CIT-411 Compiler and Automata Theory

to be updated

Masud Sir

Book: Compilers: Principles, Techniques, & Tools

Chapter 3 + 4 → 1 Set

Chapter 1

Chapter 3

Chapter 4

Chapter 6

Figure 6.1: Logical structure of a compiler front end

Figure 6.2: A compiler might use a sequence of intermediate representations

6.1 Varients of Syntax Trees 6.1.1 Directed Acyclic Graphs for Expressions

Example 6.1 : Figure 6.3 shows the DAG for the expression $a + a * (b - c) + (b - c) * d$

Figure 6.4: Syntax-directed de nition to produce syntax trees or DAG's

Figure 6.5: Steps for constructing the DAG of Fig. 6.3

6.1.2 The Value-Number Method for Constructing DAG's

Figure 6.6: Nodes of a DAG for i = i + 10 allocated in an array

6.2 Three-Address Code ⋆⋆⋆ Expression → DAG and Three Address Code

Example 6.4

6.2.1 Addresses and Instructions

An address can be one of the following:

–

Here is a list of the common three-address instruction forms:

–

Example 6.5 : Consider the statement `do i = i+1; while (a[i] < v);`

6.2.2 Quadruples

The following are some exceptions to this rule:

–

Example 6.6

Figure 6.10: Three-address code and its quadruple representation

6.2.3 Triples

Figure 6.11: Representations of `a = b * - c + b * - c ;`

⋆⋆ Comparison between Quadraples and Triples. Use Figure 6.10 and 6.11 for that

6.3.1 Type Expressions

6.5 Type Checking 6.5.2 Type Conversions

⋆⋆⋆ Example 6.22 : Consider again the following statement from Example 6.21: `if( x < 100 || x > 200 && x != y ) x = 0;` (6.13)

Mim Ma'am

Google Drive Link

Book: Introduction to Automata Theory, Languages, and Computation

Chapter 1

Chapter 2

Chapter 4

Muhtasim Sir

Book: Compilers: Principles, Techniques, & Tools

Chapter 5 : Syntax-Directed Translation

Slide Link

Syntax-Directed Definition → Syntax-Directed Translation

to be updated