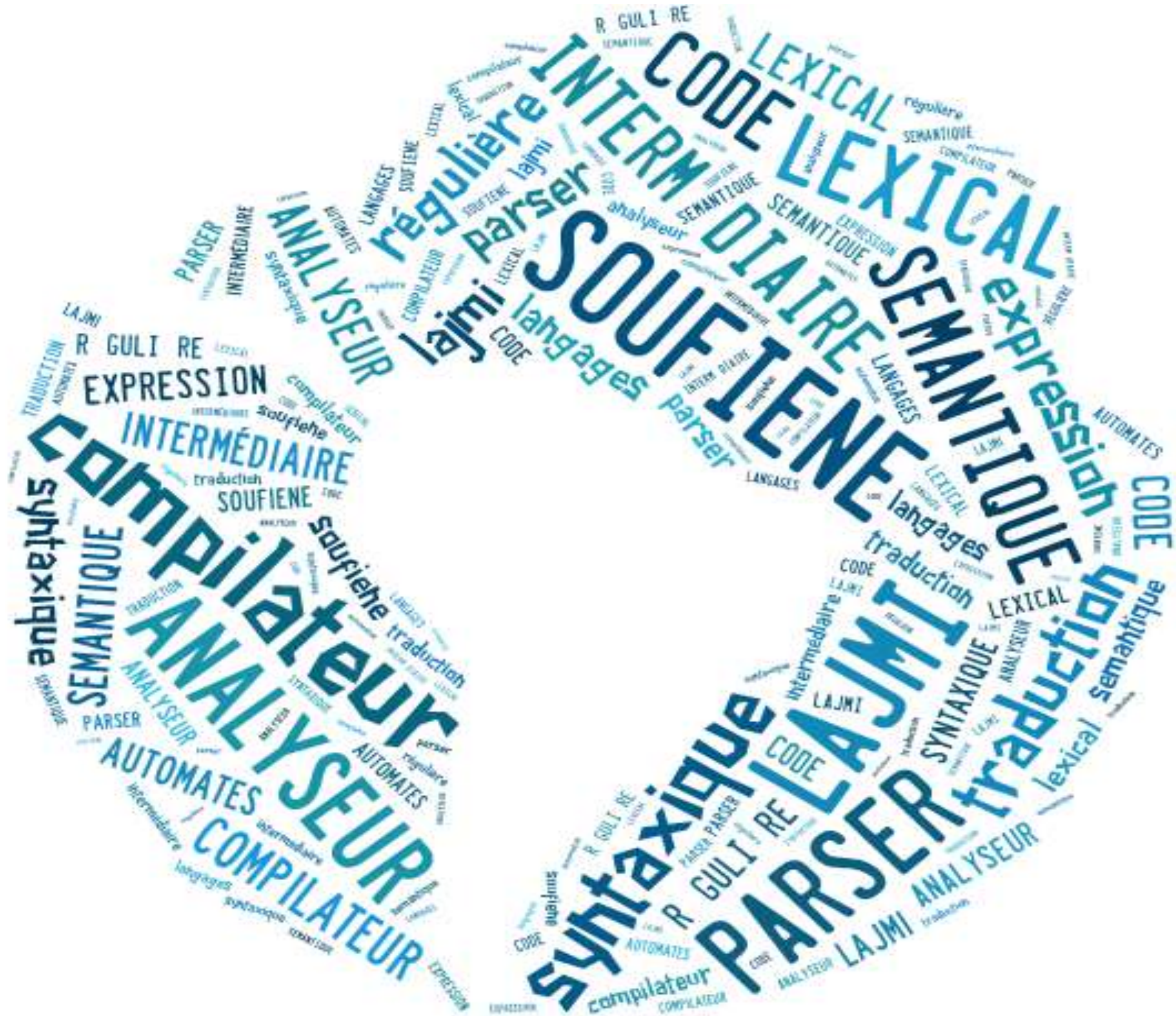
Institut International de Technologie

Cours Théorie des langages & Compilation

Analyse lexicale

Soufiene.Lajmi@gmail.com

Année universitaire 2020-2021


SOUFIENE
LEXICAL
SEMANTIQUE
compilateur
ANALYSEUR
PARSER
LAJMI
syntaxique
traduction
CODE
INTERM
DIAIRE
régulière
parser
lajmi
langages
expression
EXPRESSION
INTERMÉDIAIRE
COMPILATEUR
AUTOMATES
SYNTAXIQUE
R GULI RE
soufiene
intermédiaire
sémantique
lexical
analyseur

Sommaire

- Définition des unités lexicales
- Objectif de l'analyse lexicale
- Erreurs lexicales
- Mémorisation du texte d'entrée
- Spécification des unités lexicales

Classe d'unité lexicale

- Dans un langage de programmation:
  - Identificateur, mot clé, '(', ')', nombre, …
- Chaque classe correspond à un ensemble de mots

Le rôle de l'analyseur lexical

- L'analyseur lexical réalise également certaines tâches secondaires :
  - Élimination des caractères d'espacement : les caractères blancs (ou espaces), les tabulations et les sauts de lignes.
  - Élimination des commentaires.
  - Liaison des messages d'erreur issus du compilateur au programme source (indication de la ligne qui contient l'erreur).

Les unités lexicales

- Exemples d'unités lexicales :
  - **Mots clés (key words)** : ce sont des mots réservés au langage de programmation.
    - Exemples (langage C) : *if*, *else*, *while*, *do*, *for*, *main*, *char*, *int*, *float*, *double*, *sizeof*, …etc.
  - **Identificateurs (identifiers)** : un identificateur est un nom propre utilisé pour désigner une entité d'un programme (une variable, une fonction ou une étiquette). En C, un identificateur est une suite de caractères qui commence par une lettre ([a…z] ou [A…Z] ou « _ ») et ne contient que des lettres et/ou des chiffres [0…9]).

Les unités lexicales

- Exemples d'unités lexicales (suite) :
  - **Symboles spéciaux ou délimiteurs** : ce sont des séparateurs d'unités lexicales.
    - Exemples (langage C):
      - point virgule « ; »
      - caractère blanc ou espace « »
      - tabulation
      - saut de ligne
      - accolades « { } »
      - guillemets « " »
      - apostrophe « ' »

Les unités lexicales

- Exemples d'unités lexicales (suite) :
  - **Opérateurs** : des symboles pour des opérations de base offertes par le langage de programmation.
    - Exemples (opérateurs du langage C) : +, -, *, /, %, >, >=, ==, <=, <, !=, &, |, ^, ~, >>, <<, &&, ||, !, ?:, =, +=, -=, *=, /=, %=, &=, |=, ^=, ~=, >>=, <<=, &&=, ||=, [ ], ( ), ., ->, & (adresse), sizeof.
  - **Nombres** : suite de chiffres avec éventuellement un signe (+ ou -), un point décimal « . ».
  - **Commentaires** : chaînes de caractères délimitées entre deux symboles spéciaux. Un commentaire est un texte à ignorer.

Les unités lexicales

- Exemples d'unités lexicales (suite) :
  - Exemples de commentaires :
    - En C : **/*** Texte ***/**
    - En C++ : **//** Texte
    - En pascal : **(*** Texte ***)** ou **{** Texte **}**
    - En JavaScript : **<!--** Texte **-->**
    - En Basic : **REM** Texte
    - En visual basic : **'** Texte
    - En Matlab : **%** Texte

Les unités lexicales

- Remarque :
  - Certaines conventions de langage influent sur la complexité de l'analyse lexicale.
  - Des langages de programmation comme **Fortran** ou **Cobol** imposent que certaines constructions se trouvent à des positions fixes dans la ligne d'entrée.
  - L'alignement d'un lexème peut alors être important pour déterminer la correction d'un programme source.
  - Les langages de programmation modernes vont vers un texte d'entrée en format libre, ce qui simplifie beaucoup la tâche de l'analyseur lexical.

Attributs des unités lexicales

- L'analyseur lexical regroupe les informations sur les unités lexicales dans des **attributs** qui leur sont associés.

- En général, une unité lexicale a un seul attribut implanté comme un pointeur vers l'entrée de la table des symboles dans laquelle l'information sur l'unité lexicale est stockée.

Attributs des unités lexicales

- Remarque :
  - Pour certaines unités lexicales, aucune valeur d'attribut n'est nécessaire : le premier composant suffit pour identifier le lexème en question.
- Exemples :
  - Pour ce qui est des symboles comme >, >=, ==, <=, < et != (langage C), l'analyseur syntaxique a juste besoin de savoir que cela correspond à l'unité lexicale OP_REL (opérateur relationnel).

Attributs des unités lexicales

- Exemples (suite) :
  - Pour ce qui est des identificateurs, l'analyseur syntaxique a juste besoin de savoir que c'est l'unité lexicale **ID**.
    - Mais le générateur de code, lui, aura besoin de l'adresse de la variable correspondante à cet identificateur.
    - L'analyseur sémantique aura aussi besoin du type de la variable pour vérifier que les expressions sont sémantiquement correctes.

Erreurs lexicales

- Peu d'erreurs sont détectables au seul niveau lexical.
- Les erreurs se produisent lorsque l'analyseur est confronté à une suite de caractères qui ne correspond à aucun des modèles d'unité lexicale qu'il a à sa disposition : on dit qu'il y a une erreur lexicale.

Erreurs lexicales

- Lorsque l'analyseur est confronté à une suite de caractères qui ne correspond à aucun de ses modèles, plusieurs stratégies sont possibles :
  - **Mode panique**.
  - **Transformations du texte source**.

Erreurs lexicales

- **Transformations du texte source** :

  La correction d'erreur par transformations du texte source se fait en calculant le nombre minimum de transformations à apporter au mot qui pose problème pour en obtenir un qui ne pose plus de problèmes.

Erreurs lexicales

- On utilise des techniques de calcul de distance minimale entre des mots.

Erreurs lexicales

- Le gros problème est alors de savoir gérer la nouvelle erreur.
- La stratégie la plus simple est le mode panique :
  - En fait, en général, cette technique se contente de refiler le problème à l'analyseur syntaxique.
  - Mais c'est efficace puisque c'est lui, la plupart du temps, le plus apte à le résoudre.

Mémorisation du texte d'entrée

- Utilisation d'un tampon divisé en deux moitiés de N caractères
- Les tampons sont rechargés tour à tour
- S'il reste moins que N caractères en entrée, un caractère fdf est placé dans le tampon après les caractères d'entrée
- On gère deux pointeurs vers le tampon d'entrée.
- La chaîne entre les deux pointeurs constitue le lexème courant

Spécification des unités lexicales

- Alphabet
- Mot
- Langage
- Langages réguliers et expressions régulières
- Définitions régulières (Pour simplifier les notations)
- Extensions des expressions régulières

Langage régulier

Soit un alphabet A (fini)

- Cas de base:
  - Ø est un langage régulier
  - {ε} est un langage régulier
  - $\forall a \in A$, {a} est un langage régulier

Langage régulier

Soient L et M des langages réguliers

- Inductions:
  - L.M = { l.m | l $\in$ L et m $\in$ M} est régulier
  - L$\cup$M est régulier
  - L* = {ε} $\cup$ {www…w | w$\in$L} est régulier

Expression régulière

- Soit A un alphabet.
  - Une **expression régulière** (ou **expression rationnelle**) sur A est un objet (formel) défini récursivement par :
    - $\varnothing$ est une expression régulière qui dénote $\varnothing$.
    - ε est une expression régulière qui dénote $\{\varepsilon\}$.
    - $\forall\ a \in A$, a est une expression régulière qui dénote {a}.
    - Si r et s sont deux expressions régulières dénotant respectivement les langages R et S, alors
      - rs, r + s et r* sont des expressions régulières qui dénotent respectivement RS, R+S et R^*.

Exemples expressions régulières

- ((a|b)a)*

  {ε , aa, ba, aaaa, baaa, aaba, baba, aaaaaa, …}

- (0|1)*0

Définition régulière

- Si A est un alphabet, une définition régulière est une suite de définitions de la forme :

  d1→r1 ;

  d2→r2 ;

  … ;

  dn→rn

où di est un nom et chaque ri est une expression régulière sur les symboles de A ∪{d1,d2,…,di-1}

Exemple

- Les nombres sans signe sont des chaînes comme 5280, 39.37, 6.336E4 ou 1.894E-4

  chiffre → 0|1|2|3|4|5|6|7|8|9

  chiffres → chiffre chiffre*

  decimal-opt → . chiffre | ε

  exposant-opt → (E(+|-|e)chiffres) | ε

  nombre → chiffres decimal-opt exposant-opt

Autres notations d'expressions régulières

- r+ ≡ rr*
- r? ≡ r| ε
- [abc] ≡ (a|b|c)
- [a-g] ≡ a|b|…|f|g
- [^s] ≡ complément de l'ensemble
- ^ ≡ début de ligne
- \n ≡ la fin de ligne
- "ab" ≡ la chaîne ab
- . ≡ N'importe quelle caractère sauf '\n'

- Exemples
  - **[a-zA-Z][a-zA-Z0-9]*** : chaîne alphanumérique commençant par une lettre.
  - **[0-9]+** : un nombre entier positif.

Automate fini

- M = (Q, A, t, q0, F) où
- Q est un ensemble fini des états;
- A est l'alphabet des symboles à l'entrée;
- t est la relation de transition: Q x A x Q;
- q0 est l'état initial;
- F $\subseteq$ Q ensemble des états accepteurs

Propriété

- Si R est un langage régulier, alors il existe un automate fini M tel que L(M) = R.

Conclusion

Questions ??