TP11 – Rien ne sert de courir

Projet de programmation M1

15 Décembre 2015

On rappelle qu'en mathématiques, un graphe $G = (V, E)$ est la donnée d'un ensemble (qu'on supposera fini) de sommets V et d'un sous-ensemble $E \subseteq V \times V$ d'arêtes. Si pour tout $(u, v) \in E \Rightarrow (v, u) \in E$, on dit que le graphe est non-orienté. Sinon il est orienté. Un graphe, c'est avant tout un dessin :

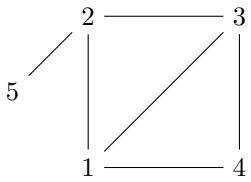

FIGURE 1 – Un graphe non orienté

Un graphe permet de modéliser de nombreux problèmes : dès que vous avez un ensemble d'objet avec des relations entre ces objets, le graphe est en général une bonne façon de modéliser le problème. On peut ajouter des informations sur un graphe. Le plus courant est d'ajouter des poids sur chaque arête, c'est-à-dire une fonction $w : E \to \mathbb{Q}$.

Vous avez vu dans le cours qu'il y avait essentiellement deux façons de représenter un graphe en machine :

— par liste d'adjacence : on a un tableau t de taille V et $t[i]$ est la liste des voisins du sommet i, avec éventuellement le poids de l'arête.

— par matrice d'adjacence : on a une matrice M de taille $V \times V$. $M[i, j]$ vaut 1 si $(i, j) \in E$ et 0 sinon. Si le graphe est pondéré, on peut choisir $M[i, j] = w(i, j)$. On a en général $w(i, j) = +\infty$ si i et j ne sont pas voisins.

Vous avez vu en cours une méthode de calcul du plus court chemin entre deux points, l'algorithme de Djikstra. On va voir ici un autre algorithme : Floyd-Warshall qui permet de trouver *tous* les plus court chemin de G en temps $O(|V|^3)$.

Exercice 1. Dans la suite G est un graphe pondéré, sa matrice d'adjacence est $M \in \mathbb{Q}^+_{n,n}$. On suppose que $M[i, i] = 0$ et que si $(u, v) \notin E$ alors $M[u, v] = +\infty$

On définit une opération $\odot$ sur $\mathbb{Q}^+_{n,n}$: $(A \odot B)$ est la matrice $n \times n$ telle que $(A \odot B)[i, j] = \min_{k=1}^{n}(A[i, k] + B[k, j])$. On note $A^{\odot n}$ la puissance $n$ième de A pour cette opération c'est-à-dire $A \odot \ldots \odot A$ itéré n fois.

1. Prouver que $(M \odot M)[i, j]$ est la longueur du plus court chemin de taille 2 de i à j. Qu'en est-il de $M^{\odot k}$ pour $k \in \mathbb{N}$? Et si $k = |V|$?
2. En remarquant que $A \odot B$ peut être interprété comme une multiplication matricielle dans le semi-anneau $(\mathbb{Q}^+, \min, +)$, donner un algorithme en temps $O(|V|^3 \log(|V|))$ qui calcule le plus court chemin de u à v pour tout $u, v \in V$.
3. Implémentez cet algorithme en C.
4. Que se passe-t-il s'il y a des poids négatifs dans le graphe ?
5. On peut en fait améliorer cet algorithme en le voyant comme une programmation dynamique. Pour $u, v \in V$ et $k \in \mathbb{N}$, on note $\mathsf{SP}(u, v, k)$ la longueur du plus court chemin de u à v de longueur k. On a $\mathsf{SP}(u, v, k) = +\infty$ s'il n'y a aucun chemin de longueur k reliant u et v.
   (a) Que vaut $\mathsf{SP}(u, v, 0)$?

(b) Que vaut $\mathsf{SP}(u, v, k+1)$ en fonction des valeurs $\mathsf{SP}(u', v', k)$?

(c) En déduire un algorithme pour trouver la longueur des plus court chemin entre chaque sommet du graphe en temps $O(|V|^3)$. Implémentez-le en C.

(d) Comment adapter cet algorithme pour reconstruire les plus courts chemins ? Implémentez le en C.