Plus courts chemins: algorithme de Floyd-Warshall

Auteur: Nicolas Gast

Calculer le plus court chemin dans un graphe intervient dans les protocoles de communication (pour le routage par exemple, avec des protocoles comme RIP ou BGP), pour calculer des trajets optimaux (dans un GPS par exemple),... De nombreux algorithmes ont été inventés dans les années 50 et utilisent chacun des mécanismes différents. On peut citer notamment l'algorithme de Floyd-Warshall (1959), l'algorithme de Dijkstra (1959) ou l'algorithme de Ford-Bellman.
Tous ces algorithmes sont quadratiques ou cubiques en le nombre de sommets du graphe, ce qui est souvent trop en pratique. On utilise alors des heuristiques, comme l'algorithme A (Peter Hart, Nils Nilsson and Bertram Raphael, 1968).*

L'algorithme de Floyd Warshall permet de calculer les plus courts chemins dans un graphe (X, A). Formulé autrement, pour tout couple de sommets $(i, j) \in S^2$, il détermine s'il existe un chemin de i à j, calcule sa longueur et construit une table de routage. Il prend en entrée la matrice d'adjacence du graphe, c'est-à-dire une matrice D_{ij} de taille $n \times n$, où $n = |X|$, telle que $D_{ij} < \infty$ si et seulement si $(i, j) \in A$. D_{ij} est alors le poids de l'arrête (i, j). Il produit en sortie deux matrices W et R, où W_{ij} est la longueur du plus court chemin entre i et j (s'il existe, $D_{ij} = +\infty$ sinon) et R_{ij} est une table de routage.

Exercice 1: Algorithme de Warshall

Dans un premier temps, on s'intéresse à calculer la clôture transitive, c'est à dire déterminer s'il existe un chemin allant de i à j pour tout couple de sommets $(i, j) \in X^2$. Les chemins de G peuvent être définis par une relation de récurrence sur les sommets intermédiaires par lesquels passent les chemins de G. Cette relation de récurrence peut-être décrite ainsi : pour un ensemble de sommets intermédiaires $T \subset X$ et un autre sommet $t \notin T$, il existe un chemin d'un sommet $i \in X$ à un sommet $j \in S$ ne passant que par des sommets de $T \cup \{t\}$ si et seulement si :

(a) il existe un chemin de i à j ne passant que par des sommets de T ;
ou
(b) il existe deux chemins, respectivement de i à t et de t à j, ne passant que par des sommets de T.

1. Écrivez de manière formelle cette relation de récurrence.
2. Quelle est la complexité d'une procédure qui implémente la relation de récurrence précédente de manière récursive et sans chercher à éliminer les appels redondants ?
3. Donnez un exemple de graphe et une séquence d'appels de la procédure récursive précédente sur ce graphe contenant des appels redondants.
4. La relation de récurrence précédente se prête particulièrement bien à l'expression d'une implémentation par programmation dynamique. Écrivez cette solution (attention à la matrice de routage).
5. Expliquez comment il est possible de déterminer l'ensemble des composantes fortement connexes de G à partir de W.

Exercice 2: Algorithme de Floyd-Warshall

On veut maintenant calculer la longueur des plus courts chemins sur un graphe pondéré.

1. Adapter l'algorithme de l'exercice précédent pour qu'il calcule la longueur des plus courts chemins et une matrice de routage.
2. Calculer la complexité de l'algorithme.
3. a) Que se passe-t-il si le graphe est orienté ?
   b) Et si dans un graphe orienté, certains arcs ont des poids négatifs ?