# M1.L5 : Série d'exercices sur la théorie du calcul + compléments

## 1 Dénombrabilité

a) L'hôtel infini partie 1 : un hôtel infini avec des chambres numérotées 1, 2, ... est plein et un nouveau client arrive : vérifier qu'on peut lui trouver une chambre bien que toutes les chambres soient utilisées.

b) L'hôtel infini partie 2 : l'hôtel est toujours plein; un bus infini arrive (avec des sièges numérotés 1, 2, ....); le bus est plein et tous ces passagers veulent une chambre dans l'hôtel. Vérifier que l'on peut trouver une chambre pour chacun d'entre eux.

c) L'hôtel infini partie 3 : l'hôtel est toujours plein. Cette fois une file infinie de bus infinis arrive (les bus sont numérotés 1, 2, ... et les sièges de chaque bus sont numérotés 1, 2, ...); chaque bus est plein et chaque passager veut une chambre. Vérifier qu'on peut trouver une chambre pour chacun d'entre eux.

## 2 Classes de problèmes P vs NP

Parmi les affirmations suivantes, laquelle ou lesquelles sont vraies?

a) Si un problème est dans la classe P, alors il est aussi dans la classe NP.
b) On sait avec certitude à l'heure actuelle que si un problème est dans la classe NP, alors il est aussi dans la classe P.
c) Si un problème est dans la classe P, alors il n'est pas dans la classe NP.
d) On sait avec certitude à l'heure actuelle qu'il existe un problème dans la classe NP qui n'est pas dans la classe P.
e) Tous les problèmes connus dans le monde sont dans la classe NP.

## 3 Classe de problèmes P vs NP *(in english)*

Which of the following discoveries would imply that P = NP:

1) Finding an exponential algorithm for sorting a list
2) Finding an exponential algorithm for a problem for which we only know a polynomial algorithm
3) Managing to color a specific graph such that no two adjacent vertices share the same color
4) Finding a polynomial algorithm for the Traveling Salesman problem
5) Finding a polynomial algorithm for a logarithmic problem

## 4 Expression de l'ordre de complexité avec plusieurs paramètres

Dans le cas général l'ordre de complexité d'un algorithme s'exprime en fonction de l'ensemble des paramètres qui expriment la taille du problème.

Dans l'exemple suivant de l'affichage du résultat de la multiplication d'une matrice rectangulaire de N lignes x M colonne par un vecteur de M éléments, les paramètres du problèmes sont N et M.

Quel est l'ordre de complexité de l'algorithme avec la notation de Landau O(…) en fonction de N et M ?

Notations en pseudocode:

Un vecteur V de N éléments est représenté de la même manière qu'une liste dans les séries précédentes. On accède à un élément en indiquant son indice de 1 à N, par exemple V(1).

L'accès à un élément d'une matrice A se fait comme en math à l'aide de deux indices qui commencent à 1 et varient jusqu'au nombre de ligne/ de colonne. On indique **d'abord l'indice de ligne suivi par l'indice de colonne** : A(1,2) désigne l'élément de la première ligne et qui est dans la deuxième colonne de cette ligne. Pour les boucles, une convention fréquente est d'utiliser l'indice **i** pour indiquer la ligne et l'indice **j** pour indiquer la colonne.

| Multiplication : matrice * vecteur |
|---|
| entrée :<br>*matrice A de N lignes et M colonnes*<br>*vecteur V de M éléments*<br>sortie :<br>*vecteur W de N éléments* |
| Pour i allant de 1 à N<br>W(i) ← 0<br>Pour j allant de 1 à M<br>W(i) ← W(i) + A(i,j)*V(j)<br><br>Pour k allant de 1 à N<br>Afficher W(k)<br><br>Sortir : W |

# 5 Manipulation de grands nombres et ordre de complexité.

AVERTISSEMENT : Pour cet exercice nous *sortons du cadre classique* utilisé pour évaluer l'ordre de complexité dans le cours et les séries précédentes.

Rappelons ce cadre « classique » : nous avons posé que les opérations arithmétiques de base utilisées dans ces exercices avaient un coût constant, indépendant de la valeur des opérandes. Cette hypothèse de travail est correcte tant qu'on travaille avec les nombres appartenant au *domaine couvert* par les types de base (entier, nombre à virgule flottante en double précision) car en effet on a bien un coût constant pour l'exécution des opérations arithmétiques sur les processeurs. Ce cadre est pertinent pour traiter un large éventail de problèmes et nous ne le remettons pas en question au sens large.

Pourquoi cet exercice sort-il du cadre classique ?: il existe quelques familles d'algorithmes qui doivent manipuler des grands nombres pour apporter certaines garanties de qualité. Une telle famille est la **cryptographie** sur laquelle le dernier cours du semestre apportera un complément d'information. Ce que nous pouvons déjà préciser c'est que des *nombres de plusieurs dizaines de chiffres* (en base 10) doivent être traités par ces algorithmes, ce qui sort du domaine couvert des types de base.

Pour les questions suivantes, **la taille N du problème** n'est plus l'amplitude du nombre **n** qui est manipulé MAIS c'**est le nombre N de chiffres de n** (par exemple en base 10). Posons que X et Y sont deux grands nombres de **N chiffres.**

a) Quel est l'ordre de complexité de l'opération X + Y ?

b) Quel est l'ordre de complexité de l'opération X * Y ?

c) Soit Z un nombre obtenu par le produit de deux grands nombres premiers P et Q. L'ordre de complexité du calcul de Z est donné par la question précédente. Quel est l'ordre de complexité de l'opération inverse : **retrouver P et Q à partir de Z** ? Le pseudocode ci-dessous est directement inspiré de la détermination de primalité d'un nombre n car le problème est très proche. La différence est qu'ici on renvoie le premier diviseur trouvé pour n. Une fois que l'on connait P ou Q, on connait l'autre.

Voici le pseudocode qui détermine
Le premier diviseur d'un nombre n.

**Premier_diviseur**

entrée : *n, entier naturel*
sortie : *premier diviseur de n*

```
Si  n ≤ 3
    Sortir : n

max ← racine(n)
x ← 2

Tant que  x ≤ max
    Si (n mod x) = 0
        Sortir  x
    x ← x + 1

Sortir  n
```

Il faut donc exprimer la complexité de ce pseudocode en fonction de « **la taille N du nombre n** ».

Information complémentaire pour cette évaluation :

La racine carrée de **n** n'est calculée qu'une seule fois ; son coût est constant ; on le néglige devant le coût des passages dans la boucle.

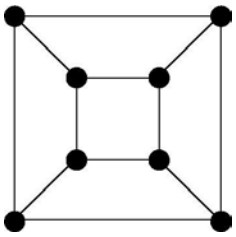
Au niveau des opérations effectuées dans la boucle il y a, à chaque passage, une opération de modulo (qu'on supposera en $O(N^2)$) et 2 comparaisons (qu'on supposera en O(N)) pour savoir si on reste dans la boucle et si le résultat du modulo est nul.

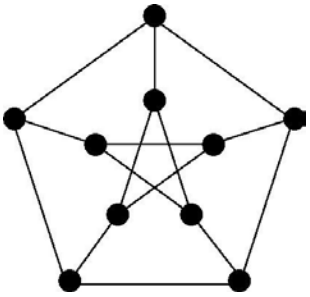
La question est finalement : comment exprime-t-on la complexité du passage dans la boucle Tant que en fonction de N ?

d) le résultat obtenu à la question précédente est-il suffisant pour déterminer si ce problème appartient à P ou pas?

# 6 Coloriage de graphes

On considère un graphe avec *n* sommets et un certain nombre d'arêtes qui relient ces sommets, comme par exemple un des deux graphes suivants :





On se pose la question générale suivante :

*Soit k ≥ 2 un nombre fixé. Avec k couleurs différentes à disposition, est-il possible de colorier les sommets d'un graphe donné de façon à ce que si deux sommets sont reliés par une arête, alors ils aient toujours des couleurs différentes?*

Avant d'aller plus loin, voici une petite application pour le cas où $k = 2$ : $n$ joueurs se retrouvent ensemble et désirent former deux équipes (pas forcément de même taille : ça simplifie le problème). Seule contrainte : un graphe comme le graphe ci-dessus indique les joueurs qui ne s'aiment pas et ne veulent donc pas faire partie de la même équipe : plus précisément, $i$ n'aime pas $j$ si et seulement si $i$ et $j$ sont reliés directement par une arête. Est-il possible de former deux équipes avec des joueurs qui n'ont aucune inimitié à l'intérieur de chaque équipe?

a) Quelle est la réponse à cette question pour chacun des graphes ci-dessus (dans le cas où $k = 2$)?

Faisons maintenant un petit calcul ensemble : pour résoudre la question en général pour un $n$ et un $k$ donnés, on a toujours l'option d'essayer *toutes* les possibilités de coloriages du graphe. Combien sont-elles, ces possibilités? Vu qu'on a $k$ choix pour chacun des $n$ sommets, on a en tout $k^n$ possibilités, autrement dit un nombre qui croît exponentiellement en $n$. Si $n$ est grand, essayer toutes les possibilités prend clairement trop de temps (même pour $k = 2$).

b) Considérons tout d'abord le cas particulier $k = 2$ et supposons que vous deviez trouver vous-même un coloriage qui marche, sans aide extérieure. Quel algorithme utiliserez-vous pour trouver une solution au problème? (qu'avez-vous fait pour répondre à la question a?)

c) Toujours dans le cas $k = 2$, combien d'opérations *au pire*[1] seront-elles nécessaires pour trouver une solution (ou au contraire conclure qu'une telle solution n'existe pas), en fonction du nombre de sommets $n$? (donner la réponse en utilisant la notation de Landau $O(\cdot)$)

d) Considérons maintenant le cas plus général $k \geq 2$ et supposons qu'on vous *donne* un coloriage avec $k$ couleurs pour un graphe donné et qu'on vous demande de *vérifier* si ce coloriage fonctionne. En fonction du nombre de sommets $n$, combien d'opérations seront-elles nécessaire pour vérifier que le coloriage fonctionne, dans le pire des cas? (utiliser à nouveau la notation de Landau $O(\cdot)$)

e*) Dans le cas particulier $k = 3$, quel algorithme utiliserez-vous pour trouver une solution au problème, si vous laisse la trouver par vous-même? (à nouveau, essayez d'abord sur les exemples de graphes ci-dessus) Et combien d'opérations seront-elles nécessaires pour trouver une solution (ou au contraire conclure qu'une telle solution n'existe pas)?

*Attention :* Cette dernière question est (beaucoup) plus difficile qu'il n'y paraît : en fait, même les plus grands scientifiques de la planète n'ont encore trouvé la réponse : ne désespérez donc pas si vous n'y arrivez pas du premier coup!

*Note historique :* C'est grâce à l'informatique qu'on a pu démontrer *mathématiquement* que 4 couleurs suffisent pour colorier tous les graphes dits *planaires*, c'est-à-dire les graphes correspondant à nos bonnes vieilles cartes de géographie (où on identifie les pays avec les sommets du graphe et les frontières communes entre deux pays avec les arêtes du graphe).

[1] . Imaginez le graphe le plus complexe possible avec $n$ sommets.