



IC-5701

Compiladores e Intérpretes

Profesor:

Ing. Allan Rodríguez Dávila, MGP

Tablas de símbolos

Validación de tipos

Diseño de analizadores

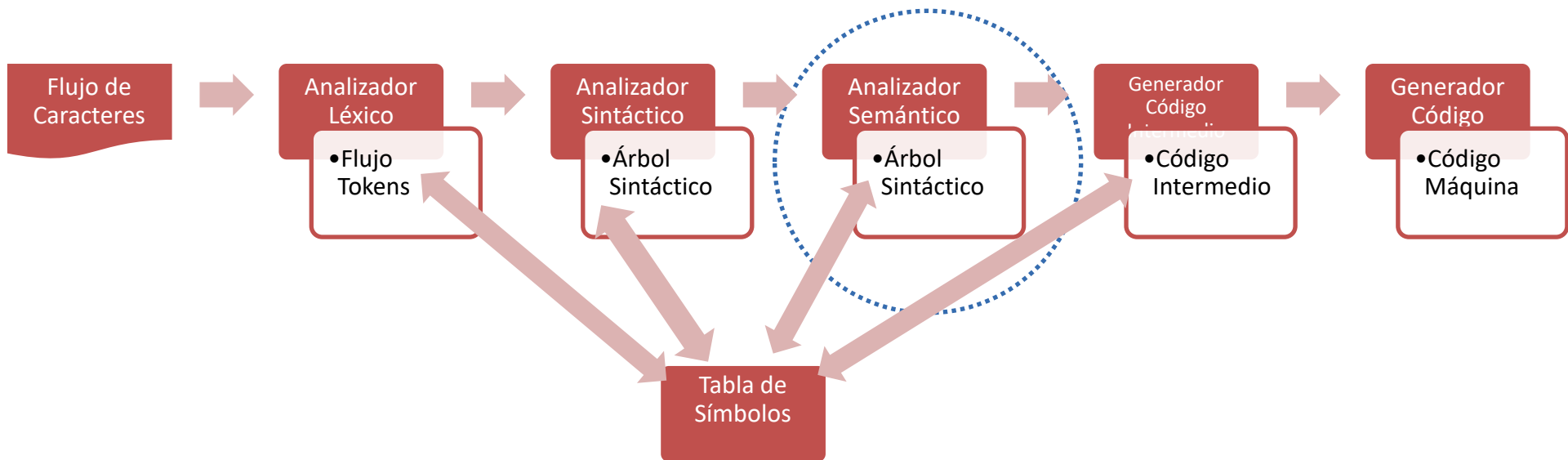
Tratamiento de errores contextuales

Análisis Semántico

Análisis Semántico

- Como las gramáticas EBNF no pueden describir todos los elementos sintácticos del lenguaje, ¿se necesita algún análisis adicional!
- Con el árbol sintáctico y la tabla de símbolos se comprueba la consistencia semántica del programa fuente con la definición del lenguaje

Compilación



Análisis Semántico

- Se recopila información sobre el tipo y se guarda (árbol o tabla).
 - Útil en la generación de código intermedio
- La parte más importante es la **comprobación de tipos**.
 - Verifica que los operandos coincidan con el operador.

```
String identificador = "Hoy es";  
return identificador + 22;
```

Principales funciones

- Identificar los tipos
- Completar la tabla de símbolos
- Comprobaciones:
 - De Tipos
 - De Flujos
 - De Unicidad
 - De Emparejamiento (operaciones)

Principales Acciones

- Sentencia de declaración
- Sentencias “ejecutables”
- Funciones y procedimientos
- Identificación de variables
- Etiquetas
- Constantes
- Conversiones de tipo
- Sobrecarga de operadores

Tratamiento de errores contextuales

Errores Contextuales

- Los errores semánticos incluyen los conflictos de tipos entre los operadores y los operandos.
- La detección en tiempo de compilación es una tarea compleja.

Manejo de Errores

- Reportar la presencia con claridad y precisión.
- Recuperarse de cada error.
- Agregar una sobrecarga mínima.
- Estrategias:
 - Modo pánico
 - Nivel de frase
 - Producción de errores
 - Corrección global

Validación de tipos

Validación de Tipos

- Un **sistema de tipos** es usado para el chequeo de tipos
- Un sistema de tipos incorpora
 - **Construcciones estáticas** del lenguaje
 - **Noción** de tipos
 - **Reglas** para asignar tipos a construcciones del lenguaje

Expresiones de Tipos

- Un **tipo compuesto** es denotado por una expresión de tipo
- Una **expresión de tipo** es
 - Un tipo básico
 - La aplicación de un constructor de tipo a otras expresiones de tipo

Expresiones de Tipos: Básicos

- Tipos atómicos definidos por el lenguaje
- Ejemplos:
 - Enteros - flotantes
 - Booleanos - caracteres
- `type_error`
 - Tipo especial que produce un error
- `void`
 - Tipo básico que denota “la ausencia de un valor”

Expresiones de Tipos: Nombres

- Ya que las expresiones de tipos pueden ser nombradas, un nombre de tipo es una expresión de tipo.
 - Literales
 - Aritméticas
 - Lógicas
 - Algebraicas
 - Condicionales

Expresiones de Tipos: Productos

- Si $T1$ y $T2$ son expresiones de tipo, $T1 \times T2$ es también una expresión de tipo
 - Producto cartesiano
 - Tuplas
 - Registros
- Asociatividad por la izquierda

Expresiones de Tipos: Arreglos

- Si T es una expresión de tipo, un $\text{array}(T, I)$ es también una expresión de tipo
 - I es una constante entera que denota el número de elementos de tipo T
 - Ejemplo:
`int foo[128];`
`array(integer, 128)`

Expresiones de Tipos: Funciones

- Matemáticamente una función mapea
 - Elementos de un conjunto (el **dominio**)
 - A elementos de otro conjunto (el **codominio**)

- Ejemplo

```
int foobar(int a, boolean b, int c)
```

```
integer * boolean * integer → integer
```

Expresiones de Tipos: Otras

- Records

- Estructuras y clases

- Ejemplo

```
class { int i; int j;}  
integer * integer
```

- Lenguajes Funcionales

- Funciones que toman funciones y retornan funciones

- Ejemplo

```
(integer → integer) * integer → (integer → integer)
```

Validación de Tipos

- Un compilador debe realizar una serie de **chequeos estáticos**
 - Consistencia
 - Equivalencia y compatibilidad
 - Conversión explícita
 - Inferencia de tipos
 - Sobrecarga de funciones y operadores
 - Funciones polimórficas

Principales Acciones

- Sentencias de declaración
 - Completar la sección de tipos de la Tabla de Símbolos
- Sentencias Ejecutables
 - Realizar comprobaciones de tipos entre los operandos implicados
- Funciones y procedimientos
 - Comprobar el número, orden y tipo de los parámetros actuales en cada llamada a una función
- Constantes
 - Comprobar que no se utilicen en la parte izquierda de una asignación

Principales Acciones

- Identificación de variables
 - Comprobar si un identificador ha sido declarado antes de utilizarlo
- Etiquetas
 - Comprobar si hay etiquetas repetidas y validación
- Conversiones y equivalencias de tipo
 - Verificación
- Sobrecarga de operadores y funciones
 - Detectar y solventar

Portafolio #3

```
float miFunc1(int x){  
    float miFloat=45.5*x;  
    char miChar = 21.5*x;  
    read(miChar);  
    return x;  
}  
  
int main(){  
    int m = 45 > "hola";  
    int w = 10+m;  
    float n = miFunc1("hola")+miFloat*56/w;  
    if(n){ n = miFunc1(1.1,2.2);}   
    int arr[10];  
    arr[w] = 10;  
}
```


Portafolio #3

```
int main(){  
    string m = "hola";  
    forRange(10;20;3){  
        int y = i^10 < 100;  
        int w = ++m;  
    }  
    print(i);  
    return 0;  
}
```

Tablas de símbolos

Resumen

- Subir a Tec Digital, Participación, Actividad 3:
 - Lectura y Resumen de Tabla de símbolos

Diseño de analizadores de contexto

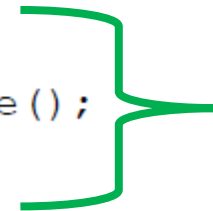
Opciones CUP

```
package grammar;

import java.util.*;
import java.io.*;
import java_cup.runtime.*;

action code
{
    Hashtable table = new Hashtable();
};
```

C P
L R
A I
S V
E A
D
A



Opciones CUP

Expr::=

```
Expr:expr PLUS Term:term  
{:  
RESULT = new Integer( expr.intValue() + term.intValue() );  
:}
```

|

```
Expr:expr MINUS Term:term  
{:  
RESULT = new Integer( expr.intValue() - term.intValue() );  
:}
```

|

```
MINUS Term:term  
{:  
RESULT = new Integer( - term.intValue() );  
:}
```

|

```
Term:term  
{:  
RESULT = term;  
:}
```

;

Código
asociado

Retornar Valor

Opciones CUP


```
Stmt ::=  
    IDENT:ident ASSIGN Expr:expr  
    {:  
    table.put( ident, expr );  
    :}  
|  
    Expr:expr  
    {:  
    System.out.println( expr.intValue() );  
    :}  
;
```



Almacenar
valores

Opciones CUP

```
|  
IDENT:ident  
{:  
Integer value = ( Integer ) table.get( ident );  
if ( value == null ) {  
    parser.report_error( "Undeclared Identifier " + ident,  
        new Symbol( sym.IDENT, identleft, identright, ident ) );  
    value = new Integer( 0 );  
}  
RESULT = value;  
:}  
;
```



Error Semántico

- Programming Language Processors in Java: compilers and interpreters. Watt, David, Brown, Deryck. Pearson Education. 2000
- Compilers: principles, techniques and tools (2da. ed.). Aho, Alfred. Pearson Education. 2007

TEC | Tecnológico
de Costa Rica