Universidad Nacional de San Martín

Licenciatura en Ciencia de Datos

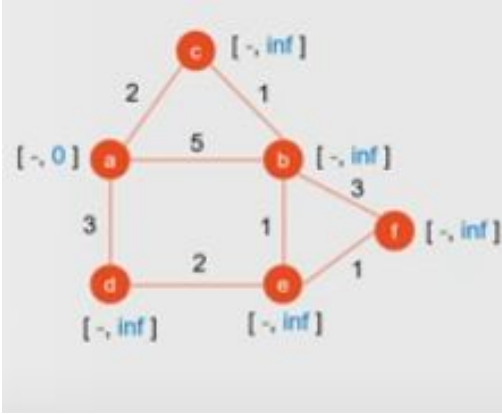
Algoritmos II

Algoritmo de Dijkstra mejorado

-Devuelve el **camino de menor costo** desde un nodo **origen** a **cualquiera** que tenga **conexión** (puede ser grafo dirigido o no dirigido). Si las **aristas NO TIENEN PESO** se asume que tienen **PESO 1**

-**SOPORTA ARISTAS CON PESO NEGATIVO**

-**NO DETECTA CICLOS NEGATIVOS** (diferencia con Bellman Ford) => **PRECONDICIÓN**



-Se incorporan **ETIQUETAS** a los **nodos:** (es el mismo)

-[predecesor, **costo min** desde **nodo origen**]

-Marca **NODO DISPONIBLE PARA SER VISITADO.** Se invierte un poco que nodos puedo visitar. Existe posibilidad de que se pueda activar como

POSIBLE VISITA = PUEDE VOLVER A ENTRAR COMO NODO DISPONIBLE

(diferencia con Dijkstra original)

-Los **nodos DISPONIBLES A VISITAR**(no marcados) se pueden guardar en una **COLA.**

Situación inicial

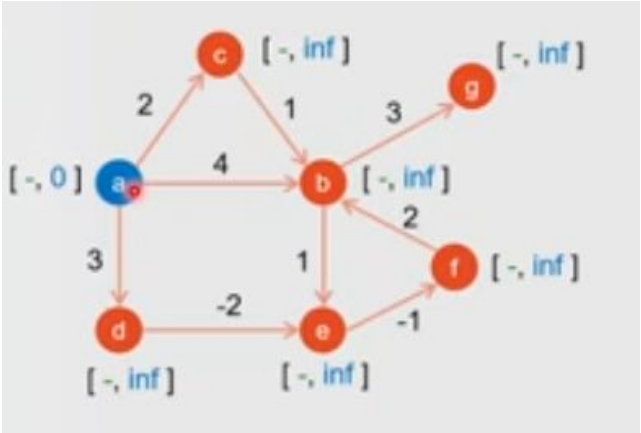
-**Costo min** nodos = **inf**

-**Predecesores = vacíos**

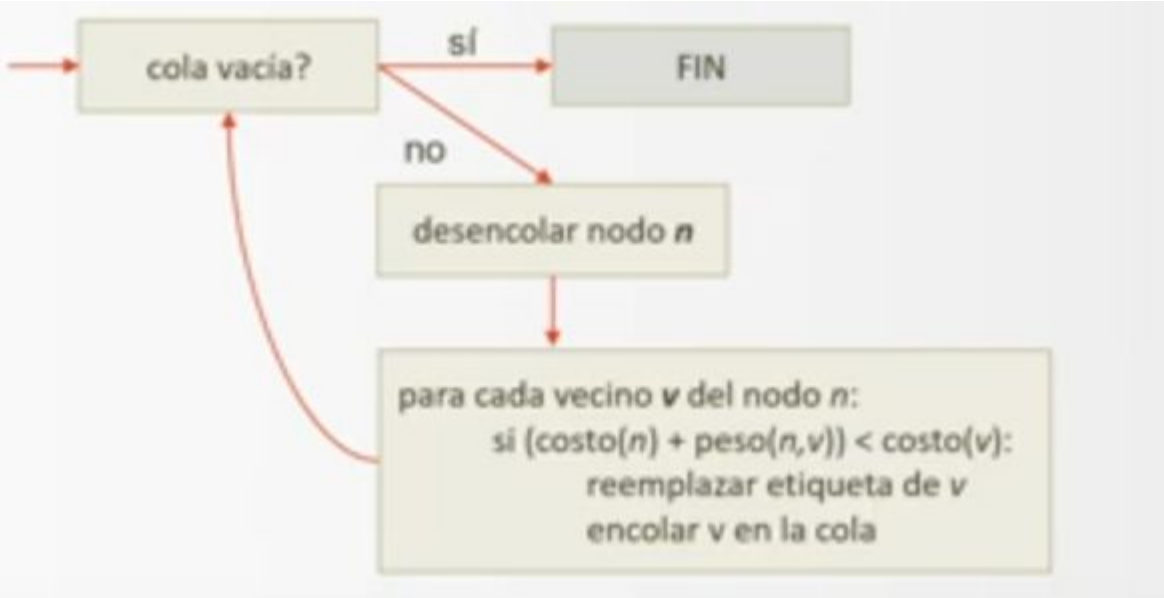
-**Costo nodo inicial = 0**

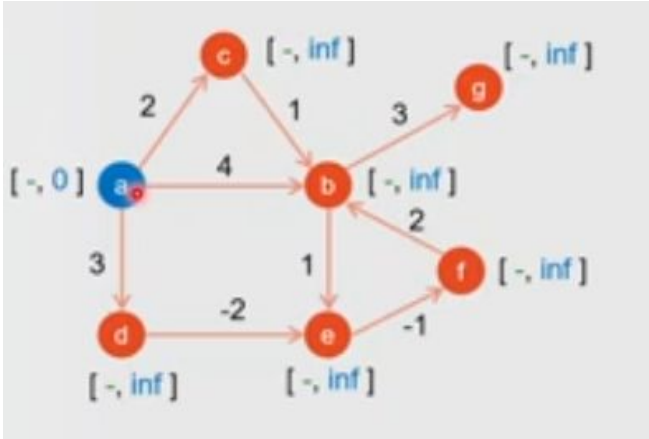
-El **nodo inicial** está **disponible** para **visitar** (es el **único**)

PUEDO USAR UNA ESTRUCTURA COMO UNA COLA (permite i<u>nsertar nodos posibles a visitar</u>) => **mientras la cola tenga algún nodo nuevo para visitar** los vamos **removiendo** haciendo de cuenta que los **estamos <u>marcando como visitado</u>** => llegamos a un punto donde **no haya más nodos y el algoritmo converge.**



Algoritmo



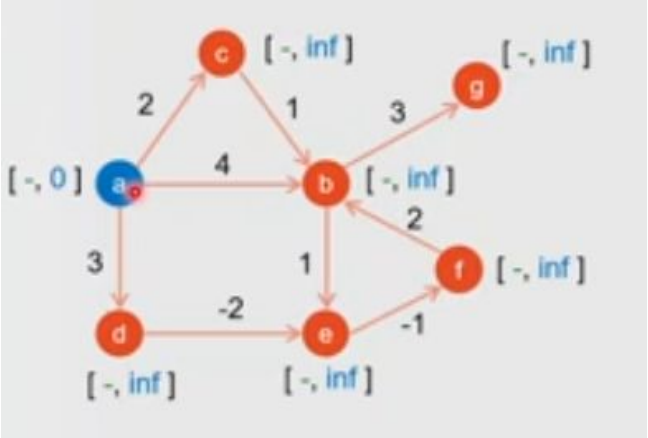


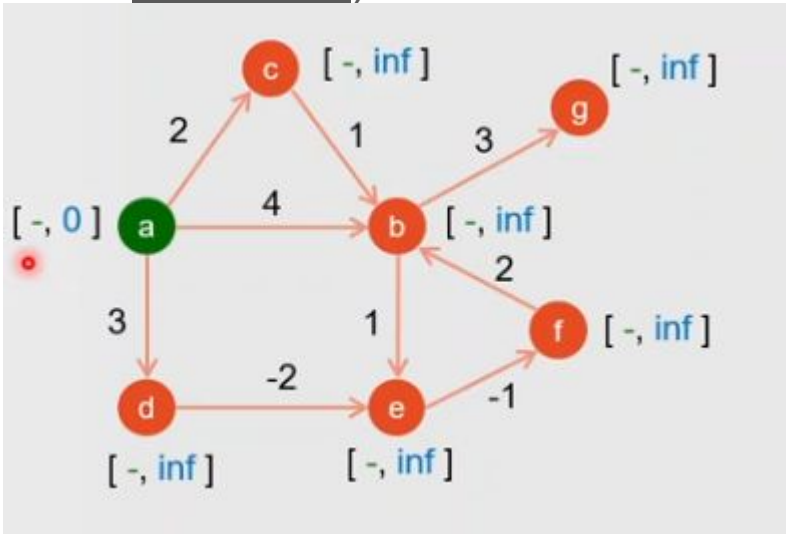
Diferencia con los otros algoritmos: si tenemos que **reemplazar la etiqueta de los vecinos,** los vamos a **agregar** en la **cola** (los habilitamos como **nodos disponibles a visitar)**

Ejemplo

-Arrancamos con **nodo a**(**único** en **azul**, **disponible** para visitar).

-Lo **sacamos** de la **cola** (pasa a **verde** para ser **analizado**)

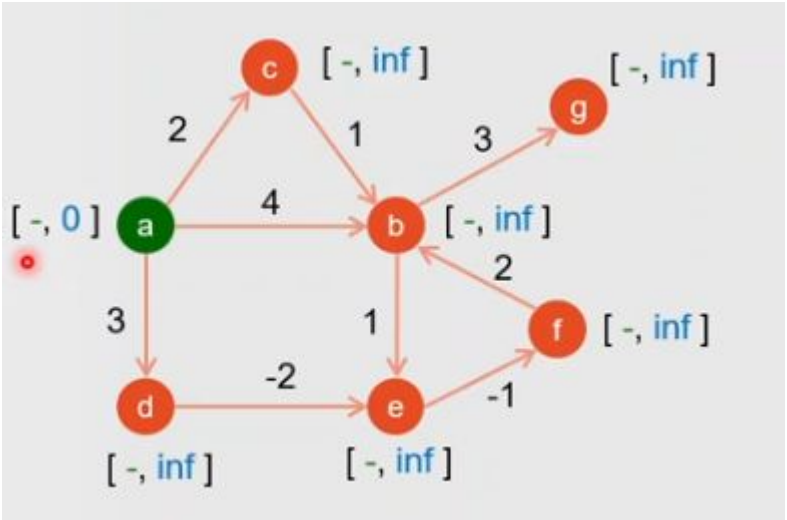


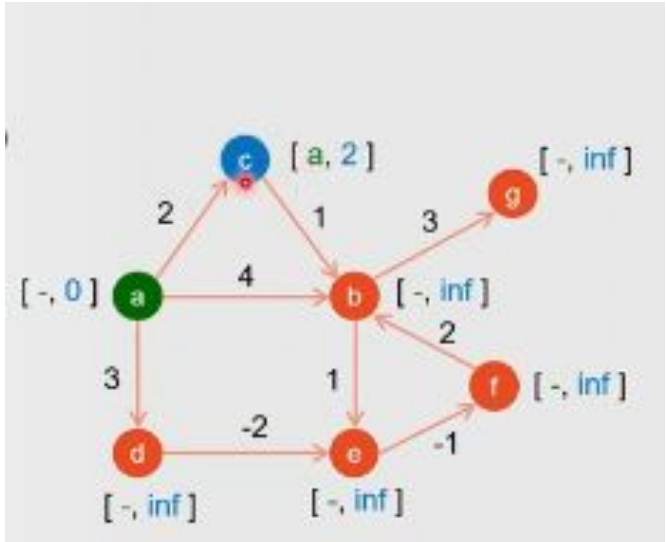


Ejemplo

-Vemos **cada vecino** que tiene **a: c, b y d**

-Vecino **c: 0+2<inf => modificación etiqueta [a,2] (como hasta ahora)** y como la modificamos lo agregamos como **disponible (azul)** en la **cola**

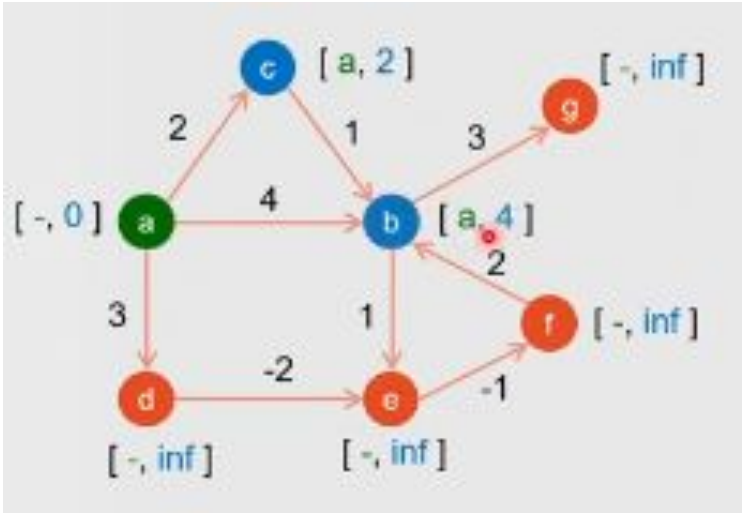




Ejemplo

-Otro nodo vecino de **a** es **b.**

-0+4 < inf => **etiqueta** [a,4] y activación para **visitar posteriormente (azul)**

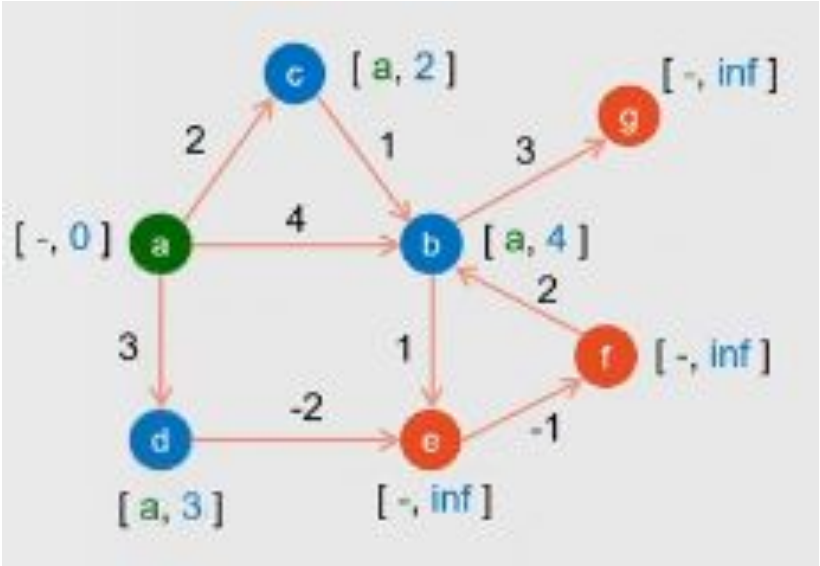


Ejemplo

-Vecino **d: 0+3 < inf => etiqueta [a,3]** y **se marca el nodo** para poder visitarlo de nuevo (**azul**).

Cola hasta ahora

tiene: c, b, d.



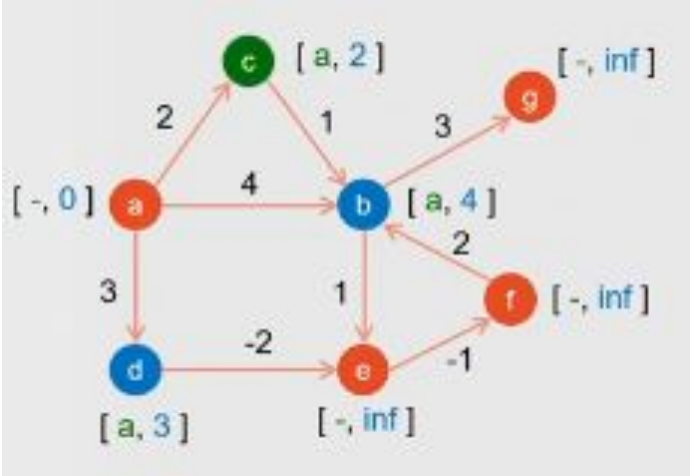
Ejemplo

Terminó el bloque **para** del nodo **a =>** el nodo se **finalizó** (ya queda marcado que **no se va a recorrer más**)

-Volvemos a validar **condición de cola.**

-Tomamos el **primer nodo** que habíamos insertado: **c** Lo ponemos en **verde** porque lo

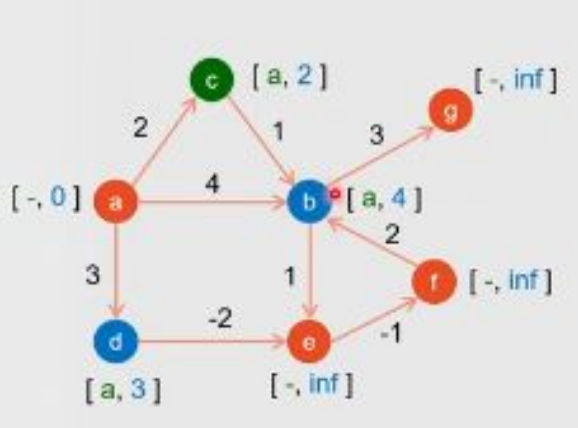
sacamos de la cola y **validamos** lo mismo con sus **vecinos.**

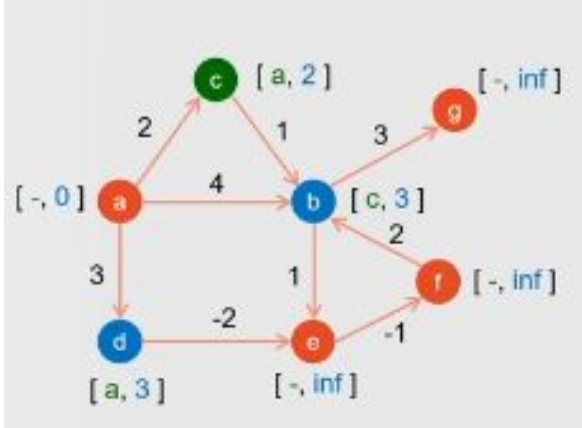


Ejemplo

-Único vecino de **c** es **b:** 2+1 = 3 <4 => **etiqueta = [c,3]**

-Como **b** ya estaba marcado en la cola, **no hace falta encolarlo** (activarlo como **azul**)



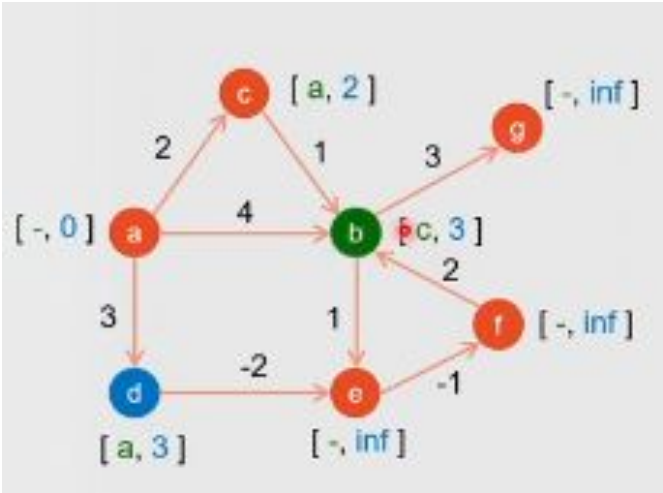


Ejemplo

-Terminó el nodo **c** (lo **<u>desmarcamos</u>, pasa a <u>naranja</u> como el nodo a**)

-Chequeamos la **cola,** y tenemos por visitar el **nodo b.**

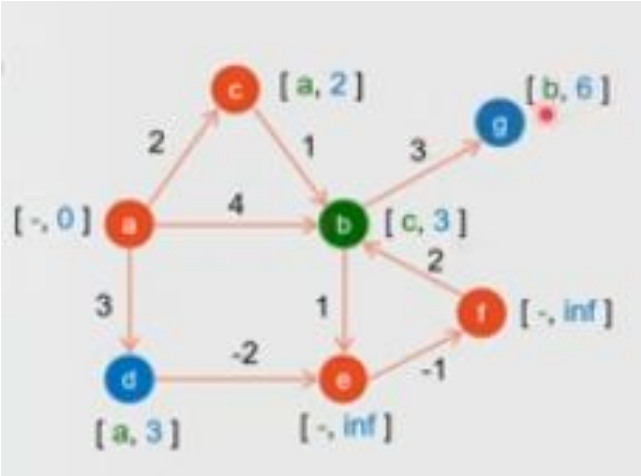
-Tiene 2 vecinos: **g** y **e** (por ser grafo **dirigido**)



Ejemplo

-Nodo **g:** etiqueta: 3 (de b)+3(de arista) =6 < inf => reemplazo **[b,6]**

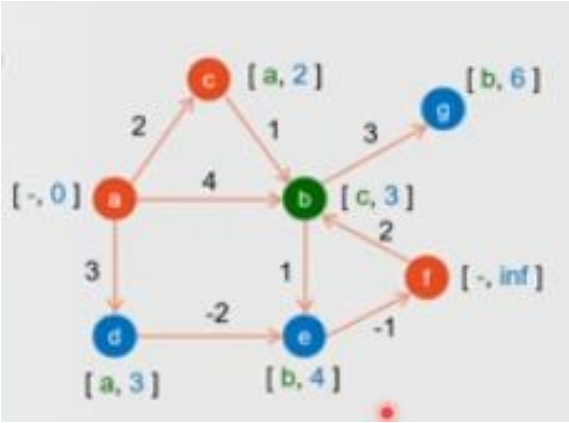
-Se marca **g** para **poder visitar en azul (cola)**



Ejemplo

-**Nodo e:** 3+1=4 < inf => reemplazo **etiqueta [b,4]**

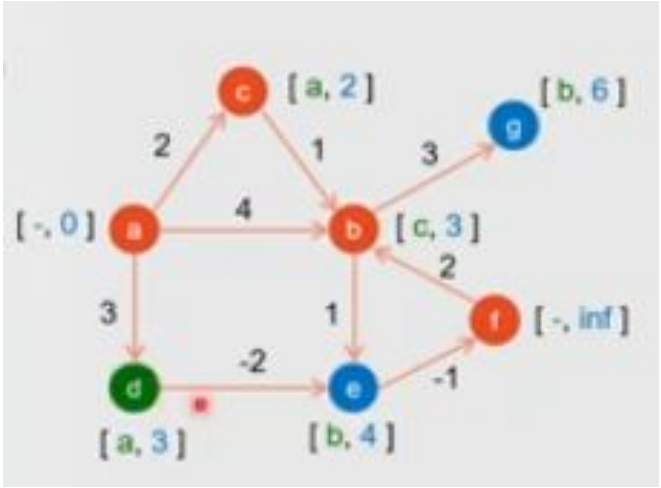
- También se marca **para visitar y agrega a la cola.**



Ejemplo

-Terminó el **nodo b** => se **desmarca (naranja)**

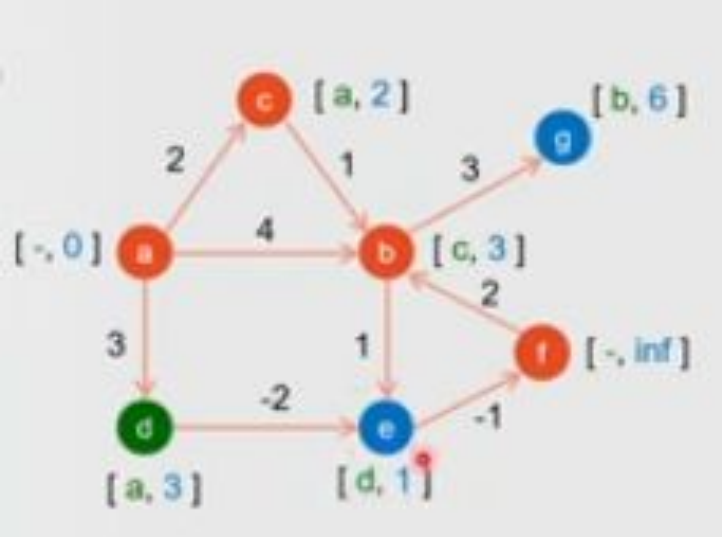
-Tomamos el **nodo d** de la **cola** a marcar como **visita (porque es el que sigue en la cola)**



Ejemplo

-Único vecino: **e.** **Etiqueta:** 3-2 = 1 < 4 => **[d,1]**

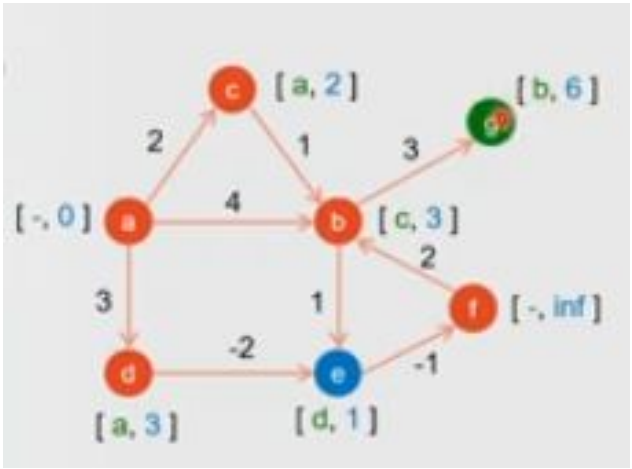
- e ya está **marcado** => no se agrega a **cola**

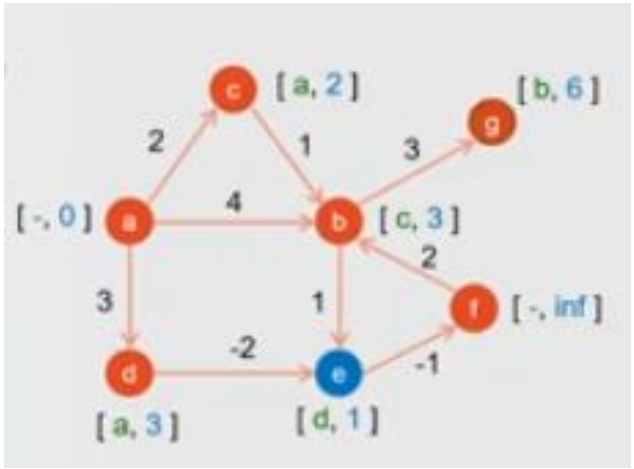


Ejemplo

-Terminó el proceso del **d,** se **desmarca** y pasamos a **g** (nodo que sigue en la cola)

-Se verifica si **tiene vecinos.** No tiene, no entra en **para,** se **desmarca** y seguimos





Ejemplo

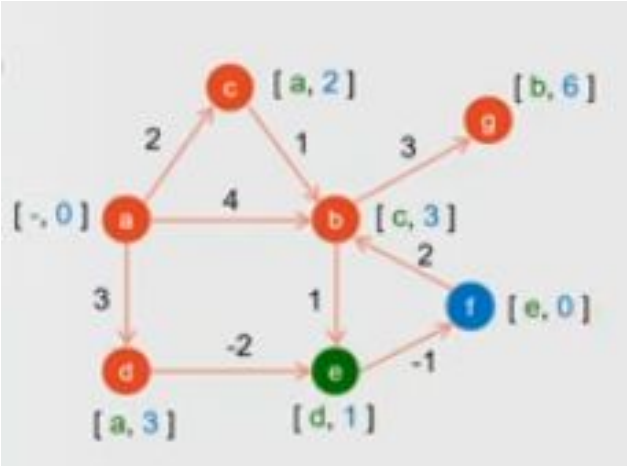
-Nos queda en la cola para **visitar** el **e.**

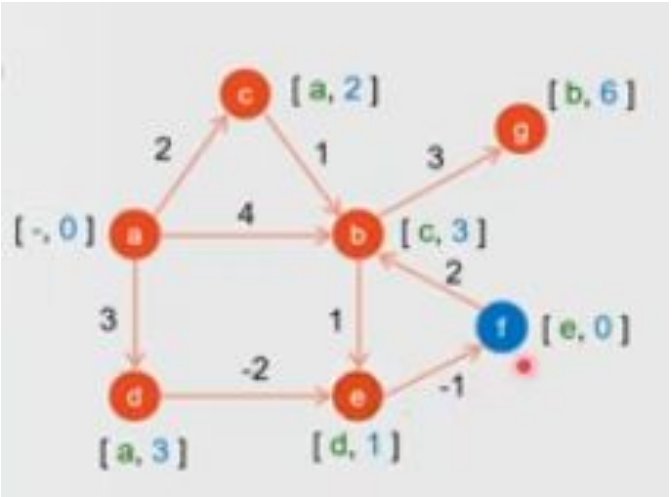
-Único vecino es el **f**:

-**Etiqueta:** 1-1=0 < inf => [e,0]

-Se **encola f**

-Se desmarca **e**

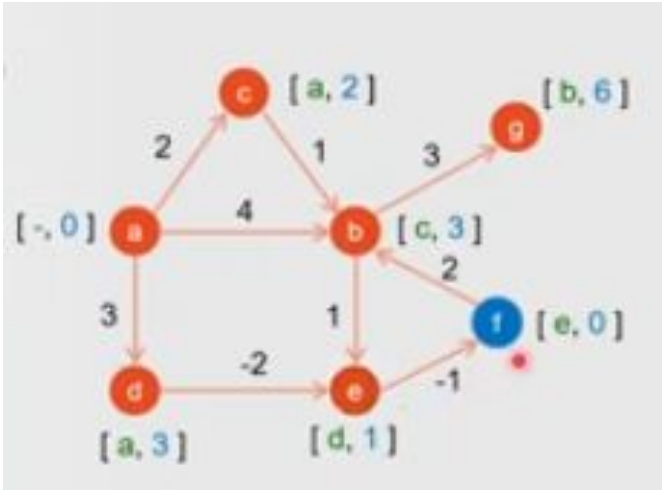




Ejemplo

-Nos queda el nodo **f** para **validar**

- f tiene de vecino al **b**



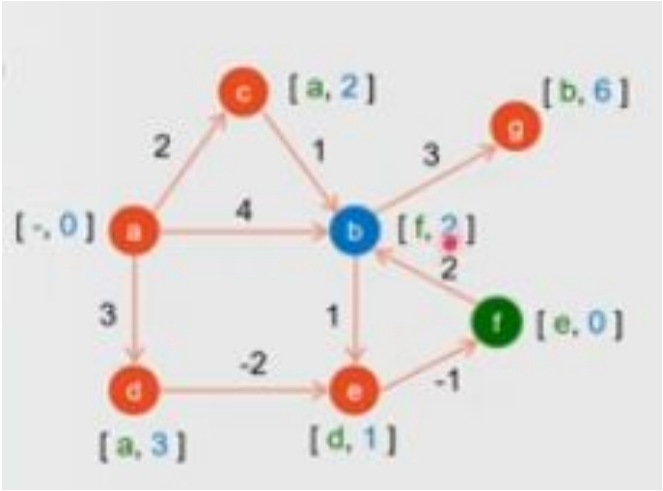
Ejemplo

-**Etiqueta:** 0+2=2<3 => **[f,2]**

- EL NODO B VUELVE A MARCARSE COMO HABILITADO A VISITAR (estaba **deshabiltado**, por haberse **editado etiqueta)**

=> puede haber **otro camino** con un **costo menor**)

HAY QUE VOLVER A COMPUTARLO!



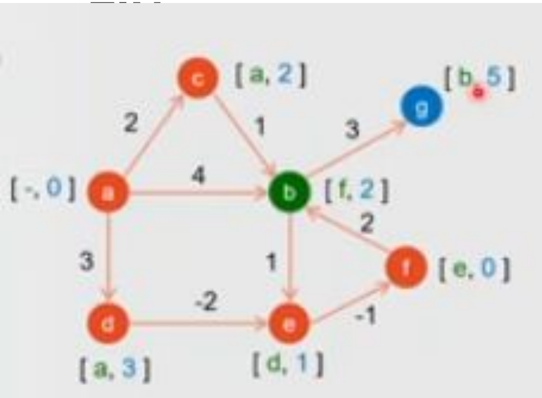
Ejemplo

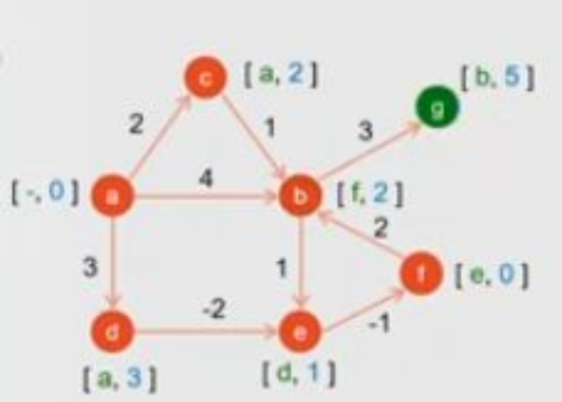
-Se **desmarca b** y se **analiza** el **g** (en el caso de **e**, el costo es mayor, así que **no se modifica la etiqueta**)

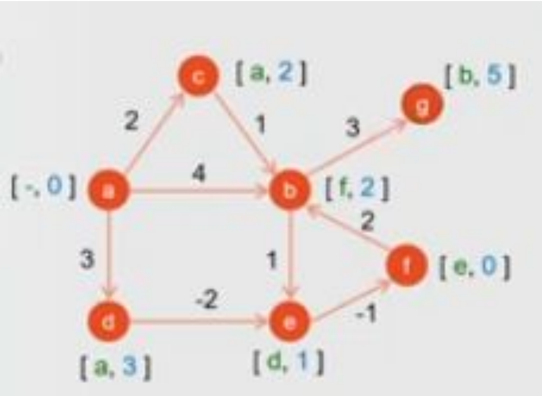
-**Etiqueta de g:** 2+3=5<6 => se reemplaza **[b,5]**

- **g** se vuelve a **encolar**

-Como g no tiene vecinos, se marca **verde**, **naranja** y no quedan nodos en **cola**





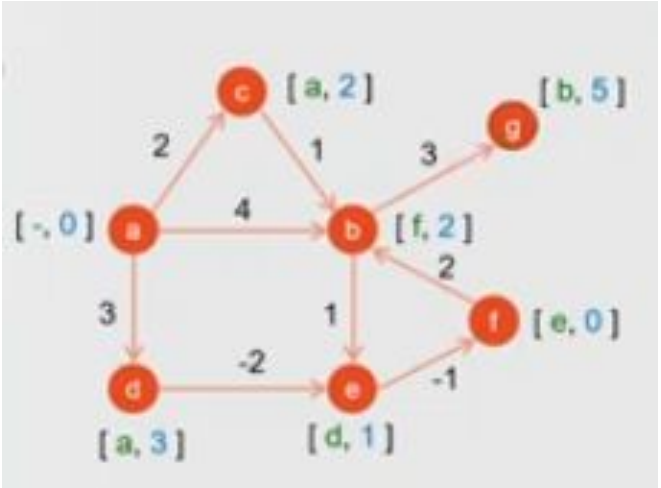


Ejemplo

Camino mínimo (a,g) = [a,d,e,f,b,g] -> Se hace de igual manera viendo **predecesores**

Costo mínimo = 5.

Es el **numero** que aparece en la etiqueta de **g**



Ejercicio

Implementar **Dijkstra mejorado y recursivo** para grafos con representación de nodos y aristas, de acuerdo al siguiente __init__:

```
    def __init__(self):
        self.nodos: set[T] = set()
        self.aristas: dict[tuple[T, T], int] = {}
```

def dijkstra_recursivo(self, inicio: T, destino: T) -> list[T]:

Nota: modificar el método **agregar_arista** para que soporte la asignación de un peso.