Algorithmique 2

L3 RI

Table des matières

1 NP-Complétude

1.1 Définitions

Réduction Soient L_1, L_2 des langages. Une réduction polynomiale de L_1 à L_2 est une fonction f calculable en temps polynomial telle que :

$$f(x) \in L_2 \iff x \in L_1$$

On note $L_1 \leq L_2$ (L_1 plus facile que L_2 = Si on sait résoudre L_2, on sait résoudre L_1)

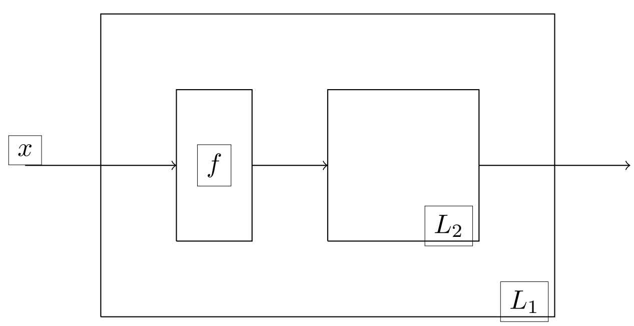

FIGURE 1 – Schéma de réduction de L_1 à L_2

Classe NP Soit L un langage. L est dit dans la classe NP s'il existe une machine de Turing non-déterministe en temps polynomial par rapport à la taille de l'entrée qui décide L.

NP-Complétude Soit L un langage. L est dit NP-dur si pour tout $L' \in \mathrm{NP}$, $L' \leq L$. L est dit NP-complet si $L \in \mathrm{NP}$ et L est NP-dur.

Remarques Si un problème NP-complet est dans P, alors P = NP. Pour montrer que L' est NP-dur, il suffit de montrer qu'il existe L NP-dur tel que $L \leq L'$.

1.2 Satisfiabilité d'une formule

Problème de décision : SAT

Entrée φ formule de la logique propositionnelle

Sortie Oui si φ est satisfiable, c'est-à-dire s'il existe une valuation v des variables qui rend φ vraie

Théorème de Cook SAT est NP-complet. (*Réduction de L à SAT pour $L \in \mathrm{NP}$ en considérant une machine de Turing $\mathcal{M}$ non-déterministe qui décide L. On construit une formule qui traduit une exécution de $\mathcal{M}$.*)

Problème de décision : i-SAT

Restriction de SAT à des formules en forme normale conjonctive (CNF) tel que chaque clause contient au plus i littéraux

Théorème 3-SAT est NP-complet. (*Réduction de SAT à 3-SAT.*)

Théorème 2-SAT est dans P. (*Réduction de 2-SAT à la détermination des CFC d'un graphe.*)

Problème de décision : MAX-2-SAT

Entrée φ formule en 2-forme normale conjonctive et $k \in \mathbb{N}$

Sortie Oui s'il existe une valuation qui satisfait au moins k clauses de φ

Théorème MAX-2-SAT est NP-complet. (*Réduction de 3-SAT à MAX-2-SAT.*)

1.3 Graphes

Problème de décision : ENS_INDEP

Entrée $G = (V, E)$ un graphe non-orienté et $k \in \mathbb{N}$

Sortie Oui s'il existe $V' \subseteq V$ tel que $|V'| = k$ et $(V' \times V') \cap E = \varnothing$

Théorème ENS_INDEP est NP-complet. (*Réduction de 3-SAT à ENS_INDEP.*)

Problème de décision : MAX_CUT

Entrée $G = (V, E)$ un graphe non-orienté et $k \in \mathbb{N}$

Sortie Oui s'il existe S_1 et S_2, $S = S_1 \uplus S_2$ et $\#\{(u, v) \in E \mid u \in S_1 \text{ et } v \in S_2\} \geq k$

Théorème MAX_CUT est NP-complet. (*Réduction de MAX-2-SAT à MAX_CUT.*)

2 Algorithmes d'approximation

3 Algorithmes probabilistes

4 Géométrie algorithmique

4.1 Enveloppe convexe

Calcul de l'enveloppe convexe

Entrée Un ensemble S de points (x, y) du plan
Sortie $C \subseteq S$ une énumération dans le sens trigonométrique des points extrémaux de l'enveloppe convexe de S

Position d'un point par rapport à une droite On suppose qu'il n'existe pas trois points alignés. p_2 est à gauche de la droite orientée $(p_0\vec{p}_1)$ si et seulement si :

$$\det(p_0\vec{p}_1, p_0\vec{p}_2) > 0$$

Marche de Jarvis Algorithme du paquet cadeau

- Partir du point de plus petite ordonnée p_0
- Prendre un point suivant, parcourir tous les points et le remplacer s'il en existe un plus à gauche de la droite orientée $(p_{actuel}\ \vec{p}_{suivant})$
- Recommencer jusqu'à retomber sur p_0

Complexité de la marche de Jarvis Si n est la taille de S et h le nombre de points dans l'enveloppe convexe : $O(nh)$

La complexité dépend de la sortie (*output sensitive*).

Balayage de Graham TODO

Complexité du balayage de Graham TODO

4.2 Points les plus rapprochés

Points les plus rapprochés

Entrée Un ensemble S de points (x, y) du plan
Sortie d^* la distance minimale entre deux points de S

Algorithme Principe de diviser pour régner, on divise le plan en deux. On obtient récursivement δ_D et δ_G. Soit $\delta = \min(\delta_G, \delta_D)$. On considère l'ensemble M des points dans la bande de largeur 2δ. $M = p_1 \dots p_k$ énumérés par ordonnées croissantes.

Lemme S'il existe $i < j$ tel que $d(p_i, p_j) < \delta$ alors il existe $i' < j' \leq i' + 8$ tel que $d(p_{i'}, p_{j'}) < \delta$.

En effet, on considère un rectangle qu'on divise en huit parties. On peut alors prendre 9 points et appliquer le principe des tiroirs.

Complexité Pré-tris en $O(n \log n)$. Au total, avec le *Master Theorem* : $O(n \log n)$

5 Algorithmes distribués