Unit-5

DAG.

* It Stands for Direct Acyclic graph.

* Three address code is used to generate DAG.

* It is an optimazation technique

eg:

a = b * c

d = b

e = d * c

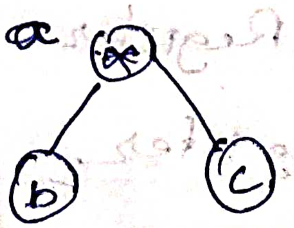
b = e

f = b + c

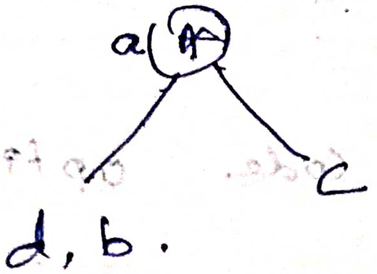
g = ~~f+d~~ d+f

Step 1:

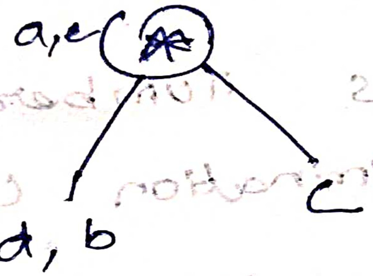
* consider the first statement



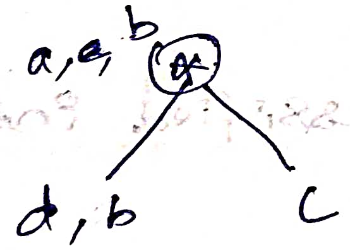
Step 2 :



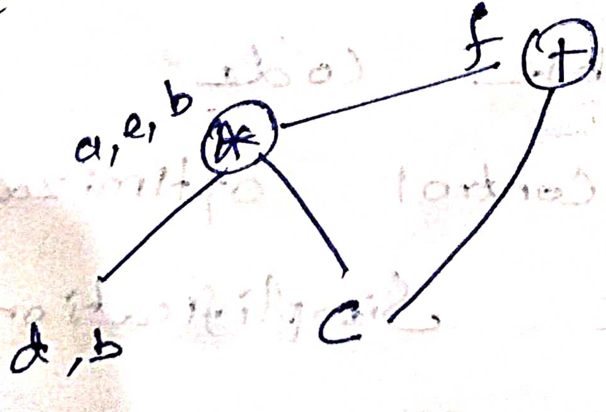
Step 3 :



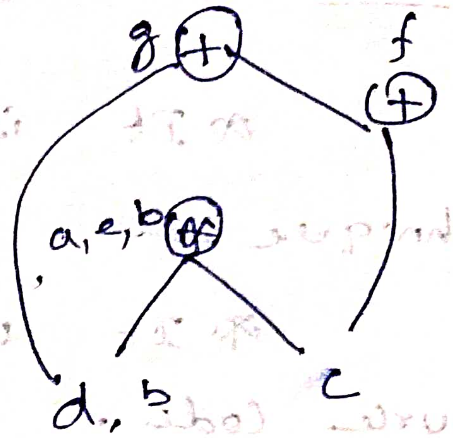
Step 4 :



Step 5 :



Step 6 :



2) Peep hole optimization:

* It is an code optimization technique

* It works on locally on source code.

* It reduces number lines and instruction elimination code

* It helps or increases system performance.

* It is classified into 4 types they are

i) Redudant instruction elimination

2) Unreachable code

3) Flow of control optimization

4) Algebraic Simplification.

1) Redudant instruction elimination:

* In this Mechanism no of instructions or codes are eliminated

eg:

```
void add ( int x)
{
    int y, z
    y = 10
    z = x + y
    return z
}
```

```
void add (int x, y)
{
    int z = x + y
    return z
}
```

```
void add (int x, int y)
{
    return x+y
}.
```

unreachable code:

* Repeated codes are slightly eliminated

```
void printf ()
{
    return x
    printf("%d", x)
}
```

Flow control optimization.

Loop codes are eliminated in this segment

```
while(x<10)
{
    x=a+b;
    printf(x)
}.
```

```
x=a+b
while(x<10)
{
    printf(x)
}.
```

Algebraic simplification:

```
a=a+0     //not useful
a=a+1     // usefoll one.
```

Principle of Source optimization

* It is an code optemization technique

* It is classified into six types.

1) common subexpression elimination
2) constant Folding
3) copy propogation
4) code motion
5) Dead code elimination
6) Induction variable elimination

1) common Subexpression Elimination.

```
a=b+c
b= a-d
c = b+c;
d= a-d
```

$\Longrightarrow$

```
a = b+c
b = a-d
c = b+c
d = b
```

Copy propagation

```
x=a
y=x+b
z=x+y
```

=>

```
x=a
y=a*b
z=a*c
```

Dead Code Elimination:

```
x=a
y=a*b,
z=a*c,
```

=>

```
y=a*b;
z=a*c
```

Code motion.

```
while(i<10)
{
  x=y+z
  i=i+1
}
```

⇐

```
x=y+z
while(i<10)
{
  i=i+1
}
```

Induction Variable: Elimination

```
t=4
while (i<10)

{
    t = i*4;
    i = i+1;
}
```

```
t=4
while(i<40)

{
    t = t+4;

}
```