L2 Info - Projet SDA II

Semestre 4A

merindol@unistra.fr

Vers un tri efficace!

Objectif : Manipuler et comparer des couples (structures de données, algorithmes) efficaces pour trier efficacement des données

Notions : Algorithmes de tris par comparaison (sélection, insertion, etc) et sans comparaisons

---

I Contexte

De nombreux algorithmes et applications reposent sur des algorithmes de tri. Il existe plusieurs manières de trier efficacement des données en fonction de leurs natures (et les hypothèses sous-jacentes les caractérisants).

II Sujet

Dans un premier temps, il s'agira de spécifier et d'évaluer (et justifier) les complexités d'algorithmes connues. Ce travail doit se faire de manière cohérente : même syntaxe de spécification notamment. Sur cette base, vous devrez implémenter en C et comparer expérimentalement les performances de ces algorithmes à celui de votre choix. De manière générale, vous apporterez un soin tout particulier à la manipulation des résultats issus de vos algorithmes afin de les "valider" plus facilement (par comparaison entre les différents algorithmes et leurs opérations) en prenant en compte les cas particuliers et les conditions à mettre en oeuvre.
La liste suivante résume les principales tâches qui sont attendues dans votre rendu (et les contraintes qui vont avec) :

- un tri efficace par sélection (i.e. HeapSort basé sur un tas binaire donc) ;
- un tri efficace par insertion (non séquentiel ou dichotomique mais basé sur un arbre de recherche équilibré) ;
- un tri efficace au pire et/ou en moyenne ayant une approche du type "diviser pour mieux régner" (e.g. MergeSort ou QuickSort) ;
- un tri (encore plus efficace si possible) de votre choix (e.g. avec une approche hybride et/ou sans comparaison) ;

Pour finir, vous menerez un jeu de tests sur un large ensemble de données pseudo-aléatoires (à générer par vos propres moyens ou via des outils de génération automatique) pour ainsi évaluer expérimentalement les performances de chacun des algorithmes analysés dans le cas "moyen". Cette partie (importante) du projet donnera lieu à une évaluation pratique de vos structures de données et algorithmes. Cette étude devra faire varier le nombre d'élements à trier, leurs caractéristiques en général (e.g. intervalles de valeurs) et représenter le résultat général sous forme d'une figure comprenant au moins quatre courbes agrémentés d'indicateurs statistiques de base (e.g., moyenne, médiane, min, max, quantiles, etc).

En options facultatives, vous pouvez étendre votre projet pour répondre aux objectifs suivants :

— utiliser vos algorithmes dans le cadre de calcul dans les graphes (e.g. arbres de poids minimum) ;
— prouver vos algorithmes et leurs complexités (temporelles et spatiales, au pire comme en moyenne).

III Modalités pratiques

Sauf exception, le projet devra être réalisé en binôme. Le langage à utiliser est le C.
En plus du code documenté et correctement indenté, vous devrez fournir un rapport clair et concis d'une dizaine de pages au maximum (jusqu'à 15 pages avec les options) reprenant les spécifications des principales procédures de votre implémentation (sans oublier de décrire leurs complexités au niveau microscopique). Pour chaque structure manipulée, illustrez et justifiez vos choix d'implémentation.

Le projet est à rendre pour le 18 décembre 2015 (date limite stricte) et sera pré-évalué en séance de TP le 14 décembre 2014. Le code et le rapport devront être déposés sous la forme d'une archive (zip ou tar.gz) sur le moodle SDA 2. En cas de problème, contactez moi à l'adresse suivante : `merindol@unistra.fr`.