Automates et Langages

Partie 2

Emmanuelle Grislin

Université Polytechnique HAUTS-DE-FRANCE

INSA 3 FISE et FISA Informatique

Partie 2

Mars 2021

Objectifs du cours 2

- Savoirs :
  - *connaissance de la hiérarchie de Chomsky*
  - *connaissance du théorème de Kleene*
  - *connaître les opérations sur les langages qui conservent et celles qui ne conservent pas la propriété de type "algébrique"*
- Savoir-faire :
  - *établir si un mot m appartient à un langage L de type 3*
  - *donner le langage reconnu par une grammaire de 3*
  - *donner une grammaire engendrant un langage de type 3*
  - *reconnaître le type d'un langage (principes)*
  - *montrer qu'un langage est algébrique en utilisant les opérations sur les langages*

Appartenance d'un mot à un langage

Problème

Un langage L étant fixé, comment savoir si un mot donné appartient à L ?

Une solution

Décrire une grammaire engendrant L qui va permettre :

- de produire tous les mots de L
- de savoir si un mot donné appartient à L

Difficulté

Complexité de la grammaire ?
Il existe différents "niveaux de complexité" de langage, et donc des grammaires engendrant ces langages : voir section suivante.

Hiérarchie de Chomsky

Noam Chomsky (années 50) :

- complexité d'une grammaire // complexité des algorithmes associés
- complexité d'une grammaire // forme des règles de production
- classification des grammaires formelles en 4 types :
  - type 3 $\subset$ type 2 $\subset$ type 1 $\subset$ type 0
  - le type 3 est le plus simple

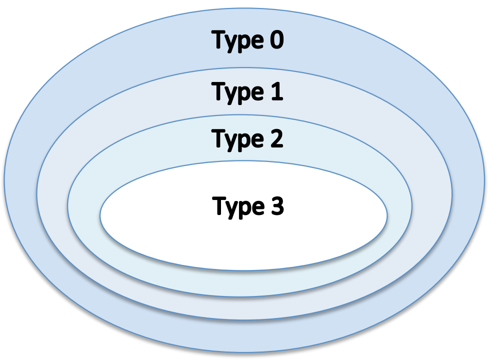


Figure – Types de grammaires

Un langage L est de type i
ssi
il existe une grammaire G
de type i et telle que

$$L = L(G)$$

Grammaire de type 3 : régulière

Grammaire régulière (ou "rationnelle")

Grammaire définie par un quadruplet $G = <\Sigma, V, S, R>$ avec

- Σ : ensemble fini de symboles terminaux
- V : ensemble fini de variables (symboles non terminaux, $\notin \Sigma$)
- S : symbole de V particulier appelé "axiome" ou "racine"
- R : ensemble fini de règles de production

  Grammaire **régulière à droite** *(resp. à gauche)* :

  toutes les règles sont de la forme :

$$X \rightarrow \omega Y \qquad (resp.\ X \rightarrow Y\omega)$$

$$X \rightarrow \omega \qquad \text{avec } X, Y \in V \text{ et } \omega \in \Sigma^*$$

- langages reconnus par les automates à états finis et les expressions régulières (vus en Partie 1)

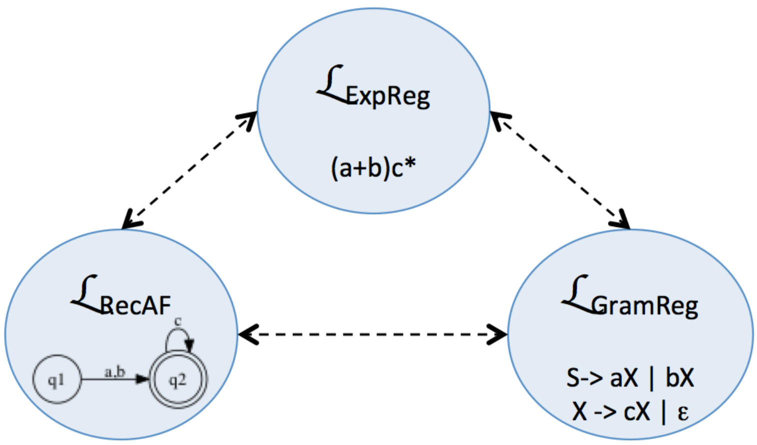
Théorème de Kleene

Théorème de Kleene

Equivalence : $\mathcal{L}_{RecAFD} = \mathcal{L}_{ExpReg} = \mathcal{L}_{GramReg}$

avec

- $\mathcal{L}_{RecAFD}$ la classe des langages reconnaissables par un automate fini
- $\mathcal{L}_{ExpReg}$ la classe des langages qui peuvent être décrits par une expression régulière
- $\mathcal{L}_{GramReg}$ la classe des langages engendrés par une grammaire régulière



Transformation automate fini en grammaire

Pour tout automate fini $\mathcal{A} =< \Sigma \cup \{\epsilon\}, Q, q_0, F, \Delta >$,
il existe une grammaire régulière à droite qui génère $L(A)$:

$G =< \Sigma, V, S, R >$ avec

- $V = Q$ (ensemble des variables = ensemble des états),
- S =symbole de V associé à q_0 (axiome associé à l'état initial),
- $R = \{X \rightarrow \omega Y | (q_X, \omega, q_Y) \in \Delta\} \cup \{X \rightarrow \epsilon | X \in F\}$

Transformation grammaire régulière à droite en automate

Pour toute grammaire régulière à droite $G =< \Sigma, V, S, R >$,
il existe un automate fini qui reconnaît $L(G)$:

$\mathcal{A} =< \Sigma \cup \{\epsilon\}, Q, q_0, F, \Delta >$, avec

- Q : un état q_X pour chaque symbole X non terminal (de V),
- l'état initial q_0 correspond à l'axiome S,
- F : états dont les non terminaux X de G ont une règle du type
  $X \rightarrow \epsilon$
- $\Delta = \{(q_X, \omega, q_Y) | \exists\, (X \rightarrow \omega Y) \in R\}$

Grammaire de type 2 : algébrique (ou ”hors contexte” ou ”non contextuelle”)

Grammaire algébrique

- Règles de production de la forme :

$$T \rightarrow u$$
$$\text{avec } T \in V \text{ et } u \in (\Sigma \cup V)^*$$

- la partie gauche de la règle contient un unique non terminal

- le type de la plupart des langages de programmation
  ex. :
  - langage C : http://www.cs.man.ac.uk/~pjj/bnf/c_syntax.bnf
  - Prolog : http://cseweb.ucsd.edu/classes/fa09/cse130/misc/prolog/prolog_tutorial.pdf
- besoin d'automates à pile pour les reconnaître
- **détaillé dans la section suivante** du cours

Grammaire de type 1 : contextuelle

Grammaire contextuelle (ou ”sensible au contexte”)

- Règles de production de la forme :

$$\alpha \rightarrow \beta$$
$$\text{avec } \alpha \text{ et } \beta \in (\Sigma \cup V)^* \text{ et } |\beta| \geq |\alpha|$$

- partie gauche de la règle est non vide
- et partie droite contient plus de symboles que la gauche avec une exception pour $X \rightarrow \epsilon$

- toute grammaire de type 1 peut aussi s'écrire :

$$\gamma X \delta \rightarrow \gamma \beta \delta$$
$$\text{avec } \gamma, \beta, \delta \in (\Sigma \cup V)^*,\ |\beta| \geq 1,\ X \in V$$

- γ et δ sont appelés les **contextes** (gauche et droit) de la règle

Grammaire de type 0 : générale

Grammaire générale (ou "non contrainte")

- Règles de production de la forme :

$$\alpha \rightarrow \beta$$
$$\text{avec } \alpha \text{ et } \beta \in (\Sigma \cup V)^*$$

- Pas de contrainte sur les parties gauches et droites des règles.

Type d'un langage

Comment connaître le type d'un langage L ?

- si il existe une **expression régulière** ou un **automate** à états fini qui reconnaît L, alors
  L est de type 3 (régulier) par le théorème d'équivalence de Kleen
- si il existe une grammaire G de type i qui engendre L,
  alors L est au moins de type i (peut-être $> i$ si G est d'une complexité plus grande que nécessaire)
  - on regarde **la forme des règles** des grammaires qui l'engendrent : existe-t-il une grammaire de type 3 ? sinon, de type 2 ? etc.
- par déduction en utilisant **les propriétés sur les langages** :

  ex. :
  - l'union de 2 langages algébriques est un langage algébrique
  - le **lemme de l'Etoile** donne une condition que les mots d'un langage régulier doivent nécessairement satisfaire (Cf. cours Partie 1)

Type d'un langage

Comment connaître le type d'un langage L ?

- si il existe une **expression régulière** ou un **automate** à états fini qui reconnaît L, alors
  L est de type 3 (régulier) par le théorème d'équivalence de Kleen
- si il existe une grammaire G de type i qui engendre L,
  alors L est au moins de type i (peut-être $> i$ si G est d'une complexité plus grande que nécessaire)
  - on regarde **la forme des règles** des grammaires qui l'engendrent : existe-t-il une grammaire de type 3 ? sinon, de type 2 ? etc.
- par déduction en utilisant **les propriétés sur les langages** :

Union de langages algébriques

Soient les langages L_1 et L_2 deux langages algébriques sur l'alphabet Σ,

Union :

- $L_1 \cup L_2 = \{u \in \Sigma^* \mid u \in L_1 \text{ ou } u \in L_2\}$
- l'union de 2 langages algébriques est algébrique

Démonstration :

- $L_1 = L(G_1)$ avec $G_1 = \langle \Sigma, V_1, S_1, R_1 \rangle$ et $L_2 = L(G_2)$ avec $G_2 = \langle \Sigma, V_2, S_2, R_2 \rangle$ où $V_1 \cap V_2 = \emptyset$ (ensembles de variables disjoints),
- soit S une nouvelle variable,
  $L_1 \cup L_2$ est engendré par la grammaire algébrique :
  $G = \langle \Sigma, V_1 \cup V_2 \cup \{S\}, S, R_1 \cup R_2 \cup \{S \to S_1 \mid S_2\} \rangle$
- la règle ajoutée est de la forme $T \to u$ avec $T \in V$ et $u \in (\Sigma \cup V)^*$

Union de langages algébriques

Exemple :

- $L = \{u \in \{a,b\}^* \mid u = a^n b^n \text{ ou } u = b^n a^n\,;\, n \geq 0\}$
- $L = \{u \in \{a,b\}^* \mid u = a^n b^n\,;\, n \geq 0\} \cup \{u \in \{a,b\}^* \mid u = b^n a^n\,;\, n \geq 0\}$
- $L_1 = L(G_1)$ avec $G_1 = \langle \{a,b\}, \{S_1\}, S_1, \{S_1 \to aS_1b \mid \epsilon\}\rangle$
  et $L_2 = L(G_2)$ avec $G_2 = \langle \{a,b\}, \{S_2\}, S_2, \{S_2 \to bS_2a \mid \epsilon\}\rangle$
- L est engendré par la grammaire algébrique :
  $G = \langle \{a,b\}, \{S_1, S_2, S\}, S, R\rangle$ avec R :

$$\begin{array}{rcl|l} S & \to & S_1 & S_2 \\ S_1 & \to & aS_1b & \epsilon \\ S_2 & \to & bS_2a & \epsilon \end{array}$$

Produit de langages algébriques

Soient les langages L_1 et L_2 deux langages algébriques sur l'alphabet Σ,

Produit :

- $L = L_1\, L_2 = \{u \in \Sigma^* \mid \exists v_1 \in L_1 \text{ et } v_2 \in L_2,\; u = v_1\, v_2\}$
- le produit de 2 langages algébriques est algébrique

Démonstration :

- $L_1 = L(G_1)$ avec $G_1 = \langle \Sigma, V_1, S_1, R_1\rangle$ et $L_2 = L(G_2)$ avec $G_2 = \langle \Sigma, V_2, S_2, R_2\rangle$ où $V_1 \cap V_2 = \emptyset$ (ensembles de variables disjoints),
- soit S une nouvelle variable,
  $L_1\, L_2$ est engendré par la grammaire algébrique :
  $G = \langle \Sigma, V_1 \cup V_2 \cup \{S\}, S, R_1 \cup R_2 \cup \{S \to S_1\; S_2\}\rangle$
- la règle ajoutée est de la forme $T \to u$ avec $T \in V$ et $u \in (\Sigma \cup V)^*$

Produit de langages algébriques

Exemple :

- $L = \{u \in \{a,b\}^* \mid u = a^n b^n b^p c^p \,;\, n \geq 0 \text{ et } p \geq 0\}$
- $L = L_1 L_2$ avec $L_1 = \{u \in \{a,b\}^* \mid u = a^n b^n \,;\, n \geq 0\}$ et $L_2 = \{u \in \{a,b\}^* \mid u = b^p c^p \,;\, p \geq 0\}$
- $L_1 = L(G_1)$ avec $G_1 = \langle \{a,b\}, \{S_1\}, S_1, \{S_1 \to aS_1b \mid \epsilon\}\rangle$
  et $L_2 = L(G_2)$ avec $G_2 = \langle \{a,b\}, \{S_2\}, S_2, \{S_2 \to bS_2c \mid \epsilon\}\rangle$
- L est engendré par la grammaire algébrique :
  $G = \langle \{a,b\}, \{S_1, S_2, S\}, S, R\rangle$ avec R :

$$\begin{array}{lcl} S & \to & S_1S_2 \\ S_1 & \to & aS_1b \mid \epsilon \\ S_2 & \to & bS_2c \mid \epsilon \end{array}$$

Etoile de langages algébriques

Soit le langage algébrique L sur l'alphabet Σ,

Etoile :

- $L^* = \bigcup_{n \geq 0} L^n = L^0 \cup L^1 \cup L^2 \cup \ldots L^n$
- l'étoile d'un langage algébrique est algébrique

Démonstration :

- L^* est l'union de produits de langages algébriques $\Rightarrow$ algébrique
- L engendré par $G = \langle \Sigma, V, S, R\rangle$
  Soit S' une nouvelle variable,
  L^* est engendré par la grammaire algébrique :
  $G' = \langle \Sigma, V \cup \{S'\}, S', R\{S' \to S\ S' \mid \epsilon\}\rangle$
- la règle ajoutée est de la forme $T \to u$ avec $T \in V$ et $u \in (\Sigma \cup V)^*$

Etoile de langages algébriques

Exemple :

- Avec $L = \{u \in \{a,b\}^* \mid u = a^n b^n \,;\, n \geq 0\}$,
  - $L^0 = \emptyset$, $L^1 = L$
  - $L^2 = LL^1 = \{u \in \{a,b\}^* \mid u = a^n b^n a^p b^p \,;\, n \geq 0 \text{ et } p \geq 0\}$ algébrique car produit de 2 algébriques
  - $L^3 = LL^2 = \{u \in \{a,b\}^* \mid u = a^n b^n a^p b^p a^q b^q \,;\, n \geq 0,\ p \geq 0 \text{ et } a \geq 0\}$ algébrique car produit de 2 algébriques
  - ...
  - $L^* = \bigcup_{n \geq 0} L^n$ algébrique car union d'algébriques
- L est engendré par la grammaire algébrique :
  $G = \langle \{a,b\}, \{S, S'\}, S', R \rangle$ avec R :

$$\begin{array}{rcll} S' & \rightarrow & SS' & \mid \epsilon \\ S & \rightarrow & aSb & \mid \epsilon \end{array}$$

Intersection de langages algébriques

Soit le langage algébrique L sur l'alphabet Σ,

Intersection :

- $L_1 \cap L_2 = \{u \in \Sigma^* \mid u \in L_1 \text{ et } u \in L_2\}$
- l'intersection de 2 langages algébriques n'est pas toujours algébrique

Intersection de langages algébriques

Un exemple de "conservation" du type algébrique :

- $L = \{u \in \{a,b\}^* \mid u = a^n b^p \,;\, p > n \geq 0\}$
  L est algébrique, engendré par la grammaire algébrique :
  $G = \langle \{a,b\}, \{S,T\}, S, R \rangle$ avec R :

$$\begin{array}{lcccc} S & \rightarrow & aSb & \mid & T \\ T & \rightarrow & bT & \mid & b \end{array}$$

- $L = L_1 \cap L_2$ avec :
- $L_1 = \{u \in \{a,b\}^* \mid u = a^n b^p \,;\, p \geq n \geq 0\}$
  et $L_2 = \{u \in \{a,b\}^* \mid u = a^n b^p \,;\, p \geq 0, n \geq 0, n \neq p\}$
  - $L_1 = \{a^n b^n\}\{b^q\}$ produits d'algébriques
  - $L_2 = \{a^n b^p \mid p < n\} \cup \{a^n b^p \mid p > n\}$ même raisonnement que pour L_1
- L est donc algébrique et intersection de 2 langages algébriques

Intersection de langages algébriques

Un exemple de non "conservation" du type algébrique :

- $L_1 = \{u \in \{a,b\}^* \mid u = a^n b^p c^n \,;\, p \geq 0, n \geq 0\}$
  L_1 est algébrique
- $L_2 = \{u \in \{a,b\}^* \mid u = a^n b^p c^p \,;\, p \geq 0, n \geq 0\}$
  L_2 est algébrique
- $L_1 \cap L_2 = \{u \in \{a,b\}^* \mid u = a^n b^n c^n \,;\, n \geq 0\}$
  $L_1 \cap L_2$ n'est pas algébrique (il est contextuel)

- **l'intersection ne conserve pas nécessairement le type algébrique**

Complémentaire de langage algébrique

Soit le langage algébrique L sur l'alphabet Σ,

Complémentaire :

- $\complement L = \{u \in \Sigma^* \mid u \notin L\}$
- le complémentaire d'un langage algébrique n'est pas toujours algébrique

Complémentaire de langage algébrique

Exemple :

- Soient L_1 et L_2, 2 langages algébriques avec $L_1 \cap L_2$ non algébrique.
- 3 cas possibles :
  1. $\complement L_1$ et $\complement L_2$ sont algébriques :
     $\complement L_1 \cup \complement L_2 = \{u \in \Sigma^* \mid u \notin L_1$ ou $u \notin L_2\}$ alg. car union de 2 alg.
     $\complement(\complement L_1 \cup \complement L_2) = \{u \in \Sigma^* \mid u \in L_1$ et $u \in L_2\} = L_1 \cap L_2$ non alg. par hyp.
     Exemple où le $\complement$ d'un algébrique est non algébrique.
  2. $\complement L_1$ algébrique et $\complement L_2$ non algébrique :
     L_2 est algébrique et $\complement L_2$ non algébrique.
  3. $\complement L_1$ non algébrique et $\complement L_2$ algébrique :
     L_1 est algébrique et $\complement L_1$ non algébrique.

- **le complémentaire ne conserve pas nécessairement le type algébrique**

Conclusion du cours 2 : objectifs atteints ?

Savoirs :

- connaissance de la hiérarchie de Chomsky :
  - Un langage L est de type i ssi il existe une grammaire G de type i telle que $L = L(G)$
  - 4 niveaux de complexité :
    - type 3 : régulier, il existe une expression régulière et règles de la forme $X \to \omega Y$ *(resp.* $X \to Y\omega$*)* ou $X \to \omega$ avec $X, Y \in V$ et $\omega \in \Sigma^*$
    - type 2 : algébrique (hors contexte), règles de la forme
      $$X \to u \text{ avec } X \in V \text{ et } u \in (\Sigma \cup V)^*$$
    - type 1 : contextuel, règles de la forme
      $$\alpha \to \beta \text{ avec } \alpha \text{ et } \beta \in (\Sigma \cup V)^* \text{ et } |\beta| \geq |\alpha|$$
    - type 0 : général (sans contrainte), règles de la forme
      $$\alpha \to \beta \text{ avec } \alpha \text{ et } \beta \in (\Sigma \cup V)^*$$
- connaissance du théorème de Kleene : $\mathcal{L}_{RecAFD} = \mathcal{L}_{ExpReg} = \mathcal{L}_{GramReg}$
- connaître les opérations sur les langages qui conservent et celles qui ne conservent pas le type"algébrique" :
  - l'union, le produit et l'étoile conservent le type "algébrique"
  - l'intersection et le complémentaire ne conservent pas nécessairement le type "algébrique"

Conclusion du cours 2

Objectifs atteints ?

Savoir-faire : Cf. TDs

- établir si un mot m appartient à un langage L de type 3 (régulier) : donner une grammaire pour L et une dérivation de S à m
- donner le langage reconnu par une grammaire de type 3 : sous forme d'expression régulière
- donner une grammaire engendrant un langage de type 3 : soit directement, soit à partir d'un automate à états finis
- reconnaître le type d'un langage (principes) :
  1. par élimination en partant du type le plus contraint (type 3)
  2. en produisant une grammaire engendrant ce langage
  3. en le "décomposant" en tant que résultat d'opérations (union, produit ou étoile) sur des langages algébriques