Objectif

- Algorithmes approchés

Exercice 1 — [Knapsack : résultat d'inapproximabilité]

Considérons le problème d'optimisation suivant, dit **KNAPSACK** :

KNAPSACK

- INSTANCE : Deux entiers n et B ; Deux vecteurs d'entiers $u = (u_1, \ldots, u_n)$ et $w = (w_1, \ldots, w_n)$.
- BUT : Minimiser la fonction $\sum_{j=1}^{n} w_j x_j$ telle que :

$$\sum_{j=1}^{n} u_j x_j \leq B,$$
$$x \in \{0,1\}^n.$$

Soit p un entier fixé.

1. Démontrer que si $P \neq NP$ alors, pour tout entier naturel p, il ne peut exister un algorithme p-approché A tel que :

$$|A(I) - \texttt{Opt}(I)| \leq p, \ \forall I \in \texttt{Inst}(\textbf{KNAPSACK}).$$

Exercice 2 — [Problème Bin-Packing]

Soit n un entier non nul. Considérons n couples $(a_1, t_1), \ldots, (a_n, t_n)$ tels que, pour tout indice j appartenant à $\{1, \ldots, n\}$, a_j indique le nom de l'article j et t_j indique sa taille. Nous supposerons, sans perte de généralité, que toutes les tailles appartiennent à l'intervalle $]0, 1]$. Enfin, nous disposons d'un nombre illimité de conteneurs (*bins en anglais*) tous de taille 1.

Le problème du BIN-PACKING est un problème d'optimisation dans lequel nous cherchons à déterminer le nombre *minimum* de conteneurs nécessaires pour empaqueter (*packing en anglais*) les n articles.

Partie A : Étude de la complexité

Dans cette partie nous vous proposons d'étudier la complexité du problème BIN-PACKING. Pour cela nous considérerons le problème de partition d'un ensemble, nommé SET-PARTITION, suivant :

SET-PARTITION

- INSTANCE : Un sous-ensemble E de $\mathbb{N}^*$;
- QUESTION : Existe-il un sous-ensemble X de E tel que :

$$\sum_{e \in X} e = \sum_{e \in E \setminus X} e.$$

1. Démontrer que le problème SET-PARTITION est **NP**-Complet. Vous pouvez utiliser une réduction du problème SUBSET-SUM.
2. Préciser la version *décision* du problème BIN-PACKING. Nous la noterons BIN-PACKING-DEC
3. Démontrer que le problème BIN-PACKING-DEC est **NP**-Complet.
4. Quelle est la classe de complexité du problème BIN-PACKING.

Partie B : Étude d'une heuristique

Dans cette partie nous vous proposons d'étudier la garantie de performance d'une heuritique pour le problème BIN-PACKING.

Considérons l'heuristique qui consiste à empaqueter dans l'ordre les articles $a_1, \ldots, a_n$ et un nouveau conteneur est utilisé à chaque fois qu'un article ne peut pas être empaqueté dans un des conteneurs déjà utilisés. Appelons H cette heuristique.

1. Rappeler la définition d'un algorithme approché avec un ratio de performance ρ.
2. Montrer que l'heuristique H est un algorithme 2-approché pour le problème BIN-PACKING.

Exercice 3 [Problème Δ-TSP]

Dans ce problème, il sera question d'une heuristique avec garantie de performance pour le problème du *voyageur de commerce euclidien* (Δ-TSP).

Soit $G = \langle V, E, \gamma \rangle$ un graphe non orienté et tel que $\gamma : E \to \mathbb{R}$ soit une valuation des arêtes de G. Le graphe G est qualifié d'euclidien si la valuation est *euclidienne*, c'est-à-dire, pour trois sommets quelconques u, v et w appartenant à V tels que uv, uw et wv appartiennent à E nous avons :

$$\gamma(uv) \leq \gamma(uw) + \gamma(wv).$$

Le problème (Δ-TSP) consiste à déterminer dans G le *plus court circuit Hamiltonien*. Un circuit est dit *hamiltonien* si, en le parcourant, nous visiterons *une et une seul fois* tous les sommets du graphe G.

Partie A : Circuit Eulérien

Dans cette partie du problème, nous traiterons le problème du *circuit eulérien* dans un graphe non orienté donné.

Soit G un graphe non orienté (pas nécessairement valué). Un circuit de G est qualifié *d'eulérien* si en le parcourant nous visiterons toutes les arêtes de G *une et une seule fois*.

1. Démontrer que les propositions suivantes sont équivalentes :
   - ✍ G est un graphe eulérien
   - ✍ G est connexe et tous ses sommets sont de degré pair
   - ✍ Il existe une partition $\{C_1, \ldots, C_s\}$ de l'ensemble E telle que pour tout k les arêtes de C_k forment un circuit où chaque sommet est visité une et seule fois (on dit aussi circuit élémentaire ou cycle).
2. En déduire un algorithme pour déterminer un circuit eulérien dans G.
3. A quelle classe de complexité appartient le problème de décision suivant : Soit G un graphe non orienté. Le graphe G est-il Eulérien ? Justifier votre réponse.

Partie B : Heuristique 2-approchée pour le Δ-TSP

Dans cette seconde partie, nous vous proposons d'étudier un algorithme 2-approché pour le problème (Δ-TSP). Pour cela, nous considérerons l'instance particulière formée du graphe complet dont les sommets sont les 7 points du plan dont nous avons représenté l'arbre de longueur minimum les couvrants (nous vous laissons le choix des valeurs). Cette arbre sera nommé T

1. Soit H le graphe obtenu à partir de l'arbre couvrant T en dupliquant seulement ses arêtes. Donc, H possède les mêmes sommets que le graphe G. Démontrer que H est Eulérien.

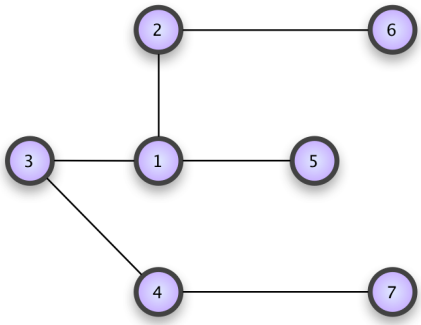

FIGURE 1 – Graphe exemple G

2. Déterminer, en utilisant votre algorithme de la partie précédente, un circuit Eulérien dans H. Vous prendrez soin de garder l'ordre de parcours des arêtes de H. Comparer le poids du circuit Eulérien trouvé avec celui de l'arbre couvrant T.
3. Construire un circuit Hamiltonien à partir du circuit Eulérien de la question précédente. Comparer son poids par rapport à celui d'un circuit Hamiltonien de poids minimum. Que remarquez-vous ?
4. Proposer un algorithme 2-approché pour le problème (Δ-TSP).