**INF8775 – Analyse et conception d’algorithmes**

Rapport TP1 – Hiver 2022

|  |  |
| --- | --- |
| **Nom, prénom, matricule des membres** | Souque, Guillaume, 1839071  Juillet, Hugo, 1830925 |
| **Note finale / 30** | 0 |

# Informations techniques

* Répondez directement dans ce document DOCX. Veuillez ne pas inclure le texte en italique servant de directive.
* La correction se fait sur ce même rapport.
* Vous devez faire une remise électronique sur Moodle avant le 23 Février à 23h59 en suivant les instructions suivantes :
  + Vos fichiers doivent être remis dans une archive zip à la racine de laquelle on retrouve :
    - Ce rapport au format DOCX.
    - Un script nommé *tp.sh* servant à exécuter les différents algorithmes du TP. L’interface du script est décrite à la fin du rapport.
    - Le code source et les exécutables.
    - Si le langage que vous utilisez nécessite une phase de compilation, veuillez joindre un Makefile afin que nous puissions le compiler en cas de problème avec vos exécutables. Si nous ne sommes pas en mesure de tester votre code, vous perdrez des points de respect d’interface et de qualité de code !
* Vous avez le choix du langage de programmation utilisé mais vous devrez utiliser les mêmes langage, compilateur et ordinateur pour toutes vos implantations. Le code et les exécutables soumis devront être compatibles avec les ordinateurs de la salle L-4714.
* Si vous utilisez des extraits de codes (programmes) trouvés sur Internet, vous devez en mentionner la source, sinon vous serez sanctionnés pour plagiat.

# Présentation des résultats

|  |  |
| --- | --- |
|  | / 1,5pts |

## Tableau des résultats

|  |  |  |  |
| --- | --- | --- | --- |
| Taille d’exemplaire (n) | Temps d’exécution moyen (secondes) | | |
| Brute | Diviser | Diviser avec seuil |
| 1000 | 0,615521 | 0,008802 | 0,008201 |
| 5000 | 15,78361 | 0,046409 | 0,045399 |
| 10000 | 63,53669 | 0,097406 | 0,096812 |
| 50000 | 1579,956 | 0,565643 | 0,559532 |
| 100000 | 6637,955 | 1,170754 | 1,153813 |
| 500000 | NA | 6,614127 | 6,371668 |

La taille d’exemplaire 500000 n’a pas été testée pour l’algorithme brute. En effet, si le taux de croissance de l’algorithme brute s’apparente à . Alors l’exécution de l’exemplaire entrainerait un temps d’exécution fois plus grand que l’échantillon 100000, ce qui donnerait 50 heures d’exécution en moyenne.

# Analyse et discussion

## Citez la consommation théorique du temps de calcul pour les algorithmes, en notation asymptotique.

|  |  |
| --- | --- |
|  | / 1,5 pt |

2. La consommation théorique de l’algorithme force brute est de .
3. La consommation théorique de l’algorithme diviser pour régner est de .
4. La consommation théorique de l’algorithme diviser pour régner avec seuil est de .

## Tests de puissance

|  |  |
| --- | --- |
|  | / 5 pt |

## 

## Que pouvez-vous déduire du test de puissance ?

Le test de puissance nous permet d’obtenir une idée générale du taux de croissance de nos algorithmes. On peut voir pour l’algorithme brute que le taux de croissance s’approche d’une croissance polynomiale de degré . Cette valeur se rapproche de la valeur de croissance théorique de .

Pour les algorithmes diviser pour régner et diviser pour régner avec seuil, on peut constater une valeur de sans le seuil et avec. On peut donc constater que les deux algorithmes semblent avoir une croissance polynomiale de complexité similaire qui se rapproche de la croissance théorique de .

## Test du rapport

|  |  |
| --- | --- |
|  | / 5 pt |

## 

Le rapport effectué pour l’algorithme brute est : avec . Avec le temps en **secondes**.

Le rapport effectué pour l’algorithme diviser est : avec . Avec le temps en **millisecondes**.

Le rapport effectué pour l’algorithme diviser avec seuil est : avec . Avec le temps en **secondes**.

## Que pouvez-vous déduire du test du rapport ?

* 1. Pour les algorithmes diviser pour régner avec et sans seuil, on peut constater que le test converge vers une constante b telle que pour les deux algorithmes et pour une hypothèse de . Cela voudrait dire que notre hypothèse est validée, et que les deux algorithmes suivent bien une croissance s’apparentant a avec une constante multiplicative de valeur 0,0000025 environ.

Pour l’algorithme brute. On peut constater une convergence vers la valeur pour une hypothèse de . Cela voudrait dire que notre hypothèse est validée et que le taux de croissance de la complexité de l’algorithme brute s’apparente bien a , avec une constante multiplicative de 0,0000067 environ.

## Test des constantes

|  |  |
| --- | --- |
|  | / 5 pt |

## 

## Que pouvez-vous déduire du test des constantes ?

Pour le test de constante, on peut constater des valeurs de constantes multiplicatives assez proches de celles trouvées lors du test de rapport. Pour les algorithmes diviser pour régner, les valeurs sont identiques tandis que la valeur de constante pour l’algorithme brute est plus faible ( contre ). Celles-ci confirment donc les hypothèses posées précédemment mais aussi réaffirme les valeurs de constantes multiplicatives de ces algorithmes.

## Discutez de l’impact du seuil de récursivité.

|  |  |
| --- | --- |
|  | / 2 pt |

La valeur du seuil de récursivité sélectionné est de . Pour des valeurs plus élevées, le temps d’exécution de l’algorithme brute devient trop élevé sur les sous-exemplaires. Le seuil de présente aussi de meilleures performances que l’algorithme diviser pour régner sans seuil mais la valeur a présenté les meilleurs résultats.

## Suite à cette analyse, indiquez sous quelles conditions (taille d’exemplaire ou autre) vous utiliseriez chacun de ces algorithmes. Prenez en compte la complexité spatiale et temporelle, le temps de calcul et la difficulté d'implémentation. Justifiez.

|  |  |
| --- | --- |
|  | / 3 pt |

Pour des très petites tailles d’exemplaire (), l’algorithme brute présente de meilleures performances. Cependant pour des tailles d’exemplaires supérieures, les algorithmes diviser pour régner sont à privilégier. Entre les algorithmes avec ou sans seuil, le temps d’exécution est très proche. L’algorithme sans seuil présente un taux de croissance légèrement plus bas que l’algorithme avec seuil (d’après le test de puissance) mais l’algorithme avec seuil possède une constante plus basse. Cela veut dire que sur de très très gros exemplaires (), l’algorithme sans seuil serait préférable. Pour des tailles inferieures, l’algorithme avec seuil est préférable.

# Autres critères de correction

## Respect de l’interface tp.sh

|  |  |
| --- | --- |
|  | / 1 pt |

## Qualité du code

|  |  |
| --- | --- |
|  | / 5 pt |

* + - * 1. Validité des solutions
        2. Qualité de l'implémentation

Présence de commentaires

## Présentation générale

|  |  |
| --- | --- |
|  | / 1 pt |

* Concision
* Qualité du français

## Pénalité retard

|  |
| --- |
| 0 |

* -15% de la note / journée de retard, arrondi vers le haut. Les TPs ne sont plus acceptés après 3 jours.