Problemes algorísimia

Miquel Pere Baztán Grau
Bernat Dosrius Lleonart
Xavier Momplet Gil
Emma Ventura Font

Universitat Politècnica de Catalunya

21 de desembre 2024

Setmana 2 (20 febrer)

Problema 1.3 (celebritat?)

En una festa, un convidat es diu que és una celebritat si tothom el coneix, però ell no coneix a ningú (tret d'ell mateix). Les relacions de coneixença donen lloc a un graf dirigit: cada convidat és un vèrtex, i hi ha un arc entre u i v si u coneix a v.

(a) **Doneu una formalització de la propietat de ser celebritat.**

Sigui $G = (V, E)$ un graf dirigit. Una celebritat és un vèrtex

$$v \in V \mid \forall v' \in V \ v' \neq v \implies (v', v) \in E \ i \ (v, v') \notin E$$

(b) **Doneu un algorisme que, donat un graf dirigit representat amb una matriu d'adjacència, indica si hi ha o no cap celebritat. En el cas que hi sigui, cal dir qui és. El vostre algorisme ha de funcionar en temps $\mathcal{O}(n)$, on n és el nombre de vèrtexs.**

Notem que, per la definició de celebritat, només pot existir una celebritat donat un graf dirigit simple. També cal notar que la matriu amb la que treballarem en aquest problema és una matriu que només té $1's$ i $0's$ a les seves entrades. Sigui a_{ij} l'entrada de la i-èssima fila i la j-èssima columna de la matriu d'adjacència, aleshores:

$$a_{ij} = \begin{cases} 0 & \text{si } (i,j) \in E \\ 1 & \text{si } (i,j) \notin E \end{cases}$$

Per a determinar si el graf té una celebritat, començarem des de la posició $(0, 0)$ de la matriu, anem augmentant j fins a trobar un 1, això ens indica que el vèrtex de la fila en la que ens trobem no és famós (ja que coneix a un altre vèrtex). Un cop trobat aquest 1 saltem a la fila j ($i = j$) i seguim agumentant j fins a trobar un 1 o fins a arribar a $j == n$. En el moment en que $j == n$, tenim un candidat a celebritat (i-èssim vèrtex). Per a comprovar si i és una celebritat cal recòrrer la i-èssima fila i la j-èssima columna, cal comprovar:

$$\forall\ k \in [n-1] \mid k \neq i\ a_{kj} == 1$$

$$\forall\ l \in [n-1]\ a_{il} == 0$$

Aquestes dues condicions ens diuen que el vèrtex i és conegut per tots els altres vèrtex i ell mateix no coneix a cap altre vèrtex i, per tant, és una celebritat. Si no es verifiquen les dues condicions anteriors quan $j == n$, aleshores no existeix cap celebritat en G.

Problema 1.9 (és fortament connex?)

Un graf dirigit és fortament connex quan, per cada parell de vèrtexs u, v, hi ha un camí de u a v. Doneu un algorisme per determinar si un graf dirigit és fortament connex.

La idea de l'algorisme és fer un DFS (arrelant l'arbre del recorregut en un node $v \in G$ qualsevol) i adonar-nos de que un graf dirigit serà fortament connex si i només sí, és semiconnex i quan ordenem els vèrtexos (segons l'ordre de visita $ndfs[v]$, $v \in G$), passa el següent:

$\forall u \in G$ tal que $u \neq v$ (l'arrel) el número més baix que es pot arribar seguint un camí des de u fins a algun dels seus descendents en l'arbre de recorregut del DFS (pujant amb una aresta de retrocès), anomenem-lo $mesAlt[u]$, és més petit estricte que $ndfs[u]$. És a dir, que tot vèrtex (excepte l'arrel) es pot "escapar"(o tornar) del seu pare i anar més amunt de l'arbre. Matemàticament, $mesAlt$[u] $<$ $ndfs$[u].

Amb un dibuix,

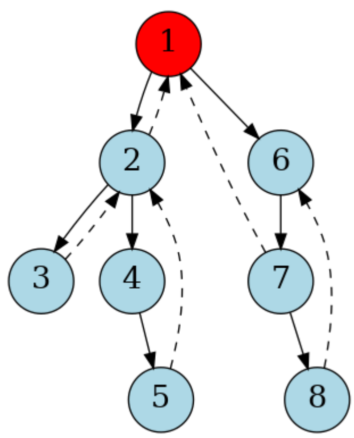

Figura 1: Exemple d'arbre fortament connex (les arestes discontínues són les arestes de retrocès)

L'algorisme és el següent:

Algorithm 1 DFS(G)

Require: G graf
Ensure: G és fortament connex?

```
for v ∈ V(G) do
    visitat[v] := false
    ndfs[v] := 0
end for

numDfs := 0
esFortamentConnex := true

v:= un vèrtex qualsevol de G
FortConnRec(G, v, v, esFortamentConnex, ndfs, visitat)

if not esFortamentConnex then
    return false
end if

for v ∈ V(G) do
    if not visitat[v] then
        return false
    end if
end for

return esFortamentConnex
```

Algorithm 2 FortConnRec(G, v, pare, esFortamentConnex, ndfs, visistat)

Require: G graf

```
numDfs := numDfs + 1; ndfs[v] := numDfs
visistat[v] := true
mesAlt[v] = ndfs[v]

for w ∈ G.ADJACENT(v) do
    if not visitat[w] then
        FortConnRec(G, w, v, esFortamentConnex, ndfs, visitat)
        mesAlt[v] := min(mesAlt[v], mesAlt[w])
    else
        mesAlt[v] := min(mesAlt[v], ndfs[w])
    end if
end for

if pare != v then
    esFortamentConnex := esFortamentConnex or mesAlt[v] ≥ ndfs[v]
end if
```

Nota: Per calcular mesAlt[v] usem el mesAlt[w] dels fills directes que no hem visitat (arestes contínues de l'arbre de la figura) i el ndfs[w] dels nodes que ja hem visitat (arestes discontínues) i prenem el mínim de tots ells.

Problema 1.10 (és semiconnex?)

Un graf dirigit $G = (V, E)$ és semiconnex si, per qualsevol parell de vèrtexs $u, v \in V$, tenim un camí dirigit de u a v o de v a u. Doneu un algorisme eficient per determinar si un graf dirigit G és semiconnex. Demostreu la correctesa del vostre algorisme i analitzeu-ne el cost. Dissenyeu el vostre algorisme fent us d'un algorisme que us proporcioni les components connexes fortes del graf en temps $\mathcal{O}(n + m)$.

Problema 1.12 (clique max?)

Donat un graf no dirigit $G = (V, E)$ i un subconjunt de vèrtex V1, el subgraf induït per $V1$, $G[V1]$ té com a vèrtex $V1$ i con a arestes totes les arestes a E que connecten vèrtexs en $V1$. Un clique és un subgraf indiut per un conjunt C on tots els vèrtexs estan connectats entre ells. Considereu el següent algorisme de dividir-i-vèncer per al problema de trobar un clique en un graf no dirigit $G = (V, A)$.

CliqueDV(G)

Enumereu els vèrtexs V com $1, 2, \ldots, n$, on $n = |V|$
Si $n = 1$ tornar V
Dividir V en $V_1 = \{1, 2, \ldots, \lfloor n/2 \rfloor\}$ i $V_2 = \{\lfloor n/2 \rfloor + 1, \ldots, n\}$
Sigui $G_1 = G[V_1]$ i $G_2 = G[V_2]$
C_1= CliqueDV(G_1) i C_2= CliqueDV(G_2)
$C_1^+ = C_1$ i $C_2^+ = C_2$
for $u \in C_1$ **do**
    if u està connectat a tots els vèrtexs a C_2^+ **then**
        $C_2^+ = C_2^+ \cup \{u\}$
for $u \in C_2$ **do**
    if u està connectat a tots els vèrtexs a C_1^+ **then**
        $C_1^+ = C_1^+ \cup \{u\}$
Retorneu el més gran d'entre C_1^+ i C_2^+

Figura 2: Enter Caption

Contesteu les següents preguntes:

(a) Demostreu que l'algorisme CliqueDV sempre retorna un subgraf de G que és un clique.

(b) Doneu una expressió asimptòtica del nombre de passos de l'algorisme CliqueDV.

(c) Doneu un exemple d'un graf G on l'algorisme CliqueDV retorna un clique que no és de grandària màxima.

(d) Creieu que és fàcil modificar CliqueDV de manera que sempre done el clique màxim, sense incrementar el temps pitjor de l'algorisme? Expliqueu la vostra resposta

Assignació setmana 2 (20 febrer): 1.3, 1.9, 1.10, 1.12 Extra 1: Definiu formalment la propietat connex(G) que descriu si un graf G=(V,E) és connex. Extra 2: Definiu formalment la propietat clique(G) que descriu si un graf G=(V,E) és complet. Extra 3: Sigui B(n) el nombre de fulles en un arbre binari complet amb n nodes. Demostreu mitjançant inducció que: B(n)= n+1 / 2, per tot n 1. (per a tots els grups)

Extra 1

Donat un graf $G = (V, E)$ direm que aquest graf és connex:

$$G \text{ és connex} \iff \forall u, v \in V \mid v \neq u \ \exists \ C(u, v)$$

On $C(u, v)$ és un camí entre u i v que definirem de la següent forma:

$$C(u, v) = \{u_0, u_1, \ \dots \ , u_n \in V \mid u_0 = u, \ u_n = v, (u_i, u_{i+1}) \in E \ \ \forall i \in [n-1]\}$$

Extra 2

Donat un graf $G = (V, E)$ direm que aquest graf és complet:

$$G \text{ és complet} \iff \forall v, u \in V \mid u \neq v \ (v, u) \in E$$

Extra 3

Volem veure que tot arbre binari complet amb n nodes té un total de $\frac{n+1}{2}$ fulles, entenent com a fulles aquells nodes que no són pares de cap altre node. Notem que només existeixen arbres binaris complets amb n senar ($n = 1 + 2 * pares$). Ho veurem per inducció:

Cas base ($n = 1$):

En aquest cas només tenim $n = 1$ nodes, i per tant, només tenim $\frac{n+1}{2} = 1$ fulles, per tant, en el cas base es verifica la propietat que volem demostrar.

Pas inductiu:

Suposem que tenim un arbre binari complet amb $n-1$ nodes i suposem que la propietat que volem demostrar és certa, aleshores tenim un total de $\frac{n}{2}$ fulles. Ara, per a construir un arbre binari complet de $n+1$ nodes hem d'afegir dos fills a qualsevol d'aquestes fulles (el cas d'un arbre amb n nodes no té sentit ja que només existeixen els arbres binaris complets amb n nodes si n és imparell, per tant si $n-1$ és imparell, el següent arbre binari complet tindrà $n+1$ nodes). Aleshores, la fulla que escollim deixa de ser-ho i passem a tenir dues noves fulles (els fills d'aquesta fulla). D'aquesta forma el còmput global de fulles queda:

$$\frac{n}{2} - 1 + 2 = \frac{n}{2} + 1 = \frac{(n+1)+1}{2}$$

Per tant hem vist que:

$$\forall B_n \text{ arbre binari complet amb } n \text{ nodes } (n \equiv 1 \ (mod2)) \implies \#Fulles(B_n) = \frac{n+1}{2}$$

Miquel Pere Baztán Grau
Bernat Dosrius Lleonart
Xavier Momplet Gil
Emma Ventura Font

Universitat Politècnica de Catalunya