

Quelques algorithmes simples dont l'analyse n'est pas si simple

Michel Habib habib@liafa.jussieu.fr
<http://www.liafa.jussieu.fr/~habib>

Algorithmique Avancée M1 Bioinformatique, Octobre 2008

Plan

Histoire d'une grenouille

Le crêpier maladroit

Des algorithmes

Tri par retournements

Les fonctions de Collatz

La fonction très réursive Tak

La pauvre grenouille

- ▶ Une grenouille tombe au fond d'un puits de 50 mètres de profondeur.

La pauvre grenouille

- ▶ Une grenouille tombe au fond d'un puits de 50 mètres de profondeur.
- ▶ Pour s'en sortir chaque jour elle monte de 3 mètres, mais chaque nuit elle glisse de 2 mètres

La pauvre grenouille

- ▶ Une grenouille tombe au fond d'un puits de 50 mètres de profondeur.
- ▶ Pour s'en sortir chaque jour elle monte de 3 mètres, mais chaque nuit elle glisse de 2 mètres
- ▶ Au bout de combien de jours la grenouille va-t-elle sortir du puits ?

Réponses possibles :

- ▶ 50 jours ?

Réponses possibles :

- ▶ 50 jours ?
- ▶ ou plutôt 48 jours

Histoire d'une grenouille

Le crêpier maladroit

Des algorithmes

Tri par retournements

Les fonctions de Collatz

La fonction très réursive Tak

Le crêpier débutant

- ▶ Un crêpier débutant n'a pas réussi à faire des crêpes de même taille, et il se retrouve devant un tas de crêpes de tailles différentes.

Le crêpier débutant

- ▶ Un crêpier débutant n'a pas réussi à faire des crêpes de même taille, et il se retrouve devant un tas de crêpes de tailles différentes.
- ▶ Il veut les ranger en ordre de taille décroissante, les plus petites en haut.

Le crêpier débutant

- ▶ Un crêpier débutant n'a pas réussi à faire des crêpes de même taille, et il se retrouve devant un tas de crêpes de tailles différentes.
- ▶ Il veut les ranger en ordre de taille décroissante, les plus petites en haut.
- ▶ Pour ce faire il ne dispose que d'une opération : retourner le haut du tas de crêpes en insérant son palet en bois entre deux crêpes.

Comment aider le crêpier ?

1. Est-ce toujours possible ?

Comment aider le crêpier ?

1. Est-ce toujours possible ?
2. Comment le faire avec le moins de mouvements possibles ?

- ▶ Avec $2n - 1$ opérations c'est possible !

- ▶ Avec $2n - 1$ opérations c'est possible !
- ▶ Gates et Papadimitriou on montré :
$$\frac{17}{16}n \leq \textit{Optimum} \leq \frac{5}{3}n$$

- ▶ Avec $2n - 1$ opérations c'est possible !
- ▶ Gates et Papadimitriou on montré :
$$\frac{17}{16}n \leq \textit{Optimum} \leq \frac{5}{3}n$$
- ▶ Et si chaque crêpe a une face brûlée, est-ce toujours possible ?

Histoire d'une grenouille

Le crêpier maladroit

Des algorithmes

Tri par retournements

Les fonctions de Collatz

La fonction très récursive Tak

Algorithmes

Définition

Ensemble de règles opératoires dont l'application permet de résoudre un problème au moyen d'un nombre fini d'opérations.

Algorithmes

Définition

Ensemble de règles opératoires dont l'application permet de résoudre un problème au moyen d'un nombre fini d'opérations.

Origine de ce mot

Abu Ja'far Mohamed Ibn Al-Khowârizmi

(Père de Ja'far, fils de Mohamed, né à Khowârizimi)

Mathématicien persan IX ème siècle qui a écrit le premier recueil d'algorithmes.

Quelques algorithmes simples dont l'analyse n'est pas si simple

└ Des algorithmes

Surtout ne pas écrire Algor^ythme!!!

Histoire d'une grenouille

Le crêpier maladroit

Des algorithmes

Tri par retournements

Les fonctions de Collatz

La fonction très réursive Tak

Un algorithme très simple de V. Chvátal

Données : σ une permutation de $[1, n]$

Résultat : une permutation σ' telle que $\sigma'(1) = 1$

tant que $\sigma(1) \neq 1$ **faire**

 Renverser l'ordre des $\sigma(1)$ premiers éléments

fin

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

▶ 5 2 7 1 3 6 4

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

▶ 5 2 7 1 3 6 4

▶ 3 1 7 2 5 6 4

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

▶ 5 2 7 1 3 6 4

▶ 3 1 7 2 5 6 4

▶ 7 1 3 2 5 6 4

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

▶ 5 2 7 1 3 6 4

▶ 3 1 7 2 5 6 4

▶ 7 1 3 2 5 6 4

▶ 4 6 5 2 3 1 7

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

▶ 5 2 7 1 3 6 4

▶ 3 1 7 2 5 6 4

▶ 7 1 3 2 5 6 4

▶ 4 6 5 2 3 1 7

▶ 2 5 6 4 3 1 7

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

▶ 5 2 7 1 3 6 4

▶ 3 1 7 2 5 6 4

▶ 7 1 3 2 5 6 4

▶ 4 6 5 2 3 1 7

▶ 2 5 6 4 3 1 7

▶ 5 2 6 4 3 1 7

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

▶ 5 2 7 1 3 6 4

▶ 3 1 7 2 5 6 4

▶ 7 1 3 2 5 6 4

▶ 4 6 5 2 3 1 7

▶ 2 5 6 4 3 1 7

▶ 5 2 6 4 3 1 7

▶ 3 4 6 2 5 1 7

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

- ▶ 6 3 1 7 2 5 4
- ▶ 5 2 7 1 3 6 4
- ▶ 3 1 7 2 5 6 4
- ▶ 7 1 3 2 5 6 4
- ▶ 4 6 5 2 3 1 7
- ▶ 2 5 6 4 3 1 7
- ▶ 5 2 6 4 3 1 7
- ▶ 3 4 6 2 5 1 7
- ▶ 6 4 3 2 5 1 7

Un algorithme très simple de V. Chvátal

Considérons σ une permutation de $[1,7]$

▶ 6 3 1 7 2 5 4

▶ 5 2 7 1 3 6 4

▶ 3 1 7 2 5 6 4

▶ 7 1 3 2 5 6 4

▶ 4 6 5 2 3 1 7

▶ 2 5 6 4 3 1 7

▶ 5 2 6 4 3 1 7

▶ 3 4 6 2 5 1 7

▶ 6 4 3 2 5 1 7

▶ 1 5 2 3 4 6 7

OUF!

À vous de faire sur la permutation :

3 1 5 7 4 6 2

Combien de retournements ?

Conséquences :

Cela doit toujours s'arrêter !!

Un algorithme très simple de V. Chvátal

Terminaison

On peut montrer que l'algorithme précédent termine en un nombre fini d'étapes.

Ebauche de preuve

▶ 6 3 1 7 2 5 4

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

▶ 5 2 7 1 3 6 4

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

▶ 5 2 7 1 3 6 4

▶ 0 1 0 0 0 0 0

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

▶ 5 2 7 1 3 6 4

▶ 0 1 0 0 0 0 0

▶ 3 1 7 2 5 6 4

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

▶ 5 2 7 1 3 6 4

▶ 0 1 0 0 0 0 0

▶ 3 1 7 2 5 6 4

▶ 0 0 0 0 1 1 0

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

▶ 5 2 7 1 3 6 4

▶ 0 1 0 0 0 0 0

▶ 3 1 7 2 5 6 4

▶ 0 0 0 0 1 1 0

▶ 7 1 3 2 5 6 4

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

▶ 5 2 7 1 3 6 4

▶ 0 1 0 0 0 0 0

▶ 3 1 7 2 5 6 4

▶ 0 0 0 0 1 1 0

▶ 7 1 3 2 5 6 4

▶ 0 0 1 0 1 1 0

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

▶ 5 2 7 1 3 6 4

▶ 0 1 0 0 0 0 0

▶ 3 1 7 2 5 6 4

▶ 0 0 0 0 1 1 0

▶ 7 1 3 2 5 6 4

▶ 0 0 1 0 1 1 0

▶ 4 6 5 2 3 1 7

Ebauche de preuve

▶ 6 3 1 7 2 5 4

▶ 0 0 0 0 0 0 0

▶ 5 2 7 1 3 6 4

▶ 0 1 0 0 0 0 0

▶ 3 1 7 2 5 6 4

▶ 0 0 0 0 1 1 0

▶ 7 1 3 2 5 6 4

▶ 0 0 1 0 1 1 0

▶ 4 6 5 2 3 1 7

▶ 0 0 0 0 0 0 1 ...

Complexité

On peut borner par $O(2^n)$ on considérant que l'on va de

0 0 0 0 0 0 0

à

1 1 1 1 1 1 1

Complexité

On peut borner par $O(2^n)$ on considérant que l'on va de

0 0 0 0 0 0 0

à

1 1 1 1 1 1 1

Conjecture de Chvátal

L'algorithme est en $O(n^2)$?

Des exemples difficiles :

6 2 1 10 11 8 12 3 4 7 9 5
nécessite 172 retournements

Des exemples difficiles :

6 2 1 10 11 8 12 3 4 7 9 5
nécessite 172 retournements

2 8 11 12 4 1 10 9 3 6 7 5
nécessite 171 retournements

Des exemples difficiles :

6 2 1 10 11 8 12 3 4 7 9 5
nécessite 172 retournements

2 8 11 12 4 1 10 9 3 6 7 5
nécessite 171 retournements

Les plus mauvais cas avec 12 éléments

Moralité

- ▶ Il n'est pas si facile d'analyser la complexité (le comportement) d'un algorithme possédant une seule instruction.
- ▶ Ces opérations de retournement interviennent **dans la nature** sur les génomes et sont étudiées en bioinformatique (tri par inversion de permutations).

Tout ce que l'on sait à ce jour

- ▶ Il existe un exemple en $O(n^2)$
- ▶ En moyenne c'est $O(n)$

Histoire d'une grenouille

Le crêpier maladroit

Des algorithmes

Tri par retournements

Les fonctions de Collatz

La fonction très réursive Tak

Prenez un entier positif ; s'il est pair, divisez-le par 2 ; s'il est impair, multipliez-le par 3 et ajoutez lui 1 ?
Réitérez ce processus sur plusieurs exemples
que semble-t-il se passer ?

Prenez un entier positif ; s'il est pair, divisez-le par 2 ; s'il est impair, multipliez-le par 3 et ajoutez lui 1 ?
Réitérez ce processus sur plusieurs exemples
que semble-t-il se passer ?

Partons de l'entier 7, et regardons la suite alors construite : 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 4, 2, 1, 4, 2, 1.

Dès qu'on tombe sur 1 le calcul s'arrête.

Essayons avec les entiers : 8, 9, 10

Le problème de Syracuse :

```
f(n) : si  $n = 1$  alors  
    retourner 1  
sinon  
    si n est pair alors  
        retourner  $f(n/2)$   
    sinon  
        retourner  $f(3n + 1)$   
fin  
fin
```

Le problème de Syracuse :

```
f(n) : si n = 1 alors  
    retourner 1  
sinon  
    si n est pair alors  
        retourner  $f(n/2)$   
    sinon  
        retourner  $f(3n + 1)$   
fin  
fin
```

Expérimentalement (i.e. en programmant cette fonction) pour tout entier n fixé, $f(n) = 1$ (c.a.d. le calcul s'arrête en un temps fini).

Il est conjecturé que ce programme s'arrête pour toute valeur de n .

Ce problème dont l'énoncé est très simple, a été posé en 1930 par Lothar Collatz, puis transmis par Helmut Hasse puis enfin par Stanislas Ulam et a passionné tous les mathématiciens qui l'ont considéré. On le connaît aussi sous le nom du problème $3x + 1$

Que sait-on aujourd'hui ?

- ▶ Plus d'une centaine d'articles publiés sur la question

Que sait-on aujourd'hui ?

- ▶ Plus d'une centaine d'articles publiés sur la question
- ▶ Pour 75 % des entiers la conjecture a été démontrée.

Que sait-on aujourd'hui ?

- ▶ Plus d'une centaine d'articles publiés sur la question
- ▶ Pour 75 % des entiers la conjecture a été démontrée.
- ▶ Mais pour le reste ?????

Histoire d'une grenouille

Le crêpier maladroit

Des algorithmes

Tri par retournements

Les fonctions de Collatz

La fonction très réursive Tak

Un dernier exercice

une simple fonction réursive : la fonction Tak

Pour i, j, k entiers on définit la fonction Tak proposée par Takeuchi pour le test des implémentations de la récurtivité dans des langages fonctionnels de type Lisp (cf. R.P. Gabriel) :

Tak(i, j, k) : si $i \leq j$ alors

retourner j

sinon

$Tak(Tak(i - 1, j, k), Tak(j - 1, k, i), Tak(k - 1, i, j))$

fin

Un dernier exercice

une simple fonction réursive : la fonction Tak

Pour i, j, k entiers on définit la fonction Tak proposée par Takeuchi pour le test des implémentations de la récurtivité dans des langages fonctionnels de type Lisp (cf. R.P. Gabriel) :

Tak(i, j, k) : si $i \leq j$ alors

retourner j

sinon

$Tak(Tak(i - 1, j, k), Tak(j - 1, k, i), Tak(k - 1, i, j))$

fin

Montrer que cet algorithme doublement récurtif s'arrête pour tout triplet d'entiers (i, j, k) et trouver ce qu'il calcule.