Scribe Submission- Report (date – 30th Oct 2013)

By Nikhil Agrawal (11CS30021)

Topics Covered :

1. Intermediate Code generation
   a. Syntax Tree
   b. 3 – address Code

Intermediate Code Generation

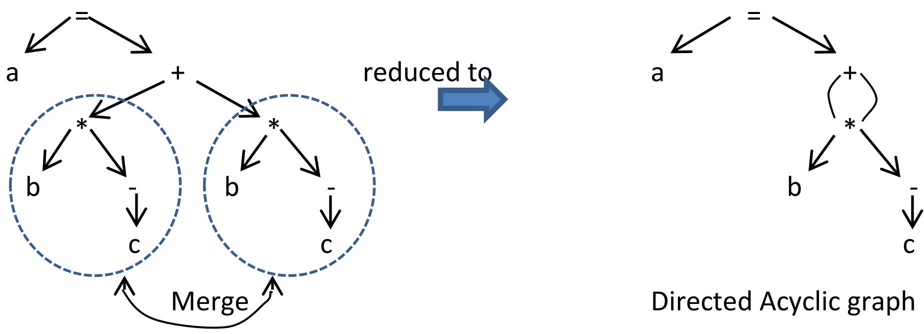
How are Intermediate code Represented?

Options:

1. Syntax Tree
2. 3-address code

Syntax tree

a = b* (-c) + b*(-c)



Directed Acyclic graph (DAG)

Let Grammar be

E → E + T

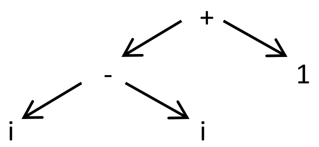
E → E – T

E → T

T → id

T → num

Convert   1) i – i + 1 in Directed Acyclic graph

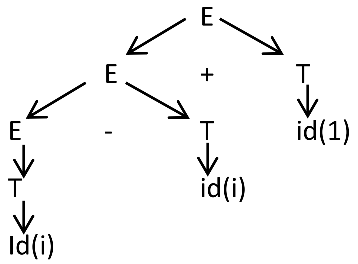


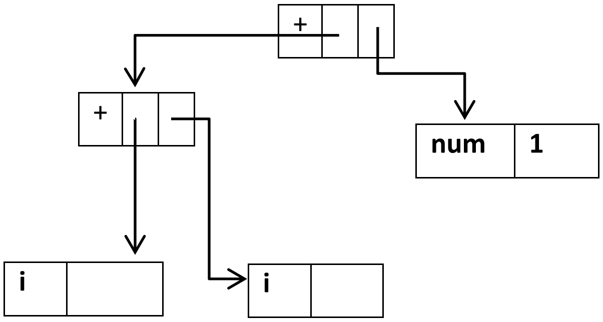
Attributes Related to Non-Terminal Symbols

- Node Address

Semantic Rules

E → E_1 + T	E.node_address = new node(+,E_1.node_address,T.node_address)
E → E_1 – T	E.node_address = new node(+,E_1.node_address,T.node_address)
E → T	E.node_address = T. node_address
T → id	T. node_address = new leaf(id, ptr to symbol table)
T → num	T. node_address = new leaf(num, number value)





Note:- For constructing a compact DAG just before creation of new node we check whether such a node is already been created or not

Leaf node:

- Token name (id, num)
- Attribute

Internal node:

- Operator info
- Left Child and Right child

Storage OF DAG:

Dag is stored in Table in which each entry represents a node (both internal And leaf node)

1	**i**	**Pointer To symbol Table**	
2	**Num**	**1**	
3	**+**	**1**	**2**
4	=	**1**	**3**

Note : We need to implement a Efficient searching (Hashing).

Signature for particular node: < op , l , r > or < id, attribute >

Three Address Code Generation

- Form the intermediate representation
- At most 3 different Address
- At most one operator at the right

X + Y * Z → $t_1 = Y * Z$

$t_2 = x + t_1$

Note:

The assignment operator is not the one normally used (or the one present in the input grammar) means it calculate the value of the sub-expression on the right and assign it to a temporary variable.

Exercise: Find the 3-address code for the instruction:

a+ a* (b – c) + (b – c) * d

Soln: t1 = b – c

t2 = t1 * d

t3 = a*t1

t4 = t2 + t3

t5 = a + t4