Automates et Calculabilité

– Motifs et expressions régulières –

Leçon 5 : 17 nov 2003

Uwe Nestmann

EPFL-I&C

AFN$_\epsilon$: AFN avec ϵ-transitions

Un AFN$_\epsilon$ est une structure

$$M = (\mathcal{Q}, \Sigma \cup \{\boldsymbol{\epsilon}\}, \Delta, S, F)$$

qui fonctionne comme un AFN, mais sur l'alphabet $\Sigma \cup \{\, \boldsymbol{\epsilon} \,\}$.

L'idée étant qu'un tel automate peut,
à tout moment lors de la procédure d'acceptation d'un mot,
suivre une $\boldsymbol{\epsilon}$-transition sans lire le prochain symbole du mot,
comme si l'automate lisait le mot vide ϵ.

Attention: $\boldsymbol{\epsilon} \neq \epsilon$. $|\epsilon| = 0$. $|\boldsymbol{\epsilon}| = 1$. $|\boldsymbol{\epsilon\epsilon}| = 2$.

Avantages:

- permet de réduire le nombre d'états initiaux et finaux;
- permet de réduire le nombre de transitions.

ϵ-fermeture

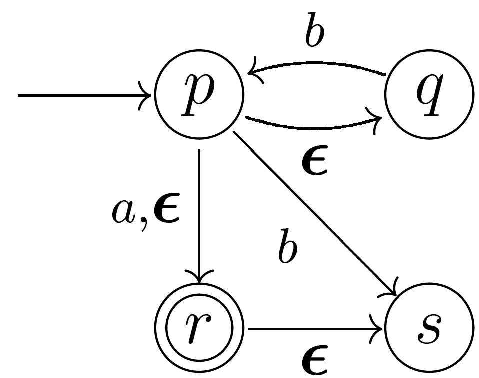
Soit $(\mathcal{Q}, \Sigma \cup \{\boldsymbol{\epsilon}\}, \Delta, S, F)$ un AFN$_{\boldsymbol{\epsilon}}$.

La ***ϵ-fermeture*** d'un ensemble d'états $P \subseteq \mathcal{Q}$ est définie par

$$\mathrm{C}_{\boldsymbol{\epsilon}}(P) \stackrel{\mathrm{def}}{=} \bigcup_{x \in \{\, \boldsymbol{\epsilon} \,\}^*} \widehat{\Delta}(P, x)$$

elle donne l'ensemble de tous les états atteignables par zéro, une, ou plusieurs ϵ-transitions à partir des états inclus dans P.

Exemple: Soit $\begin{array}{rcl} \mathcal{Q} & = & \{\, p, q, r, s \,\} \\ \Sigma & = & \{\, a, b \,\} \end{array}$ et S, F, Δ :



Calculer les ϵ-fermetures de tous les « singletons ».
Quel est le langage accepté par cet automate?

Langage accepté par un AFN$_\epsilon$

Soit $N = (\mathcal{Q}, \Sigma \cup \{\boldsymbol{\epsilon}\}, \Delta, S, F)$ un AFN$_{\boldsymbol{\epsilon}}$.

Nous définissons $\widehat{\Delta}_{\boldsymbol{\epsilon}} : 2^{\mathcal{Q}} \times \Sigma^* \to 2^{\mathcal{Q}}$ par :

$$\widehat{\Delta}_{\boldsymbol{\epsilon}}(P, \epsilon) \stackrel{\text{def}}{=} \mathrm{C}_{\boldsymbol{\epsilon}}(P) \qquad \widehat{\Delta}_{\boldsymbol{\epsilon}}(P, xa) \stackrel{\text{def}}{=} \mathrm{C}_{\boldsymbol{\epsilon}}\left(\bigcup_{q \in \widehat{\Delta}_{\boldsymbol{\epsilon}}(P,x)} \Delta(q, a)\right)$$

L'automate N accepte le mot $x \in \Sigma^*$ si

$$\widehat{\Delta}_{\boldsymbol{\epsilon}}(S, x) \cap F \neq \emptyset$$

Le *langage accepté par* N, dénoté par $L_{\boldsymbol{\epsilon}}(N)$,
est l'ensemble des mots qui sont acceptés par N.

$$L_{\boldsymbol{\epsilon}}(N) \stackrel{\text{def}}{=} \{\, x \in \Sigma^* \mid \widehat{\Delta}_{\boldsymbol{\epsilon}}(S, x) \cap F \neq \emptyset \,\}$$

Elimination des ϵ-transitions

Soit $N = (\mathcal{Q}, \Sigma \cup \{\boldsymbol{\epsilon}\}, \Delta, S, F)$ un AFN$_{\boldsymbol{\epsilon}}$.

Idée: Tous les états atteignables depuis un état p à l'aide des ϵ-transitions sont rendus accessibles sans celles-ci.

$$\textcircled{p} \xrightarrow{\quad\boldsymbol{\epsilon}^*\quad} \textcircled{q} \xrightarrow{\;a\;} \textcircled{r}$$

L'automate $N' = (\mathcal{Q}, \Sigma, \Delta', S, F')$ défini par

$$\begin{aligned} \Delta'(q,a) &\stackrel{\text{def}}{=} \widehat{\Delta}(\mathrm{C}_{\boldsymbol{\epsilon}}(\{q\}), a) \\ F' &\stackrel{\text{def}}{=} \{\, q \in \mathcal{Q} \mid \mathrm{C}_{\boldsymbol{\epsilon}}(\{q\}) \cap F \neq \emptyset \,\} \end{aligned}$$

élimine les ϵ-transitions en préservant le langage accepté.
Il existe d'autres procédures d'élimination . . .

Théorème: $L(N') = L_{\boldsymbol{\epsilon}}(N)$.

Propriétés de clôture (III)

Soit $A_1, A_2 \subseteq \Sigma^*$ deux langages réguliers,
il existe donc M_i tel que M_i accepte A_i pour $i \in \{1, 2\}$.
Alors,

- $A_1 A_2$ est régulier.

  Construire M_{12} par « concaténation » de M_1 et M_2.

  Théorème: $L_\epsilon(M_{12}) = A_1 A_2$.

- A_1^* est régulier.

  Construire M_1^* par « fermeture » de M_1.

  Théorème: $L_\epsilon(M_1) = A_1^*$.

Résumé

- AFD, AFN, et AFN$_\epsilon$ ont la même capacité,
  mesurée par la catégorie de langages acceptés par ceux-ci.
- Il est possible de construire l'un à partir de l'autre . . .
- Par rapport aux AFD, les AFN et AFN$_\epsilon$ simplifient souvent
  la définition d'un automate acceptant un langage donné.

Motivation

Il n'est pas toujours satisfaisant de définir un langage par

$$A = \{\, x \in \Sigma^* \mid \text{ l'avant-dernier symbole dans } x \text{ est un } 1 \,\}$$

où il faut utiliser des expressions assez compliquées (ou même le langage naturel) pour exprimer ce que l'on veut dire.

$$A = \{\, x \in \Sigma^* \mid \text{il existe } y \in \Sigma^*, z \in \Sigma \text{ tel que } x = y \cdot 1 \cdot z \,\}$$

Dans cette leçon, nous allons apprendre un nouveau formalisme — les ***expressions régulières*** — qui nous permet de définir certains langages de manière compacte, algébrique, et sans ambiguïtés.

Motifs avec UNIX

`rm *.pdf`

Principe du filtrage des motifs (« pattern matching ») :

0. ‘**`*`**’ est le motif qui correspond à la répétition d’un nombre fini des caractères qui peuvent être utilisés dans les noms de fichiers.
1. Reconnaître (ou *filtrer*) tous les noms des fichiers dans le répertoire actuel qui finissent par **`.pdf`**
2. Effacer tous ces fichiers . . .

Exercices peu recommandés :

`rm *`

`rm -rf *`

Motifs

Un motif α est *une chaîne* de symboles sur un alphabet Σ de forme particulière *qui représente un langage* $L(\alpha)$, c'est-à-dire un sous-ensemble de l'ensemble des mots Σ^*.

Pour définir la signification d'un motif, on dénote l'ensemble des mots que son langage contient :

$$L(\alpha) \stackrel{\text{def}}{=} \{\, x \in \Sigma^* \mid x \text{ est filtré par } \alpha \,\}$$

<u>**Exemple:**</u> (UNIX) :
$L($**`*.pdf`**$)$ correspond à l'ensemble des noms de fichiers du répertoire courant définis par une sequence de n'importe quel longueur (finie) de caractères se terminant par **`.pdf`**.

Motifs, définition inductive

L'ensemble des ***motifs atomiques*** est $\Sigma \cup \{\, \boldsymbol{\epsilon}, \boldsymbol{\emptyset}, \#_\Sigma, @_\Sigma \,\}$.
L'ensemble des ***motifs composés*** est défini par
les opérateurs unaires ${}^*, {}^+, \sim_\Sigma$, et les opérateurs binaires $+, \cap, \cdot$.

L'ensemble M_Σ des motifs sur Σ est défini par induction :

$$\frac{a \in \Sigma}{a \in \mathbf{M}_\Sigma} \qquad \frac{}{\boldsymbol{\epsilon} \in \mathbf{M}_\Sigma} \qquad \frac{}{\boldsymbol{\emptyset} \in \mathbf{M}_\Sigma} \qquad \frac{}{\#_\Sigma \in \mathbf{M}_\Sigma} \qquad \frac{}{@_\Sigma \in \mathbf{M}_\Sigma}$$

$$\frac{\alpha \in \mathbf{M}_\Sigma}{\alpha^* \in \mathbf{M}_\Sigma} \qquad \frac{\alpha \in \mathbf{M}_\Sigma}{\alpha^+ \in \mathbf{M}_\Sigma} \qquad \frac{\alpha \in \mathbf{M}_\Sigma}{\sim_\Sigma \alpha \in \mathbf{M}_\Sigma} \qquad \frac{\alpha \in \mathbf{M}_\Sigma \qquad \beta \in \mathbf{M}_\Sigma}{\alpha + \beta \in \mathbf{M}_\Sigma}$$

$$\frac{\alpha \in \mathbf{M}_\Sigma \qquad \beta \in \mathbf{M}_\Sigma}{\alpha \cap \beta \in \mathbf{M}_\Sigma} \qquad \frac{\alpha \in \mathbf{M}_\Sigma \qquad \beta \in \mathbf{M}_\Sigma}{\alpha \cdot \beta \in \mathbf{M}_\Sigma}$$

Attention

- Formellement, on devrait aussi rajouter la règle

$$\frac{\alpha \in \mathbf{M}_\Sigma}{(\alpha) \in \mathbf{M}_\Sigma}$$

  qui permet d'utiliser des parenthèses pour clarifier la portée syntaxique des opérateurs dans les motifs.

- $\mathbf{M}_\Sigma$ est lui-même un langage spécifique sur l'alphabet $(\Sigma \cup \{\, \boldsymbol{\epsilon}, \boldsymbol{\varnothing}, \#_\Sigma, @_\Sigma, +, \cap, \cdot, {}^*, {}^+, \sim_\Sigma, (,)\, \})^*$.

- *Priorité des opérateurs* :
  $\alpha + \beta\gamma = \alpha + (\beta\gamma)$
  $\alpha + \beta^* = \alpha + (\beta^*)$
  En cas de doute, utiliser les parenthèses ...

Langages representé par les motifs

Nous définissons $L : \mathbf{M}_\Sigma \to 2^{\Sigma^*}$.
Donc, pout tout $\alpha \in \mathbf{M}_\Sigma$, $L(\alpha) \subseteq \Sigma^*$ est un langage sur Σ.
Soit $a \in \Sigma$.

$$\begin{aligned}
L(a) &\stackrel{\text{def}}{=} \{\, a \,\} \\
L(\boldsymbol{\epsilon}) &\stackrel{\text{def}}{=} \{\, \epsilon \,\} \\
L(\boldsymbol{\varnothing}) &\stackrel{\text{def}}{=} \emptyset \\
L(\#_\Sigma) &\stackrel{\text{def}}{=} \Sigma \\
L(@_\Sigma) &\stackrel{\text{def}}{=} \Sigma^*
\end{aligned}$$

$$\begin{aligned}
L(\alpha + \beta) &\stackrel{\text{def}}{=} L(\alpha) \cup L(\beta) \\
L(\alpha \cap \beta) &\stackrel{\text{def}}{=} L(\alpha) \cap L(\beta) \\
L(\alpha \cdot \beta) &\stackrel{\text{def}}{=} L(\alpha)L(\beta) \\
L(\sim_\Sigma \alpha) &\stackrel{\text{def}}{=} \Sigma^* \setminus L(\alpha) \\
L(\alpha^*) &\stackrel{\text{def}}{=} L(\alpha)^* \\
L(\alpha^+) &\stackrel{\text{def}}{=} L(\alpha)^+
\end{aligned}$$

Conventions :
souvent, on écrit $\#, @, \sim$ à la place de $\#_\Sigma, @_\Sigma, \sim_\Sigma$.

Attention

- $\boldsymbol{\epsilon}$ et ϵ ont des significations différentes.
  ø et $\emptyset$ ont des significations différentes.

- Parfois, on écrit $x \in a^*b^*$
  mais l'on devrait écrire $x \in L(a^*b^*)$.

- Normalement, on écrit $\alpha\beta$ à la place de $\alpha \cdot \beta$.

Exemples (I)

Si $\Sigma = \{ a, b, c \}$, alors $L(\#) = \{ a, b, c \}$.
Si $\Sigma = \{ a, b, c, d, e \}$, alors $L(\#) = \{ a, b, c, d, e \}$.

Soit Σ un alphabet.

Pour tout α, nous avons $\epsilon \in L(\alpha^*)$, donc ϵ est filtré par α^*.

Si $x \in \Sigma^*$ est un mot sur Σ, alors x est aussi un motif.
Dans ce cas, $L(x) =$

$L((\alpha_1 \cdot \alpha_2) \cdot \alpha_3) = L(\alpha_1 \cdot (\alpha_2 \cdot \alpha_3))$ $\qquad \alpha_1 \cdot \alpha_2 \cdot \alpha_3$
$L((\alpha_1 + \alpha_2) + \alpha_3) = L(\alpha_1 + (\alpha_2 + \alpha_3))$ $\qquad \alpha_1 + \alpha_2 + \alpha_3$

Si $x_1, x_2 \ldots, x_n \in \Sigma^*$ sont des mots sur Σ,
alors $L(x_1 + x_2 + \ldots + x_n) =$

Motifs equivalents

On dit de deux motifs α et β qu'ils sont equivalents,
si $L(\alpha) = L(\beta)$.
Nous utilisons la notation $\alpha \stackrel{L}{=} \beta$.

Nous avons donc vérifié
que $(\alpha_1 \cdot \alpha_2) \cdot \alpha_3 \stackrel{L}{=} \alpha_1 \cdot (\alpha_2 \cdot \alpha_3)$, et
que $(\alpha_1 + \alpha_2) + \alpha_3 \stackrel{L}{=} \alpha_1 + (\alpha_2 + \alpha_3)$.

Exemples (II)

Soit $\Sigma = a, b$.
Exprimer $@a@a@a@$ en langage naturel.
Exprimer $@b@ab@a$ en langage naturel.

Soit $\Sigma = 0, 1$.
Exprimer $@1@1@1@$ en langage naturel.
Exprimer $\#\#\#\#\#\#\#\#\#\#$ en langage naturel.
Exprimer $@1\#$ en langage naturel.

Soit $\Sigma = a, b, c \ldots, z$.
Exprimer $\# \cap \sim a$ en langage naturel.
Exprimer $(\# \cap \sim a)^*$ en langage naturel.
Exprimer $(\# \cap \sim a)^* + @b(\# \cap \sim a)^*$ en langage naturel.

Soit $\Sigma = a, b$.

Vérifier que $(\# \cap \sim a)^* + @b(\# \cap \sim a)^* \stackrel{L}{=} \boldsymbol{\epsilon} + @b$.

Exemples (III)

Soit Σ.

Exprimer **ø***.
Exprimer $\sim(\#@)$.

Exprimer $\#^*$.

Exprimer $\alpha\alpha^*$.

Comparer $L(\alpha \cap \beta)$ et $L(\sim(\sim\alpha + \sim\beta))$.

Soit $\Sigma = \{\, a_1, a_2 \ldots, a_n \,\}$ un alphabet fini.
Comparer $L(a_1 + a_2 + \ldots + a_n)$ et $L(\#)$.

Soit $\Sigma = \{\, a, b, c \,\}$.
Exprimer $@a(\boldsymbol{\epsilon} + b)a\#$.

Expressions régulières

A l'usage, on aime avoir une large sélection de motifs.
Pour les preuves, on préfère avoir une sélection minimale.

Pour cela, l'ensemble $\mathbf{R}_\Sigma$ des expressions régulières est défini comme un sous-ensemble des motifs M_Σ. Il est caractérisé par les opérateurs $\Sigma \cup \{\, \boldsymbol{\epsilon}, \boldsymbol{\emptyset}, +, \cdot, {}^*, (,) \,\}$.

Par induction :

$$\frac{a \in \Sigma}{a \in \mathbf{R}_\Sigma} \qquad \frac{}{\boldsymbol{\epsilon} \in \mathbf{R}_\Sigma} \qquad \frac{}{\boldsymbol{\emptyset} \in \mathbf{R}_\Sigma}$$

$$\frac{\alpha \in \mathbf{R}_\Sigma}{\alpha^* \in \mathbf{R}_\Sigma} \qquad \frac{\alpha \in \mathbf{R}_\Sigma \qquad \beta \in \mathbf{R}_\Sigma}{\alpha + \beta \in \mathbf{R}_\Sigma} \qquad \frac{\alpha \in \mathbf{R}_\Sigma \qquad \beta \in \mathbf{R}_\Sigma}{\alpha \cdot \beta \in \mathbf{R}_\Sigma}$$

Langage d'une expression régulière

Le langage $L(\alpha)$ associé à une expression régulière α est défini exactement de la même manière que pour les motifs.

Soit $a \in \Sigma$.

$$
\begin{aligned}
L(a) &\stackrel{\text{def}}{=} \{\, a \,\} \\
L(\boldsymbol{\epsilon}) &\stackrel{\text{def}}{=} \{\, \epsilon \,\} \\
L(\boldsymbol{\emptyset}) &\stackrel{\text{def}}{=} \emptyset \\
L(\#_\Sigma) &\stackrel{\text{def}}{=} \Sigma \\
L(@_\Sigma) &\stackrel{\text{def}}{=} \Sigma^*
\end{aligned}
$$

$$
\begin{aligned}
L(\alpha + \beta) &\stackrel{\text{def}}{=} L(\alpha) \cup L(\beta) \\
L(\alpha \cap \beta) &\stackrel{\text{def}}{=} L(\alpha) \cap L(\beta) \\
L(\alpha \cdot \beta) &\stackrel{\text{def}}{=} L(\alpha) L(\beta) \\
L(\sim_\Sigma \alpha) &\stackrel{\text{def}}{=} \Sigma^* \setminus L(\alpha) \\
L(\alpha^*) &\stackrel{\text{def}}{=} L(\alpha)^* \\
L(\alpha^+) &\stackrel{\text{def}}{=} L(\alpha)^+
\end{aligned}
$$

Equivalence

Théorème:
Soit $A \subseteq \Sigma^*$. Les propositions suivantes sont équivalentes :

(1) Il existe un AFD M tel que $L(M) = A$.
(A est un ***langage régulier***.)

(2) Il existe une ***expression régulière*** α tel que $L(\alpha) = A$.

(3) Il existe un motif α tel que $L(\alpha) = A$.

Preuve :

- ☐ (2) $\Leftrightarrow$ (3)
- ☐ (2) $\Rightarrow$ (1)
- ☐ (1) $\Rightarrow$ (2)

$(2) \Rightarrow (1)$

Il faut démontrer que pour tout expression régulière α, il existe un AFN$_{\boldsymbol{\epsilon}}$ N tel que $L(M) = L(\alpha)$.

La démonstration se fait par induction sur la définition de $L(\alpha)$ qui se base sur la construction inductive des expressions α.

- 3 cas de base :
  - $a \in \Sigma$
  - $\boldsymbol{\epsilon}$
  - **ø**
- 1 opérateur unaire : α^*
- 2 opérateur binaires :
  - $\alpha + \beta$
  - $\alpha \cdot \beta$

$(2) \Rightarrow (1)$

Il faut démontrer que pour tout AFN$_\epsilon$ N,
il existe une expression régulière α telle que $L(M) = L(\alpha)$.