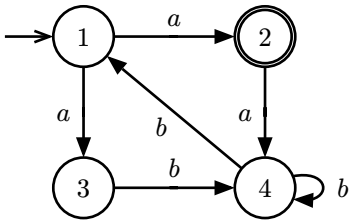
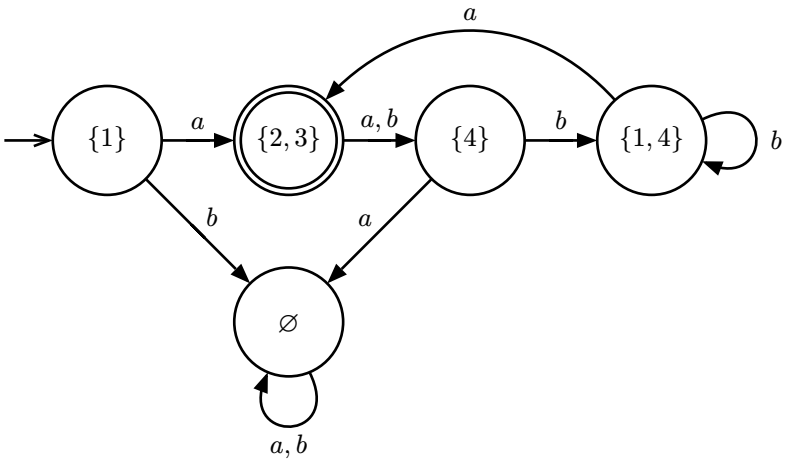
I – Algorithme de déterminisation

Déterminiser l'automate suivant en utilisant l'algorithme du cours :



L'état initial est $\{1\}$; les états finaux sont les ensembles contenant 2. Pour chaque ensemble X, la transition sur c mène à $\bigcup_{q\in X}\delta(q,c)$. Les seuls ensembles accessibles sont $\{1\}$, $\{2,3\}$, $\{4\}$, $\{1,4\}$ et $\varnothing$:



II – Clôture des langages reconnaissables

Si $m = m_1...m_n$ est un mot, son miroir est $\widetilde{m} = m_n...m_1$. Le miroir d'un langage L est $\widetilde{L} = \{\widetilde{m} \mid m \in L\}$.

1. Montrer que le miroir d'un langage reconnaissable est reconnaissable.

Si $A = (\Sigma, Q, I, F, E)$ reconnaît L, l'automate $\widetilde{A} = \left(\Sigma, Q, F, I, \widetilde{E}\right)$, où $\widetilde{E} = \{(p,a,q) \mid (q,a,p) \in E\}$, reconnaît $\widetilde{L}$. En effet, inverser un chemin acceptant étiqueté m donne un chemin acceptant étiqueté $\widetilde{m}$ dans $\widetilde{A}$, et réciproquement.

Pour un langage L sur Σ, on définit :
- $\mathrm{Pref}(L) = \{u \in \Sigma^* \mid \exists v \in \Sigma^*, uv \in L\}$, l'ensemble des préfixes ;
- $\mathrm{Suff}(L) = \{u \in \Sigma^* \mid \exists v \in \Sigma^*, vu \in L\}$, l'ensemble des suffixes ;
- $\mathrm{Fact}(L) = \{u \in \Sigma^* \mid \exists v, w \in \Sigma^*, vuw \in L\}$, l'ensemble des facteurs.

2. Donner des expressions régulières pour $\mathrm{Pref}(a^*b)$ et $\mathrm{Pref}((ab)^*)$.

$\mathrm{Pref}(a^*b)$ est dénoté par $a^*(\varepsilon|b)$ et $\mathrm{Pref}((ab)^*)$ par $(ab)^*(\varepsilon|a)$.

3. Montrer que si L est reconnaissable, alors $\mathrm{Pref}(L)$, $\mathrm{Suff}(L)$ et $\mathrm{Fact}(L)$ le sont aussi.

Soit $A = (\Sigma, Q, I, F, E)$ reconnaissant L. Notons Q' les états coaccessibles et Q'' les états accessibles. Un mot est préfixe d'un mot de L exactement lorsqu'il étiquette un chemin de I vers Q'. Donc (Σ, Q, I, Q', E) reconnaît $\mathrm{Pref}(L)$. De même, (Σ, Q, Q'', F, E) reconnaît $\mathrm{Suff}(L)$ et (Σ, Q, Q'', Q', E) reconnaît $\mathrm{Fact}(L)$: le chemin étiqueté par le suffixe ou le facteur se prolonge en un chemin acceptant de A.

Autre solution : $\mathrm{Suff}(L) = \widetilde{\mathrm{Pref}\left(\widetilde{L}\right)}$ puisque $\widetilde{xy} = \tilde{y}\tilde{x}$. En effet,

$$m \in \widetilde{\mathrm{Pref}\left(\widetilde{L}\right)} \iff \exists v, \widetilde{m}v \in \widetilde{L} \iff \exists v, \tilde{v}m \in L \iff m \in \mathrm{Suff}(L).$$

Enfin $\mathrm{Fact}(L) = \mathrm{Suff}(\mathrm{Pref}(L)) = \mathrm{Pref}(\mathrm{Suff}(L))$:

$$m \in \mathrm{Suff}(\mathrm{Pref}(L)) \iff \exists u, um \in \mathrm{Pref}(L) \iff \exists u, v, umv \in L \iff m \in \mathrm{Fact}(L).$$

4. Montrer que si L est régulier, alors $\mathrm{Pref}(L)$, $\mathrm{Suff}(L)$ et $\mathrm{Fact}(L)$ le sont aussi. Puisqu'on montrera que régulier et reconnaissable sont équivalents, il s'agit d'une preuve alternative.

Pour une expression régulière e, construisons par induction une expression $P(e)$ dénotant les préfixes de son langage $\mathcal{L}(e)$:

$$P(a) = \varepsilon|a,\ P(\varepsilon) = \varepsilon,\ P(\varnothing) = \varnothing,$$

$$P(e_1|e_2) = P(e_1)|P(e_2),$$
$$P(e_1e_2) = \begin{cases} \varnothing & \text{si } \mathcal{L}(e_2) = \varnothing \\ P(e_1)|e_1P(e_2) & \text{sinon} \end{cases},$$
$$P(e_1^*) = \varepsilon|e_1^*P(e_1).$$

Pour la concaténation, un préfixe s'arrête dans le premier mot ou continue dans le second ; la première possibilité exige que le second langage soit non vide. Pour l'étoile, un préfixe est vide, ou constitué de mots complets suivis du préfixe d'un mot. La vacuité se décide elle-même par induction : $\varnothing$ est vide, ε et les lettres ne le sont pas, une union est vide si ses deux termes le sont, une concaténation si l'un l'est, et une étoile ne l'est jamais.

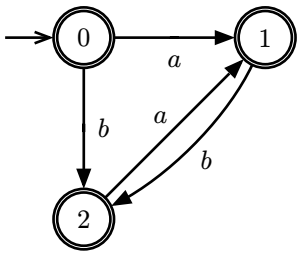
La stabilité des langages réguliers par miroir et les identités de la question 3 donnent alors la régularité de $\mathrm{Suff}(L) = \widetilde{\mathrm{Pref}\left(\widetilde{L}\right)}$ puis de $\mathrm{Fact}(L) = \mathrm{Suff}(\mathrm{Pref}(L))$.

III – Reconnaissable ou non ?

Pour chacun des langages suivants, dire s'il est reconnaissable ou non. Justifier. Pour un mot m, $|m|_a$ désigne le nombre d'occurrences de a.

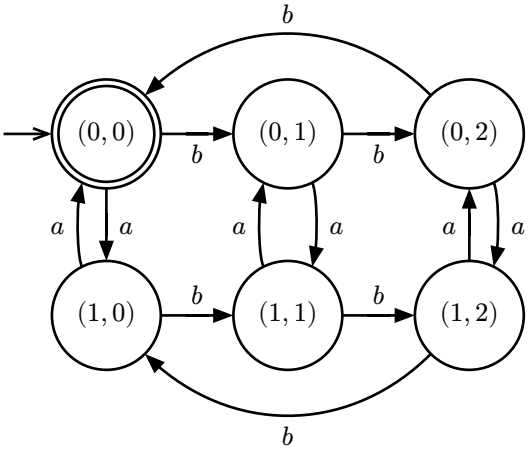
1. L_1 est l'ensemble des mots sur $\{a, b\}$ sans lettres consécutives égales.

Il est reconnaissable par l'automate ci-dessous. L'état 1 (respectivement 2) signifie que la dernière lettre lue est a (respectivement b). L'état 0 accepte le mot vide ; une répétition de lettre bloque la lecture.



2. L_2 est l'ensemble des mots sur $\{a, b\}$ ayant un nombre pair de a et un nombre de b multiple de 3.

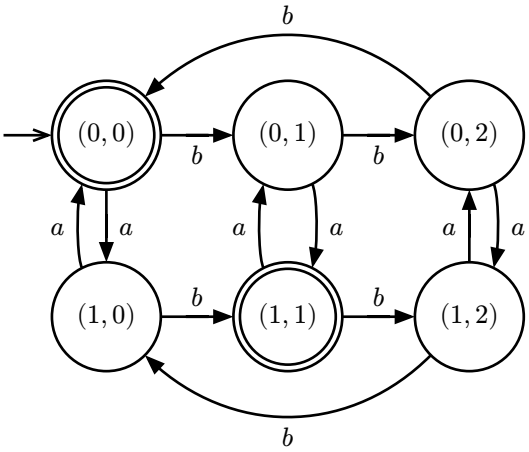
Un automate à deux états suit la parité du nombre de a et un automate à trois états le reste modulo 3 du nombre de b. Leur produit reconnaît L_2. Son état (i, j) signifie que le nombre de a est congru à i modulo 2 et celui de b à j modulo 3. Il commence et accepte en $(0, 0)$.



Sur a, il passe de (i, j) à $(1 - i, j)$; sur b, à $(i, (j + 1) \bmod 3)$. Cet invariant prouve que l'automate reconnaît exactement L_2.

3. $L_3 = \{m \in \{a, b\}^* \mid |m|_a \bmod 2 = |m|_b \bmod 3\}$.

Le même automate, avec $(0, 0)$ et $(1, 1)$ comme états finaux, reconnaît L_3 :

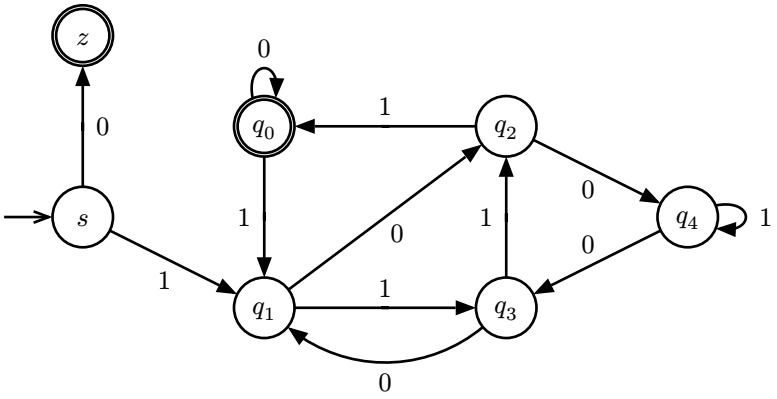


4. $L_4 = \{m \in \{a,b\}^* \mid |m|_a = |m|_b\}$.

Supposons L_4 reconnaissable et soit $n \geq 1$ une taille de pompage. Le mot $m = a^n b^n$ appartient à L_4 et a une taille au moins n. Le lemme de l'étoile donne $m = xyz$ avec $|xy| \leq n$, $y \neq \varepsilon$ et $xy^k z \in L_4$ pour tout $k \in \mathbb{N}$. Le mot y ne contient que des a ; pour $k = 0$, xz contient strictement moins de a que de b. Contradiction : L_4 n'est pas reconnaissable.

5. L_5 est l'ensemble des écritures usuelles en base 2 des multiples de 5. Le mot 0 représente zéro ; toute autre écriture commence par 1, et le mot vide est exclu.

Il est reconnaissable. Pour suivre la valeur modulo 5, utiliser les états $q_0, ..., q_4$ et $\delta(q_r, c) = q_{(2r+c) \bmod 5}$: ajouter le chiffre c à droite remplace la valeur v par $2v + c$. L'état q_0 est le seul état final parmi ces cinq états. Ajouter un état initial s, non final : sa transition 1 mène à q_1, et sa transition 0 mène à un état final z sans transition sortante. On accepte ainsi 0 et les écritures commençant par 1 de valeur divisible par 5.



6. $L_6 = \{a^p \mid p \text{ est un nombre premier}\}$.

Supposons L_6 reconnaissable et soit n une taille de pompage. Choisissons un nombre premier $p \geq n + 2$. Le lemme de l'étoile appliqué à a^p donne $a^p = a^{i_1}(a^{i_2})a^{i_3}$ avec $i_1 + i_2 \leq n$, $i_2 > 0$ et $a^{i_1 + k i_2 + i_3} \in L_6$ pour tout $k \in \mathbb{N}$. Prenons $k = i_1 + i_3$. La taille obtenue est $q = (i_1 + i_3)(1 + i_2)$. Les deux facteurs sont strictement supérieurs à 1, car $i_3 = p - i_1 - i_2 \geq 2$. Donc q n'est pas premier : contradiction.

IV – Réciproque du lemme de l'étoile

Sur $\Sigma = \{a, b, c\}$, soit $L_1 = \{ab^n c^n \mid n \in \mathbb{N}\}$, $L_2 = \{a^n u \mid n \in \mathbb{N} \setminus \{1\}, u \in \{b,c\}^*\}$ et $L = L_1 \cup L_2$.

1. Soit $m \in L$ tel que $|m| \geq 2$. Montrer qu'il existe des mots x, y, z tels que :

$$m = xyz,\ |xy| \leq 2,\ y \neq \varepsilon,\ xy^*z \subset L.$$

- Si $m \in L_1$: $m = ab^n c^n$ avec $n \geq 1$. On choisit $x = \varepsilon$, $y = a$ et $z = b^n c^n$. Alors $xyz \in L_1$ et $xy^k z \in L_2$ pour $k \in \mathbb{N} \setminus \{1\}$.
- Si $m \in L_2$: $m = a^n u$ avec $n \neq 1$ et $u \in \{b,c\}^*$.
  - Si $n = 0$: on écrit $u = u_1 u'$ avec $u_1 \in \{b,c\}$ et on choisit $x = \varepsilon$, $y = u_1$, $z = u'$. Alors $xy^*z \subset L_2$.
  - Si $n = 2$: on choisit $x = \varepsilon$, $y = aa$, $z = u$. Alors $xy^k z = a^{2k} u \in L_2$ pour tout $k \in \mathbb{N}$.
  - Si $n \geq 3$: on choisit $x = \varepsilon$, $y = a$, $z = a^{n-1} u$. Alors $xy^*z \subset L_2$.

Dans chaque cas, $m = xyz$ et $1 \leq |y| = |xy| \leq 2$.

2. Montrer que L n'est pas régulier. En déduire que la réciproque du lemme de l'étoile est fausse.

Si L était régulier, la stabilité de $\mathcal{L}_{\text{reg}}$ par intersection montrerait que $\mathcal{L}(ab^*c^*) \cap L = L_1$ est régulier.
Supposons par l'absurde que L_1 soit régulier et soit $n \geq 1$ donné par le lemme de l'étoile.

Le mot ab^nc^n s'écrirait xyz avec $|xy| \leq n$, $y \neq \varepsilon$ et $xy^*z \subset L_1$. Supprimer y enlève soit l'unique a, soit uniquement des b, sans toucher aux c. Dans les deux cas, $xz \notin L_1$, contradiction. Ainsi L n'est pas régulier, bien qu'il vérifie la conclusion du lemme de l'étoile.

Remarque : On peut aussi utiliser $\{b^nc^n \mid n \in \mathbb{N}\} = a^{-1}L_1$.

V – Algorithmes sur les automates

Soit $A = (\Sigma, Q, I, F, E)$ un automate à $n = |Q|$ états et $p = |E|$ transitions.
Décrire un algorithme pour résoudre chacun des problèmes suivants en précisant représentation et complexité.

1. $\mathcal{L}(A) = \varnothing$?
2. Plus petit mot de $\mathcal{L}(A)$?
3. $\mathcal{L}(A)$ fini ?
4. Pour deux automates A_1, A_2 : $\mathcal{L}(A_1) = \mathcal{L}(A_2)$?

1. Il est vide si et seulement s'il n'existe aucun chemin de I vers F. Faire un parcours (profondeur ou largeur) depuis I et tester si un état final est atteint. Avec une représentation par liste d'adjacence, chaque état est visité au plus une fois et chaque transition examinée au plus une fois : temps $O(n+p)$, espace auxiliaire $O(n)$.
2. Faire un parcours en largeur depuis I et mémoriser pour chaque état visité le mot le plus court pour l'atteindre. Le plus petit mot de $\mathcal{L}(A)$ est le plus court parmi ceux menant à un état final.
3. $\mathcal{L}(A)$ est infini exactement lorsqu'il existe un cycle accessible et coaccessible.

   Émonder A puis réaliser un parcours en profondeur depuis I : A possède un cycle orienté si et seulement si un arc arrière (revenant sur un sommet en cours d'appel récursif) est trouvé. Avec une représentation par liste d'adjacence, le temps et l'espace total, émondage compris, sont $O(n+p)$.
4. - Déterminiser et compléter les automates : $O(|\Sigma|\ 2^{n_1}) + O(|\Sigma|\ 2^{n_2})$.
   - Construire l'automate produit $A_1 \times A_2$ pour reconnaître la différence symétrique $L_1 \Delta L_2 = L_1 \cup L_2 \setminus L_1 \cap L_2$ (avec $F = F_1 \times (Q_2 \setminus F_2) \cup (Q_1 \setminus F_1) \times F_2$) : $O(n_1 n_2)$.
   - Tester si $\mathcal{L}(A_1 \times A_2) = \varnothing$ avec Q1 : $O(n_1 n_2 + p_1 p_2)$.

VI – Taille discriminante

Les automates sont sans transitions ε.

1. Soit A un automate. Décrire un algorithme pour déterminer la plus petite taille d'un mot reconnu par A et préciser sa complexité.

Effectuer un parcours en largeur depuis tous les états initiaux, à distance 0. La première distance d'un état final est la taille cherchée ; si aucun n'est atteint, le langage est vide. Avec des listes d'adjacence, chaque état et chaque transition sont traités au plus une fois : temps $O(n+p)$ pour n états et p transitions, espace auxiliaire $O(n)$.

2. Soit A un automate à n états et de langage $\mathcal{L}(A)$. Montrer que $\mathcal{L}(A) = \varnothing$ si et seulement si $\mathcal{L}(A)$ ne contient aucun mot de taille strictement inférieure à n.

$\Longrightarrow$: immédiat.

$\Leftarrow$: supposons $\mathcal{L}(A) \neq \varnothing$ et choisissons un mot accepté m de taille minimale. Si $|m| \geq n$, son chemin acceptant passe par au moins $n+1$ occurrences d'états. Deux sont égales par le principe des tiroirs. Supprimer le cycle entre elles donne un chemin acceptant de longueur strictement plus petite, contradiction. Ainsi $|m| < n$. Si $n = 0$, il n'existe aucun chemin acceptant et le résultat reste valable.

3. Soient $A_1 = (\Sigma, Q_1, i_1, F_1, \delta_1)$ et $A_2 = (\Sigma, Q_2, i_2, F_2, \delta_2)$ deux ADC à n_1 et n_2 états, de langages $L_1 \neq L_2$. Notons $l(L_1, L_2)$ la plus petite taille d'un mot appartenant à l'un des deux langages mais pas à l'autre. Montrer que $l(L_1, L_2) < n_1 n_2$.

Le produit d'états $Q_1 \times Q_2$, initial (i_1, i_2), de transition $\delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a))$ et de finaux $(F_1 \times (Q_2 \setminus F_2)) \cup ((Q_1 \setminus F_1) \times F_2)$ reconnaît $L_1 \triangle L_2$. Ce langage est non vide et le produit possède $n_1 n_2$ états. La question 2 donne un mot distinguant de taille strictement inférieure à $n_1 n_2$.

VII – Ensembles distinguants

Soit Σ un alphabet fini et $L \subset \Sigma^*$. Un *ADC* est un automate déterministe complet, noté $A = (Q, \Sigma, q_0, F, \delta)$. On rappelle qu'un langage est régulier si et seulement s'il est reconnu par un ADC. Pour les exemples, on fixe $\Sigma = \{a, b\}$ et $H = \mathcal{L}((a \mid b)^*ab(a \mid b)^*)$.

Un mot w est un *suffixe distinguant* pour u et v si exactement l'un des mots uw et vw appartient à L. Un ensemble $D \subset \Sigma^*$ est *distinguant* pour L si deux mots distincts de D ont toujours un suffixe distinguant.

I – Ensembles distinguants

1. Montrer que $\{\varepsilon, a, ab\}$ est distinguant pour H en donnant un suffixe distinguant pour chaque paire.

Chaque case donne un suffixe distinguant pour les mots de sa ligne et de sa colonne.

	ε	a
a	b	
ab	ε	ε

En effet, $ab \in H$ tandis que $\varepsilon, a, b \notin H$.

2. Montrer que, dans un ADC reconnaissant L, deux mots conduisant au même état depuis q_0 n'ont aucun suffixe distinguant. En déduire une borne sur le nombre d'états à l'aide d'un ensemble distinguant fini, puis montrer que l'existence d'un ensemble distinguant infini entraîne que L n'est pas régulier.

Si $\delta^*(q_0, u) = \delta^*(q_0, v)$ alors, pour tout $w \in \Sigma^*$,

$$\delta^*(q_0, uw) = \delta^*(\delta^*(q_0, u), w) = \delta^*(\delta^*(q_0, v), w) = \delta^*(q_0, vw).$$

Les mots uw et vw sont donc acceptés ou rejetés simultanément. Ainsi $u \longmapsto \delta^*(q_0, u)$ est injective sur tout ensemble distinguant D et $|Q| \geq |D|$. Si D est infini, aucun ADC ne peut reconnaître L, donc L n'est pas régulier.

3. Trouver un ensemble distinguant infini pour $K = \{a^n b^n \mid n \in \mathbb{N}\}$ et en déduire que K n'est pas régulier.

Prenons $D = \{a^n \mid n \in \mathbb{N}\}$. Si $i \neq j$, le suffixe b^i distingue a^i et a^j : $a^i b^i \in K$ tandis que $a^j b^i \notin K$. L'ensemble D est distinguant et infini, donc K n'est pas régulier.

Distinguer par des préfixes et des contextes. Dans les deux questions suivantes, $D \subset \Sigma^*$ est infini. On note $\widetilde{m}$ le miroir de m et $\widetilde{L} = \{\widetilde{m} \mid m \in L\}$. On rappelle que le miroir d'un langage régulier est régulier.

4. On suppose que, pour tous mots distincts $u, v \in D$, il existe un *préfixe distinguant* x tel qu'exactement l'un des mots xu et xv appartient à L. Montrer que L n'est pas régulier.

Le miroir est une bijection, donc $\widetilde{D}$ est infini. Comme $\widetilde{xu} = \tilde{u}\tilde{x}$ et $\widetilde{xv} = \tilde{v}\tilde{x}$, tout préfixe distinguant x fournit un suffixe distinguant $\tilde{x}$ pour les mots miroirs. Ainsi $\widetilde{D}$ est distinguant pour $\widetilde{L}$, qui n'est donc pas régulier. Par stabilité des langages réguliers par miroir, L n'est pas régulier.

5. On suppose que, pour tous mots distincts $u, v \in D$, il existe un *contexte distinguant* $(x, y) \in (\Sigma^*)^2$ tel qu'exactement l'un des mots xuy et xvy appartient à L. Montrer que L n'est pas régulier.

Indication : dans un ADC à n états, combien de fonctions distinctes $f_u : q \longmapsto \delta^*(q, u)$ les mots peuvent-ils induire ?

Supposons qu'un ADC à n états reconnaisse L. Il n'existe que n^n fonctions de Q dans Q. Comme D est infini, deux mots distincts $u, v \in D$ induisent la même fonction $f_u = f_v$. Pour tous $x, y \in \Sigma^*$, en posant $q = \delta^*(q_0, x)$,

$$\delta^*(q_0, xuy) = \delta^*\big(f_{u(q)}, y\big) = \delta^*\big(f_{v(q)}, y\big) = \delta^*(q_0, xvy).$$

Aucun contexte ne distingue donc u et v, d'où une contradiction. Plus généralement, tout ensemble dont les mots sont deux à deux distinguables par des contextes possède au plus n^n éléments.

VIII – Automates et palindromes

ENS 2018 MP oral

- *Expliquer les pistes envisagées et les difficultés rencontrées afin de permettre le dialogue avec le jury.*

On fixe un alphabet fini Σ avec $|\Sigma| > 1$. Un mot $w = a_1 ... a_n$ est un palindrome si $a_i = a_{n-i+1}$ pour tout $1 \leq i \leq n$. On note Π le langage des palindromes et $\mathcal{L}(A)$ le langage reconnu par un automate fini A sur Σ. Les automates sont sans transitions ε.

1. Soit $\Pi_n = \Pi \cap \Sigma^n$. Montrer que, pour tout ADC A et tout $n \in \mathbb{N}$, si $\mathcal{L}(A) \cap \Sigma^{2n} = \Pi_{2n}$, alors A a au moins $|\Sigma|^n$ états.

Associons à $u \in \Sigma^n$ l'état $f(u)$ atteint après sa lecture. Si $u \neq v$ et $f(u) = f(v)$, l'acceptation de $u\overline{u}$ entraîne celle de $v\overline{u}$, où $\overline{u}$ est le miroir de u. Ce dernier mot n'est pas un palindrome : ses deux moitiés imposeraient $u = v$. Ainsi f est injective, d'où la borne, également valable pour $n = 0$.

2. En déduire que Π n'est pas régulier.

Si un ADC à N états reconnaissait Π, la question 1 donnerait $N \geq |\Sigma|^N \geq 2^N > N$, contradiction.

3. Étant donné un automate fini A, peut-on calculer un automate A_Π reconnaissant $\mathcal{L}(A) \cap \Pi$?

Non en général : pour $\mathcal{L}(A) = \Sigma^*$, ce langage serait Π, qui n'est pas régulier.

4. Pour $m = b_1 ... b_n$, on note $\overline{m} = b_n ... b_1$ son miroir. Étant donné A, peut-on calculer un automate A'_Π reconnaissant $\{m \in \Sigma^* \mid m\overline{m} \in \mathcal{L}(A)\}$?

Oui. Écrivons $A = (Q, I, F, \delta)$ et renversons ses transitions pour obtenir l'automate miroir $\overline{A}$. Construisons le produit B d'états $Q \times Q$, initiaux $I \times F$ et finaux $D = \{(q, q) \mid q \in Q\}$, avec

$$(p, r) \xrightarrow{a} (p', r') \iff (p, a, p') \in \delta \ \text{ et } \ (r', a, r) \in \delta.$$

Par récurrence sur la taille de m, un chemin de B étiqueté m de (i, f) à (p, r) correspond exactement à un chemin $i \xrightarrow{m} p$ et à un chemin $r \xrightarrow{\overline{m}} f$ dans A. Il est acceptant si et seulement si $p = r$, ce qui permet de concaténer ces deux chemins en un chemin acceptant de A étiqueté $m\overline{m}$. Réciproquement, scinder un chemin acceptant de A étiqueté $m\overline{m}$ après la lecture de m donne un tel chemin acceptant de B. Ainsi $\mathcal{L}(B) = \{m \in \Sigma^* \mid m\overline{m} \in \mathcal{L}(A)\}$: prendre $A'_\Pi = B$.

5. On note $\Pi_{\text{pair}} = \bigcup_{n\in\mathbb{N}} \Pi_{2n}$. Proposer un algorithme déterminant si $\mathcal{L}(A) \cap \Pi_{\text{pair}}$ est vide, fini ou infini. Discuter ses complexités en temps et en espace.

Les parcours doivent être décrits avec leur structure de données et leur complexité.

La fonction $m \mapsto m\overline{m}$ est une bijection de $\mathcal{L}(B)$ vers $\mathcal{L}(A) \cap \Pi_{\text{pair}}$. Dans B, calculer les états accessibles et coaccessibles par deux parcours, puis conserver leur intersection. Si elle est vide, le langage est vide. Sinon, il est infini exactement lorsque ce sous-automate contient un cycle : on peut alors répéter ce cycle sur un chemin acceptant ; sans cycle, tout chemin a une longueur bornée. Un parcours en profondeur détecte les cycles.

Posons $N = |Q|$ et m_a le nombre de transitions de A étiquetées a. Le produit a N^2 états et $M = \sum_{a\in\Sigma} m_a^2$ transitions, construites en appariant les transitions de même lettre. Construction et parcours coûtent $O(N^2 + M)$ en temps et en espace, donc $O((N + |\delta|)^2)$. Pour un automate déterministe, $m_a \leq N$, d'où $O((1 + |\Sigma|)N^2)$.

6. Modifier l'algorithme de la question 5 pour calculer la cardinalité de $\mathcal{L}(A) \cap \Pi_{\text{pair}}$ lorsqu'elle est finie, en supposant A déterministe. Comment la complexité est-elle affectée ?

Même si A est déterministe, B ne l'est pas nécessairement. En revanche, chaque chemin acceptant de B étiqueté m correspond bijectivement à un chemin acceptant de A étiqueté $m\overline{m}$: il est unique. Après émondage, le graphe est acyclique dans le cas fini. Dans l'ordre topologique inverse, calculer

$$c(s) = \mathbf{1}_{s\in D} + \sum_{s \xrightarrow{a} t} c(t).$$

La somme porte sur les transitions, y compris lorsque plusieurs lettres mènent au même état. Le résultat est $\sum_{s\in I\times F} c(s)$, en ne conservant que les états utiles. L'unicité des chemins acceptants justifie que ce compte de chemins soit un compte de mots. Il y a $O(N^2 + M)$ opérations arithmétiques et autant de mémoire structurelle. En coût binaire, si les compteurs ont au plus b bits, les additions coûtent $O(b)$ et les compteurs $O(N^2 b)$ bits ; on peut prendre $b = O(1 + N^2 \log(1 + |\Sigma|))$, puisque les mots acceptés ont une taille inférieure à N^2.

7. Modifier les algorithmes des questions 5 et 6 pour traiter $\mathcal{L}(A) \cap \Pi$.

Pour chaque $a \in \Sigma$, garder le même produit et remplacer D par $D_a = \{(p, r) \mid (p, a, r) \in \delta\}$. Il reconnaît $\{m \mid ma\overline{m} \in \mathcal{L}(A)\}$, avec la même correspondance entre chemins acceptants. Appliquer les tests et le comptage au cas pair et à chaque lettre centrale, puis sommer les cardinalités. Les palindromes ainsi obtenus sont disjoints selon leur parité et leur lettre centrale, même si les langages de demi-mots se recoupent. Le temps est multiplié au plus par $1 + |\Sigma|$; les calculs successifs réutilisent le même graphe et le même espace de travail.