

Examen 1 2018 Q5

Quelle est la complexité de l'algorithme suivant :

algo5	
entrée : Entier positif n	
sortie : valeur	
① Si $n \leq 2$	) $\Theta(1)$ a
Sortir : 3	
Sinon	)
Sortir : $4 \log(\underbrace{\text{algo5}(n-2))}_{(2)}) + 1$	

(2)
(3)

$\Rightarrow C(n) \in \Theta(n)$

$\Theta(n)$

$\Theta(1)$

$\Theta(\log(n-2))$

$\Theta(\log n)$

1) Compteur

$$C(n) = \underbrace{a}_{(1)} + \underbrace{C(n-2)}_{(2)} + b$$

$$C(n) = C(n-2) + k$$

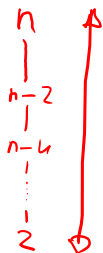
le reste
de (3)
une fois (2)
fait
1/7

Examen 1 2018 Q5

Quelle est la complexité de l'algorithme suivant :

algo5
entrée : Entier positif n
sortie : valeur
Si $n \leq 2$
Sortir : 3
Sinon
Sortir : $4 \log(\text{algo5}(n-2)) + 1$

2) arbre des appels



$$\Theta\left(\frac{n}{2}\right) = \Theta(n)$$

3) incrémenter et compter

Combien ça augmente si l'entrée augmente de 2 ? $\rightarrow \Delta C(n)=1 \Rightarrow C(n) \in \Theta(\log n)$

Examen 1 2018 Q5

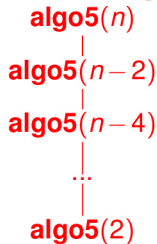
Quelle est la complexité de l'algorithme suivant :

algo5
entrée : <i>Entier positif n</i> sortie : <i>valeur</i>
Si $n \leq 2$ Sortir : 3 Sinon Sortir : $4 \log(\text{algo5}(n-2)) + 1$

👉 $\Theta(n)$

Rappel : vous avez trois moyens :

1. *Compter* les instructions :
 $C(n) = a + C(n-2) + b$
2. *dessiner* le graphe des appels :



3. « incrémenter et compter » :
 $\Delta C(n+2) = \text{cte}$

Examen 1 2018 Q15

Cette question est la suite de celle (Q13) posée la semaine passée que je reproduis donc ici :

On s'intéresse ici à raccourcir les répétitions de trois ou plus valeurs identiques **successives** ; par exemple à produire la liste (6, 6, 4, 4, 12, 4, 6) à partir de la liste (6, 6, 4, 4, 4, 12, 4, 6) en supprimant le 4 en cinquième position car il est présent trois fois consécutives.

À noter que :

- ▶ les seules valeurs supprimées sont celles qui sont répétées **successivement** trois fois ou plus (l'une à la suite de l'autre) ; on ne garde alors que deux de ces valeurs (cf la valeur 4 ci-dessus) ;
- ▶ toute valeur présente une ou deux fois successivement est préservée, et l'on conserve l'ordre de la liste ;
- ▶ en sortie on ne peut donc pas avoir plus de deux valeurs identiques consécutives.

Écrivez un algorithme **récuratif** pour résoudre ce problème.

Indication : on pourra séparer le premier élément et le reste de la liste.

Entrée : L Sortie : L simplifiée

① $\left[\begin{array}{l} n \leftarrow \text{taille}(L) \\ \text{Si } n \leq 2 \\ \quad \text{Sortir } L \end{array} \right.$

$$L' \leftarrow \text{simplifie}(L[2], \dots, L[n])$$

Si $L[1] = L'[1]$ et $L[1] = L'[2]$
Sortir L'

Sortir $(L[1]) \oplus L'$

① condition d'arrêt

② comment « réduire » ?

enlever un
le premier le dernier un quelconq
couper ailleurs

(6, 6, 4, 4, 4, 12, ...)

↓ appel réc.

(6, 4, 4, 4, 12, ...)

↳ réponse $\overset{L'}{\underbrace{(6, 4, 4, 12, \dots)}}$

↳ réponse (rouge): (6, 6, 4, 4, 12, ...)

(4, 4, 4, 4, 4)

appel (4, 4, 4, 4)

↳ réponse: $L' = (4, 4)$

↳ réponse : (4, 4) ne pas
ajouter 4

Examen 1 2018 Q16

Déterminez la complexité de votre algorithme (question précédente).
Justifiez votre réponse.

Leçon I.3 (calculabilité) – Points clés

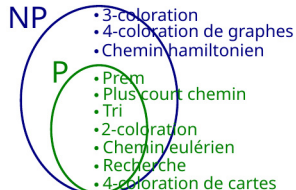
- ▶ qu'est-ce que *une* machine de Turing et **la** machine de Turing universelle
- ▶ problème = une question (portant sur ses entrées) et (une infinités d') instances
- ▶ il y a beaucoup de problèmes non calculables
exemple : problème de l'arrêt
- ▶ P et NP
- ▶ on ne sait pas si NP est inclu dans P
- ▶ la 2- et la 4-coloration de graphes planaires sont dans P
la 3-coloration de graphes est dans NP
- ▶ et d'autres...

difficulté ↑

Problèmes non calculables

- Terminaison d'algorithme
- Plus court algorithme

Problèmes calculables



dét d'alg.

problèmes

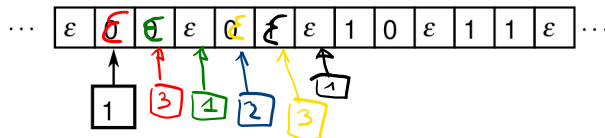
Leçon I.3 (calculabilité) – Machines de Turing

14:27

lu

	0	1	ϵ
1	$(3, \epsilon, +)$	$(4, \epsilon, +)$	$(2, \epsilon, +)$
2	$(3, \epsilon, +)$	$(4, \epsilon, +)$	$(6, \epsilon, -)$
3	$(1, \epsilon, +)$	$(1, \epsilon, +)$	$(1, \epsilon, +)$
4	$(5, \epsilon, -)$	$(3, \epsilon, +)$	$(1, \epsilon, +)$
5	$(5, \epsilon, +)$	$(5, \epsilon, +)$	$(7, 1, -)$
6	$(6, \epsilon, +)$	$(6, \epsilon, +)$	$(7, 0, -)$

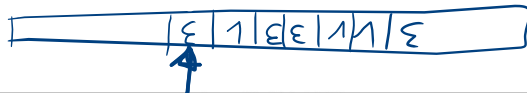
état



$\epsilon \epsilon \epsilon \epsilon \epsilon 1 \epsilon \epsilon 1 0 \epsilon \dots$

$\dots \epsilon 1 \epsilon \epsilon 1 0 \epsilon \dots$

Réponse :



Leçon I.3 (calculabilité) – Machines de Turing

Si

Ne commence

boucle

	0	1	ϵ
1	(3, ϵ , +)	(4, ϵ , +)	(2, ϵ , +)
2	(3, ϵ , +)	(4, ϵ , +)	(6, ϵ , -)
3	(1, ϵ , +)	(1, ϵ , +)	(1, ϵ , +)
4	(5, ϵ , -)	(3, ϵ , +)	(1, ϵ , +)
5	(5, ϵ , +)	(5, ϵ , +)	(7, 1, -)
6	(6, ϵ , +)	(6, ϵ , +)	(7, 0, -)

Si $lu = 0$
... (3) ...

Sinon si $lu = 1$
... (4) ...

Tant que $lu \neq \epsilon$

... ϵ 0 0 ϵ 0 1 ϵ 1 0 ϵ 1 1 ϵ ...

1

Leçon I.3 (calculabilité) – Machines de Turing

	0	1	ϵ	
1	(3, ϵ , +)	(4, ϵ , +)	(2, ϵ , +)	
2	(3, ϵ , +)	(4, ϵ , +)	(6, ϵ , -)	Si
3	(1, ϵ , +)	(1, ϵ , +)	(1, ϵ , +)	
4	(5, ϵ , -)	(3, ϵ , +)	(1, ϵ , +)	
5	(5, ϵ , +)	(5, ϵ , +)	(7, 1, -)	Sortie
6	(6, ϵ , +)	(6, ϵ , +)	(8, 0, -)	

...

ϵ	0	0	ϵ	0	1	ϵ	1	0	ϵ	1	1	ϵ
------------	---	---	------------	---	---	------------	---	---	------------	---	---	------------

 ...

↑

1

Leçon I.3 (calculabilité) – P et NP

Si un algorithme résout un problème X (dont la taille de l'entrée est n) en $\Theta(n^2 \log n)$, que peut on dire ?

$X \in P$ (donc $X \in NP$)

Leçon I.3 (calculabilité) – P et NP

Si un algorithme résout un problème X (dont la taille de l'entrée est n) en $\Theta(n^2 \log n)$, que peut on dire ?

Si un algorithme permet de vérifier en $\Theta(n^2 \log n)$ qu'une donnée de taille n est bien une instance positive (« solution ») à un problème Y , moyennant une donnée supplémentaire (certificat) de taille en $\Theta(\log n)$, que peut on dire ?

$Y \in NP$

Leçon I.3 (calculabilité) – P et NP

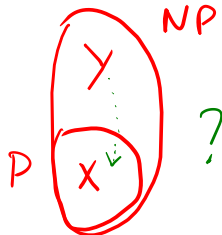
Si un algorithme résoud un problème X (dont la taille de l'entrée est n) en $\Theta(n^2 \log n)$, que peut on dire ?

Si un algorithme permet de vérifier en $\Theta(n^2 \log n)$ qu'une donnée de taille n est bien une instance positive (« solution ») à un problème Y , moyennant une donnée supplémentaire (certificat) de taille en $\Theta(\log n)$, que peut on dire ?

Un tel problème est-il dans P ?

Peut-il être dans P ?

1. X est dans P
2. Y est dans NP
3. pas forcément
4. oui c'est possible



Leçon I.3 (calculabilité) – Complexité des problèmes

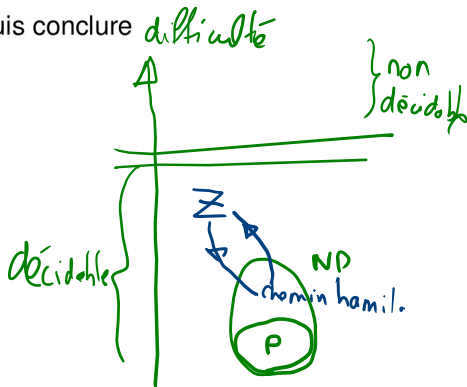
Un algorithme possible pour résoudre un problème Z est le suivant :

1. transformer les données de Z en un graphe
2. trouver un chemin hamiltonien dans ce graphe
3. retransformer ce chemin dans les données de Z , puis conclure

Que peut-on dire de Z ?

- ✓ 1. rien
2. qu'il est dans P
3. qu'il est dans NP
- ~~4. qu'il n'est pas dans P~~
- (5. (qu'il est NP -difficile)

NON !!



Leçon I.3 (calculabilité) – Complexité des problèmes

Un algorithme possible pour résoudre un problème Z est le suivant :

1. transformer les données de Z en un graphe
2. trouver un chemin hamiltonien dans ce graphe
3. retransformer ce chemin dans les données de Z , puis conclure

Que peut-on dire de Z ?

1. rien ➡ **en effet pas grand chose au niveau du cours ICC**
2. qu'il est dans P ➡ **on ne sait pas, en tout cas pas avec cet algorithme car le problème du chemin hamiltonien est dans NP**
3. qu'il est dans NP ➡ **on ne sait pas, rien ne dit qu'on puisse en vérifier une solution (instance positive) en un temps polynomial**
4. qu'il n'est pas dans P ➡ **ce serait bien présomptueux (savez vous si NP est dans P ou non ?)**
5. (qu'il est NP -difficile) ➡ **c'est en fait la définition de NP -difficile, HORS COURS**