École polytechnique | Majeure informatique
Promotion 2008

Cours *Conception et analyse d'algorithmes*
TD 9 – Révisions
24 novembre 2010

1. Arbre d'évolution optimal pour la distance d'édition.

On dispose d'un certain nombre de séquences génétiques et on souhaite expliquer l'évolution de ces séquences par un arbre tel que deux séquences voisines dans l'arbre sont à faible distance. Pour cela on considère la distance de Hamming d_H entre mots.

Un arbre d'évolution pour les séquences $f_1, f_2, \ldots, f_m$ est un arbre dont les sommets sont $f_1, f_2, \ldots, f_m$, son coût est la somme des $m-1$ distances entre les séquences reliées par une arête

i. Donner un algorithme en temps polynomial qui résout le calcul de l'arbre d'évolution de coût minimal.

On complique un peu le problème en supposant donnés deux ensembles de séquences $F = \{f_1, f_2, \ldots, f_m\}$ et $G = \{g_1, g_2, \ldots, g_n\}$ on veut expliquer l'évolution de tous les éléments de F en s'aidant d'une partie de G. Pour cela on cherche un sous-ensemble G' de G et un arbre d'évolution dont l'ensemble des sommets est $F \cup G'$ dont le coût est minimal.

On se propose de montrer que le problème de décision associé est alors $\mathcal{NP}$-complet. Pour cela on réduit RECOUVREMENT-PAR-SOMMETS à notre problème. Soit $\Gamma = (X, E)$ un graphe connexe ayant n sommets $x_1, x_2, \ldots x_n$ et m arêtes. On associe à Γ un ensemble F et un ensemble G de mots de longueur n sur l'alphabet $\{0, 1\}$ définis somme suit. On note $w_\emptyset$ le mot fait de n 0, on note w_i pour $1 \leq i \leq n$ le mot fait de $n-1$ 0 et un 1 en position i et on note $w_{i,j}$ pour $1 \leq i < j \leq n$ le mot fait de $n-2$ 0 et deux 1 en positions i et j. On définit alors l'ensemble $G = \{w_1, w_2, \ldots, w_n\}$ dont les éléments correspondent aux sommets de Γ et l'ensemble $F = \{w_\emptyset\} \cup \{w_{i,j} / \ 0 \leq i < j \leq n \text{ et } \{x_i, x_j\} \in E\}$ dont les éléments différents de $w_\emptyset$ correspondent aux arêtes de Γ.

ii. Utiliser cette construction pour montrer que la recherche d'un arbre d'évolution conditionné de coût inférieur à k est un problème $\mathcal{NP}$-complet.

2. Programmation dynamique : Alignement de séquences

On considère des séquences sur un alphabet Σ. On se donne une fonction $\delta : (\Sigma \cup \{-\})^2 \mapsto \mathbb{N}$ qui représente le *score d'alignement* entre lettres. On pourra supposer que $\delta(x, x) > 0$ pour tout x appartenant à Σ, $\delta(x, y) = \delta(y, x) < 0$ pour $x \neq y$ appartenant à $\Sigma \cup \{-\}$ et $\delta(-, -) < 0$. Dans le cours sur la programmation dynamique, on a vu qu'on pouvait obtenir le meilleur alignement pour deux séquences en espace linéaire. La technique utilisée mixait programmation dynamique et approche "diviser pour régner". On étudie ici des variantes de ce problème.

i. Proposer un algorithme en $O(n^3)$ qui étant donné trois séquences u, v, w de longueur n, détermine l'alignement réalisant le meilleur score.

ii. Proposer un algorithme en $O(n^2)$ qui étant donné deux séquences u et v de longueur n, détermine les facteurs u' et v' de u et v qui réalisent un score d'alignement maximum. On rappelle qu'un mot w' est *facteur* d'un mot w si w peut s'écrire sous la forme $w = w_1 w' w_2$.

3. Placement optimal de ressources

L'objet de ce problème est d'étudier un problème de placement optimal de ressources : on souhaite placer k supermarchés dans n villes de façon à minimser la distance maximale des villes au supermarché le plus proche.

Soit $G = (X, E)$ un graphe non orienté. Un *ensemble dominant* de G est un sous-ensemble $S \subset X$ tel que tout sommet de $X \backslash S$ est voisin d'un sommet de S.

i. Donner une version décision du problème DOMINANT suivant : étant donné un graphe G, trouver la cardinalité minimale d'un ensemble dominant de G.

ii. Montrer que la version décision de DOMINANT est NP-complet (on pourra utiliser une réduction à partir de RECOUVREMENT D'ENSEMBLES).

Pour toutes villes x et y, la distance entre ces deux villes est notée $d(x, y)$. On suppose que la fonction d satisfait l'inégalité triangulaire : ie. $d(x, y) \leq d(x, z) + d(z, y)$ pour tout triplet (x, y, z) de villes. Le problème k-CENTRES consiste à trouver, dans un graphe complet à n sommets dont les arêtes sont valuées par une fonction d satisfaisant l'inégalité triangulaire, un sous-ensemble S de k sommets minimisant la quantité :

$$\phi(S) = \max_{y \notin S} \min_{x \in S} d(x, y).$$

On note $\phi^\star$ le minimum $\min_{S \subset V} \phi(S)$.

Étant donné un graphe non-orienté $G = (X, E)$, on considère le graphe complet K_X sur X et une valuation des arêtes définie par $d(x, y) = 1$ si $(x, y) \in E$ et $d(x, y) = 2$ sinon.

iii. Utiliser cette construction et le problème DOMINANT pour montrer que la version du problème k-CENTRES est NP-complète.

iv. EN déduire que s'il existe un algorithme polynomial qui trouve, étant donné un entier k et une valuation d du graphe complet à n sommets, un ensemble S de taille k tel que $\phi(S) < 2\phi^\star$ alors `P = NP`.