

Théorie des Langages

Lionel Rieg

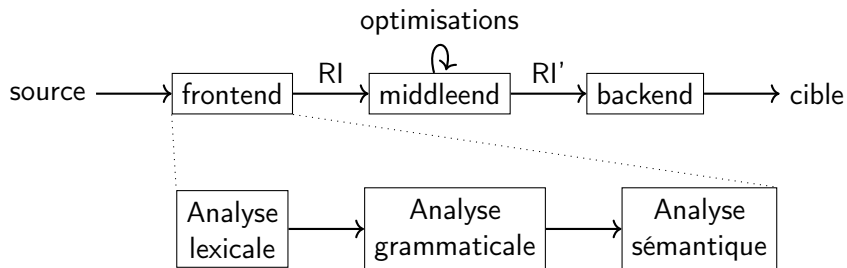
Grenoble INP - Ensimag, 1^{re} année

Motivations et objectifs

- **Théorie des langages** : étude des langages formels
- **Langage formel** : ensemble de *mots, phrases, textes, énoncés* défini « formellement », sans considération sémantique a priori
- Objectif : trouver des moyens de **définir** des langages formels, et des moyens de les **reconnaître** (savoir si un mot appartient au langage)

Pourquoi vous le faire étudier ?

- Théorie formelle en informatique
 - ~ parfois plus rigoureux que même les maths
- Exemple de modélisation/formalisation mathématique
 - ~ comment abstraire des choses pour pouvoir les traiter ?
 - ~ quels outils utiliser ? quels raisonnements ? etc.
- Utilisé en **compilation**



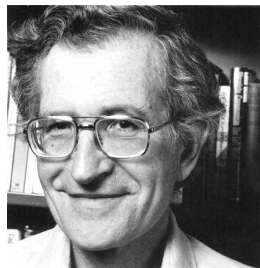
À quoi ça sert ?

- Définition de langages de programmation
 \leadsto Analyse syntaxique d'un programme
- Description de la structure de documents XML
- Calculabilité, complexité (cf. cours RO)
- Recherche de texte dans un document
- Génération automatique d'images
- Bioinformatique
- Traitement automatique des langues naturelles
- Cryptographie
- Contrôle de systèmes
- ...

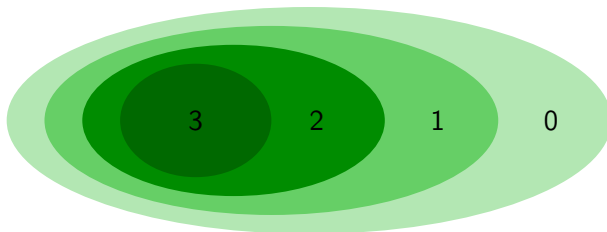
La référence

Noam Chomsky (1928-), États-Unis

- Linguiste, philosophe, logicien, activiste
- 1956 : définition des **grammaires formelles**
 - ▶ Ensemble de règles permettant d'engendrer des langages formels
 - ▶ Classification des grammaires (et des langages engendrés) en fonction de la forme de leurs règles
 - ▶ **Hiérarchie de Chomsky**



Hiérarchie de Chomsky



Type	Langage	Engendré par	Reconnu par
0	Calculatoirement énumérable	Grammaire générale	Machine de Turing
1	Sous-contexte	Grammaire sous-contexte	Machine de Turing linéairement bornée
2	Hors-contexte	Grammaire hors-contexte	Automate à pile
3	Régulier	Grammaire linéaire à droite	Automate fini

Fonctionnement du cours de TL

Organisation

- 12 CM en amphis, 12 séances de TD
- Matériel disponible en ligne sur Chamilo
 - ▶ Diapos (en ligne chaque semaine)
 - ▶ Recueil d'exercices de TD (également en papier)
- 3 grandes parties dans le cours
 - ▶ Langages et automates, ≈ 4 CM (programme de MP option info, MPI)
 - ▶ Grammaires, ≈ 2 CM (programme de MPI)
 - ▶ Construction d'analyseurs, ≈ 3 CM (hors-programme de prépa)

Évaluation

- Examen terminal (80% de la note)
 - ▶ Durée : 2h
 - ▶ Documents autorisés : une feuille manuscrite recto-verso
- projet en binôme : (20% de la note)
à rendre pour le **dimanche 7 décembre 2025 à 23h59**

Théorie des Langages

Cours 1 : Rappels mathématiques, théorème du point fixe, langages

Lionel Rieg

Grenoble INP - Ensimag, 1^{re} année

Éléments de logique

Énoncés mathématiques écrit dans un langage dédié : les **formules logiques**

Notations des opérateurs

\wedge	« et »	conjonction
\vee	« ou »	disjonction (inclusive)
\neg	« non »	négation
\implies	« implique »	implication
\exists	« il existe »	quantification existentielle
\forall	« pour tout »	quantification universelle



L'ordre des quantificateurs est important !

$\forall n \in \mathbb{N}. \exists m \in \mathbb{Z}. n + m = 0$ vs $\exists m \in \mathbb{N}. \forall n \in \mathbb{N}. n + m = 0$

Continuité vs continuité uniforme : $\forall \varepsilon. \forall x. \exists \delta. \dots$ vs $\forall \varepsilon. \exists \delta. \forall x. \dots$

Priorités ? $\neg > \wedge, \vee > \implies$; \forall, \exists en général en dernier

Implication

S'il fait beau, nous irons à la plage.

il fait beau \implies nous irons à la plage

Que se passe-t-il s'il ne fait pas beau ? Peut-on aller à la plage ?

Tables de vérité

\implies		Aller à la plage		\neq	\iff		
		✓	✗			✓	✗
Il fait beau	✓	✓	✗		✓	✓	✗
	✗	✓	✓		✗	✗	✓

En logique classique, $A \implies B = \neg A \vee B$

Négation

- Lois de De Morgan : $\neg(A \wedge B) = \neg A \vee \neg B$
 $\neg(A \vee B) = \neg A \wedge \neg B$
- Avec l'implication : $\neg(A \implies B) = \neg(\neg A \vee B) = A \wedge \neg B$
- Avec les quantificateurs : $\neg(\exists x. A) = \forall x. \neg A$
 $\neg(\forall x. A) = \exists x. \neg A$

Exemple :

$$\begin{aligned} & f \text{ n'est pas uniformément continue} \\ = & \neg \left(\forall \varepsilon > 0. \exists \delta > 0. \forall x. \forall y. d(x, y) \leq \delta \implies d(f(x), f(y)) \leq \varepsilon \right) \\ = & \exists \varepsilon > 0. \neg \left(\exists \delta > 0. \forall x. \forall y. d(x, y) \leq \delta \implies d(f(x), f(y)) \leq \varepsilon \right) \\ = & \exists \varepsilon > 0. \forall \delta > 0. \neg \left(\forall x. \forall y. d(x, y) \leq \delta \implies d(f(x), f(y)) \leq \varepsilon \right) \\ = & \exists \varepsilon > 0. \forall \delta > 0. \exists x. \exists y. \neg \left(d(x, y) \leq \delta \implies d(f(x), f(y)) \leq \varepsilon \right) \\ = & \exists \varepsilon > 0. \forall \delta > 0. \exists x. \exists y. d(x, y) \leq \delta \wedge \neg \left(d(f(x), f(y)) \leq \varepsilon \right) \\ = & \exists \varepsilon > 0. \forall \delta > 0. \exists x. \exists y. d(x, y) \leq \delta \wedge d(f(x), f(y)) > \varepsilon \\ = & \text{on peut trouver des points } x, y \text{ arbitrairement proches } \leq \delta \\ & \text{mais dont les images par } f \text{ sont éloignées } > \varepsilon \end{aligned}$$

Schéma de compréhension

Ensemble des éléments x qui vérifient une propriété P :

$$\{x \mid P(x)\}$$

Exemples : $\{n \mid n \text{ pair}\}$
 $\{(n, m, p) \mid n^2 + m^2 = p^2\}$ (triplets pythagoriciens)

On peut mettre plusieurs arguments à gauche et à droite :

$$\{2^n 5^m, 3^n 5^m \mid n \in \mathbb{N}, m \in 2\mathbb{N}\}$$

Point culturel : le paradoxe de Russell

Si $P(x) \stackrel{\text{def}}{=} x \notin x$ et $A \stackrel{\text{def}}{=} \{y \mid P(y)\}$, a-t-on $A \in A$?

$$A \in A \iff P(A) \iff A \notin A$$

Solutions possibles :

- interdire d'écrire $\{y \mid P(y)\}$ sans limite sur y
- introduire des **types** pour distinguer les sortes d'objets

Notations ensemblistes

- produit cartésien $A \times B$: des couples d'un élément de A et de B

$$A \times B \stackrel{\text{def}}{=} \{(x, y) \mid x \in A \wedge y \in B\}$$

- union **disjointe** $A \uplus B$ (ou $A \sqcup B$) :
☞ union mais sans confusion entre les éléments communs

Exemple : $\mathbb{R} \cup \mathbb{N} = \mathbb{R}$ mais $\mathbb{R} \uplus \mathbb{N} \neq \mathbb{R}$

Comment faire ?

- ▶ renommer les éléments communs
- ▶ ajouter une étiquette à chaque élément en fonction de son origine

$$A \uplus B \stackrel{\text{def}}{=} (\{0\} \times A) \cup (\{1\} \times B)$$

- Généralisable au cas n -aire :

$$A \times B \times C \stackrel{\text{def}}{=} \{(x, y, z) \mid x \in A \wedge y \in B \wedge z \in C\}$$

$$A \uplus B \uplus C \stackrel{\text{def}}{=} (\{0\} \times A) \cup (\{1\} \times B) \cup (\{2\} \times C)$$

Théorèmes du point fixe

Définition (Point fixe)

Un **point fixe** d'une fonction $f : E \rightarrow E$ est un point x tel que $f(x) = x$.

Il y a toute une famille de théorèmes qui en donnent :

- **Théorème de Banach** : E espace métrique complet et f contractante
 \leadsto point fixe unique, atteint par itération de f
Exemple : $E = [-0.4, 0.4]$, $f : x \mapsto x^2$, point fixe : 0
- **Théorème de Kleene** : E ordre partiel complet et f Scott-continue
 \leadsto plus petit point fixe atteint par itération de f depuis le minimum \perp
Exemple : $E = (\mathcal{P}(V), \subseteq)$, $f(X) = (X \cap \alpha) \cup \beta$, point fixe : β
- **Théorème de Knaster-Tarski** : E trellis complet et f croissante
 \leadsto plus petit et plus grand points fixes, preuve non constructive
Exemple : $E = \{\text{types de données}\}$, $f(X) = A \times X \cup \{\text{nil}\}$
 - ▶ plus petit point fixe : listes finies d'éléments de A
 - ▶ plus grand point fixe : E
 - ▶ autre point fixe : listes finies et infinies

Théorème de Kleene pour $\mathcal{P}(E)$, \subseteq

Soit E un ensemble et f une application de $\mathcal{P}(E) \rightarrow \mathcal{P}(E)$.

Définition : fonctions continues au sens de (Dana) Scott

La fonction f est **continue** si pour toute suite croissante $(A_i)_{i \in \mathbb{N}}$ de $\mathcal{P}(E)$ (c.-à-d. $\forall i, A_i \subseteq A_{i+1}$), on a $f(\bigcup_{i \in \mathbb{N}} A_i) = \bigcup_{i \in \mathbb{N}} f(A_i)$.

RMQ : f **continue** implique f **croissante** : $X \subseteq Y \Rightarrow f(X) \subseteq f(Y)$
Toutes les fonctions croissantes « usuelles » sont continues.
La composée de deux fonctions continues est continue.

Théorème (Kleene, 1938)

Si f est continue, le plus petit point fixe (pppf) $\mu(f)$ de f est $\bigcup_{i \in \mathbb{N}} f^i(\emptyset)$.

RMQ :

- $f^0 = \text{Id}_{\mathcal{P}(E)}$ donc $f^0(\emptyset) = \emptyset$
- f croissante donne $f^i(\emptyset) \subseteq f^{i+1}(\emptyset)$ et $\bigcup_{0 \leq i \leq k} f^i(\emptyset) = f^k(\emptyset)$
- S'il existe k tq $f^k(\emptyset) = f^{k+1}(\emptyset)$, technique
alors $\forall i \geq k, f^i(\emptyset) = f^k(\emptyset)$ et $\bigcup_{i \in \mathbb{N}} f^i(\emptyset) = f^k(\emptyset)$ de calcul
En particulier, si E est de cardinal fini n , alors $k \leq n$.

Démonstration du théorème du point fixe de Kleene

Trois affirmations à démontrer :

1. f et $(f^i(\emptyset))_{i \in \mathbb{N}}$ sont croissantes
2. $\bigcup_{i \in \mathbb{N}} f^i(\emptyset)$ est un point fixe de f
3. tout point fixe de f contient $\bigcup_{i \in \mathbb{N}} f^i(\emptyset)$

- Montrons que f est croissante.

Soit $X \subseteq Y$. Montrons que $f(X) \subseteq f(Y)$. On définit la suite croissante $(A_i)_{i \in \mathbb{N}}$ par $A_0 = X$ et $A_i = Y$ pour $i > 0$.

Alors $f(Y) = f(\bigcup_{i \in \mathbb{N}} A_i) = \bigcup_{i \in \mathbb{N}} f(A_i) = f(X) \cup f(Y)$.

- Montrons par récurrence sur i que $f^i(\emptyset) \subseteq f^{i+1}(\emptyset)$.

$$i = 0 \quad f^0(\emptyset) = \emptyset \subseteq f^1(\emptyset)$$

$$i > 0 \quad \text{Par HR, } f^i(\emptyset) \subseteq f^{i+1}(\emptyset). \text{ Comme } f \text{ est croissante,}$$
$$f^{i+1}(\emptyset) = f(f^i(\emptyset)) \subseteq f(f^{i+1}(\emptyset)) = f^{i+2}(\emptyset)$$

Démonstration du théorème du point fixe de Kleene

Trois affirmations à démontrer :

1. f et $(f^i(\emptyset))_{i \in \mathbb{N}}$ sont croissantes
2. $\bigcup_{i \in \mathbb{N}} f^i(\emptyset)$ est un point fixe de f
3. tout point fixe de f contient $\bigcup_{i \in \mathbb{N}} f^i(\emptyset)$

Comme f est continue et $\bigcup_{i \in \mathbb{N}} f^i(\emptyset)$ est croissante,

$$f\left(\bigcup_{i \in \mathbb{N}} f^i(\emptyset)\right) = \bigcup_{i \in \mathbb{N}} f(f^i(\emptyset)) = \bigcup_{i > 0} f^i(\emptyset) = \bigcup_{i \in \mathbb{N}} f^i(\emptyset) \quad \text{car } f^0(\emptyset) = \emptyset.$$

Soit X un point fixe de f . Montrons qu'il contient $\bigcup_{i \in \mathbb{N}} f^i(\emptyset)$.
Pour cela, on montre par récurrence sur i qu'il contient $f^i(\emptyset)$.

$$i = 0 \quad f^0(\emptyset) = \emptyset \subseteq X$$

$$i > 0 \quad \text{Par HR, } f^i(\emptyset) \subseteq X.$$

$$\text{Comme } f \text{ est croissante, } f^{i+1}(\emptyset) = f(f^i(\emptyset)) \subseteq f(X) = X.$$

Exemples de points fixes

Exemple (Accessibilité dans une carte)

Soit V un ensemble de villes et $R : V \mapsto \mathcal{P}(V)$ les routes qui les relient. L'ensemble des villes accessibles depuis $v \in V$ est le plus petit point fixe de $f(X) = R(X) \cup \{v\}$

Exemples :

- plus petite solution de l'équation $f(X) = (X \cap A) \cup B = X$:
 $f^0(\emptyset) = \emptyset, \quad f(\emptyset) = B, \quad f(B) = (B \cap A) \cup B = B \quad \text{car } B \cap A \subset B$
Stabilisation donc $\mu(f) = B$
- plus petite relation d'équivalence qui contient R :
 $\mu(f)$ pour $f(r) \stackrel{\text{def}}{=} r \cup R \cup \text{Id} \cup r^{-1} \cup r \circ r$

Autres exemples plus tard dans le cours :

- solution d'un **système d'équations**
- langage d'une grammaire HC
- transformations d'automates

Généralisation aux systèmes d'équations ensemblistes

Pour f de $\mathcal{P}(E_1) \times \dots \times \mathcal{P}(E_n) \rightarrow \mathcal{P}(E_1) \times \dots \times \mathcal{P}(E_n)$

Extension de \subseteq et \cup : composante par composante

Pour (X_1, \dots, X_n) et (Y_1, \dots, Y_n) de $\mathcal{P}(E_1) \times \dots \times \mathcal{P}(E_n)$, on a

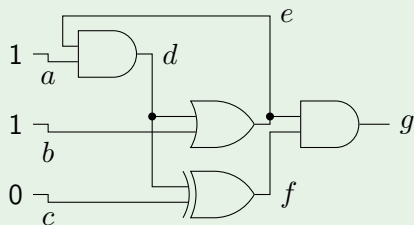
$$(X_1, \dots, X_n) \subseteq (Y_1, \dots, Y_n) \stackrel{\text{def}}{=} \forall i, X_i \subseteq Y_i$$

$$(X_1, \dots, X_n) \cup (Y_1, \dots, Y_n) \stackrel{\text{def}}{=} (X_1 \cup Y_1, \dots, X_n \cup Y_n)$$

$$\perp \stackrel{\text{def}}{=} (\emptyset, \dots, \emptyset)$$

Exercice : Démontrer le théorème du point fixe dans ce cas.

Exemple (États stables d'un circuit logique)



$$\begin{cases} a = \{1\}; & b = \{1\}; & c = \{0\} \\ d = a \wedge e \\ e = b \vee d \\ f = c \oplus d \\ g = e \wedge f \end{cases} \quad \text{sur } \mathcal{P}(\{0,1\})$$

Calcul d'une image par commutation de point fixe

Lemme (Lemme de commutation)

Pour $k \in \{1, 2\}$, soit f_k applications **continues** de $\mathcal{P}(E_k) \rightarrow \mathcal{P}(E_k)$

Soit g application **continue** de $\mathcal{P}(E_1) \rightarrow \mathcal{P}(E_2)$ avec

$$g(\emptyset) = \emptyset \quad \text{et} \quad g \circ f_1 = f_2 \circ g$$

Alors, on a :

$$\mu(f_2) = \bigcup_{i \in \mathbb{N}} f_2^i(\emptyset) = g\left(\bigcup_{i \in \mathbb{N}} f_1^i(\emptyset)\right) = g(\mu(f_1))$$

Démonstration :
$$g\left(\bigcup_{i \in \mathbb{N}} f_1^i(\emptyset)\right) = \bigcup_{i \in \mathbb{N}} g(f_1^i(\emptyset)) = \bigcup_{i \in \mathbb{N}} f_2^i(g(\emptyset)) = \bigcup_{i \in \mathbb{N}} f_2^i(\emptyset)$$

Intérêt :

- $g(\mu(f_1))$ est lui-même un point fixe : $\mu(f_2)$
- $\mu(f_2)$ peut être plus simple à calculer que $\mu(f_1)$

ou réciproquement

Exemple : E_1 infini et E_2 fini

- Donne des informations sur $\mu(f_1)$ sans le calculer

Langages

Définitions

Définitions (Vocabulaire, mot)

- Un **vocabulaire** (ou alphabet) est un ensemble **fini** quelconque. Ses éléments sont appelés des **symboles** (ou lettres).
- Un **mot** sur un vocabulaire V est une **séquence finie** de symboles de V .

Exemples

V	mot sur V	notations abrégées
$\{a, b, \dots, z\}$	$[b, r, o, c, c, c, o, l, i]$	<i>brocccoli</i> , <i>broc³oli</i>
$\{0, \dots, 9\}$	$[2, 0, 2, 0]$	2020, $(20)^2$
$\{a, b, ab\}$	$[ab], [a, b]$	<i>ab</i> ? (<i>ab</i>) vs (<i>a</i>)(<i>b</i>)

Définition

- On note ε le mot correspondant à la séquence vide (« *mot vide* »).

Définitions (suite)

Définition (longueur d'un mot)

Soit V un vocabulaire et soit $u = u_1 \cdots u_n$ un mot sur V .

La **longueur** de u est alors n , et on note $|u| = n$.

Remarque

En particulier, on a $|\varepsilon| = 0$.

Définitions

- Pour $n \in \mathbb{N}$, V^n est l'ensemble des mots sur V de longueur n . Par abus de notation, on identifie V et V^1 .
- V^+ est l'ensemble des mots sur V de longueur au moins 1.
- **V^* est l'ensemble des mots sur V .**
- Pour $w \in V^*$ et $a \in V$, $|w|_a$ est le nombre d'occurrences de a dans w .

Exemples

Exemples

- Soient $V = \{a, b\}$ et $w = ababbb$. Alors $|w| = 6$ et $|w|_b = 4$.
- Soient $V = \{cd, dc\}$ et $w = cdcddc$. Alors $|w| = 6$ et $|w|_{dc} = 3$.

Proposition

On a les égalités suivantes :

$$\begin{aligned} V^* &= \bigcup_{n \geq 0} V^n \\ V^+ &= \bigcup_{n > 0} V^n \end{aligned}$$

Concaténation

Définition

Soit V un vocabulaire, $u = u_1 \cdots u_n$ et $v = v_1 \cdots v_m$ deux mots de V^* . La **concaténation de u et v** , notée $u.v$, est le mot de V^* défini par

$$u.v = u_1 \cdots u_n v_1 \cdots v_m$$

Exemple

Soient $u = bac$ et $v = aacb$.

Alors $u.v = bacaacb$.

Théorème

$(V^*, \cdot, \varepsilon)$ est un monoïde (« groupe sans inverse »)

- *associative* : $u.(v.w) = (u.v).w$
- ε *élément neutre* : $\varepsilon.u = u.\varepsilon = u$

Notation

On pourra noter uv au lieu de $u.v$.

Concaténation (suite)

Proposition

Si $|u| = i$ et $|v| = j$, alors $|uv| = i + j$.

Définitions

Soient $v, z \in V^*$. On dit que v est un :

- **facteur** de z ssi $\exists u, w \in V^*$ tels que $z = u.v.w$
- **préfixe** de z ssi $\exists w \in V^*$ tel que $z = v.w$
- **suffixe** de z ssi $\exists u \in V^*$ tel que $z = u.v$

Langages

Définition

On appelle **langage** sur V tout **sous-ensemble** de V^* .

Exemples

- $\emptyset \subseteq \{a, b\}^*$
- $\{a, b\}^* \subseteq \{a, b\}^*$
- $\{abab, ab, abba\} \subseteq \{a, b\}^*$
- $\{a^n b^n \mid n \geq 0\} \subseteq \{a, b\}^* \quad (\{\varepsilon, ab, aabb, aaabbb, \dots\})$
- « Ensemble des programmes Python » \subseteq Unicode*

Remarque

On s'intéressera en TL à **définir** et **reconnaître** des sous-ensembles de V^* .

Opérations sur les langages

Définition

Soient L et M deux langages sur V (c.-à-d. L et $M \subseteq V^*$).

Par analogie avec les opérations sur les mots, on définit :

$$L \cup M \stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\}$$

$$L.M \stackrel{\text{def}}{=} \{w_1w_2 \mid w_1 \in L \text{ et } w_2 \in M\}$$

$$\forall i > 0, L^i \stackrel{\text{def}}{=} L.L^{i-1} \quad \text{concaténation de } i \text{ mots de } L$$

$$L^0 \stackrel{\text{def}}{=} \{\varepsilon\}$$

$$L^* \stackrel{\text{def}}{=} \bigcup_{i \geq 0} L^i \quad \text{et} \quad L^+ \stackrel{\text{def}}{=} \bigcup_{i > 0} L^i$$

Notation : on pourra noter LM au lieu de $L.M$.

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \{c^n d^{2n} \mid n \geq 0\}$. Alors

$$\begin{aligned} LM &= \{a^n b^n c^p d^{2p} \mid n, p \geq 0\} \\ L^* &= \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \dots a^{n_k} b^{n_k} \mid k \geq 0 \text{ et } n_1, \dots, n_k \geq 0\} \end{aligned}$$

Question

Si L est un langage, peut-on avoir $\varepsilon \in L^+$? Oui, ssi $\varepsilon \in L$

Exemple: $L = \{\varepsilon, a\}$, $L^2 = \{\varepsilon, a, aa\}$, $L^+ = \{a^n \mid n \geq 0\} = L^*$

Une notation utile : les expressions régulières

Afin de faciliter l'écriture de langages, on s'autorise à retirer les accolades des singletons :

- ϵ représente le singleton $\{\epsilon\}$
- Pour tout $x \in V$, x représente le singleton $\{x\}$
- $E_1 + E_2$ représente $E_1 \cup E_2$

Exemple : $a.(a + b)^*$ est une expression régulière (ER) sur $V = \{a, b\}$.

$$a.(a + b)^* = \{a\} . (\{a\} \cup \{b\})^* = \{a\} . \{a, b\}^* = \{a\} . V^*$$

Exercice : Quels langages sont représentés par les ER suivantes ?

- $a^*.b^* = \{a\}^* . \{b\}^* = \{a^n b^m \mid n, m \in \mathbb{N}\}$ (sur $V = \{a, b\}$)
- $(a.b)^* = \{ab\}^* = \{(ab)^n \mid n \in \mathbb{N}\}$
- $((a + b)^2)^* = \{aa, ab, ba, bb\}^* = \{\text{mots sur } \{a, b\} \text{ de longueur paire}\}$
- $(a + b + \epsilon)^n = \{\text{mots sur } \{a, b\} \text{ de longueur au plus } n\}$

En fait, les expressions régulières sont plus qu'une simple notation.

Abréviations

- On pourra noter $E_1 E_2$ à la place de $E_1.E_2$
- Parenthésage :
 - ▶ « . » et « + » associatifs
 - ▶ Priorités : comme en arithmétique
 - « * » plus prioritaire que « . » plus prioritaire que « + »

Exemple

Soit $E = ((a.(b + (c + (d.(c^*))))).(a.b)^*).$

On pourra simplement noter $E = a(b + c + dc^*)(ab)^*.$

Question : à quoi servent les ER ?

- Intérêt pratique : **recherche de motif** (cf. grep, awk, sed, etc.)
- Intérêt théorique : description inductive des langages réguliers
(on y reviendra plus tard)
aussi utilisé en pratique ! (description des entiers décimaux en Python)

Les ER dans les langages de programmation

Utilisées dans beaucoup d'outils (grep, awk, sed, etc.)
et pour vérifier des entrées (date, adresse email, éviter les injections)

V contient les caractères alphanumériques + ...

\wedge $\$$ début/fin de ligne

$\backslash <$ $\backslash >$ début/fin de mot

$[liste]$ $[\wedge liste]$ un caractère dans ou hors de la *liste*

$.$ un caractère quelconque

$\backslash c$ protection du caractère spécial c (ex : $\$$, $.$)

Répétitions $*$

$+$

$?$ 0 ou 1 fois

$\{n\}$ n fois

$\{n,p\}$ entre n et p fois

$|$ choix

$()$ regroupement

Exercices

Exercice 1 : simplification

Soit E une expression régulière. Simplifier les expressions suivantes :

- | | | |
|-----------------------------|-----------------------------|------------------------------|
| 1. $E.E^* + \epsilon = E^*$ | 3. $\epsilon^* = \epsilon$ | 5. $\emptyset.E = \emptyset$ |
| 2. $\epsilon.E = E$ | 4. $\emptyset^* = \epsilon$ | 6. $\emptyset + E = E$ |

Exercice 2 : adresse mail

Donner une expression régulière pour les adresses email.

Version simplifiée (<https://www.regular-expressions.info/email.html>) :

- $V_1 = [a-z0-9!\#\$\%&'*+/=/?\^_'\{\}~\text{-}]$ et $V_2 = [a-z0-9]$
- première partie : V_1 plus des . **non consécutifs** et **non aux bords**
- le caractère « @ »
- deuxième partie : V_2 plus des - **non consécutifs** et **non aux bords**, le tout possiblement répété avec des . entre les répétitions
- la deuxième partie doit avoir une taille de 63 caractères au maximum

Point fixe et langages

Le théorème du point fixe permet de contruire des langages.

Exemple

Soit $V = \{a, b\}$. On pose $f : X \mapsto \{a\} . X . \{b\} \cup \{\varepsilon\}$ et $L = \mu(f)$.

Alors

$$f^0(\emptyset) = \emptyset$$

$$f^1(\emptyset) = \emptyset \cup \{\varepsilon\} = \{\varepsilon\}$$

$$f^2(\emptyset) = \{ab\} \cup \{\varepsilon\} = \{\varepsilon, ab\}$$

$$f^3(\emptyset) = \{ab, aabb\} \cup \{\varepsilon\} = \{\varepsilon, ab, aabb\}$$

$$f^i(\emptyset) = \{a^k . b^k \mid k < i\}$$

$$L = \bigcup_{i \in \mathbb{N}} f^i(\emptyset) = \{a^k . b^k \mid k \in \mathbb{N}\}$$

On peut le généraliser aux systèmes d'équations.

Exercice : De quelle équation $L^* = \bigcup_{i \in \mathbb{N}} L^i$ est-il la solution ?

Lemme de commutation et langages

Le lemme de commutation permet de calculer l'image par une fonction continue d'un langage défini comme un pppf d'un système d'équation.

Exemple

Décider $\varepsilon \in L$ avec L défini comme un pppf.

On pose $\varepsilon(L) : \mathcal{P}(V^*) \rightarrow \mathcal{P}(\{\varepsilon\}) = \{\emptyset, \{\varepsilon\}\}$
 $\stackrel{\text{def}}{=} L \cap \{\varepsilon\}$

Par commutation de pppf avec $g(X_1, \dots, X_k) = (\varepsilon(X_1), \dots, \varepsilon(X_k))$ et $f_1 : \mathcal{P}(V^*) \times \dots \times \mathcal{P}(V^*) \rightarrow \mathcal{P}(V^*) \times \dots \times \mathcal{P}(V^*)$ l'équation du système.

On cherche $f_2 : \mathcal{P}(\{\varepsilon\}) \times \dots \times \mathcal{P}(\{\varepsilon\}) \rightarrow \mathcal{P}(\{\varepsilon\}) \times \dots \times \mathcal{P}(\{\varepsilon\})$ telle que $f_2 \circ g = g \circ f_1$.

Puis on calcule $\bigcup_{i \in \mathbb{N}} f_2^i(\emptyset)$ qui converge en au plus k itérations !

Lemme de commutation et langages

Le lemme de commutation permet de calculer l'image par une fonction continue d'un langage défini comme un pppf d'un système d'équation.

Exemple

Décider $\varepsilon \in L$ avec L défini comme un pppf.

On pose $\varepsilon(L) : \mathcal{P}(V^*) \rightarrow \mathcal{P}(\{\varepsilon\}) = \{\emptyset, \{\varepsilon\}\}$
 $\stackrel{\text{def}}{=} L \cap \{\varepsilon\}$

$$\begin{cases} L = \{b\} \cup L'.L' \\ L' = \{a\}.L \end{cases} \quad f_1 : \begin{pmatrix} L \\ L' \end{pmatrix} \mapsto \begin{pmatrix} \{b\} \cup L'.L' \\ \{a\}.L \end{pmatrix}$$

$$f_2 \begin{pmatrix} \varepsilon(L) \\ \varepsilon(L') \end{pmatrix} = \begin{pmatrix} \varepsilon(\{b\} \cup L'.L') \\ \varepsilon(\{a\}.L) \end{pmatrix} = \begin{pmatrix} \cancel{\varepsilon(\{b\})} \cup \varepsilon(L') \cap \varepsilon(L') \\ \cancel{\varepsilon(\{a\})} \cap \varepsilon(L) \end{pmatrix} = \begin{pmatrix} \varepsilon(L') \\ \emptyset \end{pmatrix}$$

$$\text{D'où } f_2^0 \begin{pmatrix} \emptyset \\ \emptyset \end{pmatrix} = \begin{pmatrix} \emptyset \\ \emptyset \end{pmatrix} \text{ et } f_2^1 \begin{pmatrix} \emptyset \\ \emptyset \end{pmatrix} = \begin{pmatrix} \emptyset \\ \emptyset \end{pmatrix} \quad \text{Ainsi, } \mu(f_2) = (\emptyset, \emptyset) \text{ et } \varepsilon \notin L.$$

À vous !

Sur le vocabulaire $V = \{a, b\}$, on considère le système suivant :

$$\begin{cases} L_1 = \{a\} . L_2 \cup \{b\} . L_1 \cup \{\varepsilon\} \\ L_2 = \{a\} . L_1 \cup \{b\} . L_2 \end{cases}$$

1. Calculer $p_a(L_1)$ et $p_a(L_2)$ pour $p_a(L) = \{|w|_a \bmod 2 \mid w \in L\}$.
(la parité du nombre de a dans les mots de L)
2. (★) Montrer que
$$\begin{cases} L_1 = \{w \in V^* \mid |w|_a \text{ pair} \} \\ L_2 = \{w \in V^* \mid |w|_a \text{ impair} \} \end{cases}$$

Solution : question 1

$$f_1 \left(\begin{smallmatrix} X \\ Y \end{smallmatrix} \right) = \left(\begin{smallmatrix} \{a\} \cdot Y \cup \{b\} \cdot X \cup \{\varepsilon\} \\ \{a\} \cdot X \cup \{b\} \cdot Y \end{smallmatrix} \right) \quad \text{et} \quad g \left(\begin{smallmatrix} X \\ Y \end{smallmatrix} \right) = \left(\begin{smallmatrix} p_a(X) \\ p_a(Y) \end{smallmatrix} \right)$$

donc

$$f_2 \left(\begin{smallmatrix} p_a(X) \\ p_a(Y) \end{smallmatrix} \right) = \left(\begin{smallmatrix} (p_a(Y) + 1) \cup p_a(X) \cup \{0\} \\ (p_a(X) + 1) \cup p_a(Y) \end{smallmatrix} \right)$$

avec $X + 1 = \{(n + 1) \bmod 2 \mid n \in X\}$

D'où

$$f_2^1 \left(\begin{smallmatrix} \emptyset \\ \emptyset \end{smallmatrix} \right) = \left(\begin{smallmatrix} \{0\} \\ \emptyset \end{smallmatrix} \right) \quad f_2^2 \left(\begin{smallmatrix} \emptyset \\ \emptyset \end{smallmatrix} \right) = \left(\begin{smallmatrix} \{0\} \\ \{1\} \end{smallmatrix} \right) \quad f_2^3 \left(\begin{smallmatrix} \emptyset \\ \emptyset \end{smallmatrix} \right) = \left(\begin{smallmatrix} \{0\} \\ \{1\} \end{smallmatrix} \right)$$

Ainsi, $p_a(L_1) = \{0\}$ et $p_a(L_2) = \{1\}$,

c.-à-d. tous les mots de L_1 ont un nombre pair de a
et tous les mots de L_2 ont un nombre impair de a .

Solution : question 2

On note $L'_1 \stackrel{\text{def}}{=} \{w \in V^* \mid |w|_a \text{ pair}\}$ et $L'_2 \stackrel{\text{def}}{=} \{w \in V^* \mid |w|_a \text{ impair}\}$.

La question précédente montre déjà que $L_1 \subseteq L'_1$ et $L_2 \subseteq L'_2$.

Montrons donc $L'_1 \subset L_1$ et $L'_2 \subseteq L_2$.

Soit $w \in V^*$. On montre $(w \in L'_1 \implies w \in L_1) \wedge (w \in L'_2 \implies w \in L_2)$ par récurrence sur $|w|$.

- $|w| = 0$: Alors $w = \varepsilon$ donc $w \in L'_1$, $w \in L_1$ et $w \notin L'_2$.
- $|w| > 0$: On distingue le premier symbole de w .
 - a : Alors $w = aw'$. Si $w \in L'_1$, alors $w' \in L'_2$ donc, par hypothèse de récurrence sur w' , $w' \in L_2$. Ainsi $aw' \in \{a\}.L_2 \subseteq L_1$.
Même raisonnement si $w \in L'_2$.
 - b : Alors $w = bw'$. Si $w \in L'_1$, alors $w' \in L'_1$ donc, par hypothèse de récurrence sur w' , $w' \in L_1$. Ainsi $bw' \in \{b\}.L_1 \subseteq L_1$.
Même raisonnement si $w \in L'_2$.

Ainsi, on a bien montré $L_1 = L'_1$ et $L_2 = L'_2$.