I Règles opératives

Pour chacune des propositions suivantes sur des expressions rationnelles, donner une preuve ou un contre-exemple :

1. $(e^*)^* \equiv e^*$
2. $(e_1|e_2)^* \equiv e_1^*|e_2^*$
3. $(e_1e_2)^* \equiv e_1^*e_2^*$
4. $(e_1|e_2)^* \equiv (e_1^*e_2^*)^*$

II Petites questions

1. Écrire une expression rationnelle dont le langage est l'ensemble des mots sur $\{a, b, c\}$ contenant exactement un a et un b (et un nombre quelconque de c).
2. Montrer que le langage sur $\{0, 1\}$ des écritures en base 2 de nombres divisibles par 4 est rationnel.
3. Écrire une expression rationnelle dont le langage est l'ensemble des mots sur $\{a, b, c\}$ ne contenant pas de a consécutifs (aa ne doit pas apparaître).
4. Écrire une expression rationnelle dont le langage est l'ensemble des mots sur $\{a, b, c\}$ contenant exactement deux a et tels que tout c est précédé d'un b.
5. Si $x \in \mathbb{R}$, on note $L(x)$ l'ensemble des préfixes des chiffres de x après la virgule. Par exemple, $L(\pi) = \{\varepsilon, 1, 14, 141, 1415...\}$. En sachant que $\frac{1}{6} = 0.1666...$ et $\frac{1}{7} = 0.142857142857...$, montrer que $L(\frac{1}{6})$ et $L(\frac{1}{7})$ sont rationnels.
6. Montrer plus généralement que $L(x)$ est rationnel si $x \in \mathbb{Q}$ (on montrera plus tard que c'est en fait une équivalence).

III 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 $\mathcal{H}(L) = \{u \in \Sigma^* \mid \exists v \in L,\ d(u, v) \leq 1\}$.

3. Donner une expression rationnelle du voisinage de Hamming de $L(0^*1^*)$.
4. Montrer que si L est un langage rationnel alors $\mathcal{H}(L)$ est un langage rationnel.
   Indice : On pourra définir par récurrence une fonction f telle que, si e est une expression rationnelle d'un langage rationnel L sur $\Sigma = \{0, 1\}$, $f(e)$ est une expression rationnelle de $\mathcal{H}(L)$.
5. Écrire la fonction f précédente en Caml, en utilisant le type suivant d'expression rationnelle :

```
type 'a regexp =
    | Vide | Epsilon | L of 'a (* L a est la lettre a *)
    | Union of 'a regexp * 'a regexp
    | Concat of 'a regexp * 'a regexp
    | Etoile of 'a regexp
```

IV Hauteur d'étoile

La hauteur d'étoile h d'une expression régulière est définie récursivement de la manière suivante :

- $h(e) = 0$ si e est $\emptyset$, ε ou une lettre.
- $h(e_1 + e_2) = \max(h(e_1), h(e_2))$.
- $h(e_1e_2) = \max(h(e_1), h(e_2))$.
- $h(e^*) = h(e) + 1$.

1. Quelle est la hauteur d'étoile de $(ba^*b)^*$?
2. Écrire la fonction h : `'a regexp -> int` en OCaml.

La hauteur d'étoile d'un langage L est la plus petite hauteur d'étoile d'une expression rationnelle e de langage L.

3. Que peut-on dire des langages de hauteur d'étoile 0 ?

V Clôture par sous-mot (oral ENS info)

On fixe un alphabet Σ. Étant donné deux mots $w, w' \in \Sigma^*$, on dit que w' est un sur-mot de w, noté $w \preccurlyeq w'$, s'il existe une fonction strictement croissante ϕ de $\{1, \dots, |w|\}$ dans $\{1, \dots, |w'|\}$ telle que $w_i = w'_{\phi(i)}$ pour tout $1 \leq i \leq |w|$, où $|w|$ dénote la longueur de w et w_i dénote la i-ème lettre de w. Étant donné un langage L, on note $\overline{L}$ le langage des sur-mots de mots de L, c'est-à-dire $\overline{L} := \{w' \in \Sigma^* \mid \exists w \in L, w \preccurlyeq w'\}$.

1. On pose L_0 le langage défini par l'expression rationnelle ab^*a, et L_1 le langage défini par l'expression rationnelle $(ab)^*$. Donner une expression rationnelle pour $\overline{L_0}$ et pour $\overline{L_1}$.
2. Montrer que, pour tout langage L, on a $\overline{\overline{L}} = \overline{L}$.
3. Existe-t-il des langages L' pour lesquels il n'existe aucun langage L tel que $\overline{L} = L'$?
4. Montrer que, pour tout langage régulier L, le langage $\overline{L}$ est également régulier.
5. On admettra pour cette question le résultat suivant : pour toute suite $(w_n)_{n \in \mathbb{N}}$ de mots de Σ^*, il existe $i < j$ tels que $w_i \preccurlyeq w_j$.

   Montrer que, pour tout langage L (non nécessairement régulier), il existe un langage fini $F \subseteq L$ tel que $\overline{F} = \overline{L}$.
6. Un langage L est clos par sur-mots si, pour tout $u \in L$ et $v \in \Sigma^*$ tel que $u \preccurlyeq v$, on a $v \in L$. Déduire de la question précédente que tout langage clos par sur-mots est régulier.
7. On considère un langage L arbitraire, non nécessairement régulier, et on souhaite construire effectivement un automate pour reconnaitre $\overline{L}$. Comment peut-on procéder, et de quelles opérations sur L a-t-on besoin? Discuter de l'efficacité de cette procédure.
8. Un langage L est clos par sous-mots si, pour tout $u \in L$ et $v \in \Sigma^*$ tel que $v \preccurlyeq u$, on a $v \in L$. Montrer que tout langage clos par sous-mots est régulier.
9. Démontrer le résultat admis à la question 5.

VI Utilisation de la programmation dynamique sur les mots

Résoudre les problèmes suivants sur des chaînes de caractères s, t par programmation dynamique :

1. Trouver la longueur maximum d'un facteur de s qui soit un palindrome.
   Remarque : l'algorithme de Manacher permet de résoudre ce problème en complexité linéaire.
2. Trouver la longueur maximum d'un sous-mot de s qui soit un palindrome.
3. Trouver la longueur maximum d'un sous-mot commun à s et t.
4. Trouver la longueur maximum d'un facteur bien parenthésé de s, où s contient uniquement des parenthèses ouvrantes et fermantes. Par exemple, `(())` et `()()` sont bien parenthésées mais pas `)()` ni `(()`.
   Remarque : on peut le faire de façon plus classique en utilisant une pile.

VII Lemme d'Arden

On s'intéresse au lemme suivant :

Lemme d'Arden

Soient A et B deux langages sur un même alphabet Σ. On considère l'équation

$$X = AX \cup B$$

d'inconnue $X \subset \Sigma^*$.

- Le langage $L = A^*B$ est la plus petite solution de cette équation (au sens de l'inclusion).
- Si $\varepsilon \notin A$, alors $L = A^*B$ est l'unique solution de l'équation.

Remarques :

- Il n'y a aucune hypothèse de rationnalité sur les langages A et B.On peut toutefois remarquer que si A et B sont rationnels, il en est de même de $L = A^*B$.
- Il sera plus agréable de noter $+$ au lieu de $|$ dans cet exercice, et de confondre expression régulière et langage associé : on parlera par exemple de l'équation $X = (a+b)X+c$, et l'on s'autorisera à écrire $a(b+c) = ab+bc$ au lieu de $a(b+c) \equiv ab+bc$ ou $L(a(b+c)) = L(ab+bc)$.

1. En admettant le lemme d'Arden, résoudre l'équation $X = (a+b)X + c$.
2. En admettant le lemme d'Arden, résoudre l'équation $X = (a+b+\varepsilon)X + c$.
3. Démontrer le lemme d'Arden.