Université USTHB – Bab-Ezzouar Bab-Ezzouar, 14 Novembre 2018

Faculté de l’Electronique et de l’Informatique, Département de l’Informatique Année universitaire 2018/2019

1ère année Master Informatique, Semestre 1 Semestre 1

Module : Conception et Complexité des Algorithmes

-------------------------------------------------------------------------------------------------------------------------------------------------------

**Série de Travaux Pratiques n° 4 (TP n° 4)**

**Algorithmes de complexité temporelle linéaire O(an) (a>1)**

**Note :** Un minimum de connaissances en programmation est indispensable pour le suivi des séances des travaux pratiques.

L’objet de ce TP est l’étude expérimentale de l’algorithme de résolution du problème des ″tours de Hanoi″. On s’intéresse à l’algorithme récursif. C’est un problème classique en informatique. Il montre la puissance et la lisibilité des algorithmes définis de façon [récursive](http://fr.wikipedia.org/wiki/R%C3%A9cursivit%C3%A9). On utilise le langage de programmation C.

**Le problème des ″tours de Hanoi″**

Il est décrit dans wikipédia (encyclopédie universelle libre) comme suit :

C’est un jeu de réflexion imaginé par le [mathématicien](http://fr.wikipedia.org/wiki/Math%C3%A9maticien) français [*Édouard Lucas*](http://fr.wikipedia.org/wiki/%C3%89douard_Lucas), et consistant à déplacer des disques de diamètres différents d'une tour de ″départ″ à une tour d' ″arrivée″ en passant par une tour ″intermédiaire″ et ceci en un minimum de coups, tout en respectant les 2 règles suivantes :

1. on ne peut déplacer plus d'un disque à la fois,
2. on ne peut placer un disque que sur un autre disque plus grand que lui ou sur un emplacement vide. On suppose que cette dernière règle est également respectée dans la configuration de départ.

Il est publié dans le tome 3 de ses *Récréations mathématiques*, parues à titre posthume en [1892](http://fr.wikipedia.org/wiki/1892). Sous le titre ″*Les brahmes tombent*″, Lucas relate que :

″ *N. Claus de Siam a vu, dans ses voyages pour la publication des écrits de l'illustre Fer-Fer-Tam-Tam, dans le grand temple de* [*Bénarès*](http://fr.wikipedia.org/wiki/B%C3%A9nar%C3%A8s)*, au-dessous du dôme qui marque le centre du monde, trois aiguilles de diamant, plantées dans une dalle d'airain, hautes d'une coudée et grosses comme le corps d'une abeille. Sur une de ces aiguilles, Dieu enfila au commencement des siècles, 64 disques d'or pur, le plus large reposant sur l'airain, et les autres, de plus en plus étroits, superposés jusqu'au sommet. C'est la tour sacrée du* [*Brahmâ*](http://fr.wikipedia.org/wiki/Brahm%C3%A2)*. Nuit et jour, les prêtres se succèdent sur les marches de l'autel, occupés à transporter la tour de la première aiguille sur la troisième, sans s'écarter des règles fixes que nous venons d'indiquer, et qui ont été imposées par Brahma. Quand tout sera fini, la tour et les brahmes tomberont, et ce sera la fin des mondes* ″[].

**Rem :** On note qu’un jeu à 64 disques requiert un minimum de 264-1 déplacements. En admettant qu'il faille 1 seconde pour déplacer un disque, ce qui fait 86 400 déplacements par jour, la fin du jeu aurait lieu au bout d'environ 213 000 milliards de jours, ce qui équivaut à peu près à 584,5 milliards d'années, soit 43 fois l'âge estimé de l'univers (13,7 milliards d'années selon certaines sources).

1- Ecrire un algorithme récursif qui résout le problème des ″tours de Hanoi″. On suppose qu’il y a n disques à transférer (n est un entier naturel, n>=1).

2.1- Calculer la complexité temporelle de cet algorithme, notée f(n). (cette question est supposée être traitée en séances de TD, sinon la laisser pour les séances ultérieures).

2.2- Calculer la complexité spatiale de cet algorithme, notée s(n). (là aussi, on a la même remarque que celle de la question 2-1).

3- Ecrire avec le langage C le programme correspondant.

4- Mesurer les temps d’exécution T pour un échantillon de données de la variable n et représenter les résultats sous la forme d’un tableau (ci-dessous).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **…** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** |
| **T(s)** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **29** | **30** | **31** | **32** | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** | **…** | **…** | **…** | **64** |
| **T(s)** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

5- Développer un programme de mesure du temps d’exécution du programme qui a en entrée les données de l’échantillon ci-dessus et en sortie les temps d’exécution. Les données et les mesures du temps sont à enregistrer dans des tableaux notés respectivement Tab1 et Tab2.

6- Représenter par un graphe, noté Gf(n), les variations de la fonction de la complexité temporelle en fonction de n ; et par un autre graphe, noté GT(n), les variations du temps d'exécution T(n) en fonction de n. Utiliser pour cela un logiciel graphique tel que excel.

7- Interprétation des résultats.

7.a- Les mesures du temps obtenues correspondent-elles au meilleur cas, au pire cas, au cas moyen ou au cas exact ?

7.b- Que remarque-t-on sur les données de l'échantillon et sur les mesures obtenues ? Peut-on déduire, même de façon approximative, une fonction T(n) reliant T et n ; c'est-à-dire une fonction T(n) permettant de déterminer directement la valeur de T à partir de n.

**Ind :** comparer chaque nombre n avec le suivant ; et chaque mesure du temps avec la suivante.

7.c- Comparer entre la complexités théorique et la complexité expérimentale (çàd., les mesures expérimentales). Les prédictions théoriques sont-elles compatibles avec les mesures expérimentales ?

**Rem :** Il est intéressant de chercher une v**ersion itérative pour cet algorithme.**