SIT 152

Algorithmique & analyse de complexité
TP : comparaison d'algorithmes de tri

- Programmation en langage C
- Analyse de complexité
- Analyse expérimentale
- Confrontation des analyses et synthèse
- Proposition de recommandations

Expédiez vos commentaires à :

Philippe Lenca — *philippe.lenca@telecom-bretagne.eu*

Bernard Prou — *bernard.prou@telecom-bretagne.eu*

Performances des algorithmes de tri

Vous devez étudier les algorithmes de tris vus en cours (tri à bulle, tri par sélection, tri par insertion, tri rapide, tri fusion). On vous donne pour cela une archive à récupérer sur `moodle` contenant 5 fichiers :

- `data.c` contient les fonctions (spécifiées dans `data.h`) relatives à la lecture d'un tableau de données à partir d'un fichier, l'écriture d'un tableau de données dans un fichier, la génération de données triées dans un tableau, la génération de données triées en ordre inverse dans un tableau, la génération de données aléatoires dans un tableau ;
- `tris.c` contient les fonctions (spécifiées dans `tris.h`) correspondant aux algorithmes de tris ; horreur ⌢̈ `tris.c` est vide et vous devez insérerez les fonctions C (le langage C est étudié en SIT 151) nécessaires ⌣̈. Avant de programmer assurez-vous de bien comprendre les algorithmes (par exemple, en simulant à la main le fonctionnement de chacun d'entre-eux sur des données de taille réduite) ;
- `utilisation.c` contient la fonction main pour exécuter de façon paramétrée des tris.

L'archive contient également un `Makefile` permettant de (re)générer l'exécutable `a.exe` par la commande `make a.exe` ainsi que `fic10` et `fic100` des exemples de fichier de données.

Prise en main des programmes

- étudiez le `Makefile` ; Pouvez-vous compiler `utilisation.c` sans avoir écrit `tris.c` (grâce au spécifications) ? Est-ce que seul le nécessaire est (re)compilé ?

- étudiez enfin les `*.c` et `*.h` pour comprendre la structure globale du projet, les structures de données, les options du programme (vous pouvez également exécuter `a.exe` pour afficher les différentes options), etc.

Programmation et analyse de performances

- étudiez la complexité des différents algorithmes de tri ;
- programmez les algorithmes de tri en langage C dans `tris.c` avec les spécifications de `tris.h` ;
- placez convenablement différents *compteurs* (structure `structSondes` définie dans `tris.h`) afin d'étudier expérimentalement les performances de ces algorithmes ;
- définissez un protocole expérimental (données, paramètres, situations, etc.) ;
- analysez les résultats des expériences réalisées (comparez les avec l'analyse de complexité) ;
- proposez des recommandations et réalisez un bilan de votre travail.

Remarques :

- les structures à trier sont des tableaux d'entiers. Les clés et les données sont confondues mais vous pouvez évaluer les performances des algorithmes en tenant compte du fait que les clés sont difficiles à comparer et les données coûteuses à déplacer. Le but du TP est de simuler ce qui se passe et non de réaliser concrètement ce qui se passe ;
- deux séances sont programmées pour le TP. Toutes les fonctions de tri devront être réalisées avant le début de la seconde séance. Cette dernière devra être consacrée aux expériences.

Bon courage.