Module « Paradigmes et langages de programmation »

Sujet proposé par Olivier Lecarme

2006–2007

Démonstrateur de tris

Supposez un professeur d'informatique qui désire construire un démonstrateur général des algorithmes de tri. Ce démonstrateur est à la fois une bibliothèque d'outils de programmation et un outil d'expérimentation et d'explication. Parmi les outils de programmation, il faut fournir en particulier :

- plusieurs manières de construire des tableaux ou fichiers de valeurs (triées à l'endroit, triées à l'envers, toutes identiques, triées « au hasard » selon une bonne répartition, rangées de manière à mettre en évidence les performances les plus mauvaises de certains algorithmes, etc.) ;
- affichage graphique dynamique de l'évolution de l'ordre des clés, si cela est possible facilement avec le langage de programmation utilisé ; si ça ne l'est pas, on peut imaginer facilement une manière de représenter cette évolution de manière visuelle avec de simples histogrammes ; faire appel à `gnuplot` peut être une bonne idée ;
- mesures précises du temps de calcul, ainsi que des nombres d'opérations caractéristiques des tris (comparaisons et mouvements de clés), et présentation claire des statistiques obtenues sur un grand nombre d'essais, avec des nombres de clés croissants ;
- etc.

Enfin, le programme doit disposer d'une interface interactive, graphique ou non, qui permet de l'utiliser pour toutes les expérimentations qu'il propose.

Les algorithmes de tri programmés doivent l'être d'une manière qui utilise au mieux les possibilités du langage de programmation utilisé. Une bonne réutilisation des parties communes paraît en particulier indispensable. Attention à ne pas transformer des algorithmes $\mathcal{O}(n \log n)$ en algorithmes $\mathcal{O}(n^3)$ ou pire, par une utilisation maladroite de copies inutiles et de la récursivité ! À part les différentes variantes du tri par fusion et le cas très particulier du tri par base (tri des trieuses électro-mécaniques), les algorithmes de tri performants se font « sur place », sans copie des données, et posent donc le problème crucial de la *modification triviale*, que vous ne pouvez pas traiter de la même manière suivant le langage de programmation.

L'étude approfondie de la bonne manière de traiter ces problèmes suivant le paradigme de programmation choisi est donc l'aspect important de ce sujet, bien plus que la réalisation d'une interface graphique luxueuse.