INF8775 – Analyse et conception d’algorithmes

TP1 – Automne 2023

|  |  |
| --- | --- |
| **Nom, prénom, matricule des membres** | Nom, Prénom, matricule  Nom, Prénom, matricule |
| **Note finale / 13** | 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 10 octobre à 23h55 pour le groupe 1, 3 octobre à 23h55 pour le groupe 2 et 5 octobre à 23h55 pour le groupe 3.
* Vos fichiers doivent être remis dans une archive zip à la racine de laquelle on retrouve:
  + Ce rapport sous 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.
  + Vous avez le choix du langage de programmation utilisé mais vous devrez utiliser les mêmes langage, compilateur et ordinateur pour toutes vos implantations. Notez que le code et les exécutables soumis seront testés sur les ordinateurs de la salle L-4714 et doivent être compatibles avec cet environnement. En d’autres mots, tout doit fonctionner correctement lorsque le correcteur exécute votre script tp.sh sur un des ordinateurs de la salle.
* Si vous utilisez des extraits de codes (programmes) trouvés sur Internet, vous devez en mentionner la source, sinon vous serez sanctionnés pour plagiat.
* On vous encourage à lire le guide intitulé « guide bash » sur Moodle pour faire vos graphiques. C’est un guide qui a été conçu pour un ancien TP, mais il contient beaucoup d’informations utiles.

# Mise en situation

Ce travail pratique se répartit sur deux séances de laboratoire et porte sur l’analyse empirique et hybride des algorithmes. À la section 3.2 des notes de cours, trois approches d’analyse de l’implantation d’un algorithme sont décrites. Vous les mettrez en pratique pour des algorithmes de tri.

Vous implanterez les algorithmes de tri par dénombrement et de tri rapide (quicksort). Un rappel de ces algorithmes sera fait lors de la séance de laboratoire. Vous ferez trois versions de l'algorithme de tri rapide :

1. Pivot sur le 1er élément et seuil de récursivité à 1
2. Pivot sur le 1er élément et seuil de récursivité déterminé expérimentalement
3. Pivot sur un élément aléatoire et seuil de récursivité déterminé expérimentalement

Les exemplaires dont la taille est en deçà du seuil de récursivité ne sont plus résolus récursivement mais plutôt directement avec un algorithme de tri de votre choix prenant un temps dans Θ(n2) en pire cas (tri par insertion, par sélection, à bulles, etc.)

# Jeu de données

Vous trouverez dans l’archive du TP un script python *gen.py* servant à générer les exemplaires. Ce script s’exécute de la manière suivante :

gen.py -S TAILLE [-n NB\_EXEMPLAIRES] [-m MAGNITUDE] [-r RANDOM\_SEED] [-p PSEUDO-TRI]

TAILLE correspond à la taille des exemplaires.

NB\_EXEMPLAIRES correspond au nombre de suites que vous voulez générer pour chaque taille

MAGNITUDE correspond à la valeur maximale possible dans les exemplaires. Par exemple, pour -m 100000, les exemplaires comportent des nombres de 0 à 100000.

RANDOM\_SEED correspond à la *seed* utilisée pour la génération aléatoire des listes de nombres.

PSEUDO-TRI est une variable binaire qui détermine si l’exemplaire sera presque trié.

***Les séries à générer sont les suivantes :***

1. *Une série pour laquelle la taille des exemplaires augmente*
2. *Une série pour laquelle la magnitude des exemplaires augmente*
3. *Une série pour laquelle la taille des exemplaires ET la magnitude des valeurs à trier augmente*
4. *Une série presque triée pour laquelle la taille des exemplaires augmente*

# Présentation des résultats

|  |  |
| --- | --- |
| 0 | / 4 pt |

### Tableau des résultats

*Pour chacun des quatre algorithmes, mesurez le temps moyen d’exécution pour un groupe d'exemplaires de même taille (ex. 10 exemplaires de taille 1000), et rapportez-le dans un tableau pour toutes les tailles dans chaque série générée (donc séparément pour chaque groupe de même taille et chaque série générée). Les séries à évaluer selon les algorithmes sont les suivantes :*

* *Tri par dénombrement : séries 1, 2 et 3.*
* *Tris avec pivot (les 3 versions) séries 1 et 4.*

*Lorsque vous calculez les temps d’exécution, vous devez séparer le temps de chargement du jeu de test du temps d’exécution de votre algorithme. Vous devrez donc insérer les sondes temporelles à l’intérieur de votre code.*

### Tests de puissance

*Pour chacun des algorithmes et chacune des séries d’exemplaires, appliquez le test de puissance et rapportez les graphiques ici.*

### Test du rapport

*Pour chacun des algorithmes et chacune des séries d’exemplaires, appliquez le test du rapport en utilisant chacune des analyses en meilleur cas, en pire cas et en moyenne et rapportez les graphiques ici.*

### Test des constantes

*Pour chacun des algorithmes et chacune des séries d’exemplaires, appliquez le test des constantes et rapportez les graphiques ici. Dans chaque cas, choisissez l’analyse (meilleur cas, pire cas, cas moyen) la plus appropriée selon les résultats du test précédent (test du rapport).*

# Analyse et discussion

|  |  |
| --- | --- |
| 0 | / 6 pt |

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

### Citez la consommation théorique en meilleur cas, en pire cas et en moyenne du temps de calcul pour les algorithmes, en notation asymptotique.

*Nul besoin de faire une preuve, on demande seulement de citer.*

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

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

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

### Suite à cette analyse, indiquez sous quelles conditions (taille d’exemplaire ou autre) vous utiliseriez chacun de ces algorithmes. Justifiez.

# Autres critères de correction

### Respect de l’interface tp.sh

|  |  |
| --- | --- |
| 0 | / 1 pt |

Utilisation

tp.sh -a [counting | quick | quickSeuil | quickRandomSeuil] -e [path\_vers\_exemplaire]

Arguments optionnels

-p affiche les nombres triés en ordre croissant sur une ligne, sans texte superflu

-t affiche le temps d’exécution en ms, sans unité ni texte superflu

Important: l’option -e doit accepter des fichiers avec des paths absolus.

### Qualité du code

|  |  |
| --- | --- |
| 0 | / 1 pt |

### Présentation générale

|  |  |
| --- | --- |
| 0 | / 1 pt |

* Concision
* Qualité du français

### Pénalité retard

|  |
| --- |
| 0 |

* -1 pt / journée de retard, arrondi vers le haut. Les TPs ne sont plus acceptés après 3 jours.