

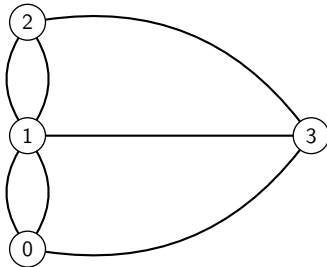
Els grafs: xarxes, camins i connexions

De la matemàtica discreta a la realitat

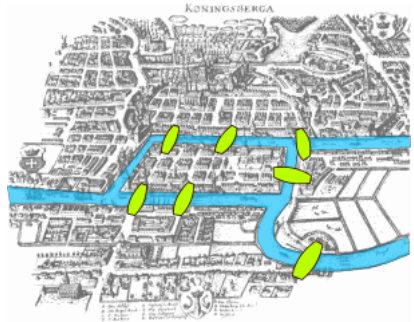
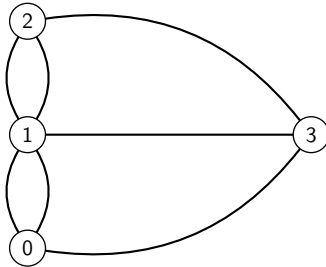
Aniol Garcia i Serrano

Presentació del treball, Gener 2017

El primer graf



El primer graf



Breu història



Leonhard Euler



Gottfried W. Leibniz

Els grafs: xarxes, camins i connexions

De la matemàtica discreta a la realitat

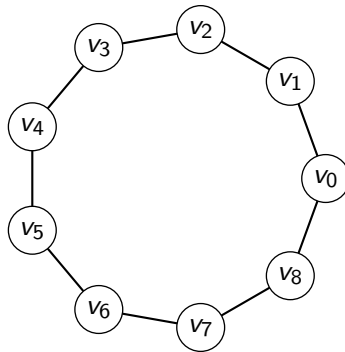
Aniol Garcia i Serrano

Presentació del treball, Gener 2017

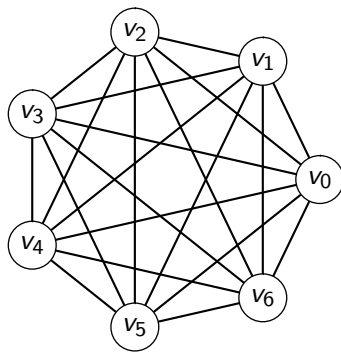
Objectius

- Conèixer la teoria de grafs
- Estudiar i implementar algorismes
- Mostrar-ne algunes de les aplicacions

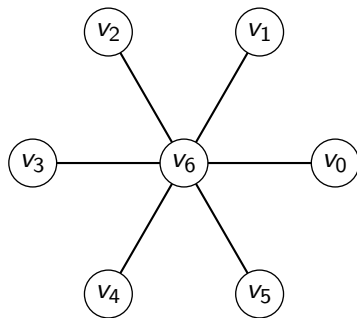
Tipus de grafs



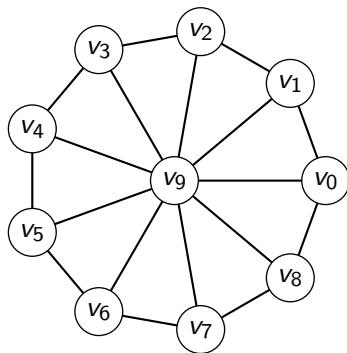
Cicle



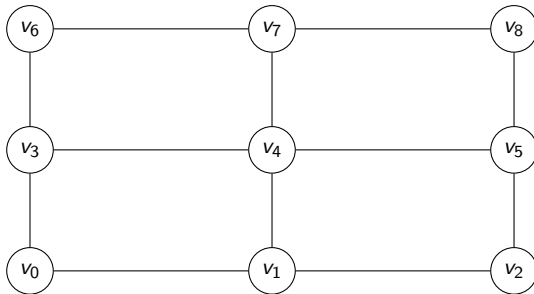
Complet



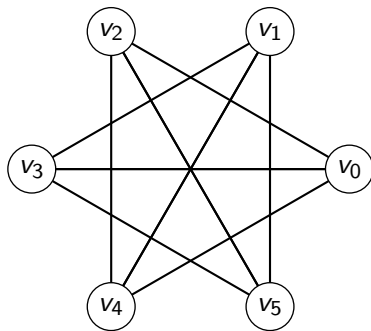
Estrella



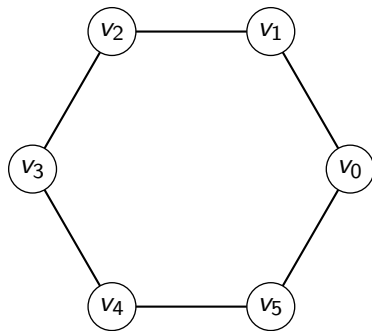
Roda



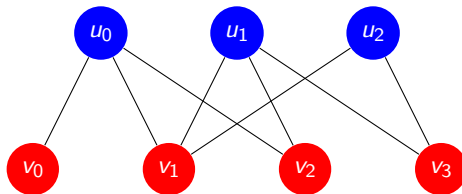
Xarxa



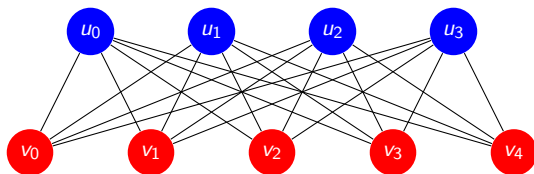
Graf G



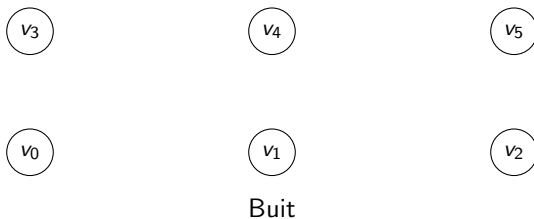
Complementari de G



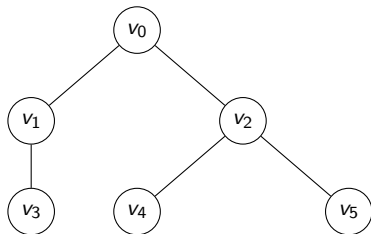
Bipartit



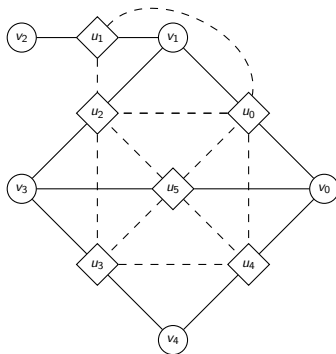
Bipartit complet



Nul



Arbre



Graf Lineal

Propietats i demostracions

A tall d'exemple:

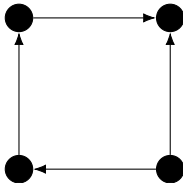
Propietat: El graf lineal d'un graf amb n nodes, e arestes i amb vèrtexs de graus $g(v_i)$ té $n' = e$ nodes i e' arestes, on

$$e' = \frac{1}{2} \sum_{i=1}^n g(v_i)^2 - e$$

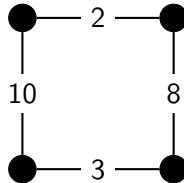
Demostració: Cada node v_i amb grau $g(v_i)$ del graf original generarà un graf complet de $g(v_i)$ nodes ($K_{g(v_i)}$). Un graf complet té $\binom{n}{k} = \frac{n(n-1)}{2}$ arestes, per tant, en aquest cas se'n generen $\frac{g(v_i)(g(v_i)-1)}{2}$. Però això es compleix per a cada vèrtex, i llavors podem escriure

$$\sum_{i=1}^n \frac{1}{2} g(v_i)(g(v_i)-1) = \frac{1}{2} \sum_{i=1}^n (g(v_i)^2 - g(v_i)) = \frac{1}{2} \sum_{i=1}^n g(v_i)^2 - \underbrace{\frac{1}{2} \sum_{i=1}^n g(v_i)}_{\substack{2|E| \\ |E|}} = \frac{1}{2} \sum_{i=1}^n g(v_i)^2 - e$$

Altres classificacions



Graf dirigit

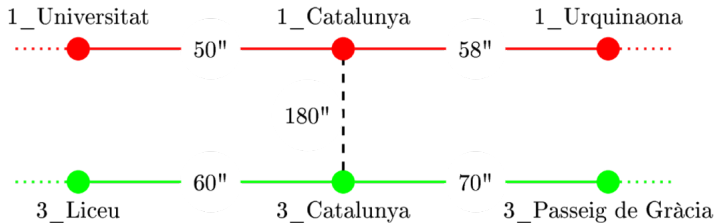


Graf ponderat

Algorismes

- Camins
 - Euler
 - Hamilton
 - Dijkstra
 - Bellman-Ford
 - Floyd-Warshall
- MST
 - Kruskal
 - Prim
- Exploració
 - DFS

Organització




```
def metro(Adj, inici, final, k): #on k és el temps de parada acada estació
    recorregut=[]

    print "Punt inicial:", inici.decode("ISO-8859-15")

    print "Punt final:", final.decode("ISO-8859-15")

    dist, tree = OrderedDijkstra(Adj, inici, k)

    i = final
    while tree[i] != inici:
        recorregut.append(tree[i])
        i = tree[i]

    recorregut.append(inici)
    recorregut.reverse()

    total= dist[final]-k

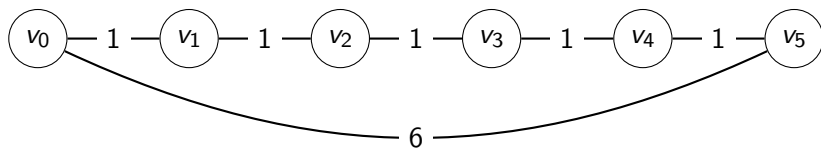
    print "Temps amb estacions del recorregut:", dist[final], "Temps real:", total

    if total < 60:
        print "Temps total del recorregut:",int(total), "segons"
    else:
        minuts = total/60
        segons = (total%60)
        print "Temps total del recorregut:", int(minuts),"minuts i", int(segons), "segons"

    print "Recorregut:",
    print "[",
    for i in range(0,len(recorregut)):
        print recorregut[i].decode("ISO-8859-15")+",",

    print final.decode("ISO-8859-15"),"]"
```

```
def OrderedDijkstra(Adj, s, k):  
    Q = dict.fromkeys(Adj.keys(), float("inf"))  
    dist = dict.fromkeys(Adj.keys(), float("inf"))  
    tree = {}  
    Q[s] = 0  
    while Q:  
        u = min(Q, key=Q.get)  
        dist[u] = Q[u]  
        for v in Adj[u]:  
            if v in Q:  
                if Q[v] > Q[u] + Adj[u][v]:  
                    Q[v] = Q[u] + Adj[u][v] + k  
                    tree[v] = u  
        Q.pop(u)  
  
    return dist, tree
```

Exemple d'execució

```
metro(metro_barcelona, "4_Llucmajor", "9S_Aeroport T1", 25)
```

Punt inicial: 4_Llucmajor

Punt final: 9S_Aeroport T1

Temps net del recorregut: 3565

Temps total del recorregut: 59 minuts i 25 segons

Recorregut: [4_Llucmajor, 4_Maragall, 4_Guinardó Hospital de
→ Sant Pau, 4_Alfons X, 4_Joanic, 4_Verdaguer, 5_Verdaguer,
→ 5_Diagonal, 5_Hospital Clínic, 5_Entença, 5_Sants Estació,
→ 5_Plaça de Sants, 5_Badal, 5_Collblanc, 9S_Collblanc,
→ 9S_Torrassa, 9S_Can Tries Gornal, 9S_Europa Fira, 9S_Fira,
→ 9S_Parc Logístic, 9S_Mercabarna, 9S_Les Moreres, 9S_El Prat
→ Estació, 9S_Cèntric, 9S_Parc Nou, 9S_Mas Blau, 9S_Aeroport
→ T2, 9S_Aeroport T1]

Conclusions

- Objectius assolits.
- Podem trobar grafs a una gran quantitat d'àmbits i amb moltes aplicacions diferents.
- Importància de la computació en procediments matemàtics.
- Queda molta feina per fer.

Programari lliure

- Ubuntu i Debian com a S.O.
- Python com a llenguatge de programació
- LaTeX i els seus paquets per a la redacció de la memòria
- Git com a sistema de control de versions
- Geogebra, Spyder, Scilab, Vim, Meld... i molts d'altres!

Més informació del treball a
aniolgarcia.github.io/grafs

Moltes gràcies!

Aniol Garcia i Serrano
aniolgarcia@gmail.com