Les castors affairés

Épreuve pratique d'algorithmique et de programmation
Concours commun des Écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juin/Juillet 2016

Attention !

N'oubliez en aucun cas d'indiquer votre nom et votre u_0 sur votre fiche réponse

Important.

Il vous a été donné un numéro u_0 qui servira d'entrée à vos programmes. Les réponses attendues sont généralement courtes et doivent être données sur la fiche réponse fournie à la fin du sujet. À la fin du sujet, vous trouverez en fait deux fiches réponses. La première est un exemple des réponses attendues pour un $\widetilde{u_0}$ particulier (précisé sur cette même fiche et que nous notons avec un tilde pour éviter toute confusion!). Cette fiche est destinée à vous aider à vérifier le résultat de vos programmes en les testant avec $\widetilde{u_0}$ au lieu de u_0. Vous indiquerez vos réponses (correspondant à votre u_0) sur la seconde et vous la remettrez à l'examinateur à la fin de l'épreuve.

En ce qui concerne la partie orale de l'examen, lorsque la description d'un algorithme est demandée, vous devez présenter son fonctionnement de façon schématique, courte et précise. Vous ne devez en aucun cas recopier le code de vos procédures!
Quand on demande la complexité en temps ou en mémoire d'un algorithme en fonction d'un paramètre n, on demande l'ordre de grandeur en fonction du paramètre, par exemple: $O(n^2)$, $O(n \log n)$,...
Il est recommandé de commencer par lancer vos programmes sur de petites valeurs des paramètres et de ***tester vos programmes sur des petits exemples que vous aurez résolus préalablement à la main ou bien à l'aide de la fiche réponse type fournie en annexe***. Enfin, il est recommandé de lire l'intégralité du sujet avant de commencer afin d'effectuer les bons choix de structures de données dès le début.

L'une des grandes contributions scientifiques d'Alan Turing a été l'introduction d'une formalisation du calcul mécanique aujourd'hui appelée « machine de Turing ». Une machine de Turing peut être vue comme un automate fini équipé d'une bande infinie sur laquelle se déplace une tête qui peut lire et écrire des `0` ou des `1` [1]. La notion de machine de Turing est utile pour raisonner sur l'ensemble des fonctions calculables par ordinateur : informellement, tout ce qui pourrait calculer un ordinateur disposant d'une quantité infinie de mémoire peut être calculé par une machine de Turing, et inversement.
Le but du sujet est de simuler l'exécution de machines de Turing afin de trouver des machines qui s'arrêtent soit après un nombre maximum d'étapes, soit en laissant un maximum de `1` écrits sur la bande. Ce problème, appelé jeu du castor affairé (« busy beaver ») [2], est lié à des notions importantes en informatique théorique.

1 Machines de Turing

Une machine de Turing (ou une machine dans la suite du sujet) est une paire (Q, δ) :

— Q est un ensemble fini d'états, qui sera toujours dans ce sujet $\{0, \dots n-1\}$ pour un certain $n > 0$,

— $\delta : Q \times \{\texttt{0}, \texttt{1}\} \to (Q \cup \{\bot\}) \times \{\texttt{0}, \texttt{1}\} \times \{-1, 1\}$ est la fonction de transition (ou la table des transitions).

0 est l'état initial, et $\bot$ est un état spécial qui sera appelé l'état terminal ; quand l'exécution d'une machine de Turing atteint l'état terminal, elle s'arrête.
Une transition est une paire $(q, x) \in Q \times \{\texttt{0}, \texttt{1}\}$. Une transition (q, x) est dite terminale si $\delta(q, x) = (\bot, y, d)$ pour un $y \in \{\texttt{0}, \texttt{1}\}$ et $d \in \{-1, 1\}$.
Informellement, $\delta(q, x)$ décrit ce que fait la machine quand elle se trouve en état q, et que le symbole courant en face de la tête de lecture est x. Si $\delta(q, x) = (q', y, d)$, alors la machine va (dans cet ordre) :

1. écrire y à la position courante,
2. déplacer la tête de d cases (si $d = 1$, alors la tête se déplace d'une case vers la droite, et si $d = -1$, la tête se déplace d'une case vers la gauche), et
3. passer dans l'état q'.

Ce mécanisme sera décrit formellement dans la suite de cette section. Tout au long du sujet, on utilisera $\leftarrow$ comme un synonyme de -1, et $\rightarrow$ comme un synonyme de 1 lors des déplacements de la tête.
À l'instar d'un automate fini, une machine de Turing peut être représentée par un graphe orienté, où les sommets sont les états et les arcs représentant les transitions. Si $\delta(q, x) = (q', y, d)$, alors il y aura un arc entre q et q', et cet arc sera étiqueté x:y,d.

1. Dans ce sujet, on ne travaillera que sur les machines de Turing binaires, c'est-à-dire, travaillant sur l'alphabet $\{\texttt{0}, \texttt{1}\}$, et commençant sur une configuration vide. Des définitions ont donc été simplifiées en conséquence.
2. Ce problème a été introduit en 1962 par Tibor Radó, et est toujours un sujet de recherche actuel en informatique théorique.

q \ x	0	1
0	$(1, 1, \leftarrow)$	$(\perp, 0, \rightarrow)$
1	$(2, 1, \rightarrow)$	$(0, 1, \leftarrow)$
2	$(0, 1, \rightarrow)$	$(1, 1, \rightarrow)$

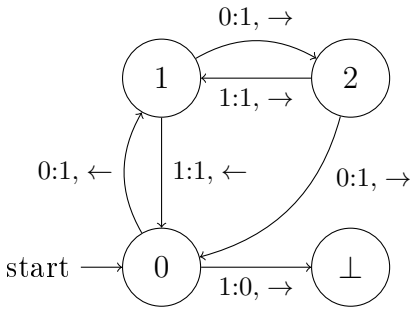


FIGURE 1 – Exemple d'une machine de Turing à 3 états ($T_{3,26\,489}$ pour $u_0 = 42$). À gauche : sa représentation sous forme d'une table de transitions, et à droite : sous forme d'automate.

Génération de Machines de Turing. Si $a > 0$ et $b \geq 1$ sont deux entiers, on note $a \bmod b$ le reste de la division euclidienne de a par b, autrement dit l'unique entier r avec $0 \leq r < b$ tel qu'il existe un entier q satisfaisant $a = bq + r$.
Soit $(u_i)_{i \in \mathbb{N}}$ la suite telle que u_0 est l'entier qui vous a été donné (à reporter sur votre fiche réponse), et pour tout $i > 0$:

$$u_{i+1} = 16\,807 \times u_i \mod m$$

où $m = 2^{31} - 1 = 2\,147\,483\,647$.

Question 1 *Calculez :*
a) $u_{10} \bmod 1234$ ***b)*** $u_{10\,000} \bmod 1234$ ***c)*** $u_{1\,000\,000} \bmod 1234$

On note $T_{n,t} = (Q, \delta)$ la machine suivante :

— $Q = \{0, \ldots n-1\}$

— pour tout $q \in \{0, \ldots, n-1\}$ et $x \in \{0, 1\}$:

$$\delta(q, x) = (q' \ , \ u_{z+2} \bmod 2 \ , \ 2 \times (u_{z+3} \bmod 2) - 1)$$

où $z = t + 8q + 4x$ et :

$$q' = \begin{cases} \perp \text{ si } u_{z+1} \bmod n = 0 \\ u_z \bmod n \text{ sinon.} \end{cases}$$

Question 2 *Écrivez les machines* ***a)*** $T_{2,2}$, ***b)*** $T_{2,3}$, ***c)*** $T_{3,4}$.

Question 3 *Pour chacun des k suivants, donnez le nombre d'entiers t, avec $0 \leq t < 1\,000$ et tels que $T_{4,t}$ possède exactement k transition(s) terminale(s).*
a) $k = 0$, ***b)*** $k = 1$, ***c)*** $k = 3$.

Question à développer pendant l'oral 1 *Donnez le nombre de machines à n états. Pour quelles valeurs de n est-il envisageable de parcourir toutes les machines à n états sur un ordinateur ?*

Configuration. Une configuration représente l'état de la bande, la position de la tête, et l'état dans laquelle se trouve une machine, à un instant donné lors de l'exécution de la machine. Formellement, une configuration est un triplet (B, p, q) où :

— B, représentant l'état de la bande, est une fonction de $\mathbb{Z}$ dans $\{\mathtt{0}, \mathtt{1}\}$,

— $p \in \mathbb{Z}$ est la position de la tête,

— $q \in Q \cup \{\bot\}$ est l'état courant de la machine.

On note par $\mathcal{C}$ l'ensemble des configurations.
Initialement, la bande de la machine de Turing est entièrement remplie de `0`, sa tête est en position 0, et la machine est dans l'état 0. La configuration initiale, notée C_0, est donc $(B_{\mathtt{0}}, 0, 0)$ où $B_{\mathtt{0}}$ est la fonction qui associe `0` à tous les entiers.

Opération. Étant donné une fonction $B : X \to Y$, et $(a, b) \in X \times Y$, on notera par $B[a \to b]$ la fonction de X vers Y telle que $\forall x \in X \setminus \{a\}$, $(B[a \to b])(x) = B(x)$, et $(B[a \to b])(a) = b$.
Soit $T = (Q, \delta)$ une machine. On définit la fonction $\mathrm{Op}_T : \mathcal{C} \to \mathcal{C}$, représentant une opération de la machine, telle que $\mathrm{Op}_T((B, p, q)) = (B', p', q')$ où :

— $\delta(q, B(p)) = (q', y, d)$,

— $B' = B[p \to y]$ (c'est-à-dire, B' est identique B, sauf en p où elle vaut y),

— $p' = p + d$.

Notez que cette fonction n'est pas définie si $q = \bot$.

Exécution. Une exécution d'une machine T, notée E_T, est une séquence (finie ou infinie) de configurations $(C_0, C_1, \ldots)$, telle que :

— C_0 est la configuration initiale,

— pour tout $0 \leq i < \mathrm{card}(E_T) - 1$, $\mathrm{Op}_T(C_i) = C_{i+1}$,

— si E_T est finie, alors l'état de sa dernière configuration $C_{\mathrm{card}(E_T)-1}$ est $\bot$,

où $\mathrm{card}(E)$ est la cardinalité de la séquence E : ∞ si elle est infinie, et k si elle est finie avec $E = (C_0, \ldots C_{k-1})$.
La figure 2 présente le début de l'exécution de la machine donnée en figure 1.

Question à développer pendant l'oral 2 *Continuez l'exécution de la machine donnée en figure 2. Donnez la cardinalité de cette exécution.*

Indication. Dans la suite du sujet, on ne dépassera jamais plus de 250 itérations et en conséquence la position de la tête sera toujours inférieure ou égale à 250 en valeur absolue (sauf pour la dernière instance des deux dernières questions).

Question à développer pendant l'oral 3 *Quelle structure de données utilisez-vous pour coder une configuration ? Quelle structure de données pourriez-vous utiliser s'il n'y avait pas de borne sur la position de la tête ?*

Question 4 *Pour chacun des K suivants, donnez le nombre d'entiers t, avec $0 \leq t < 1\,000$ et tels que $\mathrm{card}(E_{T_{4,t}}) \leq K$.*
a) $K = 2$, ***b)*** $K = 10$, ***c)*** $K = 100$.

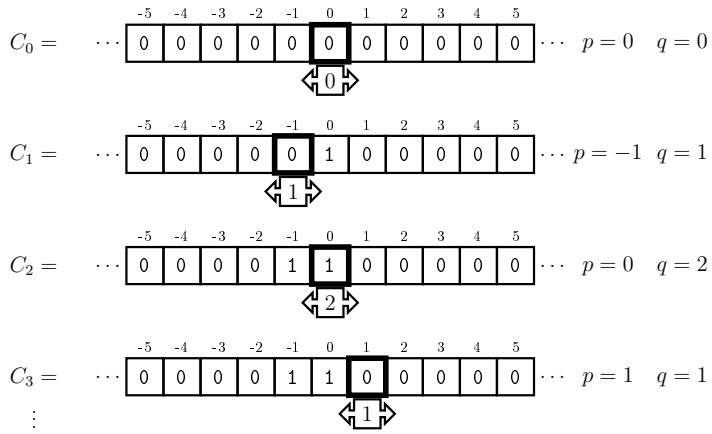


FIGURE 2 – Début de l'exécution de la machine de Turing de la figure 1.

Question à développer pendant l'oral 4 *Dans sa définition originale, une machine de Turing travaille sur un alphabet Σ fini, qui n'est pas forcément de taille 2 (c'est-à-dire, les symboles de la bande peuvent être n'importe quelle lettre de Σ). Expliquez, informellement, comment simuler une machine de Turing travaillant sur un alphabet Σ de taille au moins 3 par une machine de Turing travaillant sur $\{\texttt{0}, \texttt{1}\}$.*

2 Détection de non-terminaison

Dans la section 3, vous allez devoir trouver des machines dont l'exécution termine, mais telle que cette exécution soit la plus longue possible, ou qu'elle ait le plus grand score possible (le score étant le nombre de `1` sur la bande de la dernière configuration). Ce sont ces machines qui sont appelées des castors affairés.
Pour ce faire, il faut pouvoir détecter et écarter les machines qui ne s'arrêtent jamais. Il s'agit d'un problème nommé « le problème de l'arrêt », qui a une grande importance en informatique théorique. On peut montrer qu'il s'agit d'un problème « indécidable » : il n'existe pas, et il est impossible qu'il existe, un algorithme qui prend en paramètre une machine de Turing, et qui dit si l'exécution de cette machine est finie ou infinie.
Néanmoins il est possible, dans certains cas, de détecter que l'exécution sera infinie. Sur les machines avec peu d'états (le cas de ce sujet), la plupart des cas de non-terminaison sont faciles à détecter. L'objectif de la section courante est de présenter et tester quelques conditions suffisantes pour avoir l'assurance que $\text{card}(E_T) = \infty$.
Notez bien : il est possible de traiter la section 3 en n'ayant traité que partiellement la section 2. De plus, les sous-sections 2.1 et 2.2 sont indépendantes.

2.1 Sous-ensemble bouclant

Dans certains cas, on peut savoir qu'une exécution ne s'arrêtera pas, sans pour autant comprendre ce que fait la machine : par exemple, lorsque la machine ne possède pas de transition finale.
Un sous-ensemble d'états $Q' \subseteq Q$ est bouclant si pour tout $q \in Q'$ et pour tout $x \in \{\texttt{0}, \texttt{1}\}$, $\delta(q, x) = (q', y, d)$ avec $q' \in Q'$.

Si l'exécution d'une machine arrive dans une configuration (B, p, q), où $q \in Q'$ pour un sous-ensemble bouclant Q', alors l'exécution ne termine pas. En effet, les états ultérieurs de l'exécution resteront dans Q', sans possibilité d'arriver sur $\perp$.
Étant donné une machine T, on note $L(T)$ l'union de tous les ensembles bouclants de T.

Question à développer pendant l'oral 5 *$L(T)$ est-il lui même bouclant ? Comment calculer $L(T)$? Quelle est la complexité de votre algorithme (en n, le nombre d'états) ?*

Question 5 *Pour chaque (n, M) suivants, donnez la paire (a, b) où :*

- *a est le nombre d'entiers t, avec $0 \leq t < M$ et tels que $0 \in L(T_{n,t})$ (c.-à-d. que l'état initial fait partie de $L(T_{n,t})$), et*
- *b est le nombre d'entiers t, avec $0 \leq t < M$ et tels que $\mathrm{card}(L(T_{n,t})) = \frac{n}{2}$.*

a) $(n, M) = (4, 1\,000)$,
b) $(n, M) = (8, 10\,000)$,
c) $(n, M) = (32, 100\,000)$.

Question 6 *Pour chaque (n, M, K) suivants, donnez le triplet (f, b, i) où :*

- *f est le nombre d'entiers t, avec $0 \leq t < M$, et tels que $\mathrm{card}(E_{T_{n,t}}) \leq K$,*
- *b est le nombre d'entiers t, avec $0 \leq t < M$, et tels que l'état de la $(K{+}1)$-ème configuration (si elle existe) dans l'exécution de $T_{n,t}$ appartient à $L(T_{n,t})$,*
- *$i = M - f - b$ (c'est-à-dire, le nombre de machines de la séquence pour lesquelles on ne sait pas conclure en détectant les sous-ensembles bouclants en au plus K itérations).*

a) $(n, M, K) = (2, 1\,000, 20)$,
b) $(n, M, K) = (3, 5\,000, 50)$,
c) $(n, M, K) = (4, 10\,000, 200)$.

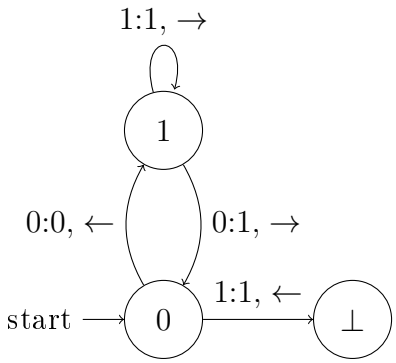
2.2 Bouclage de configurations

Dans la majorité des cas restants, il faut comprendre ce que fait la machine pour pouvoir détecter que son exécution ne termine pas. Pour faire cela, il faut regarder la suite des configurations dans l'exécution, et détecter qu'un schéma va se répéter. Par exemple, s'il existe $i < j$ tels que $C_i = C_j$ dans une exécution $E = (C_0, C_1, \ldots)$, alors cette exécution sera infinie.
On se propose de détecter un cas plus général : des exécutions qui vont boucler en répétant le même schéma, mais en se décalant vers la droite. La figure 3 présente une machine, avec son exécution sous forme compacte, pour laquelle c'est le cas.
On dit qu'une exécution $E = (C_0, C_1, \ldots)$ boucle à droite s'il existe $i < j$ tels que :

- $q_i = q_j$
- $p_i \leq p_j$
- pour tout $\ell \geq r$, $B_i(\ell) = B_j(\ell + p_j - p_i)$

où :

- $C_k = (B_k, p_k, q_k)$ pour tout $0 \leq k < \mathrm{card}(E)$,



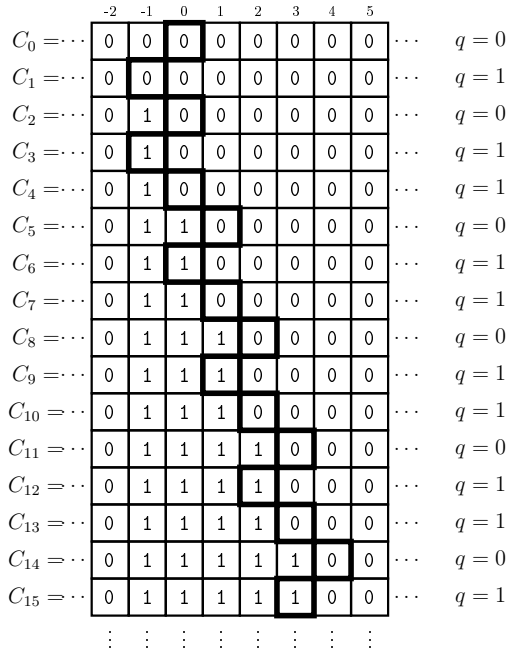


FIGURE 3 – Exemple de bouclage à droite.

— $r = \min_{i \leq k \leq j} p_k$ (c'est-à-dire, r est la plus petite position de la tête parmi toutes les configurations $C_i, \ldots, C_j$).

On dit que l'exécution boucle à droite avant K opérations si on a la condition supplémentaire $i < j \leq K$.
De manière similaire, on peut définir une exécution qui boucle à gauche. On dit qu'une exécution boucle si elle boucle à droite ou si elle boucle à gauche.

Question à développer pendant l'oral 6 *Expliquer pourquoi une exécution qui boucle à droite est infinie.*

Question 7 *Pour chaque (n, M, K) suivant, donnez le triplet (f, b, i) où :*

— *f est le nombre de machines T dans la séquence $(T_{n,t})_{0 \leq t < M}$ telles que $\text{card}(E_T) \leq K$,*

— *b est le nombre de machines T dans la même séquence dont l'exécution boucle (à gauche ou à droite) avant K opérations.*

— *$i = M - f - b$ (c'est-à-dire, le nombre de machines dans la séquence pour lesquelles on ne sait pas conclure en détectant les bouclages, en au plus K itérations).*

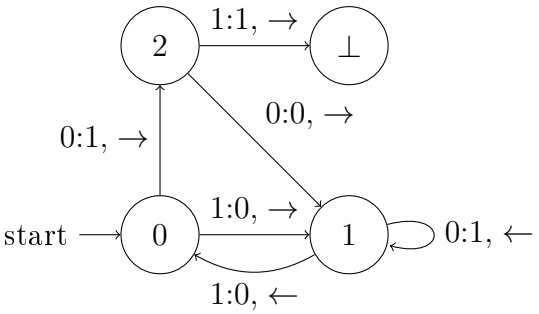
a) $(n, M, K) = (2, 1\,000, 20)$,
b) $(n, M, K) = (3, 5\,000, 50)$,
c) $(n, M, K) = (4, 10\,000, 200)$.

Question à développer pendant l'oral 7 *Donnez une machine dont l'exécution est infinie, mais qui ne boucle pas à gauche ou à droite. Que fait cette machine ? Proposez un algorithme qui détecterait des comportements similaires.*

2.3 Transitions inaccessibles

Indication. À partir de maintenant, les tests de détection de non-terminaison deviennent significativement plus difficiles à programmer, tout en écartant toujours moins de machines. Nous vous conseillons dès à présent de commencer la section 3, et de revenir à la section 2.3 pendant que votre programme de la section 3 s'exécutera.

Il se peut qu'une transition ne soit jamais utilisée dans l'exécution d'une machine. C'est par exemple le cas dans la machine suivante.



Supposons que la machine termine, avec l'exécution $(C_0, \ldots, C_{i+1})$ pour un $i \in \mathbb{Z}$. Pour arriver à la transition terminale $(2, \mathtt{1})$, la configuration $C_i = (B_i, p_i, q_i)$ doit se trouver en état $q_i = 2$, avec un $B_i(p_i) = \mathtt{1}$. La transition utilisée lors de l'opération pour passer de C_{i-1} à C_i doit donc être $(0, \mathtt{0})$, car c'est la seule transition arrivant à l'état 2. On a donc forcément : $p_{i-1} + 1 = p_i$, $q_{i-1} = 0$, $B_{i-1}(p_{i-1}) = \mathtt{0}$ et $B_{i-1}(p_i) = \mathtt{1}$. La configuration C_i ne peut pas être la configuration initiale, car un $\mathtt{1}$ est présent sur la bande. En continuant le même raisonnement, la transition utilisée lors de l'opération de C_{i-2} à C_{i-1} doit être $(1, \mathtt{1})$. On a donc : $p_{i-2} - 1 = p_{i-1}$ (et donc $p_i = p_{i-2}$) et $B_{i-2}(p_{i-2}) = \mathtt{1}$. Or, puisque $\mathrm{Op}_T(C_{i-2}) = C_{i-1}$, on a $B_{i-1}(p_{i-2}) = \mathtt{0}$, qui est en contradiction avec le fait que $B_i(p_i) = B_{i-1}(p_i) = \mathtt{1}$.
Une transition (q, x) d'une machine T est inaccessible si elle n'est jamais utilisée dans l'exécution de T (c.-à-d. qu'il n'y a aucune configuration (B_i, p_i, q_i) dans E_T avec $B_i(p_i) = x$ et $q_i = q$). Dans l'exemple précédent, la transition $(2, \mathtt{1})$ est donc inaccessible.

Une configuration partielle est un triplet $P = (B, p, q)$ tel que

- $B : \mathbb{Z} \to \{\mathtt{0}, \mathtt{1}, *\}$ représente un état partiel de la bande,
- $p \in \mathbb{Z}$ (la position de la tête),
- $q \in Q \cup \{\bot\}$ (l'état de la machine).

On voit les configurations comme des cas particulier de configurations partielles. Informellement, une configuration partielle P représente toutes les configurations C telles qu'on peut obtenir C depuis P en remplacant chaque $*$ par soit $\mathtt{0}$, soit $\mathtt{1}$.
Étant donné une machine T, on note R_T la fonction suivante qui associe un ensemble de configurations partielles à une configuration partielle :

$$R_T(B, p, q) = \bigcup_{\substack{q' \in Q \\ x \in \{\mathtt{0},\mathtt{1}\} \\ d \in \{-1,1\}}} \{(B[p\text{-}d \to x], p\text{-}d, q') : \delta(q', x) = (q, y, d) \text{ et } B(p\text{-}d) \in \{*, y\}\}\,.$$

On note A_T la clôture réflexive et transitive de R_T, c'est-à-dire que $P' \in A_T(P)$ si et seulement si il existe $k \geq 1$ et une séquence $(P_1, \dots P_k)$ avec $P_1 = P$, $P_k = P'$ et pour tout $1 \leq i < k$, $P_{i+1} \in R_T(P_i)$.
Étant donné $q \in Q$ et $x \in \{\texttt{0}, \texttt{1}\}$, on note par $P(q, x)$ la configuration partielle $(B, 0, q)$ avec $B(0) = x$ et $B(i) = *$ pour tout $i \neq 0$.

Question à développer pendant l'oral 8 *Informellement, que représentent $R_T(P)$ et $A_T(P)$? Comment se servir de $A_T(P(q, x))$ pour conclure qu'une transition est inaccessible ? En quoi la connaissance d'une transition inaccessible peut-elle nous aider pour détecter la non-terminaison d'une exécution ?*

Question 8 *Pour chacun des (n, M, K) suivants, combien y a-t-il d'entiers t, avec $0 \leq t < M$, tels que $T_{n,t}$ dispose d'exactement une transition terminale (q, x), et telle que* $\mathrm{card}(A_{T_{n,t}}(P(q, x))) \leq K$,
a) $(n, M, K) = (2, 1\,000, 20)$
b) $(n, M, K) = (3, 5\,000, 50)$
c) $(n, M, K) = (4, 10\,000, 100)$

La fin de la section est dédiée à la démonstration formelle des propriétés de R_T.
On dit qu'une configuration partielle est valide si $B(p) \neq *$. On munit les configurations partielles d'un ordre partiel $\preccurlyeq$ tel que $(B, p, q) \preccurlyeq (B', p', q')$ si et seulement si $p = p'$, $q = q'$ et pour tout $i \in \mathbb{Z}$, $B(i) \neq * \Rightarrow B'(i) = B(i)$.
On peut définir Op sur les configurations partielles valides de la même manière que sur les configurations : $\mathrm{Op}_T((B, p, q)) = (B[p \to y], p + d, q')$ où $(q', y, d) = \delta(q, B(p))$. Notez que Op_T n'est pas définie sur les configurations partielles non valides, ni quand $q = \bot$.
Si $\mathcal{P}$ est un ensemble de configurations partielles, et P' une configuration partielle, on note $\mathcal{P} \preccurlyeq P'$ s'il existe un $P \in \mathcal{P}$ tel que $P \preccurlyeq P'$.

Question à développer pendant l'oral 9 *Montrez que pour toutes configurations partielles P et P', $R_T(P) \preccurlyeq P'$ si et seulement si $P \preccurlyeq \mathrm{Op}_T(P')$. Concluez ensuite sur A_T.*

3 Les castors affairés

On note par $\mathcal{T}_n$ l'ensemble des machines de Turing à n états (sans compter l'état final). On définit :

$$\Lambda(n) = \max_{\substack{T \in \mathcal{T}_n \\ \text{card}(E_T) < \infty}} \text{card}(E_T)$$

c'est-à-dire la taille de la plus longue exécution d'une machine de Turing à n états, parmi les exécutions qui terminent. Une machine à n états est à survie maximum si $\text{card}(E_T) = \Lambda(n)$.
On définit le score d'une configuration comme étant le nombre de `1` sur la bande (c'est-à-dire $\text{score}((B, p, q)) = |\{i \in \mathbb{Z} : B(i) = 1\}|$). Le score d'une machine T est le score de la dernière configuration de l'exécution de la machine. Notez que le score n'est défini que si l'exécution termine. Enfin :

$$\Gamma(n) = \max_{\substack{T \in \mathcal{T}_n \\ \text{card}(E_T) < \infty}} \text{score}(T)$$

est le score maximum d'une machine de Turing à n états.
Un castor affairé est une machine à n états telle que $\text{score}(T) = \Gamma(n)$.
Le but de cette section est de chercher des castors affairés et des machines à survie maximum. Notez que $\Lambda(n)$ et $\Gamma(n)$ ne sont pas forcément atteints par la même machine.

Pour déterminer $\Lambda(n)$ et $\Gamma(n)$, il vous faudra parcourir toutes les machines à n états, et éliminer au plus tôt les machines qui ont une exécution infinie.

Question à développer pendant l'oral 10 *Il n'est pas nécessaire de tester toutes les machines à n états au sens de la définition de la section 1, et de la question orale 1. Donnez les différentes heuristiques que vous voyez pour limiter l'espace de recherche.*

Pour écarter les machines qui ont une exécution infinie, on peut utiliser les algorithmes de la section 2. Mais attention toutefois : certains tests de détection de bouclage prennent plus de temps que d'autres, et il faudra faire attention à bien ordonner vos tests.

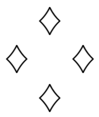
Indication. Pour tout $n \in \{2, 3, 4\}$, on a $\Lambda(n) \leq 250$. À l'heure actuelle, on ne connaît que des bornes inférieures pour $\Lambda(5)$ et $\Gamma(5)$...

Question 9 *Donnez la valeur (ou une borne inférieure) que vous obtenez pour $\Lambda(n)$ (c.-à-d. la plus longue exécution) et la machine qui donne cette longeur d'exécution pour :*
a) *$n = 2$* ***b)*** *la machine correspondante à 2 états,*
c) *$n = 3$* ***d)*** *la machine correspondante à 3 états,*
e) *$n = 4$* ***f)*** *la machine correspondante à 4 états,*
g) *$n = 5$* ***h)*** *la machine correspondante à 5 états.*

Question 10 *Donnez la valeur (ou une borne inférieure) que vous obtenez pour $\Gamma(n)$ (c.-à-d. le plus grand score) et la machine qui donne ce score pour :*
a) *$n = 2$* ***b)*** *la machine correspondante à 2 états,*
c) *$n = 3$* ***d)*** *la machine correspondante à 3 états,*
e) *$n = 4$* ***f)*** *la machine correspondante à 4 états,*
g) *$n = 5$* ***h)*** *la machine correspondante à 5 états.*

Pour chaque n, donnez la meilleure machine que vous trouvez, même si vous n'êtes pas sûr qu'il s'agisse de celle qui atteint $\Lambda(n)$ ou $\Gamma(n)$.
En plus de vos réponses sur la fiche, il vous est demandé d'écrire vos machines de Turing « championnes » dans des fichiers (un fichier par machine) sur votre clef USB, en essayant de respecter le format du fichier `exemple_figurel.txt`, qui décrit la machine de la figure 1.

Question à développer pendant l'oral 11 *Peut-on calculer une borne supérieure, même très grossière, de $\Lambda(n)$ (pour un n quelconque) ? Pourquoi ?*



Fiche réponse type: Les castors affairés

$\widetilde{u_0}$: 42

Question 1

a) 329

b) 470

c) 880

Question 2

a)

$q \backslash x$	0	1
0	$(1, 1, \leftarrow)$	$(1, 0, \leftarrow)$
1	$(\bot, 1, \leftarrow)$	$(\bot, 0, \leftarrow)$

b)

$q \backslash x$	0	1
0	$(1, 0, \rightarrow)$	$(\bot, 0, \rightarrow)$
1	$(0, 0, \rightarrow)$	$(\bot, 0, \leftarrow)$

c)

$q \backslash x$	0	1
0	$(2, 1, \rightarrow)$	$(2, 1, \leftarrow)$
1	$(0, 1, \leftarrow)$	$(2, 0, \leftarrow)$
2	$(1, 0, \leftarrow)$	$(0, 0, \rightarrow)$

Question 3

a) 73

b) 201

c) 243

Question 4

a) 283

b) 549

c) 550

Question 5

a) $(168, 51)$

b) $(2007, 136)$

c) $(20023, 16)$

Question 6

a) $(644, 129, 227)$

b) $(2862, 833, 1305)$

c) $(5150, 2008, 2842)$

Question 7

a) $(644, 356, 0)$

b) $(2862, 2137, 1)$

c) $(5150, 4844, 6)$

Question 8

a) 44

b) 296

c) 688

Question 9

a) ???

b) ???

c) ???

d) ???

e) ???

f) ???

g) ???

h) ???

Question 10

a) ???

b) ???

c) ???

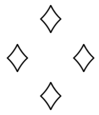
d) ???

e) ???

f) ???

g) ???

h) ???



Fiche réponse: Les castors affairés

Nom, prénom : ..... **u_0 :** .....

Question 1

a)

b)

c)

Question 2

a)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)

b)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)

c)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)
2	(, ,)	(, ,)

Question 3

a)

b)

c)

Question 4

a)

b)

c)

Question 5

a)

b)

c)

Question 6

a)

b)

c)

Question 7

a)

b)

c)

Question 8

a)

b)

c)

Question 9

a)

b)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)

c)

d)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)
2	(, ,)	(, ,)

e)

f)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)
2	(, ,)	(, ,)
3	(, ,)	(, ,)

g)

h)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)
2	(, ,)	(, ,)
3	(, ,)	(, ,)
4	(, ,)	(, ,)

Question 10

a)

b)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)

c)

d)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)
2	(, ,)	(, ,)

e)

f)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)
2	(, ,)	(, ,)
3	(, ,)	(, ,)

g)

h)

$q \backslash x$	0	1
0	(, ,)	(, ,)
1	(, ,)	(, ,)
2	(, ,)	(, ,)
3	(, ,)	(, ,)
4	(, ,)	(, ,)