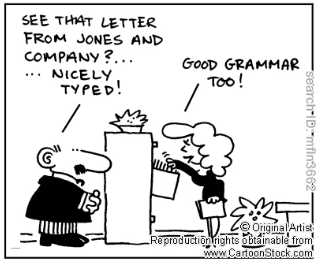
4 – Grammaires régulières

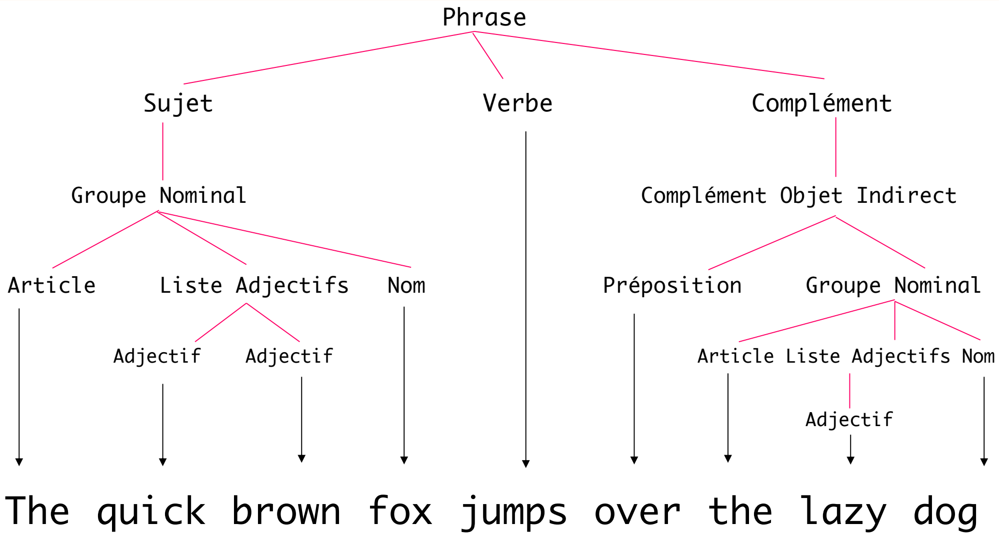


Origine

- dans les années 50 :
  tentatives de formalisation des langues naturelles
- but :
  décrire précisément les règles permettant de construire des phrases syntaxiquement correctes d'une langue
- semi-échec de la linguistique mais réussite totale pour des langues bien plus simples :

  les langages informatiques !

Exemple



Réécriture

- mécanisme qui permet d'engendrer des familles d'objets (figures géométriques, étiquettes de graphes, polyominos, mots, fractales ...)
- on dispose d'un objet initial : l'axiome A
- ainsi que de règles de réécriture i.e. un ensemble d'éléments du type

  $\alpha \rightarrow \beta$ (lire : *l'objet α se réécrit en celui β.*)
- elle s'appuie sur une relation de dérivation binaire transitive notée $\Rightarrow^*$ et définie par :
  - si $\alpha \rightarrow \beta$ alors $\alpha \Rightarrow \beta$
  - si $\alpha \rightarrow \beta$ et si en remplaçant dans γ l'objet α par celui β on obtient δ alors $\gamma \Rightarrow \delta$
  - $\Rightarrow^*$ est la fermeture transitive de $\Rightarrow$
- l'ensemble engendré est : $\{\varphi \,,\ A \Rightarrow^* \varphi\}$

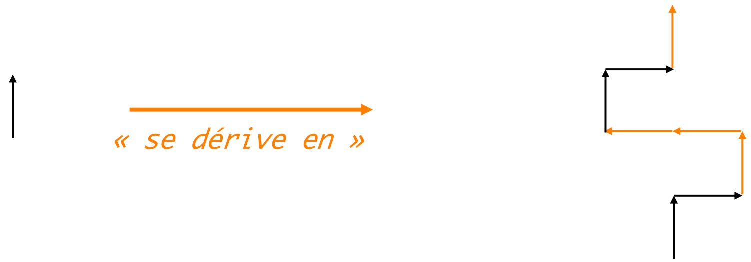
Application : image de synthèse

Exemple

Une seule règle de réécriture :
(F signifie « flèche »)

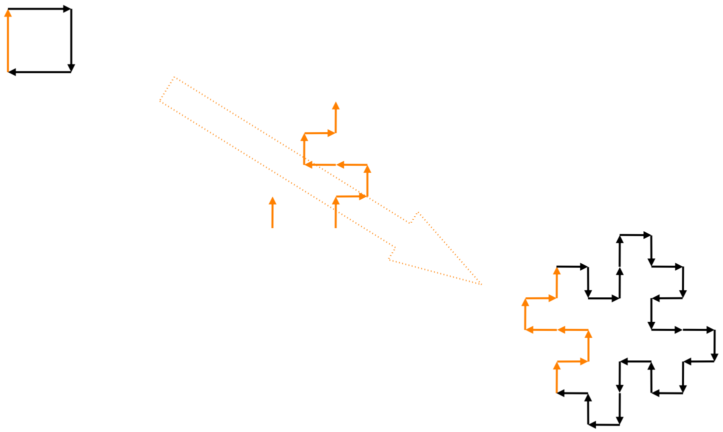
F → F + F - F - FF + F + F - F

(+ signifie que l'on tourne à droite
 - signifie que l'on tourne à gauche)

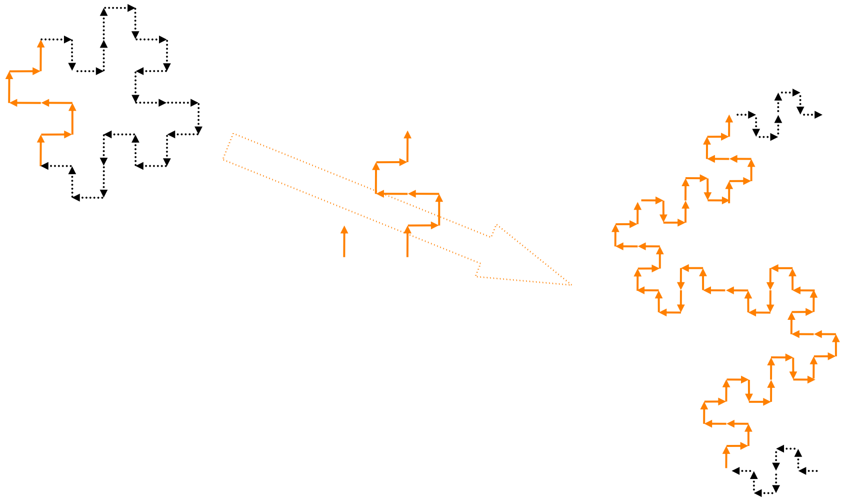


Une étape de réécriture

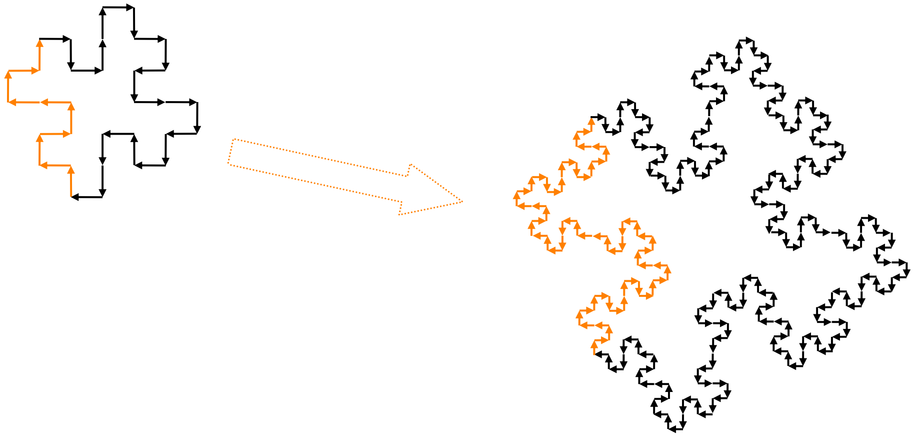
F + F + F + F



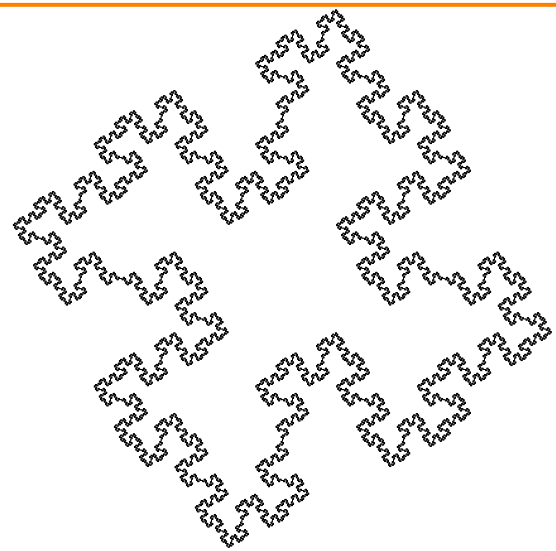
Début de l'étape 2



2^e étape entière



Avec la tortue de Python



Génération de langages

- un automate fini accepte les mots d'un langage rationnel
  on dit que le langage rationnel est reconnu par cet automate
  un automate est donc un système de reconnaissance
- une expression régulière décrit un langage rationnel,
  c'est un formalisme pour leur éviter la notation ensembliste
- mais comment énumérer les mots d'un langage rationnel ?
  dès lors qu'il est infini, *a priori* c'est impossible ...

  Problème *comment engendrer de façon automatique*
  tous les mots d'un langage rationnel donné ?*

- si le langage est fini, on les énumère tous
- s'il est infini, on veut énumérer des mots de longueur arbitrairement grande

*rationnel, juste pour commencer ...

Grammaire

- Une grammaire G est un quadruplet (N,T,P,A) où :
  - N est un ensemble fini de symboles non-terminaux
  - T est l'alphabet de symboles terminaux
  - P est un ensemble fini de productions (ou de règles)
    $\{ \varphi \rightarrow \psi \ , \ \varphi \in (N \cup T)^+, \ \psi \in (N \cup T)^* \}$
  - A est l'axiome
- Propriétés
  - le vocabulaire est $V = N \cup T$
  - $N \cap T = \varnothing$
  - (P,A) est un système de réécriture.
- Le langage engendré par la grammaire G est
  - $L(G) = \{ \varphi \in T^*, \ A \Rightarrow^* \varphi \}$
  - L(G) : ensemble des mots sur T dérivant de l'axiome A.

Exemples

- $G = (N,T,P,A)$ avec :
  - $N = \{A, B\}$ ensemble des non-terminaux
  - $T = \{a, b\}$ ensemble des terminaux
  - $P = \{ A \to a\,A,\ A \to b\,B,\ B \to b\,B,\ B \to \varepsilon \}$
  - A est l'axiome

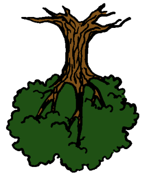
☞ L(G) est décrit par a^*b^+

- $G' = (N',T',P',A)$ avec :
  - $N' = \{ A \}$, axiome A
  - $T' = \{ 0, 1 \}$
  - $P' = \{ A \to 0\,A\,1,\ A \to \varepsilon \}$

☞ $L(G') = \{0^n 1^n,\ n \geq 0\}$ *(non-rationnel)*

ATTENTION
à ne pas inventer une expression régulière pour un langage non rationnel ...

Arbre de dérivation



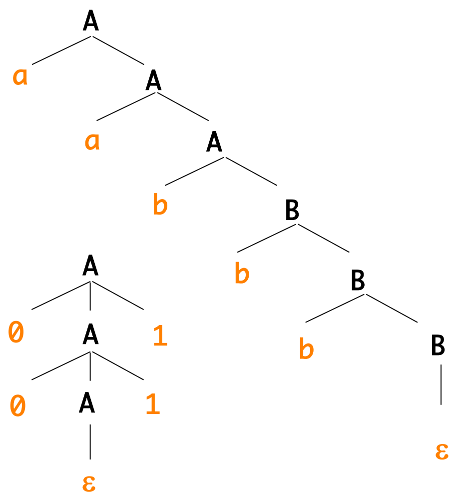
Soit $G = (N,T,P,A)$ une grammaire avec des productions avec *un seul non-terminal par partie gauche.*

- un arbre de dérivation pour un mot w engendré par G est un arbre dont :
  - la racine est étiquetée par l'axiome A
  - les feuilles sont étiquetées par des éléments de
    $$T \cup \{\varepsilon\}$$
  - les nœuds internes le sont par des éléments de N
  - un nœud interne étiqueté B a des fils étiquetés de gauche à droite $\alpha_1, \alpha_2, \ldots \alpha_n$, s'il existe dans P une production :
    $$B \to \alpha_1\,\alpha_2\ \ldots\ \alpha_n$$
  - w est formé de la concaténation des feuilles lues dans un parcours en profondeur de l'arbre.

Exemples

- $G = (N,T,P,A)$ avec :
  - $N = \{ A, B \}$
  - $T = \{ a, b \}$
  - $P =\{ A \to a\,A \mid b\,B,\ B \to b\,B \mid \varepsilon \}$

- $G' = (N',T',P',A)$ avec :
  - $N' = \{ A \}$
  - $T' = \{ 0, 1 \}$
  - $P' = \{ A \to 0\,A\,1 \mid A \to \varepsilon \}$



Le symbole | permet juste de regrouper les partie droites d'une même partie gauche.

Grammaires régulières*

- Une grammaire (N,T,P,A) est régulière *à droite* si les productions de P sont de la forme

  $B \to a\,C$
  ou $B \to a$
  ou $B \to \varepsilon$

  avec $B \in N$, $C \in N$ et $a \in T$

- Une grammaire (N,T,P,A) est régulière *à gauche* si les productions de P sont de la forme :

  $B \to C\,a$
  ou $B \to a$
  ou $B \to \varepsilon$

  avec $B \in N$, $C \in N$ et $a \in T$

il existe un algorithme pour passer d'une grammaire régulière à gauche à une grammaire régulière à droite engendrant le même langage (cf. TD 4)

Théorème un langage L est **rationnel** si et seulement si il existe une **grammaire régulière** qui engendre les mots de L.

* aussi dites grammaires linéaires.

Exemple & contre-exemple

- G = (N,T,P,A) est régulière à droite :
  - P = { A → a A | b B
    B → b B | ε }
  - L(G) est décrit par l'expression régulière : $a^* b^+$
- G' = (N',T',P',A) n'est pas une grammaire régulière :
  - P' = { A → 0 B | ε
    B → A 1 }
  - L(G') = { $0^n 1^n$, $n \geq 0$ }

Un langage non-rationnel <u>ne peut pas</u> être engendré par une grammaire régulière.

Ainsi, une grammaire obtenue en panachant les productions d'une grammaire régulière à gauche et celle d'une grammaire régulière à droite n'est généralement pas une grammaire régulière.

A.F.D. → grammaire régulière *

Le théorème précédent se prouve à l'aide d'une double construction.

Premier sens :

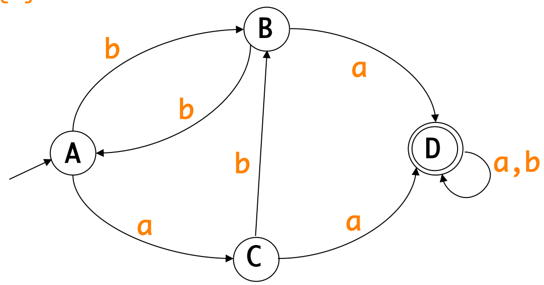
- soit un automate fini déterministe donné :
  $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$
- construisons G = (N,T,P,A) une grammaire régulière pour L($\mathcal{A}$) :
  - l'ensemble des non-terminaux N est celui des états Q de $\mathcal{A}$ **
  - l'ensemble des terminaux T égale Σ
  - l'axiome A de G correspond à l'état q_0
  - pour tout (q,σ,q') de δ il crée la production $q \to \sigma\ q'$ dans P **
  - pour tout f de F, il crée la production $f \to \varepsilon$ dans P **

* on pourrait aussi partir d'un A.F.N.
** à un renommage près.

Exemple

$\mathcal{A} = (\Sigma, Q, \delta, A, F)$

Σ = {a,b}, Q = {A,B,C,D}, F = {D}



G = (N,T,P,A)

N = {A,B,C,D}
T = {a,b}
P = { A → a C
A → b B
B → a D
B → b A
C → a D
C → b B
D → a D
D → b D
D → ε }

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

Grammaire régulière → A.F.N.

Construction réciproque :

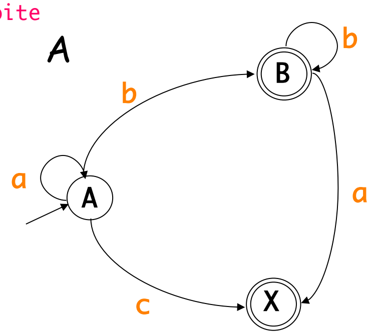
- soit G = (N,T,P,A) une grammaire régulière à droite
  construisons un automate fini $\mathcal{A}$ qui accepte L(G) :
  $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$
- on ajoute un nouveau non-terminal X pour les productions $B \to \alpha$
  et on les remplace par $B \to \alpha\ X$ et $X \to \varepsilon$
- les états de Q correspondent aux non-terminaux $N \cup \{X\}$ *
- l'alphabet Σ coïncide avec l'ensemble T des terminaux
- l'état q_0 correspond à l'axiome A de la grammaire
- pour toute production $B \to \alpha\ C$ de P, on crée (B,α,C) * dans δ
- pour toute production $B \to \varepsilon$, l'état correspondant à B est final dans $\mathcal{A}$.

* à un renommage près.

Exemple

G = (N,T,P,A) une grammaire régulière à droite

N = {A,B}
T = {a,b,c}
P = { A → a A
A → b B
A → c   (devient A → c X et X → ε)
B → b B
B → a   (devient B → a X)
B → ε   }



$\mathcal{A} = (\Sigma, Q, \delta, A, F)$

Σ = {a,b,c}
Q = {A,B,X}
F = {B,X}

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

E : $a^* b^+ (\varepsilon + a) + a^* c$