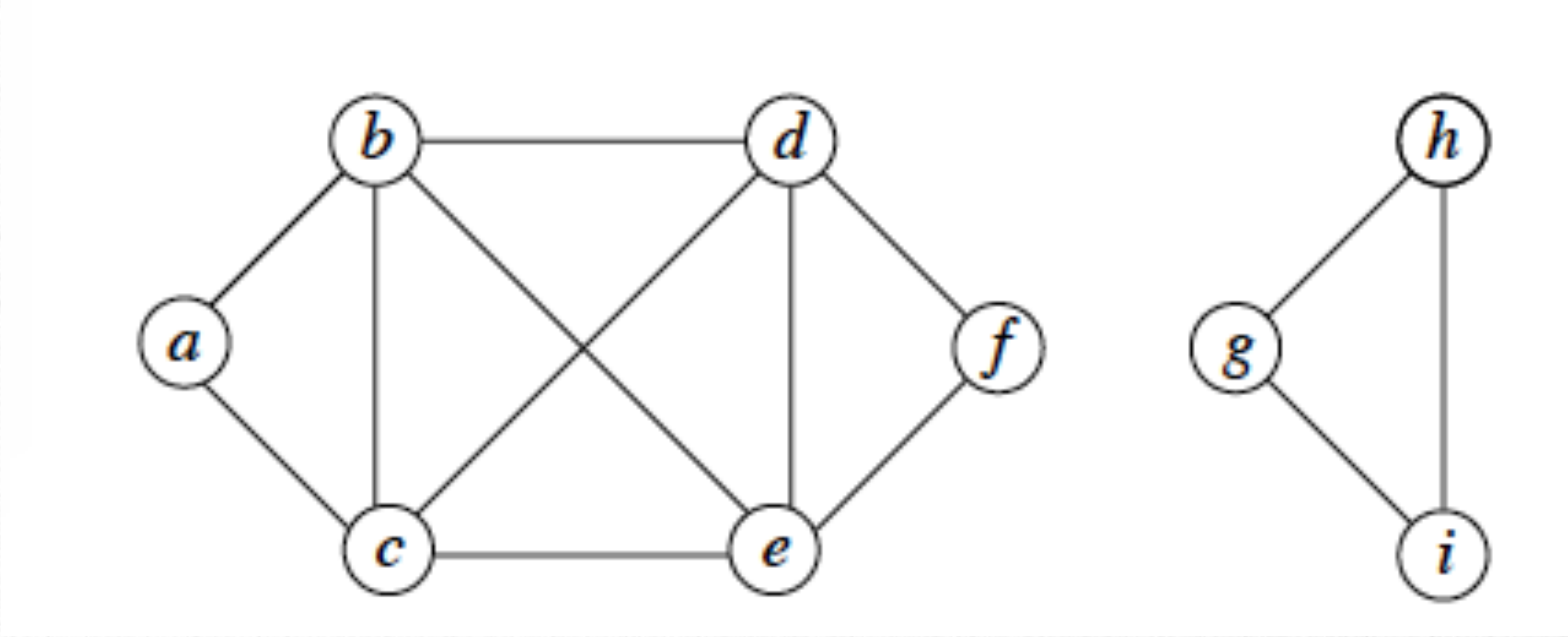
(a)



(b)



(c)

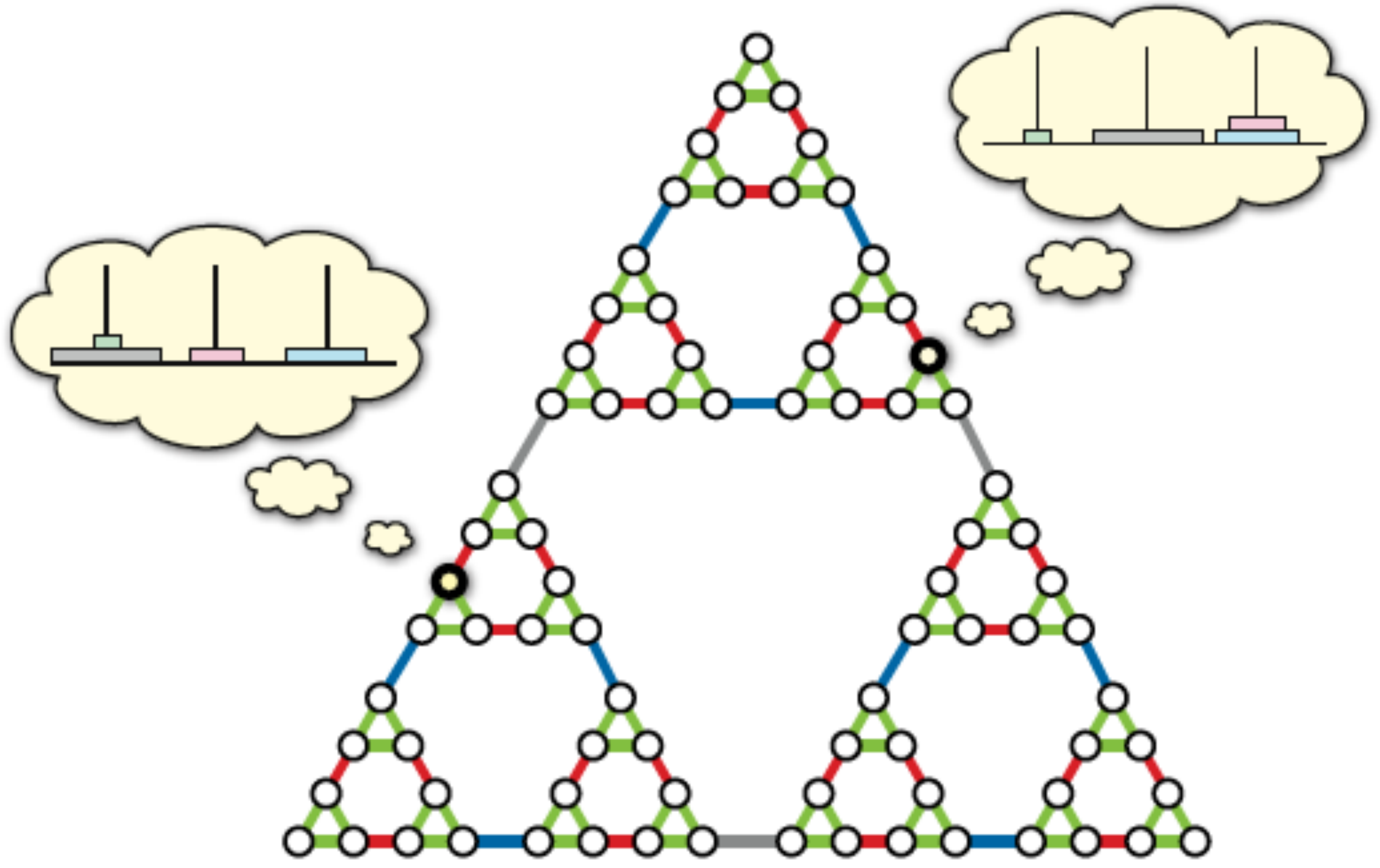


Ejemplo (2)

- Dependency graph.
- Gráfica acíclica dirigida.
- Vértices: subproblemas recursivos distintos.
- Aristas: existe una si es necesario evaluar el segundo subproblema para la evaluación del primero.
- Ej. Recurrencia de Fibonacci.

Ejemplo (3)

- Configuration graph.



The configuration graph of the 4-disk Tower of Hanoi.

J. Erickson. Notas curso de Algoritmos. http://web.engr.illinois.edu/~jeffe/teaching/algorithms/

Ejemplo (4)

- Finite state automata.

Estructuras de Datos de Gráficas

- Dos estructuras de datos para representar gráficas:
  - listas de adyacencia,
  - matrices de adyacencia.

Matrices de adyacencia

- $O(V^2)$ memoria, independiente del número de aristas en la gráfica.
- Si la gráfica es no dirigida, matriz simétrica en la diagonal.
- Como es simétrica, $A = A^T$.
- Gráfica con pesos. NIL donde no hay valor.
- ¿Cuándo es apropiado usar una representación con matrices de adyacencia? ¿Cuándo no?

Matrices de adyacencia

- ¿Cuánto nos toma decidir si una arista está en la gráfica o no?
  - O(1)
- ¿Listar a todos los vecinos de un vértice? ¿Cómo hacemos?
  - O(V)

Listas de adyacencia

- Arreglo de listas ligadas, una lista por vértice.
- Cada lista ligada almacena a todos los vecinos de su vértice correspondiente.
- Gráficas no dirigidas: (u,v) se almacena dos veces.
- Gráficas dirigidas: cada arista se almacena una sola vez.
- El espacio de almacenamiento requerido es:
  - O(V+E).

Listas de adyacencia

- ¿Encontrar la lista de vecinos de un nodo?
  - O(1+deg(v)) tiempo, recorriendo la lista de vecinos.
- ¿(u,v) es una arista de la gráfica?
  - O(1+deg(v)) tiempo, recorriendo la lista de vecinos.
- Para gráficas no dirigidas podemos mejorar el tiempo a:
  - O(1+min{deg(u),deg(v)} recorriendo simultáneamente las dos listas y parar cuando se encuentra la arista o se llega al final de una lista.

Recorridos en gráficas

- Supondremos para el resto de la clase gráficas no dirigidas.
- Supongamos que queremos visitar cada nodo en una gráfica conectada.
- Recorrido genérico de gráficas almacena un conjunto candidato de aristas en una estructura de datos "la bolsa".
  - una pila o una cola son tipos particulares de bolsa.