Algorítmica

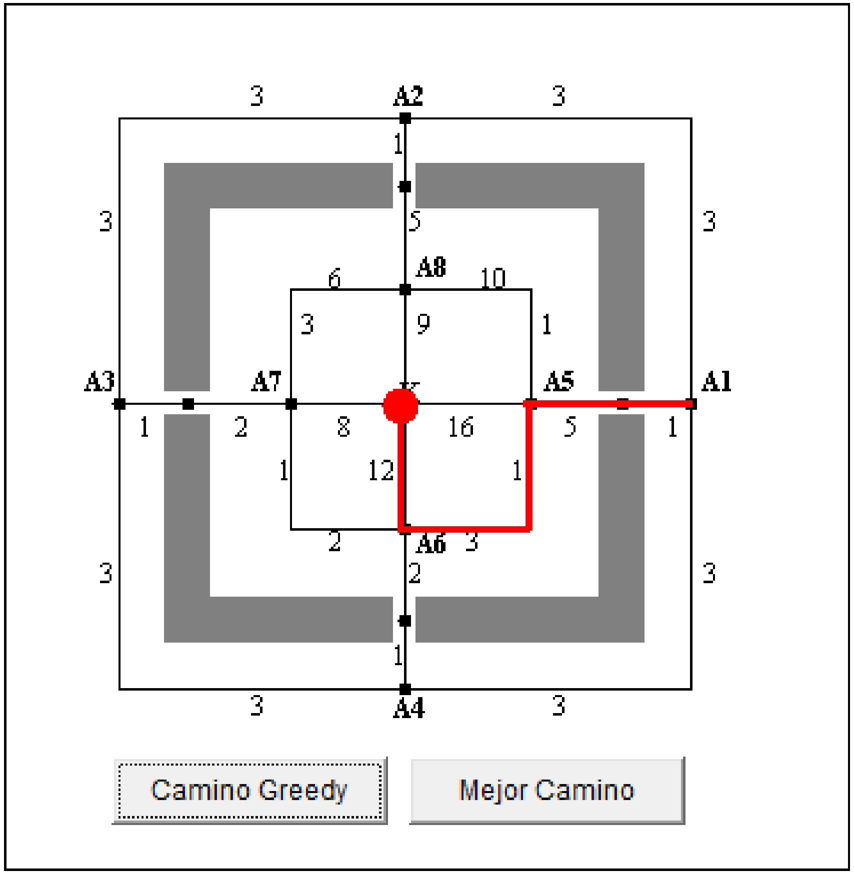
Capítulo 3. Algoritmos Greedy*

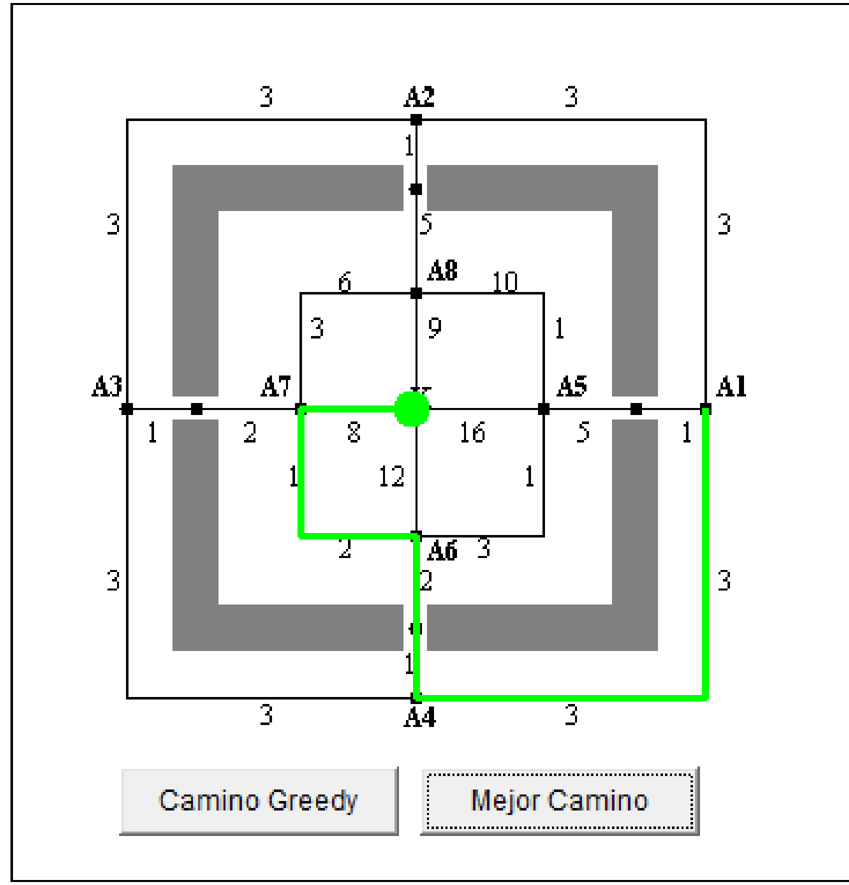
Tema 7. Algoritmos Greedy

- El enfoque greedy
- Repaso de grafo

* Voraz en español

La filosofía greedy





El enfoque greedy

- La idea básica consiste en seleccionar en cada momento lo mejor entre un conjunto de candidatos, sin tener en cuenta lo ya hecho, hasta obtener una solución para el problema.
- Cuando para resolver un problema se le aplica el **enfoque greedy**, el algoritmo resultante se denomina **Algoritmo Greedy**
- El enfoque greedy solo se puede aplicar a problemas que reúnan un conjunto de características: **Problemas Greedy.**
- Suelen ser problemas de optimización

Características de un problema greedy

- Para poder resolver un problema con un enfoque greedy, ha de reunir las 6 siguientes características, que no son necesarias, pero si suficientes:
  1. Un **conjunto de candidatos**: Las tareas a ejecutar, los nodos de un grafo, etc.
  2. Una lista de **candidatos ya usados.**
  3. Un criterio (**función) solución** que dice cuándo un conjunto de candidatos forma una solución (no necesariamente óptima).

Características de un problema greedy

4. Un criterio que dice cuándo un conjunto de candidatos (sin ser necesariamente una solución) es factible, es decir, podrá llegar a ser una solución (no necesariamente óptima).
5. Una función de selección que indica en cualquier instante cuál es el candidato más prometedor de los no usados todavía.
6. Una función objetivo que a cada solución le asocia un valor, y que es la función que intentamos optimizar (a veces coincide con la de selección)

Enfoque greedy

- Un algoritmo Greedy procede siempre de la siguiente manera:
  - Se parte de un conjunto de candidatos a solución vacío: $S = \emptyset$
  - De la lista de candidatos que hemos podido identificar, con la función de selección, se coge el mejor candidato posible,
  - Vemos si con ese elemento podríamos llegar a constituir una solución: Si se verifican las condiciones de factibilidad en S
  - Si el candidato anterior no es válido, lo borramos de la lista de candidatos posibles, y nunca mas es considerado
  - Evaluamos la función objetivo. Si no estamos en el óptimo, seleccionamos con la función de selección otro candidato y repetimos el proceso anterior hasta alcanzar la solución.
- La primera solución que se consigue suele ser la Solución Óptima del problema

El enfoque greedy

- FUNCION GREEDY

  $S = \varnothing$

  Mientras S no sea una solución y $C \neq \varnothing$ Hacer:

      X = elemento de C que maximiza SELEC (X)

      $C = C - \{X\}$

      Si $(S \cup \{X\}$ es factible Entonces $S = S \cup \{X\}$

  Si S es una solución entonces devolver S

  caso contrario “NO HAY SOLUCION”

- C es la lista de candidatos.
- El enfoque Greedy suele proporcionar soluciones óptimas, pero no hay garantía de ello. Por tanto, siempre habrá que estudiar la **corrección del algoritmo** para verificar esas soluciones.

Ejemplo

- Se desea dar cambio usando el menor número posible de monedas de 1, 5, 10 y 25
- ¿Es greedy el problema?:
  - Candidatos: 1, 5, 10, 25 (con una moneda de cada tipo por lo menos).
  - Usados: Podremos definirlo.
  - Solución: Lista de candidatos tal que la suma de los mismos coincida exactamente con el cambio pedido.
  - Criterio de factibilidad: Que no se supere el cambio.
  - Criterio de selección: Se escoge la moneda de mayor valor entre las disponibles.
  - Objetivo: El número de monedas ha de ser mínimo.

Ejemplo

- Si tenemos, por ejemplo, una moneda de 100 que queremos cambiar y la máquina dispone de 3 monedas de 25, 1 de 10, 2 de 5 y 25 de 1, entonces la primera solución que alcanza el algoritmo greedy directo es justamente la Solución Óptima: 3 de 25, 1 de 10, 2 de 5 y 5 de 1.
- Pero si tenemos 10 monedas de 1, 5 de 5, 3 de 10, 3 de 12 y 2 de 25, la solución del algoritmo para una moneda de 100 sería 2 de 25, 3 de 12, 1 de 10 y 4 de 1, en total diez monedas.
- La solución no es óptima, ya que existe otra mejor que utiliza nueve monedas en vez de diez, que es 2 de 25, 3 de 10 y 4 de 5.

Greedy y Óptimo

- Los algoritmos greedy no alcanzan soluciones óptimas siempre
- Esta desventaja es una ventaja en problemas en los que es imposible (o muy difícil) alcanzar el óptimo: Heurísticas Greedy:
  - Problema del Coloreo de un Grafo,
  - Problema del Viajante de Comercio,
- Pueden alcanzar óptimos locales, pero no los óptimos globales de los problemas.
- Por eso en cada caso habrá que demostrar la corrección del algoritmo

Almacenamiento óptimo en cintas

- Almacenar n programas en una cinta de longitud L.
- Cada programa i tiene una longitud l_i , $1 \leq i \leq n$
- Los programas podrán guardarse en la cinta si y solo si la suma de todas las longitudes es menor que L
- Suponemos que siempre que vamos a recuperar un programa, la cinta está posicionada al principio.
- Si los programas están colocados en el orden $I = \{i_1, i_2, \ldots i_n\}$ el tiempo t_j necesario para recuperar i_j es proporcional a

$$\sum_{1 \leq k \leq j} l_{i_k}$$

Almacenamiento óptimo en cintas

- Todos los programas se recuperan del mismo modo, siendo el tiempo medio de recuperación (TMR),

$$(1/n)\sum_{1\leq j\leq n} t_j$$

- Nos piden encontrar una permutación (un orden) de los n programas tal que cuando esten almacenados en la cinta el TMR sea mínimo.
- Minimizar el TMR es equivalente a minimizar

$$D(I)=\sum_{1\leq j\leq n}\sum_{1\leq k\leq j} l_{i_k}$$

Ejemplo

- El problema reúne todos los ingredientes greedy

- Sea n = 3 y $(l_1,l_2,l_3) = (5,10,3)$

Orden I		D(I)
1,2,3	5 + 5 + 10 + 5 + 10 + 3	= 38
1,3,2	5 + 5 + 3 + 5 + 3 + 10	= 31
2,1,3	10 + 10 + 5 + 10 + 5 + 3	= 43
2,3,1	10 + 10 + 3 + 10 + 3 + 5	= 41
3,1,2	3 + 3 + 5 + 3 + 5 + 10	= 29
3,2,1	3 + 3 + 10 + 3 + 10 + 5	= 34

Solución Greedy

- Partiendo de la cinta vacía

  Para i := 1 to n do

  grabar el siguiente programa mas corto

  ponerlo a continuación en la cinta

- El algoritmo escoge lo más inmediato y mejor sin tener en cuenta si esa decisión será la mejor a largo plazo.

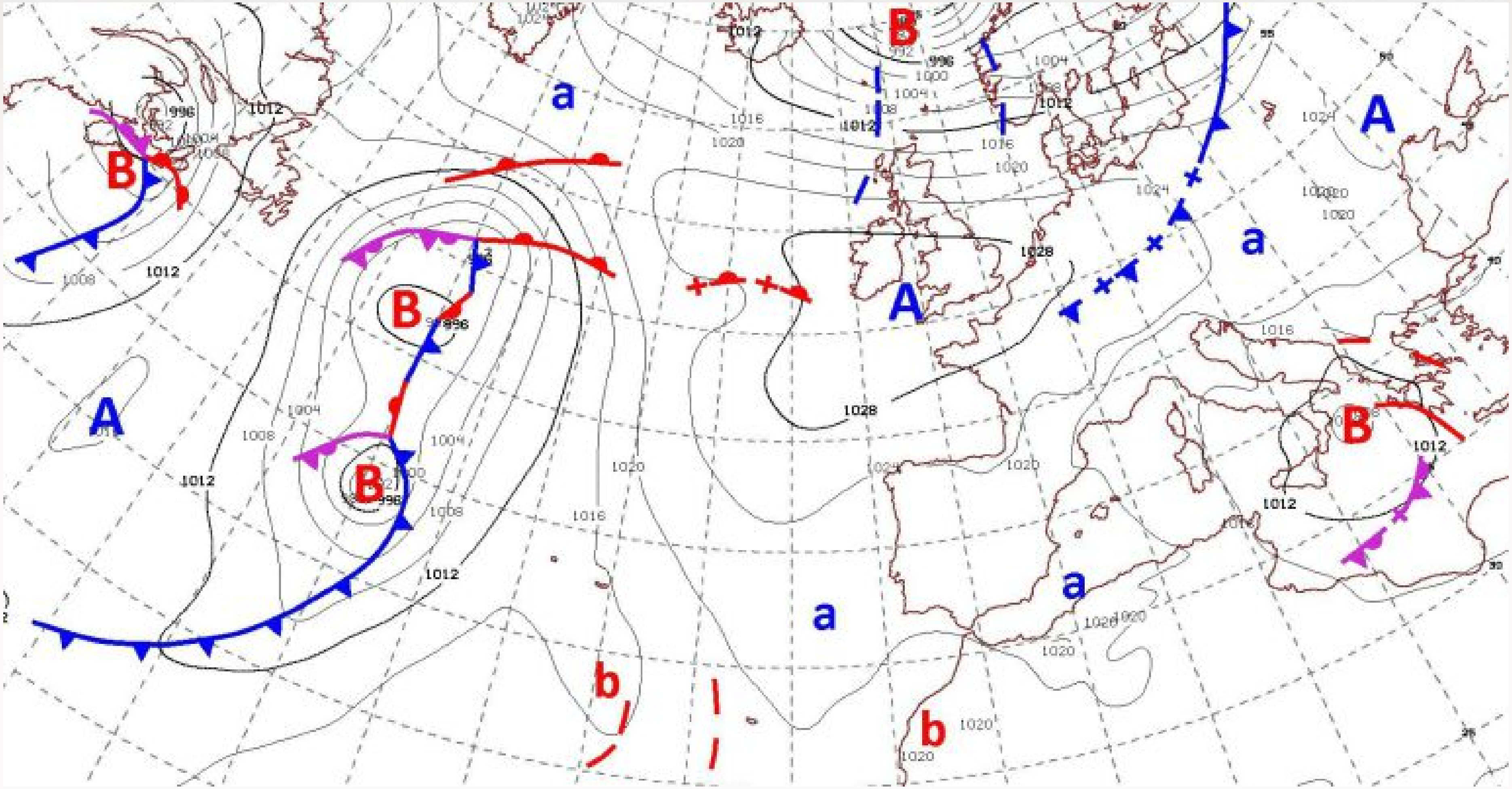
Almacenamiento óptimo en cintas

- Teorema
- Si $l_1 \leq l_2 \leq \ldots \leq l_n$ entonces el orden de colocación $i_j = j$, $1 \leq j \leq n$ minimiza

$$\sum_{k=1}^{n} \sum_{j=1}^{k} l_{i_j}$$

- Para todas las posibles permutaciones de ij
- Pero ¿siempre podremos considerar un óptimo local como uno global?

Greedy y Óptimos



Greedy y Óptimos

- Hay casos en los que un óptimo local podrá considerarse como global
- Típico problema de optimización, $f: R^n \rightarrow R$,

$$\text{Min } \{f(x) : x \in S\}$$

- A $\{x \in R^n: x \in S\}$ se le llama solución factible del problema.
- Si $x^* \in S : f(x^*) \leq f(x), \forall x \in S$, entonces x^* es un óptimo global.
- Si $x^* \in S$ y existe un entorno $V(x^*)$ tal que

$$f(x^*) \leq f(x), \forall x \in S \cap V(x^*)$$

entonces x^* es un óptimo local.

Greedy y Óptimos

- Supongamos que S (el conjunto factible) es un conjunto contenido en R^n ($S \subseteq R^n$)
- S no vacío y convexo;
- Sea $f: S \rightarrow R$ y el problema

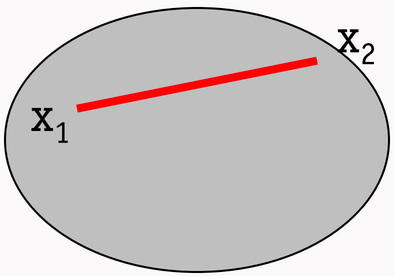
$$\text{Min: } f(x),\ x \in S$$

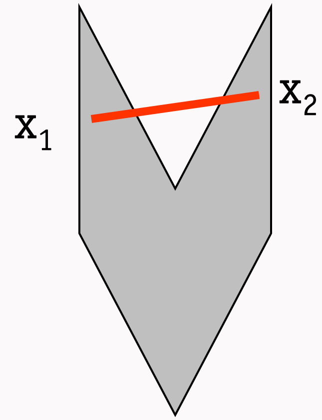
- Si x^* es un óptimo local, entonces si f es una **función convexa**, el óptimo local es un óptimo global.

¿Se verificaba esto en el anterior problema del cambio de monedas?

Convexidad

- S es un conjunto convexo cuando $\forall\ x_1, x_2 \in S$ y $\forall\ \lambda \in [0,1] \rightarrow \lambda\ x_1 + (1 - \lambda)\ x_2 \in S$, lo que se traduce como: "Dados dos puntos del conjunto S, todo el segmento lineal que los une está en el conjunto".





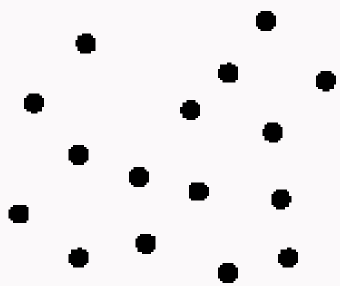
- En el caso del problema del cambio de monedas, el conjunto factible no era convexo

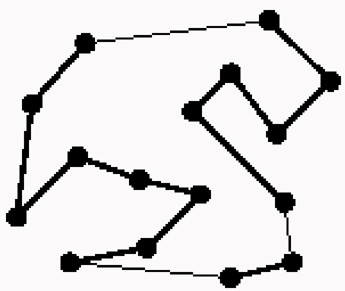
Algoritmos Greedy para Grafos

- ¿Por que hay que estudiar grafos?
- Podemos abstraer grafos de distintas situaciones físicas del mundo real para:
  - Resolver problemas de recorridos dando servicios eficientes a nuestros clientes o a los usuarios de los sistemas:
  - Por ejemplo el Problema del Viajante de Comercio
  - Diseñar Redes poco costosas de computadores, de telefonía, etc.
- Son básicos en Inteligencia Artificial, pero también en Arquitectura y en otras muchas Ingenierías
- Desde luego en Robótica

Algoritmos Greedy para Grafos

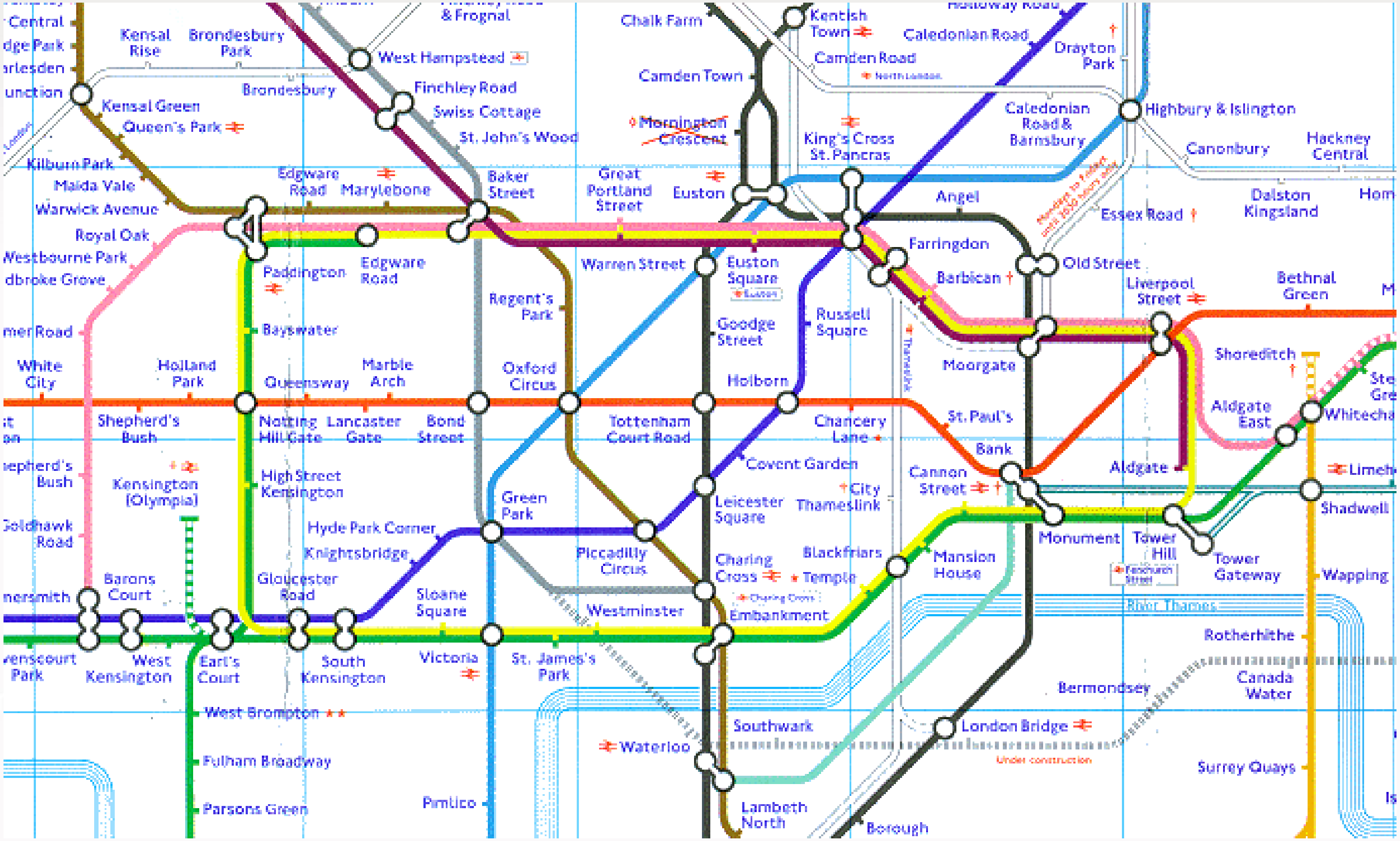
- Supongamos que tenemos un robot que maneja un soldador.
- Para que el robot haga las soldaduras que queremos, debemos darle el orden en que debe visitar los puntos de soldadura, de modo que visite (y suelde) el primer punto, luego el segundo, etc., hasta que concluya su tarea
- Como los robots son caros, necesitamos encontrar el orden que minimiza el tiempo (es decir, la distancia recorrida) que tarda en realizar todas las soldaduras del panel



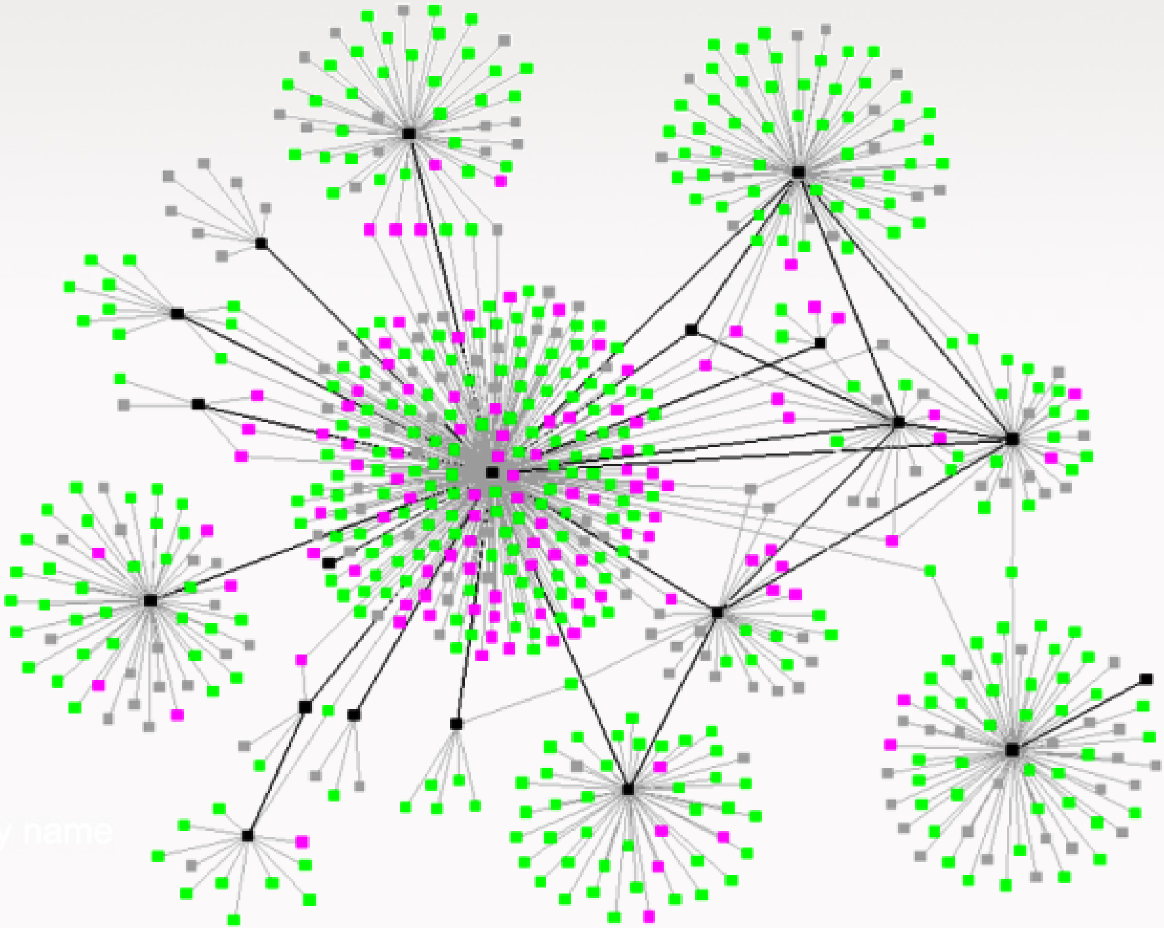


- Pero hay muchos más…

Ejemplos de grafos

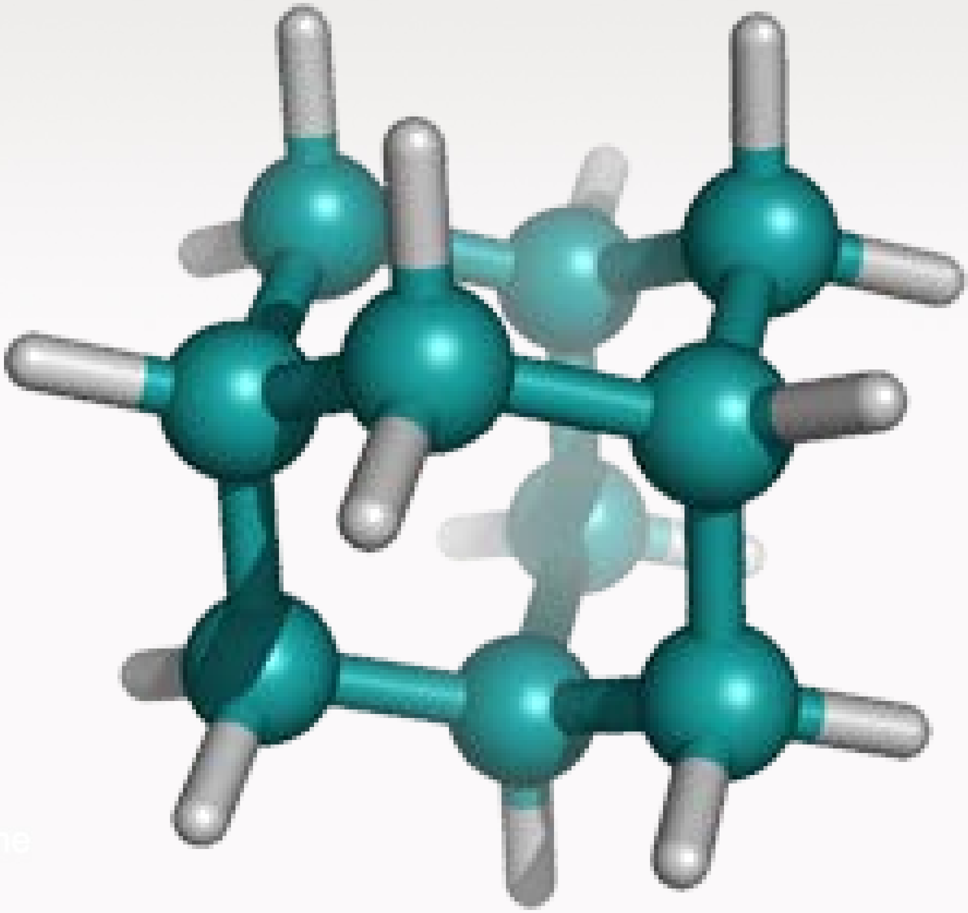


Ejemplos de grafos



Redes Sociales

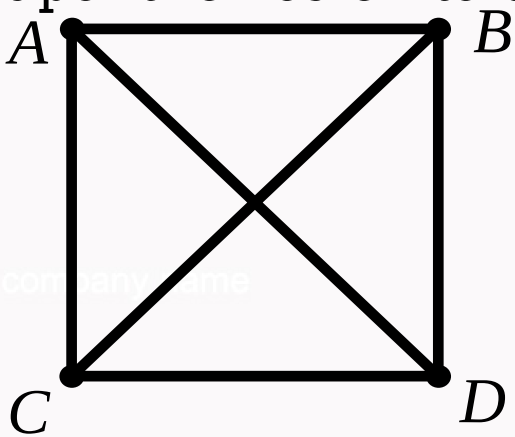
Ejemplos de grafos

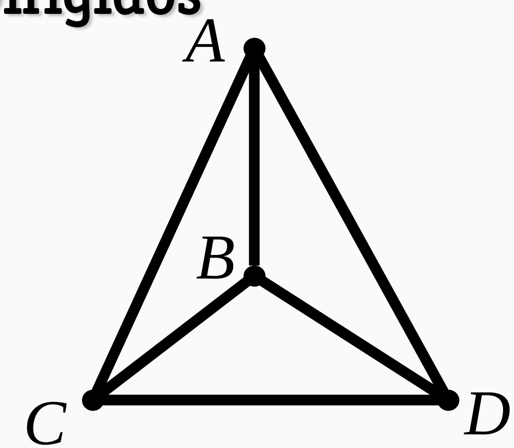


Modelos químicos

Nociones básicas de grafos

- Un grafo se define con dos conjuntos:
  - Un conjunto de **vértices** (nodos), y
  - Un conjunto de **aristas**
- Cuando las aristas tienen origen y final (dirección), se habla de **Grafos Dirigidos**, y en lugar de aristas tendremos arcos. Tambien hay **Grafos Ponderados**
- Supondremos en lo que sigue **Grafos no Dirigidos**





Definición formal

- Sea $X = \{x_1, x_2, \ldots, x_n\}$ un conjunto finito, no vacío. Sea $A \subseteq X \times X$ una relación. Al par $G = (X,A)$ se le llama **grafo dirigido.**
- Sea $I_X = \{(x_i, x_i), i = 1, 2, \ldots, n\}$ y $X\_^{2} = (X \times X) - I_X$ (para eliminar lazos). Definimos en $X\_^{2}$ una relación R tal que:

$$(x_i, x_j)R(x_h, x_k) \Leftrightarrow (x_i,x_j) = (x_h,x_k) \text{ ó } (x_i, x_j) = (x_k, x_h)$$

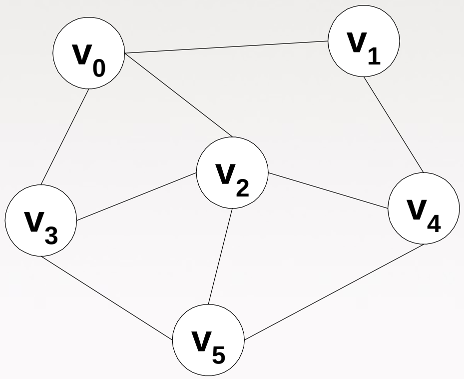
(para eliminar la dirección de los puntos)

- R es una relación de equivalencia. Definimos el conjunto cociente $(X\_^{2} / R)$ para esa relación. Entonces, al par

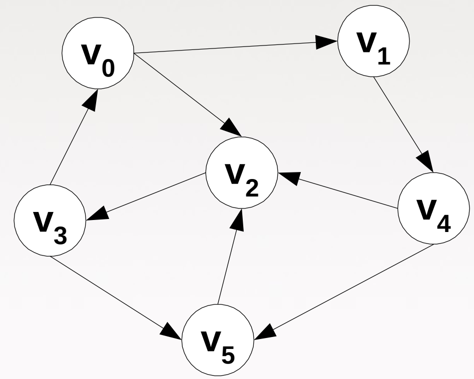
$$G = (X, B), \text{ con } B \subseteq X\_^{2} / R$$

se le llama **grafo no dirigido**

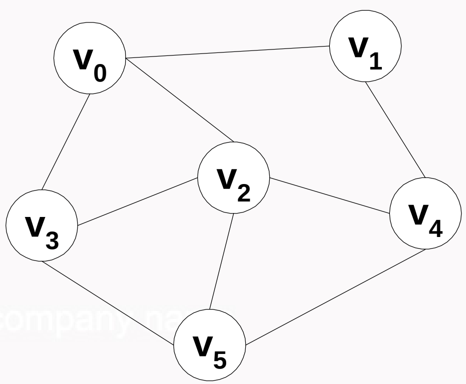
Ejemplos



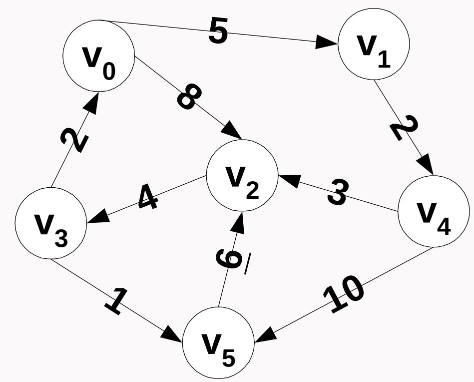
No Dirigido



Dirigido



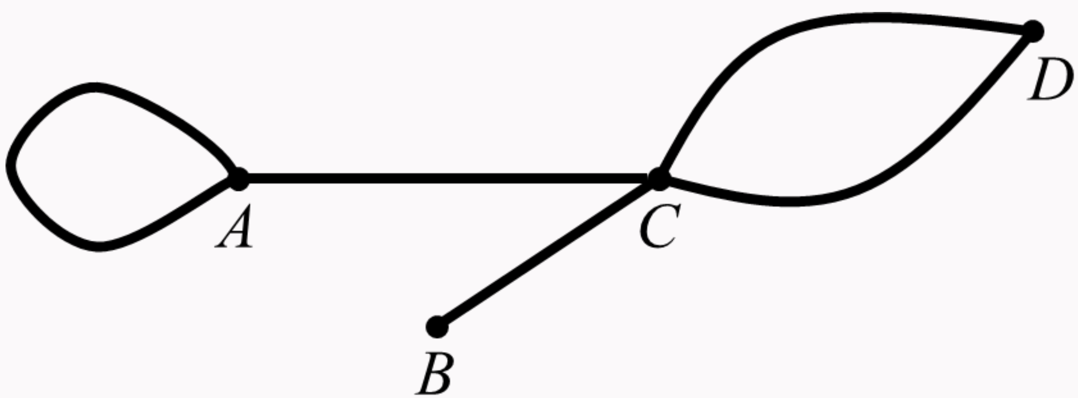
No Ponderado



Ponderado

Nociones básicas de grafos

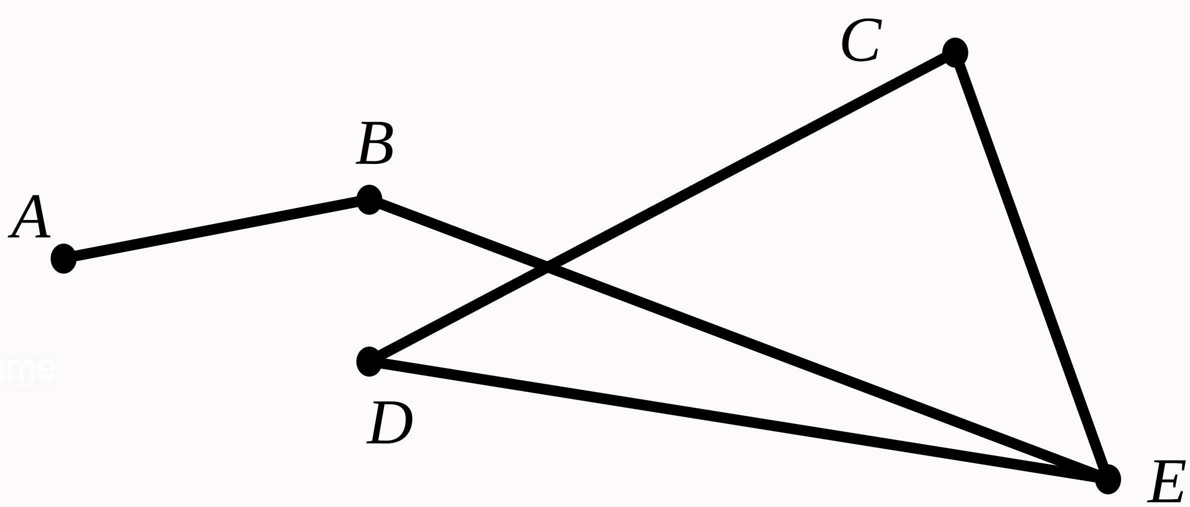
- Podemos unir dos vértices varias veces, obteniendo múltiples aristas
- Podemos unir un vértice a si mismo, para formar un lazo



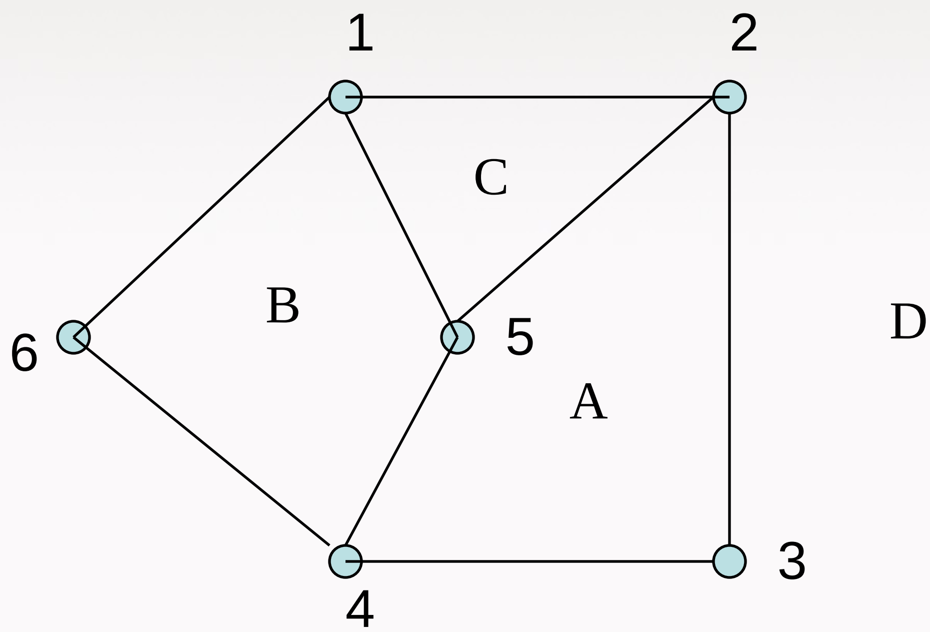
- Un grafo es completo si cualquier par de vértices distintos está unido por una arista

Nociones básicas de grafos

- Dos vértices son adyacentes si existe una arista que los une (B y E son adyacentes, pero el B y el D no)
- Dos aristas son adyacentes si comparten un vértice (AB y BE son adyacentes, pero las AB *y* CE no)



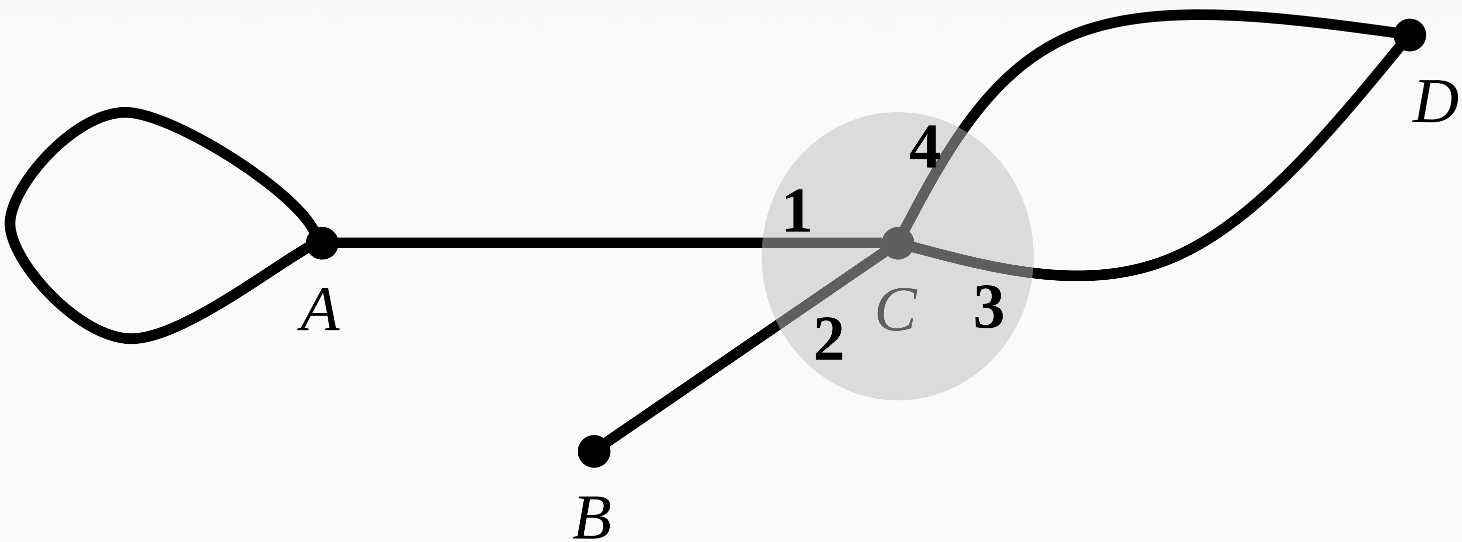
Nociones básicas de grafos



Un grafo se llama plano si no puede pintarse en el plano (o en una esfera) sin que se crucen sus aristas.

Nociones básicas de grafos

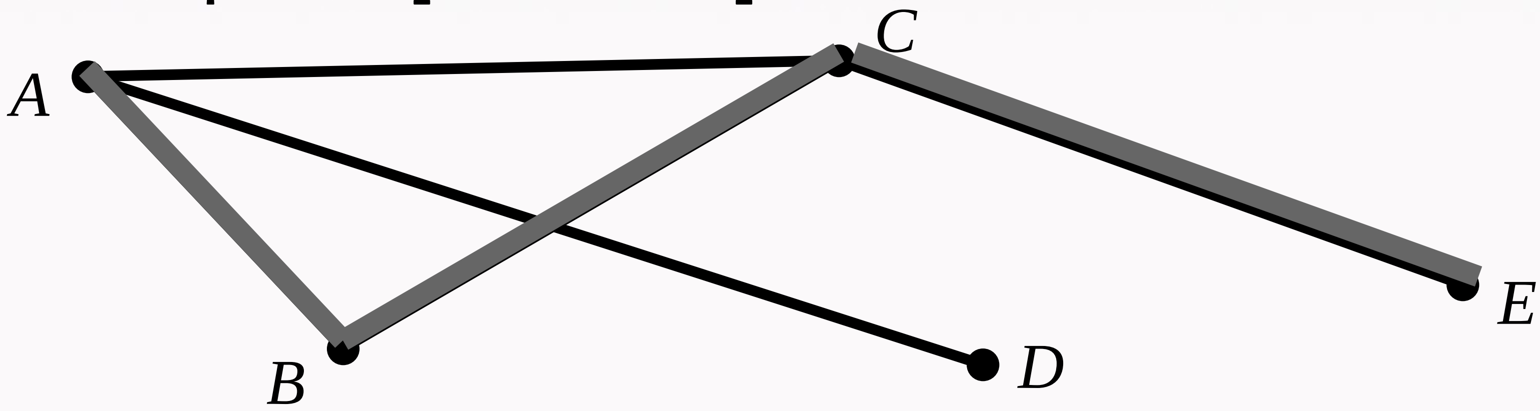
- El grado de un vértice es el número de aristas que pasan por ese vértice.



Gra(C) = 4, Gra(A) = 3, Gra(B) = 1, Gra(D) = 2

Nociones básicas de grafos

- Un **camino** es una sucesión de aristas distintas adyacentes.
  - ¡No se permite repetir aristas en un camino!

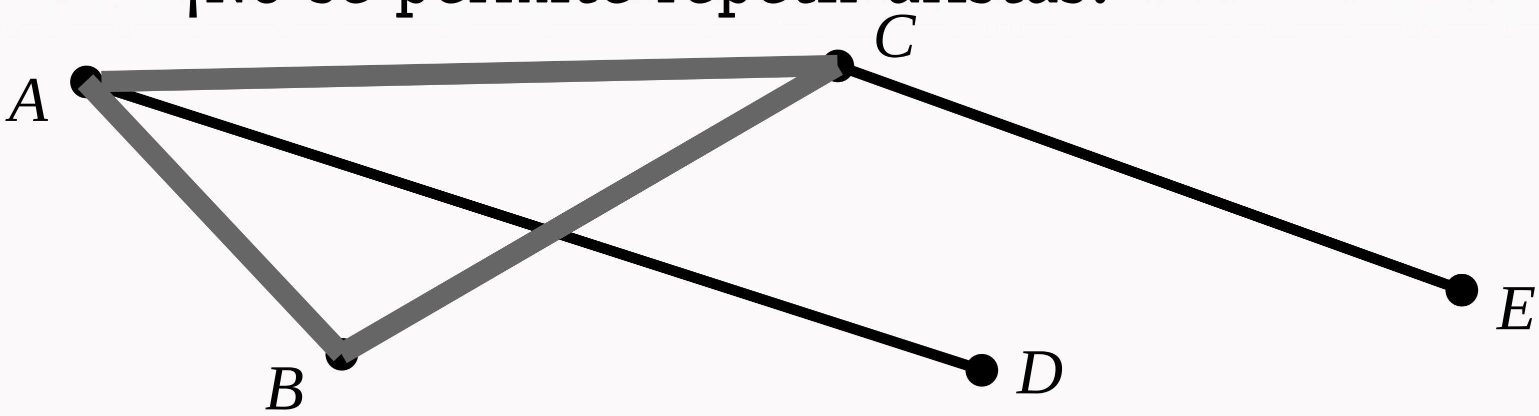


Las aristas AB, BC, y CE forman el camino ***A,B,C,E***

Las aristas AD, DA y AC no forman un camino (repetición)

Nociones básicas de grafos

- Un **circuito** es un camino que comienza y termina en el mismo vértice.
  - ¡No se permite repetir aristas!

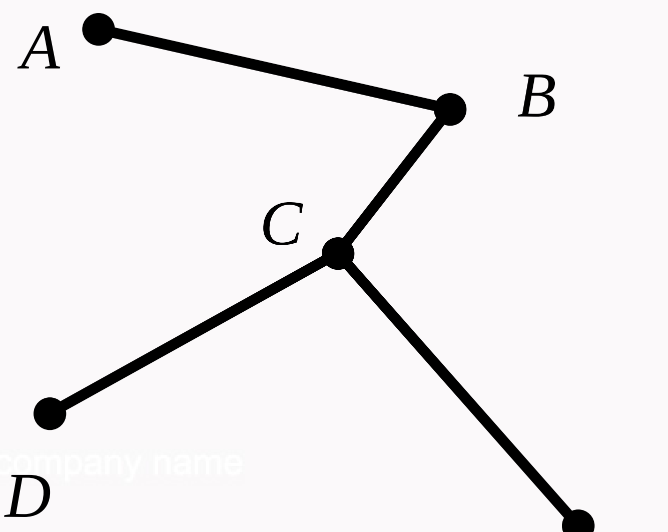


AB, BC, and CA forman el circuito ***A,B,C,A***

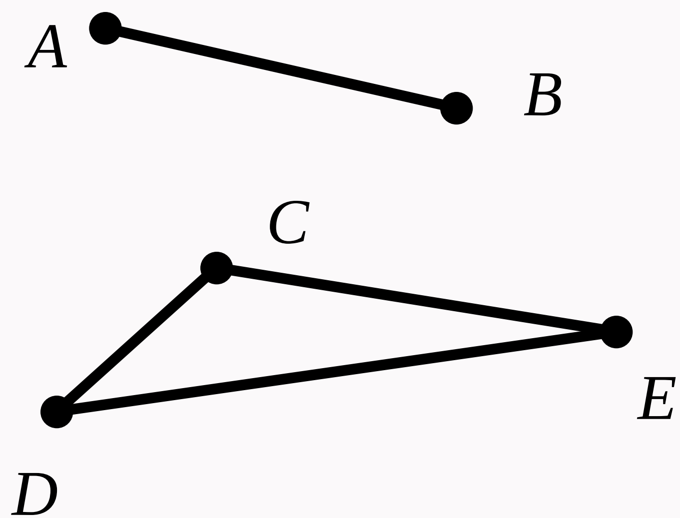
Las aristas BA y AD no forman un circuito

Nociones básicas de grafos

- Un grafo se dice **conexo** si cualesquiera dos vértices pueden unirse por un camino. Si un grafo no es conexo, se llama **inconexo**

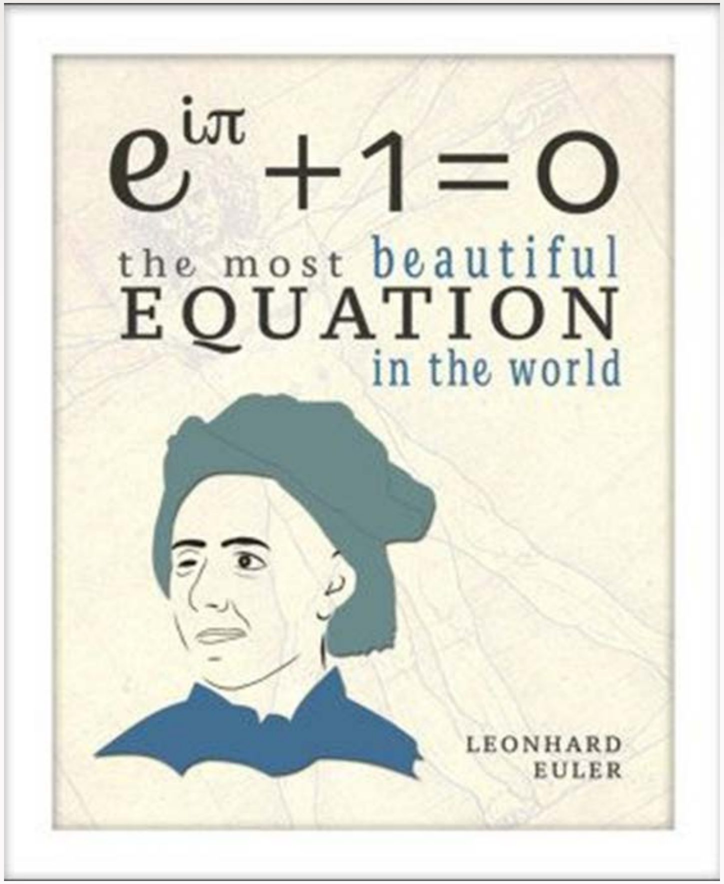


Conexo



Inconexo

Nociones básicas de grafos

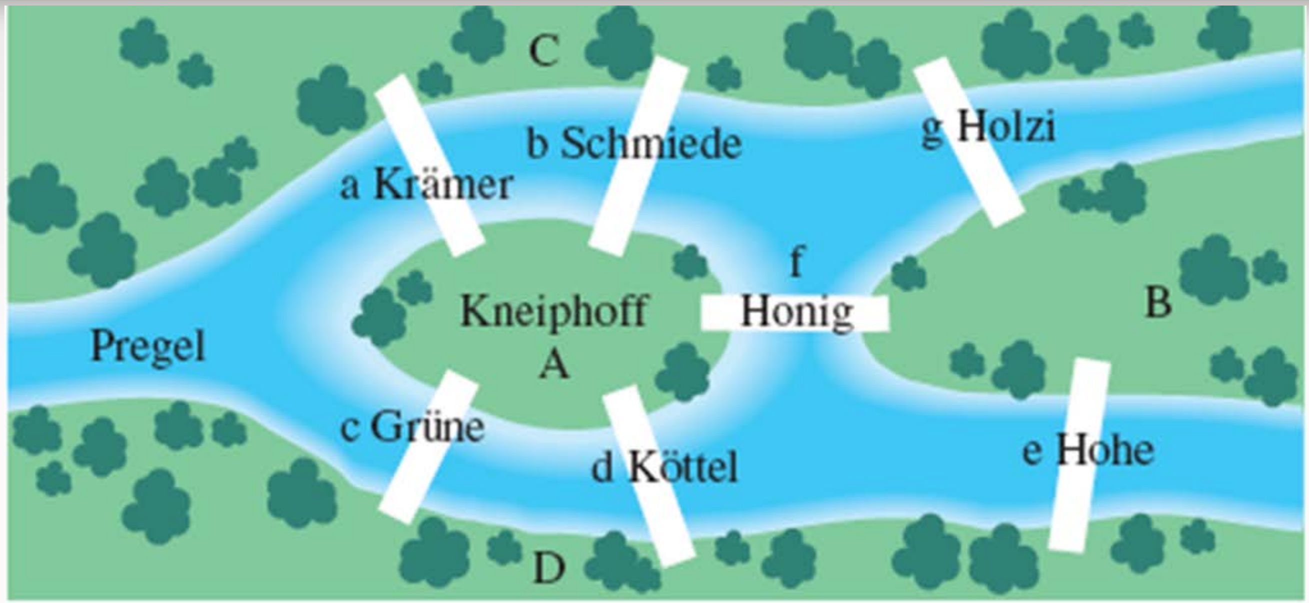


Leonhard Euler (1707-1783) ha sido el matemático mas prolífico de todos los tiempos, con contribuciones importantes en Geometría, Cálculo, Física, … y Grafos.

Aproximadamente la mitad de sus publicaciones las escribió después de quedar ciego. Cuando perdió la vista comentó: "Así ahora me distraeré menos."

Tuvo por lo menos 13 hijos.

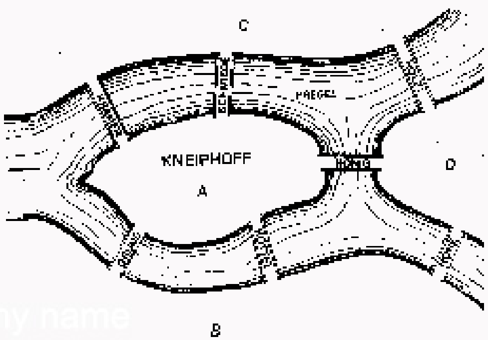
Nociones básicas de grafos

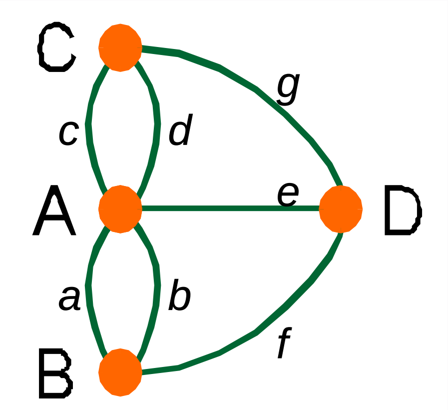


En la ciudad de **Königsberg (**en la antigua Prusia oriental; hoy Kaliningrado, Rusia) hay una isla llamada **"Kneiphoff**" bordeada por el rio **Pregel** y hay **7 puentes** conectando las orillas.
El problema es saber si una persona puede recorrer todos estos puentes, pasando por todos y cada uno de ellos solamente una vez, y volviendo a su punto de partida.

Nociones básicas de grafos

- Cuando Euler llego a Königsberg, habia consenso en la imposibilidad de hacer aquel recorrido, pero nadie lo aseguraba con certeza
- Euler planteó el problema como uno de grafos:
  - Cada parte de tierra supondría un vértice, y
  - Cada puente representaría una arista





Y en 1736 demostró la imposibilidad de dar un paseo como el que se quería dar.

Nociones básicas de grafos

- Un **Camino Euleriano** es un camino que pasa a través de cada arista del grafo una y solo una vez
- Un **Circuito Euleriano** es un circuito que pasa por cada arista del grafo una y solo una vez
- **Teorema de Euler**
  - Si todos los vértices de un grafo son de grado impar, entonces no existen circuitos eulerianos.
  - Si un grafo es conexo y todos sus vértices son de grado par, existe al menos un circuito euleriano.

Nociones básicas de grafos

- Un **Circuito Hamiltoniano** es un circuito que pasa a través de cada vértice una y solo una vez, y termina en el mismo vértice en el que comenzó.
- Los **Circuitos Hamiltonianos y los Circuitos Eulerianos** son conceptos distintos y separados: En un grafo podemos tener de unos y no de otros.
- A diferencia de los Circuitos Eulerianos, no tenemos un resultado simple que nos diga si un grafo tiene o no Circuitos Hamiltonianos.

William R. Hamilton

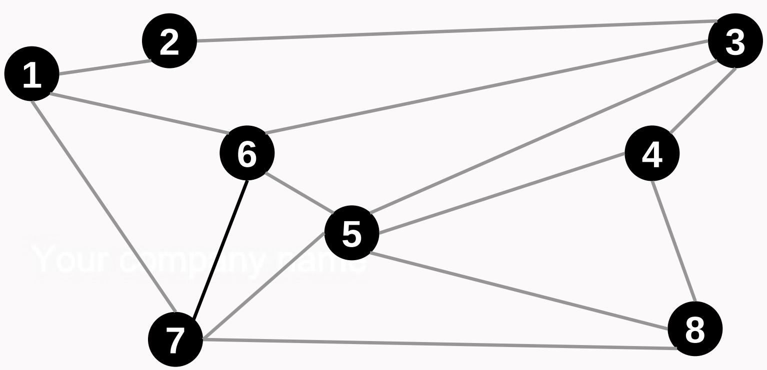
Ejemplos

- La determinación de un Circuito Euleriano mínimo es lo que se conoce con el nombre del Problema del Cartero Chino:
  - Encontrar el circuito de longitud mínima que recorre cada arista de un grafo al menos una vez.
- A la búsqueda de un Circuito Hamiltoniano mínimo se la conoce con el nombre de Problema del Viajante de Comercio:
  - Hallar el circuito de longitud mínima que recorre todos los nodos de un grafo una y solo una vez, comenzando y terminando por el mismo vértice

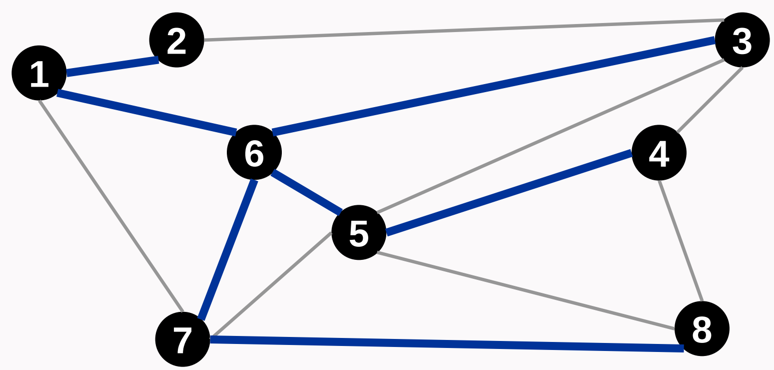
Árbol Generador de un Grafo

- Sea T = (V, F) un subgrafo de G = (V, E).
  - T es un árbol generador de G:
  - T es acíclico y conexo.
  - T es conexo y tiene |V| - 1 arcos.
  - T es acíclico y tiene |V| - 1 arcos.

G = (V, E)

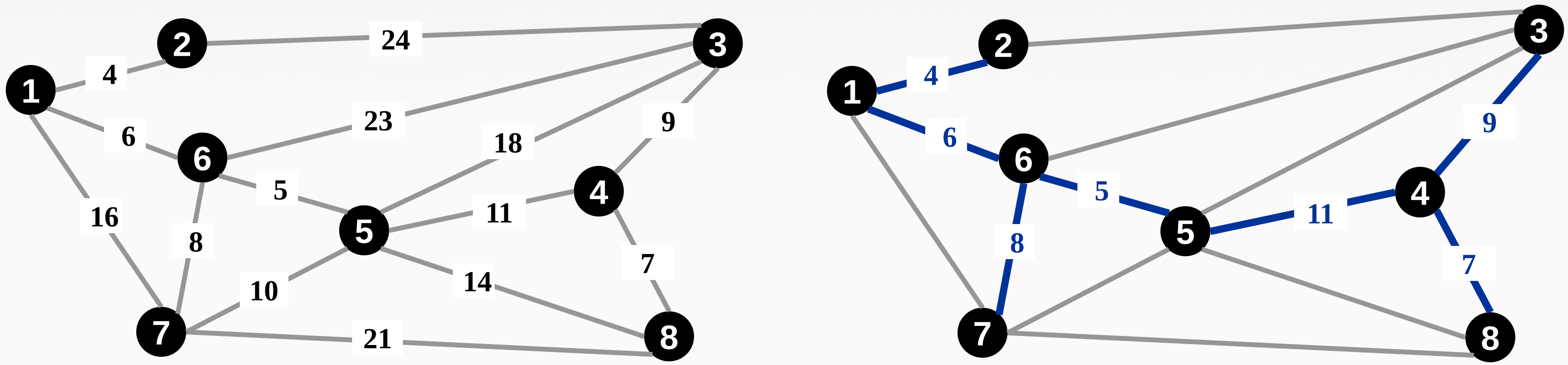


T = (V, F)



Árbol Generador Mínimo

- Dado un grafo conexo G con pesos en sus arcos c_e, un **Árbol Generador Mínimo** es un árbol generador de G en el que la suma de los pesos de sus arcos es mínima.



G = (V, E) T = (V, F) l(T) = 50

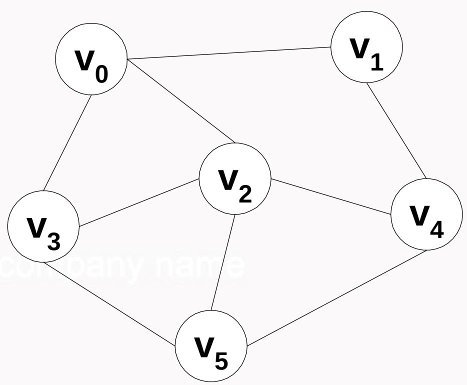
- **Teorema de Cayley** (1889). Hay n^{n-2} árboles generadores de K_n (el grafo completo de n vértices)
  - Por tanto el empleo de la fuerza bruta para encontrar el AGM de un grafo no es un método recomendable

La matriz de adyacencia

- Si suponemos un grafo G = (X, E) con n vértices, entonces su matriz de adyacencia es:

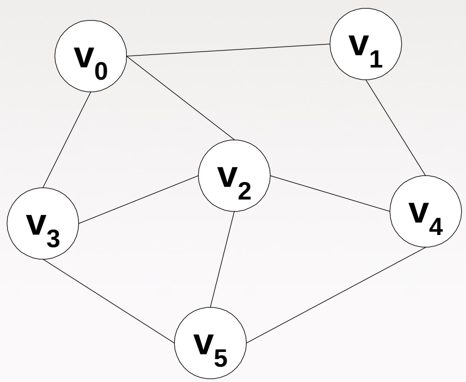
$$A_G(i, j) = \begin{cases} 1.....si(x_i, x_j) \in E \\ 0.....si(x_i, x_j) \notin E \end{cases}$$

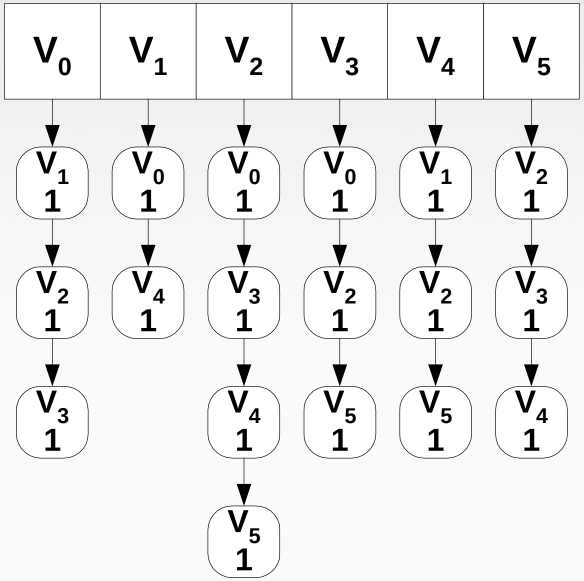
- Cuando el grafo es ponderado, el valor que aparece en cada casilla es el peso de la arista correspondiente

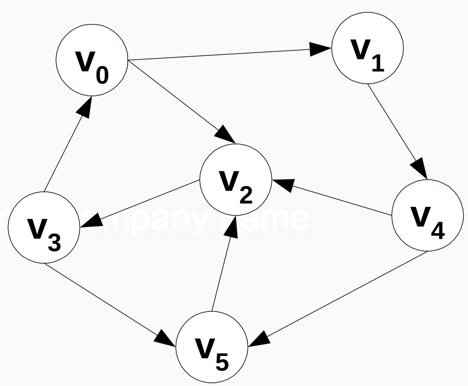


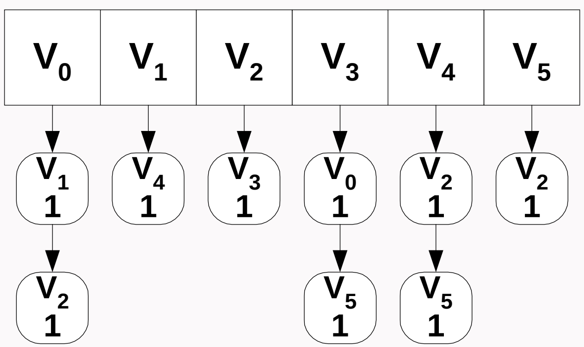
left $\rightarrow$ right	**0**	**1**	**2**	**3**	**4**	**5**
0	-	1	1	1	-	-
1	1	-	-	-	1	-
2	1	-	-	1	1	1
3	1	-	1	-	-	1
4	-	1	1	-	-	1
5	-	-	1	1	1	-

Representación por listas de adyacencia









Matriz de Incidencia

- Si se trata de un grafo no dirigido, entonces la matriz es:

$$B_G(i, j) = \begin{cases} 1.....si...x_i..esta..en..a_j \\ 0.....en..otro..caso \end{cases}$$

donde a_j es la arista que conecta x_i con x_j.

- Si se trata de un grafo dirigido, entonces la matriz es:

$$B(i, j) = \begin{cases} +1.....si..x_i..es..inicial..en..a_j \\ 0.....en..otro..caso..(bucle) \\ -1.....si..x_i..es..final..en..a_j \end{cases}$$

donde a_j es el arco que conecta x_i con x_j.