Pràctica Final de Curs

Ernest Jorba Calafell

5 de juny de 2022

Resum

Aquest document és la memòria de la pràctica de final de curs realitzada per l'Ernest Jorba Calafell en l'assignatura de Programació - Informàtica 1

Índex

1 Extres

- Menú gràfic realitzat amb la llibreria *Tkinter*. Aquest menú es troba i es pot executar en el fitxer *menu_grafic.py*.

- He substituït les classes *Conjunt* i *Matriu* per les funcions pròpies de Python, és a dir, *set* i *numpy* respectivament.

- Mètode per veure si un gràf és fortament connex o no. Això es pot fer mitjançant la matriu que et retorna l'algoritme de Warshall i la definició del mateix. Si en aquesta matriu tenim algun zero, vol dir que des d'algun vèrtex u no tenim cap camí que ens porti a v i, per tant, no és fortament connex.

- Llistar els fitxers que tens guardats per estalviar temps a l'usuari (i possibles errors) al moment d'especificar el graf que vol carregar i aplicar algoritmes o modificar.

- Poder veure el graf en format PDF. Això ho he aconseguit gràcies a la llibreria *graphviz*

- Possibilitat de veure l'arbre minimal. Això ho faig gràcies al conjunt que retorna l'algoritme de Prim. Com que cada element del conjunt és un parell ordenat de dos vèrtexs (és a dir, una aresta), només cal que vagi iterant per tots els elements del conjunt i anar afegint les arestes a un graf buit, que li dic *arbre*.

- Modificació a l'algoritme de Prim. Con ja he escrit en algun comentari en el codi, l'algoritme de Prim de la manera que l'hem implementat no funciona del tot bé en grafs que tenen un o més vèrtexs desconectats (grau d'entrada i sortida igual a zero). La modificació que he fet ha sigut identificar aquests casos i enviar un missatge d'error per evitar que retorni un resultat erroni.

  Aquests casos són quan es forma algun llaç (aresta que surt i entra del mateix vèrtex), i són fàcilment identificables en el moment d'afegir l'aresta al conjunt T perquè els dos extrems són iguals.

- En la classe Matriu he sobreescrit els mètodes *__mult__* i *__pow__* perquè abans de canviar la classe Matriu per numpy, la manera que tenia de veure si un graf era fortament connex o no era mitjançant un sumatori de matrius. En els apunts de Matemàtica Discreta tenim la següent proposició:

Sigui A^k la potència k-èssima de la matriu A i sigui $a_{i,j}^{(k)}$ l'entrada en la fila i, columna j de la matriu A^k. Llavors, $a_{i,j}^{(k)}$ compta el nombre de recorreguts entre v_i i v_j de longitud k.

El que es pot fer per veure si un graf és fortament connex o no, és mirant si en

$$B = \sum_{k=1}^{d} A^K$$

la matriu B hi ha algun zero, on d és el diàmetre del graf. Si és que sí, el graf no és fortament connex. Altrament sí que ho és.

2 Exemples

2.1 Graf 1

- Fitxer del graf:

  4

  0 1 1

  1 2 3

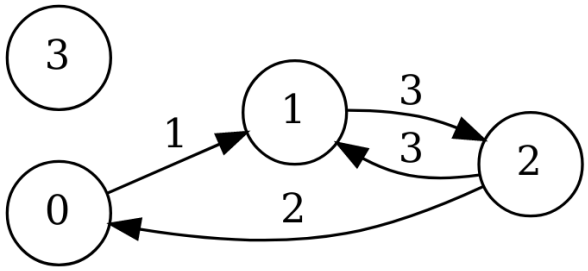
  2 0 2

  2 1 3

- Matriu d'adjacència del graf:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

- Visualització del graf amb *graphviz*:



- Veure si és connex:

  No és connex

- Algoritme de Profunditat amb inici 1:

  {0, 1, 2}

- Algoritme de Warshall:

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

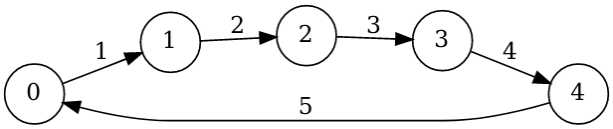
- Algoritme de Prim i visualització de l'arbre minimal:
  No es pot aplicar l'algoritme de Prim a aquest graf

2.2 Graf 2

- Fitxer del graf:
  5
  0 1 1
  1 2 2
  2 3 3
  3 4 4
  4 0 5
- Matriu d'adjacència del graf:

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 4 \\ 5 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- Visualització del graf amb *graphviz*:



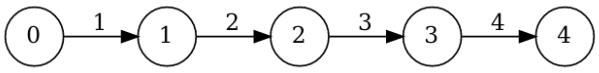
- Veure si és connex:
  Sí que és connex
- Algoritme de Profunditat amb inici 0:
  {0, 1, 2, 3, 4}
- Algoritme de Warshall:

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

- Algoritme de Prim:

  {(0, 1), (1, 2), (2, 3), (3, 4)}

- Arbre minimal:



2.3 Graf 3

- Fitxer del graf:

  5

  0 1 1

  1 2 2

  2 0 4

  2 3 3

  3 0 1

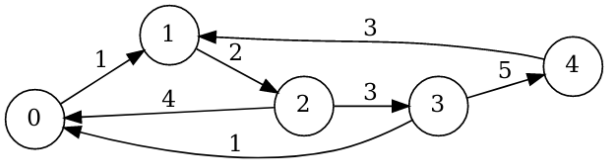
  3 4 5

  4 1 3

- Matriu d'adjacència del graf:

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 4 & 0 & 0 & 3 & 0 \\ 1 & 0 & 0 & 0 & 5 \\ 0 & 3 & 0 & 0 & 0 \end{pmatrix}$$

- Visualització del graf amb *graphviz*:



- Veure si és connex:

  Sí que és connex

- Algoritme de Profunditat amb inici 4:

  $\{0, 1, 2, 3, 4\}$

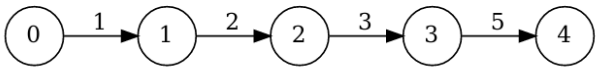
- Algoritme de Warshall:

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

- Algoritme de Prim:

  $\{(0, 1), (1, 2), (2, 3), (3, 4)\}$

- Arbre minimal:



2.4 Graf exemple del Moodle

- Fitxer del graf:

  6

  0 1 4

  0 2 3

  1 2 3

  1 2 5

  1 3 2

2 3 7

3 4 2

4 0 4

4 1 4

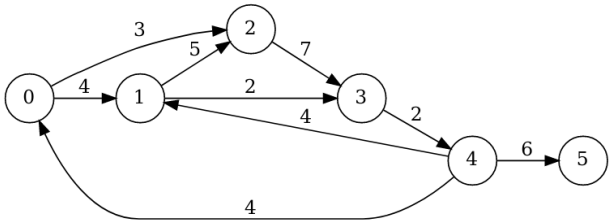
4 5 6

Nota: En el fitxer hi ha dues arestes que van del vèrtex 1 al 2, però com que el

- Matriu d'adjacència del graf:

$$\begin{pmatrix} 0 & 4 & 3 & 0 & 0 & 0 \\ 0 & 0 & 5 & 2 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 4 & 4 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- Visualització del graf amb *graphviz*:



- Veure si és connex:

  No és connex

- Algoritme de Profunditat amb inici 0:

  {0, 1, 2, 3, 4, 5}

- Algoritme de Warshall:

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

- Algoritme de Prim:
  {(0, 1), (3, 4), (4, 5), (0, 2), (1, 3)}
- Arbre minimal: