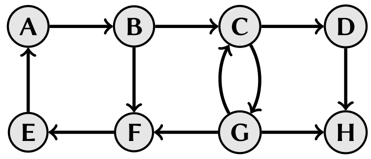
Outils logiques et algorithmiques – TD 4 – Chemins et terminaison

Exercice 1 (Vrai ou faux ?) À propos du graphe suivant :



1. Il existe un chemin de F à G.
2. Il existe un chemin de D à C.
3. Il existe un cycle passant par D.
4. Il existe un chemin ayant D comme point de départ et comme point d'arrivée.

□

Exercice 2 (Circuits Hamiltoniens) L'*hypercube* de dimension n est un graphe défini comme suit :

— ses sommets sont les mots de longueur n sur l'alphabet $\{0, 1\}$,

— il y a une arête entre le sommet $a_1 \ldots a_n$ et le sommet $b_1 \ldots b_n$ si et seulement si les deux mots diffèrent d'une seule lettre, c'est-à-dire s'il existe un entier i entre 1 et n tel que $a_i \neq b_i$ et pour tout entier j entre 1 et n, si $j \neq i$ alors $a_j = b_j$.

On appelle *circuit hamiltonien* un cycle qui passe une unique fois par chaque sommet.

1. Dessiner les graphes HC_0, HC_1, HC_2, HC_3.
2. Quel sont le nombre de sommets, le nombre d'arêtes, et le degré des sommets du graphe HC_n ?
3. Construire un circuit hamiltonien pour l'hypercube de dimension 2.
4. On suppose que l'on connait un circuit hamiltonien dans l'hypercube de dimension n, montrer comment en déduire un circuit hamiltonien pour l'hypercube de dimension $n + 1$.
5. En déduire des circuits hamiltoniens des hypercubes de dimensions 3 et 4.

□

Exercice 3 (Petits trains) Mes enfants rangent leurs petits trains. Ils optent pour la technique suivante : prendre un train non rangé au hasard, s'il contient un seul élément le ranger dans la caisse, sinon le séparer en deux trains plus petits (et les laisser avec les trains non rangés). Recommencer tant qu'il reste des trains à ranger.

1. On considère les trains non rangés. À chaque étape, comment évolue le nombre de trains ? Le nombre de wagons (on néglige la nuance entre wagon et locomotive) ? Le nombre de trains d'une taille donnée ? Ces éléments peuvent-ils donner un ordre justifiant que ce rangement terminera un jour ?
2. On propose de représenter une configuration des trains à ranger par un graphe dont les sommets sont les wagons. Que pourraient représenter les arêtes ? Utiliser les caractéristiques du graphe pour justifier que le rangement terminera bien.

□

Exercice 4 (Bug de terminaison) Voici une réalisation de la recherche dichotomique en java. On prétend comme d'habitude que ce programme termine à coup sûr avec le variant `hi-lo`.

```java
static boolean search(int e, int[] t) {
    int lo = 0, hi = t.length;
    while (lo < hi) {
        int mid = lo + (hi-lo)/2;
        if (t[mid] == e) return true;
        else if (t[m] < e)  { lo = mid; }
        else /* t[m] > e */ { hi = mid; }
    }
    return false;
}
```

Montrer que l'exécution de ce programme est susceptible de ne pas terminer dans certains cas, et donc que ce programme est erroné. *Indication : essayez sur des exemples très, très simples.* □

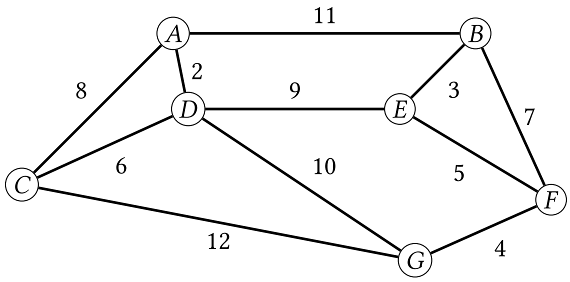
Exercice 5 (Tournée générale !) Dans cet exercice, on regarde des graphes non orientés dont les arêtes ont des longueurs (strictement positives). Un graphe est *connexe* si pour tous deux sommets s_1 et s_2 il existe un chemin entre s_1 et s_2. Une *couverture connexe* d'un graphe $G = (S, A)$ est un graphe connexe $G' = (S', A')$ avec $S = S'$ et $A' \subseteq A$. Une *couverture connexe minimale* est une couverture connexe pour laquelle la somme des longueurs des arêtes est minimale.

1. Montrer qu'un graphe admettant une couverture connexe est lui-même connexe.
2. Montrer qu'une couverture connexe minimale n'a pas de cycle.
3. Donner un graphe pour lequel il existe plusieurs couvertures connexes minimales.
4. Considérons un graphe G, un cycle ρ dans G, et une arête a de ce cycle dont la longueur est strictement supérieure aux longueurs des autres arêtes de ρ. Montrer qu'une couverture connexe de G contenant a ne peut pas être minimale.
5. En déduire que si toutes les arêtes ont des longueurs différentes, alors la couverture minimale connexe est unique.

L'algorithme de Kruskal construit une couverture connexe minimale :

```
G' = (S, ensemble vide)
trier A par ordre de longueurs croissantes
pour chaque a dans A
    si a relie deux sommets qui ne sont pas joignables par un chemin dans G'
    alors ajouter a à G'
renvoyer G'
```

6. Appliquer l'algorithme au graphe suivant.



7. Quels éléments définissent la complexité en temps de cet algorithme ?

Considérons maintenant un ensemble E de points du plan à deux dimensions. Une *tournée* de E est un itinéraire partant de l'un des points de E et y revenant après être passé par chacun des autres points au moins une fois. Trouver une tournée de longueur minimale est un problème algorithmiquement difficile [1]. On va montrer ici que l'on peut en revanche trouver efficacement une tournée dont la longueur ne dépasse pas le double de la longueur minimale.

8. Proposer une manière d'associer un graphe à l'ensemble E, et de traduire notre objectif dans le vocabulaire des graphes.
9. Montrer que toute tournée a une longueur strictement plus grande que la somme des longueurs des arêtes d'une couverture connexe minimale.
10. Montrer comment, à partir d'une couverture connexe minimale, on peut construire une tournée dont la longueur est deux fois la somme des longueurs des arêtes de cette couverture.

□

1. Le problème est *NP-complet*, et on ne connaît que des algorithmes dont la complexité est exponentielle en le nombre de points.