Presentació

Aquesta PAC aprofundeix en el concepte de complexitat computacional que cobreix els continguts estudiats en els mòduls 6 i 7 de l'assignatura. Els exercicis treballen els conceptes de mesures de complexitat, la reducció i completesa, la classe NP-complet i alguns dels problemes intractables més importants que es coneixen.

Competències

En aquesta PAC es treballen les següents competències del Grau en Enginyeria Informàtica:

- Capacitat per utilitzar els fonaments matemàtics, estadístics i físics per comprendre els sistemes TIC.
- Capacitat per analitzar un problema en el nivell d'abstracció adequat a cada situació i aplicar les habilitats i coneixements adquirits per a resoldre'l.

Objectius

Els objectius concrets d'aquesta PAC són:

- Entendre els conceptes d'intractabilitat i no-determinisme.
- Conèixer les diferents classes de complexitat i saber classificar els problemes en cada una d'aquestes.
- Entendre el concepte de reducció entre problemes i saber demostrar quan un problema és NP-complet.
- Reconèixer problemes intractables que apareixen de forma habitual en informàtica i en enginyeria.
- Entendre i saber aplicar les tècniques bàsiques de reducció polinómica dels problemes NP-complets.

Descripció de la PAC

1. (Valoració d'un 20% = 10% + 10%)

   Donats els següents problemes:

   1) FLOYD: Donat un graf G, retorna la matriu de distàncies mínimes entre cada parell de vèrtexs.
   2) CLIQUE: Donat un graf G, retorna l'ordre t del subgraf complet (K_t) més gran que podem trobar a G.
   3) PARTICIO: Donat un conjunt d'enters S, retorna SÍ si es pot dividir el conjunt original en dos subconjunts $S_1, S_2 \subset S$ tals que $S_1 \cup S_2 = S$, $S_1 \cap S_2 = \emptyset$ i

   $$\sum_{s_1 \in S_1} s_1 = \sum_{s_2 \in S_2} s_2.$$

      En cas contrari, retorna NO.
   4) SUMA: Donat un conjunt S de n nombres, retorna la seva suma.
   5) CONJUNT_DOMINANT: Donat un graf G, retorna el menor subconjunt S de vèrtexs tal que, tot vèrtex $v \notin S$ serà adjacent a un vèrtex de S.

   Indiqueu:

   (a) La menor classe de complexitat a la que pertany cada problema i el seu tipus (decisió, càlcul, optimització). Justifiqueu breument la resposta.

   (b) Quines de les següents reduccions són possibles? Justifiqueu si existeix o no la reducció. En el cas que l'última reducció sigui possible, doneu una funció de reducció.

      i. FLOYD $\leq_p$ CLIQUE.
      ii. CONJUNT_DOMINANT $\leq_p$ FLOYD.
      iii. CONJUNT_DOMINANT $\leq_p$ CLIQUE.
      iv. SUMA $\leq_p$ FLOYD.

Solució:

(a) 1) FLOYD: Es tracta d'un problema d'optimització amb complexitat polinomial $O(n^3)$ i, per tant, pertany a P.

   2) CLIQUE: Es tracta d'un problema d'optimització. És un problema NP-Complet. Cal avaluar totes les combinacions de nodes fins a trobar el major t pel qual K_t és un subgraf de G. A més a més, cada conjunt pot verificar-se en temps polinòmic.

3) PARTICIO: Es tracta d'un problema de decisió NP-Complet. Existeixen, com en el cas anterior 2^n formes d'escollir el primer subconjunt S_1, quedant el segon subconjunt determinat de forma automàtica. A més a més, cadascuna d'aquestes opcions pot verificar-se en temps polinòmic.
4) SUMA: Es tracta d'un problema de càlcul i la seva complexitat és $O(n)$ ja que hem de recórrer tot el conjunt per sumar els seus elements. Per tant, pertany a P.
5) CONJUNT_DOMINANT: Es tracta d'un problema d'optimització (volem el menor dels subconjunts dominants) i és NP-Complet. Tal i com passa amb PARTICIO, requereix provar tots els subconjunts de vèrtexs per saber si són dominants i prendre el menor d'ells. A més a més, cadascun d'aquests subconjunts poden verificar-se en temps polinòmic.

(b) i. És possible. Com que FLOYD és un problema P, existeix una reducció a un problema NP.
ii. No és possible tret que $P = NP$. No podem reduir un problema NP a un problema P.
iii. És possible. Per ser tots dos problemes NP-Complets, poden reduir-se l'un a l'altre.
iv. És possible. Una reducció podria ser la creació d'un graf trajecte en el que els pesos de cada aresta siguin cadascun dels elements del conjunt a sumar. La major de les distàncies que retorni Floyd serà la suma de tots els elements.

---

2. (Valoració d'un 15% = 5% + 10%)

Considereu les següents expressions lògiques:

1) $(a \vee b) \wedge c$,
2) $(a \wedge b) \vee (c \wedge \bar{a})$,
3) $(\bar{a} \wedge (b \vee \bar{c})) \wedge (a \vee b)$.

(a) Transformeu a FNC aquelles que no estiguin en FNC.

(b) Empleneu la següent taula de veritat indicant quines expressions tenen valors de veritat per a cada cas. Existeix algun conjunt de valors que faci certes totes les expressions? I algun que les faci totes falses?

a	b	c	1)	2)	3)
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Solució:

(a) Transformació a FNC:

1) $(a \vee b) \wedge c$ ja està en FNC.
2) A continuació, es detalla la conversió pas a pas:
   * Apliquem la propietat distributiva: $(a \vee c) \wedge (a \vee \bar{a}) \wedge (b \vee c) \wedge (b \vee \bar{a})$.
   * Eliminem $a \vee \bar{a}$ ja que sempre és cert: $(a \vee c) \wedge (b \vee c) \wedge (b \vee \bar{a})$.
   * Eliminem $b \vee c$, ja que no afecta al resultat. Si b i c són certs, l'expresió és certa i, si tots dos són falsos, sempre serà falsa independentment de que aquest terme pertanyi o no a l'expressió. Finalment, obtenim

$$(\bar{a} \vee b) \wedge (a \vee c).$$

3) Aquesta expressió està en FNC encara que pot simplificar-se:
   * Canviem l'ordre dels termes: $\bar{a} \wedge (a \vee b) \wedge (b \vee \bar{c})$.
   * Podem simplificar $\bar{a} \wedge (a \vee b)$ per $\bar{a} \wedge b$, ja que a no pot ser cert i fals a la vegada: $\bar{a} \wedge b \wedge (b \vee \bar{c})$.
   * Com a l'apartat anterior, si b és certa, el tercer terme serà cert independentment del valor de $\bar{c}$, i si b és falsa, l'expressió serà falsa també de forma independent al valor de $\bar{c}$. Per tant, podem eliminar el tercer terme i obtenim

$$\bar{a} \wedge b.$$

(b) Taula de veritat:

a	b	c	1)	2)	3)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	1
1	0	0	0	0	0
1	0	1	1	0	0
1	1	0	0	1	0
1	1	1	1	1	0

De la taula de veritat podem comprovar que, per exemple, $a = 0$, $b = 0$, $c = 0$ fa falses totes les expressions, mentre que $a = 0$, $b = 1$, $c = 1$ les fa certes totes.

---

3. (Valoració d'un 20% = 5% + 5% + 5% + 5%)

Una entitat bancària que ha sofert diferents processos de reestructuració i integració amb altres entitats financeres, ha inclòs noves plataformes de pagament i interacció amb els seus usuaris i està en procés d'implantació de noves estratègies de classificació de clients mitjançant tècniques d'aprenentatge automàtic.

Cada dia es duen a terme milers de processos automatitzats per actualitzar els riscos dels clients, assegurar que totes les transaccions han acabat correctament, decidir l'efectiu a portar als caixers, etc.

Al banc detecten que cada vegada més processos no arriben a finalitzar-se mai, sent precís que els responsables del sistema els reiniciïn o els executin pas a pas, o que es realitzin processos manuals perquè acabin correctament.

- S'encarrega a l'empresa Grafs_i_Complexitat (GiC) que realitzi una anàlisi de processos. El conjunt de tots els processos a analitzar és $P = \{p_1, p_2, ..., p_n\}$ i es construeix un mapa de dependències on el procés p_j estarà connectat amb p_k si p_j depèn de p_k.
- Inicialment l'equip de GiC comença identificant els cicles d'ordre 3. Es determina que un dels objectius principals de l'anàlisi consistirà a identificar cicles de qualsevol longitud dins del mapa de processos.
- En alguna reunió es posa l'accent en que, si es dibuixa el diagrama d'un procés $p \in P$ i els processos dels quals depèn, en un cas ideal, es tractaria d'un arbre.
- Com a recomanació general a l'entitat bancària es recomana introduir mecanismes de traçabilitat en els processos (la informació de traçabilitat ens diu quan comença i acaba un procés, quin va ser el procés anterior i quin és el procés següent que s'executarà). Donat que el nombre de processos és extremadament alt i el temps de programació escàs, s'opta per assegurar que cada procés o bé inclogui informació de traçabilitat o la inclogui algun dels seus veïns.

Contesteu raonadament a les següents preguntes, tenint en compte que un inventari inicial ha identificat 25.000 processos i 20.000 dependències:

(a) Indiqueu si es tracta d'un graf dirigit o no dirigit, e identifiqueu quins són els vèrtexs i les arestes. Què vol dir en termes del graf que existeixin més processos que dependències?

(b) Per què es requereixen identificar tots els cicles? Què representa un cicle de longitud n?

(c) Per què en un cas ideal les dependències dels processos tenen forma d'arbre?

(d) Indiqueu quin problema, dels descrits en els materials, permet identificar els processos en els quals s'han d'implantar mecanismes de traçabilitat. Indiqueu el nombre màxim d'opcions que tindria que verificar un algorisme per trobar la solució (o solucions) òptimes al nostre problema. Si aquest algorisme és capaç de verificar 1000 opcions per segon, quin temps trigaria a obtenir la solució?

Solució:

(a) Es tracta d'un graf dirigit on els vèrtexs seran els processos i les arestes les relacions de dependència. En termes del graf, que existeixin més processos que dependències ens indica que no es tracta d'un graf connex i que existiran processos que no depenen de cap altre.

(b) Un cicle representa un problema crític ja que forma una relació de dependència tancada entre diversos processos. A la pràctica, un cicle d'ordre 3, significaria que el procés p_1 depèn de p_2 que depèn de p_3 que al seu torn torna a dependre de p_1, pel que mai acaba. Un cicle de qualsevol longitud suposa el mateix bloqueig, però trigant més a produir-se.

(c) Donat que un arbre és un graf acíclic, que les dependències tinguin forma d'arbre significa que els processos no tenen dependències cícliques entre si.

(d) Un problema dels descrits als materials que soluciona el problema plantejat és VERTEX_COVER. El problema VERTEX_COVER necessita comprovar tots els conjunts possibles per trobar el de menor cardinalitat. En el pitjor dels casos, necessita realitzar 2^n validacions, on n és el nombre de vèrtexs. Tenint en compte que pot realitzar 1000 validacions per segon, això suposa $\frac{2^{25000}}{1000}$ segons.

4. (Valoració d'un 15% = 5% + 5% + 5%)

Indiqueu (i justifiqueu) si les següents afirmacions són certes o falses. Cada apartat és independent.

(a) Si $A \in NP$ i B és NP-Complet, llavors $A \leq_p B$ i $B \leq_p A$.

(b) Siguin A, B i C tres problemes. Si $A \in P$, i sabem que $C \leq_p B$ i $B \leq_p A$, aleshores $C \in P$.

(c) Si A és un problema amb solucions que poden validar-se en temps polinomial, aleshores $A \leq_p 3SAT$.

Solució:

(a) Fals, tret que $P = NP$. Podem assegurar la primera afirmació ($A \leq_p B$) però no la segona ($B \leq_p A$) ja que A podria ser $P \subseteq NP$ i, llavors, si $P \neq NP$ no pot ser $B \leq_p A$.

(b) Cert. Si $C \leq_p B$ i $B \leq_p A$, aleshores per la propietat transitiva de la reducció polinòmica, podem afirmar que $C \leq_p A$. Com que $A \in P$, tenim que $C \in P$.

(c) Cert. El problema A pertany a NP i podrà reduir-se a $3SAT$ per ser $3SAT$ NP-Complet.

5. (Valoració d'un 30%=15%+15%)

El problema KNAPSACK, o problema de la motxilla, consisteix a optimitzar la solució d'un problema tenint en compte una restricció de cost.

Es diu problema de la motxilla ja que la seva descripció original establia la necessitat de maximitzar el valor d'un conjunt d'objectes a transportar en una motxilla.

En la descripció original es vol maximitzar el valor econòmic dels objectes transportats (el valor) sense excedir la capacitat de la motxilla (el cost).

Existeixen altres definicions del problema de la motxilla en les quals el valor pot referir-se, per exemple, a la quantitat d'informació que va en un fitxer i el cost al seu espai d'emmagatzematge en disc.

La forma més habitual de plantejar el problema de la motxilla és la coneguda com $0-1$ **Knapsack** que limita la quantitat de còpies de cada objecte que podem utilitzar a una o a cap.

(a) **Algorisme Voraç**

Encara que el problema d'optimització associat a KNAPSACK és NP-Difícil, es pot escriure un algorisme voraç que retorna una solució aproximada.

Donat un conjunt d'objectes, cadascun amb un cost i un valor, podem calcular el seu benefici com

$$benefici = \frac{valor}{cost}.$$

Sigui C un conjunt d'objectes ordenats pel seu benefici. Associem cada objecte i de C amb el camp COST que conté el cost de l'objecte. Computacionalment, podem accedir al cost de l'objecte utilitzant la instrucció i.COST que es considera 1 operació computacional. Sigui c la capacitat. El següent podria ser un algorisme voraç per obtenir una solució aproximada al problema.

```
funció V(C, c)
  inici
    ocupacio←0
    objectes← {}
    per i∈C
      si ocupacio + i.COST ≤ c
        aleshores
              ocupacio←ocupacio + i.COST
              objectes←objectes ∪ i
      fisi
    fiper
    retorn objectes
  fi
```

i. Indiqueu la complexitat d'aquest algorisme voraç.

ii. Trobeu una instància del problema (capacitat de la motxilla i llista d'objectes amb cost, valor i benefici) per a la que l'algorisme voraç proporcioni un resultat no òptim.

iii. Arran d'aquest exemple, per què creieu que l'algorisme funciona millor quan la capacitat és molt major que el cost mitjà dels objectes? Per què?

(b) **Instàncies de Knapsack**

Indiqueu, dels següents problemes, quins poden ser una instància de KNAPSACK. Si ho són indiqueu també què representaria el cost, el valor i la capacitat.

i. Cobertura de xarxa mòbil: Donat un pressupost màxim per antenes de telefonia mòbil, maximitzar la cobertura de xarxa mòbil.

ii. Emergències: Donada una xarxa de carreteres, tenim un graf associat en el qual cada ciutat és un node i les arestes representen les connexions per carretera. Volem assegurar que cada ciutat o una de les seves veïnes, té un centre d'atenció d'emergències.

iii. Selecció: Un repartidor vol maximitzar el benefici de la seva ruta de repartiment minimitzant el seu cost de transport. Per preparar la seva ruta disposa d'una llista de ciutats i el cost de viatjar entre cadascuna d'elles així com el benefici esperat a cada ciutat.

Solució:

(a) **Algorisme Voraç**

i. L'algorisme té una complexitat $O(n)$, sent n el nombre d'objectes. Veiem que recorre tots els objectes de la llista una única vegada ja que, segons l'enunciat, aquests ja venien ordenats per benefici.

ii. Amb capacitat total $c = 4$ i la seqüència C està formada pels següents elements:

- a: cost 3, valor $3,1$ i benefici $1,03$,
- b: cost 2, valor 2 i benefici 1,
- c: cost 2, valor 2 i benefici 1,

L'algorisme voraç inclouria únicament l'objecte a, amb un valor total de 3.1 i un cost de 3, quan pot veure's que la solució òptima consistiria en els objectes b i c amb un valor total de 4 i un cost de 4.

iii. L'algorisme voraç funciona millor quan c és molt més gran que el cost mitjà ja que l'error total que podem cometre és major quan deixem buits grans sense emplenar com en l'exemple anterior.

(b) **Instàncies de Knapsack**

i. Cobertura: pot ser una instància de KNAPSACK on la capacitat és el pressupost total, el valor és el nombre de clients als quals dóna servei una antena i el cost el preu d'instal·lar l'antena.

ii. Emergències: no és una instància de KNAPSACK sinó de VERTEX_COVER.

iii. Selecció: No és una instància de KNAPSACK sinó una variació del TSP anomenada TSP amb beneficis, en aquest cas pot semblar que té les característiques d'un KNAPSACK però, la diferència més important és que, cada vegada que anem a una ciutat varien tots els nostres costos de viatge pel que no podem tractar-ho com un problema de motxilla.

Recursos

Recursos Bàsics

- Mòdul didàctic 6. Complexitat computacional.
- Mòdul didàctic 7. Problemes intractables.
- Col·lecció de problemes

Recursos Complementaris

- PACs i exàmens de semestres anteriors.
- Programari per a l'estudi d'algorismes sobre grafs.
- Enllaç: Applets interactius sobre algorismes de grafs.

Criteris d'avaluació

- La PAC s'ha de resoldre **de forma individual**.
- És necessari justificar la resposta a cadascun dels apartats. Es valorarà tant la correctesa de la resposta com la justificació donada.
- En els apartats on calgui aplicar algun algorisme, es valorarà la tria de l'algorisme apropiat, els passos intermedis, el resultat final i les conclusions que se'n derivin.

Format i data de lliurament

Cal lliurar **un únic document** PDF amb les respostes a tots els exercicis. El nom del fitxer ha de ser: **PAC3_Cognom1Cognom2Nom.pdf**.

Aquest document s'ha de lliurar a l'espai de **Lliurament i Registre d'AC** de l'aula **abans de les 23:59 del dia 20/12/2018**. **No s'acceptaran lliuraments fora de termini.**