

TP5 JFLAP

Informatique théorique et automates

L3 CDA

Langages rationnels

Exercice 1

Dans cet exercice nous souhaitons construire l'automate fini minimal reconnaissant le langage :

$$L = (x + y)^* \setminus (x + y)^* xyx(x + y)^*$$

1. Construire l'automate reconnaissant le langage $L_1 = (x + y)^*$ et le sauvegarder.
2. Construire l'automate reconnaissant le langage $L_2 = (x + y)^* xyx(x + y)^*$ et le sauvegarder.
3. Utiliser le fait que $L_1 \setminus L_2 = L_1 \cap \overline{L_2} = \overline{\overline{L_1} \cup L_2}$ pour construire l'automate reconnaissant $L_1 \setminus L_2$.
4. Le minimiser.

Exercice 2

Dans cet exercice nous allons comparer les langages $(x + yy + yx(y + xx)^* xy)^* yx(y + xx)^*$ et $x^* y(yx^* y)^* x(y^* x(yx^* y)^* x)^* y^*$ en s'aidant de JFLAP.

1. Construire l'automate reconnaissant le langage $L_1 = (x + yy + yx(y + xx)^* xy)^* yx(y + xx)^*$, le déterminer, le minimiser et le sauvegarder.
2. Construire l'automate reconnaissant le langage $L_2 = x^* y(yx^* y)^* x(y^* x(yx^* y)^* x)^* y^*$, le déterminer, le minimiser et le sauvegarder.
3. Si vous n'êtes pas convaincus que les deux automates précédents sont les mêmes (les étiquettes sous les états sont différentes) vous pouvez aller dans **Test** et faire **Compare Equivalence**.

Exercice 3

Soit $L_1 = (b^*aa^*b(aa^*b)^*b)^*b^*aa^*b(aa^*b)^*$ et $L_2 = b^*aa^*b(a+b)^*$ et nous souhaitons montrer que $L_1 \subset L_2$. De manière générale un langage L_1 est inclus dans un langage L_2 si et seulement si $L_1 \setminus L_2 = \emptyset$ c'est-à-dire (pour que cela soit exploitable avec JFLAP) si et seulement si $\overline{L_1} \cup L_2 = \emptyset$.

1. Construire l'automate associé à L_1 , le déterminer, le minimiser et le sauvegarder.
2. Construire l'automate associé à L_2 , le déterminer, le minimiser et le sauvegarder.
3. Complémenter l'automate reconnaissant L_1 .
4. Faire l'union de l'automate précédent avec celui reconnaissant L_2 , le déterminer, le minimiser et le compléter. Normalement on obtient un automate réduit à un seul état non accepteur qui est bien l'automate reconnaissant le langage vide \emptyset . Si vous essayez de le convertir en un langage régulier JFLAP refuse (il lui faut au moins un état accepteur)!
5. Proposer des expressions régulières plus simples pour L_1 et L_2 et prouver qu'elles sont équivalentes aux premières.

Exercice 4

Soient $L_1 = (c + bb^*c + aa^*c)^*(\varepsilon + bb^* + aa^*)$ et $L_2 = (b + c + aa^*c)^*(\varepsilon + aa^*)$.

1. Montrer que $L_1 \subset L_2$.
2. Décrire en une phrase chacun des langages L_1 et L_2 .

Exercice 5 : pumping lemma

Le pumping lemma est un moyen pour essayer de prouver qu'un langage infini n'est pas rationnel. Le principe est le suivant. Si on suppose que le langage qui nous intéresse est rationnel alors il est reconnu par un automate fini. Cet automate possède q états ($q > 0$). Alors si on prend un mot w de longueur supérieure ou égale à q on est sûr qu'au moins un état va être visité deux fois lors de la lecture du mot. Le mot va pouvoir être décomposé sous la forme $w = xyz$ avec y le sous-mot de longueur supérieure ou égale à 1 qui permet de visiter deux fois le même état (on parcourt donc une boucle dans le graphe de l'automate sur lecture de y). Par conséquent l'ensemble des mots xy^iz fait partie du langage également. Pour prouver que le langage n'est pas régulier il suffit (mais ce n'est pas toujours possible) alors d'exhiber une bonne valeur de i de telle manière que xy^iz ne soit pas dans le langage.

1. Dans la fenêtre d'accueil de JFLAP sélectionner **Regular Pumping Lemma** puis sélectionner le premier langage $L = \{a^n b^n, n \geq 0\}$. Aller ensuite dans **Explain**. Si on vous dit que le langage n'est pas rationnel vous allez pouvoir trouver une décomposition (appelée

partition dans **JFLAP**) de w . Par exemple choisir $m = 5$, puis $y = a$ (avec le slider) et faire **Set xyz** et il choisit pour vous $i = 0$ et on n'a plus autant de a que de b .

2. Pour chacun des langages proposés dans **Pumping Lemma**, si le langage n'est pas rationnel (voir cela grâce à **Explain**) trouver une partition qui permet de générer un mot qui n'est pas dans le langage, sinon trouver une expression rationnelle pour exprimer le langage (ce qui prouvera qu'il est bien rationnel).