[SAK] p 74

908

Automates Finis . Exemples et Applications

[CAR] p 35

Motivations :

- Modèle le plus simple de machine (digicode, ...)
- Application au traitement de texte : analyse lexicale et recherche de motif
- Application en théorie des langages

I Qu'est-ce qu'un automate ?

1 : Définition et représentation d'automates finis

déf : Un automate A sur un alphabet Σ est un quintuplet $(Q, \Sigma, \delta, I, F)$ avec : Q ensemble fini des états, $I \subseteq Q$ ensemble des états initiaux, $F \subseteq Q$ ensemble des états finaux et $\delta : Q \times \Sigma \longrightarrow \mathcal{P}(Q)$ fonction de transition.

Ex :

Rq : Pour implémenter un automate, on peut utiliser une matrice de transition représentant la fonction de transition, ainsi que deux ensembles I et F.

déf : Un chemin dans un automate est une suite de transitions : $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} q_n$

Le mot $a_1 \cdots a_n$ est l'étiquette du chemin.

déf : Un chemin est acceptant si $q_0 \in I$ et $q_n \in F$. On dit alors que le mot $a_1 \cdots a_n$ est accepté par A.

[SAK] p61 - 62

déf : Le langage accepté par A est :

$$\mathcal{L}(A) = \{ w \in \Sigma^* \mid \exists p \in I, \exists q \in F, p \xrightarrow{w} q \}$$

déf : Un langage $L \subseteq \Sigma^*$ est dit reconnaissable s'il existe un automate fini A tel que $L = \mathcal{L}(A)$.

On note $Rec(\Sigma^*)$ l'ensemble des langages reconnaissables.

Ex : Pour l'automate précédent $\mathcal{L}(A) = \{ w \in \Sigma^* / w$ contient un nombre pair de a $\}$

2 : Propriétés des automates

déf : $q \in Q$ est accessible si $\exists p \in I$, $w \in \Sigma^*$ tel que $p \xrightarrow{w} q$,

$q \in Q$ est co-accessible si $\exists f \in F$, $w \in \Sigma^*$ tel que $q \xrightarrow{w} f$.

[SAK] p 74

déf : Un automate A est dit émondé si tous ses états sont accessibles et co-accessibles.

déf : Un automate A est dit complet si $\forall p \in Q$, $\forall a \in \Sigma$ $\delta(p, a) \neq \emptyset$.

[SAK] p 83

Ex : voir figure A.

prop : Pour tout automate fini, on peut construire un automate complet (resp. émondé) équivalent, qui reconnaît le même langage.

[SAK] p 74 / 84

Application : On peut décider si $\mathcal{L}(A) = \emptyset$ et si $\mathcal{L}(A)$ est infini [illegible]

[SAK] p 77

déf : On appelle ε-transition une transition étiquetée par le mot vide : $p \xrightarrow{\varepsilon} q$.

prop : Pour tout automate avec ε-transitions, on peut construire un automate sans ε-transitions qui reconnaît le même langage.

[SAK] p 86

3 : Déterminisation

déf : Un automate est déterministe si $|I| = 1$ et $\forall p \in Q$, $\forall a \in \Sigma$, $|\delta(p, a)| \leq 1$.

déf : Pour tout automate $A = (Q, \Sigma, \delta, I, F)$, on définit l'automate des parties qui lui est associé :

$$A_p = (\mathcal{P}(Q), \Sigma, \delta_p, \{I\}, F_p)$$

avec $F_p = \{ P \subseteq Q / P \cap F \neq \emptyset \}$, $\delta_p(P, a) = \{ q \in Q / \exists p \in P, q \in \delta(p, a) \}$

[CAR] p 42 - 43

prop : L'automate des parties est déterministe.

Thm : Pour tout automate fini, on peut construire un automate fini déterministe (et complet) qui reconnaît le même langage.

Ex : voir figure B.

Rq : La construction de l'automate déterministe équivalent a une complexité exponentielle et c'est optimal.

II Propriétés des langages reconnaissables

1 : Propriétés de clôture et langages rationnels

prop : Si $L, L' \in Rec(\Sigma^*)$, alors $L \cup L'$, $L \cap L'$, $\bar{L}$, L^*, $L.L' \in Rec(\Sigma^*)$

[SAK] p 67/68

[BBC] p 30

Cor : On peut décider de l'égalité et de l'inclusion de deux langages

[SAK] p133

Déf : On définit inductivement une expression rationnelle sur Σ par
- $\emptyset$ et a sont des expressions rationnelles $\forall a \in \Sigma$
- Si E et F sont deux expressions rationnelles alors $E+F$, $E.F$ et E^* sont des expressions rationnelles.

Déf : A chaque expression rationnelle E, on associe un langage $L(E)$ défini inductivement par :
- $L(\emptyset) = \emptyset$, $L(a) = \{a\}$
- $L(E+F) = L(E) \cup L(F)$, $L(E.F) = L(E).L(F)$, $L(E^*) = L(E)^*$

On dit que E dénote $L(E)$.

Déf : Un langage L est rationnel s'il existe une expression rationnelle E telle que $L = L(E)$.
On note $\mathrm{Rat}(\Sigma^*)$ l'ensemble des langages rationnels.

Ex : (Langages minces)
* Un langage est dit mince s'il est dénoté par une expression de la forme : $v_1 v_1'^* w_1 + \dots + v_n v_n'^* w_n$ avec $v_i, v_i', w_i \in \Sigma^*$
* Un langage L est dit à croissance bornée si le nombre de mots de L de longueur n est borné par une constante indépendante de n.

[SAK] p 94

[DEV 1] L est mince $\Leftrightarrow$ $L \in \mathrm{Rec}(\Sigma^*)$ et L est à croissance bornée

Thm : (de Kleene)
Soit Σ un alphabet fini. Un langage L de Σ^* est rationnel si et seulement si il est reconnaissable, ie
$$\mathrm{Rat}(\Sigma^*) = \mathrm{Rec}(\Sigma^*)$$

[SAK] p157

Rq : L'algorithme de Thompson permet, à partir d'une expression rationnelle, de construire un automate qui reconnaît le même langage.

Ex : Construction par algorithme de Thompson. Voir figure C

2: Lemme de l'étoile

Lemme 1 : Si L est reconnaissable, $\exists N \in \mathbb{N}$ tel que $\forall f \in L$, $|f| \geq N$, il existe une factorisation $f = u v w$ avec v non vide telle que $\forall n \in \mathbb{N}$, $u v^n w \in L$.

[SAK] p 78

Ex : • On peut montrer à l'aide du lemme 1 que $\{a^n b^n / n \in \mathbb{N}\}$ n'est pas reconnaissable
• Le langage $\{a^n b^n / n \in \mathbb{N}\} \cup \Sigma^* ba \Sigma^*$ satisfait le lemme 1 mais n'est pas reconnaissable.

Lemme 2 : Si $L \in \mathrm{Rec}(\Sigma^*)$, $\exists N \in \mathbb{N}$ tel que pour tout $f \in L$ et toute factorisation $f = g_1 h g_2$ avec $|h| \geq N$, il existe une factorisation $h = u v w$ avec v non vide telle que $\forall n \in \mathbb{N}$, $g_1 u v^n w g_2 \in L$

Ex : Le langage $\{(aab)^n (abb)^m / n \in \mathbb{N}\} \cup \Sigma^* \{aaa, bbb\} \Sigma^*$ vérifie le lemme 2 mais n'est pas reconnaissable.

Lemme 3 : Si $L \in \mathrm{Rec}(\Sigma^*)$, $\exists N \in \mathbb{N}$ tel que pour tout $f \in L$ et toute factorisation de la forme $f = u v_1 \dots v_N w$ où les v_i sont tous non vides, il existe (j,k) tel que $0 \leq j < k \leq N$ et $\forall n \in \mathbb{N}$, $u v_1 \dots v_j (v_{j+1} \dots v_k)^n v_{k+1} \dots v_N w \in L$

Ex : Le langage $\{a^{k_1} b a^{k_2} b \dots a^{k_n} b / \exists i, i \neq k_i\}$ satisfait le lemme 3 mais n'est pas reconnaissable.

[SAK] p128

Thm : $L \in \mathrm{Rec}(\Sigma^*) \Leftrightarrow L$ satisfait le lemme 3

Rq : la [illegible] de ce [illegible] la théorème d'Ehrenfeucht, Parikh et Rozenberg.

III Minimisation

1: Automate minimal et automate des résiduels

Déf : Soit L un langage et $u \in \Sigma^*$. On appelle résiduel de L par u l'ensemble $u^{-1}L = \{w \in \Sigma^* / uw \in L\}$
On note $R(L) = \{u^{-1}L / u \in \Sigma^*\}$ l'ensemble des résiduels de L.

Ex : $\Sigma = \{a, b\}$ et $L = \{w \in \Sigma^* / w$ contient au moins un $a\}$
$\varepsilon^{-1}L = L$, $a^{-1}L = \Sigma^*$, $b^{-1}L = L$ et $a^{-1}\Sigma^* = b^{-1}\Sigma^* = \Sigma^*$
On a alors $R(L) = \{L, \Sigma^*\}$

[BBC] p 312/317

Prop : Soit $L \in \mathrm{Rec}(\Sigma^*)$ et $A = (Q, \Sigma, \delta, i, F)$ un automate fini déterministe complet et accessible qui reconnaît L. En notant pour $q \in Q$, $L_q(A) = \{w \in \Sigma^* / \exists p \in F, \delta(q, w) = p\}$, on a :
$$\{L_q(A) / q \in Q\} = R(L) \quad (*)$$

Cor : L reconnaissable $\Leftrightarrow$ $R(L)$ fini

Rq : L'équation $(*)$ montre que tout automate déterministe complet reconnaissant L possède au moins $|R(L)|$ états.

Déf : Soit $L \in \mathrm{Rec}(\Sigma^*)$ et A un automate qui le reconnaît. A est dit minimal s'il est déterministe complet et s'il possède le nombre minimal d'états parmi les automates déterministes complets reconnaissant L.

Déf : Pour tout langage $L \subset \Sigma^*$, on définit l'automate des résiduels par : $A_L = (R(L), \Sigma, \delta_L, L, F_L)$
avec $F_L = \{u^{-1}L / u \in L\}$ et $\delta_L(u^{-1}L, a) = (ua)^{-1}L$

[CAR] p 46/48

Prop : Soit L [illegible]. Alors [illegible]

[illegible]

[illegible]

2 : [illegible]

Déf : Soit A un automate [illegible]

[illegible]

$$[illegible]$$

Déf : Soit [illegible] $\mathcal{A} = (Q, \Sigma, \delta, I, F)$, [illegible]

[illegible] $\mathcal{A}_{\equiv} = (Q_{\equiv}, \Sigma, \delta_{\equiv}, I, F_{\equiv})$

avec $F_{\equiv} = \{[q] \mid q \in F\}$ et $\delta_{\equiv}([q], a) = [\delta(q, a)]$

Prop : Soit [illegible]. Alors [illegible]

Déf : Soit $\mathcal{A}$ [illegible]

[illegible]

$$[illegible]$$

Prop : L'automate [illegible] est minimal

Algorithme de Moore :

Soit $\mathcal{A} = (Q, \Sigma, \delta, I, F)$ [illegible]

[illegible]

$$[illegible]$$

$$[illegible]$$

Prop : Si $\mathcal{A}$ est [illegible]

[illegible]

Rq : La complexité de l'algorithme [illegible]

[illegible]

- L'algorithme [illegible]

[illegible]

III Applications

1 : Application aux traitements de texte

[COR] p 916 ↓ Cormen

a) Recherche de motifs

Soit $T \in \Sigma^*$ un texte [illegible] et $P \in \Sigma^*$ un motif. [illegible] On a $|T| = n$, $|P| = m$, [illegible]

[illegible]

Déf : $\forall k \leq m$, on note P_k le préfixe de P de taille k

Déf : [illegible]

Déf : On définit [illegible] $\mathcal{A} = (Q, \Sigma, \delta, I, F)$

avec : $Q = \{0, \dots, m\}$, [illegible], $F = \{m\}$, [illegible]

Prop : [illegible]. [DEV 2]

b) Analyse lexicale

L'analyse lexicale est la première phase d'une compilation. On dispose d'une suite de caractères [illegible]

[illegible]

[illegible]

Ex : . Le code "int a = 12;" est identifié comme une suite de lexèmes : $E_{int}\,E_{\_}\,E_{id}\,E_{\_}\,E_{=}\,E_{\_}\,E_{ic}\,E_{;}$ qui signifie qu'on a repéré un type, un espace, une chaîne de caractère, un espace, un =, un espace, un entier, un ;.

. On souhaite identifier "int" comme un type et non une suite de caractères. E_{int} est prioritaire sur E_{id}

[illegible]

[CAR] p 164

2 : Application à la théorie logique

Déf : [illegible]

[illegible]

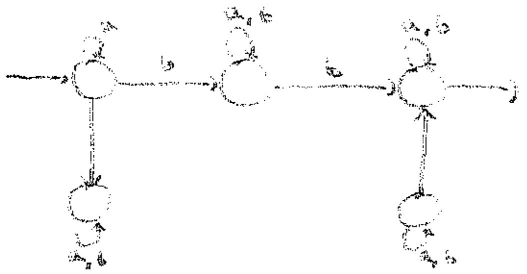
Ex : L'arithmétique de Presburger est une théorie du 1er ordre des entiers muni de l'égalité et de l'addition. ie l'ensemble des formules logiques φ définies par :

- [illegible]
- [illegible]

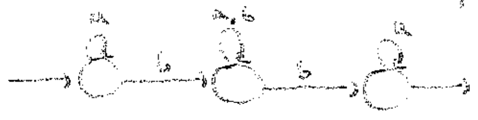
[illegible] : L'arithmétique de Presburger est [illegible] [DEV3]

Annexe

Figure A : $L = \{w \in \Sigma^* \,/\, w$ contient au moins 2 b$\}$



Automate non-émondé mais complet reconnaissant L

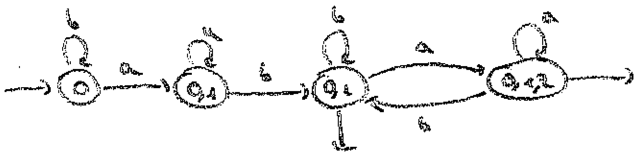


Automate émondé mais non-complet reconnaissant L

Figure B : $L = \Sigma^* ab \, \Sigma^*$

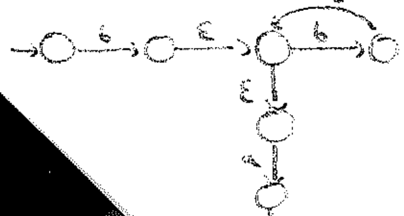
$\mathcal{A}$ 

Automate non déterministe reconnaissant L



Déterminisation à l'aide de l'automate $\mathcal{A}$

Figure C : $L = bb^*a$



Automate de Thompson reconnaissant L

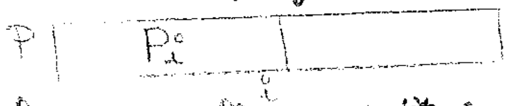
L'automate des occurrences

Cadre et notations

- P motif donné $|P| = m$
- T chaîne textuelle $|T| = n$

↳ On suppose $m \leq n$

- $x \sqsupset y$: "x est suffixe de y"
- P_i = i ème préfixe de P



- fonction suffixe $\sigma : \Sigma^* \to \{0, \dots, m\}$
  $\sigma(x) = \max\{k, P_k \sqsupset x\}$

(la longueur du plus long préfixe de P qui est un suffixe de x)

Automate $\mathcal{A}$

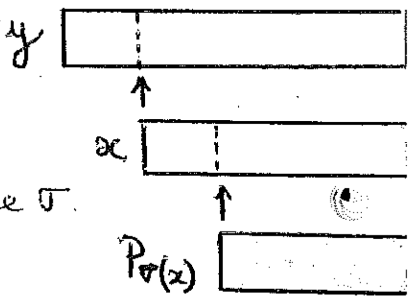
- $Q = \{0, \dots, m\}$
- $I = \{0\}$
- $F = \{m\}$
- $\delta(q, a) = \sigma(P_q a)$
- fonction d'état final ϕ tq
  $\begin{cases} \phi(\varepsilon) = 0 \\ \phi(xa) = \delta(\phi(x), a) \end{cases}$

Théorème : $\left.\begin{array}{l} \forall i \in \{0, \dots, n\}\ \phi(T_i) = \sigma(T_i). \\ \text{En particulier, } \mathcal{L}(\mathcal{A}) = \Sigma^* P. \end{array}\right\}$ Correction
De plus, l'automate $\mathcal{A}$ est minimal.

DEMONSTRATION Correction

Lemme 1 $\forall x, y \in \Sigma^*, x \sqsupset y \Rightarrow \sigma(x) \leq \sigma(y)$

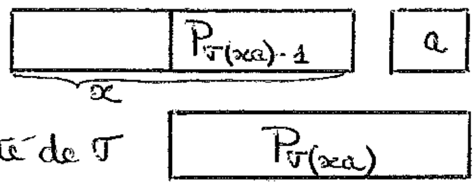
Preuve $\left.\begin{array}{l} P_{\sigma(x)} \sqsupset x \\ x \sqsupset y \end{array}\right\} P_{\sigma(x)} \sqsupset y$



donc $\sigma(x) \leq \sigma(y)$ par maximalité de σ.

Lemme 2 $\forall x \in \Sigma^*, \forall a \in \Sigma, \sigma(xa) \leq \sigma(x) + 1$

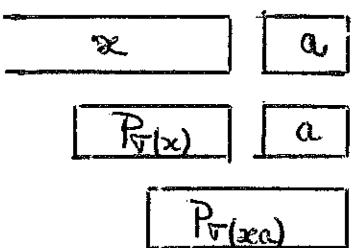
Preuve
- Si $\sigma(xa) = 0$, l'inégalité est satisfaite par positivité de σ
- Sinon, $P_{\sigma(xa)} \sqsupset xa$ par déf de σ



donc $P_{\sigma(xa)-1} \sqsupset x$

d'où $\sigma(xa) - 1 \sqsupset \sigma(x)$ par maximalité de σ

Lemme 3 $\forall x \in \Sigma^*, \forall a \in \Sigma, \sigma(xa) = \sigma(P_{\sigma(x)} a)$



Preuve
- $P_{\sigma(x)} \sqsupset x$ (déf de σ) $\Rightarrow P_{\sigma(x)} a \sqsupset xa \overset{\text{(lemme 1)}}{\Rightarrow} \sigma(P_{\sigma(x)} a) \leq \sigma(xa)$
- $\left.\begin{array}{l} P_{\sigma(x)} a \sqsupset xa \\ P_{\sigma(xa)} \sqsupset xa \\ |P_{\sigma(xa)}| \leq |P_{\sigma(x)} a| \text{ (lemme 2)} \end{array}\right\} P_{\sigma(xa)} \sqsupset P_{\sigma(x)} a$

$\overset{\text{(lemme 1)}}{\Rightarrow} \sigma(xa) \leq \sigma(P_{\sigma(x)} a)$

DÉMONSTRATION Théorème Récurrence sur i

- $i = 0$: $T_0 = \varepsilon$ et $\phi(\varepsilon) = 0 = \sigma(\varepsilon)$

- Supposons l'hypothèse vraie au rang i ;

au rang $i+1$: $\phi(T_{i+1}) = \phi(T_i\, a)$ où $a = T[i+1]$

$$
\begin{aligned}
&= \delta(\phi(T_i), a) && \text{def } \phi \\
&= \delta(\sigma(T_i), a) && \text{par HR} \\
&= \sigma(P_{\sigma(T_i)}\, a) && \text{def } \delta \\
&= \sigma(T_i\, a) && \text{lemme 3}
\end{aligned}
$$

$$\underline{\phi(T_{i+1}) = \sigma(T_{i+1})}$$

DÉMONSTRATION MINIMALITÉ

Soient i, j deux états quelconques, $0 \leq i < j \leq m$

Montrons que $i \nsim j$ pour l'equivalence de Nérode,

cela revient à montrer que $L_i \neq L_j$ où $L_k = \{w \in \Sigma^*, \delta(k, \omega) = m \in F$

Pour ça, posons $\forall k \in \{i, j\}$, $Q_k \in \Sigma^*$ tq $P = P_k\, Q_k$

On a $\phi(P_j Q_j) = \phi(P) = m$ donc $Q_j \in L_j$

(car $\phi(P_j) = \sigma(P_j) = j$ par le théorème)

en revanche $\phi(P_i\, Q_j) < m$ car $|P_i\, Q_j| < m$ donc $Q_j \notin L_i$

CQFD.

L'automate $\mathcal{A}$ est minimal

+ e y ?

Théorème L'arithmétique de Presburger est décidable

DÉMONSTRATION

A montrer : il est décidable de savoir si une formule close est vraie

Soit φ une formule logique close, on note $x_1, \ldots, x_n$ ses variables.

Soit $L = \{\lambda = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{pmatrix} \in \mathbb{N}^n \,/\, \lambda \text{ satisfait } \varphi\}$

Montrons que L est rationnel

- On suppose φ est sous forme prénexe

$$\varphi = Q_1 x_1 \, Q_2 x_2 \ldots Q_n x_n \, \psi$$

- Posons $\forall k \in [\![0, n]\!]$, $\begin{cases} \varphi_k = Q_{k+1} x_{k+1} \ldots Q_n x_n \psi \\ \varphi_0 = \varphi \\ \varphi_n = \psi \end{cases}$

$(x_1, \ldots, x_k)$ sont des variables libres dans φ_k, on notera $\varphi_k(x_1, \ldots, x_k)$

- Définissons un codage des k-uplets d'entiers :

$$\begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_k \end{pmatrix} = \begin{pmatrix} * \\ \\ \end{pmatrix}\begin{pmatrix} * \\ \\ \end{pmatrix}\begin{pmatrix} * \\ \\ \end{pmatrix} \cdots \begin{pmatrix} * \\ \\ \end{pmatrix}$$

écriture en binaire, en ligne de chaque entier, les bits de poids faibles sont à gauche. Eventuellement on ajoute des zéros en tête pour les rendre de la même longueur.

- Posons $X_k = \left\{ \begin{pmatrix} n_1 \\ \vdots \\ n_k \end{pmatrix} / \varphi_k(n_1, \ldots, n_k) \text{ est vraie} \right\}$

On procède par récurrence descendante sur le nombre de variables libres de φ.

On va construire un automate $\mathcal{A}_k$ qui accepte les codages binaires des éléments de X_k

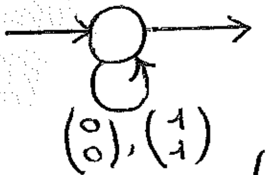
INITIALISATION : Construction de $\mathcal{A}_n$ qui accepte les n-uplets qui satisfont ψ.

- ψ est combinaison booléenne de formules atomiques :

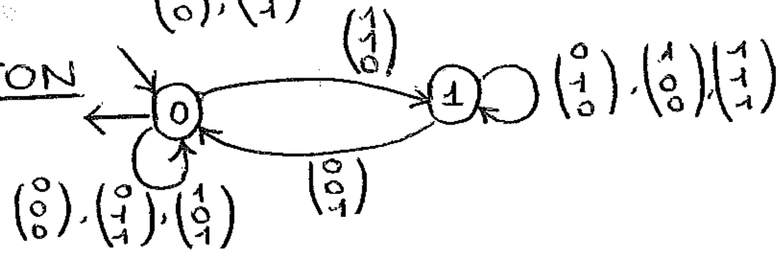
$$x_i = x_j \quad \text{ou} \quad x_i + x_j = x_k$$

Comme la classe des langages rationnels est close pour les opér[ations] booléenne, il suffit de construire un automate pour chacune des formules atomiques.

EGALITÉ



ADDITION



HÉRÉDITÉ Supposons construit l'automate $\mathcal{A}_{k+1}$.

Considérons $\varphi_k = Q_{k+1} x_{k+1} \varphi_{k+1}$ avec $Q_{k+1} \in \{\forall, \exists\}$

- Si $Q_{k+1} = \forall$, alors $\varphi_{k+1} = \neg \exists x_{k+1} \neg \varphi_{k+1}$ et la classe des langages rationnels est close par complémentation.

- On peut donc supposer que $Q_{k+1} = \exists$.

L'automate $\mathcal{A}_k$ est obtenu à partir de $\mathcal{A}_{k+1}$ en "oubliant" la dernière composante :

$$\pi_k : \begin{aligned} \Sigma^{k+1} &\longrightarrow \Sigma^k \\ (b_1, \dots, b_{k+1}) &\longmapsto (b_1, \dots, b_k) \end{aligned}$$

- $\mathcal{A}_k$ a les mêmes états que $\mathcal{A}_{k+1}$
- Si $p \xrightarrow{\lambda} q$ est une transition de $\mathcal{A}_{k+1}$,
  alors $p \xrightarrow{\pi_k(\lambda)} q$ est une transition de $\mathcal{A}_k$
- $\mathcal{A}_k$ a les mêmes états initiaux que $\mathcal{A}_{k+1}$
- Notons F_i les états finaux de $\mathcal{A}_i$, alors $F_i = F_{i+1} \cup \{q / \delta(q, \begin{pmatrix}0\\ \vdots\\ 0\end{pmatrix}^*) \in$

CONCLUSION

L'automate $\mathcal{A}_0$ accepte au moins un mot (ssi) la formule φ est vraie

Le problème du vide est décidable

D'où la conclusion.

Langages minces

Définition : L est mince si $L = u_1 v_1^* w_1 + \dots + u_N v_N^* w_N$

Définition : L est à croissance bornée si $\exists M > 0$ tel que $\forall n \in \mathbb{N}$, $\# \{ w \in L \mid |w| = n \} \leq M$

Théorème : L est mince ssi $L \in Rec(\Sigma^*)$ et L est à croissance bornée

preuve :

$\Rightarrow$ | Supposons L mince : $L = u_1 v_1^* w_1 + \dots + u_N v_N^* w_N$

L est dénoté par une expression rationnelle donc $L \in Rec(\Sigma^*)$ d'après le théorème de Kleene.

Soit $n \in \mathbb{N}$, il y a au plus un mot de $u_i v_i^* w_i$ de taille n.

Donc $\# \{ w \in L \mid |w| = n \} \leq N$

Donc L est à croissance bornée.

$\Leftarrow$ | Supposons $L \in Rec(\Sigma^*)$ et à croissance bornée.

Soit $\mathcal{A}$ automate déterministe et émondé reconnaissant L.

Lemme 1 : Soit C composante fortement connexe de $\mathcal{A}$. Si C n'est pas réduite à un cycle, alors L n'est pas à croissance bornée.

preuve : Si C n'est pas réduite à un cycle, alors $\exists p, q, r \in C$ et $\exists a, b \in \Sigma$ tels que $p \xrightarrow{a} q$ et $p \xrightarrow{b} r$ avec $q \neq r$

$\mathcal{A}$ déterministe et $q \neq r \Rightarrow a \neq b$

C fortement connexe $\Rightarrow \exists t_1, t_2 \in \Sigma^*$ tels que $q \xrightarrow{t_1} p$ et $r \xrightarrow{t_2} p$ et $t_1 \neq \varepsilon$, $t_2 \neq \varepsilon$

posons $y_1 = a t_1$ et $y_2 = b t_2$

puis $v_1 = y_1^{h_1}$, $v_2 = y_2^{h_2}$

avec $h_1 = \dfrac{\mathrm{ppcm}(|y_1|,|y_2|)}{|y_1|}$

$h_2 = \dfrac{\mathrm{ppcm}(|y_1|,|y_2|)}{|y_2|}$

On a $|v_1| = |v_2| = \mathrm{ppcm}(|y_1|,|y_2|) = m$

Comme $a \neq b$, tous les mots de $\{v_1, v_2\}^n$ sont distincts

et donc $\forall n \in \mathbb{N}$, $\# \{v_1, v_2\}^n = 2^n$

$\mathcal{A}$ émondé $\Rightarrow$ $\exists\, u, v \in \Sigma^*$ et $f \in F$ tels que

$i \xrightarrow{u} p$ et $p \xrightarrow{v} f$

$\Rightarrow \forall n \in \mathbb{N}$, $u\{v_1, v_2\}^n v \subseteq L$

et $\forall n \in \mathbb{N}$, $\# \{ w \in L \,/\, |w| = |u| + |v| + nm \} \geq 2^n$

$\Rightarrow$ L n'est pas à croissance bornée

~~Lemme 2 : Si C_1 et C_2 sont deux composantes fortement connexes disjointes de $\mathcal{A}$ qui peuvent se joindre par un chemin, alors L n'est pas à croissance bornée~~

preuve : Supposons que $\exists\, p_1 \in C_1$, $p_2 \in C_2$ et $x \in \Sigma^*$ tels que $p_1 \xrightarrow{x} p_2$

$C_1 \neq C_2 \Rightarrow x \neq \varepsilon$

$\Rightarrow x = a x'$ avec $a \in \Sigma$

On a donc $p_1 \xrightarrow{a} q \xrightarrow{x'} p_2$

Quitte à changer p_1, on peut supposer que $q \notin C_1$

C_i fortement connexe $\Rightarrow \exists\, y_1, y_2 \in \Sigma^*$ tels que

$y_1 \neq \varepsilon$, $y_2 \neq \varepsilon$ et

$p_1 \xrightarrow{y_1} p_1$, $p_2 \xrightarrow{y_2} p_2$

Posons $v_1 = y_1^{h_1}$ et $v_2 = y_2^{h_2}$

avec $h_i = \dfrac{\mathrm{ppcm}(|y_1|,|y_2|)}{|y_i|}$

On a $|v_1| = |v_2| = \mathrm{ppcm}(|y_1|,|y_2|) = m$

$\mathcal{A}$ déterministe et $q \notin C_1 \Rightarrow a \neq$ première lettre de y_1

$\Rightarrow \forall n$, les $v_1^k x v_2^{n-k}$, $k \in \{0, \dots, n\}$

sont tous distincts et de même taille.

et émondé $\Rightarrow$ $\exists\, u, v \in \Sigma^*$, $f \in F$ tels que

$$i \xrightarrow{u} p_1 \quad , \quad p_2 \xrightarrow{w} f$$

$$\Rightarrow \quad u v_1^k x v_2^{n-k} w \in L \quad \forall k \in \{0, \dots, n\}, \forall n$$

donc $\forall n \in \mathbb{N}$, $\# \{ w \in L \,/\, |w| = |u| + |v| + |x| + m\, n \}$

$$\geqslant n+1$$

$\Rightarrow$ L n'est pas à croissance bornée.

<u>Retour à la preuve du Thm :</u>

L est à croissance bornée donc :

- par le lemme 1, on déduit que toutes les composantes fortement connexes sont des cycles
- par le lemme 2, deux cycles ne peuvent être reliés par un chemin.

Donc un calcul réussi de L soit ne passe par aucun cycle, soit passe par un unique cycle.

Ceux passant par aucun cycles sont en nombre fini car tous les états qui le compose sont différents. On note $b_1, \dots, b_m$ les étiquettes de ces calculs.

Ceux passant par un unique cycle sont de la forme :

$$i \xrightarrow{u} \underset{\substack{\downarrow \\ \text{1er état} \\ \text{du cycle}}}{p} \xrightarrow{v} q \xrightarrow{v'} \underset{\substack{\downarrow \\ \text{dernier} \\ \text{état du} \\ \text{cycle}}}{q} \xrightarrow{w} f$$

ils fournissent des mots de $u v (v')^* w \subseteq L$

Le nombre de possibilité pour p et q étant finies donc il existe un nombre finis d'expression $u_1 v_1^* w_1, \dots, u_N v_N^* w_N$ possibles.

On a répertorié tous les calculs possibles donc

$$L = b_1 + \dots + b_m + u_1 v_1^* w_1 + \dots + u_N v_N^* w_N$$

$\Rightarrow$ L est mince.