Les machines de Turing

Corrigé partiel de la feuille de travaux dirigés n°1

23 septembre 2017

1. La machine Addition_binaire est une solution (sauf que la machine exemple calcule une addition au lieu de vérifier l'exactitude d'une addition.

2. Le problème se décompose en deux parties. La recherche du milieu, et ensuite la comparaison bit par bit. Comma la comparaison nécessite un temps de $O(n^2)$, on peut se contenter de la recherche du médiane en même temps. Ceci dit, on peut trouver la médiane de manière plus efficace.

3. Sur deux bandes on peut recopier le mot sur la deuxième bande, ce qui permet d'effectuer les comparaisons en temps linéaire. Il devient donc intéressant de trouver la médiane en temps linéaire aussi. Cette dernière tâche est aussi facile.

4. Soit L le langage défini ci dessus.
$L = \{m \in \{a\}^\star : \exists k \in \mathbb{N},\ |m| = k^2\}$
Observons tout d'abord que $(n+1)^2 - n^2 = 2n + 1$ et que $2(n+1) + 1 - (2n+1) = 2$

i	1		2		3		4		5
i^2	1		4		9		16		25
$(i+1)^2 - i^2$		3		5		7		9	
$2(i+1)+1-(2i+1)$			2		2		2		2

D'où l'on déduit un algorithme très simple pour engendrer les carrés parfaits en unaire.
Soit A le nombre de bâtonnets à ajouter, initialisé à 1.
Soit N le nombre de bâtonnets posés initialisé à 1.
On répète $A := A + 2$; $N := N + A$
Le scénario de la machine de Turing est alors le suivant :
Soit A le nombre de a supplémentaires à effacer sur le ruban 1

1. L'entrée $m \in \{a\}^\star$ est inscrite sur le ruban 1 et A, initialisé à 1 sur le ruban 2
2. on efface sur le ruban 1 autant de a que de bâtonnets inscrits sur le ruban 2 ;
3. on ajoute deux bâtonnets sur le ruban 2 et on recommence en (2)
4. on accepte si on a tout effacé sur le ruban 1 et qu'on est à la droite du dernier bâtonnet du ruban 2 et on rejette autrement

La fonction de transition de la machine de Turing est alors décrite dans le tableau suivant :

	$L1$	$L2$	$E1$	$E2$	$\Delta 1$	$\Delta 2$	Q
E	a	1	B	1	$\rightarrow$	$\rightarrow$	E
E	B	1	B	1	$-$	$-$	NO
E	B	B	B	B	$-$	$-$	Acc
E	a	B	a	1	$-$	$\rightarrow$	A
A	a	B	a	1	$-$	$\leftarrow$	R
R	a	1	a	1	$-$	$\leftarrow$	R
R	a	B	a	B	$-$	$\rightarrow$	E

où E est l'état initial Acc celui d'acceptation et NO celui de rejet. Les autres états s'interprètent comme suit : E où on efface autant de a que de 1, A où on ajoute 1 sur le ruban 2 pour mettre à jour le nombre de a à effacer et R pour retourner au début du ruban 2.

5. a) Une des idées qui peuvent venir à l'esprit est d'utiliser le deuxième ruban comme un compteur qu'on incrémente au fur à mesure. Ainsi on initialise le compteur à 0, puis lors de la lecture de chaque lettre du mot donnée incrémente le compteur d'une unité.
Comme pour une donnée de longueur n le résultat sera de longueur $\log n$, chaque augmentation nécessite $O(\log n)$ opérations. Comme le nombre d'augmentations est n la complexité totale est en $O(n \log n)$.
b) Ce même principe peut être utilisé pour construire une machine à une bande de complexité $O(n^2)$. En effet, il suffit de positionner le compteur après l'extrémité gauche des données. Dans ce cas le travail qui concerne les augmentations sera en $O(n \log n)$. Par contre, il faudra pour chaque caractère parcourir le mot jusqu'au résultat, ce qui nécessitera un temps $O(n^2)$ d'où la complexité en $O(n^2)$ (voir la machine Compteur_simple).
c) Pour améliorer la complexité de la machine, on souhaite un algorithme en $O(n \log n)$ avec un seul ruban. Ceci implique qu'on ne peut parcourir plus que $O(\log n)$ fois les données - ce qui implique de récupérer en moyenne un bit du résultat à chaque passage. Par ailleurs, le log de la complexité incite à chercher l'utilisation de la dichotomie.
La machine que nous proposons ne fait que compter modulo 2 en effaçant lors de chaque passage la moitié du mot. Son scénario est le suivant :
On supposera dans la suite que la chaîne d'entrée m est sur un alphabet à une seule lettre a et qu'on dispose d'un marqueur de début de chaîne et d'un marqueur de fin de chaîne. Au début du calcul, le ruban contient donc $\sharp m\$$ comme entrée. A la fin du calcul, la longueur en binaire de m sera inscrite sur le ruban à la suite du $ de la fin.

à l'aller la machine dispose de deux états principaux
- un état dont la signification est que la machine a lu un nombre pair de a (état 1) ;
- un état dont la signification est que la machine a lu un nombre impair de a (état 2).

Lorsque la machine est dans l'état 1, elle remplace le caractère courant par un blanc tandis que lorsque la machine est dans l'état 2, elle conserve le caractère courant. Si on passe le $ de fin dans l'état 2, on ajoute un 1 à la fin de la chaîne de caractères qui représente la longueur de m ; si on passe le $ de fin dans l'état 1, on ajoute un 0.

au retour on retourne au début du mot en conservant le ruban intact sauf si il n'y a plus que des caractères blancs entre le $ et le $\sharp$ auquel cas on a terminé

La fonction de transition de la machine est alors la suivante :

état	L	E	depl.	nouv. état
1	$\sharp$	$\sharp$	$\rightarrow$	1
	B	B	$\rightarrow$	1
	a	B	$\rightarrow$	2
	$	$	$\rightarrow$	3
2	B	B	$\rightarrow$	2
	$	$	$\rightarrow$	4
	a	a	$\rightarrow$	1
3	0	0	$\rightarrow$	3
	1	1	$\rightarrow$	3
	B	0	$\leftarrow$	5
4	0	0	$\rightarrow$	4

état	L	E	depl.	nouv. état
4	1	1	$\rightarrow$	4
	B	1	$\leftarrow$	5
5	a	a	$\leftarrow$	5
	B	B	$\leftarrow$	5
	0	0	$\leftarrow$	5
	1	1	$\leftarrow$	5
	$\sharp$	$\sharp$	$\rightarrow$	1
	$	$	$\leftarrow$	6
6	B	B	$\leftarrow$	6
	a	a	$\leftarrow$	5
	$\sharp$	$\sharp$	$-$	7

1 est l'état initial et signifie qu'on a lu un nombre pair de a que l'on remplace par B ; 2 signifie qu'on a lu un nombre impair de a ; 3 qu'on a traversé le marqueur de fin de mot et qu'on doit ajouter 0 à la fin de la longueur ; 4 qu'on a traversé le marqueur de fin de mot et qu'on doit ajouter 1 à la fin de la longueur ; 5 est l'état qui nous permet de retourner au début du ruban ; 6 qui teste s'il n'y a que des blancs entre le marqueur de début et le marqueur de fin auquel cas, on passe dans l'état 7, état final, sinon retourne au début du mot.
Nous illustrons ci-dessous le fonctionnement de la machine de Turing sur le mot aaa

1	$\underline{\sharp}$	a	a	a	\$	B	B
1	$\sharp$	$\underline{a}$	a	a	\$	B	B
2	$\sharp$	B	$\underline{a}$	a	\$	B	B
1	$\sharp$	B	a	$\underline{a}$	\$	B	B
2	$\sharp$	B	a	B	$\underline{\$}$	B	B
4	$\sharp$	B	a	B	\$	$\underline{B}$	B
5	$\sharp$	B	a	B	$\underline{\$}$	1	B
6	$\sharp$	B	a	$\underline{B}$	\$	1	B
6	$\sharp$	B	$\underline{a}$	B	\$	1	B
5	$\sharp$	$\underline{B}$	a	B	\$	1	B
5	$\underline{\sharp}$	B	a	B	\$	1	B
1	$\sharp$	$\underline{B}$	a	B	\$	1	B

1	$\sharp$	B	$\underline{a}$	B	\$	1	B
2	$\sharp$	B	B	$\underline{B}$	\$	1	B
2	$\sharp$	B	B	B	$\underline{\$}$	1	B
4	$\sharp$	B	B	B	\$	$\underline{1}$	B
4	$\sharp$	B	B	B	\$	1	$\underline{B}$
5	$\sharp$	B	B	B	\$	$\underline{1}$	1
5	$\sharp$	B	B	B	$\underline{\$}$	1	1
6	$\sharp$	B	B	$\underline{B}$	\$	1	1
6	$\sharp$	B	$\underline{B}$	B	\$	1	1
6	$\sharp$	$\underline{B}$	B	B	\$	1	1
6	$\underline{\sharp}$	B	B	B	\$	1	1
7	$\sharp$	B	B	B	\$	1	1

Voir la machine Compteur_efficace)

d) Avec deux rubans, au lieu d'effacer sélectivement, on recopie les caractères restants sur l'autre ruban, tout en les effaçant sur le ruban initial et on itère le procédé. Au premier passage, on lit donc une chaîne de longueur n, au second une chaîne de longueur $n/2$ (par excès ou par défaut) etc plus les $\log n$ bits de la taille. En sommant, on obtient

$$\sum_{i=0\ldots \log n} \left(n/2^i + i\right) = 2n + \log n$$

donc asymptotiquement en $O(n)$.

Remarque : Une analyse plus fine de notre première machine permet de montrer que notre estimation $O(n \log n)$ était trop large. En effet, nous avons supposé que chaque augmentation nécessite $O(\log n)$ transitions. Le coût réel d'une incrémentation dépend en effet de la terminaison du nombre qu'on incrémente. Ainsi, si le nombre se termine par un 0, alors le coût est 1 et ceci se produit dans la moitié des cas ! Si le nombre ne se termine pas par un 0, alors la question est si l'avant-dernier chiffre est 0 ou non. Si le nombre se termine par 01 alors le coût est de 4 transitions (ce qui inclut le calcul et aussi le repositionnement de la tête de lecture). Et ce cas se produit dans un quart des cas. De manière générale, si le nombre incrémenté se termine par un 0 suivi de i fois 1, alors le coût de l'incrémentation est $2(i+1)$ et ce cas se produit avec une probabilité $\frac{1}{2^i}$. Ainsi nous pouvons déduire une analyse plus précise de la complexité :

$$\sum_{i\geq 0} \frac{n}{2^i} 2(i+1) = 4n \sum_{i\geq 0} \frac{i}{2^i} = 8n$$

. Ce qui nous permet de conclure que notre première machine de conception très simple était déjà linéaire !

6. a) La machine de Turing à deux rubans qui reconnait L fonctionne un peu à la façon d'un automate à pile :

- pour chaque 0, elle inscrit un caractère sur le deuxième ruban
- pour chaque 1, elle efface un caractère sur le deuxième ruban
- si le deuxième ruban est vide, le mot d'entrée est reconnu.

Sa fonction de transition est la suivante :

	$L1$	$L2$	$E1$	$E2$	$\Delta 1$	$\Delta 2$	Q
q_0	0	B	0	X	$\rightarrow$	$\rightarrow$	q_0
	1	B	1	B	$\rightarrow$	$\leftarrow$	q_1
q_1	1	X	1	B	$\rightarrow$	$\leftarrow$	q_1
	B	X	B	B	$-$	$\leftarrow$	q_2
q_2	B	B	B	B	$-$	$-$	q_f

Dont la complexité est linéaire.

b) On remplace chaque 0 par un X, chaque 1 par un Y et on vérifie à la fin qu'il n'y ait plus que des caractères B, X, Y sur le ruban. La fonction de transition de cette machine est la suivante :

état	L	E	depl.	nouv. état
q_0	0	X	$\rightarrow$	q_1
	Y	Y	$\rightarrow$	q_3
q_1	0	0	$\rightarrow$	q_1
	Y	Y	$\rightarrow$	q_1
	1	Y	$\leftarrow$	q_2
q_2	0	0	$\leftarrow$	q_2
	Y	Y	$\leftarrow$	q_2
	X	X	$\rightarrow$	q_0
q_3	Y	Y	$\rightarrow$	q_3
	B	B	$-$	q_f

Dont la complexité est en $O(k^2)$. Comme $n = 2k$, la complexité est donc en $O(n^2)$.

c) Avec un seul ruban. On utilise la machine qui donne la longueur binaire du mot d'entrée. Le scénario est le suivant :
- — on compte le nombre de zéros en les effaçant avec X ;
- — on compte le nombre de uns en les effaçant avec Y en vérifiant simultanément que le nombre de uns est égal au nombre de zéros ;
- — on vérifie syntaxiquement que le mot inscrit à la fin est bien de la forme $BX^\star Y^\star$ si le nombre de uns est égal au nombre de zéros auquel cas on accepte l'entrée.

Cette machine fonctionne en temps $O(n \log n)$.

Avec deux rubans, le plus simple est de recopier l'entrée sur le second ruban, puis de ramener la tête de l'un des rubans sur le premier caractère (p.e. le premier) et de parcourir de manière synchrone les deux chaînes. Tant qu'on lit le couple (0,1) ou (1,0) ça marche, on accepte si on lit simultanément le couple (B, B) et on rejette autrement.

7. Avec un seul ruban. On utilise la machine qui donne la longueur binaire du mot d'entrée. Le scénario est le suivant :
- — on compte la parité du nombre de zéros en effaçant un sur deux avec X ;
- — on compte la parité du nombre de uns en effaçant un sur deux avec Y en vérifiant simultanément que les deux parités sont égales ; puis on recommence.

Cette machine fonctionne en temps $O(n \log n)$ (voir la machine Autant_a_que_b).