Langages et Automates

Cours : CM1 Introduction

Clément AGRET

3 x 45 minutes

5 min pause

12h30 → 13h15

13h20 → 14h05

14h10 → 14h55

ok

Compilateur

Un compilateur est un utilitaire de traduction. Qui traduit un `programme` d'un langage source vers un langage cible, en signalant d'éventuelles erreurs.

Programmeur (Humain)	Système d'exploitation
C/C++, Pascal, Algol, FORTRAN, assembleur, ..	Exécutable(Binaire)
for (i = 0; i <= 10; i++) s += 1;	1000111010100

gcc

exec

langage source → **compilateur** → langage cible

C

↓ ?

erreurs

Compilateur

$1954/57 \rightarrow$ FORTRAN (Mathematical FORmula TRANslating System)
Plus tard, $\rightarrow$ PASCAL
Nombreux autres compilateurs ont suivi $\rightarrow$ YACC, (Yet Another Compiler of Compiler).
$1955/65 \rightarrow$ Des linguistes, philosophes et mathématiciens ont défriché la partie théorique en proposant une description et une classification des langages et des grammaires.

Les tâches d'analyse d'un compilateur.

$A + B = C$ $\qquad$ $(A + B)\cancel{)} = C$ trop

Les premières tâches d'un compilateur sont de faire :

1) l'analyse lexicale : reconnaître les éléments constitutifs de la chaîne entrée, c'est-à-dire du code source, et en dresser la liste.
2) l'analyse syntaxique : vérifier la conformité avec les règles de constitution du code. Par exemple, l'expression **(A+B)) = C** est syntaxiquement incorrecte (parenthèses...).
3) l'analyse sémantique : analyser le sens et fixer une interprétation.

Par exemple dans **si A alors si B alors C sinon D**, choisir à quel **si** se rapporte le **sinon**. Dans ce qui suit, nous nous occuperons essentiellement de l'analyse syntaxique.

Si A alors
Si ~~B~~ A alors
C
Sinon $\boxed{D}$

$\neq$

Si A alors
Si B alors $\boxed{C}$
Sinon $\boxed{D.}$

La notion de grammaire.

$\Sigma = \{L, E, V, I, U, X, C, H, A, T, R, \sqcup\}$

espace ⊔ Space.

Considérons, par exemple, la phrase suivante :

LE VIEUX CHAT ATTRAPE LE PETIT RAT

Le but est de construire une grammaire qui permette de produire cette phrase.
Il faut tout d'abord préciser les « ingrédients » nécessaires :
Ce sera l'alphabet des symboles ou lexique.
Dans notre cas, nous pouvons prendre comme alphabet l'ensemble:

$$\Sigma = \{\text{LE}, \text{VIEUX}, \text{PETIT}, \text{CHAT}, \text{RAT}, \text{ATTRAPE}\}$$

$\boxed{\Sigma}$ ok ?

Analyse syntaxique.

Σ du langage C : $\Sigma = \{\text{int}, \text{char}, \text{for}, \text{do}, \ldots\}$

français

Noter que le terme «alphabet » n'a pas ici le sens habituel A, B, C,...
L'alphabet contient les «briques de base» avec lesquelles on peut former tout ce qu'on veut former.
Dans un langage de programmation, les mots réservés, balises, etc...
Comme : program, real, <head>, </head>, sont dans l'alphabet de symboles.

Analyse syntaxique.

⚠ FRANÇAIS ≠ C++ ≠ Chinois.

Voyons maintenant comment les symboles sont assemblés.
Dans la structure de la phrase, on peut distinguer :

- un groupe sujet
- un verbe
- un groupe complément d'objet (CO)

Les groupes sujet et CO sont eux-mêmes des groupes nominaux ; un groupe nominal est formé d'un article suivi d'un nom, lui-même précédé ou suivi d'adjectifs.

Analyse syntaxique.

LA VIEUX CHAT MANGE LE PETIT RAT

phrase → LE VIEUX RAT ATTRAPE LE VIEUX RAT

Voici un exemple de (petite) grammaire pouvant produire notre phrase; elle a 7 règles de grammaire (on dit aussi de production, ou de réécriture) :

1. <phrase> → <groupe sujet> <verbe> <groupe CO>
2. <groupe sujet> → <groupe nominal>
3. <groupe nominal> → <article> <adjectif> <nom>
4. <article> → LE, LA
5. <nom> → RAT, CHAT
6. <adjectif> → VIEUX, PETIT
7. <verbe> → ATTRAPE, MANGE

8 : <groupe CO> → <article adjectif, nom>

Syntaxe

OK

Non OK

→ EASY

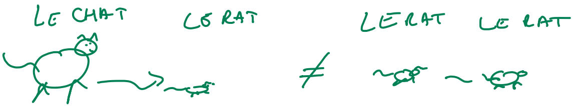
→ HARD

?

Suite

- Le point de départ s'appelle l'axiome. Dans notre exemple, c'est <phrase>.
- Les variables sont les «ingrédients» qui peuvent encore être remplacés par d'autres (<phrase>, <article>, <groupe CO>, etc.).
- Les règles 1 à 3 sont des règles syntaxiques (la première règle concerne toujours l'axiome). Les règles 4 à 7 sont des règles complètement terminales (ou : lexicales).
- Le langage engendré par la grammaire est l'ensemble de toutes les phrases que l'on peut produire à partir de l'axiome en utilisant des règles de grammaire, une phrase étant une chaîne de symboles.

Exemples de phrases



Phrases « grammaticalement correctes», c'est-à-dire des chaînes de symboles que l'on peut produire à partir de l'axiome, en utilisant les règles précédentes :

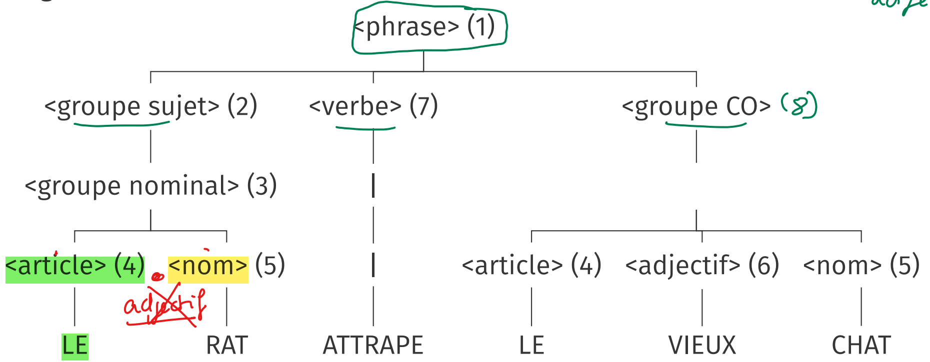
LE RAT ATTRAPE LE VIEUX RAT   c'est ok ?

On peut associer à chaque phrase ainsi produite un arbre de dérivation où figurent les variables auxquelles on a appliqué des règles de grammaire.
Nous avons ajouté aux nœuds de cet arbre de dérivation, pour faciliter la compréhension, les numéros des règles qui ont été appliquées aux variables.

Arbre de dérivation

article nom adjectif
≠
nom article adjectif
≠
adjectif nom article
≠

Un arbre de dérivation pour la phrase LE RAT ATTRAPE LE PETIT CHAT est représenté dans la figure suivante.



LE ~~VIEUX~~ RAT ATTRAPE LE PETIT CHAT est-ce correct ? Non

≠ nom

Définitions.

a partir d'un alphabet de symbole on peut créer une phrase.

L'alphabet des symboles est un ensemble fini. On le notera en général Σ dans la suite du cours.

1 $\Sigma = \{$LE, VIEUX, PETIT, CHAT, RAT, ATTRAPE$\}$
2 $\Sigma = \{0, 1, 2, ..., 9, +, \times, -, /, (,)\}$

Le second alphabet permet d'écrire les expressions arithmétiques sur les nombres entiers.
Il y a notamment les quatre opérations et les parenthèses.
Le premier problème sera de pouvoir distinguer si une expression est « correctement écrite ».

!

Exemple

$2 \times (31 - 6) + 8$ est correcte, mais $\boxed{2(31 - 6) + 8}$ ou $(21+) \times 4$ ne le sont pas.

Une chaîne est une suite finie de symboles.
La longueur d'une chaîne est le nombre de ses symboles.

OK ?

Exemple

Σ_1

$\Sigma^0 = \{\varepsilon\}$ $\qquad \varepsilon^{\not{o}}$

LE LE CHAT est une chaîne de longueur 3 sur le premier alphabet. Les expressions arithmétiques (correctes ou non !) sont des chaînes sur le second alphabet.
La chaîne vide, notée λ ne contient aucun symbole : ϵ; sa longueur est nulle.

- Σ^n : L'ensemble de toutes les chaînes de longueur n.
- $\Sigma^0 = \sigma$.: Attention, cet ensemble n'est pas vide : il contient la chaîne vide ϵ ⚠
- Σ^* : C'est Σ^n pour $n \geq 0$ C'est donc l'ensemble de toutes les chaînes, chaîne vide comprise.
- Σ^+ : Est la réunion desΣ^n pour $n \geq 0$.

$\Sigma^n = \{\Sigma^0 + \Sigma^1 + \Sigma^2 \ldots\}$

$\Sigma = \{a, b\}$

$\Sigma^3 = ?\ \{aaa, aab, aba, \ldots bbb\}$

$n=1 \quad \Sigma^1 = ?\ \{a, b\}$

$n=2 \quad \Sigma^2 = ?\ \{aa, ab, ba, bb\}$

taille Σ^2 $\quad |\Sigma^2| \stackrel{?}{=} 4 \quad |\Sigma^3| \stackrel{?}{=} 2^3 = 8 + 1$

5 ok $\qquad \varepsilon$

Concaténation

$\Sigma = \{a, b\}$ $\Sigma = \{\varepsilon, a, b\}$.

$u = aba$

$v = bbb$.

Q? $u.v = ?$

$uv = \underline{ababbb}$

On peut concaténer des chaînes de symboles, c'est-à-dire les «coller» les unes derrière les autres, de la même façon que plusieurs textes peuvent être assemblés les uns derrière les autres pour former un nouveau texte.
Si u et v sont deux chaînes de symboles, leur concaténation sera notée $u.v$ ou plus simplement uv.
Un langage sur Σ est un sous-ensemble de Σ^*.

Exemples.

Alphabet ≠ langage

Pouvez vous me donner un mot que je n'ai pas listé qui est dans l'ensemble L2

Sur l'alphabet $\Sigma = \{a, b, c, d\}$, les ensembles suivants sont des langages :

- $L_1 = \{ac, abbc\}$
- L'ensemble L_2 de toutes les chaînes qui commencent par a.
- L'ensemble L_3 de toutes les chaînes de longueur paire.

$L_2 = \{a, aa, ab, ac, ad, \ldots$ aaaaaaa ... $\}$

abd ok

Notation.

Afin de faciliter la lecture, nous adopterons en général une notation a, b, c,... pour des symboles de Σ et u, v, w, x, ... pour des chaînes de symboles. (mots)

Symboles: a, b, c

mots:

$u = aab$

$v = bab$

$x = aaacb$

$L_3 = \{aa, ab, ac, ad, bb, \ldots aaaa, \ldots$

Opération sur les langages.

Un alphabet de symboles Σ étant fixé, voyons les opérations que nous appliquerons aux langages sur Σ.

1. Opérations ensemblistes usuelles (réunion, intersection, complémentaire).
2. Concaténation de langages.

ok ?

u et v

$\boxed{\cap \ \cup \ -}$

$u.v \neq v.u$

$u \cap v \qquad \bar{u}$

$u \cup v$

Opération ensemblistes

Σ^* Σ^n et Σ^+

Les langages sur Σ ne sont rien d'autre que des sous-ensembles de Σ^*.
Les opérations habituelles que l'on connaît sur les sous-ensembles s'appliquent à ces langages.

Concaténation de langages

$u.v = uv$

aba
bbb

$\boxed{uv = ababbb.}$

Comme nous savons concaténer des chaînes de symboles, nous pourrons également concaténer deux langages, c'est-à-dire collecter dans un nouvel ensemble, noté $L1.L2$, toutes les chaînes que l'on peut obtenir en concaténant une chaîne de $L1$ avec une chaîne de $L2$.

$L_1 = \{aa, ab\}$

$L_2 = \{ba, bb\}$

$L_1 . L_2 = \{aaba, aabb, abba, abbb\}$

$\boxed{Ok?}$

Concaténation $L1.L2$

$L = \{a, b, ab\}$

$L^2 = \{aa, ab, aab, ba, bb, bab, aba, abb, abab\}$

Par récursivité, nous pourrons concaténer aussi un nombre fini de langages. En particulier, si L est un langage, on notera :

$$L^0 = \{\lambda\}$$

$$L^1 = L$$

$$L^2 = L.L$$

Puis on définit de manière récursive

$L^n = L^{n-1}.L$ (Concaténation de n exemplaires de L

$L^3 = L^2.L$

$L^3 = \{aaa, aba, aaba, \ldots\}$