Langages et Automates : LA3
Partie 5 : Problèmes Algorithmiques - Lemme de l'Étoile

Problèmes Algorithmiques - Reconnaissance

Problème

Entrée : un mot $w = a_1...a_n$ et une E.R.
Sortie : OUI si et seulement si w appartient au langage décrit par E.R.

On utilise un des algos du cours (ex. Glushkov) pour transformer l'E.R. en automate fini équivalent.
Si on déterminise, on prend le risque d'avoir un automate deterministe de taille exponenetiellement grande par rapport à la taille de l'E.R., donc ce n'est pas la bonne méthode. Il faut savoir donc résoudre le problème efficacement pour un automate non-déterministe.

Problème

Entrée : un mot $w = a_1...a_n$ et un automate déteministe.
Sortie : OUI si et seulement si w est accepté par l'automate.

Facile : rapide, temps de calcul est proportionnel à la longueur du mot, il suffit d'effectuer l'unique calcul du mot par l'automate et voir si la destination est un état acceptant. Notons que cela ne dépend pas de la taille de l'automate.

Problème

Entrée : un mot $w = a_1...a_n$ et un automate non déterministe A
Sortie : OUI si et seulement si w est accepté par l'automate.

Ici l'automate n'est pas déterministe : plus compliqué (déterminiser l'automate est une mauvaise idée car on sait que cette opération peut prendre un temps exponentiel en la taille de l'automate)

Algo de reconnaissance pour non déterministe (en temps $|w||Q_A|^2$)

$E \leftarrow Initiaux(A)$
Pour i de 1 à n
    $E' \leftarrow \emptyset$
    Pour tout q dans E
        $E' \leftarrow E' \cup \delta(q, a_i)$
    $E \leftarrow E'$
retourne OUI si $F \cap E \neq \emptyset$, NON sinon

Problèmes Algorithmiques

On a vu dans le cours des algorithmes sur les automates les opérations suivantes :

- intersection, union, complémentaire, difference,
- miroir, Prefixe, Suffixe, Facteur, Sous-mot...

De plus on aussi vu des algorithmes pour passer de E.R a automate et réciproquement. On a donc en particulier des algorithmes pour calculer sur les E.R toutes ces opérations.
(Attention cependant, le passage automate vers E.R peut engendrer des E.R de taille exponentielle en la taille de l'automate)

Problèmes Algorithmiques - Inclusion

Pour résoudre un problème du type :

Entrée : Deux langages rationnels L_1 et L_2 décrits par E.R
Sortie : OUI si et seulement si $L_1 = L_2$ (ou $L_1 \subset L_2$, ou L_1 et L_2 disjoints)

il suffit de savoir efficacement le problème suivant :

Entrée : Un automate $\mathcal{A}$
Sortie : OUI si et seulement le langage reconnu par A est non vide

Cela vient des propriétés de cloture ensemblistes et du fait qu'on peut écrire par exemple

$$\begin{aligned} L_1 = L_2 &\Leftrightarrow L_1 \subset L_2 \ \ ET \ \ L_2 \subset L_1 \\ &\Leftrightarrow (L_1 \cap \overline{L_2}) = \emptyset \ \ ET \ \ (L_2 \cap \overline{L_1}) = \emptyset \\ &\Leftrightarrow (L_1 \cap \overline{L_2}) \cup (L_2 \cap \overline{L_1}) = \emptyset \end{aligned}$$

Problèmes Algorithmiques

Pour un gros automate, comment résoudre la question suivante :

Le langage décrit par cet automate est-il non vide ?

On se moque des lettres sur les transitions, on veut juste savoir s'il existe un chemin partant de l'état initial vers un des états finaux.
$\rightarrow$ Problème de graphes orientés

Réponse : On peut le faire en temps proportionnel à la taille de l'automate (son nombre de transitions) par un parcours en largeur (voir cours algorithmique). Cela marche aussi sur un automate non deterministe (il n'est donc ni nécessaire, ni malin de le déterminiser pour répondre çà cette question)

Lemme d'Itération (ou lemme de l'étoile)

On ca ici chercher à montrer que certains langages NE sont PAS rationnels.
On va raisonner par l'absurde en utlisant un théoèreme qui dit : "Si L rationnel, alors ...." et arriver à une contradiction.

Lemme d'Itération (ou lemme de l'étoile)

Observation

Si un automate a N états et u est un mot accepté par cet automate de longueur au moins N, alors lors du calcul de ce mot par l'automate on va nécessairement passer deux fois par un même état, et on peut même garantir que cela se produit avant d'avoir lu la $(N+1)$ième lettre de u.

Lemme d'Itération (ou lemme de l'étoile)

Observation

Si un automate a N états et u est un mot accepté par cet automate de longueur au moins N, alors lors du calcul de ce mot par l'automate on va nécessairement passer deux fois par un même état, et on peut même garantir que cela se produit avant d'avoir lu la $(N+1)$ième lettre de u.

Soit u un tel mot et soit q un état par lequel le calcul passe deux fois. On peut donc décomposer u en $u_1.u_2.u_3$ de telle façon que :

- $\delta^*(q_0, u_1) = q$
- $\delta^*(q, u_2) = q$
- $\delta^*(q, u_3) = q_f$ où q_f état final de l'automate.
- $u_2 \neq \varepsilon$ et $|u_1.u_2| \leq N$

ainsi, lorsqu'on lit le mot u_2 a partir de l'état q, on revient dans l'état q.
Ceci implique que pour tout entier k, le mot $u_1.(u_2)^k.u_3$ est un mot accepté par l'automate.

Lemme d'Itération (ou lemme de l'étoile)

Application

Le langage $L = \{a^n b^n,\ n \in \mathbb{N}\}$ n'est pas un langage reconnaissable.

Preuve

Supposons par l'absurde que $\mathcal{A}$ est un AFD reconnaissant L.
Si N est le nombre d'états de $\mathcal{A}$, l'observation précédente s'applique au mot $a^N b^N$.
Il existe donc u_1, u_2, u_3 tels que $u_1.u_2.u_3 = a^N b^N$ et

- $u_2 \neq \varepsilon$ et $|u_1.u_2| \leq N$
- Pour tout entier k, $u_1.(u_2)^k.u_3 \in L$

En particulier pour $k = 0$, le mot $u_1 u_3$ doit appartenir à L.
Puisque $|u_1.u_2| \leq N$, on sait que u_2 ne contient que des a. Mais alors le mot $u_1 u_3$ NE PEUT PAS être de la forme $a^n b^n$ (il a moins de a que de b).
On obtient une contradiction et le langage L n'est donc pas reconnaissable.

Lemme d'Itération (ou lemme de l'étoile)

De façon plus formelle, l'observation précédente peut s'énoncer

Lemme (Lemme d'Itération)

Si L est un langage reconnaissable, alors il existe un entier N tel que pour tout mot m de L de longueur au moins N, il existe une décomposition en $m = u.v.w$ avec $|uv| \leq N$ et v non vide tel que tout mot de la forme $u.v^k.w$ est aussi dans L.

De façon encore plus formelle :

Lemme (Lemme d'Itération)

Si L est un langage reconnaissable, alors $\exists N \in \mathbb{N}$ tel que :
$\forall u \in L$ tel que $|u| \geq N$, $\exists (u_1, u_2, u_3)$ tels que :

- $u = u_1.u_2.u_3$
- $u_2 \neq \varepsilon$ et $|u_1.u_2| \leq N$
- $\forall k \in \mathbb{N}$, $u_1.(u_2)^k.u_3 \in L$

Dans les deux cas, il faut comprendre le N comme la taille d'un automate reconnaissant L.

Lemme d'Itération (ou lemme de l'étoile)

Pour montrer que certains langages ne sont pas reconnaissables, on utilise une version encore plus fine de ce lemme : le motif qui se répète n'intervient pas que dans les N premières lettres du mot mais dans n'importe quelle "fenêtre" de taille N dans le mot.

Lemme (Lemme d'Itération)

Si L est un langage reconnaissable, alors $\exists N \in \mathbb{N}$ tel que $\forall u \in L$, $\forall u', u'', u'''$ tels que $u = u'.u''.u'''$ avec $|u'| \geq n$, $\exists (u_1, u_2, u_3)$ tels que :

- $u'' = u_1.u_2.u_3$
- $u_2 \neq \varepsilon$ et $|u_1.u_2| \leq N$
- $\forall k \in \mathbb{N}$, $(u'.u_1.(u_2)^k.u_3.u''') \in L$

Exercice

Montrer que le langage $\{a^m b^n, m \geq n\}$ n'est pas reconnaissable

Lemme d'Itération - Remarques

- Le lemme n'est pas une équivalence : Il existe des langages non reconnaissables qui vérifient la propriété, comme par exemple $\{u.mir(u).v,\ u \neq \varepsilon, v \neq \varepsilon\}$ ou $\{u \in \{a,b\}^*, |u|_a = |u|_b\}$ (exercice)
- Une idée générale à retenir : Les AFD sont des machines a mémoire finie. Ils ne savent pas compter.

Lemme d'Itération - Conséquences

Avec le raisonnement du Lemme on peut aussi montrer le résultat suivant :

Proposition

Si L est reconnu par A à n états, L est infini si et seulement si il existe un mot de L de longueur comprise entre n et $2n$.

Ce résultat implique qu'on peut résoudre le problème algorithmique suivant :

Cardinalité k

Entrée : Une expression rationnelle E.
Sortie : OUI si E reconnait exactement k mots, NON sinon

On construit un automate équivalent, on note n son nombre d'états. On teste les mots de longueur inférieure à $2n$. Si exactement k ont longueur plus petite que n et 0 ont longueur entre n et $2n$, on rénpod OUI, et NON sinon

Langages non reconnaissables et Clotures

Attention, l'intersection ou l'union de deux langages non reconnaissables n'est pas nécessairement non reconnaissable.

Cependant on peut utiliser aussi les propriétés de cloture des langages reconnaissables pour prouver qu'un langage n'est pas reconnaissable.

Exemple : $\{u \in \{a,b\}^*, |u|_a = |u|_b\}$ n'est pas reconnaissable.
En effet si on note

- $L_1 = \{u \in \{a,b\}^*, |u|_a = |u|_b\}$
- $L_2 = \{a^n b^n, n \in \mathbb{N}\}$
- $L_3 = \{a^p b^q,\ (p,q) \in \mathbb{N}^2\}$

Or,

- L_2 n'est pas reconnaissable
- L_3 est reconnaissable (correspond à l'expression rationnelle a^*b^*)
- $L_2 = L_1 \cap L_3$

Puisque l'interstion de deux langages reconnaissables est reconnaissables, on en déduit que L_1 n'est pas reconnaissable (sinon L_2 le serait).