Intro

31 août 2021 13:10

⟵⟶ Dans ce cours

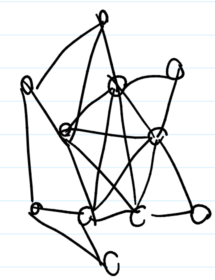
iFT436 - Algos et struct. données
Problèmes "faciles"
$O(n)$ $O(n \log n)$ $O(n^2)$

↓

iFT800: Algo
Problèmes "difficiles"
Pas d'algo en temps poly connu. (NP-complet)

iFT503/711

ex: clique max dans un graphe



ex: faire un horaire qui respecte des contraintes

ex: commis voyageur

Que faire?

Algos d'approx

- Algo rapide
- Pas tjrs optimal
- APP $\leq$ 3 · OPT (ex)

1re moitié

Algos faiblement exponentiels

- Complexité paramétrée
- $O(2^k \cdot n)$ où k est un paramètre petit

2e moitié

ex: max clique en $O(2^k \cdot n)$
où k = degré max du graphe

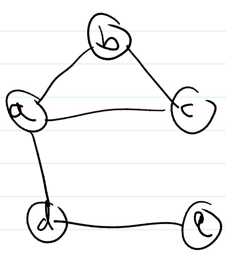
Cours théorique

- on veut des algo avec garanties démontrable
- applications ≠ algos vus en classe

- on veut des algo avec garanties démontrable
- applications ≠ algos vus en classe
  applications = TECHNIQUES vues
- "public cible" : designer d'algos
  ≠ utilisateurs d'algos

Notions préalables

- maths discrètes (MAT115)
  ex : ensembles, séquence, permut.
  $\cap \quad \cup \quad \setminus \quad \subseteq \quad \emptyset \quad |X| \quad \in$
  $$|\{X : X \subseteq S\}| = 2^{|S|}$$
- notation O : évaluer la complexité d'un algo
- techniques de preuve : directe, induction, contradiction
- graphes $G = (V, E)$
  V : sommets, E : arêtes



$V = \{a, b, c, d, e\}$

$E = \{ab, bc, ac, ad, de\}$

- Faites les exercices !

Exemple d'algo d'approx.

- Il y a n cours à donner en info.
  $U = \{u_1, u_2, \ldots, u_n\}$

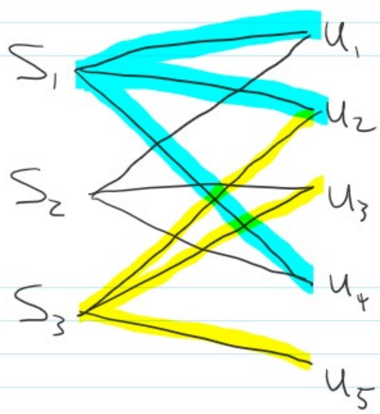
u_1 = machine learn.
u_2 = algo
...

- Il y a n cours à donner en info. u_2 = algo
  $$U = \{u_1, u_2, \dots, u_n\}$$
  ...
  u_n = maths discrète

- Pour chaque cours, il faut un prof capable de le donner.
- Chaque prof candidat a 3 expertises

  Ensemble des profs
  $$S = \{S_1, S_2, \dots, S_m\}$$
  où chaque S_i est un ens. de 3 elts de U

<u>But</u>: choisir un # min de profs tel que tous les cours sont couverts.



$S_1 = \{u_1, u_2, u_4\}$
$S_2 = \{u_1, u_3, u_4\}$
$S_3 = \{u_2, u_3, u_5\}$

Sol opt: $\{S_1, S_3\}$

3-SET-COVER

<u>Entrée</u>: ensemble $U = \{u_1, u_2, \dots, u_n\}$
ensemble $S = \{S_1, S_2, \dots, S_m\}$ où
$\forall S_i \in S, |S_i| = 3$ et $S_i \subseteq U$

<u>Sortie</u>: sous-ensemble $S^* \subseteq S$ de taille min. t.q.
$$\bigcup_{S_i \in S^*} S_i = U$$

$3SC(U = \{u_1, \dots, u_n\}, S = \{S_1, \dots, S_m\})$
$S^* = \{\}$
n tours pour $i = 1..n$

n tours

pour $i = 1..n$

$O(n)$ si u_i n'est pas couvert par S^*

$O(m)$ Soit $S' \in S$ t.q. $u_i \in S'$

si S' n'existe pas, return ∞

$O(1)$ S^*.insert(S')

return S^*

$$O(n \cdot (n + m)) = O(n^2 + nm)$$

Cet algo est une 3-approx

Soit OPT la valeur d'une sol. optimale.
Soit APP la valeur retournée par l'algo ($|S^*|$)

$$\boxed{APP \leq 3 \cdot OPT} \qquad \boxed{APP \leq \left(\max_{S' \in S} |S'|\right) \cdot OPT}$$

- Borne sur OPT

  Il y a n elts à couvrir

  Chaque S_i d'une solution couvre au plus 3 elts

  $\Rightarrow OPT \geq \lceil n/3 \rceil \geq n/3 \Rightarrow n \leq 3\,OPT$

- Borne sur APP

  Au pire, l'algo ajoute un S_i à S^* différent pour chaque elt de U. (sauf peut-être le premier S_i qui couvre 3 cours)

  $APP \leq n - 2 \leq n$

But: $APP \leq 3 \cdot OPT$

$APP \leq n \leq 3 \cdot OPT$