Pourquoi enseigner l'algorithmique ?

Eric Reyssat

Université de Caen Laboratoire de Mathématiques Nicolas Oresme

IREM de Basse Normandie

Résumé

Je montrerai à travers quelques aspects de l'algorithmique et exemples d'algorithmes ce que leur enseignement peut apporter dans la formation scientifique. L'exposé sera construit autour d'une définition commentée de ce qu'est un algorithme. Ce sera l'occasion aussi de réfléchir à certaines difficultés de cet enseignement.

Le texte présenté ci dessous reprend les diapositives de l'exposé réalisé lors du colloque.

Pourquoi enseigner l'algorithmique ?

- provocateur : c'est au programme du lycée
- semi-provocateur : faire faire le travail par les autres
- plus motivant de mon point de vue : former à l'analyse de problèmes, à la décision dans les choix, la précision du langage, et accessoirement concrétiser les objets ou situations mathématiques , comprendre ou fabriquer des programmes informatiques.

Définition

Définition

Un **algorithme** est une suite finie ordonnée d'actions élémentaires pour accomplir une tâche fixée à partir des données prévues en entrée.

Définition

Définition

Un **algorithme** est une suite finie ordonnée d'actions élémentaires pour accomplir une tâche fixée à partir des données prévues en entrée.

[début des affichages] $\texttt{affiche}([1,1].\mathit{sqrt}(1^2+1^2))$;
$\texttt{affiche}([1,2].\mathit{sqrt}(1^2+2^2))$;
$\texttt{affiche}([1,3].\mathit{sqrt}(1^2+3^2))$;
$\texttt{affiche}([1,4].\mathit{sqrt}(1^2+4^2))$;
$\texttt{affiche}([2,2].\mathit{sqrt}(2^2+2^2))$;
$\texttt{affiche}([2,3].\mathit{sqrt}(2^2+3^2))$;
$\texttt{affiche}([2,4].\mathit{sqrt}(2^2+4^2))$;
$\texttt{affiche}([3,3].\mathit{sqrt}(3^2+3^2))$;
$\texttt{affiche}([3,4].\mathit{sqrt}(3^2+4^2))$;
[fin des affichages] $\texttt{affiche}([4,4].\mathit{sqrt}(4^2+4^2))$;

Suite finie

Définition

Un **algorithme** est une **suite finie** ordonnée d'actions élémentaires pour accomplir une tâche fixée à partir des données prévues en entrée.

Suite finie

Définition

Un **algorithme** est une **suite finie** ordonnée d'actions élémentaires pour accomplir une tâche fixée à partir des données prévues en entrée.

Algorithme factorielle

entrée : un entier $n \geq 1$ **sortie :** l'entier $f = n!$

[initialisation] $f \leftarrow 1$

[une boucle] **pour** i **de** 1 **à** n **faire**

$f \leftarrow f * i$

fin(pour)

[retour de la valeur] f

La boucle compte pour autant d'actions qu'il y a de tours de boucle : dépend de n.

Suite finie : complexité

Définition

Un **algorithme** est une **suite finie** ordonnée d'actions élémentaires pour accomplir une tâche fixée à partir des données prévues en entrée.

Conséquences :

- Prouver la finitude (indécidable, souvent facile, pas toujours !)
- Complexité

Suite finie : complexité

Problème de Syracuse

entrée : un entier n **sortie :** ????

$a \leftarrow n$
tant que $a > 1$ **faire**
  si a est pair **alors**
    $a \leftarrow a/2$
  sinon
    $a \leftarrow 3a + 1$
  fin(si)
fin(tantque)

$42 \rightarrow 21 \rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$

Suite finie : complexité

Problème de Syracuse

entrée : un entier n **sortie :** ????

$a \leftarrow n$
tant que $a > 1$ **faire**
  si a est pair **alors**
    $a \leftarrow a/2$
  sinon
    $a \leftarrow 3a + 1$
  fin(si)
fin(tantque)

$42 \rightarrow 21 \rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$

$7 \rightarrow 22 \rightarrow 11 \rightarrow 34 \rightarrow 17 \rightarrow 52 \rightarrow 26 \rightarrow 13 \rightarrow 40 \rightarrow 20 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$

Suite finie : boucles

pour
...
fin(pour)

différence essentielle : pb de finitude

tant que
...
fin(tantque)

différence technique minime

répéter
...
jusqu'à

Suite finie : complexité

Vous êtes né l'année a et avez commencé à travailler l'année t :

Algorithme

[boucle jeunesse] **pour** i **de** a **à** $t-1$ **faire**
    `s'épanouir en apprenant`
[fin de boucle] **fin(pour)**
[boucle active] **pour** i **de** t **à** $mystère(a,t)$ **faire**
    `travailler`
    `cotiser`
[fin de boucle] **fin(pour)**
[enfin ou déjà ?] `Retraite`

Complexité ?

Introduction Définition Suite finie Actions élémentaires Suite ordonnée Tâche Accomplir Données d'entrée

Suite finie : complexité

Vous êtes né l'année a et avez commencé à travailler l'année t :

Algorithme

[boucle jeunesse] **pour** i **de** a **à** $t-1$ **faire**
    `s'épanouir en apprenant`
[fin de boucle] **fin(pour)**
[boucle active] **pour** i **de** t **à** $mystère(a,t)$ **faire**
    `travailler`
    `cotiser`
[fin de boucle] **fin(pour)**
[enfin ou déjà ?] `Retraite`

Complexité ?
Y a-t-il un moyen plus rapide d'arriver au même résultat ?

Complexité

Complexité d'un algorithme

$C(n) =$

nombre d'opérations
au pire
sur données de taille n

Complexité

- $O(\log n)$: logarithmique ☺☺
- $O(n)$: linéaire
- $O(n^2)$: quadratique ☹
- $O(n^k)$: polynomiale ☺☺
- $O(a^n)$: exponentielle ↯

Mauvaises nouvelles :
$\Omega(n^2), \Omega(2^n), \ldots$

Complexité

Maxime

Avec une horloge à 1GHz , la notion de complexité asymptotique devient plus importante qu'à 10MHz : les constantes cachées dans les O perdent de l'importance.

Exercice

Entre ces algorithmes dont les complexités ont été estimées (ou admises), préciser le meilleur choix en fonction de la taille des données

Complexité

Exemple

Animation à 2 GHz de 50 objets définis comme enveloppes convexes de n points :

	$n = 50$	$n = 500$
$20n \log n$	0.1 ms	1ms
$3n^2$	0.2 ms	10ms
n^4	0.1 s	30 min

Tracer les courbes, où se croisent-elles, ...

Complexité

Intérêt pédagogique

- Choisir entre plusieurs méthodes pour résoudre un problème, la complexité est l'un des arguments
- Gagner en bon sens sur les ordres de grandeur :

Actions élémentaires

Définition

Un **algorithme** est une suite finie ordonnée **d'actions élémentaires** pour accomplir une tâche fixée à partir des données prévues en entrée.

- Actions indivisibles
- Actions compréhensibles et maîtrisées par l'exécutant

Actions élémentaires

Exercice :

Algorithme

```
blanchir les os
ajouter un bouquet garni
glacer à blanc des petits oignons
préparer un roux blanc
...
```

Quel est le résultat de cet algorithme ?

Actions élémentaires

Algorithmes d'Eratosthène

entrée : un entier $n \geq 2$ **sortie :** la liste des premiers jusque n.

```
[initialiser] p_0, ..., p_n ← vrai
pour i de 2 à n faire
  si p_i alors
    barrer les multiples de i
  fin(si)
fin(pour)
récolte les p_i vrais dans une liste L
```

```
[initialiser p] pour i de 2 à n faire p_i ← vrai fin(pour)
pour i de 2 à n faire
  si p_i alors
    k ← 2
[boucle tq]   tant que ki ≤ n faire
[multiples de i]   p_ki ← faux
      k ← k+1
    fin(tantque)
  fin(si)
fin(pour)
[initialiser l] L ← [ ]
pour i de 2 à n faire
[premiers premiers]   si p_i alors L ← L, i fin(si)
fin(pour)
[sortie] L
```

Le premier algorithme est plus lisible mais masque la double boucle.

Actions élémentaires : structures de données

Structures de données : tables, listes, piles, ...
avec opérations (élémentaires "par définition") : extraire un élément, le premier, le dernier, etc...

D'où représentation d'objets : chiffres d'un nombre $[a_0, \ldots, a_n]$, point $[x, y]$, polynôme $[a_0, \ldots, a_n]$, droite $[a, b, c]$, etc.

Exercice

- Écrire un algorithme qui à partir des chiffres de deux entiers $[a_0, \ldots, a_n]$ $[b_0, \ldots, b_m]$ trouve les chiffres de la somme
- Écrire un algorithme qui à partir deux points $[x, y], [x', y']$ trouve une équation $[a, b, c]$ de la droite qui les joint.
- Écrire un algorithme qui à partir d'un polynôme $[a_0, \ldots, a_n]$ calcule son carré.

Actions élémentaires : pédagogie

Intérêt pédagogique

- Savoir communiquer (= [se faire] comprendre) avec
  - une machine (précision)
  - un haut dignitaire (respect des formules protocolaires)
  - un étranger (langage)
  - un enfant (vocabulaire restreint)
  - un employé pressé (résumer ses demandes)
  - ... ou niais (détailler ses demandes)
- Tester sa maîtrise de certaines notions : révision par la pratique

Suite ordonnée

Définition

Un **algorithme** est une **suite** finie **ordonnée** d'actions élémentaires pour accomplir une tâche fixée à partir des données prévues en entrée.

Suite ordonnée

Exercice

Algorithme ou pas ?

$a \leftarrow 1$	$a \leftarrow 1$	$a \leftarrow 1$
$b \leftarrow 1$	$b \leftarrow 1$	$b \leftarrow 1$
$a \leftarrow -ab$	$b \leftarrow -ab$	$a \leftarrow -ab$
$b \leftarrow -ab$	$a \leftarrow -ab$	$b \leftarrow -ab$
tant que $a < 10$ `faire`	**tant que** $a < 10$ `faire`	**tant que** $a < 10$ `faire`
$a \leftarrow 2a$	$a \leftarrow 2a$	$a \leftarrow 2a$
fin(tantque)	**fin(tantque)**	**fin(tantque)**

Suite ordonnée

Exercice

Même résultat ?

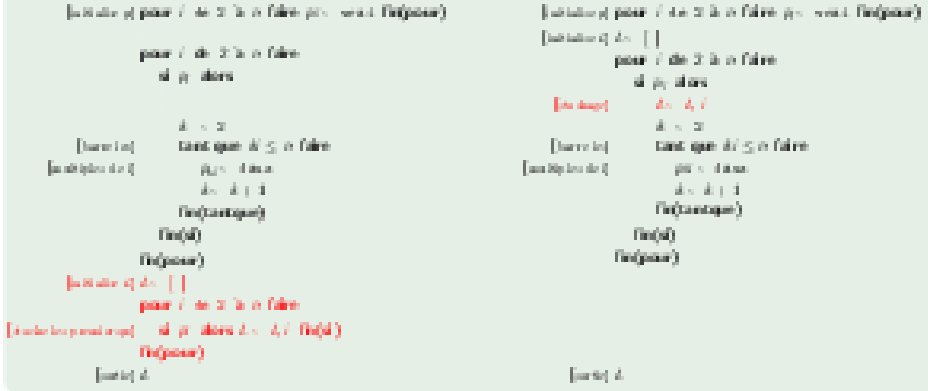
a ← 3	a ← 3	a ← 3
b ← 4	b ← 4	b ← 4
a ← a+b	b ← a+b	b ← a+b
b ← a+b	a ← a+b	a ← a+b
écrire(a,b)	écrire(a,b)	écrire(b,a)

version interactive (wims)

Suite ordonnée

Algorithmes d'Eratosthène

entrée : un entier $n \geq 2$ **sortie :** la liste des premiers jusque n.



Même résultat ?

Suite ordonnée

Algorithme

```
si tu es poli alors
   mouche ton nez
   dis bonjour à la dame
sinon
   dis bonjour à la dame
   mouche ton nez
fin(si)
```

Suite ordonnée

"algorithme" non déterministe

entrée : un entier n produit de deux nombres premiers
sortie : la factorisation (prouvée) de n.

$a \leftarrow$ un facteur premier de n
$b \leftarrow n/a$
écrire comme à l'école
la multiplication de a par b

Introduction Définition Suite finie Actions élémentaires Suite ordonnée Tâche Accomplir Données d'entrée

Suite ordonnée

$P = NP$?

Suite ordonnée

factorielle itérative

```
                        procédure factorielle(n)
[initialisation]        f ← 1
[une boucle]            pour i de 1 à n faire
                            f ← f * i
                        fin(pour)
[retour de la valeur]   f
```

factorielle récursive

```
procédure factorielle(n)
si n = 1 alors
    1
sinon
    n factorielle(n − 1)
fin(si)
```


Suite ordonnée

Suite ordonnée : intérêt pédagogique

- Comprendre la dépendance logique des idées, approcher la notion de preuve.
- Structurer sa pensée
- Approcher la notion de récursivité
- Nouveau critère de choix
- Apprendre à gérer

Tâche

Définition

Un **algorithme** est une suite finie ordonnée d'actions élémentaires pour accomplir **une tâche** fixée à partir des données prévues en entrée.

Tâche

- Plusieurs (ou aucun) algorithmes pour une tâche : deux notions de fonction.
- Complexité d'un problème
- Méthodes algorithmiques générales (diviser pour régner, glouton, retour arrière)
- Définition précise de la tâche.

Tâche : intérêt pédagogique

Tâche : intérêt pédagogique

- Ne pas confondre la fin et les moyens.
- Le choix entre algorithmes force l'expérimentation.
- Apprendre sur l'exemple à focaliser sur l'essentiel.
- Penser en terme de méthodes générales
- Décrire précisément un objectif

Accomplir

Euclide : pgcd(x,y)

```
[initialisation] a ← x
                 b ← y
[boucle]         tant que b > 0 faire
                   r ← a − b
                   a ← r
                   si a < b alors
[échange]            r ← a
                     a ← b
                     b ← r
                   fin(si)
                 fin(tantque)
[retour]         a
```

Invariant : $pgcd(a, b) = pgcd(x, y)$.
Donc à la fin $a = pgcd(a, 0) = pgcd(x, y)$.

Accomplir

Définition

Un **algorithme** est une suite finie ordonnée d'actions élémentaires pour **accomplir** une tâche fixée à partir des données prévues en entrée.

Introduction Définition Suite finie Actions élémentaires Suite ordonnée Tâche Accomplir Données d'entrée

Accomplir

Définition

Un **algorithme** est une suite finie ordonnée d'actions élémentaires pour **accomplir** une tâche fixée à partir des données prévues en entrée.

Accomplir : intérêt pédagogique

- Apprendre à se méfier des machines
- Expérimenter par batterie de tests

Accomplir

Définition

Un **algorithme** est une suite finie ordonnée d'actions élémentaires pour **accomplir** une tâche fixée à partir des données prévues en entrée.

Accomplir : intérêt pédagogique

- Apprendre à se méfier des machines
- Expérimenter par batterie de tests

Données d'entrée

Définition

Un **algorithme** est une suite finie ordonnée d'actions élémentaires pour accomplir une tâche fixée à partir des **données prévues en entrée**.

Données d'entrée

- Doivent être précisées : a-t-on envisagé les cas suivants pgcd(3), pgcd(3,-4), pgcd(3,4,5), pgcd(3,π), pgcd(chien,chat)
- Récursivité
- Type

Données d'entrée : intérêt pédagogique

Données d'entrée : intérêt pédagogique

- Prendre conscience de la nature des objets et savoir à quels objets les notions s'appliquent