Feuille de TD n° 2 : Résiduels, Automates non-déterministes, liste chaînées

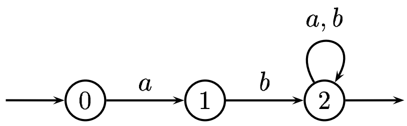
A désigne un alphabet fini. Le mot vide est noté ε.

Exercice 1 : Résiduels
Soient $\mathcal{L} \subseteq A^*$ un langage et $u \in A^*$ un mot. On appelle *résiduel* du langage $\mathcal{L}$ (à gauche) par rapport à u, et on note $u^{-1} \cdot \mathcal{L}$, le langage :

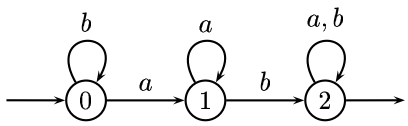
$$u^{-1} \cdot \mathcal{L} = \{v \in A^* \mid u \cdot v \in \mathcal{L}\}.$$

Autrement dit, $u^{-1} \cdot \mathcal{L}$ est l'ensemble des mots de $\mathcal{L}$ commençant par u, auxquels on a retiré ce préfixe u.

1. Calculer les résiduels du langage $\mathcal{L} = \{\varepsilon, abb, baaba\}$ par rapport aux mots a, b, ab, ba et bb.
2. Calculer, pour tous les mots u de A^*, le résiduel de $\mathcal{L}$ par rapport au mot u dans les exemples suivants (avec $A = \{a, b\}$) :
   - si $\mathcal{L} = \{a^p b^q \mid p, q \geqslant 0\}$;
   - si $\mathcal{L} = \{a^n b^n \mid n \geqslant 0\}$.
3. Soit $\mathcal{L}$ un langage. Montrer que $\varepsilon^{-1} \cdot \mathcal{L} = \mathcal{L}$ et que, pour tous mots u et v dans A^*, $(uv)^{-1} \cdot \mathcal{L} = v^{-1} \cdot (u^{-1} \cdot \mathcal{L})$.



(a) Automate $\mathcal{A}_1$



(b) Automate $\mathcal{A}_2$

Exercice 2 : Automates déterministes et non déterministes
Un automate est déterministe si, à chaque étape, quand il lit une lettre, ne peut passer que dans un seul état au plus, c'est à dire n'a qu'une seule transition possible au plus.

1. L'automate $\mathcal{A}_1$ est-il un automate déterministe ?
2. Et l'automate $\mathcal{A}_2$?
3. $\mathcal{A}_1$ est-il complet ou émondé ? et $\mathcal{A}_2$?
4. Dessinez l'automate $\mathcal{A}_2'$ obtenu à partir de l'automate $\mathcal{A}_2$ en renversant le sens de chaque flèche. L'automate $\mathcal{A}_2'$ est-il déterministe ? complet ou émondé ?
5. Après lecture du mot ab, dans quel(s) état(s) peut-il se trouver ? Même question après lecture du mot ba.
6. D'après vous, quels sont les mots reconnus par $\mathcal{A}_2'$?

Exercice 3 : Listes chaînées (rappel d'IF122). Les mots finis sur un alphabet sont des séquences de caractères. Pour ce TD, nous souhaitons représenter de telles séquences à l'aide de listes chaînées.

1. Écrire une classe `Liste` permettant de représenter des listes simplement chaînées contenant des données de type `Object`.
2. Écrire les constructeurs `Liste()` et `Liste(Object o, Liste l)` permettant de construire une liste vide, et une liste de premier élément `o` et d'éléments suivants `l`

3. Écrire la méthode `boolean estVide()` qui teste si une liste ne contient aucun élément, et les méthodes `Object tete()` et `Liste queue()` qui renvoient respectivement le premier élément de la liste, et la liste privée de son premier élément.
4. Écrire les méthodes suivantes : `int longueur()` qui renvoie la longueur de la liste, `Object element(int i)` qui renvoie le i-ème élément de la liste, et `Liste sousListe(int i, int j)` qui renvoie la portion de la liste située entre les indices `i` et `j` de la liste courante.
5. Écrire les méthodes `int indiceDe(Object o)` qui renvoie l'indice de première occurrence de l'objet `o` s'il apparait dans la liste, et -1 sinon, et `boolean equals(Liste l)` qui teste si la liste courante et la liste `l` sont identiques.
6. Écrire la méthode `String toString()` qui convertit la liste en chaîne de caractères prête à être affichée.

Exercice 4 : Conjugaison

Deux mots u et v sont dits *conjugués* s'il existe deux mots w_1 et w_2 tels que $u = w_1 \cdot w_2$ et $v = w_2 \cdot w_1$. En d'autres termes, v s'obtient à partir de u par permutation cyclique de ses lettres.

1. Montrer que la conjugaison est une relation d'équivalence, c'est-à-dire :
   - tout mot u est conjugué à lui-même ;
   - si u est conjugué à v, alors v est conjugué à u ;
   - si u est conjugué à v et v est conjugué à w, alors u est conjugué à w.
2. Montrer que u et v sont conjugués si et seulement s'il existe un mot w tel que $u \cdot w = w \cdot v$.