

# Matemática Discreta I

## Clase 21 - Caminatas eulerianas, ciclos hamiltonianos

FAMAF / UNC

3 de junio de 2021

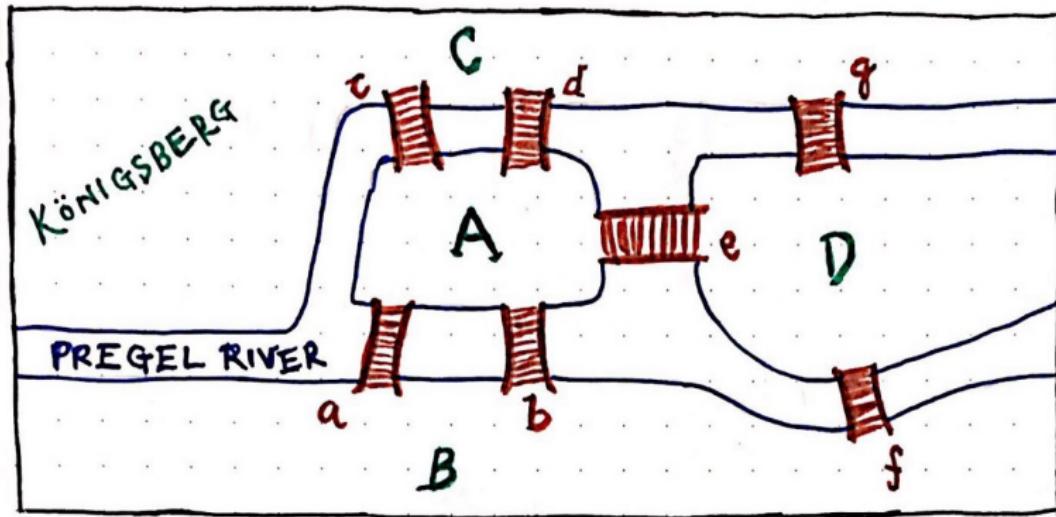
# Los puentes de Könisberg

## Pregunta

¿Es posible cruzar todos los puentes pasando una y solo una vez por cada uno? Leonhard Euler (1707-1783).

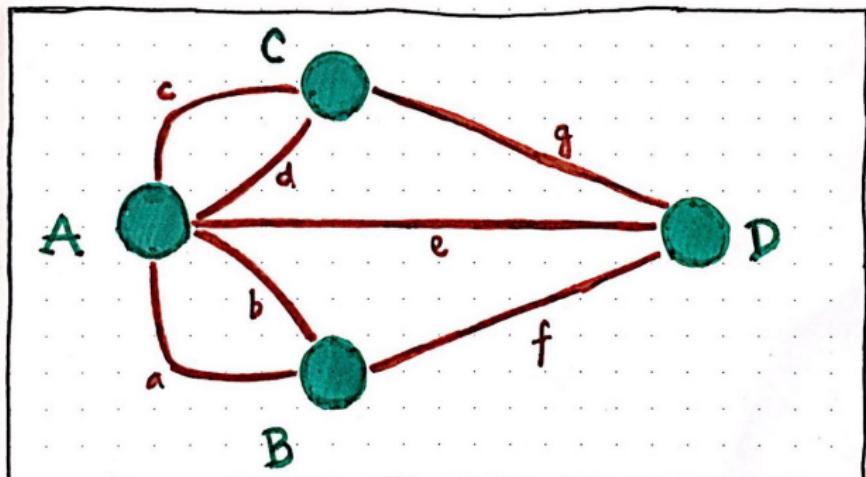


## Los puentes de Königsberg - versión 2



The Seven Bridges of Königsberg

## Los puentes de Königsberg - versión 3



*The Seven Bridges of Königsberg — Revisualized*

Ahora el problema se reduce a encontrar una caminata que use cada arista (o puente) una y solo una vez.

## Observación

$$\delta(A) = 5, \delta(B) = 3, \delta(C) = 3, \delta(D) = 3.$$

Euler, abstrayéndose del problema concreto, razonó de la siguiente manera:

- Supongamos que el vértice de partida es  $x$  y el de llegada es  $y$ . Todos los demás los llamo vértices intermedios.
- En un vértice intermedio cada vez que entro por un puente salgo por otro. Eso me “aporta” 2 a la valencia.
- Terminado el proceso usé todos los puentes, por lo tanto en cada vértice intermedio la cantidad de entradas más la cantidad de salidas es la valencia. Como la cantidad de entradas es igual a la cantidad de salidas, la valencia en los vértices intermedios es par.

## Concluyendo

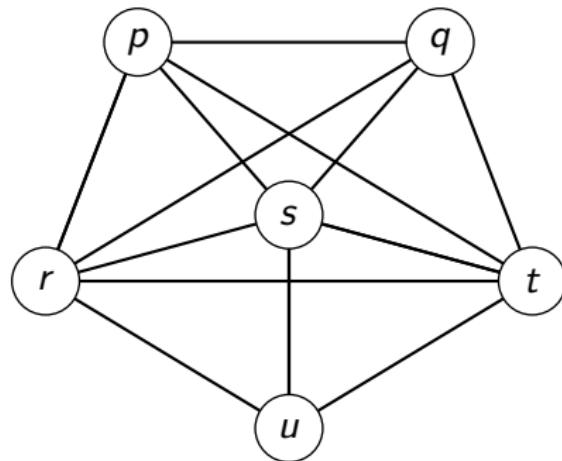
- Si hay una caminata que pasa por cada puente una y solo una vez, la cantidad de vértices con valencia impar es a lo sumo 2.
- Como todos los vértices en el problema tienen valencia impar, el problema no tiene solución.

Como veremos más adelante el problema de los puentes de Könisberg puede ser generalizado y también vale la recíproca.

**Cita:** Euler, Leonhard (1736). *Solutio problematis ad geometriam situs pertinentis*. Comment. Acad. Sci. U. Petrop 8, 128-40 (en latín)

## Ejemplo

¿Es posible recorrer el siguiente gráfo con una caminata que pase por cada vértice una sola vez y volver al de partida? ¿Es posible hacer una caminata que pasa por cada arista una sola vez?



## Solución

Para la primera pregunta una posibilidad es el ciclo  $p, q, t, s, u, r, p$ .

La segunda pregunta tiene respuesta negativa:

Las valencias son

$$\delta(p) = 4, \quad \delta(q) = 4, \quad \delta(s) = 5, \quad \delta(r) = 5, \quad \delta(u) = 3, \quad \delta(t) = 5.$$

Como en los puentes de Königsberg, al haber más de 2 vértices de valencia impar, no es posible recorrer todas las aristas una sola vez.



## Definición

Un *ciclo hamiltoniano* en un grafo  $G$  es un ciclo que contiene a todos los vértices del grafo.

Una *caminata euleriana* en un grafo  $G$  es un caminata que usa todas las aristas de  $G$  exactamente una vez. Una caminata euleriana que comienza y termina en un mismo vértice se llama también *círculo euleriano*.

## Teorema

Un grafo conexo con más de un vértice tiene un circuito euleriano si y sólo si todos los vértices tienen grado par. Un grafo conexo con más de un vértice posee caminatas eulerianas de  $v$  a  $w$ , con  $v \neq w$  si y sólo si  $v$  y  $w$  son los únicos vértices de grado impar.

C. Hierholzer (1840-1871) mostró, poco antes de su muerte, un algoritmo fácilmente implementable para encontrar un circuito euleriano para cualquier grafo con vértices de grado par.

El mismo algoritmo sirve para grafos con dos vértices de grado impar.

Para explicar el algoritmo de Hierholzer nos es útil la siguiente definición: un *recorrido maximal* es un recorrido que respetando la condición de no repetir aristas no es posible continuarlo.

La idea clave del algoritmo de Hierholzer es la siguiente: en un grafo de valencias pares (conexo o no conexo) todo recorrido maximal termina en el vértice original.

# Idea de la demostración (1)

Algoritmo de Hierholzer (grafo con todas valencias pares):

- (1) **Paso 1.** Elija cualquier vértice inicial  $v$  y haga un recorrido maximal.  
(recorrido cerrado, puede no cubrir todas las aristas).

## (2) Paso iterativo

- i) Mientras exista un vértice  $u$  en la caminata ya realizada, pero que tenga aristas que no formen parte de la caminata, inicie otro recorrido maximal (será de  $u$  a  $u$ ) siguiendo las aristas no utilizadas.
- ii) Inserte esta caminata a la caminata anterior en  $u$  para formar una caminata nueva (más larga).
- iii) Si no cubrió todas las aristas vuelva a i).

## Idea de la demostración (2)

En el paso iterativo el subgrafo que obtenemos luego de quitar las aristas recorridas es par. Esto nos permite hacer caminatas cerradas por aristas no utilizadas desde cada vértice con aristas no utilizadas.

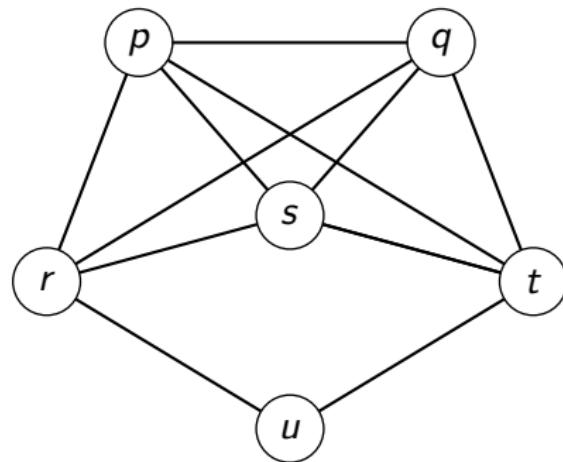
El caso de un grafo donde todas las valencias son pares excepto 2 se puede reducir al anterior: si deseamos una caminata euleriana que empiece por  $v$  y termine en  $w$ .

- Comenzamos en  $v$  y haga una caminata que no repita aristas hasta que se detenga.
- Elimine la aristas utilizadas. El grafo que queda es par y ahí utilizamos el algoritmo para ese caso.



## Ejemplo

Encontremos un circuito euleriano del siguiente grafo.



## Solución

Debemos primero observar que debe existir un circuito euleriano, pues  $\delta(p) = 4$ ,  $\delta(q) = 4$ ,  $\delta(r) = 4$ ,  $\delta(s) = 4$ ,  $\delta(t) = 4$ , y  $\delta(u) = 2$ , es decir todos los vértices tienen grado par.

Apliquemos el algoritmo de Hierholzer partiendo desde  $u$ . Una caminata posible con origen en  $u$  y que vuelva a  $u$  es

$$u, r, p, q, s, t, u.$$

Ahora elijamos  $q$  que es un vértice que pertenece a la caminata pero que tiene aristas que no son parte de la caminata. Una caminata que no toca aristas usadas y que parte de  $q$  y regresa a  $q$  es  $q, t, p, s, r, q$ . Insertamos en  $q$  esta caminata a la caminata anterior y obtenemos:

$$u, r, p, \mathbf{q}, t, p, s, r, \mathbf{q}, s, t, u.$$

(En negrita la caminata insertada).



# Recorrido maximal

Mostramos en pseudocódigo el algoritmo de recorrido maximal.

```
def r_max(L, v_ini):
    # pre: L grafo, v_ini vértice de L
    # post: recorrido maximal que comienza en v_ini
    sub_caminata = [v_ini]  # sub caminata
    p0 = v_ini
    while len(L.adyacentes(p0)) > 0:
        p1 = L.adyacentes(p0)[0] # p1 primer adyacente a p0
        sub_caminata.append(p1) # agrega p1 a caminata
        L.quitar_arista((p0,p1)) # quita arista {p0, p1}
        p0 = p1
    return sub_caminata
```

# Algoritmo de Hierholzer

Mostramos en pseudocódigo el algoritmo de Hierholzer

```
# pre: G grafo admisible
# post: cuando termina 'c' es una caminata
#       o circuito euleriano que empieza en 0.
libres = G.copiar() # sub grafo de aristas no utilizadas
c = r_max(libres, v_ini) # recorrido maximal
while len(libres.aristas()) > 0:
    for v in libres.vertices():
        if len(libres.adyacentes(v)) > 0 and v in c:
            pos = cam.index(v)
            c  =  c[:pos] + r_max(libres, v) + c[pos+1:]
```