Diviser pour régner

Pierre Karpman

Lycée Champollion MP2I
https://membres-ljk.imag.fr/Pierre.Karpman/CPGE/2025/

Table des matières

Paradigme algorithmique

Principe

- Donner un cadre pour la conception d'algorithmes
- Suggère une approche pour résoudre un problème
  - Qui sera adaptée ou non…

Exemples de paradigmes de cette année

- Dichotomie
- Diviser pour régner « DPR » (*divide & conquer*)
- Rencontre au milieu (*meet-in-the-middle*)
- Programmation dynamique « ProgDyn »
- Glouton (*greedy*)
- Retour sur trace (*backtracking*)

Objectifs

- Connaître les paradigmes
- Savoir choisir une approche adaptée dans un cas « simple » et la mettre en œuvre

Pas un objectif

Définir formellement les paradigmes

- Par exemple aucun intérêt d'avoir un théorème disant qu'un algo est un algo glouton…

Table des matières

Diviser pour régner

Principe

Résoudre un problème en :

1. Le divisant en plusieurs sous-instances pour le même problème, généralement en nombre constant (dépend de l'algorithme, et pas de l'entrée)
   - Cette division doit être possible pour pouvoir appliquer une méthode DPR…
2. Résolvant chacune des sous-instances récursivement
3. Combinant les solutions des sous-instances pour obtenir une solution complète
   - Étape parfois assez triviale

Bestiaire

Tri fusion

Exemple « paradigmatique »

Pour trier (un tableau, une liste) de n éléments on :

1. Le divise en deux sous-listes (quelconques) de $\lfloor n/2 \rceil$ éléments
2. Trie ces sous-listes (récursivement)
3. Construit le résultat en *fusionnant* les deux sous-listes triées

Coût : $T(n) \lessapprox 2T(n/2) + an = O(n \log n)$

Bestiaire 2

Tri rapide

Pour trier (un tableau, une liste) de n éléments on :

1. Le divise en deux sous-listes d'éléments $< / \geq$ qu'un *pivot* p
   - Faisable *en place* pour un tableau
   - *Cf.* DS2
2. Trie ces sous-listes (récursivement)
3. Construit le résultat en concaténant les deux sous-listes triées
   - Particulièrement intéressant pour une version tableau en place : rien à faire

Coût :

- Pire cas : $T(n) \leq T(n-1) + an = O(n^2)$
- *Espéré* (au sens probabilistique) : $T(n) = O(n \log n)$ (l'année prochaine ?)

Bestiaire 3

Exponentiation rapide

Pour calculer a^n on :

1. Calcule $a^{n/2}$
2. Met le résultat au carré, et le multiplie par $a^{n \mod 2}$

- Les deux sous-instances sont identiques (par symétrie)
- La recombinaison est triviale
- Pas un exemple très « riche »

Multiplication d'entiers/polynômes « de Karatsuba »

Cf. précédemment

Bestiaire 4

Multiplication de matrice « de Strassen »

Pour multiplier deux matrices (par ex. $n \times n$) on :

1. Découpe chaque matrice en 4 matrices bloc de tailles proches (par ex. $n/2 \times n/2$)
2. Calcule récursivement **7** produits de matrices sur les matrices bloc
   - L'algorithme naïf en demanderait 8
   - On descend à 7 grâce au même genre d'idées que dans l'algorithme de Karatsuba
3. Construit le résultat (en sommant les résultats blocs)

Coût (dimension n) : $T(n) \leq 7T(n/2) + an^2 = O(n^{\log_2(7)})$

- $\log_2(7) \approx 2.81 < 3$

Bestiaire 5

Distance minimale entre deux points dans le plan

En TP ?

Bilan diviser pour régner

- *Une* approche possible lorsque un problème se découpe « bien » en sous-problèmes de même type
  - Souvent en présence de listes, tableaux, graphes, arbres… ; d'objets de grande dimension (polynômes, matrices…) ; en géométrie algorithmique
- Pas garanti d'améliorer un algorithme naïf
- Analyse de coût : établir puis résoudre l'équation de récurrence