

Automates finis

1. Langages et automates

1.1. Problème de décision

Soient L_1 et L_2 deux langages sur un alphabet A .

On définit le *langage produit* (ou *concaténation*) par :

$$L_1 \cdot L_2 = \{w \in A^* \mid w = u \cdot v, u \in L_1, v \in L_2\}$$

On note L^0 le langage vide (i.e. $\{\varepsilon\}$).

On définit L^n (avec $n \in \mathbb{N}^*$) tel que : $L^n = L \cdot L^{n-1}$ pour tout $n > 0$

L'*étoile* d'un langage L est, pour tout n dans \mathbb{N} :

$$L^* = \bigcup_{i=0}^n L^i$$

On définit également L^+ , pour tout n dans \mathbb{N}^* :

$$L^+ = \bigcup_{i=1}^n L^i$$

Par exemple, si on a $L_1 = \{a, ab\}$ et $L_2 = \{c, bc\}$, alors :

- $L_1 \cdot L_2 = \{ac, abc, abbc\}$
- $L_1^2 = \{aa, aab, aba, abab\}$

Pour un problème de *décision*, les mots sont une façon de représenter les *données* du problème. Un langage permet de représenter les *solutions* du problème. Ainsi, on associe à tout problème de décision P le langage L_P des solutions de P .

1.2. Problème du mot et automates

Étant donné un langage $L \subseteq A^*$ et un mot u de A^* , est-ce que u est dans L ? (Problème du mot)

Pour répondre à ce problème, on utilise un **automate**. Il s'agit d'un modèle de programme simple. Il reçoit en entrée un mot qu'il lit lettre à lettre et change ses états (qui sont en nombre *fini* !) en fonction de ces entrées. À la fin de son exécution, l'état dans lequel se trouve l'automate détermine si le mot lu en entrée appartient au langage recherché.

Sur un alphabet A , un automate fini est donné par $\mathcal{A} = (S, T, I, F)$ où :

- S est un ensemble fini (non vide) d'états
- $T \subseteq S \times A \times S$ est une relation de transition
- $I \subseteq S$ est l'ensemble (non vide) des états initiaux
- $F \subseteq S$ est l'ensemble des états finaux

Par exemple, sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{\text{pair}} = \{\text{nombres pairs}\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.

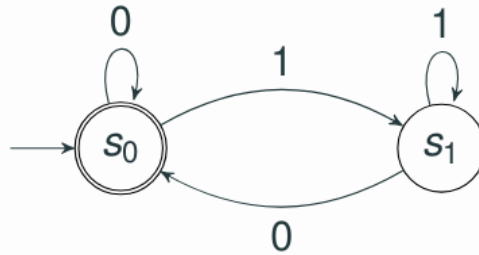


Figure 1: Un automate permettant de résoudre le problème du mot pour le langage L_{pair} .

Une exécution de \mathcal{A} est une séquence finie $s_0 a_1 s_1 \dots a_n s_n$ telle que :

- $s_0 \in I$ est un état initial
- pour tout $0 \leq i < n$, alors $(s_i, a_{i+1}, s_{i+1}) \in T$ est une transition autorisée par la relation de transition

La séquence $a_1 \dots a_n$ est un mot de A qui étiquette l'exécution. On dit qu'une exécution est *acceptante* si $s_n \in F$. Un mot est *accepté* par \mathcal{A} s'il est étiquette d'au moins une exécution acceptante.

Le langage d'un automate \mathcal{A} est noté $L(\mathcal{A})$:

$$L(\mathcal{A}) = \{u \in A^* \mid u \text{ est accepté par } \mathcal{A}\}$$

On dit que deux automates \mathcal{A} et \mathcal{B} sont *équivalents* si $L(\mathcal{A}) = L(\mathcal{B})$.

Un langage $L \subseteq A^*$ est *reconnaissable* s'il existe un automate fini \mathcal{A} sur l'alphabet A tel que :

$$L = L(\mathcal{A})$$

2. Automates déterministes, non déterministes et complets

2.1. Déterminisme

2.1.1. Définitions

Un automate est dit déterministe si chaque exécution produit la même étiquette. D'une manière formelle, un automate est déterministe si :

- il a un unique état initial
- la relation R (les transitions de l'automate) est fonctionnelle au sens suivant :

$$(p, a, q) \in R \wedge (p, a, q') \in R \implies q = q'$$

Dans un automate fini déterministe, T est une *fonction* allant de $S \times A$ vers S . On note parfois

$$T(s, a) = s'$$

pour $(s, a, s') \in T$ dans un automate déterministe.

Dans un automate fini déterministe, tout mot est étiquette d'au plus une exécution.

Figure 1 est un automate déterministe.

Un automate dit non déterministe s'il n'est pas déterministe.

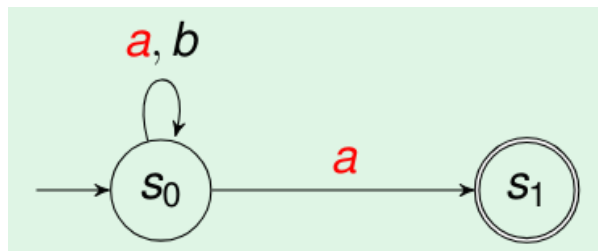


Figure 2: Un automate non déterministe

2.1.2. Déterminiser

Tout automate fini est équivalent à un automate fini déterministe.

⚠ Un automate déterministe ne possède qu'un unique état d'entrée !

Soit $\mathcal{A} = (S, T, I, F)$ un automate fini sur un alphabet A . On définit $\det(\mathcal{A}) = (\mathcal{P}(S), T', I, F')$ avec

- pour X dans $\mathcal{P}(S)$, pour $a \in E$, on a $T'(X, a) = \{s' \in S \mid \exists s \in X, (s, a, s') \in T\}$
- $F' = \{X \subseteq S \mid X \cap F \neq \emptyset\}$
- $\det(\mathcal{A})$ set déterministe
- $L(\mathcal{A}) = L(\det(\mathcal{A}))$
- *il manque un point*

2.2. Complétude

Un automate $\mathcal{A} = (S, T, I, F)$ sur un alphabet A est *complet* si :

$$\forall s \in S, \forall a \in A, \exists s' \in S, (s, a, s') \in T$$

Dans un automate complet, tout mot est étiquette d'au moins une exécution.

Tout automate fini est équivalent à un automate complet.

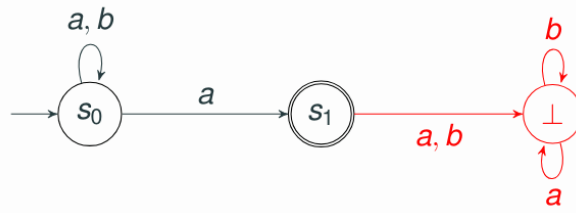


Figure 3: Automate de Figure 1 complété

Soit $\mathcal{A} = (S, T, I, F)$ un automate fini sur l'alphabet A . On construit

$$\text{comp}(\mathcal{A}) = (S \uplus \{\perp\}, T', I, F)$$

avec

$$T' = T \uplus \{(s, a, \perp) \mid s \in S \uplus \{\perp\}, a \in A, \forall s' \in S, (s, a, s') \notin T\}$$

On a que :

- $L(\text{comp}(\mathcal{A})) = L(\mathcal{A})$
- $\text{comp}(\mathcal{A})$ est complet

3. Propriétés de clôture

Si $L \subseteq A^*$ est un langage reconnaissable, alors \overline{L} est reconnaissable.

Soit \mathcal{A} tel que $L(\mathcal{A}) = L$. Sans perte de généraliste, on suppose que \mathcal{A} est complet et déterministe (sinon on le complète et on le détermine).

Si $\mathcal{A} = (S, T, i, F)$, alors on construit $\mathcal{B} = (S, T, i, F')$ avec $F' = S \setminus F$.

- $L(\mathcal{B}) = \overline{L}$
- L'approche n'est correcte que si A est déterministe et complet !

Il s'agit du même automate où les états normaux deviennent finaux et les états finaux deviennent normaux. On les inverse.

Si L_1 et L_2 sont deux langages reconnaissables de A , alors $L_1 \cup L_2$ et $L_1 \cap L_2$ sont aussi reconnaissables.

compléter avec la diapo 26