TD 4 : Matroïde Dread

26 Septembre 2024

Page web du cours : `https://people.irisa.fr/David.Baelde/algo/index.html`
Coordonnées des chargés de TD :
`malo.revel@ens-rennes.fr` `isseinie.calviac@irisa.fr`
Les TDs ont lieu les jeudis de 9h45 à 11h15, se fier à ADE pour la salle.

1 Correction de l'algorithme glouton générique

On donne ci dessous un algorithme glouton générique pour un problème de maximisation.

fonction glouton$(E, \mathcal{I}, w)$
  $S := \emptyset$
  trier les éléments de E par ordre décroissant de poids
  pour $e \in E$ dans l'ordre décroissant de poids
    si $S \cup \{e\} \in \mathcal{I}$ **alors**
      $S := S \cup \{e\}$
  renvoyer S

Question 1 – Montrer à l'aide d'un invariant que si $(E, \mathcal{I})$ est un matroïde, alors l'algorithme glouton générique renvoie bien une solution $S \in \mathcal{I}$ de poids total maximum.
(Indice : s'inspirer de la preuve de correction de l'algorithme de Kruskal vue en cours)

2 Correction de Kruskal avec des matroïdes

Dans cet exercice on va montrer que l'algorithme de Kruskal est correct en se ramenant à l'algorithme glouton générique précédent.

Pour ce faire, on prend pour E l'ensemble des arêtes du graphe non-orienté $G = (V, E)$ considéré, et pour $\mathcal{I}$ l'ensemble des forêts (sous-graphes acycliques de G), qu'on représente par des ensembles d'arêtes.

Question 2 – Justifier que $(E, \mathcal{I})$ est un système d'indépendance :
— $\emptyset \in \mathcal{I}$
— Pour tout $A \in \mathcal{I}$, si $B \subseteq A$ alors $B \in \mathcal{I}$

Nous allons montrer que $(E, \mathcal{I})$ respecte la propriété d'échange : si $A, B \in \mathcal{I}$ et $|A| < |B|$ alors il existe $e \in B \setminus A$ tel que $A \cup \{e\} \in \mathcal{I}$.

Montrons la contraposée. Soit $A, B \in \mathcal{I}$. Supposons que pour tout $e \in B \setminus A$, $A \cup \{e\} \notin \mathcal{I}$.

Définition 1. *Pour toute forêt $A \subseteq E$, on note $A^* \subseteq V^2$ la relation binaire "être dans le même arbre dans A". Autrement dit, $(u, v) \in A^*$ ssi il existe un chemin entre u et v dans A (c'est la clôture réflexive transitive de A vu comme une relation binaire sur les sommets).*

QUESTION 3 – Montrer que pour tout $\{u, v\} \in B$, $(u, v) \in A^*$.

Pour toute forêt F on note $C(F)$ l'ensemble des arbres (composantes connexes) de F. Ce sont aussi les classes d'équivalence de F^*.

QUESTION 4 – Montrer que $|C(A)| \leq |C(B)|$.

QUESTION 5 – Montrer que pour toute forêt $F \subseteq E$, $|C(F)| = |V| - |F|$.

QUESTION 6 – Conclure que $(E, \mathcal{I})$ respecte la propriété d'échange.

QUESTION 7 – (Bonus) Expliquer pourquoi l'algorithme de Kruskal est correct.

3 Ordonnancement de tâches

On considère ici un problème d'ordonnancement de tâches de durée 1 sur un unique processeur :

- entrée : un ensemble de n tâches, où la tâche numéro i est décrite par une deadline d_i et une pénalité w_i ;
- sortie : une permutation qui minimise la pénalité globale (on compte la pénalité w_i si la tâche i est faite après d_i), ou qui maximise les pénalités non attribués (i.e. la somme des pénalités des tâches à l'heures, qui finissent avant leurs deadlines)

Définition 2 (tâche à l'heure, tâche en retard)**.** *Une tâche est à l'heure quand elle est exécutée avant sa deadline. Elle est en retard sinon.*

Définition 3 (ordonnancement sous forme canonique)**.** *Un ordonnancement est sous forme canonique si on a d'abord les tâches à l'heure puis les tâches en retard, et que les tâches à l'heure sont triées par ordre de deadlines croissantes.*

QUESTION 8 – Montrer que tout ordonnancement peut être mis sous forme canonique.

Définition 4 (ensemble indépendant)**.** *Un ensemble de tâches A est indépendant s'il existe un ordonnancement dans lequel les tâches de A sont à l'heure.*

Définition 5. *On définit $(E, \mathcal{I})$ avec E un ensemble de tâches et $\mathcal{I}$ l'ensemble des ensembles indépendants de tâches.*

QUESTION 9 – Montrer que pour tout ensemble de tâches F, les affirmations suivantes sont équivalentes.

- L'ensemble F est indépendant.
- Pour $t = 1, 2, ..., n$, on a $N_t(F) \leq t$, où $N_t(F)$ est le nombre de tâches dans F dont la deadline est avant la date t.

— Si les tâches de F sont ordonnancées par ordre croissant de dates d'échéance, alors aucune tâche n'est en retard

QUESTION 10 – Montrer que $(E, \mathcal{I})$ est un matroïde.

QUESTION 11 – Y a-t-il un algorithme glouton correct pour ce problème d'ordonnancement ? Si oui, le donner.