[illegible] :- find about domain name info. by using whois

the website to get detailed info about domain name info including the owner date of registration, expiry, name of owner, the contact info etc.

- Finding IP addr.

CC

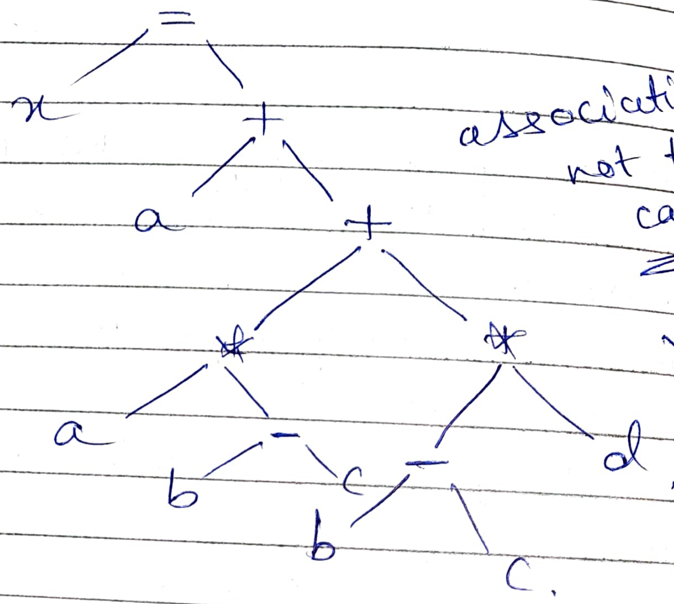
4. tokens → [syntax] → syntax tree → [type checking semantic analysis] → Intermediate Code Repr[illegible]

- <u>Intermediate Code Representat^n</u> -

1) Syntax tree. 2). DAGs (Directed Acyclic Gr[illegible]

ex :- x = a + a * (b-c) + (b-c) * d

AST

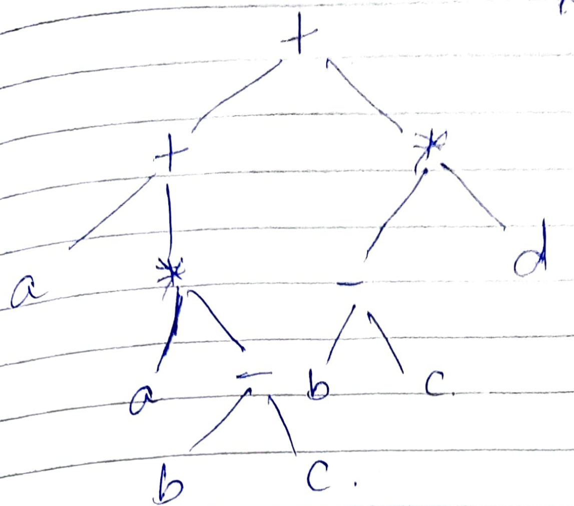


associativity not taken care off

X.

DAG

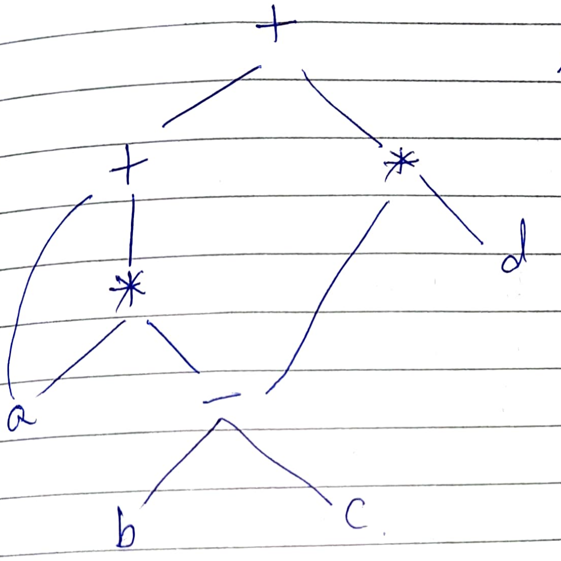
* '+' left associative.



AST

DAG.

common subexpressions removed.



3AC(3 address code) {at most 3 addresses}.

- ↳ quadruples
- ↳ triples.
- ↳ indirect triples.

ex:- a = y + x + z.

$t_1 = y + x$

$t_2 = t_1 + z$.

$a = t_2$

} 3AC for given expression

?

if(x<y) then add1 else add2 } more than 3 addresses

C me IR nhi hota.
Java has ICG thing!!

→ ~~Qa~~ quadriples— 4 columns/4 cells

	operator.	arg1	arg2.	result.
0	−	c		t_1
1	*	b	t_1	t_2
2	−	c		t_3
3	*	b	t_3	t_4
4	+	t_2	t_4	t_5.
5	=	t_5	~~a~~	a.
~~5~~ 6				

Why—* Pehle than +

ex :- a = b * − c + b * − c

⟹ '−' → Unary
hence Priority ↑
than Binary

→ triples

	op	arg1	arg2
0	−	c	
1	*	b	(0)
2	−	c	
3	*	b	(2)
4	+	(1)	(3)
5	=	a	(4).

?

→ Indirect triples.

35	(0)
36	(1)
37	(2)
~~38~~ 101	(3)
~~39~~ 108	(4)
~~40~~ 109	(5)
⋮	⋮

ss!- ICG with quadruples & triples, take any expression.

ass- SLR parser generator

Low level virtual machine

↑

- SSA (Single Static Assignment). [LLVM uses this]

$p_1 = a + b$

$q_1 = p_1 - c$

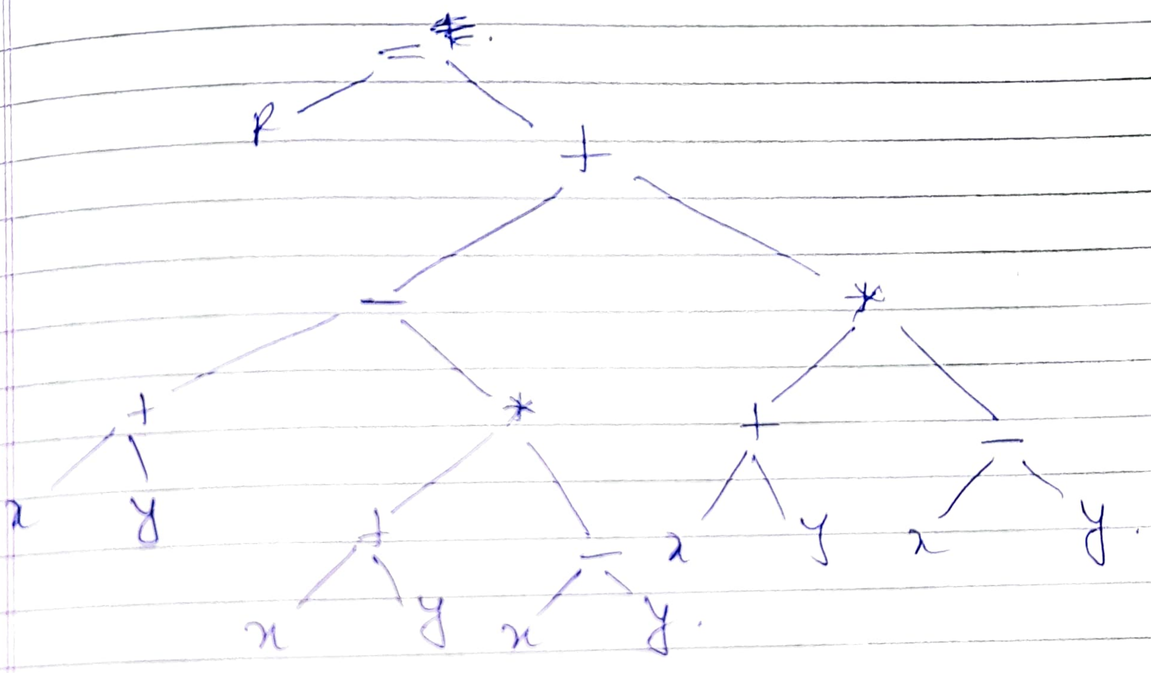
$p_2 = q_1 * d.$

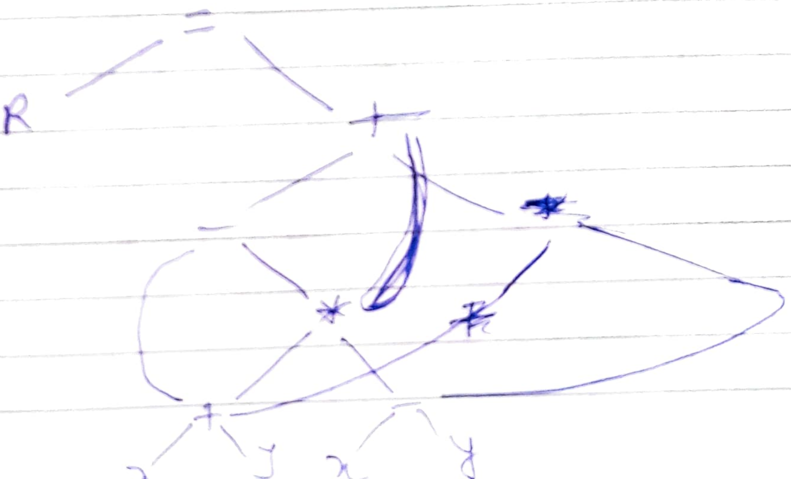
$p_3 = e - p_2$

$q_2 = p_3 + q_1$

} useful for Dataflow Analysis, at the time of optimizatⁿ, or CG.

ex:- $R = ((x+y) - ((x+y) * (x-y))) + ((x+y) * (x-y)).$





$$((x+y)-(\underline{(x+y)*(x-y)}))+(\underline{(x+y)*(x-y)})$$

	operator	org1	org2	result
0	+	x	y	t_1
1	−	x	y	t_2
2	*	t_1	t_2	t_3
3	+	x	y	t_4
4	−	t_4	t_3	t_5
5	+	x	y	t_6
6	−	x	y	t_7
7	*	t_6	t_7	t_8
8	+	t_5	t_8	t_9
9	=	t_9		~~a~~ R

	operator	org1	org2
0	+	x	y
1	−	x	y
2	*	(0)	(1)
3	+	x	y
4	−	(3)	(2)
5	+	x	y
6	−	x	y
7	*	(5)	(6)
8	+	~~t_5~~ (4)	(7)
9	=	(8)	R

Finding IP address

→ You can use ping command at your prompt.

→ This cammand i

5) Hybrid cloud Migration - Process :

techniques

1) left & shift.

2) Cloud bursting.

Pros : cost optimizat^n, flexibility, managing

Cons : Network latency.

CC

3AC

Arrays 1D $x = a[i]$

$base + i \times w$

$t_1 = i \times w$

$t_2 = a[t_1]$ $\{base + t_1\}$

→ 3AC

To find the address of any element in a 2-Dimensional array there are the following two ways:
Row Major Order
Column Major Order

2D Arrays

$$\begin{bmatrix} 00 & 01 & 02 \\ 10 & 11 & 12 \\ 20 & 21 & 22 \end{bmatrix}_{3\times 3}$$

X = a[i][j]
a[row][col]

row major

00	01	02	10	11	12	20	21	22

100.

example
↓
$100 + (2 \times 3 + 1) 4$
$= \boxed{128}$

$L = a[i][j] = base + (\underset{?}{i} \times \underset{col.}{c} + j) * \underset{\text{width of each element.}}{w}$

3AC
```
t1 = i x c
t2 = t1 + ~~t1~~ j
t3 = t2 x w
t4 = a[t3]  → NOT a+t3
x = t4.
```

col major

00	10	20	01	11	21	02	12	22

$base + (j \times \underset{row}{r} + i) * w.$

3D Arrays

a[i][j][k].   a[r][c][b].

$Loc\ a[i][j][k] = base + (\underset{?}{i} \times \underset{col.}{c} \times \underset{\text{last dim.}}{b} + j \times b~~2~~ + k) * \underset{width}{w}$

?
```
t1 = i x c            t5 = t4 + K.
t2 = t1 x b.          t6 = t5 * w
t3 = j x b            t7 = a[t6]
t4 = t2 + t3          ~~t8~~ x = t7 .
```

Q $t_0 = i \times 1024$

$t_1 = j \times 32$

$t_2 = k \times 4$

$t_3 = t_1 + t_0$

$t_4 = t_3 + t_2$

$t_5 = X[t_4]$

what is c & b ?

$j \times 32 + i \times 1024$

$j \times 32 + i \times 1024 + k \times$

$t_5 = a[j$

Ans $t_5 = X[i \times 1024 + j \times 32 + k \times 4]$

$= X\ 4(i \times 256 + j \times \boxed{8} + k)$

b

↳ integer array

$b = 8,$

$(i \times 8 \times 32$

$+ j \times 8 + k) \times 4$

$c =$

~~c=8,~~ $\boxed{c = 32,\ b = 8}$

- Boolean expression-

operator - &, |, !

precedence - not > and > or.

ex:- a and b or not c

$t_1 = \text{not } c$

$t_2 = a \text{ and } b$

$t_3 = t_2 \text{ or } t_1$

} 3AC

- Conditional statements -

→ if, while, for etc.

→ Jump or goto statements having addresses

if(a<b) then ~~x=y~~ 1 else 0

2 ways of doing it

way ① - 3AC -

100 : if $(a<b)$ goto 103

101 : $t_1 = 0$ → else part 1^{st}

102 : goto 104 → After else, exit

103 : $t_1 = 1$

104 :

way ② - 3AC -

L1 : if $(a<b)$ goto L_t

$t_1 = 0$

goto exit

L_t : $t_1 = 1$.

exit: ~~exi~~

ex:-2. if $(a<b)$ then $x=y+1$ else $x=y-1$

3AC -

100 : if $(a<b)$ goto 104

101 : ~~$x=y-1$~~ $t_1 = y-1$

102 : ~~goto 105~~ $x = t_1$.

103 : ~~$x=y+1$~~ goto 106

104 : ~~[exit or blank]~~ $t_1 = y+1$

105 : $x = t_1$

106 :

Eg value

$x = \boxed{a+b}$

↓ add^n

↓ so

$\boxed{t_1} = a+b$

$x = t_1$

ex3 :- while $(c<d)$ do

if $(a<b)$ then $x=y+1$ else $x=y-1$.

100 : if $(c<d)$ goto 102

101 : goto ~~109~~ → 109

102 : if $(a<b)$ goto 106

103 : $t_1 = y-1$

104 : $x = t_1$

105 : goto ~~108~~ 100

106 : $t_1 = y+1$

107 : $x = t_1$

108 : goto 100

109 : exit