MITRO210: Automates et données structurées

Feuille d'exercices 5

Antoine Amarilli

1 Quelques exemples sur le monoïde de transition

Dans cet exercice, on se familiarise avec la notion de monoïde de transition en l'instanciant sur des exemples.

On fixe l'alphabet $\Sigma = \{a, b\}$ qui sera utilisé dans tout l'exercice.

Question 0. Construire l'automate déterministe complet minimal du langage $L_0 = a^*b^*$.

Question 1. Calculer explicitement le monoïde de transition M_0 du langage L_0.

Question 2. Montrer que L_0 est un langage sans étoile. Qu'en déduire sur M_0 ?

Question 3. Construire l'automate déterministe complet minimal du langage $L_3 = (ab)^*$ et calculer le monoïde de transition M_3 du langage L_3. Est-il apériodique ?

Question 4. Calculer les monoïdes de transition des langages a^* et $\Sigma^*b\Sigma$. Qu'observe-t-on ?

Question 5. Construire l'automate déterministe complet minimal du langage des mots de longueur multiple de 3. Calculer son monoïde de transition. Est-il apériodique ? Comment peut-on le décrire ?

Question 6. Soit F un langage fini. Expliquer comment construire un automate déterministe complet A qui reconnaisse F. Comment décrire le monoïde de transition ? Quelle propriété a-t-il, en fonction de la longueur maximale d'un mot de F ?

2 Quelques propriétés du monoïde de transition

On cherche à présent à prouver quelques propriétés simples du monoïde de transition, en particulier son lien avec l'équivalence syntaxique.

On fixe un automate A sur l'alphabet $\Sigma = \{a, b\}$.

Question 0. Pour deux mots $u, v \in \Sigma^*$, on note $u \sim v$ si l'élément du monoïde de transition de A associé à u et à v est le même. Montrer que cette relation est une relation d'équivalence sur Σ^*.

Question 1. Montrer que, pour tous $u, v \in \Sigma^*$, si $u \sim v$ alors on a $u \in L$ si et seulement si $v \in L$.

Question 2. Montrer que $\sim$ est une *congruence* de L, c'est-à-dire une relation d'équivalence avec la propriété additionnelle suivante : pour tous mots $x, z \in \Sigma^*$, si $y \sim y'$ pour deux mots $y, y' \in \Sigma^*$ alors $xyz \sim xy'z$.

Question 3. On introduit la relation d'*équivalence syntaxique* $\equiv$ du langage L comme la relation sur Σ^* définie comme suit : pour deux mots $y, y' \in \Sigma^*$, on a $y \equiv y'$ si et seulement si pour tous mots $x, z \in \Sigma^*$, on a $xyz \in L$ si et seulement si $xy'z \in L$. Montrer que la relation d'équivalence syntaxique est une congruence.

Question 4. Montrer que, pour tout langage L, si l'on pose A l'automate déterministe complet minimal reconnaissant L et $\sim$ la relation d'équivalence définie par le monoïde de transition, si l'on pose $\equiv$ la relation d'équivalence syntaxique définie en question précédente, alors $\sim$ et $\equiv$ sont exactement les mêmes relations : pour tous mots $u, v \in \Sigma^*$, on a $u \sim v$ si et seulement si $u \equiv v$.

3 Requêtes d'appartenance de facteur

*Dans cet exercice, on cherche à adapter la construction présentée en cours pour la maintenance incrémentale, afin de s'en servir pour répondre à des requêtes dites d'*appartenance de facteur *: on souhaite calculer une structure de données sur un mot qui permettra ensuite de savoir rapidement, pour tout facteur, s'il est ou non dans le langage.*

On se fixe un langage L, qui sera considéré comme constant dans les études de complexité pour cet exercice. Étant donné un mot $w = a_1 \cdots a_n$ de longueur n de Σ^* stocké dans un tableau, on souhaite précalculer une structure de données permettant de répondre efficacement à des requêtes dites d'*appartenance de facteur*. Une telle requête consiste en un intervalle $[i, j[$ avec $1 \leq i \leq j \leq n+1$, et pour répondre à la requête il faut indiquer si le facteur $a_i \cdots a_{j-1}$ de w décrit par cet intervalle appartient à L ou non.

Question 0. Si on ne souhaite effectuer aucun précalcul, avec quelle complexité peut-on répondre à une requête d'appartenance de facteur ?

Question 1. Si on autorise un précalcul arbitraire, avec quelle complexité peut-on répondre à une requête d'appartenance de facteur ?

Question 2. On suppose que le langage L est fini. Comment peut-on efficacement répondre aux requêtes d'appartenance de facteur ?

Dans le reste de l'exercice, on ne fait plus d'hypothèse sur L. On suppose que $n = 2^l$ pour un certain l, et on considère comme en cours un arbre binaire complet T de hauteur l ayant 2^l feuilles numérotées de 1 à 2^l et en bijection avec les lettres de w. On définit pour chaque nœud n de T l'intervalle ι_n des feuilles accessibles comme dans la construction vue en cours.

Question 3. Étant donné un intervalle non-vide arbitraire $[i, j[$ avec $1 \leq i < j \leq n + 1$, expliquer comment trouver une séquence $n_1, \ldots, n_k$ de nœuds de T tels que les intervalles $\iota_{n_1}, \ldots, \iota_{n_k}$ soient deux à deux disjoints, ordonnés de gauche à droite, et que leur union soit exactement $[i, j[$. Quelle borne peut-on montrer sur le nombre de nœuds d'une telle séquence, et avec quelle complexité peut-on trouver de tels nœuds ?

Question 4. En déduire un algorithme pour répondre efficacement aux requêtes d'appartenance de facteur. Préciser la complexité en temps nécessaire pour le précalcul d'une part, et pour la réponse à chaque requête d'autre part.

Question 5 (bonus). Proposer un algorithme avec un précalcul linéaire qui permette de répondre aux requêtes d'appartenance de facteur en temps constant. Quel désavantage a cet algorithme par rapport à celui de la question précédente ?