

Análise semântica

Função, interação com o compilador
Tabela de símbolos
Análise semântica

Prof. Thiago A. S. Pardo
taspardo@icmc.usp.br

1

Tratamento semântico

- **Verificação do uso adequado** dos elementos do programa
 - **Declaração de identificadores**
 - Erro: identificador não declarado ou declarado duas vezes
 - **Compatibilidade de tipos** em comandos
 - Checagem de tipos
 - **Concordância entre parâmetros** formais e atuais, em termos de número, ordem e tipo

2

Tratamento semântico

■ Declaração de identificadores

- Verificado durante a construção da tabela de símbolos

■ Compatibilidade de tipos

- Dependente do contexto
 - Atribuição: inteiro:=inteiro, real:=inteiro, string:=cadeia de caracteres
 - Normalmente, tem-se erro quando inteiro:=real
 - Conversão implícita (coerção) ou explícita dos tipos
 - Comandos de repetição: while booleano do..., if booleano then...
 - Expressões e tipos esperados pelos operadores: inteiro+inteiro, real*real, inteiro+real, inteiro/inteiro, booleano and booleano
 - Erro: inteiro+booleano
 - Arrays: vetor[integer]

3

Tratamento semântico

■ Concordância entre parâmetros formais e atuais, em termos de número, ordem e tipo

- Por exemplo, se declarado: procedure p(var x: integer; var y: real)
 - Erros
 - procedure p(x:integer, y:integer)
 - procedure p(y:real, x:integer)
 - procedure p(x:integer)
 - Tratamento de escopo
 - Erro: variável local a um procedimento utilizada no programa principal

4

Tratamento semântico

■ Pontos importantes

- Tipo: amarração estática vs. dinâmica
 - Estática: declaração explícita do tipo, boa para compilação
 - Dinâmica: tipo inferido na execução, boa para interpretação
- Linguagens fortemente tipadas: uma linguagem é fortemente tipada se em tempo de compilação pode-se garantir que os programas aceitos irão executar sem erros de tipos
 - Pergunta: Pascal é fortemente tipada?

5

Tratamento semântico

- Pascal não é fortemente tipada
 - Intervalos de valores de variáveis não são testados estaticamente (p.ex., em tipos enumerados)
 - O elemento discriminante de registros variantes em Pascal não é testado estaticamente
 - Não há regras de compatibilidade de tipos rigorosas na especificação do Pascal

6

Tratamento semântico

■ Tipos

- ❑ Básicos (ou primitivos): booleano, char, inteiro, real, enumerado
- ❑ Estruturados: array, record, ponteiro

■ Regras de compatibilidade de tipos

- ❑ As regras de compatibilidade de tipos são geralmente da forma:
se duas expressões são equivalentes,
então retorne um certo tipo
senão erro
- ❑ Precisa-se ter uma definição precisa de quando duas expressões são equivalentes
- ❑ 2 possíveis noções de compatibilidade nas linguagens
 - Equivalência de nomes
 - Equivalência estrutural

7

Tratamento semântico

- **Equivalência de nomes:** duas variáveis possuem tipos compatíveis se
 - ❑ Têm o mesmo nome do tipo, definido pelo usuário ou primitivo
 - ❑ Ou aparecem na mesma declaração
- **Equivalência estrutural:** duas variáveis tem tipos compatíveis se possuem a mesma estrutura
 - ❑ Os tipos definidos pelo usuário são usados só como abreviatura da estrutura que representam e não introduzem qualquer característica semântica nova
 - ❑ Para se checar a equivalência, os nomes dos tipos definidos pelo usuário são substituídos pelas suas definições repetidamente até não sobraem mais tipos definidos pelo usuário

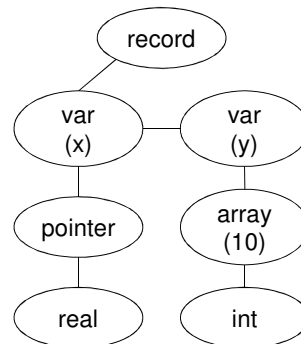
8

Tratamento semântico

- Exemplo: considere a declaração abaixo

```
record
  x: pointer to real;
  y: array[10] of int
end
```

A estrutura desse tipo poderia ser representada como:



9

Tratamento semântico

- Exemplo: para as declarações abaixo

```
type t = array[1..20] of integer;
var a, b: array[1..20] of integer;
c: array[1..20] of integer;
d: t;
e, f: record
  a: integer;
  b: t
end
```

- Pode-se observar que:
 - (a e b), (e e f) e (d, e.b e f.b) têm equivalência de nomes
 - a, b, c, d, e.b e f.b têm tipos compatíveis estruturalmente

10

Tratamento semântico

- A maioria das linguagens implementa as duas estratégias de compatibilidade de tipos
- **Sistema de tipos**: coleção de regras que atuam sobre expressões de tipos (isto é, os tipos básicos da linguagem ou os estruturados, definidos ou não pelo usuário)
- Um **verificador de tipos** implementa um **sistema de tipos**, utilizando informações sobre a sintaxe da linguagem, a noção de tipos e as regras de compatibilidade de tipos

11

Tratamento semântico

- Verificador de tipos
 - Especificado na **gramática de atributos** e implementado como tal
 - Compilação de mais de uma passagem, possivelmente
 - Comandado pela **análise sintática**
 - Possibilidade de compilação de uma única passagem

12

Tratamento semântico

- Exemplo: verificação de tipos na gramática de atributos

```
<exp>1 ::= <exp>2 div id
    se busca(id)=falso
        então ERRO("variável não declarada")
    senão se exp2.tipo<>busca(id.tipo) ou exp2.tipo<>inteiro ou
        busca(id.tipo)<>inteiro
        então ERRO("tipos inválidos para a operação")
    se não ocorreu erro então
        exp1.tipo=inteiro
        exp1.val= exp2.val / id.val
```

13

Tratamento semântico

- Exemplo: verificação de tipos em uma regra sintática de atribuição de tipos iguais

```
procedimento atribuição(Seg)
Início
    se (simbolo=id)
        então obtem_simbolo(cadeia,simbolo)
            se busca(cadeia,simbolo,cat="var")=FALSE
                então ERRO("variável não declarada")
                senão tipo1:=recupera_tipo(cadeia,simbolo,cat="var");
            senão ERRO(Seg+{simb_atrib});
    se (simbolo=simb_atrib)
        então obtem_simbolo(cadeia,simbolo)
        senão ERRO(Seg+{id});
    expressao(tipo2);
    se tipo1<>tipo2 então ERRO("tipos incompatíveis na atribuição");
    se (simbolo=simb_ponto-virgula)
        então obtem_simbolo(cadeia,simbolo)
        senão ERRO(Seg+P(comandos));
fim
```

Na LALG

■ Principais erros semânticos

- ❑ Variável ou procedimento não declarado
- ❑ Variável ou procedimento declarado mais de uma vez
- ❑ Incompatibilidade de parâmetros formais e reais: número, ordem e tipo
- ❑ Uso de variáveis de escopo inadequado
- ❑ Atribuição de um real a um inteiro
- ❑ Divisão que não é entre números inteiros
- ❑ Read e write com variáveis de tipo diferentes

15

Exercício

- **Em duplas**, especifique a gramática de atributos completa e faça o procedimento sintático correspondente para a gramática abaixo

```
<cmd-IO> ::= read ( <lista_var> ) |  
           write ( <lista_var> )  
<lista_var> ::= ident <mais_var>  
<mais_var> ::= , <lista_var> | λ
```

16