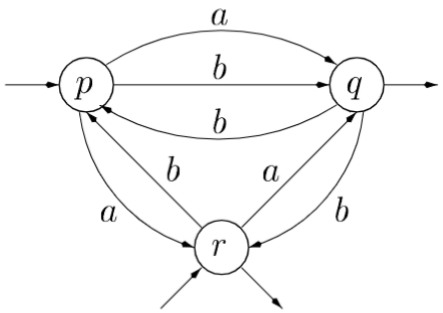
I Sujet 1 : Exercice Mines 2000

Il est demandé aux candidats et candidates une grande rigueur dans la rédaction. Si la première question est l'application d'une méthode ne demandant aucune justification, toutes les affirmations dans les solutions des autres questions doivent être rigoureusement justifiées.

L'alphabet $A = \{a,b\}$ est fixé. On considère l'automate $\mathcal{A}$ représenté ainsi :



L'ensemble des états de l'automate $\mathcal{A}$ est $\{p,q,r\}$. Les états initiaux p et r sont marqués par une flèche entrante et sans origine. Les états terminaux q et r sont marqués par une flèche sortante et sans destination. On note A^* l'ensemble des mots sur A.

1. Déterminiser l'automate $\mathcal{A}$, i.e. calculer un automate déterministe équivalent à $\mathcal{A}$.
2. Montrer que $L(\mathcal{A})$, le langage des mots acceptés par $\mathcal{A}$, est formé des mots de A^* qui ne contiennent pas le facteur aaa.
3. Montrer que le langage associé à l'expression rationnelle $b^*(a\,a^*b^*b)^*a^*$ est l'ensemble de tous les mots sur $\{a,b\}$, c'est-à-dire A^*.
4. En déduire une expression rationnelle dont le langage associé est $L(\mathcal{A})$.

Exercice 2 – Automates et langages de mots binaires

Dans cet exercice, on étudie différents langages sur l'alphabet $\mathrm{A} = \{0, 1\}$ à deux lettres. On note A^* l'ensemble des mots construits sur l'alphabet A. Le mot vide est noté ε. En OCaml, un mot sur A est implémenté par le type

```
type mot = bool list;;
```

à savoir par une liste de booléens où la lettre 0 est représentée par le booléen `false` et la lettre 1 par le booléen `true`.

11 On note L_1 le langage des mots de A qui représentent l'écriture binaire d'un entier naturel, où le bit de poids faible se situe en fin de mot. Pour assurer l'unicité de la représentation, l'écriture binaire d'un entier ne commence jamais par 0. C'est pourquoi l'entier nul est représenté par le mot vide.

11a) Donner l'écriture binaire de l'entier 41.

11b) Donner l'entier représenté par le mot 10101010.

11c) Pour un automate, que signifie la propriété d'être local standard ?

11d) Dessiner un automate local standard $\mathcal{A}_1$ reconnaissant le langage L_1.

11e) Écrire en OCaml une fonction `langage_1` de type `mot -> bool` qui prend en argument un mot m et qui renvoie `true` si et seulement si m appartient au langage L_1.

12 On note L_2 le langage dénoté par l'expression rationnelle $(0 + 1)^* \cdot 0$.

12a) Justifier que L_2 est un langage local.

12b) Dessiner un automate déterministe $\mathcal{A}_2$ reconnaissant le langage L_2.

12c) Écrire en OCaml une fonction `langage_2` de type `mot -> bool` qui prend en argument un mot m et qui renvoie `true` si et seulement si m appartient au langage L_2.

13 On note L_3 le langage reconnu par l'automate déterministe $\mathcal{A}_3$ défini par

- l'ensemble d'états $\mathrm{Q} = \{0, 1, 2\}$;
- l'état initial $i = 0$;
- l'ensemble d'états finals $\mathrm{F} = \{0\}$;
- la fonction de transition $\delta : \mathrm{Q} \times \mathrm{A} \longrightarrow \mathrm{Q}$ définie par

$$\forall\, q \in \mathrm{Q} \quad \forall\, a \in \mathrm{A} \qquad \delta(q, a) = (2q + a) \bmod 3$$

On rappelle que $(n \bmod 3)$ désigne le reste de la division euclidienne de l'entier n par 3.

13a) Dessiner l'automate $\mathcal{A}_3$.

13b) Écrire en OCaml une fonction `langage_3` de type `mot -> bool` qui prend en argument un mot m et qui renvoie `true` si et seulement si m appartient au langage L_3.

13c) Pour tout $n \in \mathbb{N}^*$, démontrer la propriété $\mathscr{P}(n)$ suivante :

$$\forall\, \omega_1, \ldots, \omega_n \in \mathrm{A} \qquad \delta^\star(0, \omega_1 \cdots \omega_n) = \sum_{k=1}^{n} \omega_k \, 2^{n-k} \bmod 3$$

où la fonction $\delta^\star : \mathrm{Q} \times \mathrm{A}^* \longrightarrow \mathrm{Q}$ est définie par

$$\forall\, q \in \mathrm{Q} \quad \forall\, \omega \in \mathrm{A}^* \quad \forall\, a \in \mathrm{A} \qquad \delta^\star(q, \varepsilon) = q \qquad \text{et} \qquad \delta^\star(q, \omega \cdot a) = \delta\big(\delta^\star(q, \omega), a\big)$$

14 On note $\mathrm{L}_4 = \mathrm{L}_1 \cap \mathrm{L}_2 \cap \mathrm{L}_3$.

14a) Décrire simplement l'ensemble des mots du langage L_4.

14b) Existe-t-il un automate reconnaissant le langage L_4 ?

II Sujet 3

II.1 Distance de Hamming

Soit Σ un alphabet. Si $u = u_1...u_n$ et $v = v_1...v_n$ sont deux mots de même longueur sur Σ, leur distance de Hamming est:

$$d(u, v) = |\{i \mid u_i \neq v_i\}|$$

1. Montrer que la distance de Hamming est une distance sur Σ^*.
2. Écrire une fonction `dist : 'a list -> 'a list -> int` calculant la distance de Hamming de deux mots de même longueur, sous forme de listes.

Étant donné un langage L sur Σ, on définit son voisinage de Hamming $\mathcal{H}(L) = \{u \in \Sigma^* \mid \exists v \in L,\ d(u, v) = 1\}$.

3. Donner une expression rationnelle du voisinage de Hamming de $L(0^*1^*)$.
4. Définir par récurrence une fonction H telle que, si e est une expression rationnelle d'un langage L sur $\Sigma = \{0, 1\}$, $H(e)$ est une expression rationnelle de $\mathcal{H}(L)$.
5. Écrire la fonction H précédente en Caml.

II.2 Questions sur les automates

1. À quelle condition nécessaire et suffisante simple le langage reconnu par un automate est vide? Décrire un algorithme pour le savoir.
2. À quelle condition nécessaire et suffisante simple le langage reconnu par un automate est fini? Décrire un algorithme pour le savoir.
3. Décrire un algorithme pour déterminer si deux automates admettent le même langage.
4. Soit A un automate à n états. Montrer que si $L(A) \neq \emptyset$ alors $L(A)$ contient un mot de longueur au plus $n - 1$.

III Sujet 4 : automate à pile

Soit Σ un alphabet fini, et Γ un autre alphabet fini appelé alphabet de pile. Un automate à pile sur Σ est un quintuplet $A = (Q, q_0, \gamma_0, \delta, F)$, où Q est un ensemble fini d'états, $q_0 \in Q$ est l'état initial, $\gamma_0 \in \Gamma$ est le symbole de pile initial, δ est une fonction de transition de $Q \times \Sigma \times \Gamma$ vers l'ensemble des parties de $Q \times \Gamma^*$, et $F \subseteq Q$ est un ensemble d'états finaux.

Une configuration de A est un couple (q, z) où $q \in Q$ est l'état et où z est un mot non-vide sur Γ appelé pile. La configuration initiale est (q_0, γ_0), et une configuration (q, z) est acceptante si $q \in F$. Lorsque l'automate est dans une configuration (q, z) et lit une lettre $a \in \Sigma$, il décompose $z = z'\gamma$ avec $\gamma \in \Gamma$ le sommet de pile, il choisit un $(q', g) \in \delta(q, a_{i+1}, \gamma)$ tel que $z'g$ soit non-vide, et il aboutit à la configuration $(q', z'g)$. L'automate à pile A accepte un mot de Σ^* s'il peut lire ses lettres dans l'ordre à partir de la configuration initiale pour parvenir à une configuration acceptante suivant ces règles.

Question 0. Étant donné un automate A sur Σ sans pile, expliquer comment construire un automate à pile A' sur Σ qui reconnaît le même langage que A.

Question 1. On prend dans cette question $\Sigma = \{a, b\}$. Proposer un automate à pile qui reconnait le langage $\{a^n b^n \mid n \geq 2\}$. Qu'en déduire?

Question 2. On prend toujours $\Sigma = \{a, b\}$. Proposer un automate à pile qui reconnait le langage $\{w \in \Sigma^* || w|_a = |w|_b\}$, où $|w|_a$ et $|w|_b$ désignent respectivement le nombre de a et de b de w.

Question 3. Pour $\eta \in \mathbb{N}^*$, étant donné un mot w de longueur n et un automate à pile A, les configurations η-tronquées sont les triplets (q, z) où $q \in Q$ et z est un mot non-vide sur Γ de longueur $\leq \eta$. Un calcul η-tronqué de A sur un mot est une séquence de configurations η-tronquées : la définition est comme auparavant sauf que, si $|z'g| \geq \eta$, on le remplace par son suffixe de longueur η. Le langage η-tronqué $L_{\leq\eta}(A)$ de A est le langage des mots sur lesquels A a un calcul η-tronqué acceptant.

Quelle est la relation entre $L_{\leq\eta}(A)$ et $L(A)$? Montrer que, pour tout automate à pile A et $\eta \in \mathbb{N}^*$, le langage η-tronqué de A est un langage régulier, et expliquer comment construire un automate sans pile qui le reconnaît.

Question 4. Soit w un mot de longueur n et A un automate à pile. Soit $\chi = (q_0, z_0), \ldots, (q_p, z_p)$ un calcul de A sur w, et notons $h_i := |z_i|$ pour tout $0 \leq i \leq p$. Pour $\eta \in \mathbb{N}^*$, une η-montagne de χ est un triplet d'indices $0 \leq l < m < r \leq p$ tels qu'on ait $h_l = h_r$, on ait $h_m - h_l \geq \eta$, et on ait $h_j > h_l$ pour tout $l < j < r$. Une montagne est une η-montagne pour un certain $\eta \in \mathbb{N}^*$.

On appelle G la plus grande longueur d'un mot de Γ^* dans une image de la fonction de transition δ. Montrer que, pour tout $\eta > G$, si $w \in L(A) \backslash L_{\leq\eta}(A)$, alors tout calcul acceptant de A sur w a une $(\eta - G)$-montagne.

Question 5. L'état de base d'une η-montagne $l < r$ dans un calcul est le couple $\beta(l, r) := (q_l, q_r, \gamma)$ où q_l, q_r sont les états des étapes l et r du calcul respectivement, et γ est le dernier symbole de la pile z_l.

Montrer que, pour tout automate à pile A, il existe $\eta_0 \in \mathbb{N}^*$ avec la propriété suivante : pour tout mot w, pour tout calcul χ de A sur w, si χ a une η_0-montagne (l, m, r), alors il existe une montagne (l', m, r') et une montagne (l'', m, r'') avec $l < l' < l'' < m < r'' < r' < r$ telles que $\beta(l', r') = \beta(l'', r'')$.

Question 6. Déduire des trois questions précédentes une forme affaiblie du lemme de pompage pour les automates à pile : pour tout langage L reconnu par un automate à pile, il existe un entier $p \in \mathbb{N}$ tel que, pour tout mot $w \in L$ avec $|w| > p$, on peut écrire $w = uvxyz$ où les mots u, v, x, y, z satisfont:

$-|vy| \geq 1$

- Pour tout $n \in \mathbb{N}$, on a $uv^n xy^n z \in L$

IV Sujet 5 : plus grand facteur dans un langage

On fixe l'alphabet $\Sigma := \{a, b\}$. Pour un langage L sur Σ, étant donné un mot $w \in \Sigma^*$, on veut calculer la longueur du plus grand facteur de w dans le langage L, notée $\mathrm{lpg}(L, w)$; on convient que cette longueur est de -1 si aucun facteur de w n'est dans L. On a donc $-1 \leq \mathrm{lpg}(L, w) \leq |w|$ où $|w|$ dénote la longueur du mot w.

Question 0. On considère le langage L_0 défini par l'expression rationnelle $(ab)^*$, et le mot $w_0 :=$ baababababa. Calculer $\mathrm{lpgf}\,(L_0, w_0)$.

Question 1. Étant donné un langage L, représenté comme un automate fini déterministe complet, et un mot $w \in \Sigma^*$, proposer un algorithme naïf pour calculer $\mathrm{lpgf}(L, w)$. Donner un pseudo-code pour cet algorithme, et déterminer sa complexité en temps et en espace.

Question 2. Proposer un algorithme plus efficace pour ce problème qui s'exécute en temps et en espace $O(|Q| \times |w|)$. Est-il important que l'automate soit déterministe?

Question 3. On dit que $w' \in \Sigma^*$ est un sous-mot de w s'il existe une fonction strictement croissante ϕ de $\{1, \ldots, |w'|\}$ dans $\{1, \ldots, |w|\}$ telle que $w'_i = w_{\phi(i)}$ pour tout $1 \leq i \leq |w'|$, où w'_i dénote la i-ème lettre de w' et de même pour w.

Étant donné un langage L représenté comme un automate et un mot w, comment calculer la longueur du plus grand sous-mot de w dans le langage L ?

Question 4. On pose le langage $L_4 := \{a^n b^n \mid n \in \mathbb{N}\}$. Proposer un algorithme efficace qui calcule $\mathrm{lpgf}\,(L_4, w)$ pour un mot d'entrée w.

Question 5. On pose le langage $L_5 := \{uu \mid u \in \Sigma^*\}$. Proposer un algorithme qui calcule $\mathrm{lpgf}\,(L_5, w)$ pour un mot d'entrée w en temps $O\left(|w|^2\right)$.

Question 6. On pose le langage L_6 des mots bien parenthésés sur Σ défini inductivement comme suit: le mot vide ε est bien parenthésé, la concaténation de deux mots bien parenthésés est bien parenthésée, et si $w \in \Sigma^*$ est bien parenthésé alors awb l'est aussi. Proposer un algorithme efficace qui calcule $\mathrm{lpgf}\,(L_6, w)$ pour un mot d'entrée w.