Interpretador de TIGER para uma Representação tipo Assembler

Projeto Orientado em Computação 2

Aluno: Gustavo Garcia Guerra

Orientadora: Mariza Andrade S. Bigonha

Esquema da Apresentação

- Introdução a linguagem TIGER
- Introdução à MVJ
- Etapas da construção do interpretador
- Exemplos

Objetivos do Projeto Proposto

A Linguagem TIGER

- Declarações
- Variáveis e Expressões
- Biblioteca de Funções

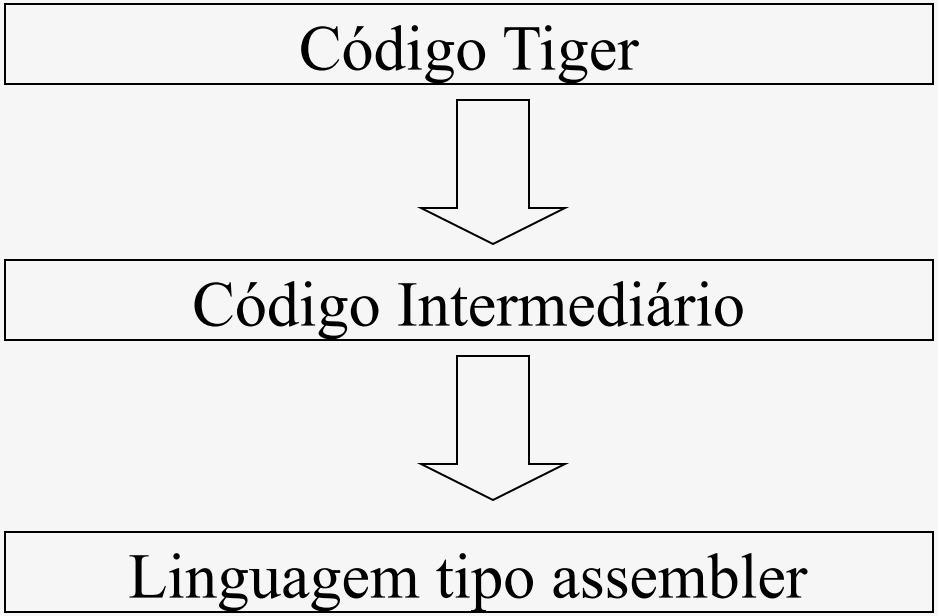
Máquina Virtual JAVA

- Arquivo *class*
- Possui quatro registradores
  - PC
  - VARS
  - OPTOP
  - FRAME

Representação tipo Assembler

- Subconjunto de instruções
- Liberdade de escolha entre plataformas:
  - x86
  - Java Assembler
- Permitir a utilização de ferramentas:
  - GNU Assembler
  - Jasmin

Etapas da Construção do Interpretador



Etapas de Construção do Interpretador

- Código intermediário em forma de árvore abstrata
- Definir um subconjunto de instruções
- Realizar alocação de registros
- Gerar a representação intermediária tipo assembler

Exemplo 1

- Arquivo de Entrada em Tiger:

```
let
  var a := 3
  var b := 4
  var c := 0
in
  c = a + b
end
```

Exemplo

```
// LET                          // LetExp:1.1
// Var Declaration of a         // VarDec:2.5
num 3                           // IntExp:2.10
// Var Declaration of b         // VarDec:3.5
num 4                           // IntExp:3.10
// Var Declaration of c         // VarDec:4.5
num 0                           // IntExp:4.10
// IN                           // LetExp:1.1
// Var Reference: c Offs=2      // SimpleVar:
    6.1
num 2                           // SimpleVar:
    6.1
ld                                      //
    SimpleVar:6.1
// Var Reference: a Offs=0      // simpleVar:6.5
num 0                           // SimpleVar:6.5
ld                              // SimpleVar:6.5
// Var Reference: b Offs=1      // simpleVar:6.9
num 1                           // SimpleVar:6.9
ld                              // SimpleVar:6.9
add                             // OpExp:6.5
eq                              // OpExp:6.1
sts 2                           // LetExp:1.1
popn  2                         // LetExp:1.1
// END                          // LetExp:1.1
popn  0
halt
```

Exemplo 2

- Arquivo de entrada em TIGER:

```
/* valid for and let */

let
in
        for i:=0 to 100 do ()
end
```

Exemplo

```
// LET              // LetExp:3.1
// IN               // LetExp:3.1
// Var Declaration of i
num 0               // IntExp:5.16
num 100             // IntExp:5.21
jmp ForLoopEnterLabel_1
                    // ForExp:5.9
ForLoopLabel_0:     // ForExp:5.9
popn  0             // ForExp:5.9
lds 1               // ForExp:5.9
add 1               // ForExp:5.9
sts 1               // ForExp:5.9
ForLoopEnterLabel_1: // ForExp:5.9
lds 1               // ForExp:5.9
lds 1               // ForExp:5.9
jle ForLoopLabel_0: // ForExp:5.9
popn  2             // ForExp:5.9
ForLoopExitLabel_2:// ForExp:5.9
popn  0             // LetExp:3.1
// END              // LetExp:3.1
popn  -1
halt 0
```

Projetos Futuros

- Realizar otimizações no código gerado
- Projetar um garbage collector
- Gerar o código assembler em diferentes padrões:
  - x86
  - Java assembly
  - Jasmin

Perguntas?