

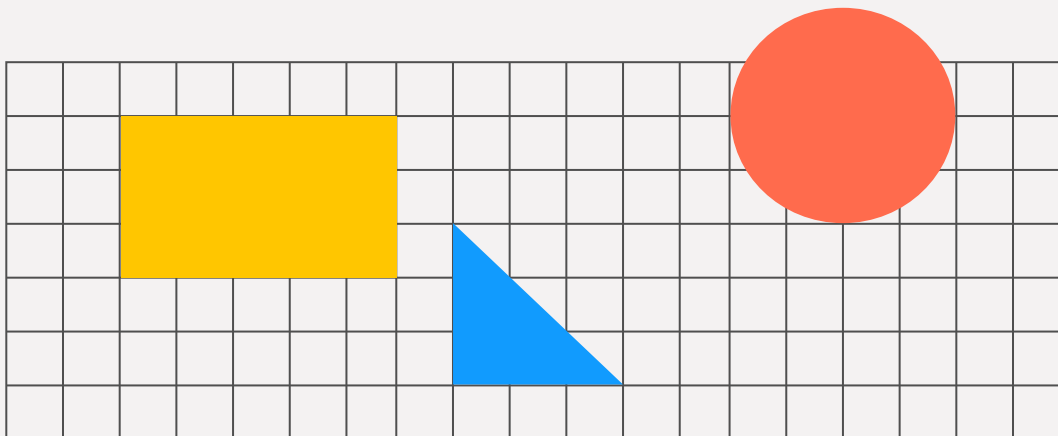
Grafos Pte. 1

Aldo Adrian Davila Gonzalez 1994122

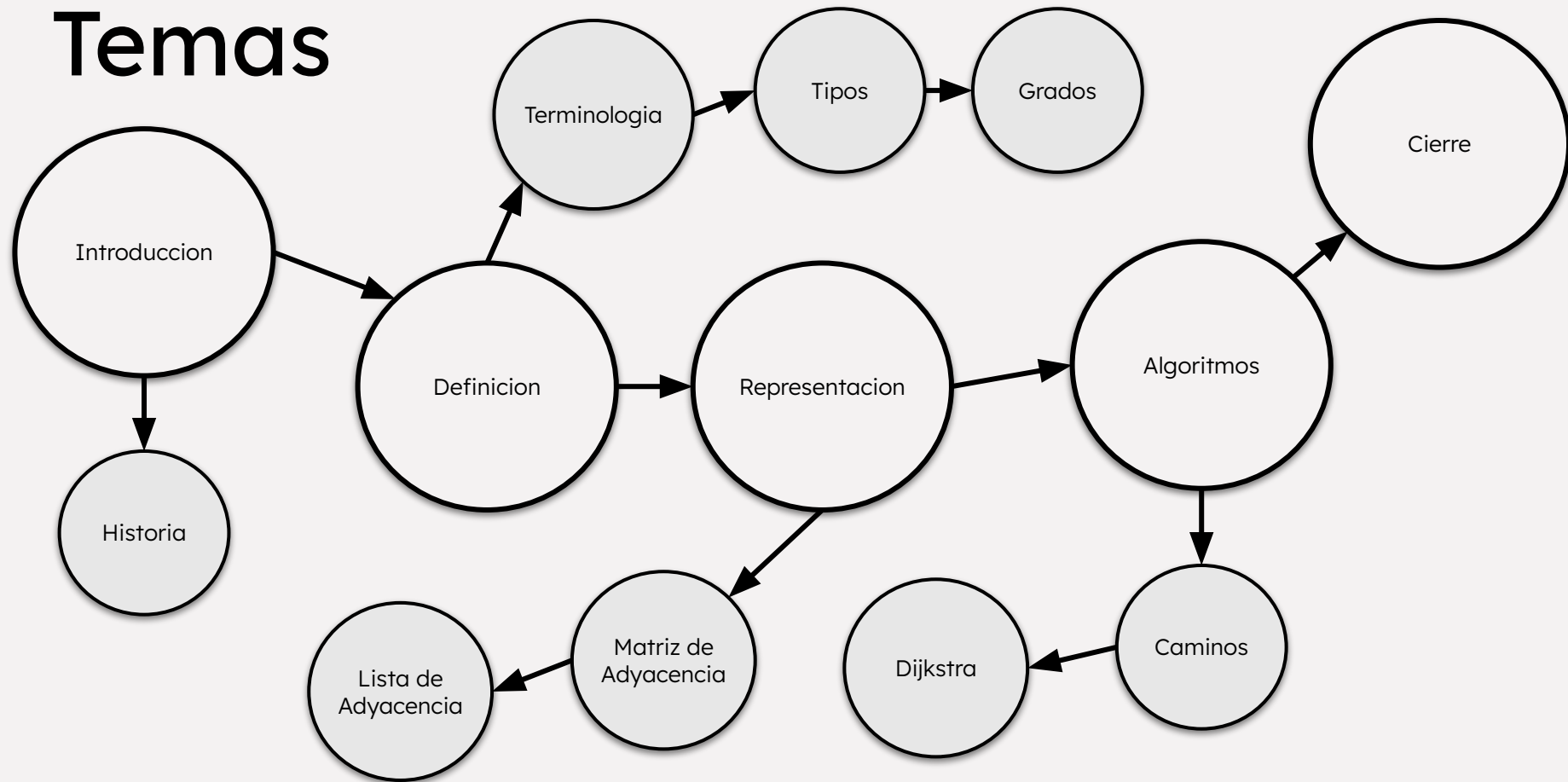
Alexis Ibarra Rodriguez 2094647

Roberto Sánchez Santoyo 2177547

Brian Chapa Valle 2093971



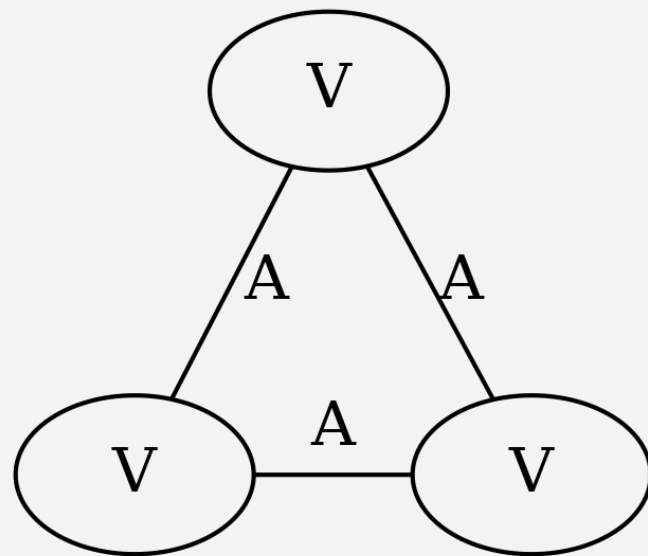
Temas



Introduccion

Los grafos son una estructura matemática formalmente definida **capaces de representar y facilitar la visualización de problemas que involucran una relación entre varios objetos**. Los grafos se componen de dos elementos, **vértices y aristas**.

La definición formal de los grafos es relativamente reciente. Aun así, **los grafos son utilizados ampliamente en la resolución de muchos problemas científicos y prácticos en la actualidad**



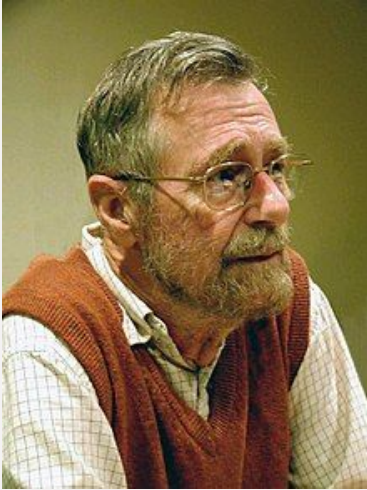
Historia

La **primera aplicación** de los grafos para resolver un problema en la historia registrada fue el caso del famoso matemático **Leonhard Euler modelando las conexiones entre los puentes de la ciudad de Königsberg** (Kaliningrad, Rusia), para determinar si había una manera de cruzarlos todos sin repetir puentes. Esto data del año 1736, pero en realidad **la definición formal de los grafos y su aplicación generalizada, recae en tiempos más modernos.**



Leonhard Euler

Historia



Edsger Dijkstra

El término de *grafo* no sería usado hasta el siglo XIX. Incluso teniendo el término acuñado, **el primer libro acerca de la teoría de grafos no se publicaría sino hasta 1936.**

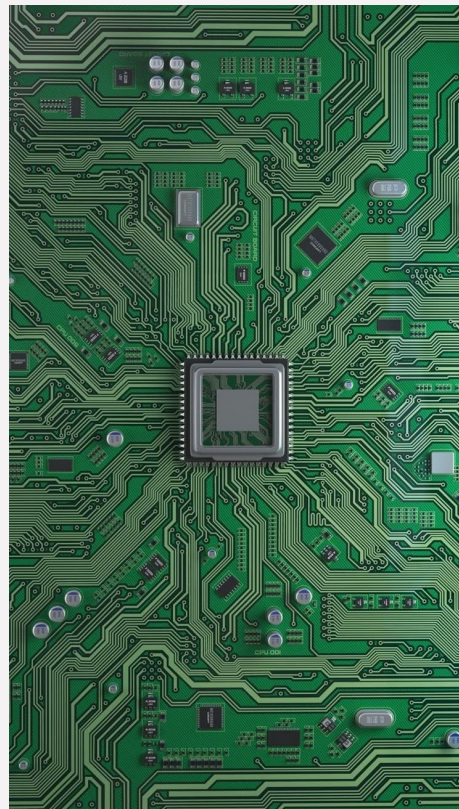
La teoría y la utilización de los grafos ganaría muchísima relevancia en el siglo XX. **Aplicaciones para los grafos fueron encontradas en el diseño de sistemas electrónicos, redes de comunicación,** entre otros. Así, **el descubrimiento de estas aplicaciones motiva a su vez la generación de nuevos conocimientos teóricos para resolver estos problemas,** como es el caso del algoritmo de Dijkstra para conocer el camino más corto que conecta dos vértices en un grafo ponderado.

Impacto de la Teo. de Grafos

Uno de los ejemplos de cómo la utilización de grafos impactan nuestra vida está presente en todos los aparatos electrónicos que utilizamos en la vida diaria.

Las esquemáticas de circuitos pueden pensarse como grafos en los que cada elemento eléctrico es un nodo unido por las aristas que representan el cableado que los conecta.

La teoría de grafos también se utiliza para ver si un circuito eléctrico puede ser dibujado de tal manera que los cables no se crucen entre sí. En esencia se intenta generar un grafo planar (que se pueda dibujar sin que se crucen las aristas) que represente el circuito, usando algoritmos existentes.

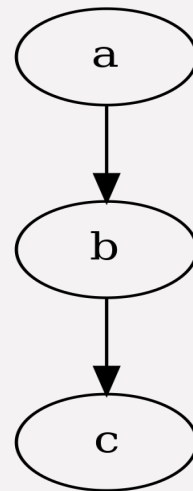


Definición de un Grafo

Los grafos se definen matemáticamente mediante la teoría de conjuntos. Un grafo cualquiera G se define como una tupla ordenada de elementos que representan su composición.

$$\text{Grafo} := G(V, A, \dots)$$

Un grafo G debe contar por lo mínimo con dos elementos **V** y **A** . **V es un conjunto** arbitrario **que representa los vértices** que conforman el grafo y **A es el conjunto de las aristas que conectan los vértices**. A veces se le añaden elementos a la definición dependiendo de qué tipo de grafo estemos definiendo.



Para este grafo en particular...

$$V = \{a, b, c\}$$

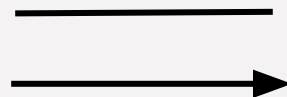
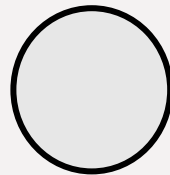
$$A = \{(a,b), (b,c)\}$$

Terminología

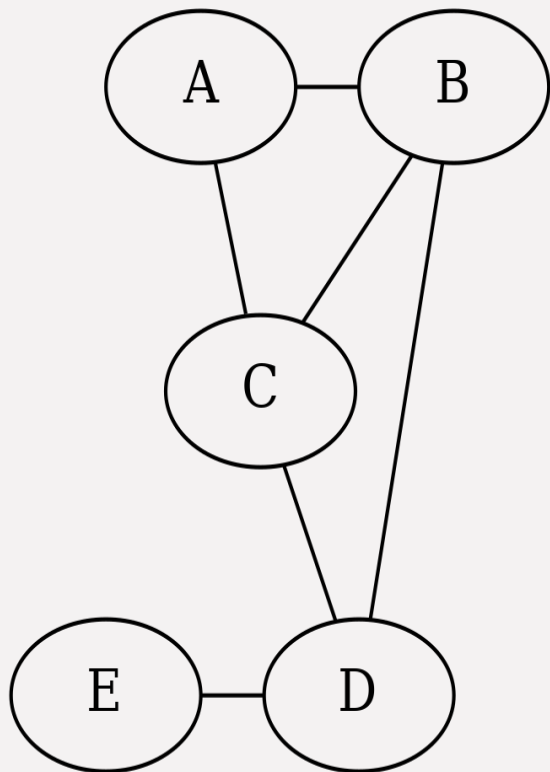
Antes de empezar tenemos que tener claros algunos conceptos a utilizar para hablar acerca de grafos.

Vértice (v): También llamado nodo, representa algún objeto de interés el cual nos interesa relacionar con otros objetos del mismo tipo. Gráficamente se representa mediante un círculo/óvalo.

Arista (a): También llamada arco, es la conexión que representa la relación entre dos nodos de un grafo. Pueden ser representadas por líneas o flechas y tener un datos asociados, dependiendo del tipo de grafo que se esté tratando.



Ejercicio



- 1) Enliste todos los vértices del siguiente grafo
- 2) Enliste todas las aristas del siguiente grafo
- 3) Teniendo en cuenta que los grafos se definen en términos de conjuntos, proponga una manera de definir el siguiente grafo.

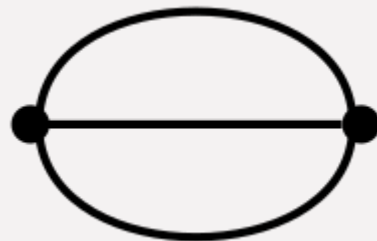
Terminología

Algunos tipos de aristas son:

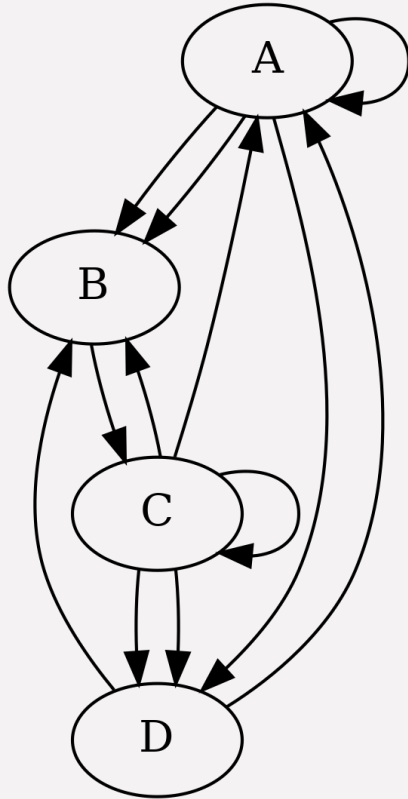
Rama: si una arista es la única que conecta dos vértices en **una dirección** se le conoce como rama.

En paralelo: si hay varias aristas conectando el mismo par de nodos **en la misma dirección** se dice que son aristas en paralelo.

Lazo o Bucle: si una arista conecta un vértice consigo mismo es considerada un bucle



Ejercicio



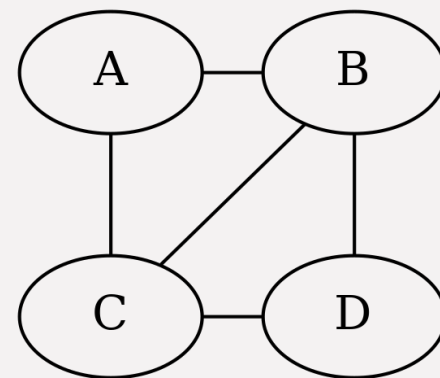
- 1) Identifique las aristas en paralelo del grafo
- 2) Identifique las aristas bucle del grafo
- 3) Las aristas que conectan al vértice B con el vértice C son ramas o son paralelas, porqué?

Terminología

Orden de un grafo ($|V|$): se refiere a la cantidad de vértices que conforman el grafo.

Tamaño de un grafo ($|A|$ o $|E|$): Es la cantidad total de aristas que tiene un grafo.

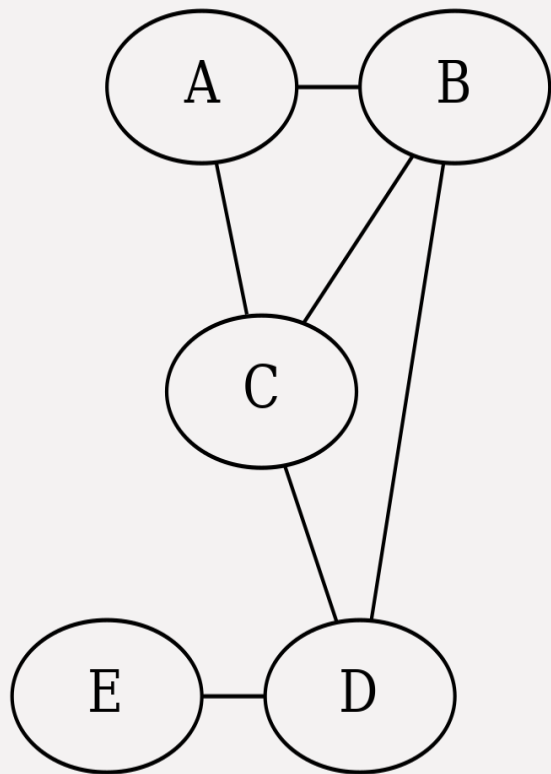
Grado de un vértice ($g(v)$ o $\deg(v)$): Es la cantidad de aristas que le corresponden a un vértice en particular.



Para este grafo....

- $|V| = 4$
- $|A| = 5$
- $g(C) = 3$

Ejercicio



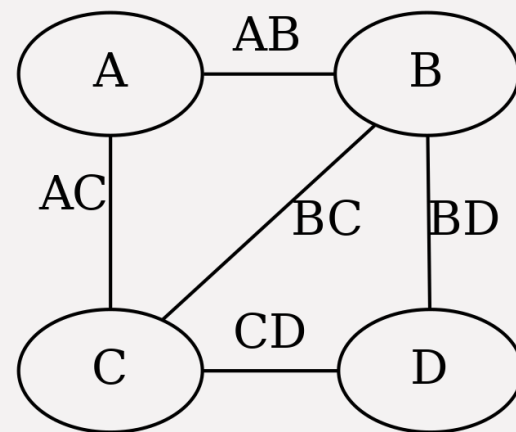
Sea el grafo anterior definido como $G(V,A)$ con $V=\{A,B,C,D,E\}$ y $A=\{\{A,C\},\{A,B\},\{B,C\},\{C,D\},\{B,D\},\{E,D\}\}$

- 1) Calcule el orden del grafo, **$|V|$**
- 2) Calcule el tamaño del grafo, **$|A|$**
- 3) Calcule el grado del vértice A, **$g(A)$**
- 4) Calcule el grado del vértice C, **$g(C)$**
- 5)Cuál será la razón por la que el orden del grafo se escribe como **$|V|$** y el tamaño **$|A|$** ?

Terminología

Adyacencia: Es un término que se usa para describir la correspondencia vértice-vértice o arista-arista. Se refiere a que dos vértices están separados por una sola arista o a que dos aristas están separadas por un solo vértice.

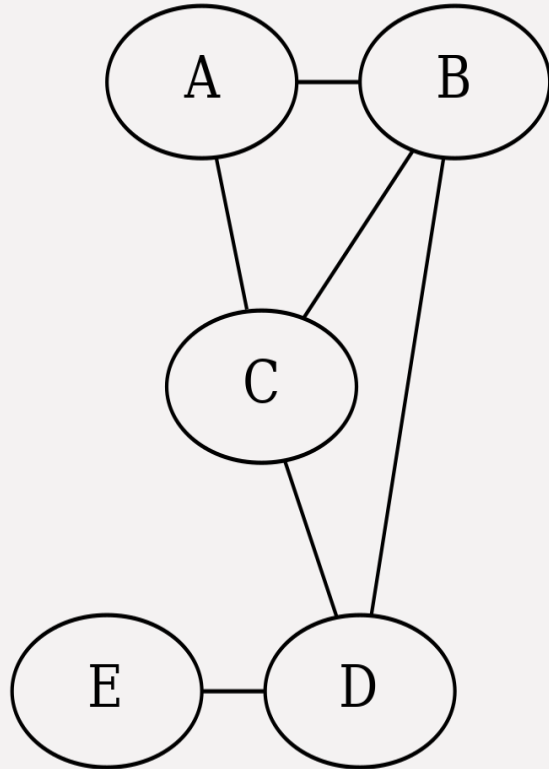
Incidencia: Se usa para describir la correspondencia entre un vértice y una arista. Si un vértice v corresponde a algún extremo de una arista a , se dice que a incide sobre v .



Para este grafo....

- BD incide en B
- BD incide en D
- C es adyacente a B
- AC es adyacente a AB

Ejercicio



- 1) Señale los vértices que son Adyacentes con el vértice A
- 2) Señale las aristas que inciden sobre el vértice B
- 3) El vértice C es incidente con el vértice A?
Por qué sí o por qué no?

Ejercicio

En relación al siguiente grafo

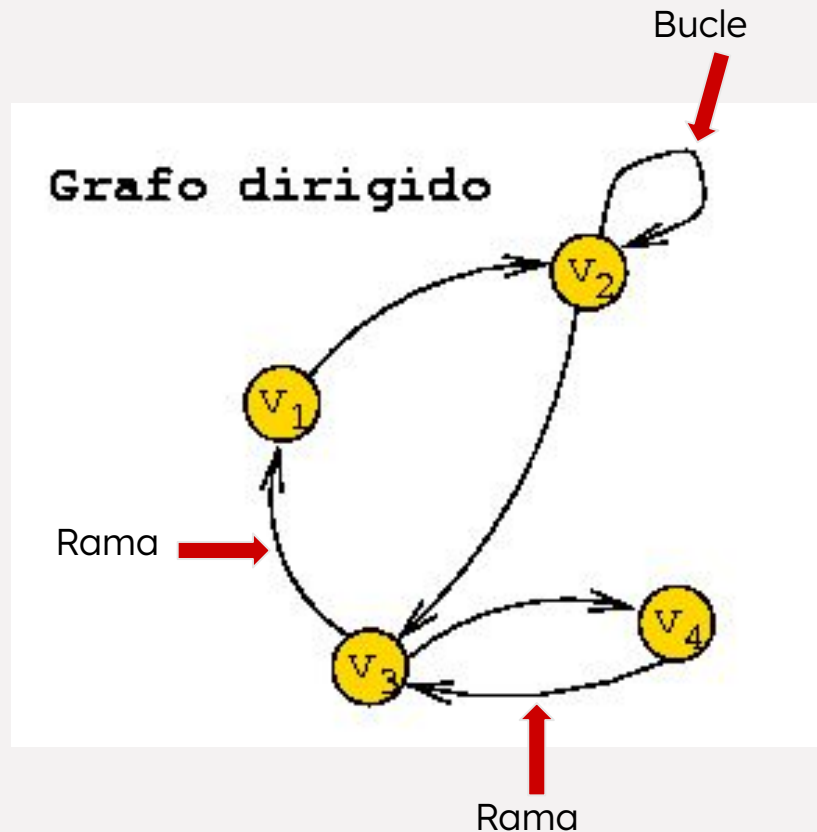
Que representa el siguiente conjunto?
 $V = \{v_1, v_2, v_3, v_4\}$

Que representa este otro conjunto?
 $A = \{(v_1, v_2), (v_2, v_2), (v_2, v_3), (v_3, v_1), (v_3, v_4), (v_4, v_3)\}$

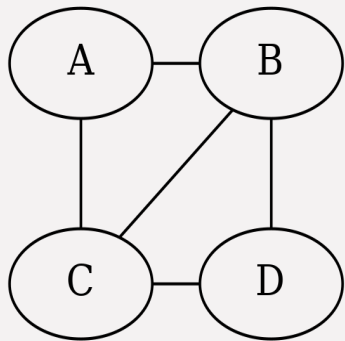
Mencione que tipo de arista es al que apunta a cada flecha roja en el diagrama

Mencione:

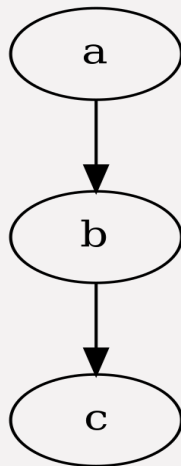
- Un par incidente
- Un par de vertices adyacentes
- Un par de aristas adyacentes



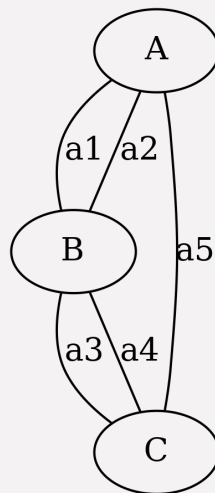
Tipos de grafos de acuerdo a su definición



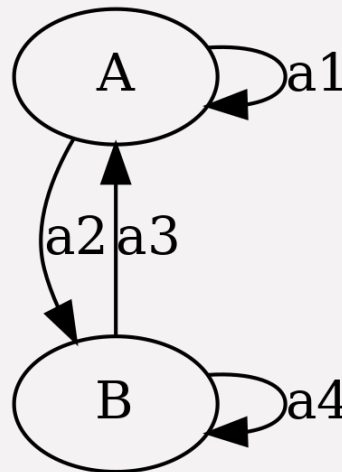
**Grafos
(no dirigidos)**



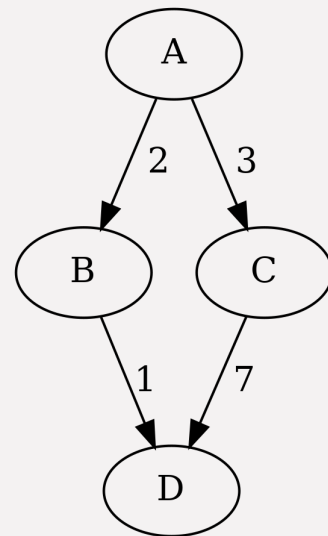
Digrafos



Multigrafos



Pseudografos



**Grafos
Ponderados**

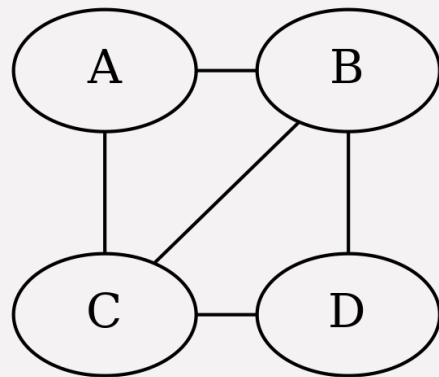
Grafos no Dirigidos

También llamados simplemente grafos, **sus aristas no tienen alguna dirección en específico**, es decir una arista que incide en v_1 y v_2 lleva tanto de v_1 a v_2 como de v_2 a v_1 . Se define como:

Grafo $:= G(V, A)$ donde $A = \{\{x, y\} \mid x, y \in V \wedge x \neq y\}$

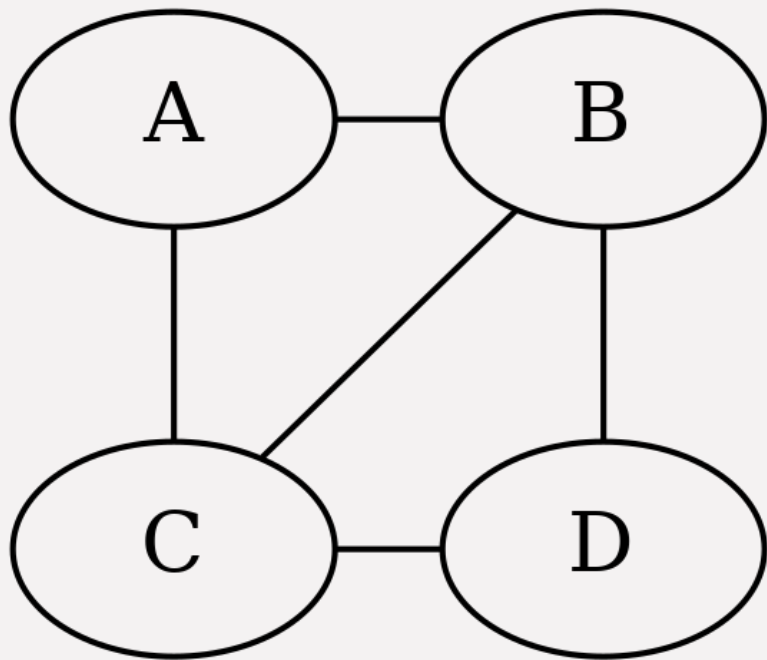
Es decir, las aristas del grafo se definen como pares **no ordenados** en los cuales los vértices deben de ser diferentes uno del otro. Por lo tanto

- El grafo no puede tener aristas bucles y debido a que A es un conjunto
- El grafo no puede tener aristas en paralelo



Ejemplo

Defina el siguiente grafo no dirigido matemáticamente



Grafo:= $G(V,A)$ tal que:

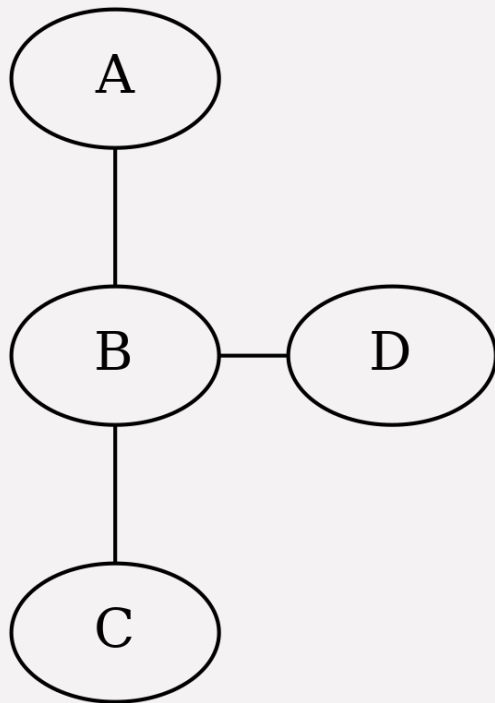
$$V=\{A, B, C, D\}$$

$$A=\{\{A,B\}, \{B,D\}, \{C,D\}, \{A,C\}, \{B,C\}\}$$

$$A=\{(A,B),(B,A),(B,D),(D,B), \\ (C,D),(D,C),(A,C),(C,A),(B,C),(C,B)\}$$

Ejercicio

Defina el siguiente grafo no dirigido matemáticamente



Grafo:= $G(V,A)$ tal que:

$$V=\{A, B, C, D\}$$

$$A=\{\{A,B\}, \{B,C\}, \{B,D\}\}$$

$$A=\{(A,B),(B,A),(B,C),(C,B),(B,D), (D,B)\}$$

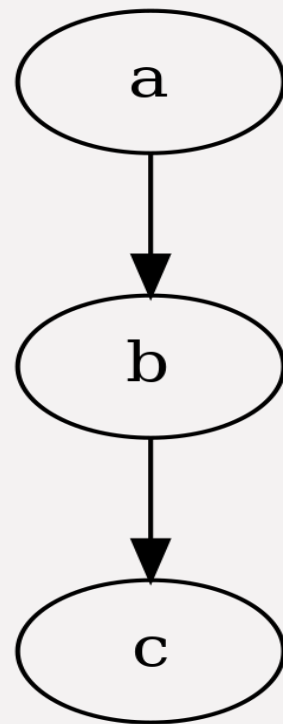
Digrafos

También llamados grafos dirigidos, son grafos en los que **cada arista tiene una dirección específica** por lo que $(v_1, v_2) \neq (v_2, v_1)$. La definición es muy similar a la de los grafos no dirigidos solo que usa pares ordenados para marcar esta distinción.

Grafo := $G(V, A)$ donde $A = \{(x, y) \mid (x, y) \in V^2 \wedge x \neq y\}$

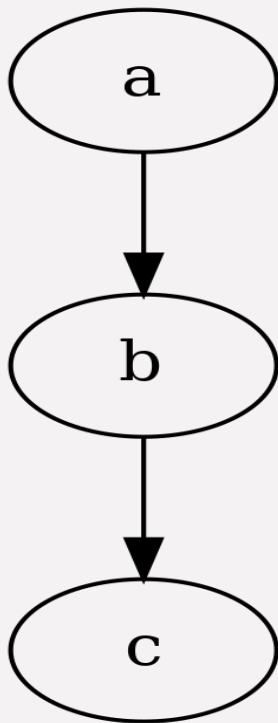
Por la definición del conjunto A tenemos que al igual a los grafos no dirigidos:

- El grafo no puede tener aristas bucle
- El grafo no puede tener aristas en paralelo



Ejemplo

Defina el siguiente digrafo matemáticamente



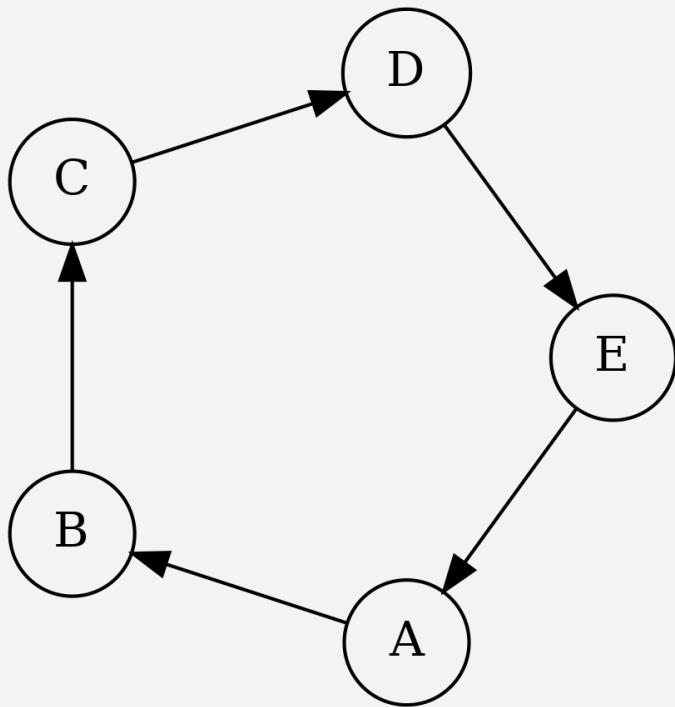
Grafo:= $G(V,A)$ tal que:

$$V=\{a, b, c\}$$

$$A=\{(a,b), (b,c)\}$$

Ejercicio

Defina el siguiente digrafo matemáticamente



Grafo:= $G(V,A)$ tal que:

$$V=\{A, B, C, D, E\}$$

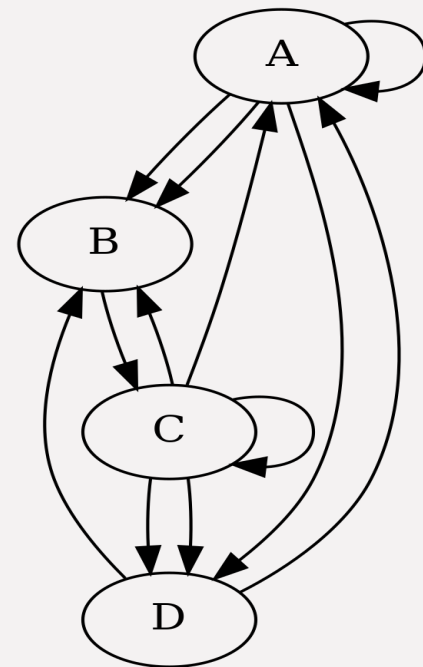
$$A=\{(A,B), (B,C), (C,D), (D,E), (E,A)\}$$

Adyacencia e Incidencia para Grafos Dirigidos

Los conceptos de adyacencia e incidencia se adaptan cuando un grafo es dirigido ya que **estas dos relaciones se vuelven asimétricas**.

Si una arista $a=(v1, v2)$ va del vertice $v1$ al vertice $v2$ se dice que $v1$ es adyacente a $v2$. No necesariamente $v2$ es adyacente a $v1$. $v2$ es adyacente a $v1$ si y sólo si existe otra arista $b=(v2,v1)$

Básicamente **el vértice de inicio es adyacente a el vértice fin pero no aplica alrevez**, a menos que exista una arista en donde la relación sea de fin a inicio



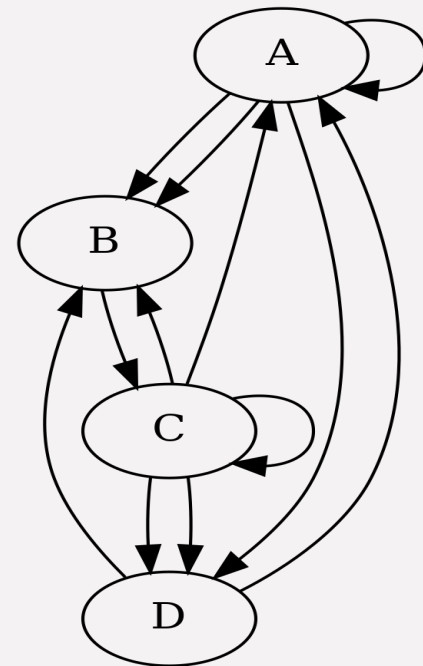
*D es adyacente a B
pero B no es adyacente
a D*

Adyacencia e Incidencia para Grafos Dirigidos

Para la incidencia también se tiene que tomar en cuenta la dirección. Sea una arista $a=(v1,v2)$ **tanto $v1$ como $v2$ siguen siendo incidentes sobre a pero se hace la distinción que:**

- a incide sobre $v1$ como vertice de entrada
- a incide sobre $v2$ como vértice de salida

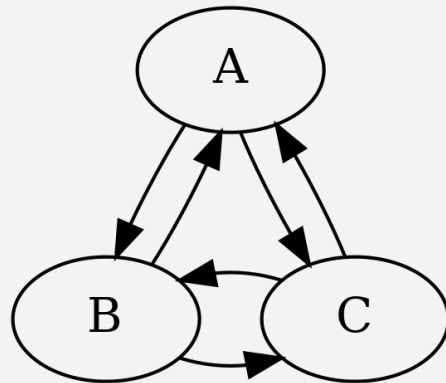
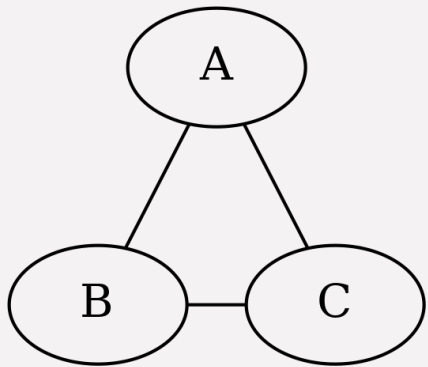
Basicamente, **la incidencia sigue aplicando para ambos extremos pero ahora podemos distinguir para qué extremo aplica**



*D es adyacente a B
pero B no es adyacente
a D*

Generalidad de los Digrafos Sobre Grafos no Dirigidos

Cualquier algoritmo o situación que se puede modelar con un grafo no dirigido se puede modelar también con un dígrafo. En vez de que una arista conecte a dos vértices, hay dos aristas, una del primer vértice al segundo y otra del segundo al primero.



Ventajas de los Grafos no Dirigidos Sobre los Digrafos

Si todos los problemas que se pueden solucionar mediante grafos no dirigidos se pueden representar mediante digrafos, porque usar grafos no dirigidos?

Los grafos no dirigidos son más sencillos de visualizar que los grafos. Si la relación entre los vértices de un grafo no es dirigida, es mejor generar una solución a partir de un grafo no dirigido ya que es la estructura que más se adecua a el problema. Esto **nos permite llegar a una solución más elegante y entendible** evitando complejidades innecesarias.



Ventajas de los Grafos no Dirigidos Sobre los Digrafos



También, al no tener que representar dirección, una implementación optimizada de **un grafo no dirigido siempre ocupará menos memoria que la representación equivalente usando un grafo dirigido** (Aproximadamente la mitad).

Es similar a preguntar porque se elegiría una lista simplemente encadenada sobre una lista doblemente encadenada.

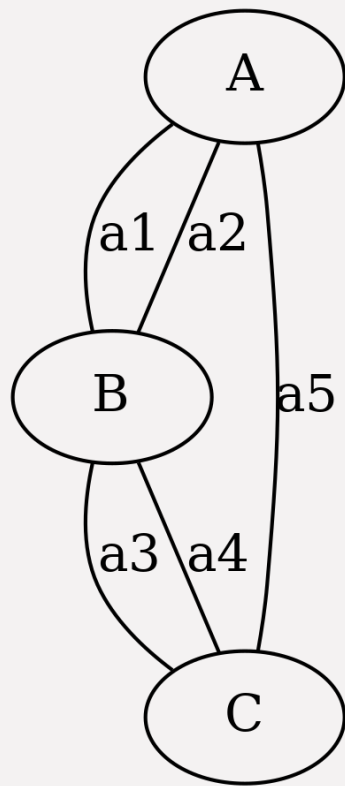
Multigrafos

Tanto los grafos no dirigidos como los dirigidos pueden expandir su definición para **permitir que un par de nodos tengan más de una sola arista conectandolos**.

Cuando un grafo no es un multigrafo (solo permite arcos rama), **se dice que es un *grafo simple***.

Las aristas de un multigrafo ya no se pueden definir como pares (ordenados o no) **de vértices** elementos de un conjunto A ya que por definición los conjuntos no aceptan elementos repetidos.

Para solucionar este problema **las aristas se convierten en elementos atómicos del grafo** y dejan de estar definidas en términos de los vértices. Esto significa en terminos practicos que cada arista tiene que poder identificarse individualmente.



Multigrafos

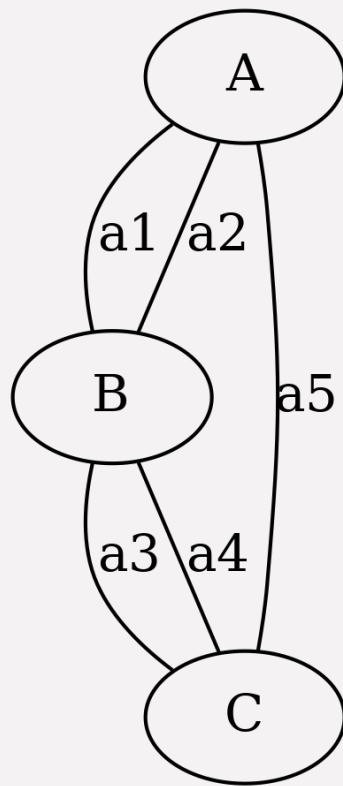
La definición se adapta de la siguiente manera:

Grafo := $G(V, A, \dots, \phi)$ donde V y A son ambos conjuntos de elementos arbitrarios y ϕ es una función que mapea cada arista elemento de A con un par ordenado (o no ordenado dependiendo el tipo de grafo) que representa que vertices conecta esa arista.

$\phi: A \rightarrow \{\{x,y\} \mid x,y \in V \wedge x \neq y\}$ multigrafos no dirigidos

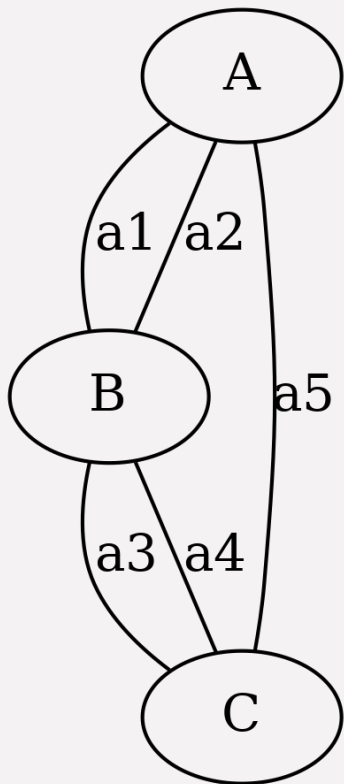
$\phi: A \rightarrow \{(x,y) \mid (x,y) \in V^2 \wedge x \neq y\}$ multigrafos dirigidos

El punto es que esto **permite que existan vértices en paralelo** pero sigue prohibiendo los bucles.



Ejemplo

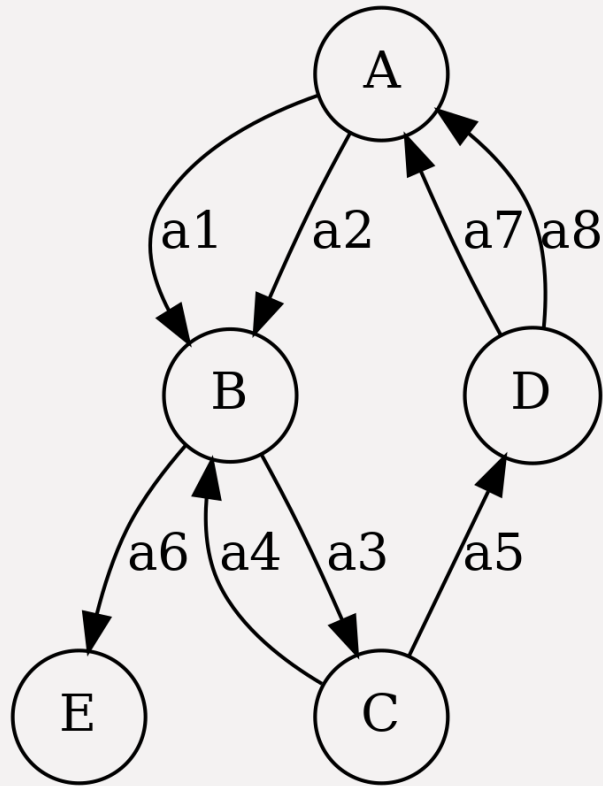
Defina el siguiente multigrafo matemáticamente



Grafo: $= G(V, A, \phi)$ tal que:

- $V = \{A, B, C\}$
- $A = \{a1, a2, a3, a4, a5\}$
- $\phi(a)$ es una función tal que:
 $\phi(a1) = \{A, B\}$
 $\phi(a2) = \{A, B\}$
 $\phi(a3) = \{B, C\}$
 $\phi(a4) = \{B, C\}$
 $\phi(a5) = \{C, A\}$

Ejercicio



Grafo: $=G(V,A,\phi)$ tal que:

- $V=\{A,B,C,D,E\}$
- $A=\{a1,a2,a3,a4,a5,a6,a7,a8\}$
- $\phi(a)$ es una función tal que:
 - $\phi(a1) = (A,B)$
 - $\phi(a2) = (A,B)$
 - $\phi(a3) = (B,C)$
 - $\phi(a4) = (C,B)$
 - $\phi(a5) = (C,D)$
 - $\phi(a6) = (B,E)$
 - $\phi(a7) = (D,A)$
 - $\phi(a8) = (D,A)$

Pseudografos

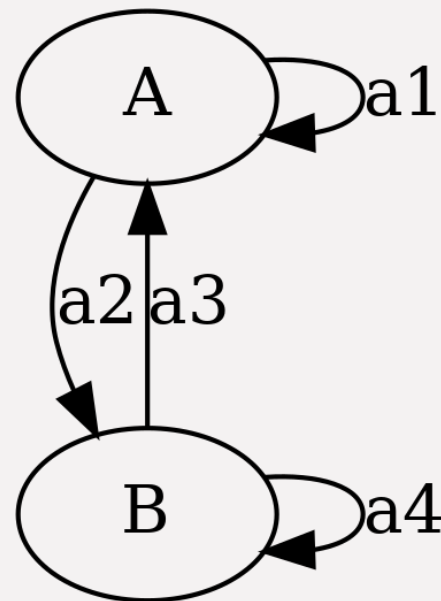
Un pseudografo es un **multigrafo que permite bucles**.

Estos son los grafos más generales en cuanto a relaciones entre elementos ya que permiten eliminar las dos restricciones previamente mencionadas. Se definen igual que los multigrafos excepto que incluye aristas con vértices duplicados.

Grafo := $G(V, A, \dots, \phi)$

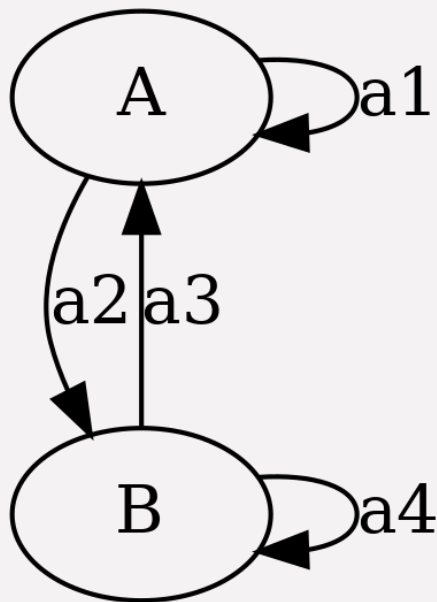
$\phi: A \rightarrow \{\{x,y\} \mid x,y \in V\}$ pseudografos no dirigidos

$\phi: A \rightarrow \{(x,y) \mid (x,y) \in V^2\}$ pseudografos dirigidos



Ejemplo

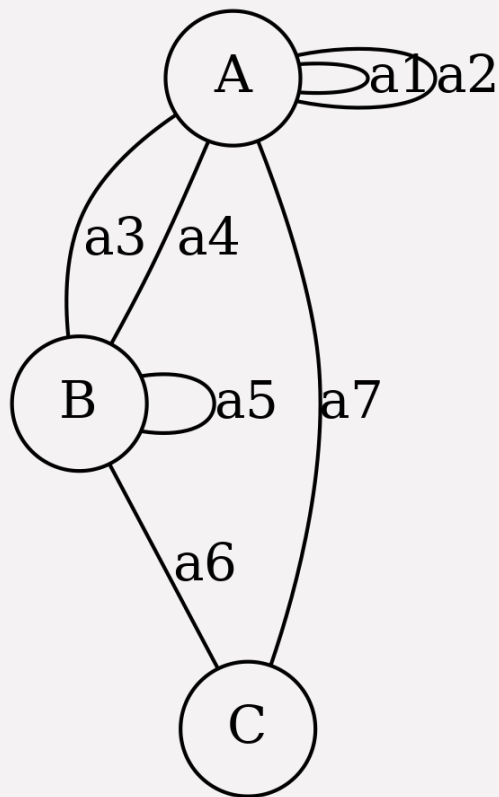
Defina el siguiente pseudografo matemáticamente:



Grafo:= $G(V, A, \phi)$ tal que:

- $V=\{A,B\}$
- $A=\{a1,a2,a3,a4\}$
- $\phi(a)$ es una función tal que:
 $\phi(a1) = (A,A)$
 $\phi(a2) = (A,B)$
 $\phi(a3) = (B, A)$
 $\phi(a4) = (B, B)$

Ejercicio



Grafo:= $G(V, A, \phi)$ tal que:

- $V=\{A,B,C\}$
- $A=\{a1,a2,a3,a4,a5,a6,a7\}$
- $\phi(a)$ es una función tal que:
 $\phi(a1) = \{A,A\}$
 $\phi(a2) = \{A,A\}$
 $\phi(a3) = \{A,B\}$
 $\phi(a4) = \{A,B\}$
 $\phi(a5) = \{B,B\}$
 $\phi(a6) = \{B,C\}$
 $\phi(a7) = \{A, C\}$

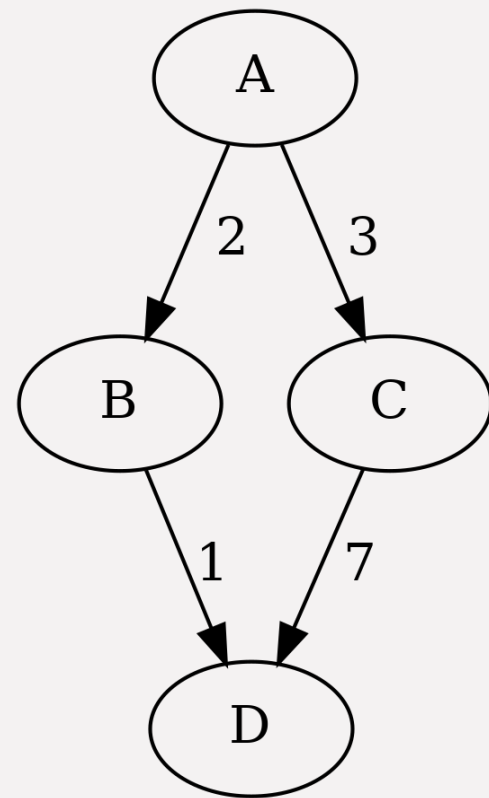
Grafos Ponderados

Un grafo ponderado es un grafo el cual **asigna un valor a cada uno de sus arcos** para representar un valor relacionado a la conexión entre los dos vértices. La definición se ajusta para incluir una función **w** con este mismo propósito.

Grafo := $G(V, A, \dots, w)$ con $w: A \rightarrow R$ tal que R es cualquier conjunto que represente los valores a asignar, por ejemplo $R=\mathbb{R}$ o $R=\mathbb{N}$.

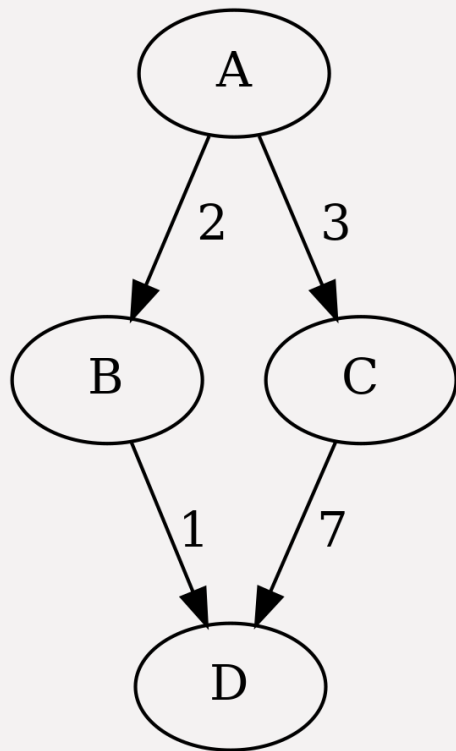
Al valor **$w(a)$** donde a es una arista cualquiera del grafo, se le conoce como el **peso del arco a**

Por ejemplo para el ejemplo de la izquierda: $w((C,D))=7$



Ejemplo

Defina el siguiente digrafo ponderado matemáticamente

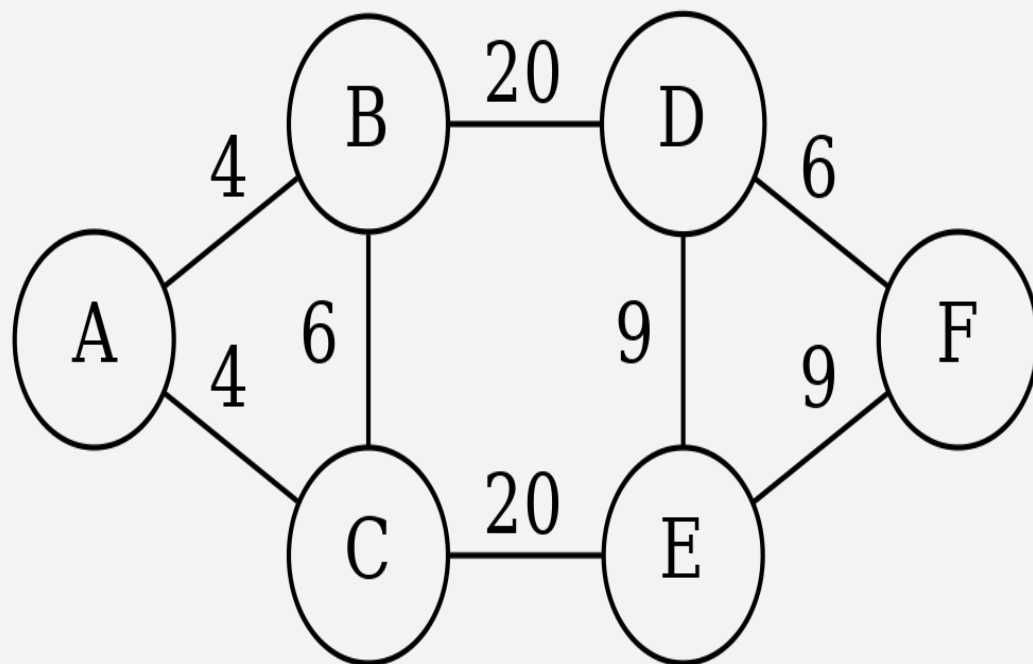


Grafo:= $G(V,A,w)$ tal que:

- $V=\{A,B,C,D\}$
- $A=\{(a,b),(a,c),(b,d),(c,d)\}$
- $w(a)$ es una función tal que:
 $w((a,b)) = 2$
 $w((a,c)) = 3$
 $w((b,d)) = 1$
 $w((c,d)) = 7$

Ejercicio

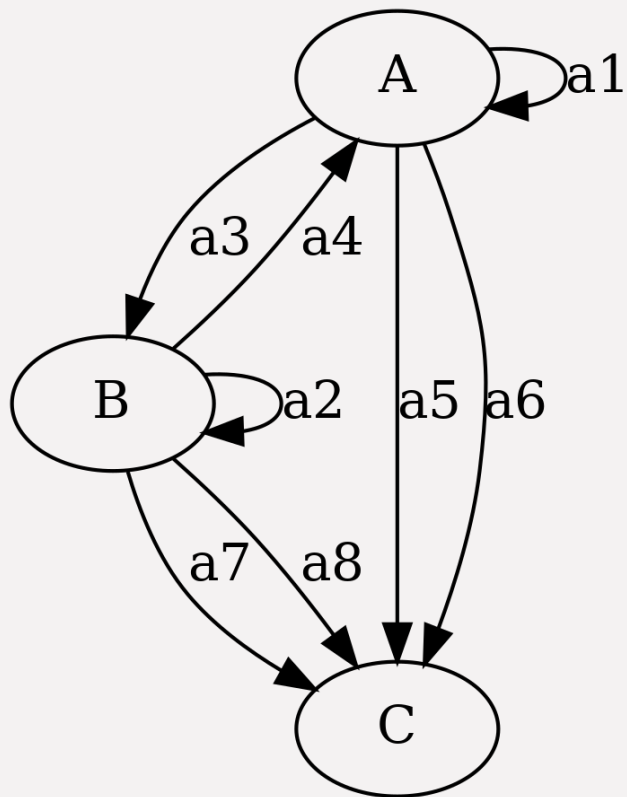
Defina el siguiente grafo no dirigido ponderado matemáticamente



Grafo:= $G(V,A,w)$ tal que:

- $V=\{A,B,C,D,E,F\}$
- $A=\{\{A,B\}, \{A,C\}, \{B,C\}, \{B,D\}, \{C,E\}, \{D,E\}, \{D,F\}, \{E,F\}\}$
- $w(a)$ sea una funcion tal que:
 $w(\{A,B\}) = w(\{A,C\}) = 4$
 $w(\{B,D\}) = w(\{C,E\}) = 20$
 $w(\{B,C\}) = w(\{D,F\}) = 6$
 $w(\{D,E\}) = w(\{E,F\}) = 9$

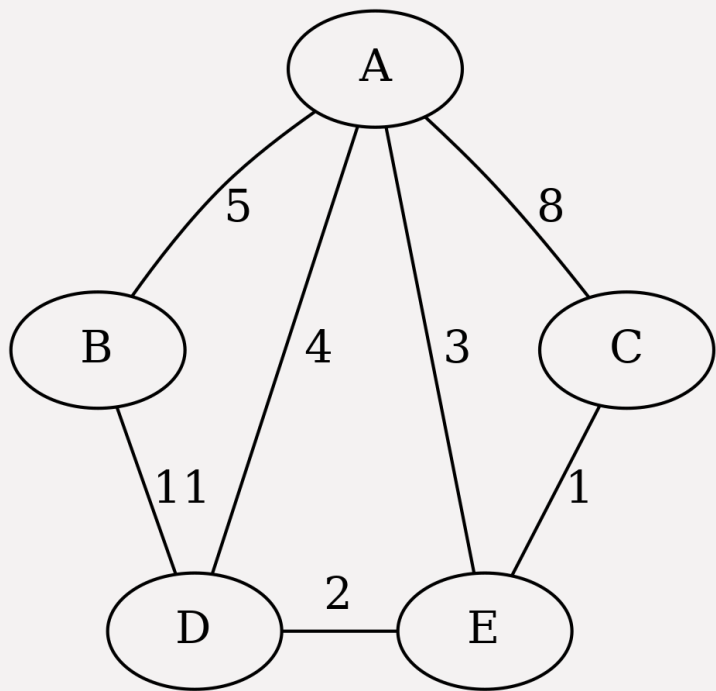
Ejercicios Generales



Para el grafo anterior:

- ☐ Identifique que tipo de grafo es
- ☐ Defina el grafo
- ☐ Calcule el orden del Grafo
- ☐ Calcule el tamaño del Grafo
- ☐ Extraiga el subconjunto de aristas lazo
- ☐ Extraiga el subconjunto de aristas en paralelo
- ☐ Mencione un par de vértices adyacentes
- ☐ Mencione un par de aristas adyacentes
- ☐ Mencione un par incidente

Ejercicios Generales



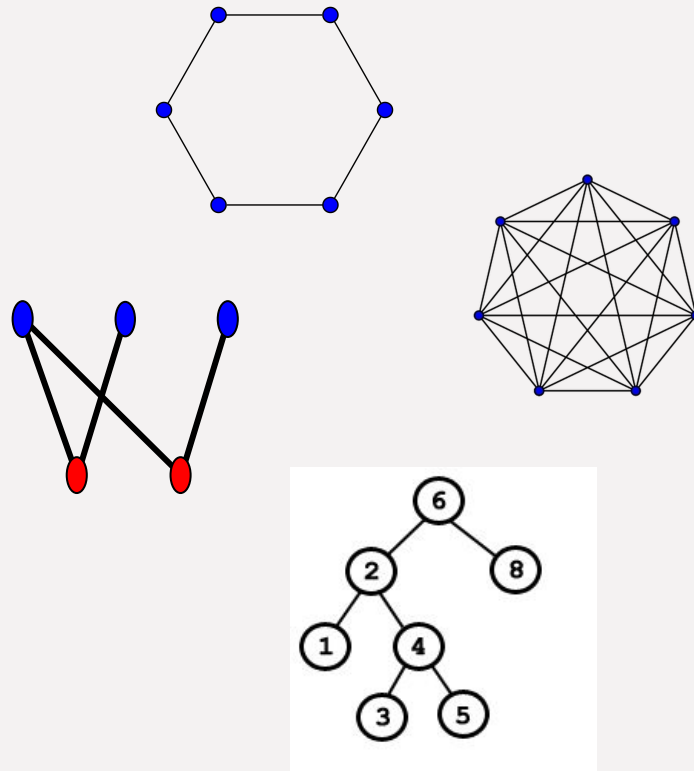
Para el grafo anterior:

- ☐ Identifique que tipo de grafo es
- ☐ Defina el grafo
- ☐ Calcule el orden del Grafo
- ☐ Calcule el tamaño del Grafo
- ☐ Extraiga el subconjunto de aristas lazo
- ☐ Extraiga el subconjunto de aristas en paralelo
- ☐ Mencione un par de vértices adyacentes
- ☐ Mencione un par de aristas adyacentes
- ☐ Mencione un par incidente

Tipos de grafos especiales

A veces **es posible clasificar un grafo** no solo por la manera en la que está definido sino **por algunos comportamientos particulares que exhiben sus vértices y aristas**.

A este tipo de grafos se les conoce como **grafos especiales** y suelen estar asociados con un tipo de comportamiento en particular.

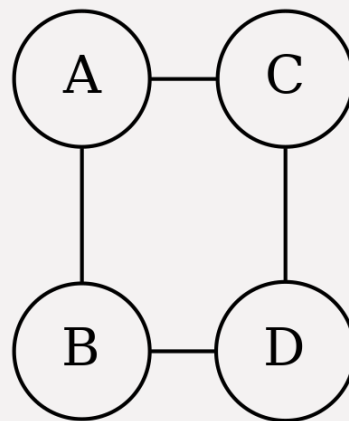


Grafo Conexo

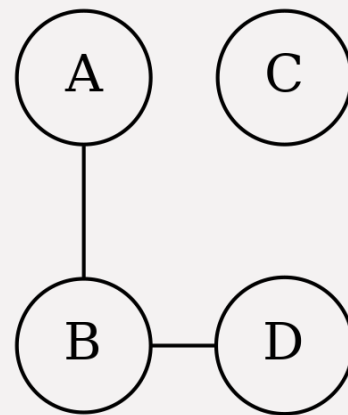
Un grafo es conexo si para cualquier par de vértices existe una manera de llegar a de uno al otro.

Un grafo no conexo se compone de dos o más grafos aislados que no están conectados entre sí por ninguna secuencia de vértices y aristas

Grafo Conexo



Grafo no Conexo

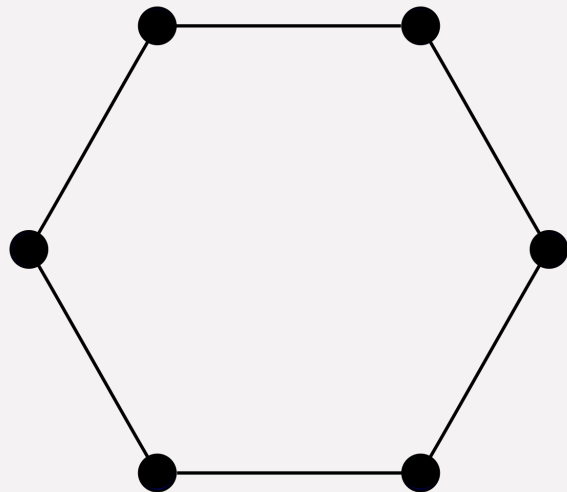


Grafo ciclo

Camino cerrado en el que **no se repite ningún vértice**, excepto el primero, que aparece tanto al inicio como al final del recorrido.

Para un grafo cíclico con **n vértices**, se utiliza la notación **C_n** , donde:

- **El número de aristas es igual a n .**
- **Cada vértice tiene un grado de 2**, ya que está conectado a dos vecinos.



Características

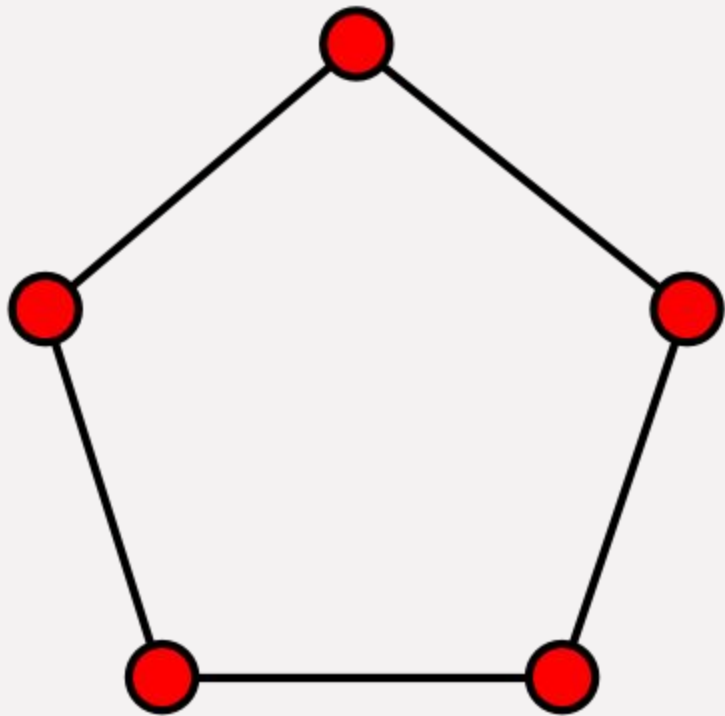
Los **grafos cíclicos** se caracterizan porque **todos sus vértices tienen un grado de exactamente dos**, lo que significa que cada vértice está conectado a dos vecinos: **uno anterior y otro siguiente en el recorrido**.

Debido a esta propiedad, estos grafos **son simétricos** y suelen representarse en forma de **polígonos regulares**, donde:

- **Los vértices** corresponden a los puntos del polígono.
- **Las aristas** representan los lados que los conectan.

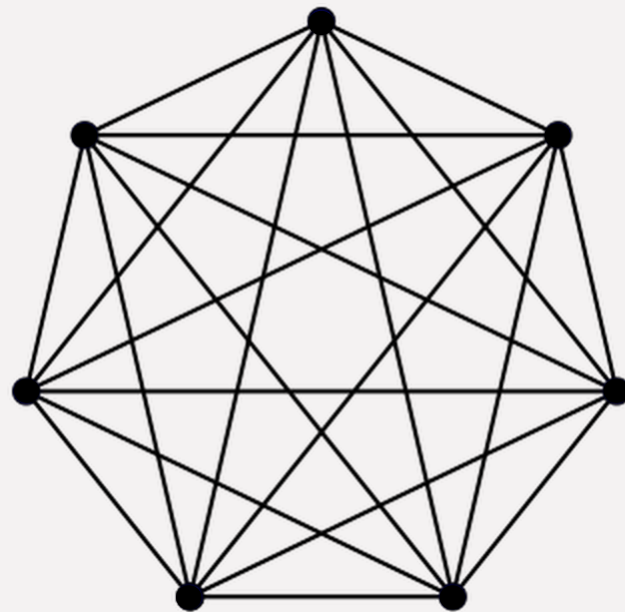
Además, los **grafos cíclicos son conexos**, lo que implica que **siempre existe un camino entre cualquier par de vértices dentro del grafo**.

Ejemplo



Grafo completo

Es aquel en el que **cada vértice está conectado a todos los demás vértices** del grafo. Se denota como K_n , donde n representa el número de vértices.



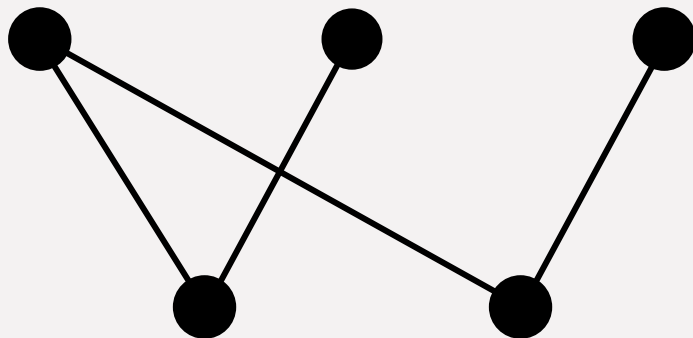
Características

- **Cada vértice tiene un grado de $(n - 1)$, ya que está conectado a todos los demás.**
- **El número de arcos está dado por: $n(n - 1) / 2$**
- **Estos grafos forman estructuras completamente interconectadas.**

Grafo bipartito

Un **grafo bipartito** es un **grafo simple** en el que los vértices pueden separarse en **dos conjuntos disjuntos** V_1 y V_2 , de tal forma que:

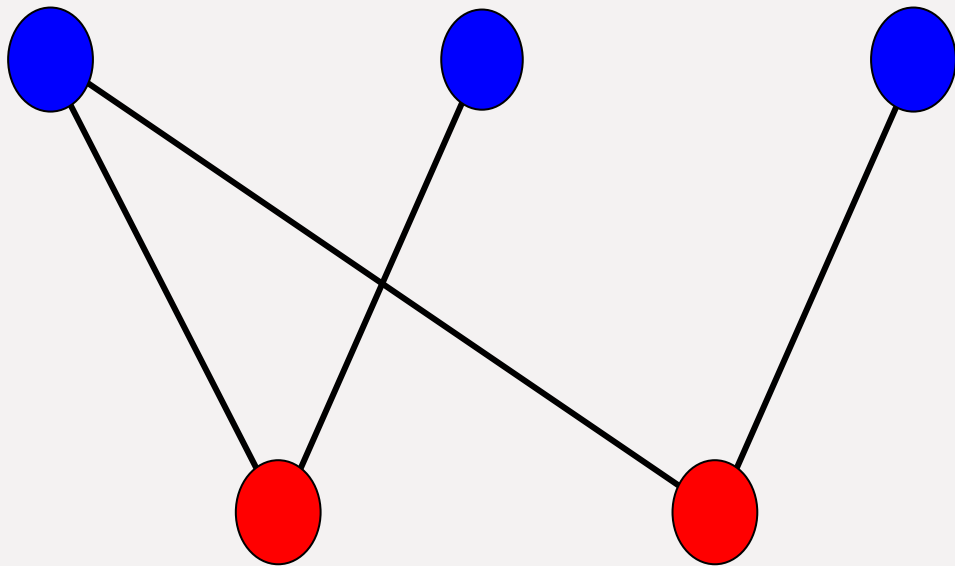
- Los vértices en V_1 solo están conectados con vértices en V_2 .
- No existen aristas entre vértices del mismo conjunto.



Características

Un **grafo es bipartito** si sus vértices pueden colorearse con **dos colores** de manera que **ninguna arista conecte vértices del mismo color**.

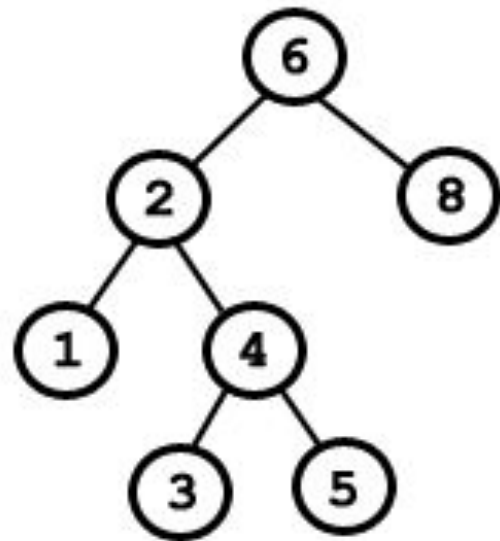
Ejemplo



Grafos Arbol

La definición de los grafos nos permite construir estructuras que tengan las mismas propiedades que un árbol.

En Teoría de Grafos, un árbol es un tipo de grafo donde cualquier par de vértices están conectados por **un solo camino**.

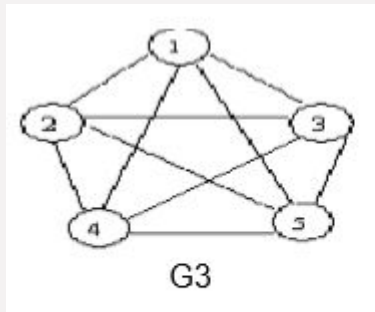


Características

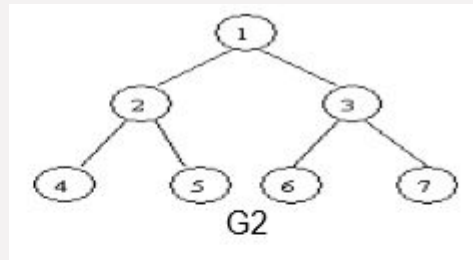
- **Es conexo:** Ya que existe un camino entre cualquier par de nodos.
- **No tiene ciclos:** No hay una secuencia de aristas que regrese al mismo nodo sin repetir aristas.
- Para n vertices, debe tener **$n - 1$ aristas.**

Ejercicio

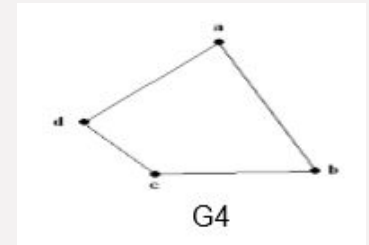
Identifique qué tipo de grafo especial es cada uno de los grafos siguientes



Grafo completo



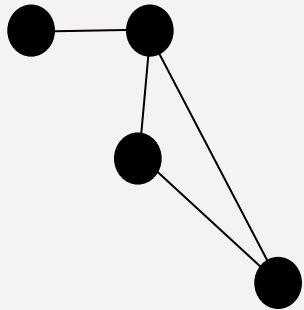
Grafo Arbol



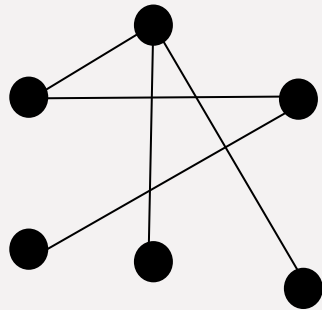
Grafo Ciclo

Ejercicio

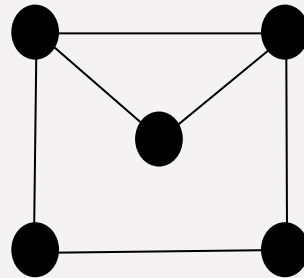
Identifica cuál de los siguientes grafos es bipartito



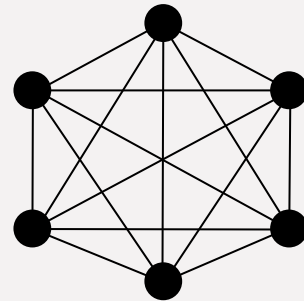
No es bipartito



Bipartito



No es bipartito

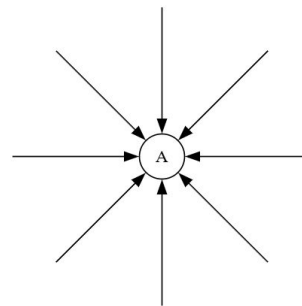
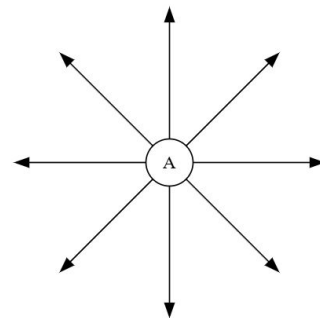


No es bipartito

Grados de Entrada y Salida de un Vértice

El grado de un vértice se define como el **número de aristas que están conectadas a este.**

En un **grafo dirigido**. Existen dos tipos de grados: el **grado de entrada** y el **grado de salida**.

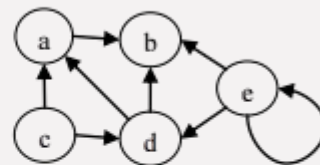


Grados de Entrada y Salida de un Vértice

El **grado de entrada** se refiere a la cantidad de aristas dirigidas hacia un vértice.

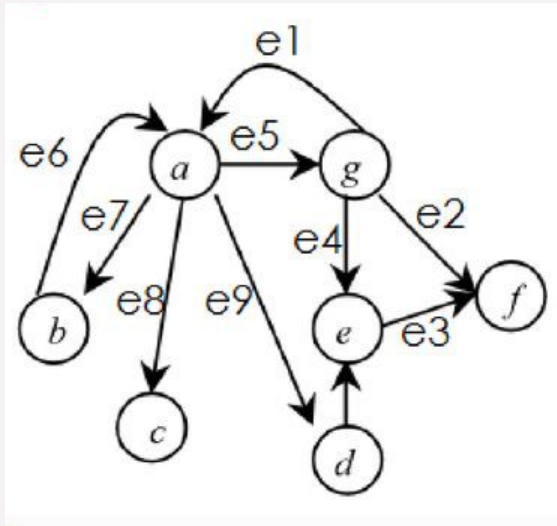
Mientras que el **grado de salida** indica la cantidad de aristas dirigidas en dirección contraria a él

Grafo dirigido:



GradoE (a) = 2	GradoS (a) = 1
GradoE (b) = 3	GradoS (b) = 0
GradoE (c) = 0	GradoS (c) = 2
GradoE (d) = 2	GradoS (d) = 2
GradoE (e) = 1	GradoS (e) = 3

¿Cuál es el grado de entrada y salida del grafo?

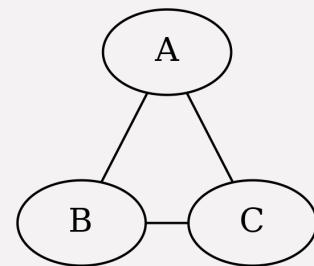


Nodo	Grado de entrada g-entrada	Grado de salida g-salida
a	2	4
b	1	1
c	1	0
d	1	1
e	2	1
f	2	0
g	1	3

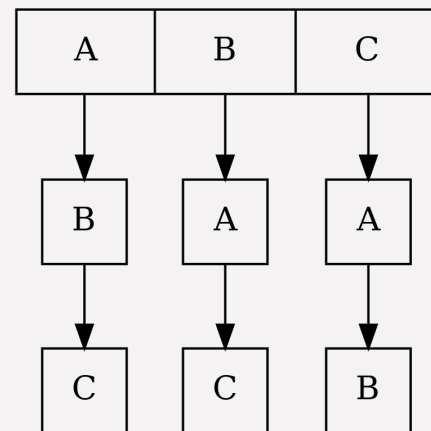
Formas de Representar un Grafo

Hasta ahora hemos visto que **los grafos se pueden representar gráficamente por medio de un dibujo, pero también existen otras maneras** de representarlos enfocadas en la manipulación de la información que contiene el grafo.

Las maneras que veremos nosotros se enfocarán en el concepto de **adyacencia**, específicamente, en la adyacencia de vértices.



$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$



Formas de Representar un Grafo

Recordatorio: Se dice que **dos vértices son adyacentes si existe un arco que los conecte directamente**.

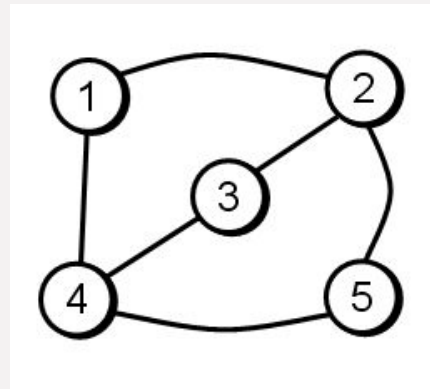
Las dos maneras de representar un grafo mediante la adyacencia de sus vértices son:

- Con una **Matriz de Adyacencia**
- Con una **Lista de Adyacencia**

Matriz de Adyacencia

Una **matriz de adyacencia** es una forma de representar un grafo utilizando una matriz, donde **las filas y las columnas corresponden a los vértices del grafo**.

Se escoge arbitrariamente uno de los ejes para representar el vértice de donde sale la arista y el otro para representar a donde llega. **La matriz de adyacencia un grafo $G(V,A)$ siempre será de $|V| \times |V|$**



M	1	2	3	4	5
1	0	1	0	1	0
2	1	0	1	0	1
3	0	1	0	1	0
4	1	0	1	0	1
5	0	1	0	1	0

Matriz para Grafos No Dirigido

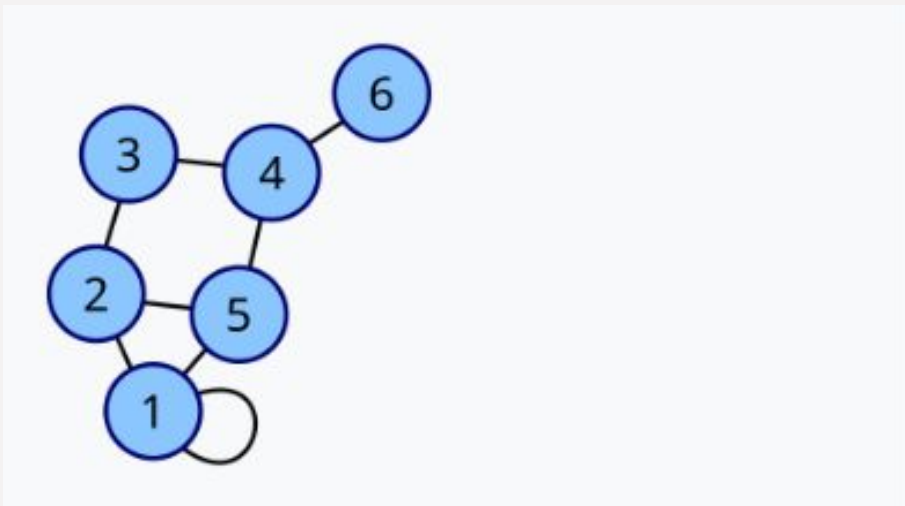
Para un **grafo no dirigido**, la matriz de adyacencia es **simétrica**.

La entrada en la posición $M(i,j)$ de la matriz representa la cantidad de arcos que conectan los vértices V_i y V_j , de la siguiente manera:

$$M(i,j) = \begin{cases} \text{Cantidad de arcos que conectan } V_i \text{ con } V_j & \text{si } V_i \neq V_j \\ \text{Cantidad de lazos de } V_i \text{ a } V_j, & \text{si } V_i = V_j \end{cases}$$

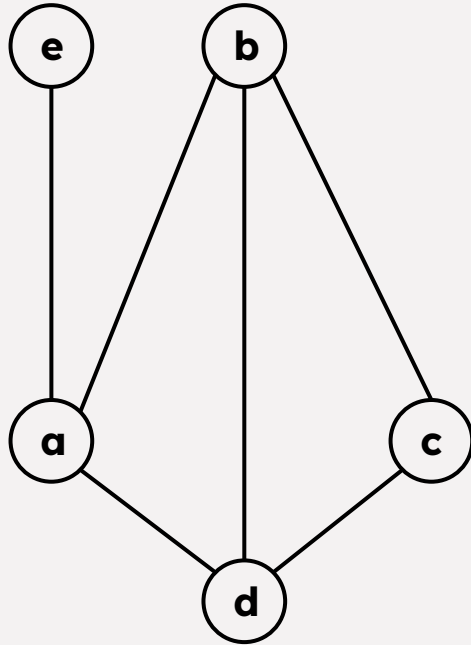
Si un vértice es aislado, su fila y columna correspondiente estarán llenas de ceros, indicando que no hay conexiones con otros vértices.

Ejemplo



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

Ejercicio



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

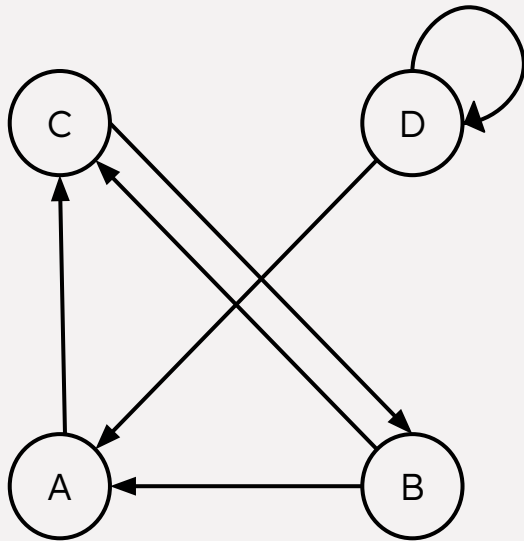
Matriz para Grafos Dirigidos

Para un **grafo dirigido**, la matriz de adyacencia cambia para reflejar la dirección de los arcos. En este caso, la entrada $M(i,j)$ de la matriz se define como:

$$M(i,j) = \begin{cases} \text{La cantidad de arcos que inician en } V_i \text{ y terminan en } V_j & \text{si } V_i \neq V_j \\ \text{Y la cantidad de lazos de } V_i \text{ a } V_j & \text{si } V_i = V_j \end{cases}$$

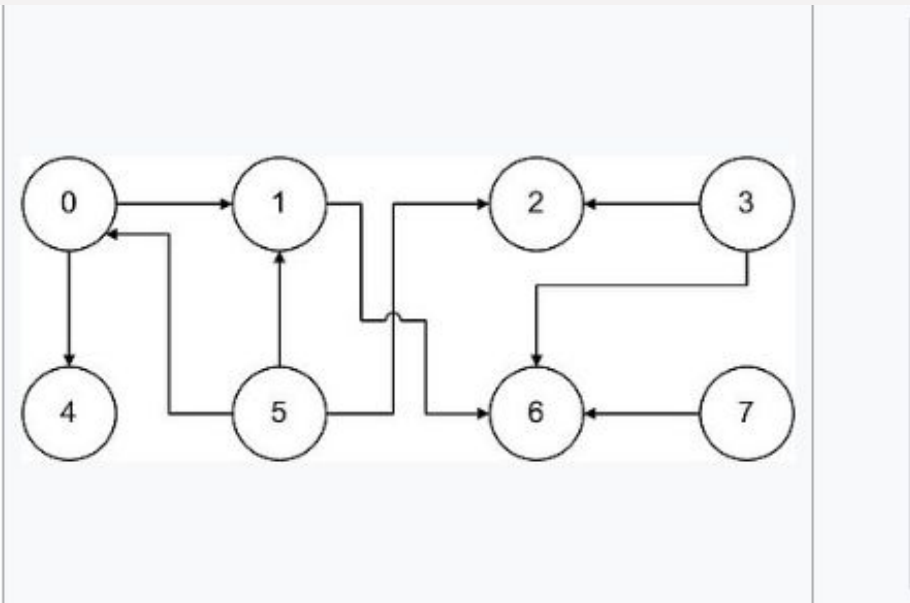
En un digrafo, las conexiones no son necesariamente simétricas, ya que un arco de V_i a V_j no implica necesariamente un arco de V_j a V_i .

Ejemplo



M	A	B	C	D
A	0	0	1	0
B	1	0	1	0
C	0	1	0	0
D	1	0	0	1

Ejercicio



M	0	1	2	3	4	5	6	7
0	0	1	0	0	1	0	0	0
1	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	1	0
4	0	0	0	0	0	0	0	0
5	1	1	1	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	1	0

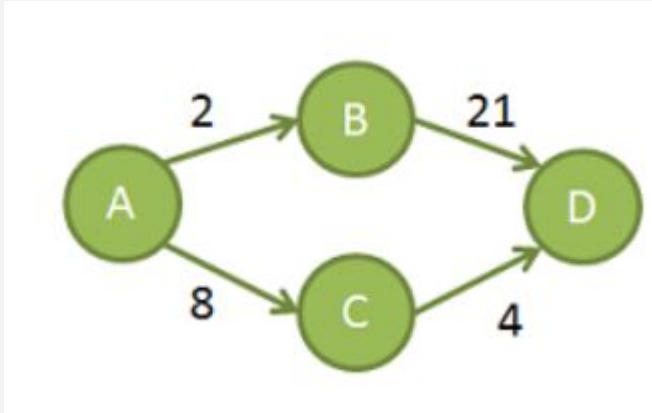
Matriz para Grafo Ponderado

La matriz de adyacencia puede ser modificada para reflejar los **pesos** o **distancias** de los arcos. En lugar de simplemente contar los arcos, se almacenan los valores de los pesos o distancias. En este tipo de representación, la entrada $M(i,j)$ puede tener los siguientes valores:

$$W(i,j) = \begin{cases} \text{Peso del arco de } V_i \text{ a } V_j, & \text{si existe arco } (V_i, V_j) \\ \infty, & \text{si no existe arco} \\ 0, & \text{si } V_i = V_j \end{cases}$$

∞ representa la **inexistencia de un arco** entre los vértices V_i y V_j , y un valor de 0 en la diagonal indica que no hay distancia.

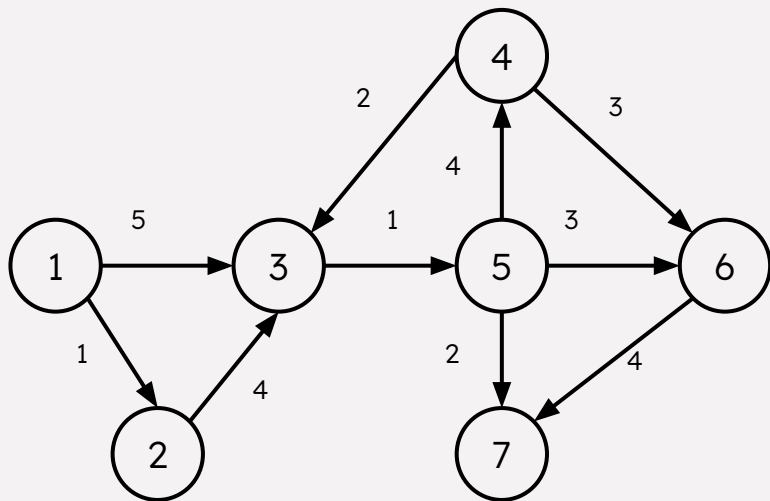
Ejemplo



M	A	B	C	D
A	0	2	8	∞
B	∞	0	∞	21
C	∞	∞	0	4
D	∞	∞	∞	0

Ejercicio

Construya la matriz de adyacencia del siguiente grafo dirigido

[illegible]

Lista de Adyacencia

Una lista de adyacencia, también conocida como una lista de vecinos, **es una forma de representar un grafo en la cual se tiene una lista de vértices adyacentes para cada vértice.**

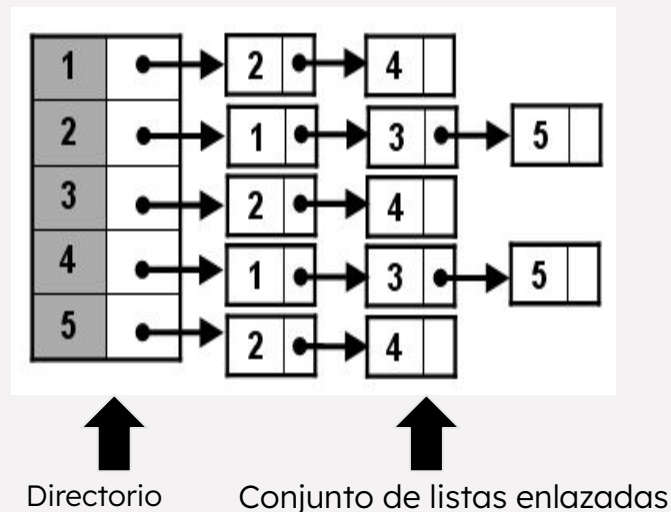
Esta representación es **recomendable cuando el número de aristas es significativamente menor a $|V|^2$** . El uso de la lista de adyacencia **permite ahorrar espacio de almacenamiento** para grafos con pocas aristas. Sin embargo, usar una lista en lugar de una matriz tiene la desventaja de que **el tiempo de búsqueda no es constante**, ya que se pierde el acceso directo que permite la matriz.

Lista de Adyacencia

Para generar esta representación **se necesita de dos partes: Un directorio y un conjunto de listas enlazadas.**

El directorio es un arreglo o lista, donde $\text{directorio}[i]$ es un apuntador a una lista enlazada que contiene los vértices (o referencias a los vertices) que están conectados al vértice i .

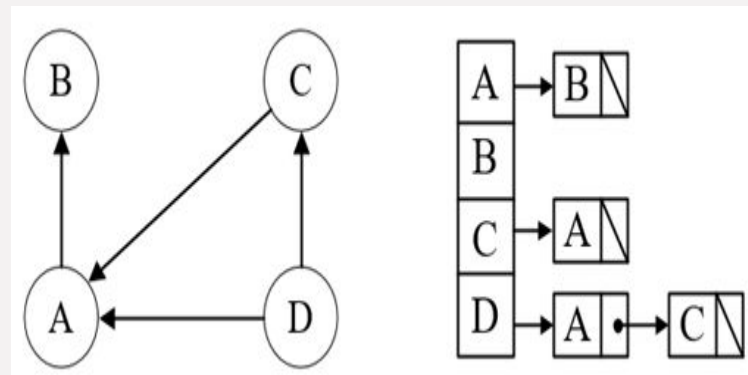
Cada registro de la lista enlazada tiene 2 campos: Un identificador y un enlace al siguiente elemento de la lista. **La lista enlazada representa las aristas que salen del nodo en cuestión.**



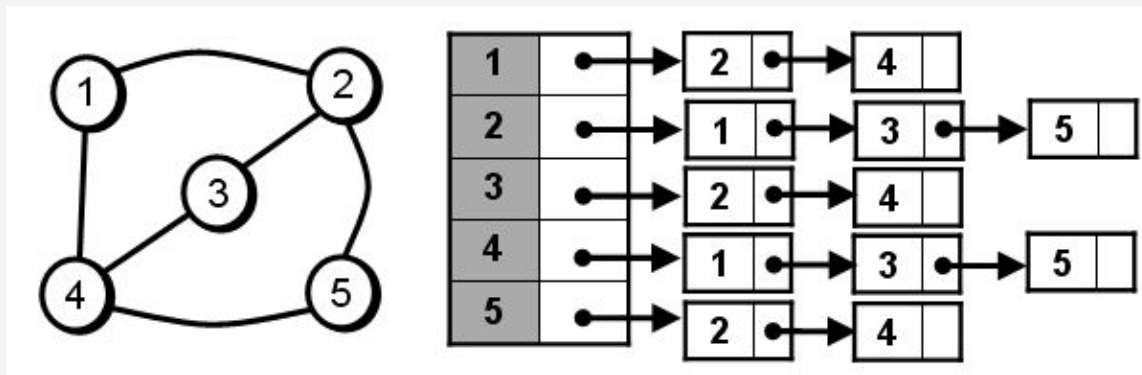
Lista de Adyacencia

La implementación de la lista de adyacencia es diferente cuando el grafo es dirigido a cuando es no dirigido

Cuando el grafo es dirigido, las conexiones entre los vértices tienen una conexión específica, esto quiere decir que en la lista de adyacencia **cada conexión se registra solo en la lista del vértice de origen**.



Lista de Adyacencia



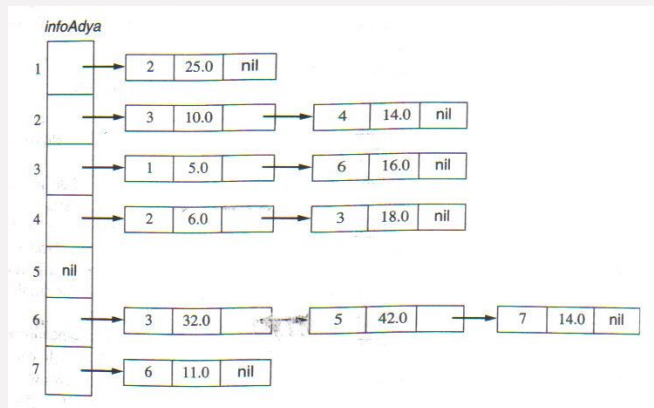
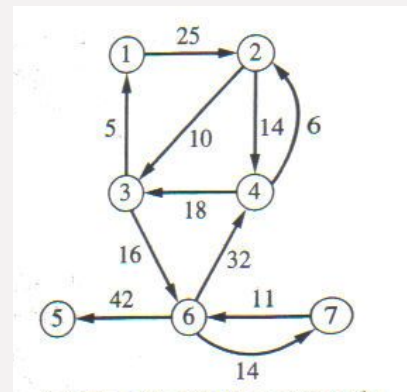
Cuando el grafo es no dirigido para cada arista entre los vértices v_1 y v_2 se deben de agregar dos nodos uno en la lista de el v_1 hasta v_2 y uno en la lista de v_2 hacia v_1 .

Para un grafo dirigido $G(V,A)$ de orden $|V|$ y tamaño $|A|$, se requieren $|V|$ entradas en el directorio y $|A|$ nodos repartidos entre las listas enlazadas.

Lista de Adyacencia para Grafos Ponderados

Cuando el grafo es ponderado, la lista de adyacencia todavía puede ser usada para representar el grafo, haciendo una modificación.

La lista de adyacencia **debe almacenar tanto el vértice adyacente como el peso de cada arista.**

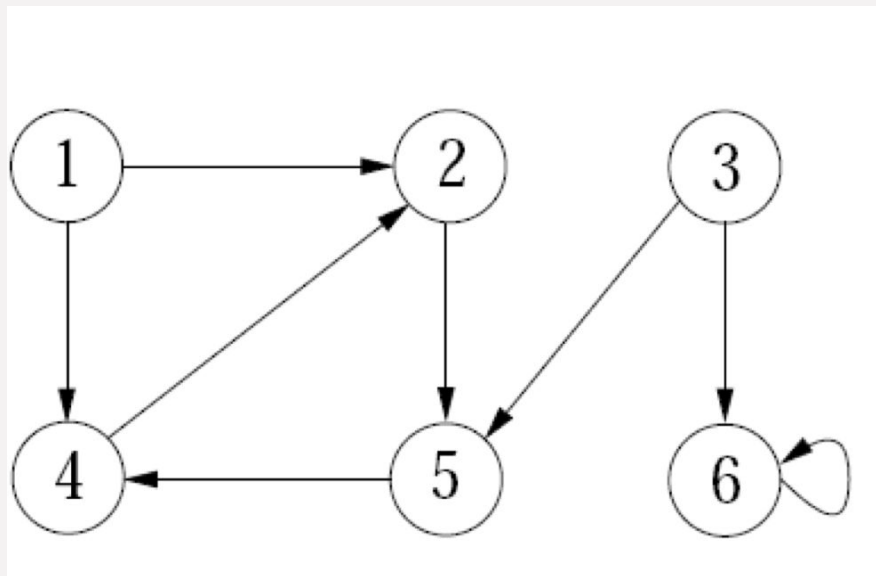
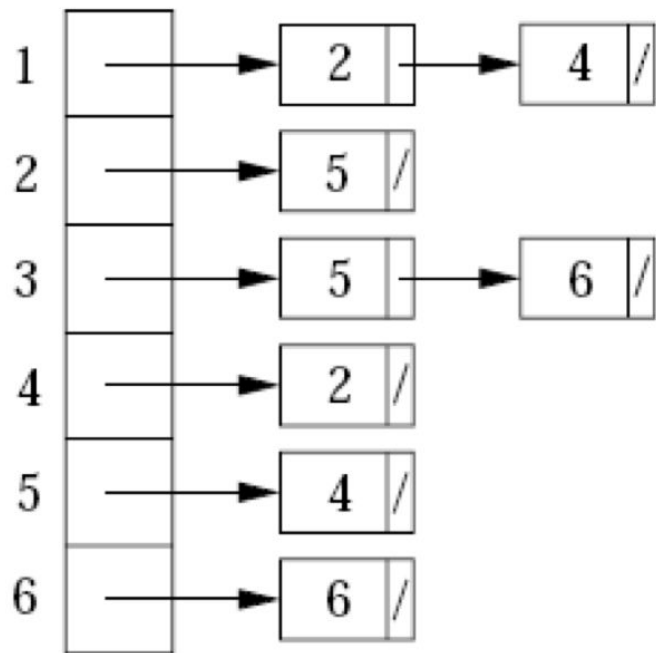


Comparativa Entre Lista y Matriz de Adyacencia

Característica	Matriz de Adyacencia	Lista de Adyacencia
<i>Uso de memoria</i>	$O(V ^2)$	$O(V + A)$
<i>Iteracion</i>	Indirecta, por indices	Directa, por punteros
<i>Acceso Aleatorio</i>	constante	lineal

Ejercicio

Construya el grafo representado por la siguiente lista de adyacencia

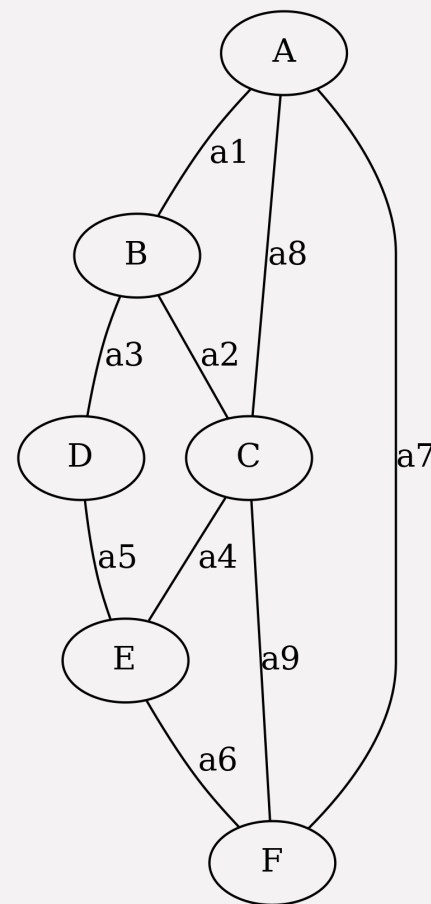


Caminos en un Grafo

Cuando analizamos un grafo nos interesa ver como sus vértices se relacionan todos entre sí. La relación entre dos vértices está dada por la arista que los une.

Si un vértice no es directamente adyacente a otro una manera de relacionar ambos es mediante el concepto de camino (también conocido como caminata o recorrido).

Un camino en un grafo es una secuencia de n aristas (a_1, a_2, \dots, a_n) que junto a una secuencia de $n+1$ vértices (v_1, v_2, \dots, v_{n+1}) tal que la arista a_i conecta a v_i con v_{i+1} .



Caminos en un Grafo

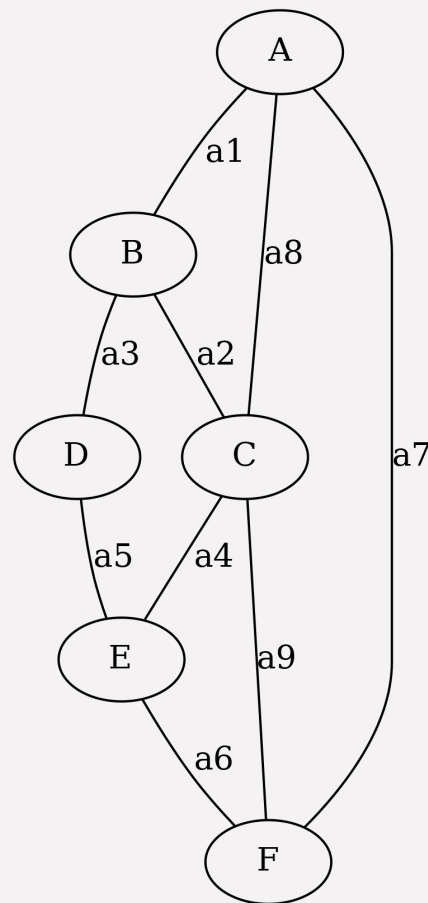
Debido a esto también se puede expresar un camino P como una serie alternante $P=(v_1,a_1,v_2,a_2,...,v_n,a_n,v_{n+1})$.

Si el grafo es un grafo simple (no hay aristas en paralelo) **el camino se puede denotar usando únicamente los vértices** recorridos ya que para cada par (ordenado o no) de vertices solo existe una arista.

Entonces

$P=(v_1,a_1,v_2,a_2,...,v_n,a_n,v_{n+1})$ es equivalente a

$P=(v_1,v_2,...,v_n) \Leftrightarrow G$ es simple



Tipos de Caminos

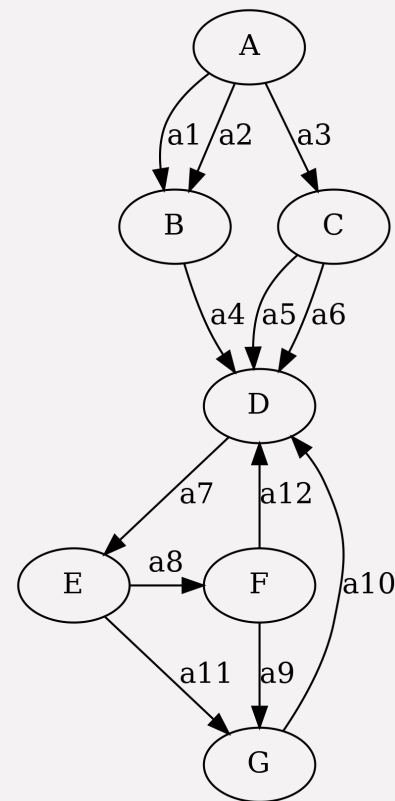
Algunos de los tipos de caminos que se pueden crear dentro de un grafo son:

- A) **Camino Abierto:** camino el cual su último nodo no es el nodo inicial
- B) **Simple:** camino abierto que no repite vértices ni aristas
- C) **Camino Cerrado:** camino el cual su último vértice es el vértice inicial
- D) **Ciclo:** camino cerrado que no repite vértices ni aristas

Ej.

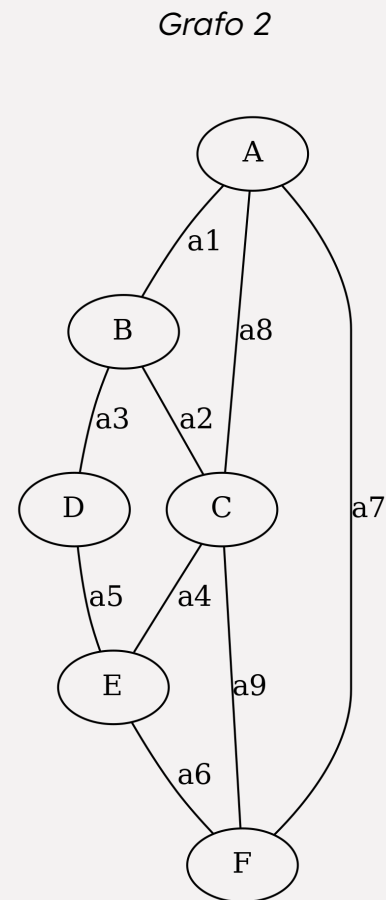
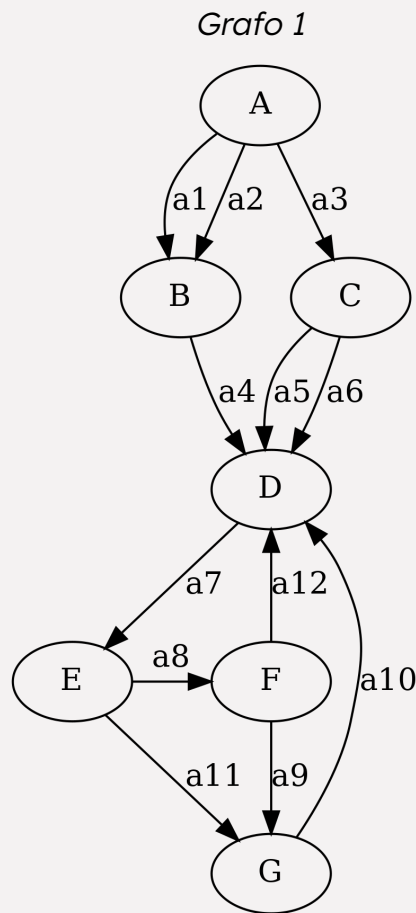
Simple: $W=(A,a1,B,a4,D)$

Ciclo: $W=(D,a7,E,a11,G,a10,D)$



Ejercicios

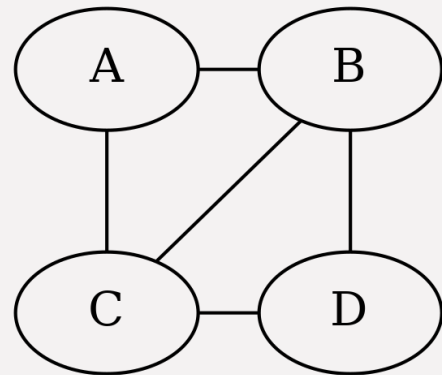
- 1) Escriba un camino que vaya del vértice A al vértice G para el Grafo no.1
- 2) Escriba un camino cerrado para el Grafo no. 2
- 3) Cual de los siguientes dos caminos es válido?
 - Para el Grafo 1: $W=(A, B, D, E)$
 - Para el Grafo 2: $W=(A, B, D, E)$



Longitud de Camino en un Grafo

La longitud de camino de $P=(v_1,a_1,v_2,a_2,...,v_n,a_n,v_{n+1})$ **se define como el número de aristas recorridas**, en este caso n . La longitud de un camino en particular P **se escribe como $|P|$** .

Un camino de longitud cero caracteriza por no tener ningún arista entonces **si $|P|=0$ entonces $P=(v)$** , el camino tiene solo un vértice. A este camino se le conoce como el camino trivial.

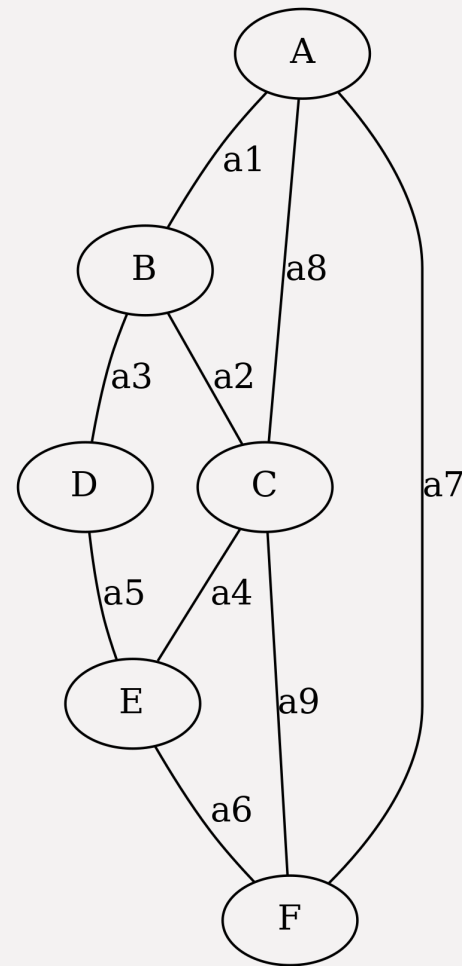


Ej. Si $P=(A,B,D)$
entonces $|P|=3$

Ejercicio

Encuentre la longitud de los siguientes caminos:

- 1) $P=(A,C,F)$
- 2) $P=(B,C,E,F,A,B)$
- 3) $P=(B,A)$

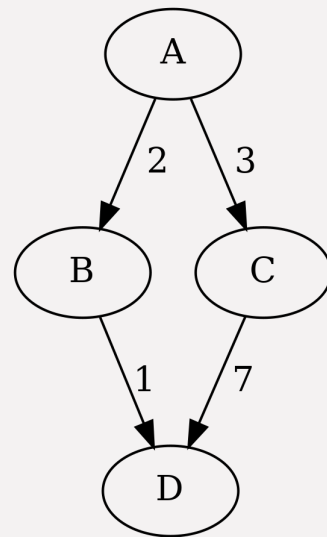


Longitud de Camino en Grafos Ponderados

Para un grafo ponderado **es más representativo usar el peso de las aristas para calcular la longitud** de un camino en vez de la la cantidad de aristas.

Si recordamos un grafo ponderado asigna un peso a cada arista mediante una función w . Podemos usar la función w para calcular la longitud de un camino tal que **la longitud del camino es igual a la suma de los pesos de las aristas recorridas**.

$|P| = w(P) = \sum w(a)$ para cada a en P

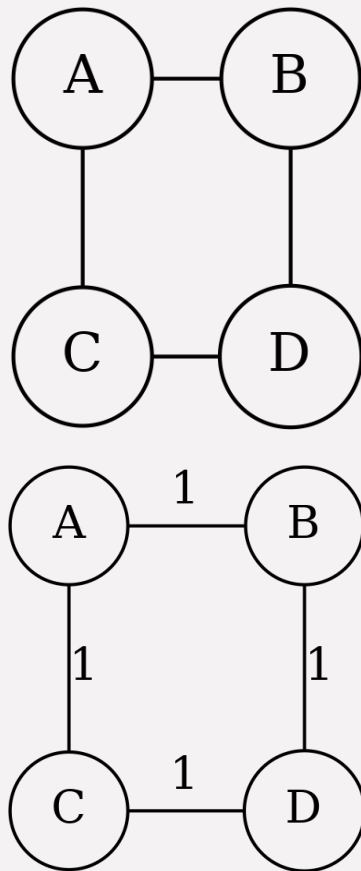


Ej. Si $P = (A, C, D)$ entonces
 $w(P) = w((A, C)) + w((C, D)) = 3 + 7 = 10$

Generalidad de Grafos Ponderados Sobre no Ponderados

Todo grafo no ponderado puede ser representado mediante un grafo ponderado tal que todos los pesos de las aristas del grafo ponderado son igual a uno.

Esto es posible ya que para un camino P en un grafo no ponderado, $|P|$ representa la cantidad de aristas en P mientras que para uno ponderado $|P|=w(P)=\sum w(a)$ y si $w(a)=1$ para toda a entonces $\sum w(a)$ es igual a la cantidad de aristas.

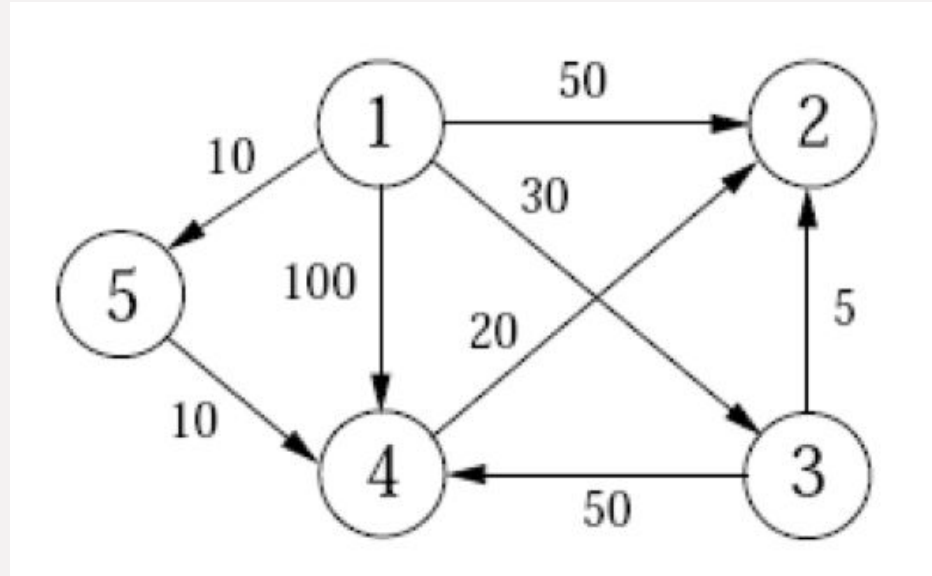


Ejercicio

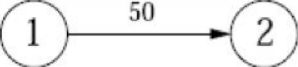

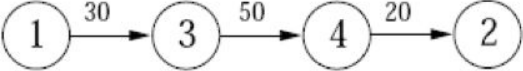

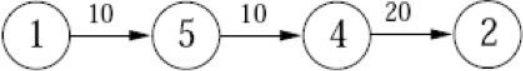
Determine la longitud de los siguientes caminos

- 1.- $P = (1,2)$
- 2.- $P = (1,3,2)$
- 3.- $P = (1,3,4,2)$
- 4.- $P = (1,4,2)$
- 5.- $P = (1,5,4,2)$

Extra: $P = (1,5,3)$



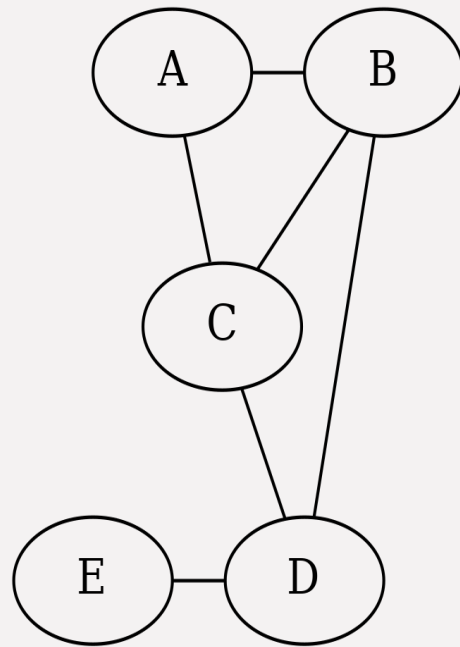
Ejercicio

Camino	Longitud (peso o coste)
	50
	35
	100
	120
	40

Obtención de Caminos desde un punto de vista Algorítmico

Ya hemos realizado ejercicios de obtención de caminos previamente. **Para obtener caminos hasta el momento, hemos usado nuestra intuición y la representación gráfica de los grafos** con los que hemos trabajado.

Tristemente, **los algoritmos no permite trabajar con ninguna de estas dos cosas**. Entonces, en qué tipo de pensamiento nos podemos basar para generar las instrucciones necesarias para encontrar caminos entre dos vértices sin acceso a estas dos cosas?

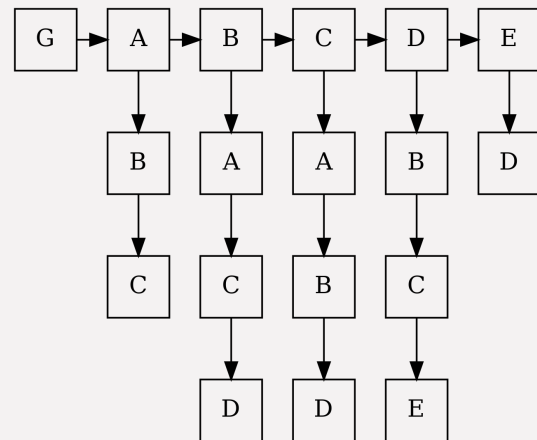


Pensando como la Computadora

A la computadora hay que hablaré en términos de, numeros, vectores, matrices y listas. Por suerte, ya vimos que los grafos tienen otras manera de representarse a parte de la forma gráfica que se adaptan mejor a estos conceptos.

Las representaciones de lista y matriz de adyacencia son más adecuadas para entender cómo generar un algoritmo capaz de encontrar un camino entre dos vértices en un grafo.

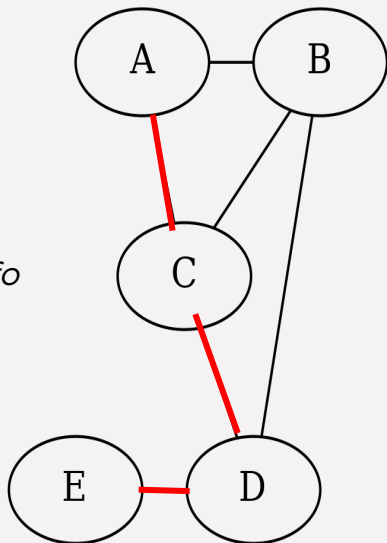
Ahora bien, que limitaciones nos imponen estas dos representaciones?



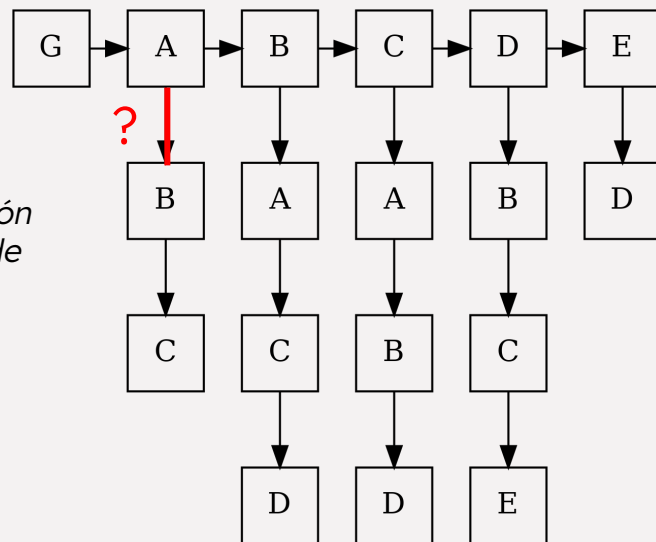
Diferencia Principal

En las representaciones de adyacencia no existe ningún concepto de dirección en comparación con la representación gráfica. Es decir, dado un vértice de inicio no sabemos donde o como se encuentra ubicado el vértice fin respecto al inicio.

*Representación
Gráfica del Grafo*

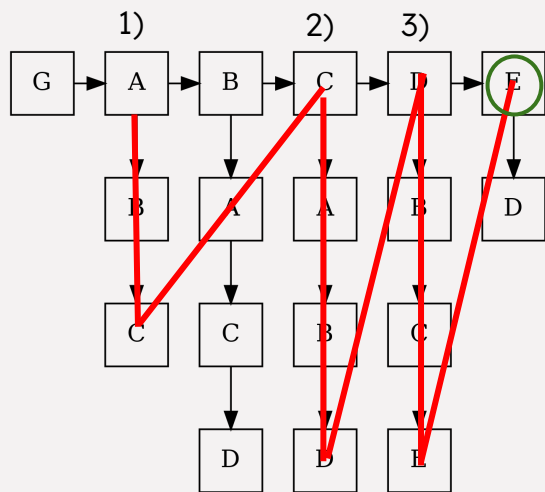


*Representación
como Lista de
Adyacencia*

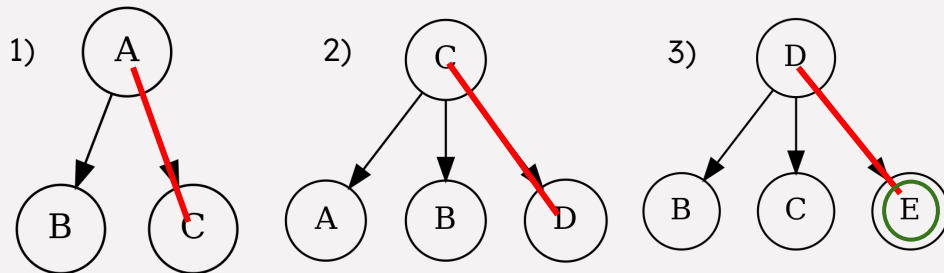


Recorriendo un Camino sin Representación Gráfica

Si intentamos recorrer el grafo siguiente por ejemplo del vértice A al vértice E entonces al iniciar sería como si nos posicionamos sobre el vértice A **solo sabiendo de la existencia de sus vecinos inmediatos**, teniendo que encontrar E de algún modo.

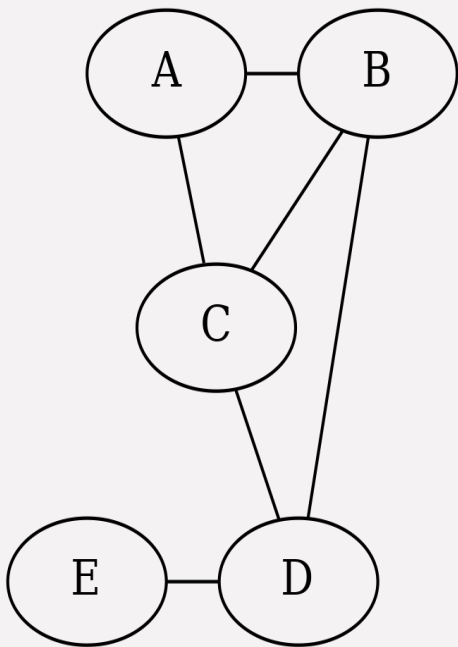


Objetivo: E



Que camino elegimos?

Debido a que la estructura de un grafo es no lineal, **pueden existir múltiples caminos válidos** que podríamos seguir para conectar 2 nodos. Por ejemplo de A a E en este grafo existen $P1=(A,C,D,E)$, $P2=(A,B,D,E)$, $P3=(A,B,C,D,E)$



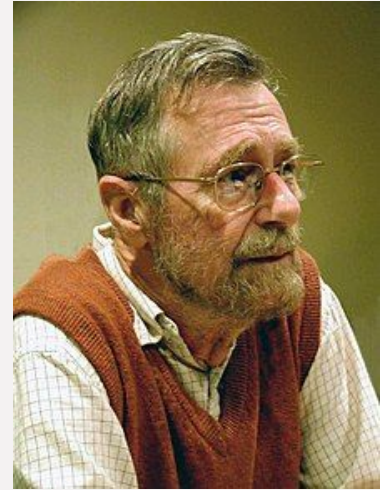
Los tres son caminos válidos. Entonces si queremos diseñar un algoritmo para obtener un camino entre nodos **tenemos que decir qué tipo de camino es el que nos interesa**

Comunmente, para las aplicaciones reales de grafos, **se intenta minimizar la longitud del camino** (la longitud del camino puede representar más cosas que solo distancias físicas, ej. costos). Por lo tanto un algoritmo de obtención de caminos generalmente se basa en encontrar el camino más corto entre dos vértices

El Problema del Camino más Corto

El problema del camino más corto **es uno de los problemas más importantes en teoría de grafos** y por lo tanto uno de los más tratados desde la formalización del área de estudio en el siglo XX.

Una de las primeras soluciones ampliamente aceptadas para este problema y definitivamente la más influyente fue la de un algoritmo desarrollado por el científico en informática holandés Edsger Dijkstra en 1956. Este algoritmo terminaría adoptando el nombre de su creador siendo conocido como el **Algoritmo de Dijkstra** y es la base fundamental en la que muchos otros algoritmos con el mismo propósito han sido modelados.



Algoritmo de Dijkstra para la obtención caminos mínimos

El algoritmo de Dijkstra es un algoritmo voraz de búsqueda **capaz de encontrar las distancias (y caminos) mínimas de un vértice a todos los demás vértices en un grafo.**

Opera bajo el principio básico de elegir la opción más óptima en un momento dado, en este caso eligiendo el camino más corto descubierto hasta el momento.

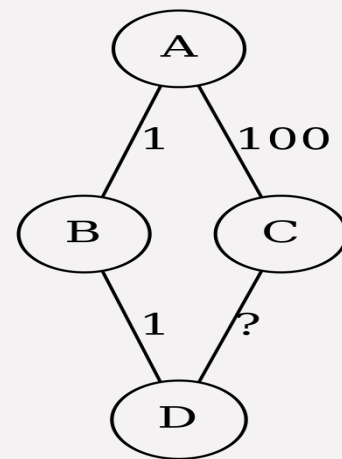


En qué Tipo de Grafos Puedo Aplicar el Algoritmo de Dijkstra?

El algoritmo de Dijkstra es **aplicable en cualquier grafo ponderado** (o no ponderado si se toma el peso de una arista como igual a uno) mientras que **el peso de cada arista sea igual o mayor a cero**.

Esto se debe a que el algoritmo asume que el camino más corto encontrado hasta el momento para llegar al nodo más cercano es realmente el más corto.

El algoritmo de **Dijkstra no discrimina entre grafos no dirigidos y digrafos**.



No se puede asegurar que $P=(A,B,D)$ sea el camino más corto si la arista (C,D) puede tomar valores negativos.

Pasos del Algoritmo de Dijkstra

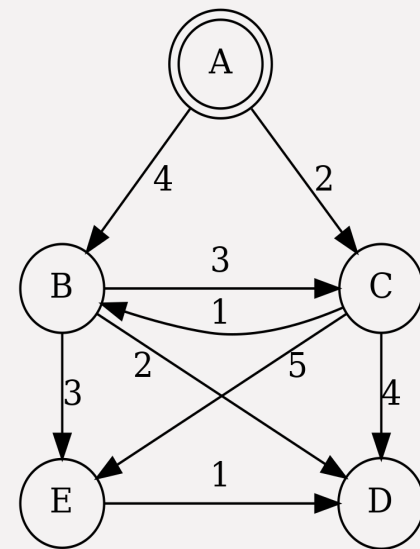
Qué información se necesita guardar?

- Conjunto de vértices no visitados, N
- Conjunto de vértices visitados, S
- La distancia mínima del vértice inicial a cada vértice, $D(v)$
- El vértice que se utilizó para llegar a cada vértice, $P(v)$

Como la guardo?

Para aplicar el algoritmo de Dijkstra a mano todos estos datos pueden guardarse en una tabla con un encabezado como el siguiente:

S	N	$D(v_i)$...	$D(v_n)$	$P(v_i)$...	$P(v_n)$
---	---	----------	-----	----------	----------	-----	----------



Pasos del Algoritmo de Dijkstra

1) Inicialización

Sea $\mathbf{G}(V,A)$ un grafo y \mathbf{vi} el vértice inicial de los caminos hacia los demás vértices:

Se crean dos conjuntos de vértices subconjuntos de V .

- $S=\{vi\}$: conjunto de vértices visitados
- $N=V-\{vi\}$: conjunto de vértices no visitados

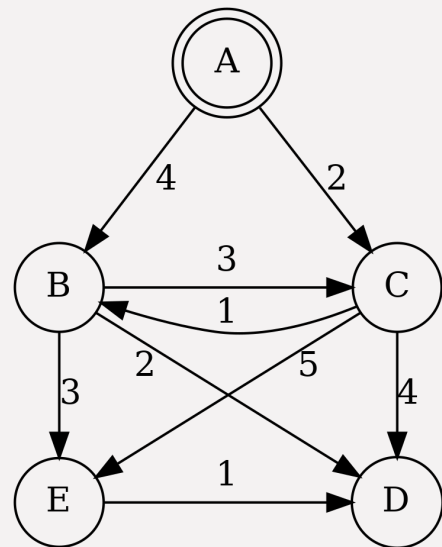
Se inicializan las distancia desde el vértice inicial \mathbf{vi} hasta cualquier otro vértice \mathbf{v} como $\mathbf{D(v)}$ como

$D(v) = 0$ si $v=vi$

$D(v) = \infty$ si $v \neq vi$

Se inicializan los vértices padre de cada vértice \mathbf{v} , $\mathbf{P(v)} = \text{Null}$ para todos los vértices del grafo

Elegimos vi como vértice de referencia vr



$S = \{A\}$

$N = \{B, C, D, E\}$

$D(A) = 0$

$D(B)=D(C)=\dots=\infty$

$P(A)=P(B)=\dots=\text{Null}$

Pasos del Algoritmo de Dijkstra

2) Parte Iterativa

mientras $S \neq \{ \}$ hacemos lo siguiente:

a) Análisis de Aristas de Salida

Para cada artista **a** que va de **vr** a **vf**:

si **vf** en N:

$$dp = D(vr) + w(a)$$

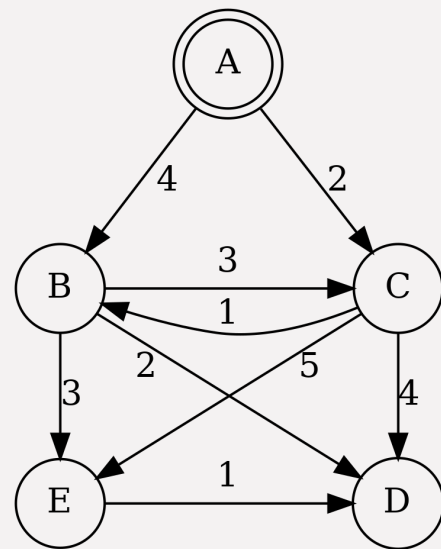
si $dp < D(vf)$:

$$D(vf) = dp$$

$$P(vf) = vr$$

b) Actualizar vr

$vr = v$ en N tal que $D(v)$ sea mínimo
pasar **vr** de N a S



dp: distancia
propuesta

Pasos del Algoritmo de Dijkstra

El algoritmo como lo explica el libro

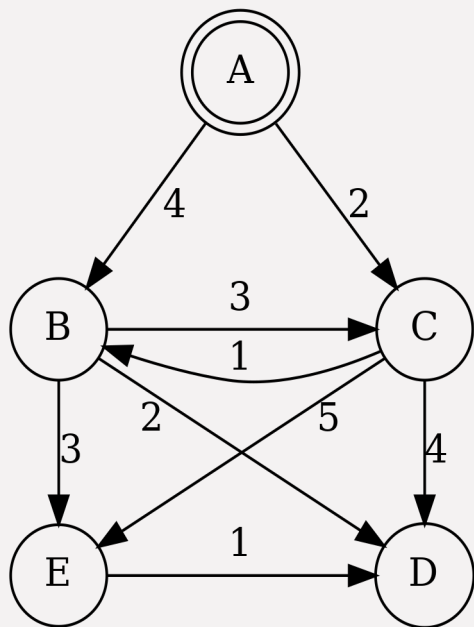
Sean S , D , M

- **S:** Conjunto de vértices visitados; $S=\{v_i\}$
- **D:** Un arreglo de $|V|$ elementos que almacene las distancias más cortas entre un vértice v_j y v_i
- **$M[i,j]$:** La matriz de pesos de un grafo

1. Agregar el vertice v_i a S
2. Repetir con i desde 2 hasta $|V|$
 - Elegir un vertice v en $(V-S)$ tal que $D[v]$ sea el minimo valor posible;
Agregar v a S ;
 - 2.1 Repetir para cada vertice w en $(V-S)$
Hacer $D[w] = \min(D[w], D[v]+M[v,w])$
 - 2.2 Fin del ciclo paso 2.1
3. Fin del ciclo paso 2

Pasos del Algoritmo de Dijkstra

Es necesaria la matriz de costos?

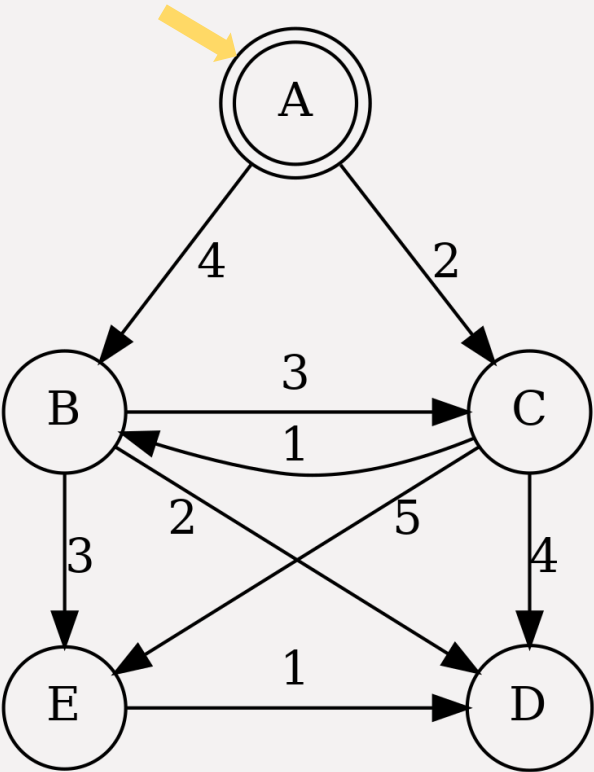


Inicio

Fin

	A	B	C	D	E
A	0	4	2	∞	∞
B	∞	0	3	2	3
C	∞	1	0	4	5
D	∞	∞	∞	0	∞
E	∞	∞	∞	1	0

Ejemplo: Algoritmo de Dijkstra



S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞

v	P(v)
A	Null
B	Null
C	Null
D	Null
E	Null

Inicializamos:

$$S = \{v_i\} = \{A\}$$

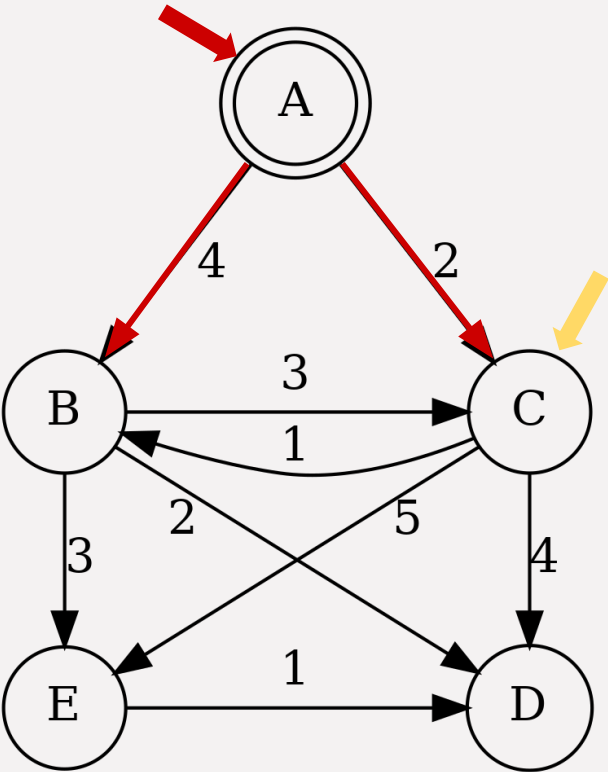
$$N = V - \{v_i\} = \{B, C, D, E\}$$

$$D(v_i) = 0$$

$$D(v_j) = \infty \quad \forall v_j \neq v_i$$

A es el vértice de referencia

Ejemplo: Algoritmo de Dijkstra



S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A}	{B, C, D, E}	0	∞	∞	∞	∞

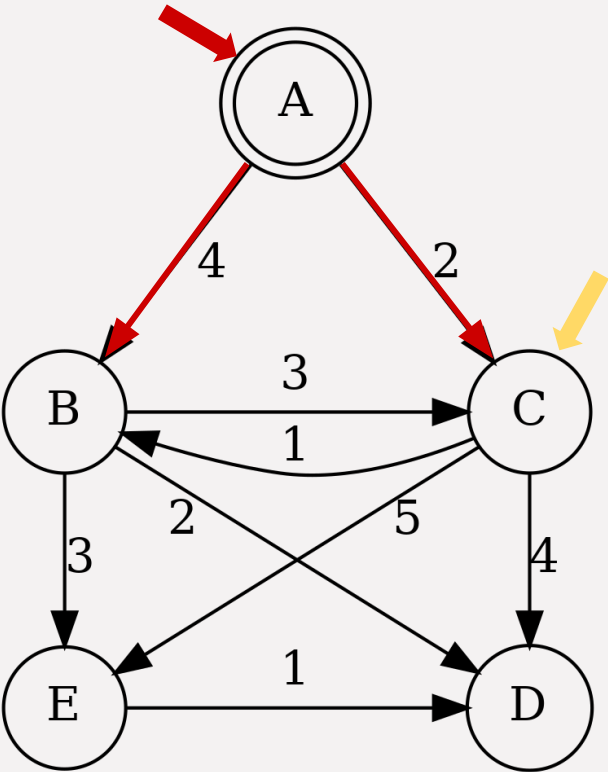
v	P(v)
A	Null
B	Null
C	Null
D	Null
E	Null

Analizamos las aristas de salida

para B: $4+0 < \infty$? Si

para C: $2+0 < \infty$? Si

Ejemplo: Algoritmo de Dijkstra



S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞

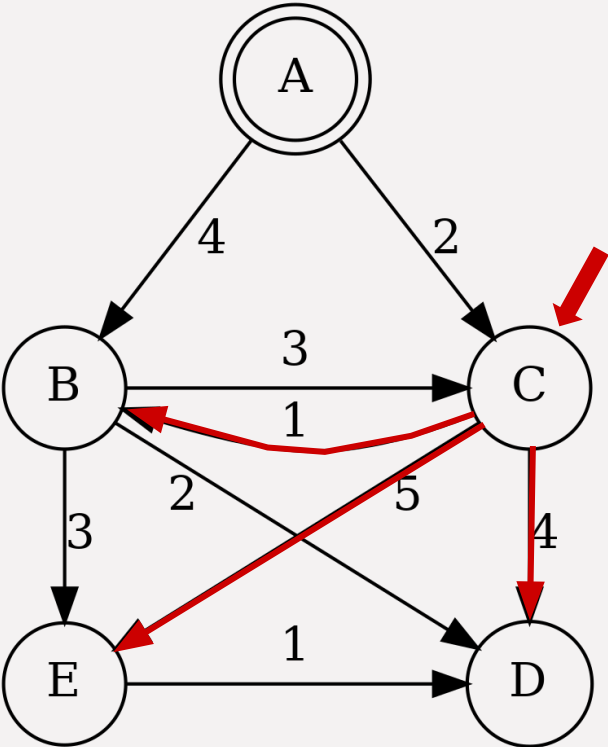
v	P(v)
A	Null
B	A
C	A
D	Null
E	Null

Actualizamos la tabla

$\min(D(v \in N)) == D(C)$
C se mueve de N a S

C es el nuevo vértice de referencia

Ejemplo: Algoritmo de Dijkstra



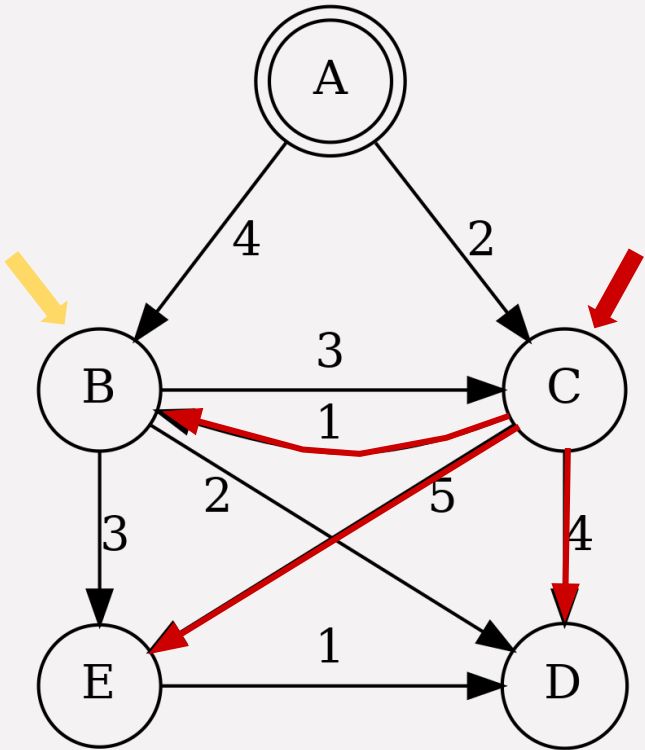
S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞

v	P(v)
A	Null
B	A
C	A
D	Null
E	Null

Analizamos las aristas de salida

para B: $1+2 < 4$? Si
para D: $4+2 < \infty$? Si
para E: $5+2 < \infty$? Si

Ejemplo: Algoritmo de Dijkstra



S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞
{A, C, B}	{D, E}	0	3	2	6	7

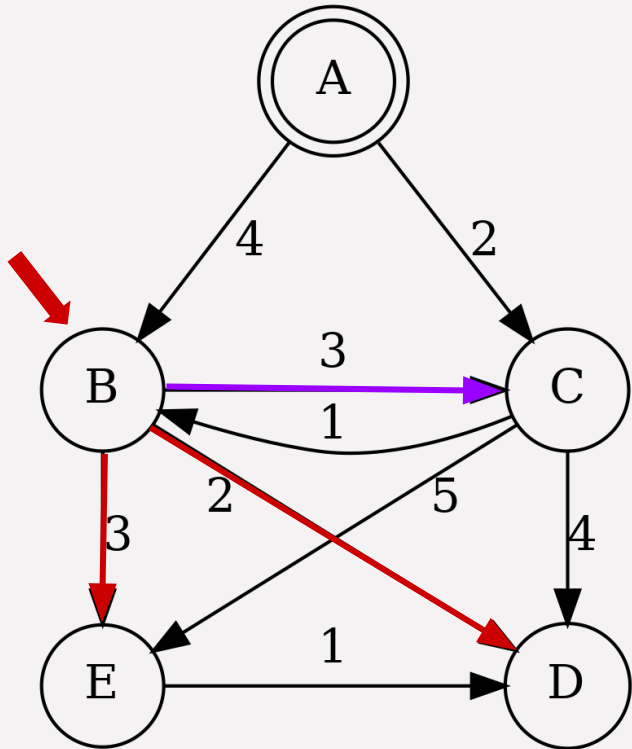
v	P(v)
A	Null
B	C
C	A
D	C
E	C

Actualizamos la tabla

$\min(D(v \in N)) == D(B)$
B se mueve de N a S

B es el nuevo nodo de referencia

Ejemplo: Algoritmo de Dijkstra



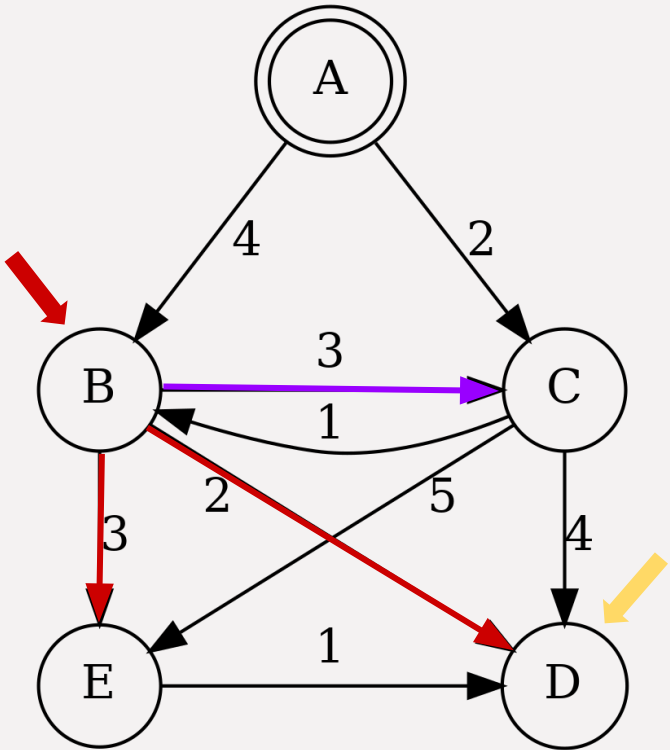
S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞
{A, C, B}	{D, E}	0	3	2	6	7
{A, C, B}	{D, E}	0	3	2	6	7

v	P(v)
A	Null
B	C
C	A
D	C
E	C

Analizamos las aristas de salida

Ignoramos C pues C esta en S
para D: $2 + 3 < 6$? Si
para E: $3 + 3 < 7$? Si

Ejemplo: Algoritmo de Dijkstra



S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞
{A, C, B}	{D, E}	0	3	2	6	7
{A, C, B, D}	{E}	0	3	2	5	6

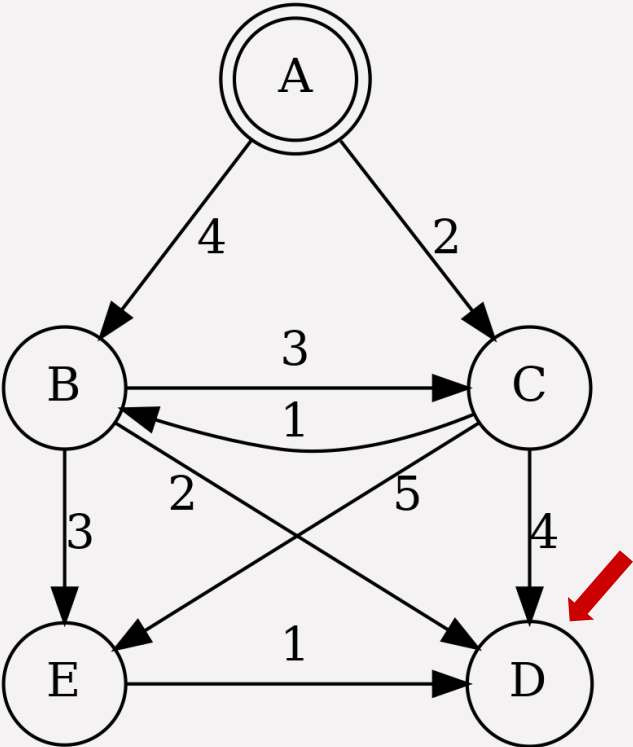
v	P(v)
A	Null
B	C
C	A
D	B
E	B

Actualizamos la tabla

$\min(D(v \in N)) == D(D)$
D se mueve de N a S

D es el nuevo nodo de referencia

Ejemplo: Algoritmo de Dijkstra



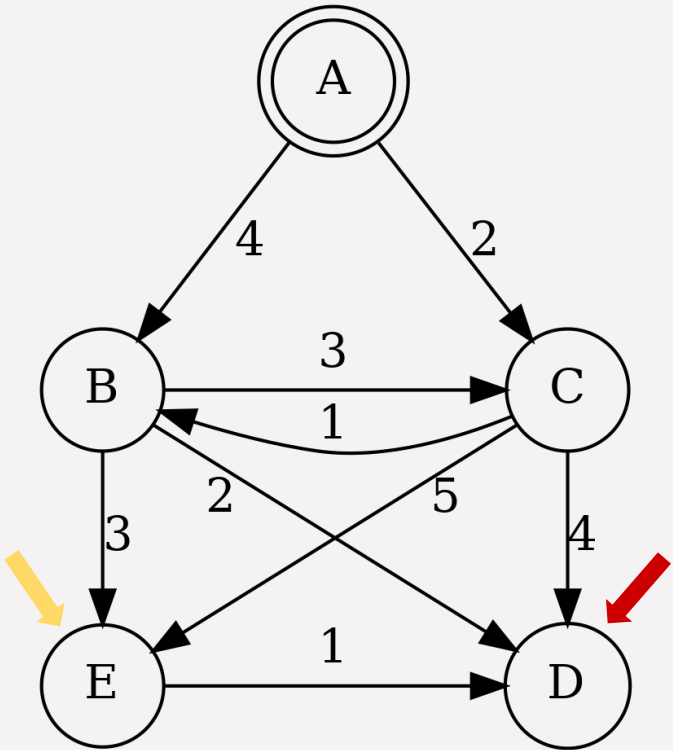
S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞
{A, C, B}	{D, E}	0	3	2	6	7
{A, C, B, D}	{E}	0	3	2	5	6
{A, C, B, D}	{E}	0	3	2	5	6

v	P(v)
A	Null
B	C
C	A
D	B
E	B

Analizamos los aristas de salida

D no tiene aristas de salida

Ejemplo: Algoritmo de Dijkstra



S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞
{A, C, B}	{D, E}	0	3	2	6	7
{A, C, B, D}	{E}	0	3	2	5	6
{A, C, B, D, E}	{}	0	3	2	5	6

v	P(v)
A	Null
B	C
C	A
D	B
E	B

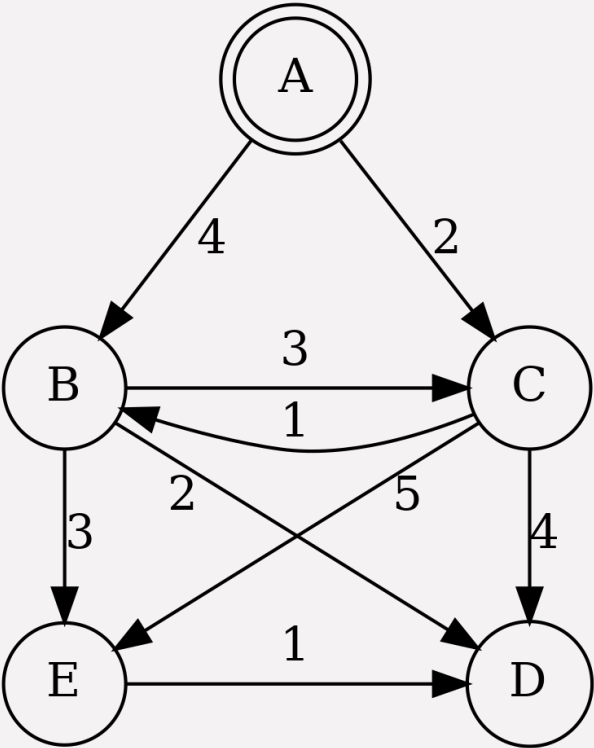
Actualizamos la tabla

$\min(D(v \in N)) == D(E)$

Movemos E de N a S

Como no hay vértices por visitar
entonces finalizamos el algoritmo

Ejemplo: Algoritmo de Dijkstra



S	N	D(A)	D(B)	D(C)	D(D)	D(E)
{A}	{B, C, D, E}	0	∞	∞	∞	∞
{A, C}	{B, D, E}	0	4	2	∞	∞
{A, C, B}	{D, E}	0	3	2	6	7
{A, C, B, D}	{E}	0	3	2	5	6
{A, C, B, D, E}	{}	0	3	2	5	6

v	P(v)
A	Null
B	C
C	A
D	B
E	B

Obtención de caminos en base a los resultados del Algoritmo

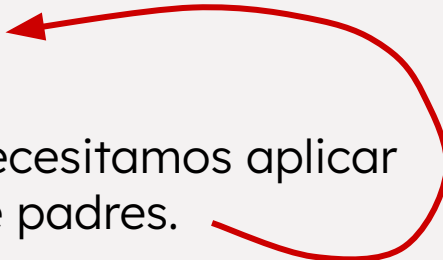
Como observamos, el algoritmo hasta ahora solo nos consigue dos “listas”, una con las distancias de los caminos más cortos.

D(A)	D(B)	D(C)	D(D)	D(E)
0	3	2	5	6

y otra con la relación entre los vértices y sus antecesoros

P(A)	P(B)	P(C)	P(D)	P(E)
NULL	C	A	B	B

Para encontrar los caminos (secuencias de vértices) necesitamos aplicar otro procedimiento sobre la segunda lista de padres.



Pasos del Algoritmo de Dijkstra

3) Construcción de Caminos

P(A)	P(B)	P(C)	P(D)	P(E)
NULL	C	A	B	B

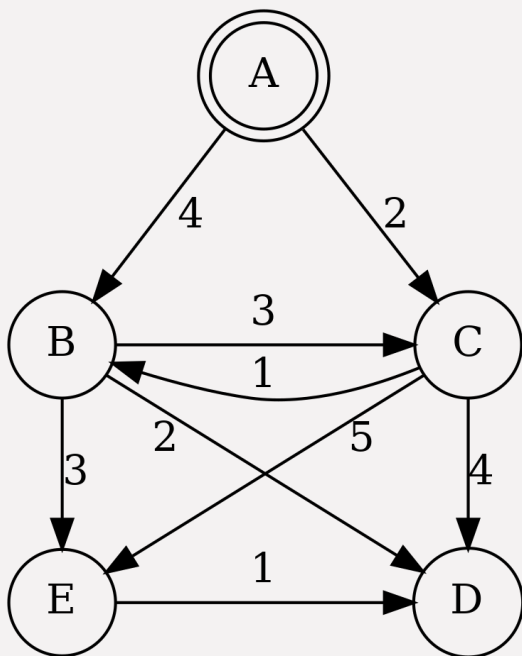
Para encontrar cada camino W con inicio en v_i y fin en v_f se recorre la lista de vértices padre de “atrás hacia adelante”.

Se crea una secuencia con v_f $W = (v_f)$ y se insertan los vértices padre por principio de la secuencia hasta que se encuentre v_i o se el padre de v_f sea NULL.

Si $P(v_f) == \text{NULL}$, entonces no se encontró un camino de a v_f a menos que $v_f == v_i$.

Entonces $W(v_i, v_f) = (v_i, \dots, P(P(v_f)), P(v_f), v_f)$

Ejemplo: Algoritmo de Dijkstra

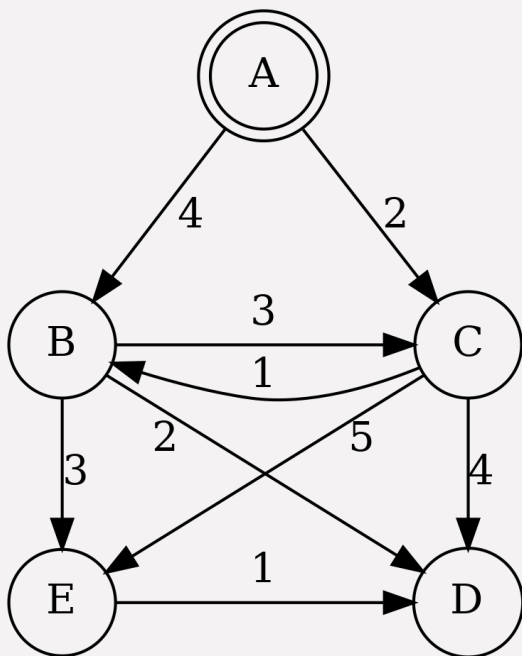


1. $W(A, A) = (... , A)$
 2. $P(A) == \text{NULL}, v_i == v_f?$ Si
 3. $W(A, A) = (A)$
-

1. $W(A, B) = (... , B)$
2. $P(B) == C, C == v_i?$ No
3. $W(A, B) = (... , C, B)$
4. $P(C) == A, A == v_i?$ Si
5. $W(A, B) = (A, C, B)$

P(A)	P(B)	P(C)	P(D)	P(E)
NULL	C	A	B	B

Ejemplo: Algoritmo de Dijkstra



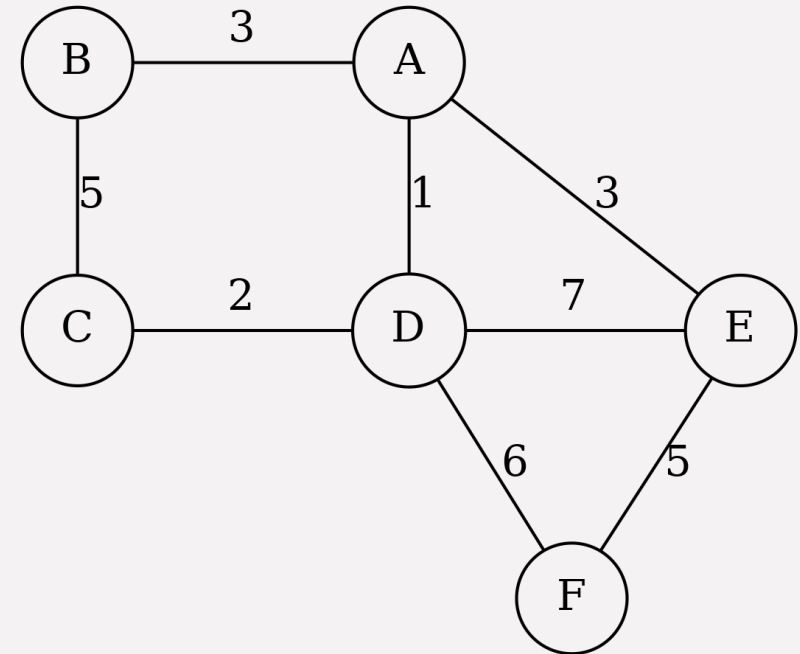
...

1. $W(A, E) = (... , E)$
2. $P(E) == B, B == v_i?$ No
3. $W(A, E) = (... , B, E)$
4. $P(B) == C, C == v_i?$ No
5. $W(A, E) = (... , C, B, E)$
6. $P(C) == A, A == v_i?$ Si
7. $W(A, E) = (A, C, B, E)$

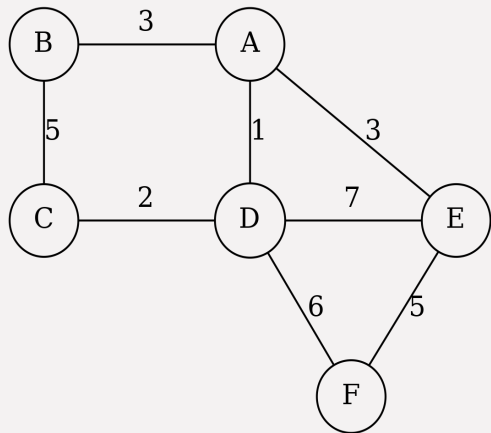
P(A)	P(B)	P(C)	P(D)	P(E)
NULL	C	A	B	B

Ejercicio

- 1) Obtenga las **distancias más cortas entre el vértice B y todos los demás vértices** del siguiente grafo. Utilice el algoritmo de Dijkstra.
- 2) Obtenga los **caminos más cortos entre el vértice B y todos los demás vértices** del siguiente grafo. Utilice el algoritmo de Dijkstra.



Ejercicio



S	N	D(A)	D(B)	D(C)	D(D)	D(E)	D(F)
{B}	{A,C,D,E,F}	∞	0	∞	∞	∞	∞
{B,A}	{C,D,E,F}	3	0	5	∞	∞	∞
{B,A,D}	{C,E,F}	3	0	5	4	6	∞
{B,A,D,C}	{E,F}	3	0	5	4	6	10
{B,A,D,C,E}	{F}	3	0	5	4	6	10
{B,A,D,C,E,F}	{}	3	0	5	4	6	10

v	P(v)
A	B
B	Null
C	B
D	A
E	A
F	D

$W(B,A) = (B,A)$

$W(B,B) = (B)$

$W(B,C) = (B,C)$

$W(B,D) = (B,A,D)$

$W(B,E) = (B,A,E)$

$W(B,F) = (B,A,D,F)$

Aplicaciones

Redes sociales: Las plataformas como Facebook, Instagram y LinkedIn utilizan grafos para modelar las conexiones entre personas. Cada usuario es un vértice, y las relaciones de amistad o seguidores son las aristas.



Redes de telecomunicaciones: Las compañías telefónicas y de internet usan grafos para representar las conexiones entre dispositivos o estaciones base, optimizando la transmisión de datos y las llamadas entre nodos.

Aplicaciones

Sistemas de recomendación: Servicios como Netflix y Spotify utilizan grafos para analizar las relaciones entre usuarios y productos, recomendando contenidos en función de conexiones entre preferencias y patrones de consumo.



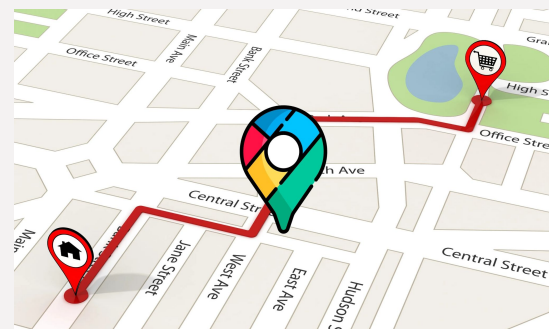
Logística y transporte: Empresas como Amazon o FedEx modelan sus rutas de entrega como grafos, optimizando tiempos y costos mediante algoritmos que buscan las rutas más eficientes.

Aplicaciones

Rutas de tráfico: Los sistemas de navegación por satélite utilizan grafos para calcular las rutas más eficientes entre dos puntos.

Búsqueda en línea: Los motores de búsqueda como Google utilizan grafos para indexar y recuperar información.

Informática: La teoría de grafos se utiliza en el diseño y análisis de algoritmos y en la optimización de bases de datos. Los grafos se utilizan para modelar las relaciones entre los datos y para encontrar la ruta más eficiente para acceder a ellos.

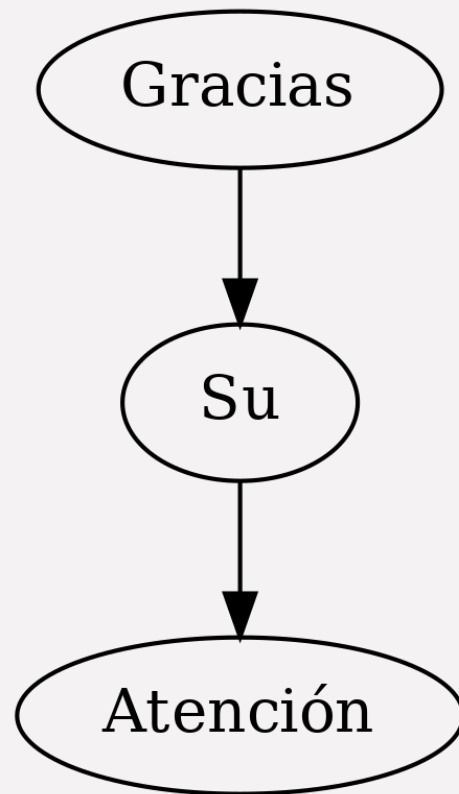


Conclusion

Los grafos son estructuras fundamentales tanto en las matemáticas como en las estructuras de datos, **que nos permiten organizar y modelar relaciones entre objetos de manera estructurada.**

Estos son usados en amplias ramas como las telecomunicaciones, teoría de juegos, inteligencia artificial, logística, etc.

Haciendo uso de ellos, **podemos darle solución a problemas complejos gracias a su capacidad para representar datos interconectados. Siendo una pieza clave en el desarrollo tecnológico y científico.**



Bibliografia

- ❑ Pythoned. (s.f.). *Una lista de adyacencia*. Pythoned. Recuperado el 28 de febrero de 2025 de <https://runestone.academy/ns/books/published/pythoned/Graphs/UnaListaDeAdyacencia.html>
- ❑ Sclar, M. (2015). *Grafos: una no tan breve introducción*. Facultad de Ciencias Exactas y Naturales Universidad de Buenos Aires. Recuperado el 28 de febrero de 2025 de <https://www.oia.unsam.edu.ar/wp-content/uploads/2013/10/Grafos.pdf>
- ❑ Clase grafos. (s.f.). *Clase grafos*. Clase grafos. Recuperado el 28 de febrero de 2025 de <https://aniei.org.mx/paginas/uam/Descargas/Recursos/ClaseGrafos.pdf>
- ❑ Navarrete, W. (s.f.). *Representación de Grafos*. Walter Navarrete Pino. Recuperado el 28 de febrero de 2025 de <https://wnavarrete.wordpress.com/representacion-de-grafos/>
- ❑ Bianco, S. (junio de 2016). *ResearchGate*. ResearchGate. Recuperado el 28 de febrero de 2025 de https://www.researchgate.net/figure/Grafo-dirigido-con-su-representacion-como-lista-de-adyacencia_fig14_309278789
- ❑ Verdejo, A. (n.d.). *Grafo bipartito*. ¡Acepta el reto! Retrieved March 2, 2025, from <https://aceptaelreto.com/pub/problems/v002/79/st/problem.pdf>
- ❑ Grafo bipartito | PPT. (2012, December 7). SlideShare. Retrieved March 2, 2025, from <https://www.slideshare.net/slideshow/grafos-bipartito-15542624/15542624>