Complexité : résumé

I Définitions

Définition .

• La complexité **en temps** d'un algorithme **dans le pire des cas** est le plus grand nombre possible d'opérations élémentaires effectuées pour obtenir le résultat en fonction de la taille de la donnée entrée.

• La complexité **en temps** d'un algorithme **dans le meilleur des cas** est le plus petit nombre possible d'opérations élémentaires effectuées pour obtenir le résultat en fonction de la taille de la donnée entrée.

• La complexité **en temps** d'un algorithme **en moyenne** est le nombre **moyen** d'opérations élémentaires effectuées pour obtenir le résultat en fonction de la taille de la donnée entrée (on considère équiprobables toutes les données de taille n).

• La notion d'*opération élémentaire* dépend du contexte où elle doit être précisée. On dit qu'on a dif'érents *modèles de calcul*.

Définition .

La complexité **en espace** d'un algorithme **dans le pire des cas** est la plus grande taille possible de la mémoire utilisée pour obtenir le résultat en fonction de la taille de la donnée entrée.

La notion de *taille* doit être précisée dans le contexte. Définitions analogues pour le pire des cas et en moyenne.

Définition .

$u_n = \Theta(v_n) \Leftrightarrow u_n = O(v_n)$ et $v_n = O(u_n)$.

Définition .

Complexités c_n usuelles :

- Bornée : $c_n = O(1)$.
- Logarithmique : $c_n = \Theta(\lg(n))$.
- Linéaire : $c_n = \Theta(n)$.
- Quasi-linéaire : $c_n = \Theta(n \lg(n))$.
- Quadratique : $c_n = \Theta(n^2)$.
- Polynomiale : $\exists k \in \mathbb{N},\ c_n = \Theta(n^k)$.
- Exponentielle : $\lambda^n = O(c_n)$, $\lambda > 1$.

II Sommes et récurrences usuelles

Théorème (Sommation de O et de Θ).

- $\displaystyle\sum_{k=1}^{n} O(k^\alpha) = O(n^{\alpha+1})$ et $\displaystyle\sum_{k=1}^{n} \Theta(k^\alpha) = \Theta(n^{\alpha+1})$.
- $\displaystyle\sum_{k=1}^{n} O(k^\alpha \lg^\beta(k)) = O(n^{\alpha+1} \lg^\beta(n))$ et $\displaystyle\sum_{k=1}^{n} \Theta(k^\alpha \lg^\beta(k)) = \Theta(n^{\alpha+1} \lg^\beta(n))$.
- Pour $\lambda > 1$, $\displaystyle\sum_{k=1}^{n} O(\lambda^k) = O(\lambda^{n+1}) = O(\lambda^n)$.

Théorème (Suites récurrentes quasi-affines).

- La récurrence $c_n = c_{n-1} + \Theta(1)$ conduit à $c_n = \Theta(n)$.
- Pour $\alpha > 0$, la récurrence $c_n = c_{n-1} + \Theta\big(n^\alpha\big)$ conduit à $c_n = \Theta\big(n^{\alpha+1}\big)$.
- Pour $\lambda > 1$, la récurrence $c_n = \lambda c_{n-1} + \Theta\big(n^\alpha\big)$ conduit à $c_n = \Theta(\lambda^n)$.
- Si $X^2 - aX - b$ a deux racines $\lambda_1 < \lambda_2$ alors la récurrence $c_n = ac_{n-1} + bc_{n-2} + \gamma$ conduit à $c_n = \Theta(\lambda_2^n)$.

Résultats analogues avec des O...

Théorème (Stratégie "diviser pour régner").

- La récurrence $c_n = a\, c_{\lfloor \frac{n}{2} \rceil} + \Theta(n^\beta)$ conduit à
  (a) $c_n = \Theta(n^\alpha) = \Theta(a^{\lg(n)})$ si $\beta < \alpha = \lg(a)$.
  (b) $c_n = \Theta(n^\alpha \lg(n)) = \Theta(\lg(n) a^{\lg(n)})$ si $\beta = \alpha = \lg(a)$.
  (c) $c_n = \Theta(n^\beta)$ si $\beta > \alpha = \lg(a)$.
- Trois cas particuliers utiles :
  (a) La récurrence $c_n = c_{\lfloor \frac{n}{2} \rceil} + \Theta(1)$ conduit à $c_n = \Theta(\lg(n))$.
  (b) Les récurrences $c_n = 2c_{\lfloor \frac{n}{2} \rceil} + \Theta(1)$ ou $c_n = c_{\lfloor \frac{n}{2} \rfloor} + c_{\lceil \frac{n}{2} \rceil} + \Theta(1)$ conduisent à $c_n = \Theta(n)$.
  (c) Les récurrences $c_n = 2c_{\lfloor \frac{n}{2} \rceil} + \Theta(n)$ ou $c_n = c_{\lfloor \frac{n}{2} \rfloor} + c_{\lceil \frac{n}{2} \rceil} + \Theta(n)$ conduisent à $c_n = \Theta(n \lg(n))$.

Résultats analogues avec des O...