Geração de Código

Etapas da Geração de Código

Geração de Código Intermediário

Otimização de Código

Geração de código Objeto

Geração de Código

Geração de Código Intermediário

- facilita o transporte entre máquinas;
- possibilita a utilização de otimizadores de código independentes;
- não exige o estudo e aprendizado de uma arquitetura específica.

Representações mais comuns:

- árvore sintática abstrata (ASA);
- máquina virtual de pilha;

Geração de Código

Otimização de Código

- **Independente de arquitetura**
  - Eliminação de rótulos NOP
  - Otimização de expressões matemáticas
  - Agrupamento de código idêntico
- **Dependente de arquitetura**
  - Escolha de instruções e registradores mais eficientes;

Geração de Código

Geração de Código Objeto

- **Converte a linguagem intermediária otimizada em código objeto com um conjunto de instruções de uma arquitetura específica;**
- Gerência de Memória;
- Montagem e Link-Edição

Geração de Código Intermediário

Instruções do Assembly do BIP

- Transferência (acesso à memória): STO, LD, LDI
- Vetores: $indr, LDV e STOV
- Aritmética: ADD, ADDI, SUB e SUBI
- Controle: HLT, CALL, RETURN
- Desvio: BEQ, BNE, BGT, BGE, BLT, BLE e JMP
- Lógica: AND, ANDI, OR, ORI, NOT, SLL, SRL
- Entrada: $in_port
- Saída de dados: $out_port
- Suporte a Rótulos

Geração de Código Intermediário

Portugol

```
procedimento principal()
declaracoes
  inteiro val, maior, cont
inicio
  cont <- 1
  maior <- 0
  enquanto (cont <= 5) faca
    leia (val)
    se (val > maior) entao
      maior <- val
    fimse
    cont <- cont + 1
  fimenquanto
  escreva (maior)
fim
```

Assembly do BIP

```
.data
  val : 0
  maior : 0
  cont : 0
.text
_PRINCIPAL:
  LDI    1
  STO    cont
  LDI    0
  STO    maior
INI_ENQ1:
  LD     cont
  STO    1000
  LDI    5
  STO    1001
  LD     1000
  SUB    1001
  BGT    FIMFACA1
  LD     $in_port
  STO    val
  LD     val
  STO    1000
  LD     maior
  STO    1001
  LD     1000
  SUB    1001
  BLE    FIMSE1
  LD     val
  STO    maior
FIMSE1:
  LD     cont
  ADDI   1
  STO    cont
  JMP    INI_ENQ1
FIMFACA1:
  LD     maior
  STO    $out_port
  HLT    0
```

Geração de Código Intermediário

Gramática Resumida do Portugol

```
<prog> ::= <lista_dec><lista_cmd>
<lista_dec> ::= <lista_dec> <dec> | <dec>
<lista_cmd> ::= <lista_cmd> <cmd> | <cmd>
<lista_id> ::= <id> ‘,’<lista_id>| <id>
<id> ::= ID | ID ‘<-’ NUM_INT
<dec> ::= <tipo> <lista_id>
<tipo> ::= INT
<cmd> ::= ID  ‘<-’  <exp2>
    | ID  ‘[’ ID ‘]’ ‘<-’ <exp2>
    | ID  ‘[’ NUM_INT ‘]’ ‘<-’ <exp2>
    | se <exp> entao <lista_cmd> <r_senao> fimse
    | enquanto <exp> faca <lista_cmd> fim_enquanto
    | para ID ‘<-’ <exp2> ATE <exp2> passo NUM <lista_cmd> fimpara
    | leia ‘(’ ID ‘)’
    | leia ‘(’ ID ‘[’ <exp2> ‘]’ ‘)’
    | escreva ‘(’ <exp3> ‘)’
<r_senao> ::= senao <lista_cmd> | î
<exp> ::= <exp> AND <exp1>  | <exp> OR <exp1>   | NOT <exp>  | <exp1>
<exp1> ::= <exp1> ‘>’ <exp2> | <exp1> ‘<’ <exp2> | <exp1> ‘>=’ <exp2>
          | <exp1> ‘<=’ <exp2> | <exp1> ‘!=’ <exp2>| <exp1> ‘==’ <exp2>
          | <exp2>
<exp2> ::= <exp2> ‘+’ <exp3>  | <exp2> ‘-’  <exp3> | <exp3>
<exp3> ::= ID  | NUM_INT | ID ‘[’ ID ‘]’ | ID ‘[’ NUM_INT ‘]’
```

Geração de Código Intermediário
Declarações de Variáveis

Portugol	BIP	Declarações
inteiro val, maior inteiro cont=1	.data val: 0 maior: 0 cont: 1	**<dec> ::= <tipo> <lista_id>** **<tipo> ::= INT** **<lista_id> ::= <id>#3 ‘,’<lista_id>|<id>#3** **<id> ::= ID #1| ID #1 atrib NUM_INT #2** **#1 nome = token.getlexema()** **#2 valor = token.getlexema()** **#3 gera_cod (nome, valor);**

Alternativa

Pode ser gerada a seção .data varrendo a tabela de símbolos.

GERA_COD

Procedimento que escreve a saída em arquivo texto ou estrutura que guarda o código.

Geração de Código Intermediário
Atribuições e Expressões Aritméticas

Portugol
cont ← 1
maior ← 0
cont ← cont + 1

M. Pilha	
LDI	1
STO	cont
LDI	0
STO	maior
LD	cont
ADDI	1
STO	cont

Expressões

```
<exp> ::= ...
<exp2> ::= <exp2> '+'  #4 <exp3>
         | <exp2> '-'  #4 <exp3>
         | <exp3>
<exp3> ::= ID #5
         | NUM_INT   #6
```

```
#4 flagOp = true;
    oper = token.getlexema();
#5 if (!flagOp) {
      gera_cod ("LD ", token.getlexema());
   } else {
      if (oper == '+')
            gera_cod ("ADD", token.getlexema());
      if (oper == '-')
            gera_cod ("SUB", token.getlexema());
      flagOp = false;
   }
#6 if (!flagOp) {
      gera_cod ("LDI", token.getlexema());
   } else {
      if (oper == '+')
            gera_cod ("ADDI", token.getlexema());
      if (oper == '-')
            gera_cod ("SUBI", token.getlexema());
      flagOp = false;
   }
```

Atribuição

```
<cmd>::= ID #21 atrib  <exp> #22
```

```
#21 nome_id_atrib = token.getlexema();
#22 gera_cod ("STO", nome_id_atrib);
```

Geração de Código Intermediário
Atribuições e Expressões Aritméticas com suporte a vetores

Portugol

```
cont ← vet [0]

maior ← vet [cont]

maior <- cont - vet [0]

vet[1] ←  cont
```

M. Pilha

```
LDI     0
STO     $indr
LDV     vet
STO     cont
LD      cont
STO     $indr
LDV     vet
STO     maior
LD      cont
STO     temp1
LD      0
STO     $indr
LDV     vet
STO     temp2
LD      temp1
SUB     temp2
STO     maior
LDI     1
STO     temp1
LD      cont
STO     temp2
LD      temp1
STO     $indr
LD      temp2
STOV    vet
```

Vetor em expressão

Se é o **primeiro** operando
- Gera código da expressão do índice
- Gera STO no $indr
- Gera LDV vet

Senao // *demais operandos*
- Gera STO temp1
- Gera código da expressão do índice
- Gera STO no $indr
- Gera LDV vet
- Gera STO temp2
- Gera LD temp1
- Gera ADD ou SUB temp2

Vetor no lado esquerdo (recebe atribuição)

Gera código da expressão do índice
Gera STO no temp1
Gera código do lado direito (exp)
Gera STO em temp2
Gera LD temp1
Gera STO no $indr
Gera LD temp2
Gera STOV no vet

Gestão de temporários

Nome	Livre
temp1	True
temp2	True
temp3	False
...	...

Metodo GetTemp()
Objetivo: Retornar um temporário livre
- Busca na lista
→ Se encontra um livre retorna-o e seta livre para False
→ Se não encontra cria um novo temp na lista retorna-o e seta livre para False

Metodo FreeTemp(temp)
Objetivo: Liberar um temporário
- Busca temp na lista e seta livre para True

Adicionar a lista de temporários na lista de declarações

BIP

```
.data
  val: 0
  maior: 0
  cont: 1
  temp1: 0
  temp2: 0
  temp3: 0
```

Gestão de temporários

Nome	Endereço	Propósito
Temporario 1	1000	Armazenar operando 1 de uma expressão
Temporario 2	1001	Armazenar operando 2 de uma expressão
Temporario 3	1002	Armazenar índice do vetor que recebe atribuição

Não modifica as declarações (dados) do programa

BIP

```
.data
  val: 0
  maior: 0
  cont: 1
```

Gestão de temporários

Nome	Endereço	Propósito
tempOp1	1000	Armazenar operando 1 de uma expressão
tempOp2	1001	Armazenar operando 2 de uma expressão
tempAtrib	1002	Armazenar índice do vetor que recebe atribuição

Não modifica as declarações (dados) do programa

BIP

```
.data
  val: 0
  maior: 0
  cont: 1
```

Geração de Código Intermediário
Entrada e saída de dados

Portugol
leia (val)
escreva (maior)
escreva (55)

M. Pilha	
LD	$in_port
STO	val
LD	maior
STO	$out_port
LDI	55
STO	$out_port

Expressões

```
<cmd> ::= leia '(' ID #7 ')'
        | escreva '(' ID #8 ')'
        | escreva '(' NUM_INT #9 ')'
```

```
#7 gera_cod ("LD ", "$in_port");
   gera_cod ("STO", token.getlexema());

#8 gera_cod ("LD ", token.getlexema());
   gera_cod ("STO", "$out_port");

#9 gera_cod ("LDI", token.getlexema());
   gera_cod ("STO", "$out_port");
```