B

Optimization of Basic Blocks

~ Many of the structure preserving transformation is implemted by constructing a DAG (Directed Acyclic Graph) for a block.

~ There is a node 'n' associated with each ~~elemt~~ statement's within the block. The children of n are those nodes corresponding to statements that are the last definitions.

~ These are prior to the statemens s of the operands used by them. (s).

Basic Block = set of statements exequted in a sequencial manner.

~ Every basic block contains one entry point & one exit point.

~ Within a basic block there are no ~~conditiat~~ control statements neither conditional nor ~~uncondi~~ unconditional.

* Properties of a DAG

1. Internal nodes in a DAG, represents operators.
2. Leaf node represents identifiers, constants
3. Internal node may also represent ~~identifier~~ result of expression.

eg:-

$t = a + b$

* Application of DAG

1. Determine the common sub expressions
2. Determines which names are inside the block & are computed outside the block.
3. Helps in determining which statement of the block could have their values computed outside the block.
4. Simplying the list of quadruples by eliminating the common sub expressions.

* Rules for the Construction of a DAG

1. In a DAG leaf node represents identifiers

names, constants. Interior node represents operators.

2. While constructing DAG, there is a check made to find if there is an existing node with same children.

- A new node is created only when such a node does not exist.
- This helps to detect common sub expression and eliminate the same.

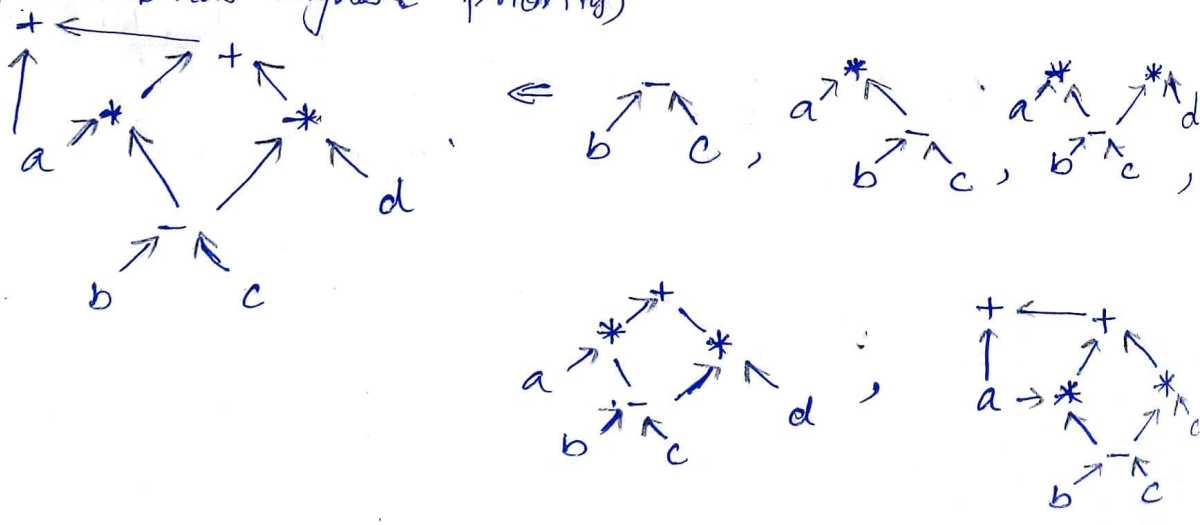
3. Assignment of the form $x = y$ must not be performed until unless it is a must.

eg:-

1. Construct the DAG for the expression:

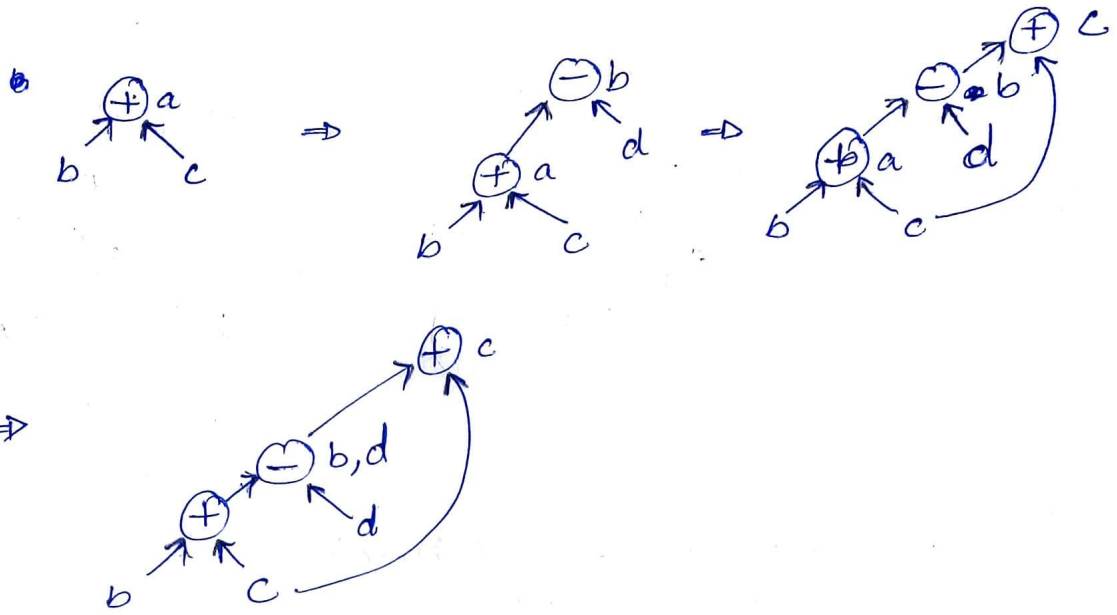
$a + a * (b-c) + (b-c) * d$

(parantup has highest priority)

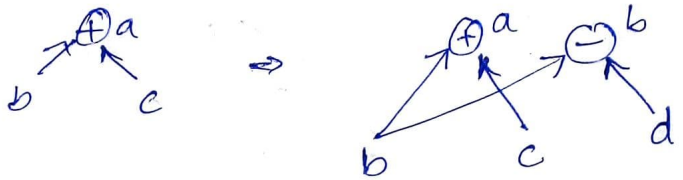


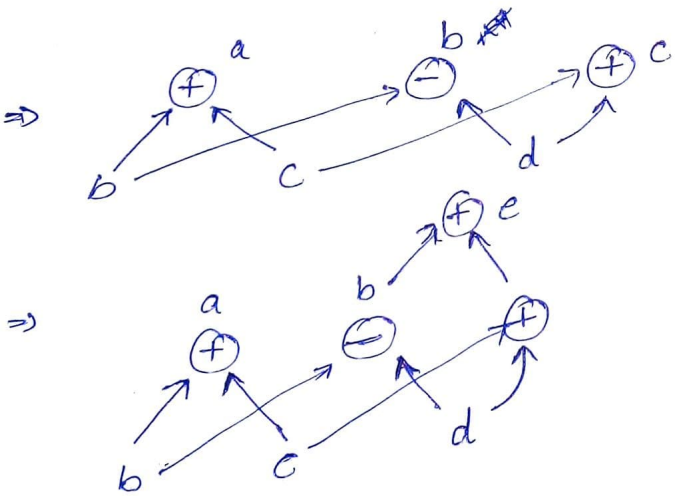
2. Construct DAG, for the block.

1. $a = b + c$
2. $b = a - d$
3. $c = b + c$
4. $d = a - d$

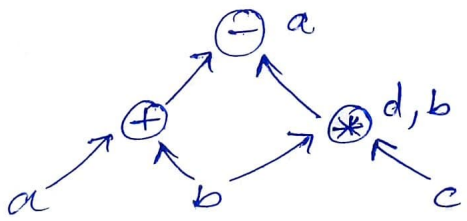


3. 1. $a = b + c$
2. $b = b - d$
3. $c = c + d$
4. $e = b + c$

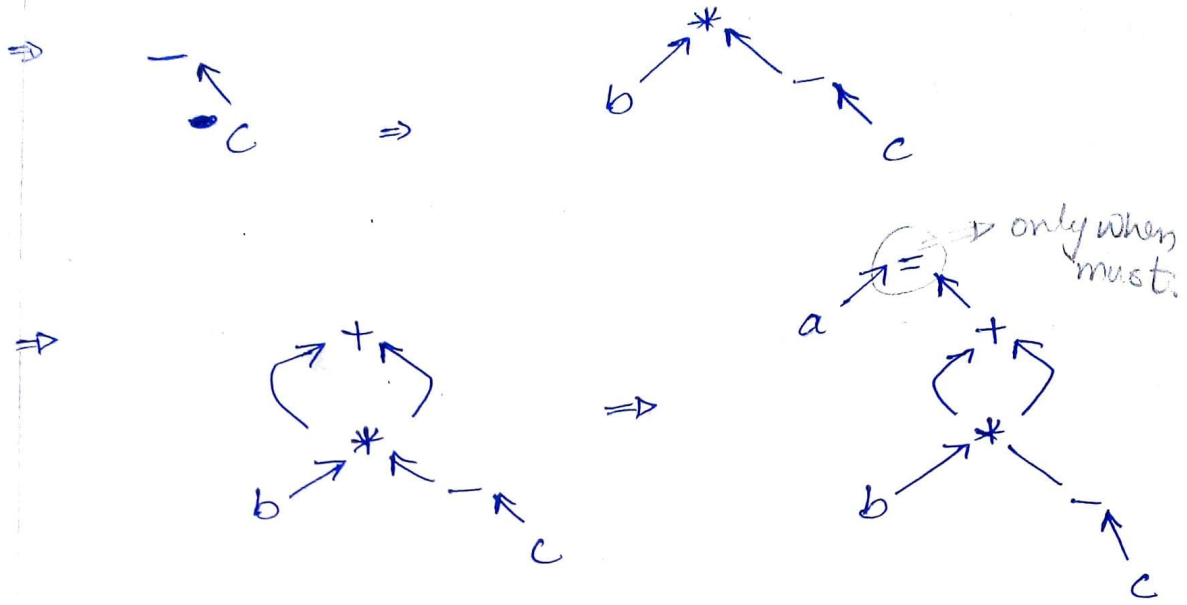




4. $d = b * c$
$e = a + b$
$b = b * c$
$a = e - d$



5. $a = b * -c + b * -c$



6. $a = (a * b + c) - (a * b - c)$