Algorithmes pour les graphes - TD sur plateforme
Exercice 6 : Ajout d'une heuristique d'ordre

Une autre façon d'améliorer les performances d'un algorithme procédant par *Branch and Bound*, consiste à choisir l'ordre dans lequel les sommets sont énumérés. L'objectif est de trouver le plus rapidement possible le circuit le plus court afin de pouvoir couper plus rapidement les autres branches. La règle utilisée pour choisir l'ordre des sommets est appelée une *heuristique d'ordre*. Pour le voyageur de commerce, une heuristique d'ordre qui donne généralement d'assez bons résultats consiste à visiter en premier les sommets les plus proches du dernier sommet visité (`vus[nbVus-1]`). Ainsi, à chaque appel récursif, il s'agit de trier le tableau `nonVus[0..nbNonVus-1]` de telle sorte que, pour tout sommet `i` $\in$ `[1,nbNonVus-1]` :
`cout[vus[nbVus-1]][nonVus[i-1]]` $\leq$ `cout[vus[nbVus-1]][nonVus[i]]`.

Votre travail : Vous devez implémenter la procédure `permut` :

```
int permut(int vus[], int nbVus, int nonVus[], int nbNonVus, int longueur, int pcc)
```

telle que :
- le tableau `vus[0..nbVus-1]` contient les sommets de la liste vus,
- le tableau `nonVus[0..nbNonVus-1]` contient les sommets de l'ensemble $nonVus$,
- la variable `longueur` contient la longueur du chemin correspondant à `vus[0..nbVus-1]`,
- la variable `pcc` contient la longueur du plus court circuit trouvé depuis le début.

La postrelation de cette fonction est la même que pour l'exercice 5, et vous utiliserez la même fonction d'évaluation que pour l'exercice 5, mais vous ajouterez l'heuristique d'ordre pour énumérer les sommets de `nonVus` en commençant par ceux qui sont le plus proches de `vus[nbVus-1]`.

Code fourni (téléchargeable sur http ://liris.cnrs.fr/csolnon/TPTSP/code3.c) : cf code fourni pour l'exercice 3.

Exemple d'exécution : Les temps CPU (en secondes) sont donnés à titre indicatif, pour un processeur 2,6 GHz Intel Core i5.

Entrée	Sortie	Temps CPU
4	72	0.00
6	91	0.00
8	123	0.00
10	134	0.00
12	161	0.00
14	182	0.00
16	198	0.00
18	230	0.00
20	261	0.06
22	281	0.05
24	299	0.28
26	313	3.33
28	326	4.61
30	349	9.79
32	361	13.21