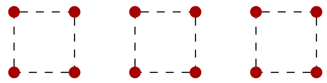
TD13 - Algorithmes d'approximation

Exercice 1

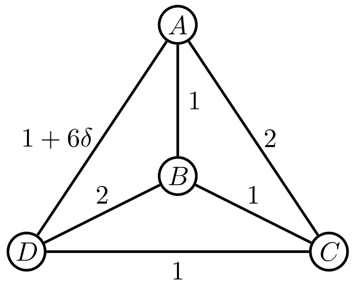
Pour tout entier naturel k non nul, on considère un ensemble V de $4k$ points formant les sommets de k carrés identiques alignés et disjoints, de coté 1, distants de 1.



Exemple avec $k = 3$ carrés et $4k = 12$ points.

Question 1.
□ **1.1.** Construire des tournées sur V en suivant l'*algorithme du point le plus proche* : on part d'un point quelconque de V. À chaque étape, on ajoute au chemin courant le point libre le plus proche du dernier point atteint. Une fois le dernier point sélectionné, on revient au point initial.
□ **1.2.** Quelle est la nature de cet algorithme ?
□ **1.3.** Quelle est sa complexité temporelle ?
□ **1.4.** La construction d'une tournée aléatoire serait-elle plus efficace ? Serait-elle optimale en temps ? en longueur ?

Question 2. On considère l'ensemble des points $V = \{A, B, C, D\}$ et les distances entre ces points sont données sur le schéma suivant. δ est un nombre réel positif.



Question 3. Partant du point A, quelle longueur de tournée, notée $\textsc{Greedy}(V)$, un algorithme glouton renvoie-t-il ?

Question 4. Quelle est celle d'une tournée optimale, notée $\textsc{Opt}(V)$?

Question 5. Pourquoi l'algorithme glouton ici mis en œuvre n'est pas une α-approximation ?

Exercice 2

Le problème du voyageur de commerce (TSP) est non seulement difficile à résoudre mais il est également difficile à approximer. Aucun algorithme polynomial ne peut l'approximer avec un facteur constant, sauf si les classes P et NP sont égales. Pour établir ce résultat, on procède par *réduction*. Le problème réduit est celui du cycle hamlitonien $\textsc{HamCycle}$ qui consiste à déterminer si un graphe donné possède un cycle ne passant qu'une seule fois par chacun de ses sommets.

Soit $H = (V_H, E_H)$ un graphe à n sommets et d_H la distance définie par :

$$d_H(u, v) = \begin{cases} 1 & \text{si } \{u, v\} \in E_H \\ n^2 & \text{sinon.} \end{cases}$$

La paire (V_H, d_H) définit une *instance* particulière de TSP.

Question 1. Soit A une α-approximation de TSP où α est une constante. Montrer que $A(V_H, d_H) < n^2$ si et seulement H possède un cycle hamiltonien.

Question 2. Si une telle approximation de TSP existe, proposer un algorithme qui permet de résoudre $\textsc{HamCycle}$.

Question 3. Conclure sachant que $\textsc{HamCycle}$ est NP-complet.

Exercice 3

L'*algorithme de Christofides* est une 1.5-approximation de TSP et constitue, à ce jour, l'un des meilleurs algorithmes d'approximation pour le TSP métrique. Il utilise la notion de *couplage parfait* de poids minimum. Il s'agit d'apparier les sommets d'un graphe pondéré (G, w) par des arêtes indépendantes de G, deux arêtes ne pouvant avoir d'extrémité commune. Un *couplage parfait* est une forêt couvrante F où chaque composante est composée d'une seule arête.

Question 1. Quelle condition simple sur son nombre de sommets un graphe doit-il vérifier pour pouvoir apparier tous ses sommets par des arêtes indépendantes ? Proposer un exemple où même si cette condition est satisfaite, G ne possède pas de couplage parfait.

Question 2. Parmi tous les couplages F de (G, w), on cherche celui de poids $w(F)$ minimum. S'il en existe un, un couplage parfait minimum peut être calculé à l'aide d'un algorithme déterministe de complexité $O(n^3)$ où n désigne la taille du graphe. On propose l'*algorithme d'approximation* dit de Christofides ci-dessous.

Algorithme 1 : Algorithme ChristofidesApprox

Entrée : une instance (V, d) de TSP
Sortie : une tournée, *ie* un ordre sur les points de V

calculer un arbre couvrant de poids minimum T sur le graphe complet défini par V et les arêtes valuées par d
calculer l'ensemble I des sommets T de degré impair
calculer le couplage parfait de poids minimum F pour le graphe induit par I
la tournée est définie par le circuit eulérien du multigraphe $T \cup F$ dans lequel on ignore les sommets déjà visités
renvoyer *la tournée*

□ **2.1.** Justifier la validité de l'algorithme et que sa complexité est polynomiale.
□ **2.2.** Montrer que son facteur d'approximation est $1/2$.

Exercice 4

On considère le problème d'optimisation SubsetSum suivant.

- **Entrée :** un tableau $T = [t_1, \dots, t_n]$ d'entiers naturels et $C \in \mathbb{N}$.
- **Sortie :** un sous ensemble $I \subset [\![1, n]\!]$ tel que $\sum_{i \in I} t_i \leq C$.
- **Optimisation :** maximiser $\sum_{i \in I} t_i$.

La version décisionnelle de ce problème est NP-complète.

Question 1. On propose d'approcher une solution à SubsetSum à l'aide de l'algorithme glouton suivant.

$S \leftarrow 0$
$I \leftarrow \varnothing$
pour i *allant de* 1 *à* n **faire**
  si $S + t_i \leqslant C$ **alors**
    $S \leftarrow S + t_i$
    $I \leftarrow I \cup \{i\}$
renvoyer I

Montrer que cet algorithme n'est pas un algorithme d'approximation à facteur constant pour SubsetSum.

Question 2. On modifie l'algorithme précédent de sorte à considérer les t_i par ordre décroissant de valeur plutôt que par ordre croissant d'indice. Cela définit un algorithme noté A.
□ **2.1.** Déterminer la complexité temporelle de A.
□ **2.2.** Montrer que A est une $1/2$-approximation de SubsetSum.
□ **2.3.** Le facteur d'approximation de A, au moins égal à $1/2$, est-il le meilleur pour cet algorithme ?