Voyageur de commerce

Épreuve pratique d'algorithmique et de programmation

Juillet 2002

Ce problème propose d'étudier quelques heuristiques pour la résolution du problème du voyageur de commerce. On se donne un ensemble de *villes*, représentées par des points du plan euclidien. La distance entre les villes est la distance euclidienne. La longueur d'un parcours de ville en ville est la somme des distances entre les villes. Un *voyageur de commerce* doit partir d'un ville de départ, visiter chacune de ces villes, une fois et une seule, et finalement revenir à sa ville de départ. L'objectif est de trouver la permutation des villes qui engendre le parcours de longueur minimale. Cette permutation est appelé *parcours ou chemin minimal*.

Ce problème est bien connu dans la littérature pour être de difficulté *non-polynomiale* : On ne connaît aujourd'hui aucun algorithme permettant de déterminer, en un temps polynomial dans le nombre de villes, le parcours de longueur minimale. (La découverte d'un tel algorithme serait un événement scientifique de première grandeur, avec un impact majeur sur le domaine !)

On s'intéresse ici à l'expérimentation de quelques approches *heuristiques* qui permettent de trouver une "bonne" solution au problème en un temps polynomial.

On considère donc à un ensemble de de points du plan, les *villes*, de cardinal N, avec N grand. On choisit ici $N = 20$. Les points sont définis à partir de la suite récurrente d'entiers suivante :
– u_0 est égal au nombre inscrit sur votre table d'examen ;
– $u_i = f(u_{i-1})$ pour $1 \leq i < 2.N$.
On pose $v_i = u_i$ modulo 100, $0 \leq i < 2.N$.

Pour $0 \leq n < N$, le point P_n a pour abscisse $x_n = v_{2i}$ et $y_n = v_{2i+1}$.

La fonction $f(u)$ est définie comme suit :

$$f(u) = u^2 \text{ modulo } 3953$$

(NB : Les numéros de table indiqués étaient 3, 4, 5, etc.)

Partie I. Préliminaires

Un *chemin* entre les n villes $P_0, \ldots, P_{n-1}$ est un parcours $(P_{i_0}, P_{i_2}, \ldots, P_{i_{n-1}})$ du voyageur de commerce parmi ces villes qui passe par chaque ville une fois et une seule et qui revient à son point de départ. La longueur du chemin est la somme des distances $(P_{i_k}, P_{i_{k+1}})$ pour $k = 0, \ldots, n-1$ avec $i_n = i_0$.

Un chemin entre les villes $P_0, \ldots, P_{n-1}$ est donc défini par une permutation de $(0, \ldots, n-1)$.

Question I.1. Quelles sont les coordonnées des villes P_0, P_5, P_{10} ?

Question I.2. Créez un tableau bidimensionnel de taille $N \times N$ contenant les distances euclidiennes entre les N villes. Quelle est la longueur du chemin *fermé* $(0, \ldots, N-1)$ passant successivement par chacune de villes et revenant à son point de départ ?

Question I.3. Combien y a-t-il de chemins entre les $n = 9$ premières villes $P_0, \ldots, P_{n-1}$? Décrivez explicitement le chemin de longueur minimale et donnez sa longueur.

Question I.4. Pour quelle valeur de n le calcul de la longueur de tous les chemins possibles entre les villes $P_0, \ldots, P_{n-1}$ par leur énumération exhaustive prend-il plus de une minute sur votre machine ? Donnez une estimation du temps qui serait nécessaire pour $n = 20$ dans une unité adaptée : jours, mois, années, etc.

Partie II. Une méthode gloutonne par insertion

Une première méthode pour trouver rapidement une solution approchée est la suivante. On part du chemin entre les villes 0 et 1. On cherche à insérer la ville 2 dans ce chemin de manière à minimiser la longueur. Si plusieurs positions sont possibles, on choisi la dernière dans une exploration gauche-droite de la liste courante des villes. On procède ainsi successivement pour chaque ville P_i, $i = 2, \ldots, N-1$.

Question II.1. Décrivez brièvement la mise en oeuvre de cette idée. Précisez en particulier les structures de données utilisées. Quel est l'ordre de grandeur du coût en temps de votre approche ?

Question II.2. Considérez le chemin $(P_0, \ldots, P_{n-1})$ de longueur $n = 9$. Quel est la meilleure insertion pour la ville P_n dans ce chemin ? Décrire le chemin obtenu et donner sa longueur.

Question II.3. Appliquez la méthode à la liste ordonnée des villes $(P_0, \ldots, P_{n-1})$ de taille $n = 9$. Donnez la suite des insertions réalisées. Décrire le chemin obtenu et donner sa longueur.

Question II.4. Pouvez-vous donner une valeur de $n > 2$ pour laquelle le chemin ainsi obtenu par cette méthode est de longueur minimale parmi tous les chemins possibles ? Pouvez-vous en donner une autre pour laquelle il ne l'est pas ? Décrivez dans chaque cas le chemin obtenu et sa longueur.

Question II.5. Quelle est la longueur du chemin obtenu par cette méthode pour les villes $P_0, \ldots, P_{N-1}$? Décrivez le chemin obtenu et donner sa longueur. Combien de temps votre programme met-il pour le trouver ?

Partie III. Une méthode itérative par optimisation locale

On considère un parcours orienté du chemin : $(P_{i_0}, P_{i_2}, \ldots, P_{i_{N-1}})$. Des villes *voisines* au rang k sur ce chemin sont deux villes de la forme $(P_{i_k}, P_{i_{k+1}})$, $k = 0, \ldots, N-1$ et $i_N = i_0$.

Considérons deux couples de villes voisines sur un chemin orienté $(P_{i_0}, \ldots, P_{i_n})$, par exemple les villes aux rangs $(k, k+1)$ et $(l, l+1)$, avec $0 \leq k < k+1 < l < l+1 \leq N$ et la convention que la ville de rang n est celle de rang 0. Soient $(P_{i_k}, P_{i_{k+1}})$ et $(P_{i_l}, P_{i_{l+1}})$ ces villes. On dit que l'on *échange* les deux segments de rangs k et l si l'on transforme ce chemin en allant de P_{i_k} à P_{i_l} et de $P_{i_{k+1}}$ à

$P_{i_{l+1}}$, et en renversant le sens du parcours initial entre P_{i_l} et $P_{i_{k+1}}$. (Il est vivement recommandé de faire un dessin !).

On appelle *signature* de cet échange les noms des villes échangées, c'est-à-dire les deux couples (i_k, i_{k+1}) et (i_l, i_{l+1}).

Question III.1. Quelle est la longueur du nouveau chemin ainsi obtenu en fonction de celle du chemin initial ? Montrer que si les deux segments initiaux se croisaient (au sens strict, c'est-à-dire que leur intersection est un point unique, différent de leurs extrémités), alors leur échange conduit à un chemin strictement plus court. En déduire qu'un chemin de longueur minimal ne peut avoir de tels croisements.

Un échange qui fait décroître la longueur du chemin est dit *intéressant*.

L'idée est donc de partir d'un chemin et d'appliquer successivement des échanges intéressants jusqu'à ne plus en trouver.

Question III.2. Montrez sur un petit exemple que le chemin résultant dépend en général de l'ordre des échanges. Est-il de longueur minimale parmi tous les chemins possibles ?

Question III.3. Décrivez brièvement une procédure qui prend en paramètre un chemin et qui cherche l'échange intéressant qui fait décroître la longueur de la plus grande quantité. Si plusieurs tels échanges sont possibles, choisissez celui qui se trouve le plus à droite : l'échange des segments de rangs i et j tels que i est maximal, et, pour i fixé, celui où j est maximal. Précisez les structures de données que vous utilisez.

Question III.4. Considérez le chemin $(P_0, \ldots, P_{n-1})$ de longueur $n = 9$. Comporte-t-il un échange intéressant ? Si oui, quel est la signature de celui qui est sélectionné dans la procédure ci-dessus ? Décrivez le nouveau chemin ainsi construit et sa longueur.

Question III.5. Appliquez répétitivement votre procédure d'échange au chemin $(P_0, \ldots, P_{n-1})$ de longueur $n = 9$ jusqu'à ce qu'il n'y ait plus aucun échange intéressant possible. Combien avez-vous fait d'échanges ? Donnez la liste de leurs signatures. Décrivez le chemin final obtenu et sa longueur.

Question III.6. Pouvez-vous donner une valeur de n pour laquelle le chemin ainsi obtenu est minimal ? Pouvez-vous en donner une autre pour laquelle il ne l'est pas ? Décrivez dans chaque cas le chemin obtenu et sa longueur.

Question III.7. Quelle est la longueur du chemin obtenu par cette méthode pour les villes $P_0, \ldots, P_{N-1}$? Donnez la suite des signatures des échanges effectués. Combien de temps votre programme met-il pour le trouver ?

Partie IV. Méthodes hybrides

Vous avez exploré ci-dessus plusieurs méthodes d'approximation fondées sur des principes complètement différents. On peut bien sûr imaginer d'*hybrider* ces méthodes entre elles, par exemple en appliquant une méthode A au chemin trivial, puis en appliquant la méthode B au chemin obtenu, etc.

Question IV.1. Proposez une hybridation des deux méthodes ci-dessus qui vous semble intéressante et justifiez soigneusement votre proposition. Appliquez-la à l'ensemble des villes $(P_0, \ldots, P_{N-1})$. Discutez des résultats. Que se passerait-il pour N encore plus grand ?