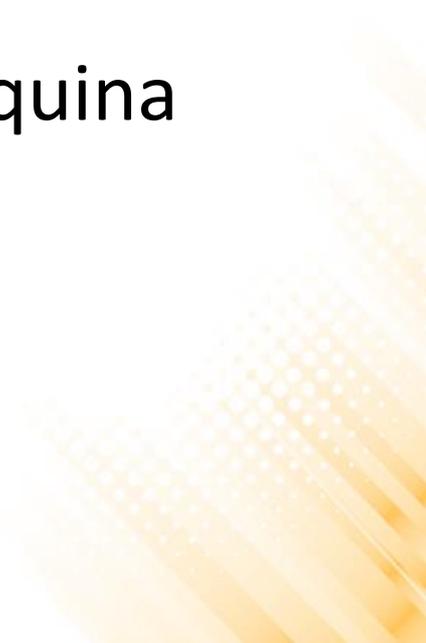


Compiladores

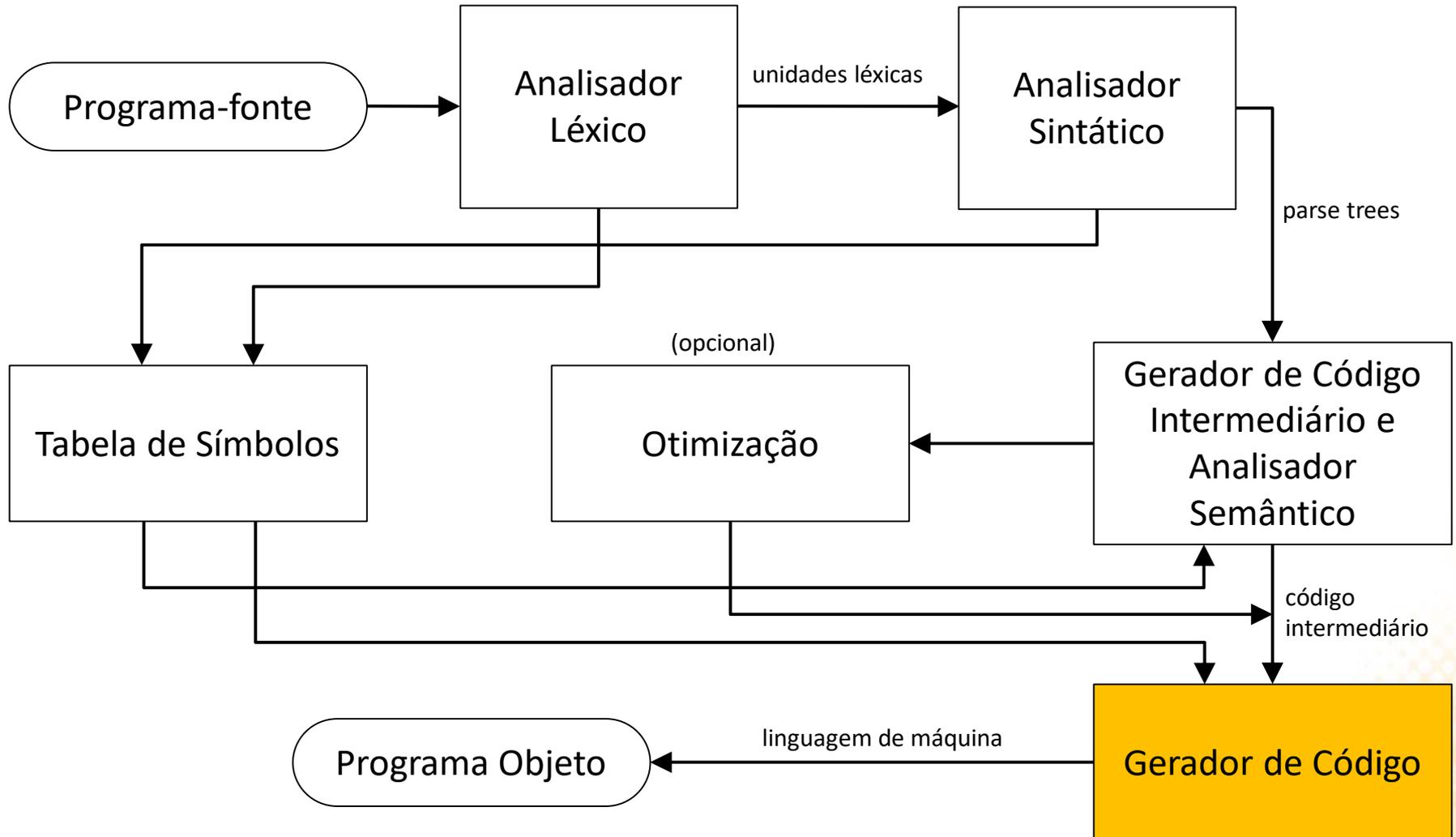
Aula 08 – Geração de Código de Máquina

Edirlei Soares de Lima

<edirlei.lima@universidadeeuropeia.pt>



Processo de Compilação



Processo de Compilação

- O **Gerador de Código** converte a versão do código intermediário do programa para um programa em linguagem de máquina equivalente.

```
gcd:  pushl %ebp
      movl  %esp,%ebp
      movl  8(%ebp),%eax
      movl  12(%ebp),%edx
.L8:  cmpl  %edx,%eax
      je    .L3
      jle  .L5
      subl %edx,%eax
      jmp  .L8
.L5:  subl %eax,%edx
      jmp  .L8
.L3:  leave
      ret
```

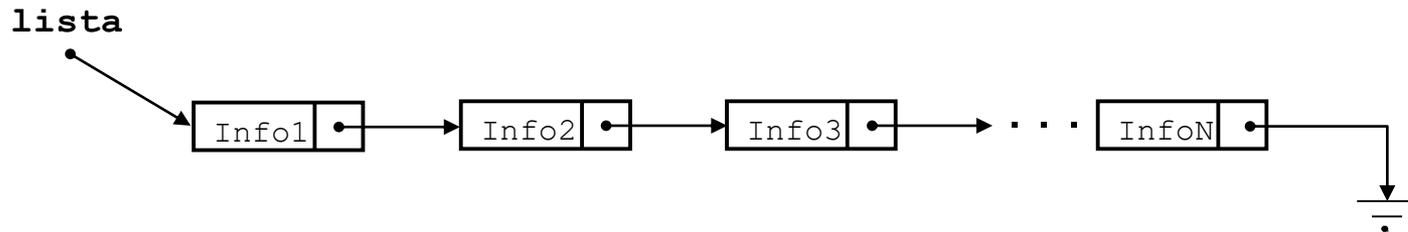
Assembly Language	Machine Code
add \$t1, \$t2, \$t3	04CB: 0000 0100 1100 1011
addi \$t2, \$t3, 60	16BC: 0001 0110 1011 1100
and \$t3, \$t1, \$t2	0299: 0000 0010 1001 1001
andi \$t3, \$t1, 5	22C5: 0010 0010 1100 0101
beq \$t1, \$t2, 4	3444: 0011 0100 0100 0100
bne \$t1, \$t2, 4	4444: 0100 0100 0100 0100
j 0x50	F032: 1111 0000 0011 0010
lw \$t1, 16(\$s1)	5A50: 0101 1010 0101 0000
nop	0005: 0000 0000 0000 0101
nor \$t3, \$t1, \$t2	029E: 0000 0010 1001 1110
or \$t3, \$t1, \$t2	029A: 0000 0010 1001 1010
ori \$t3, \$t1, 10	62CA: 0110 0010 1100 1010
ssl \$t2, \$t1, 2	0455: 0000 0100 0101 0101
srl \$t2, \$t1, 1	0457: 0000 0100 0101 0111
sw \$t1, 16(\$t0)	7050: 0111 0000 0101 0000
sub \$t2, \$t1, \$t0	0214: 0000 0010 0001 0100

Geração de Código de Máquina

- **Etapas:**
 1. Gravar o código intermediário (Assembly) em um arquivo;
 2. Gerar o código de máquina (arquivo .exe) utilizando o NASM e GCC.
- É necessário organizar o código intermediário de acordo com a estrutura de um programa Assembly, separando a seção de dados da seção do código do programa.
 - Solução: armazenar inicialmente o código intermediário em duas listas e depois gravar as linhas de código no arquivo Assembly.

Listas Encadeadas em C

- Uma **Lista Encadeada** é uma sequência de elementos, onde cada elemento tem uma informação armazenada e um ponteiro para o próximo elemento da sequência:



- Sequência encadeada de elementos, chamados de nós da lista;
- Nó da lista é representado por dois campos:
 - Informação armazenada;
 - Ponteiro para o próximo elemento da lista;
- A lista é representada por um ponteiro para o primeiro nó;
- O ponteiro do último elemento é NULL;

Listas Encadeadas em C

- Estrutura:

```
typedef struct element
{
    char *code;
    struct element *next;
} Element;
```

- Funções necessárias:

- Criação da lista:

- Element* CreateList();

- Inserção no final da lista:

- Element* InsertList(Element* list, char* code);

Listas Encadeadas em C

- Criação da lista:

```
Element* CreateList(){  
    return NULL;  
}
```

- Inserção no final da lista:

```
Element* InsertList(Element* list, char* code){  
    Element* elem = list;  
    Element* nelem = (Element*)malloc(sizeof(Element));  
    nelem->code = (char*)malloc(strlen(code) + 1);  
    strcpy(nelem->code, code);  
    nelem->next = NULL;  
    if (elem == NULL)  
        return nelem;  
    else{  
        while (elem->next != NULL){  
            elem = elem->next;  
        }  
        elem->next = nelem;  
    }  
    return list;  
}
```

Geração de Código Máquina (Implementação)

...

```
void GenerateMachineCode(Node * ast)
{
    Element* datacode = CreateList();
    Element* progcode = CreateList();
    int i = 0;
    char codeline[256];
    char path[256];
    char cmd1[1024];
    char cmd2[1024];
    datacode = InsertList(datacode, "SECTION .data\n");
    datacode = InsertList(datacode, "formatout: db \"%d\", 10, 0\n");
    progcode = InsertList(progcode, "SECTION .text\n");
    progcode = InsertList(progcode, "global _main\n");
    progcode = InsertList(progcode, "extern _printf\n");
    progcode = InsertList(progcode, "_main :\n");
    GenerateCode(ast, datacode, progcode);
}
```

...

Geração de Código Máquina (Implementação)

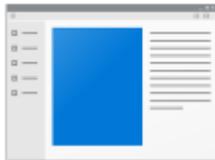
- Executando o NASM e GCC:

```
...  
  
getcwd(path, sizeof(path));  
chdir("C:\\MinGW\\bin\\");  
sprintf(cmd1, "C:\\MinGW\\bin\\nasm.exe -f win32 %s\\code.asm", path);  
system(cmd1);  
sprintf(cmd2, "C:\\MinGW\\bin\\gcc.exe %s\\code.obj -o %s\\code.exe", path, path);  
system(cmd2);  
  
...
```

Geração de Código Máquina (Implementação)

- Entrada:

```
int test;  
test = (46 + 100) / 2;  
print(test);
```



code.exe

A screenshot of a Windows Command Prompt window titled "Prompt de Comando". The window shows the command "code.exe" being executed, resulting in the output "73". The command prompt is located at "C:\Users\edir1\Desktop\ProjetoCompilador\ProjetoCompilador".

```
cmd Prompt de Comando  
C:\Users\edir1\Desktop\ProjetoCompilador\ProjetoCompilador>code.exe  
73  
C:\Users\edir1\Desktop\ProjetoCompilador\ProjetoCompilador>_
```

Leitura Complementar

- Aho, A. V., Lam, M. S., Jeffrey, R. S.
Compiladores: Princípios, Técnicas e Ferramentas. 2ª edição, Pearson, 2007.
ISBN: 978-8588639249.
 - Capítulo 8: Code Generation

