IFT2125 - Introduction à l'algorithmique

L'analyse des algorithmes (BB, chapitres 3 et 4)

Analysis of algorithms

Pierre McKenzie

DIRO, Université de Montréal

Automne 2017

Re-rappel : les ordres

Let
Soit $f : \mathbb{N} \to \mathbb{R}^{\geq 0}$.

The order (big-O) of f is
L'ordre de f est

$$O(f) = \{t : \mathbb{N} \to \mathbb{R}^{\geq 0} \mid (\exists c \in \mathbb{R}^{\geq 0})(\underbrace{\overset{\infty}{\forall}}_{\substack{\text{pour tous les } n \\ \text{suffisamment grands}}} n \in \mathbb{N})[t(n) \leq cf(n)]\}$$

for all n sufficiently large

The omega of f is
L'oméga de f est

$$\Omega(f) = \{t : \mathbb{N} \to \mathbb{R}^{\geq 0} \mid (\exists d \in \mathbb{R}^{+})(\overset{\infty}{\forall} n \in \mathbb{N})[t(n) \geq df(n)]\}$$

The exact order of f is
L'ordre exact de f est

$\Theta(f) = O(f) \cap \Omega(f)$.

Available tools (see footnote 1):

Outils disponibles[1] :

1. if $\lim_{n \to \infty} \frac{f(n)}{g(n)} \in \mathbb{R}^+$ then $f(n) \in \Theta(g(n))$,
2. if $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$ then $f(n) \in O(g(n))$ but $f(n) \notin \Theta(g(n))$, and
3. if $\lim_{n \to \infty} \frac{f(n)}{g(n)} = +\infty$ then $f(n) \in \Omega(g(n))$ but $f(n) \notin \Theta(g(n))$.

Passage extracted from BB, as well as any passage of my transparencies visibly from a book and not explicitly attributed.

1. Passage extrait de BB, de même que tout passage de mes transparents visiblement reproduit d'un livre et non explicitement attribué.

Notre approche pour l'analyse d'algos, en résumé

Our approach to the analysis of algorihms, in summary

- Analytique Analytical
- En pire cas In the worst case
- Souvent on comptera le nombre b d'exécutions d'une instruction baromètre Often we count the number b of executions of a barometer instruction
- On estimera $t(n)$ à l'ordre près, si possible à l'ordre exact-près
  We shall estimate t(n) to the order, if possible to the near exact order
  - une f qui $\overset{\infty}{\forall}$ n et sur tout exemplaire de taille n majore b de cet exemplaire vérifie $t(n) \in O(f(n))$
  - une g qui $\overset{\infty}{\forall}$ n et sur au moins un exemplaire de taille n minore b de cet exemplaire vérifie $t(n) \in \Omega(g(n))$
  - pourvu que $g \in \Omega(f)$ on conclut $t(n) \in \Theta(f(n))$

a f which ∀ n and on every copy of size n greater than b of this copy verifies t(n) ∈ O(f (n))

a g which ∀ n and on at least one copy of size n smaller than b of this copy verifies t (n) ∈ Ω (g (n))

provided that g ∈ Ω(f), we conclude t(n) ∈ Θ(f(n))

Algorithmique versus théorie de la complexité du calcul

Algorithms versus Computational Complexity Theory

Un problème P est donné.
A problem P is given.

Spirit of the algorithmic:
Ressort de l'algorithmique :

- développer un algorithme A efficace pour résoudre P
  develop an efficient algorithm A to solve P
- déterminer l'ordre exact du temps d'exécution de l'algorithme A.
  determine the exact order of the execution time of the algorithm A.

The complexity of computation:
Ressort de la complexité du calcul :

- borrow from the algorithm its best algorithm, A, for P
  emprunter à l'algorithmique son meilleur algorithme, A, pour P
- show that no algorithm does better than A, for P
  démontrer qu'aucun algorithme ne fait mieux que A, pour P
- conclure que la complexité du problème P est donnée par le temps d'exécution de A.
  conclude that the complexity of the problem P is given by the execution time of A.

Autres mesures possibles

Other possible measures

The memory used.
Can we always cut in memory?
Can we always do it in exchange for a longer time?

- La mémoire utilisée.
  Peut-on toujours couper dans la mémoire ?
  Peut-on toujours le faire en échange d'un temps plus long ?
- Le temps "parallèle".
  L'accès à m processeurs permet-il d'accélérer ?
  Idéalement : m fois plus rapide, ou mieux encore.

The "parallel" time.
Does access to m processors accelerate?
Ideally: m times faster, or better yet.

Supplement on orders (BB chapter 3)

Supplément sur les ordres (BB chapitre 3)

Ordre conditionnel

Conditional Order

The O, Ω and Θ have no more secrets (is not it?).

Les O, Ω et Θ n'ont plus de secrets (n'est-ce pas ?).

Conditional order of f (n) = notation which makes it possible to state a bound on f (n) which is only valid for some n

- Ordre conditionnel de $f(n)$ = notation qui permet d'énoncer une borne sur $f(n)$ qui ne vaut que pour certains n
- Couplé à une condition sur f,
  $t(n) \in$ ordre conditionnel de $f(n) \Rightarrow t(n) \in$ ordre inconditionnel
  Coupled with a condition on f,
  t (n) ∈ conditional order of f (n) ⇒ t (n) ∈ unconditional order

Exemple : $\Omega(f(n) \mid n$ est puissance de $2)$ versus $\Omega(f(n))$.

n is power of 2

Conditional Order

Ordre conditionnel

Définition formelle (cas de O, idem pour les autres)

Formal definition (case of O, the same for others)

Let and { right, wrong }

Soient $f : \mathbb{N} \to \mathbb{R}^{\geq 0}$ et $P : \mathbb{N} \to \{\text{vrai,faux}\}$.

then

Alors

$$O(f(n) \mid P(n))$$

and

est

$$\{t : \mathbb{N} \to \mathbb{R}^{\geq 0} \mid \exists c \in \mathbb{R}^{\geq 0}, \overset{\infty}{\forall} n \in \mathbb{N}, [\ P(n) \Longrightarrow t(n) \leq cf(n)\ ]\ \}.$$

Règle de l'harmonie ("smoothness" rule)

Cas de Θ (idem pour les autres)

Case of Θ (the same for others)

$b \in \mathbb{N}^{\geq 2}$

$t(n) \in \Theta(f(n) \mid n \text{ est puissance de } b)$.

n is power of 2

La règle de l'harmonie sert à éliminer "n est puissance de b" :

The smoothness rule is used to eliminate "n is power of b":

Si

If

- is — eventually non-decreasing

  $t(n)$ *est é.n.d. (éventuellement non décroissante), i.e.,*

  $\overset{\infty}{\forall} n \in \mathbb{N}, t(n) \leq t(n+1)$

- f(n) is smooth, i.e.

  $f(n)$ *est harmonieuse, i.e.,*

  $f(n)$ *est é.n.d et* $f(bn) \in O(f(n))$

  is — and

alors

then

- $t(n) \in \Theta(f(n))$.

Opérations sur les ordres

O(n2 + n3) has been defined, but does O(n2) + O(n3) have any meaning? No!

$O(n^2+n^3)$ a été défini, mais est-ce que $O(n^2)+O(n^3)$ a un sens ? Non !

It is therefore necessary to invent a definition. There she is :

Il faut donc inventer une définition. La voici :

$$O(f)+O(g)$$

is

est

$$\{\ h:\mathbb{N}\to\mathbb{R}^{\geq 0}\mid \exists h_1\in O(f),\exists h_2\in O(g),\ \overset{\infty}{\forall} n\in\mathbb{N},\ [h(n)=h_1(n)+h_2(n)]\ \}$$

Sometimes useful for estimating the time of a block A followed by a block B

Parfois utile pour estimer le temps d'un bloc A suivi d'un bloc B.

Extends to other operators, for example x instead of +, and to any pair of function sets, such as O (f) + Θ (g).

S'étend à d'autres opérateurs, par exemple $\times$ au lieu de $+$, et à n'importe quelle paire d'ensembles de fonctions, comme $O(f)+\Theta(g)$.

Supplement on recurrences

Supplément sur les récurrences (BB Section 4.7)

Méthode du pifomètre ("intelligent guesswork")

Intelligent guesswork method

Méthode : estimer à l'oeil la forme de la solution, puis prouver.

Method: estimate the shape of the solution by eye, then prove.

significant recurrence (here n power of b)

Ex : récurrence importante (ici n puissance de b)

$$T(b^k) = \begin{cases} c \neq 0 & \text{si } k = k_0 \\ aT(b^{k-1}) + f(b^k) & \text{si } k > k_0, \end{cases}$$

où $k_0 \in \mathbb{N}$, $a \in \mathbb{R}^{\geq 1}$, $b \in \mathbb{N}^{\geq 2}$, $f : \mathbb{N} \longrightarrow \mathbb{R}^{>0}$.

$$T(b^k) = \begin{cases} c \neq 0 & \text{si } k = k_0 \\ aT(b^{k-1}) + f(b^k) & \text{si } k > k_0, \end{cases}$$

Let and for

Posons $g(b^{k_0}) = c/(a^{k_0})$ et $g(b^k) = f(b^k)/(a^k)$ pour $k > k_0$.

Par induction sur $k \geq k_0$:

By induction on k ≥ k0

$$T(b^k) = a^k \times [g(b^{k_0}) + g(b^{k_0+1}) + \cdots + g(b^k)].$$

Already n is the power of b

Déjà $T(n) \in \Omega(\underbrace{n^{\log_b a}}_{(b^k)^{\log_b a} = a^k} \mid n$ est puissance de b).

To get better, we need to analyze [+ ... +] :

Pour obtenir mieux, il faut analyser $[+ \cdots +]$:

$$T(b^k) = \begin{cases} c \neq 0 & \text{si } k = k_0 \\ aT(b^{k-1}) + f(b^k) & \text{si } k > k_0. \end{cases}$$

Lemma ("powers of b")

Lemme (“des puissances de b”)

- If and then
  Si $\varepsilon > 0$ *et* $f(n) \in O(n^{\log_b a \, - \, \varepsilon})$ *alors*
  $T(n) \in \Theta(n^{\log_b a} \mid n \text{ est puissance de } b)$.
  n is power of b
- If and then
  Si $\varepsilon > 0$ *et* $f(n) \in O(n^{\log_b a \, + \, \varepsilon})$ *alors*
  $T(n) \in O(n^{\log_b a \, + \, \varepsilon} \mid n \text{ est puissance de } b)$.
  n is power of b
- If and then
  Si $\varepsilon \geq 0$ *et* $f(n) \in O(n^{\log_b a}(\log n)^{\varepsilon})$ *alors*
  $T(n) \in O(n^{\log_b a}(\log n)^{\varepsilon+1} \mid n \text{ est puissance de } b)$.
  n is power of b

Application à une récurrence asymptotique

Application to asymptotic recurrence

Let

of which we know only

Soit $t : \mathbb{N} \longrightarrow \mathbb{R}^{>0}$ dont on ne connaît que

$$t(n) \in a_1 t(\lceil n/b \rceil) + a_2 t(\lfloor n/b \rfloor) + O(f(n)),$$

where

où

- $f : \mathbb{N} \longrightarrow \mathbb{R}^{>0}$
- $a_1, a_2 \in \mathbb{R}^{\geq 0}$, $a_1 + a_2 \geq 1$
- $b \in \mathbb{N}^{\geq 2}$.

Le théorème qui suit résoud cette récurrence :

The following theorem solves this recurrence

Theorem (Solution of the asymptotic recurrence)

Théorème (Solution de la récurrence asymptotique)

Let

Posons $a = a_1 + a_2$.

1. if / and / and / then
   Si $\varepsilon > 0$ *et* $(\log_b a - \varepsilon) \geq 0$ *et* $f(n) \in O(n^{\log_b a \,-\, \varepsilon})$ *alors* $t(n) \in O(n^{\log_b a})$.
2. if / and / then
   Si $\varepsilon > 0$ *et* $f(n) \in O(n^{\log_b a \,+\, \varepsilon})$ *alors* $t(n) \in O(n^{\log_b a \,+\, \varepsilon})$.
3. *Si* $\varepsilon \geq 0$ *et* $f(n) \in O(n^{\log_b a}(\log n)^{\varepsilon})$ *alors* $t(n) \in O(n^{\log_b a}(\log n)^{\varepsilon+1})$.
   if / and / then

Also valid when

Vaut également lorsque

$$t(n) \in a_1 t(\lceil n/b \rceil) + a_2 t(\lfloor n/b \rfloor) + \Omega(f(n)),$$

et tous les "O" remplacés par des "Ω".

and all "O" replaced by "Ω"

Preuve (esquisse) du cas $f(n) \in O(n^{\log_b a - \varepsilon})$

Proof (sketch) of the case

$$t(n) \in a_1 t(\lceil n/b \rceil) + a_2 t(\lfloor n/b \rfloor) + O(n^{\log_b a - \varepsilon})$$

1. Choose c and k0 such that for all n ≥ b^{k0}
   Choisir c et k_0 tels que pour tout $n \geq b^{k_0}$,
   $t(n) \leq a_1 t(\lceil n/b \rceil) + a_2 t(\lfloor n/b \rfloor) + \underbrace{cn^{\log_b a - \varepsilon}}_{\text{note : non décroissante}}$
   note: not decreasing
2. Show that for any n, t(n) ≤ T(n) where
   Montrer que pour tout n, $t(n) \leq T(n)$ où
   $$T(n) = \begin{cases} \max\{t(0), t(1), \ldots, t(b^{k_0})\} & \text{si } n \leq b^{k_0} \\ a_1 T(\lceil n/b \rceil) + a_2 T(\lfloor n/b \rfloor) + cn^{\log_b a - \varepsilon} & \text{sinon.} \end{cases}$$
   if / otherwise
3. Lemme des puissances $\Rightarrow T(n) \in O(n^{\log_b a} \mid n \text{ est puissance de } b)$
   Lemma of Powers / n is power of b
4. Démo 3 $\Rightarrow T(n)$ est é.n.d.
   is e.n.d. (eventually non-decreasing)
5. $n^{\log_b a}$ est harmonieuse
   is smooth
6. Smoothness rule $\Rightarrow T(n) \in O(n^{\log_b a})$
7. Points (2) et (6) $\Rightarrow t(n) \in O(n^{\log_b a})$.

Méthode de l'équation caractéristique

Method of the characteristic equation

Déjà étudiée dans les cours préalables pour la résolution de récurrences linéaires homogènes à coefficients constants.

Already studied in prerequisite courses for the resolution of homogeneous linear recurrences with constant coefficients.

Plusieurs exemples dans BB.

Several examples in BB.

Résolution de récurrences

Linear homogeneous recurrences with constant coefficients:

Récurrences linéaires homogènes à coefficients constants:

Let the recurrence R

Soit la récurrence R

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = 0$$

Here are the steps of the resolution

Voici les étapes de la résolution:

Find the characteristic polynomial P of the recurrence R

1) Trouver le polynôme caractéristique $P(x)$ de la récurrence R

Finding the roots of P(x)

2) Trouver les racines de $P(x)$

If these roots are distinct

Si ces racines sont distinctes

The general solution is of the form tn

3) La solution générale est de la forme $t_n = \sum_{i=1}^{k} c_i r_i^n$

Solve the system of linear equations given by the initial conditions to find the value of the constants c1, c2, ..ck

4) Résoudre le système d'équations linéaires donné par les conditions initiales pour trouver la valeur des constantes $c_1, c_2, \ldots, c_k$

5) Écrire la solution t_n en fonction de ces constantes c_i

Write the solution tn according to these constants ci

Recurrence resolution

Résolution de récurrences

Linear homogeneous recurrences with constant coefficients:

Récurrences linéaires homogènes à coefficients constants:

Let the recurrence R

Soit la récurrence R

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = 0$$

Here are the steps of the resolution

Voici les étapes de la résolution:

Find the characteristic polynomial P of the recurrence R

1) Trouver le polynôme caractéristique $P(x)$ de la récurrence R

Finding the roots of P(x)

2) Trouver les racines de $P(x)$

If these roots are not all distinct

Si ces racines ne sont pas toutes distinctes

The general solution is of the form tn

3) La solution générale est de la forme $t_n = \sum_{i=1}^{\ell} \sum_{j=0}^{m_i - 1} c_{ij} n^j r_i^n$

where we have l roots r of multiplicity m

où on a ℓ racines r_i de multiplicité m_i

Solve the system of linear equations given by the initial conditions to find the value of the constants c1, c2, ..ck

4) Résoudre le système d'équations linéaires donné par les conditions initiales pour trouver la valeur des constantes $c_1, c_2, \ldots, c_k$

5) Écrire la solution t_n en fonction de ces constantes c_i

Write the solution tn according to these constants ci

Non-homogeneous case
Cas non homogène

Type of recurrence

Type de récurrence :

$$a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = b_1^n p_1(n) + b_2^n p_2(n) + \cdots$$

where ai, bj are constant

où a_i, b_j sont des constantes,

$p_i(n)$ sont des polynômes.

pi(n) are polynomials

Take as characteristic equation

Prendre comme équation caractéristique :

$$(a_0 x^k + a_1 x^{k-1} + \cdots + a_k)(x - b_1)^{1+\text{degré}(p_1)}(x - b_2)^{1+\text{degré}(p_2)} \cdots = 0.$$