INF4705 — Analyse et conception d'algorithmes

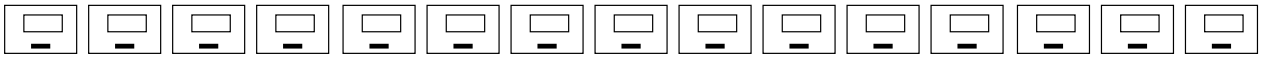
CHAPITRE 1

Prélude

- L'*algorithmique* est l'étude de la conception et de l'analyse des algorithmes. Étant donné un problème à résoudre, elle nous permet de décider quel algorithme utiliser, selon leur consommation respective des différentes ressources. Elle nous offre aussi des techniques générales pour concevoir de nouveaux algorithmes.
- Les premiers ordinateurs sont apparus dans les années '40, il y a une cinquantaine d'années ; et les premiers algorithmes alors ? Les premiers algorithmes remontent à l'Antiquité (p.e. Euclide, PGFC).
- De façon informelle, un algorithme est un ensemble de règles afin d'effectuer un calcul, sur une machine ou simplement manuellement.
- Origine du mot "algorithme" : *Muhammad ibn Musa al-Khuwârizmi*, un mathématicien perse du 9e siècle.
- Y a-t-il vraiment plusieurs façons de calculer la même chose ?

1.1 Premier exemple

> "On nous donne 15 boîtes de clous de longueur différente. Nous devons les ranger dans un classeur avec autant de tiroirs. Comment devrions-nous procéder afin de récupérer rapidement les clous dont nous avons besoin ?"



Essais :

naïf : On les range dans n'importe quel ordre puis on consulte les tiroirs de gauche à droite, arbitrairement.
- En pire cas, on consulte 15 tiroirs.
- En moyenne, on en consulte 8 (supposant toutes longueurs équiprobables) : $\frac{1}{15}\sum_{i=1}^{15} i = \frac{1}{15}\frac{15\cdot 16}{2} = 8$.
- Et si certains clous sont plus en demande ? C'est probablement une bonne idée de ranger ceux-ci dans les premiers tiroirs.

naïf probabiliste : Malheureusement pour ceux-ci, les clous qui demandent 15 consultations de tiroir sont toujours les mêmes ; existe-t-il une façon de faire plus démocratique ? Choisissons au hasard le tiroir qu'on consulte : si on est chanceux, on s'arrête là ; sinon, on biffe ce tiroir et on en choisit un autre (encore au hasard) qui ne soit pas biffé. (On aurait pu aussi tout simplement choisir au hasard le point de départ de notre recherche séquentielle.)
- Le nombre espéré de consultations de tiroir est 8 : $\frac{1}{15}\cdot 1 + \frac{14}{15}\cdot\frac{1}{14}\cdot 2 + \frac{14}{15}\cdot\frac{13}{14}\cdot\frac{1}{13}\cdot 3 + \ldots = \frac{1}{15}\sum_{i=1}^{15} i = 8$.
- C'est l'*effet Robin des Bois* : on vole de la rapidité aux "riches" pour en donner aux "pauvres" : le nombre de consultations dépend des choix aléatoires et varie d'une fois à l'autre pour une même longueur de clou.

pré-traitement : On les trie puis les range en ordre croissant de longueur (ça en vaut la peine seulement si on aura souvent à récupérer des clous). Si on consulte encore de gauche à droite, on a rien gagné. Comment faire mieux ? Peu importe quel tiroir nous consultons, nous savons que les tiroirs à gauche contiennent de plus petits clous et ceux à droite de plus grands. Consultons d'abord le tiroir du centre ...*fouille dichotomique*.

– En pire cas, on consulte 4 tiroirs.
– En moyenne, on en consulte $3.2\overline{6}$ (supposant toutes longueurs équiprobables) :
$\frac{1}{15} \cdot 1 + \frac{2}{15} \cdot 2 + \frac{4}{15} \cdot 3 + \frac{8}{15} \cdot 4 = \frac{49}{15} = 3.2\overline{6}$

parallèle : Et si nous avions un processeur (une personne) par tiroir ? Chacun consulte son tiroir et celui pour qui la longueur concorde le signale. On obtient la réponse en un temps équivalent à l'examen de 1 tiroir. La quantité de travail est cependant la même que celle de l'algorithme naïf en pire cas.

Et si nous utilisions trois processeurs ?

1.2 Second exemple

"Est-ce que l'un ou l'une d'entre vous sait multiplier deux entiers ?"

ALGORITHMES :

classique :

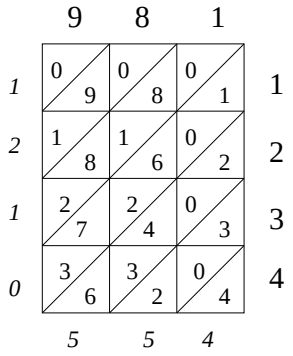
nord-américain

				9	8	1
			1	2	3	4
			3	9	2	4
		2	9	4	3	
	1	9	6	2		
	9	8	1			
1	2	1	0	5	5	4

anglais

				9	8	1
			1	2	3	4
	9	8	1			
	1	9	6	2		
		2	9	4	3	
			3	9	2	4
1	2	1	0	5	5	4

arabe :



Comment pourrait-on mesurer la quantité de travail (l'efficacité) de ces différents algorithmes ? On pourrait par exemple compter le nombre d'opérations arithmétiques nécessaires. Pour notre exemple, les deux versions de l'algorithme classique font 12 multiplications d'un chiffre avec un autre et 15 additions. L'algorithme arabe fait 12 multiplications et 20 additions. On voit que le nombre de multiplications requises (et aussi d'additions) est lié au produit du nombre de chiffres dans chacun des deux entiers : en général s'ils sont de taille m et n respectivement, nous prévoyons faire $m \times n$ multiplications.

à la russe : (déjà connu des égyptiens de l'Antiquité)

981	**1234**
490	2468
245	4936
122	9872
61	19744
30	39488
15	78976
7	157952
3	315904
1	631808
	1210554

Pourquoi fonctionne-t-il ?

Quel est l'avantage d'un tel algorithme ? On a pas besoin de mémoriser des tables de multiplication : il suffit de savoir additionner et diviser par 2. Il ressemble donc à l'algorithme implanté dans les microprocesseurs des ordinateurs binaires.

diviser-pour-régner : Pour cet algorithme, les deux entiers doivent contenir le même nombre de chiffres et, de plus, ce nombre doit être une puissance de deux. Il suffit d'ajouter au besoin des zéros à gauche : pour notre exemple, 0981 et 1234. On remplace la multiplication d'entiers à 4 chiffres par quatre multiplications d'entiers à 2 chiffres. À leur tour, nous pouvons remplacer chacune par quatre multiplications d'entiers à 1 chiffre (récursivement).

0981×1234

multiplier		décalage	résultat
09	12	4	108
09	34	2	306
81	12	2	972
81	34	0	2754
			1210554

09×12

multiplier		décalage	résultat
0	1	2	0
0	2	1	0
9	1	1	9
9	2	0	18
			108

On compte ici $4 \times 4 = 16$ multiplications, ce qui n'est pas très prometteur. Par contre, on peut faire mieux grâce à l'astuce suivante. Soient $w = 09, x = 81, y = 12, z = 34$:

$$\begin{aligned} 981 \times 1234 &= (10^2 w + x) \times (10^2 y + z) \\ &= 10^4 wy + 10^2(wz + xy) + xz \\ \text{mais } (w + x) \times (y + z) &= wy + (wz + xy) + xz \text{ , appelons ce produit } r \\ \text{soient } p &= wy \\ \text{et } q &= xz \\ \text{alors } \quad 981 \times 1234 &= 10^4 p + 10^2(r - p - q) + q \end{aligned}$$

On n'a plus besoin que de trois multiplications (p, q et r). En appliquant cette astuce récursivement, le nombre de multiplications requises est "de l'ordre de" $n \times m^{0.59}$, où $m \leq n$. Cet algorithme est plus rapide que tous les précédents, à condition que les entiers soient suffisamment grands.

1.3 Notions de base

1.3.1 problème, exemplaire, algorithme, ressource

- La multiplication de deux entiers est un exemple de *problème*; la multiplication de 981 par 1234 est un *exemplaire* de ce problème.
- Un problème admet habituellement une infinité d'exemplaires.
- La *taille* d'un exemplaire est, formellement, le nombre de bits requis pour le représenter de façon raisonnable dans un ordinateur binaire. Néanmoins, nous verrons que nous pouvons relâcher un peu cette définition et parler plutôt du nombre de composantes de l'exemplaire (p.e. nombre de chiffres pour la multiplication de deux entiers, nombre de tiroirs).
- Un *algorithme* est une suite finie d'instructions précises. Un algorithme qui résout un certain problème doit fonctionner correctement pour tous ses exemplaires. L'algorithme qui retourne simplement "1210554" fonctionne correctement pour l'exemplaire $\langle 981, 1234 \rangle$ mais n'est certainement pas un algorithme pour le problème général de la multiplication d'entiers.
- L'ensemble des exemplaires d'un problème est identifié par son *domaine de définition*. À strictement parler, les algorithmes de multiplication que nous avons vu fonctionnent pour les exemplaires $\langle x, y \rangle$ où $x, y \in \mathbb{N}$; ils ne fonctionnent pas tels quels pour des nombres négatifs ou fractionnaires.
- Les principales *ressources* d'intérêt en algorithmique sont le temps de calcul et l'espace mémoire. Lorsqu'on s'intéresse au calcul parallèle, le nombre de processeurs est également une ressource d'intérêt. Le bit s'impose comme unité de mesure de l'espace mémoire : c'est uniforme sur tous les ordinateurs. Pour le nombre de processeurs, la question ne se pose pas. Mais pour le temps d'exécution ? Une seconde de Pentium 4 équivaut-elle à une seconde de 8086 ? Le *principe d'invariance* dit que deux implantations d'un même algorithme diffèrent en efficacité d'au plus une constante multiplicative. (Si on utilise un ordinateur 100 fois plus rapide, cette constante sera 100.) Il n'y aura donc pas d'unité particulière et nous parlerons plutôt de temps d'exécution à une constante multiplicative près.

Est-ce révélateur de mesurer à une constante multiplicative près ?

1.3.2 progrès technologique vs progrès algorithmique

- 2 algorithmes, (A) $10^{-4} \times 2^n$ secondes et (B) $10^{-2} \times n^3$ secondes
- Comparez (A) et (B) pour des exemplaires de taille 10, 20, 30
- Avec un ordinateur 100 fois plus rapide : (A) $10^{-6} \times 2^n$ et (B) $10^{-4} \times n^3$
- Pour une durée fixe : (A) $n \to n + \lg 100 \approx n + 7$; (B) $n \to 100^{1/3} n \approx 5n$
- Gare au pied de la courbe ! Pour des exemplaires de petite taille, (A) est supérieur à (B)

Rencontre-t-on vraiment des progrès algorithmiques si significatifs ?

1.3.3 quelques exemples

déterminant d'une matrice $n \times n$:

- l'application de la définition récursive : $n!$
- l'élimination de Gauss-Jordan : n^3

tri de n éléments :

- par sélection, par insertion, à bulles : n^2
- par fusion, par monceau : $n \lg n$
- par dénombrement, à l'application restreinte (utilise un tableau de compteurs) : n, mais la quantité requise d'espace mémoire est proportionnelle à la valeur du plus grand entier à trier.

plus grand facteur commun de m et n ($m < n$) :

- l'application de la définition : m
  examen systématique des entiers $\leq m$ en ordre descendant jusqu'à ce qu'on en trouve un qui divise m et n exactement
- factorisons m et n puis prenons le produit de leur facteurs communs : la factorisation de grands entiers est très difficile !
- l'algorithme d'Euclide : $\lg n$

```
fonction Euclide(m,n)
tant que m > 0 faire
  t ← m
  m ← n modulo m
  n ← t
retourner n
```

transformée de Fourier : a une foule d'applications, entre autres la multiplication de polynômes. La découverte d'un algorithme "rapide" a eu un impact colossal.