CD: UNIT-3

Syntax-Directed Translation

FALL SEMESTER, YEAR (V/VI, 3rd)

FALL SESSION (2022-23)
(CD)
MS. SHWETA TIWARI
Published: September, 2022

Shweta Tiwari

shwetatiwari08@recabn.ac.in
shwetatiwari08aug@gmail.com

TOPIC On : UNIT-3

Method of Generating Intermediate Code Generation with 3 -Address Code with different form

—

By SHWETA TIWARI

Under On: Syntax-Directed Translation

PREPARED FOR
Engineering Students
All Engineering College

PREPARED BY
SHWETA TIWARI
Guest Faculty

(3.) Three - Address code

- In three address code form at the most three addresses are used to represent any statement i.e. there is at most one operator on the right side of an instruction

- The general form of three address code representatio is -

$$a := b \text{ op } c$$

where a, b or c are the operands that can be names, constants, compiler generated temporaries and op represents the operator. The operator can be fixed or floating point arithmetic or logical operators on Boolean valued data.

e.g $a = b+c+d$

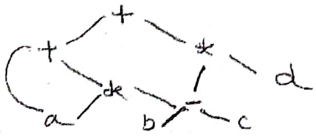
the three address Code :

$t_1 := b+c$
$t_2 := t_1+d$
$a := t_2$

Here t_1 and t_2 are the temporary names generated by the compiler. There are at the most three addresses allowed (two for operands and one for result).

- Three - address code is a linearized representation of a syntax tree or a DAG in which explicit names correspond to the interior nodes of the graph.

e.g $a + a * (b-c) + (b-c) * d$



$t_1 = b-c$
$t_2 = a * t_1$
$t_3 = a + t_2$
$t_4 = t_1 * d$
$t_5 = t_3+t_4$

Implementation of Three Address Code

- Three address code is an abstract form of intermediate code that can be implemented as a record with the address fields.

- There are three representations used for three address code such as quadruples, triples and indirect triples.

Quadruple Representation

- The quadruple is a structure with at most four fields such as op, arg1, arg2, result. The op field is used to represent the internal code for operator, the arg1 and arg2 represent the two operands used and result field is used to store the result of an expression

e.g $x := -a * b + -a * b$

The three address code is

$t_1 := \text{uminus } a$

$t_2 := t_1 * b$

$t_3 := \text{uminus } a$

$t_4 := t_3 * b$

$t_5 := t_2 + t_4$

$x := t_5$

Quadruple

Location	Op	Arg1	Arg2	result
(0)	uminus	a		t_1
(1)	*	t_1	b	t_2
(2)	uminus	a		t_3
(3)	*	t_3	b	t_4
(4)	+	t_2	t_4	t_5
(5)	:=	t_5		x

riples

In the triple representation the use of temporary variable is avoided by referring the pointers in the symbol table

eg $x := -a * b + -a * b$

Location	Op	Arg 1	Arg 2
(0)	uminus	a	
(1)	*	(0)	b
(2)	uminus	a	
(3)	*	(2)	b
(4)	+	(1)	(3)
(5)	:=	x	(4)

- A triple has only three fields op, arg1, and arg2.
- Using triples, the result of an operation x op y is referred by its position, rather than by an explicit temorary name.
- Parenthesized numbers represent pointers into the triple structure itself.

Indirect Triples

- In the indirect triple representation the listing of triples is done. And listing pointers are used instead of using statements.

Location	Op	Arg 1	Arg 2
(0)	uminus	a	
(1)	*	(11)	b
(2)	uminus	a	
(3)	*	(13)	b
(4)	+	(12)	(14)
(5)	:=	X	(15)

Location	statement
(0)	(11)
(1)	(12)
(2)	(13)
(3)	(14)
(4)	(15)
(5)	(16)

benefit of quadruples over triples can be seen in an optimizing compiler, where instructions are often moved around. with quadruples, if we move an instruction that computes a temporary t, then the instructions that use t require no change.

with triples, the result of an operation is referred to by its position, so moving an instruction may require us to change all references to that result.

- This problem does not occur with indirect triples, with indirect triples, an optimizing compiler can move an instruction by reordering the instruction list, without affecting the triples themselves.

Translate the following expression to quadruple, (4) triple and indirect triple

$-(x+y) * (z+c) - (x+y+z)$

Solⁿ Three address code:

$t_1 := x+y$

$t_2 :=$ uminus t_1

$t_3 := z+c$

$t_4 := t_2 * t_3$

$t_5 := t_1 + z$

$t_6 := t_4 - t_5$

Quadruple

Location	operator	operand 1	operand 2	result
(1)	+	x	y	t_1
(2)	uminus	t_1		t_2
(3)	+	z	c	t_3
(4)	*	t_2	t_3	t_4
(5)	+	t_1	z	t_5
(6)	−	t_4	t_5	t_6

Triple

Location	operator	operand 1	operand 2
(1)	+	x	y
(2)	uminus	(1)	
(3)	+	z	c
(4)	*	(2)	(3)
(5)	+	(1)	z
(6)	−	(4)	(5)

Indirect Triple

Location	operator	operand1	operand2
(1)	+	x	y
(2)	uminus	(11)	
(3)	+	z	c
(4)	*	(12)	(13)
(5)	+	(11)	z
(6)	-	(14)	(15)

Location	statement
(1)	(11)
(2)	(12)
(3)	(13)
(4)	(14)
(5)	(15)
(6)	(16)

Representation intermediate Code Generation using 3-Addresscode. for expression with different form "Quadruple, Triple, Indirect triple")

Question-1 Translate the following expression to ("Quadruple, Triple, Indirect Triple")

$x*y-5+Z$

Answer

3-Address code of given expression

$x*y-5+Z$

$t_1 = x*y$

$t_2 = t_1-5$

$t_3 = t_2+Z$

3-Address Code

① Quadruple Representation -
Maximum 4 field

Location	operator	Argument-1	Argument-2	Result
(0)	*	x	y	t_1
(1)	-	t_1	5	t_2
(2)	+	t_2	Z	t_3

Shweta Tiwari

(2) Triple Representation -

Maximum 3 field

Location	operator	Arg1	Arg2
(0)	*	x	y
(1)	−	(0)	5
(2)	+	(1)	z

(3) Indirect Triple

maximum 3 field

Loc	op.	Arg1	Arg2
(0)	*	x	y
(1)	−	[11]	5
(2)	+	[12]	z

Loc	Statement
(0)	[11]
(1)	[12]
(2)	[13]

Question :- Translate the fallowing expression to (Quadruple Triple, Triple, Indirect Triple).

$$z = (A * B) - (C + D) + E$$

Answer -

3-Address Code of given expression

$$z = (A * B) - (C + D) + E$$

$t_1 = A * B$

$t_2 = C + D$

$t_3 = t_1 - t_2$

$t_4 = t_3 + E$

$z = t_4$

① Quadruple Representation

Location	Operator	Argument1	Argument 2	Result
(1)	*	A	B	t_1
(2)	+	C	D	t_2
(3)	−	t_1	t_2	t_3
(4)	+	t_3	E	t_4
(5)	=	t_4	—	Z

② Triple Representation

Location	Operator	Argument 1	Argument 2
(1)	*	A	B
(2)	+	C	D
(3)	−	(1)	(2)
(4)	+	(3)	E
(5)	=	Z	(4)

③ Indirect Triple Representation

Location	Operator	Argument 1	Argument 2
(1)	*	A	B
(2)	+	C	D
(3)	−	[11]	[12]
(4)	+	[13]	E
(5)	=	Z	[14]

Location	Statement
(1)	[11]
(2)	[12]
(3)	[13]
(4)	[14]
(5)	[15]

P4-3

Question-3 Translate the fallowing expression to (Quadruple, triple, Indirect Triple)

$x = (a*b)+(c-d)*(a*b)+b$

Answer 3- Address Code of given expression

$x = (a*b)+(c-d)*(a*b)+b$

$t_1 = a*b$

$t_2 = c-d$

$t_3 = t_2 * t_1$

$t_4 = t_1 + t_3$

$t_5 = t_4 + b$

$x = t_5$

(1.) Quadruple Representation

Location	operatar	Argument 1	Argument2	Result
(0)	*	a	b	t1
(1)	-	c	d	t2
(2)	*	t2	t1	t3
(3)	+	t1	t3	t4
(4)	+	t4	b	t5
(5)	=	t5	—	x

② Triple Representation

Location	operator	Argument 1	Argument 2
(0)	*	a	b
(1)	−	c	d
(2)	*	(1)	(0)
(3)	+	(0)	(3)
(4)	+	(3)	b
(5)	=	z	(4)

③ Indirect Triple Representation

Location	operator	Argument 1	Argument 2
(0)	*	a	b
(1)	−	c	d
(2)	*	[12]	[11]
(3)	+	[11]	[14]
(4)	+	[14]	b
(5)	=	z	[15]

Location	Statement
(0)	[11]
(1)	[12]
(2)	[13]
(3)	[14]
(4)	[15)
(5)	[16]

Question-4 Representation 3-Address code of following expression with different form (Quadruple, Triple, Indirect Triple)

$((A+B)-C*(D/E))+F$

Answer

3-Address code of given expression

$((A+B)-C*(D/E))+F$

$t_1 = A+B$

$t_2 = D/E$

$t_3 = C*t_2$

$t_4 = t_1-t_3$

$t_5 = t_4+F$

① Quadruple Representation

Location	Operator	Argument 1	Argument 2	Result
(0)	+	A	B	t_1
(1)	/	D	E	t_2
(2)	*	C	t_2	t_3
(3)	−	t_1	t_3	t_4
(4)	+	t_4	F	t_5

② Triple Representation

Location	Operatar	Argument 1	Argement 2
(0)	+	A	B
(1)	/	D	E
(2)	*	C	(1)
(3)	−	(0)	(2)
(4)	+	(3)	F

③ Indirect Triple Representation

Location	operatar	Argumant 1	Argement 2
(0)	+	A	B
(1)	/	D	E
(2)	*	C	[12]
(3)	−	[11]	[13]
(4)	+	[14]	F

Location	Statement
(0)	[11]
(1)	[12]
(2)	[13]
(3)	[14]
(4)	[15]

Shweta Tiwari