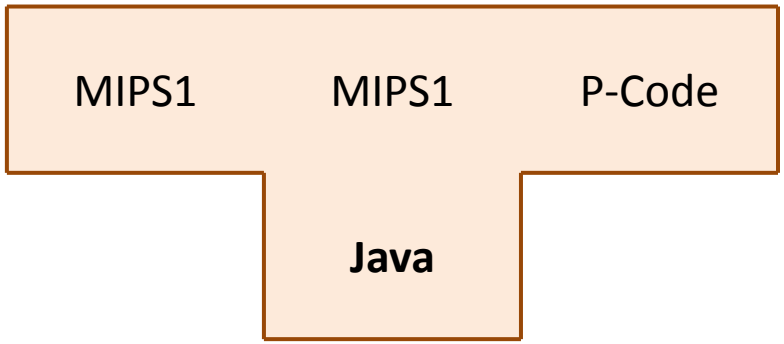
Chap. 3 : Analyseur sémantique

Introduction

- Le chapitre précédent se focalise sur l'analyse syntaxique.
- Après la vérification de la syntaxe, on doit vérifier des règles.
- Ces règles ne sont pas exprimés par la syntaxe.
- Les **identificateurs** sont les objets sémantiques du programme.
- Rappel du T-schéma du compilateur.



Syntaxe du langage MIPS1

- PROGRAM ::= **program** ID **;** BLOCK **.**
- BLOCK ::= CONSTS VARS INSTS
- CONSTS ::= **const** ID **=** NUM **;** { ID **=** NUM **;** } | ε
- VARS ::= **var** ID { **,** ID } **;** | ε
- INSTS ::= **begin** INST { **;** INST } **end**
- INST ::= INSTS | AFFEC | SI | TANTQUE | ECRIRE | LIRE | ε
- AFFEC ::= ID **:=** EXPR
- SI ::= **if** COND **then** INST
- TANTQUE ::= **while** COND **do** INST
- ECRIRE ::= **write (** EXPR { **,** EXPR } **)**
- LIRE ::= **read (** ID { **,** ID } **)**
- COND ::= EXPR RELOP EXPR
- RELOP ::= **=** | **<>** | **<** | **>** | **<=** | **>=**
- EXPR ::= TERM { ADDOP TERM }
- ADDOP ::= **+** | **-**
- TERM ::= FACT { MULOP FACT }
- MULOP ::= ***** | **/**
- FACT ::= ID | NUM | **(** EXPR **)**

Règles sémantiques

1. **Règle 1**: Toutes les déclarations dans CONSTS et VARS
2. **Règle 2**: PAS DE DOUBLE DECLARATIONS
3. **Règle 3**: Après BEGIN, tous les symboles doivent être déjà déclarés
4. **Règle 4**: Une constante ne peut changer de valeur dans le programme
5. **Règle 5**: Le ID du programme ne peut être utilisé dans le programme

Exemples d'erreurs sémantiques

Exemple 1:

```
program test;
const tata=12;
var x;
begin
  titi:=tata;
end.
```

ERR: identificateur titi non déclaré

Exemple 2:

```
program test;
const tata =12;
var x, tata ;
begin
  x:=tata;
end.
```

ERR: double déclaration

Exemple 3:

```
program test;
const tata=12;
var x;
begin
  tata :=15;
end.
```

ERR: constante ne peut changer de valeur

Exemple 4:

```
program test;
const tata=12;
var x;
begin
  x :=test ;
end.
```

ERR: nom de programme non autorisé

Exemple 5:

```
program test;
const tata=12;
var x;
begin
  read(tata );
end.
```

ERR: constante ne peut changer de valeur

Table des symboles

- Un exemple de contrôle sémantique consiste à vérifier que les identificateurs utilisés sont bien déclarés.
- Pour cela, il est nécessaire de mémoriser les identificateurs pour tester leurs déclarations.
- Cette mémorisation se fait dans la table des symboles.
- Dans cette table, on associe à chaque identificateur les informations de niveau sémantique qui lui sont associés :
  - Sa forme textuelle (nom).
  - Sa classe (s'il désigne un programme, une constante ou une variable).

Déclarations

- Définition du type de données CLASSE_IDF

```
TYPE  CLASSE_IDF= (PROGRAMME, CONSTANTE, VARIABLE)
```

- Définition du type SYMBOLES

```
STRUCTURE SYMBOLES {
        TOKEN : TOKENS  (*Contient le token du symbole *)
        NOM     : CHAINE[8] (*Contient la forme textuelle du
                                symbole*)
        CLASSE : CLASSE_IDF (*Contient la classe de l'idf*)
    }
```

Suite des déclarations

- Déclaration de la table des symboles.

```
TABLE_SYMB : SYMBOLES[500]
```

- Déclaration d'une variable qui mémorise la place d'un symbole dans la la table des symboles.

```
PLACE_SYMB : entier
```

Manipulation de la table des symboles

- La table des symboles est manipulée par deux procédures :
  - ENTRER_SYMB : ajoute le symbole courant dans la table des symboles avec la classe passée en paramètre.

```
procedure ENTRER_SYMB (c : CLASSE_IDF)
debut
 (*insére le symbole courant à la fin dans la table des symboles,
  la classe de l'dentificateur est donnée en paramètre*)
fin
```

  - CHERCHER_SYMB : cherche le symbole courant dans la table des symboles.

```
procedure CHERCHER_SYMB ()
debut
 (*cherche le symbole courant à la fin dans la table des
symboles, et affecte sa position à PLACE_SYMB, s'il n'existe
  pas alors affecter -1 à PLACE_SYMB*)
fin
```

Entrée dans la table des symboles

- La procédure ENTRER_SYMB est utilisée lors de la déclaration des symboles par l'intermédiaire de la procédure TESTE_INSERE qui vérifie que le prochain token est celui attendu et qui met à jour la table des symboles.

```
procedure TESTE_INSERE(t:TOKENS,c : CLASSE_IDF, e:ERR_CODES)
debut
  si (token=t) alors
  debut
    ENTRER_SYMB(c)
    SYMB_SUIV()
  fin
fin
```

  - Cette procédure TESTE_INSERE est appelée lors de la déclaration des symboles : CONSTS, VARS et PROGRAM.

Consultation de la table des symboles

- De manière symétrique, une procédure TESTE_CHERCHE est une évolution de la procédure TESTE_ACCEPT qui de plus recherche le symbole courant dans la table des symboles par un appel à CHERCHER_SYMB, l'indice du symbole recherché dans la table est retourné par la variable PLACE_SYMB.

```
procedure TESTE_CHERCHE(t:TOKENS, e:ERR_CODES)
debut
  si (token=t) alors
  debut
    CHERCHE_SYMB()
    SYMB_SUIV()
  fin
fin
```

  - Cette procédure est appelée lors de l'utilisation de l'identificateur : FACT, AFFEC et LIRE.