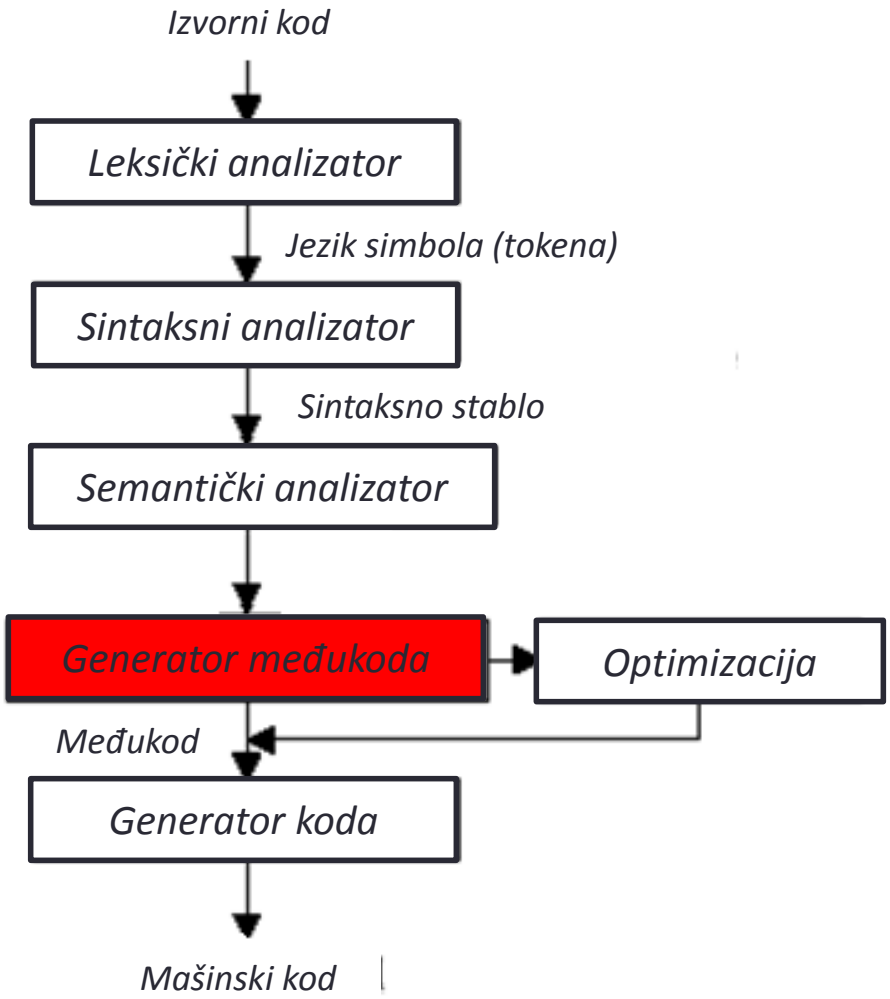
PROGRAMSI PREVODIOCI
- Generisanje međukoda -

Struktura kompilatora



Pozicija genratora međukoda

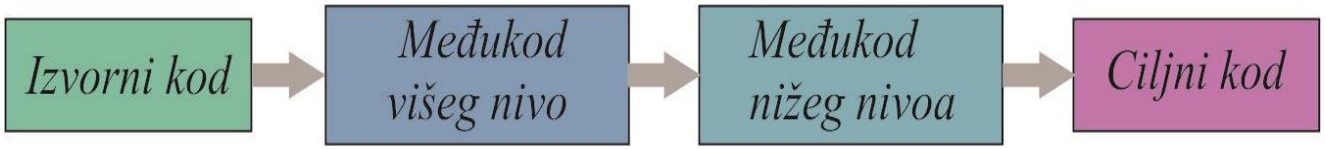
Razlozi za uvođenje međukoda:

1. Prenosivost koda. Na osnovu međukoda može se generisati kod za različite ciljne mašine
2. Na nivou međukoda može da se vrši mašinski nezavisna optimizacija

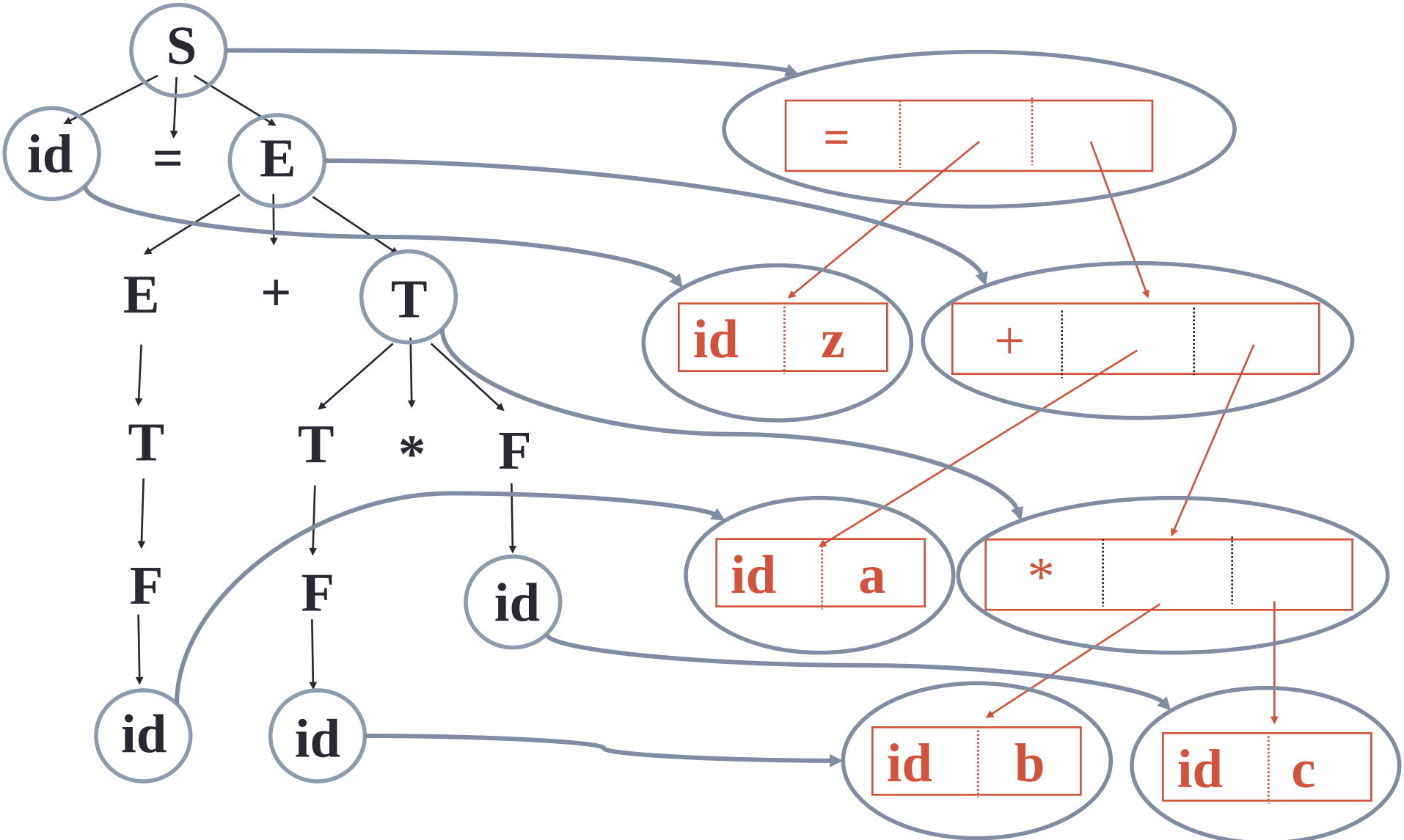
Vrste međukodova

- Međukodovi visokog nivoa:
  - Apstraktno sintaksno stablo
  - Poljska inverzna notacija
- Međukodovi niskog nivoa – pseudo-asemblerski jezici
  - Troadresni međukod

Transformacija koda tokom prevo]enja:



Generisanje apstraktnog sintaksnog stabla na osnovu sintaksnog stabla – Primer: z = a + b * c



Semantička pravila za generisanje apstraktnog sintaksnog stabla – Naredba dodeljivanja

SMENE	SEMANTIČKA PRAVILA
S->id = E	S.*nptr* = *mknode*(‘=‘, *mkleaf* (**id**, **id**.*place*), E.*nptr*)
E -> E_1 + T	E.*nptr* = *mknode* (‘+’, E_1.*nptr*, T.*nptr*)
E -> T	E.*nptr* = T.*nptr*
T -> T_1 * F	T.*nptr* = *mknode* (‘*’, T_1.*nptr*, F.*nptr*)
T -> F	T.*nptr* = F.*nptr*
F -> id	F.*nptr* := *mkleaf* (**id**, **id**.*place*)

Predstavljanje apstraktnog sintaksnog stabla u programu

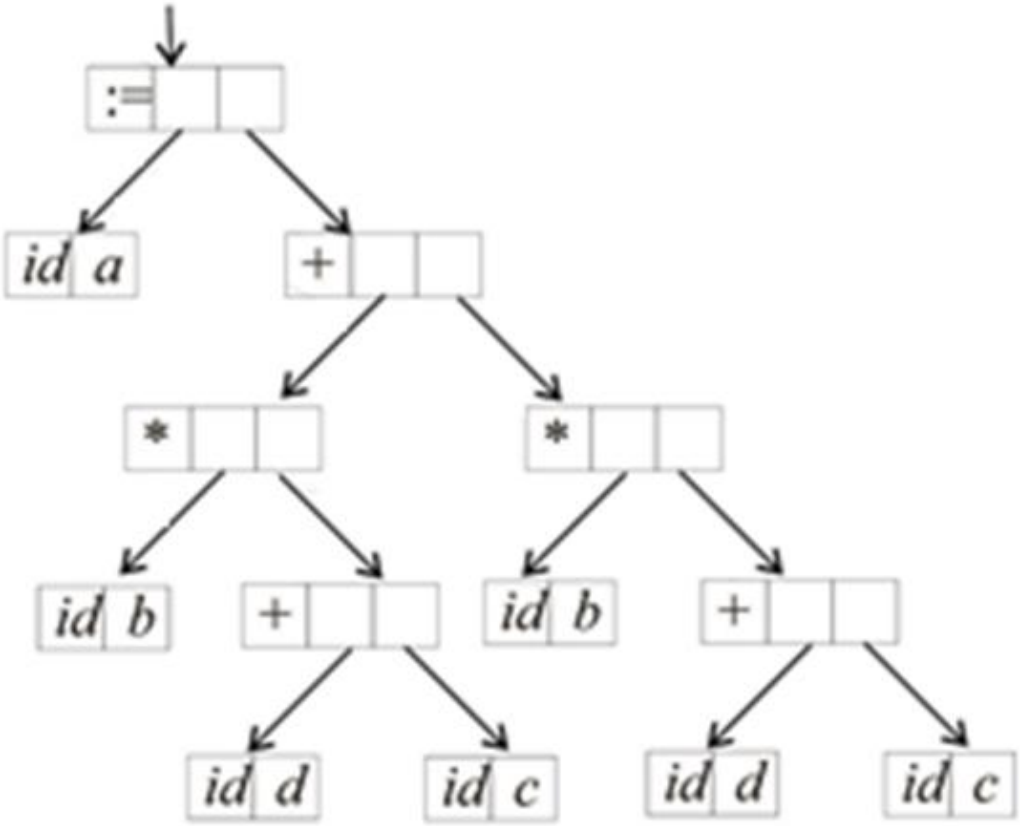
- Implementacija apstraktnog sintaksnog stabla zavisno od korišćenih fizičkih struktura podataka:
  - Dinamička (dinamičkom strukturom podataka)
  - Statička (statičkom strukturom podataka)

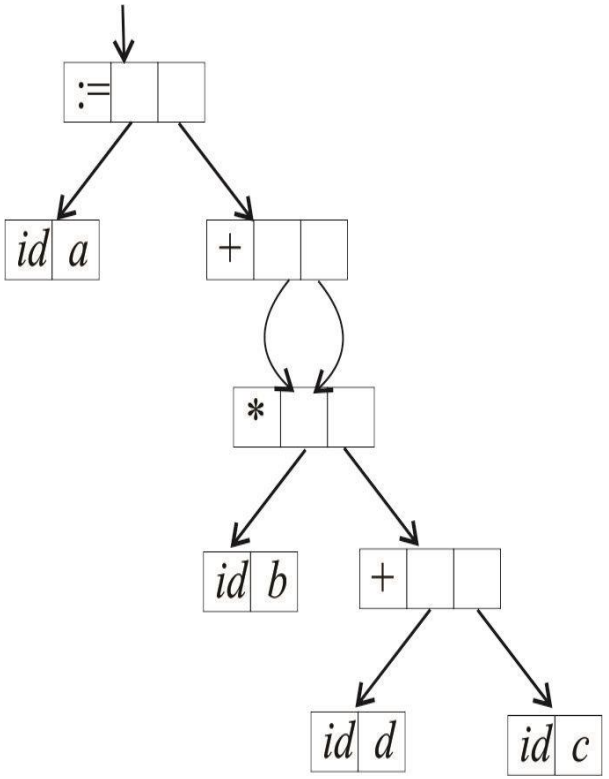
- Implementacija apstraktnog sintaksnog stabla zavisno od stepena kompresije:
  - Nekompresovana (pomoću stabla)
  - Kompresovana (pomoću dijagrama)

Dinamička implementacija apstraktnog sintaksnog stabla i sintaksnog dijagrama

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

$id = id * (id + id) + id * (id + id)$

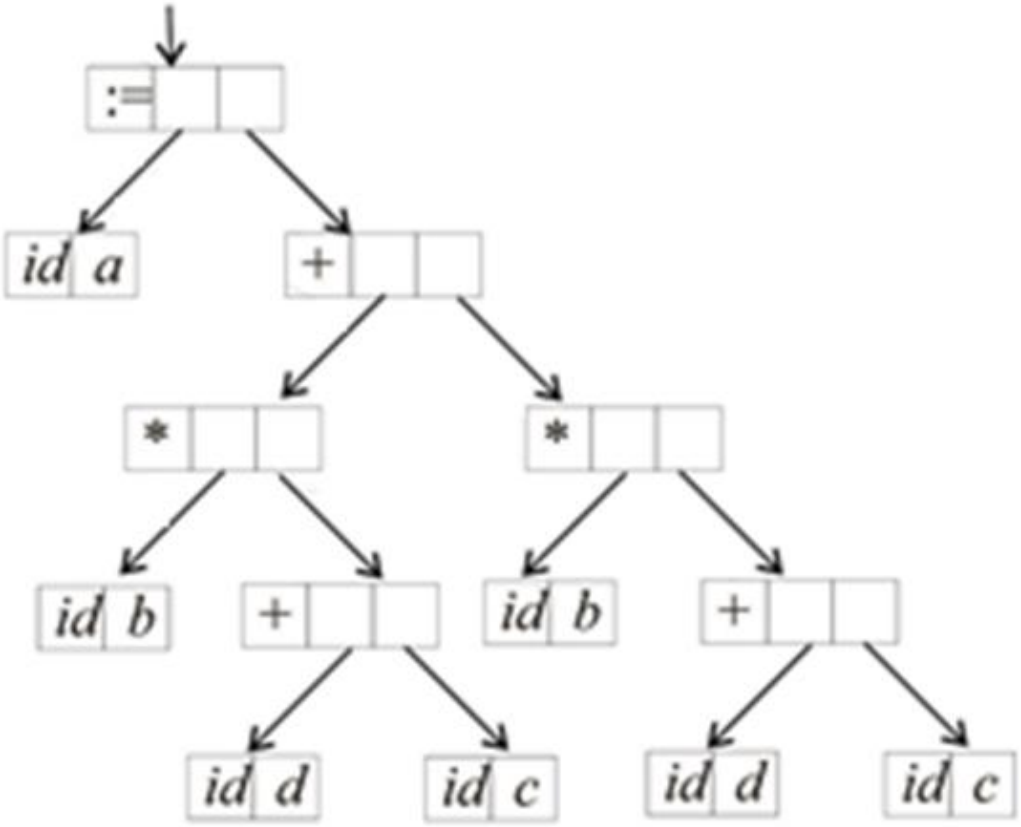




Dinamička i statička implementacija apstraktnog sintaksnog stabla

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

$$id = id * (id + id) + id * (id + id)$$

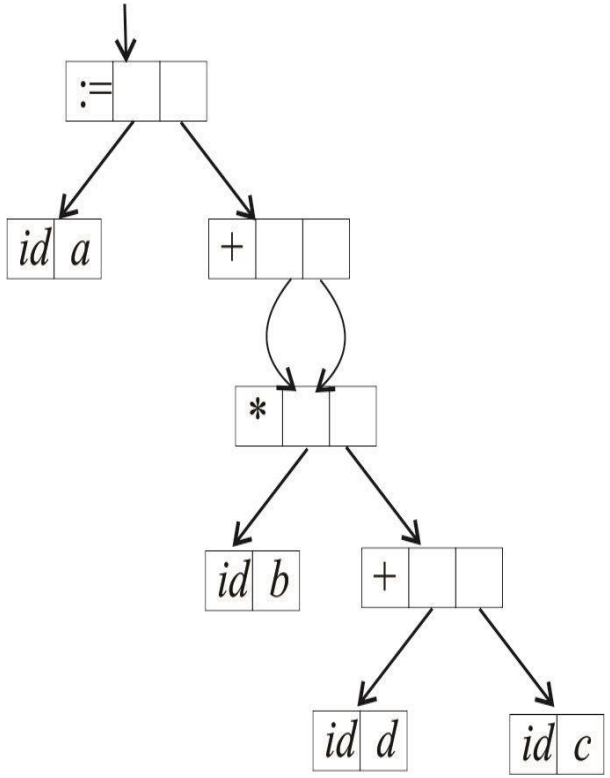


0	*id*	a	
1	*id*	b	
2	*id*	d	
3	*id*	c	
4	+	2	3
5	*	1	4
6	*id*	b	
7	*id*	d	
8	*id*	c	
9	+	7	8
10	*	6	9
11	+	5	10
12	=	0	11
13	...		

Dinamička i statička implementacija sintaksnog dijagrama

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

$$id = id * (id + id) + id * (id + id)$$



0	*id*	a	
1	*id*	b	
2	*id*	d	
3	*id*	c	
4	+	2	3
5	*	1	4
6	+	5	5
7	=	0	6
8	...		

Međukodovi niskog nivoa

- Bliski asemblerskim jezicima
- Jedna instrukcija definiše jednu elementarnu operaciju
- Jedine upravljačke strukture (kao i u asemblerskim jezicima):
  - Naredbe uslovnog i bezuslovnog skoka

- Troadresni međukod
  - Međukod niskog nivoa
  - Linearna reprezentacija apstraktnog sintaksnog stabla

Naredbe troadresnog međukoda

1. Dodela oblika x := y *op* z, gde je *op* aritmetički ili logički binarni operator.
2. Dodela oblika x := *op* y, gde je *op* aritmetički ili logički unarni operator (npr. minus, logička negacija, operatori za konverziju tipova i sl.)
3. Nredba kopiranja x := y kojom x dobija vrednost y.
4. Naredba bezuslovnog skoka go to L.
5. Naredba uslovnog skoka if x *relop* y goto L.

Naredbe troadresnog međukoda

5. param x i call p,n za poziv potprograma
   Primer:

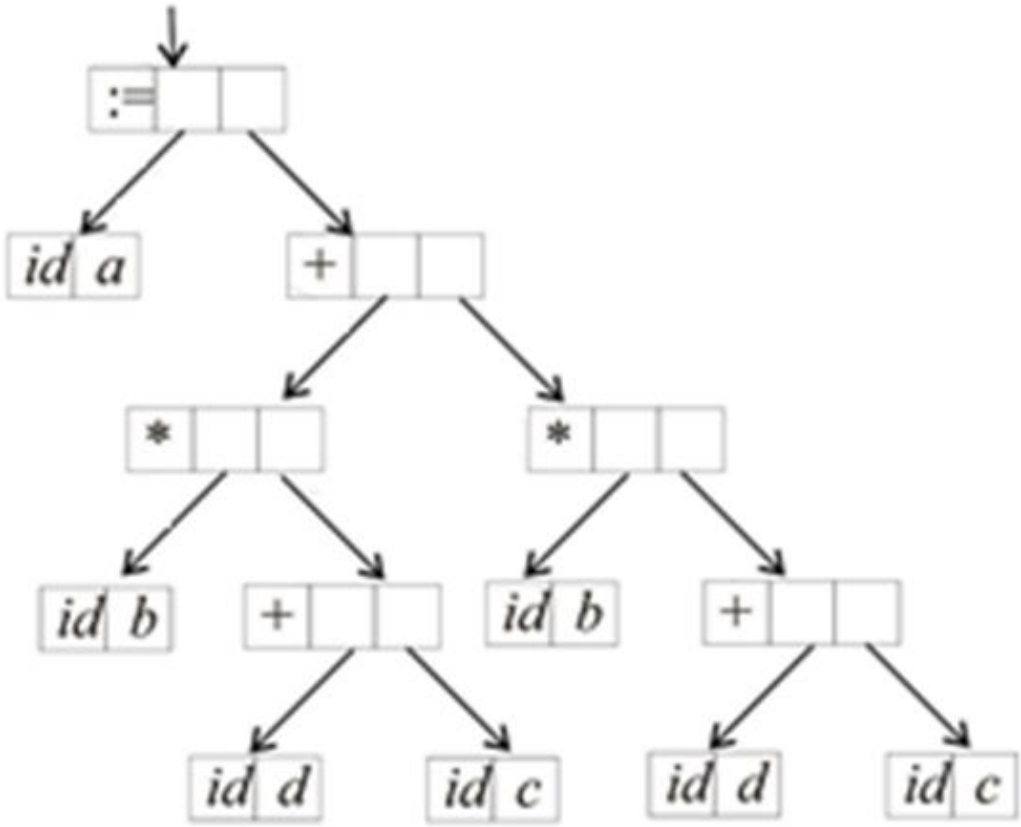
   ```
   param x1
   param x2
   param xn
   call p,n
   ```

7. return povratak iz potprograma
8. x := fcall p,n
9. return x povratak iz funkcijskog potprograma
10. Dodela sa indeksiranim promenljivama oblika: x := y [i] i x [i] : = y

Troadresni medjukod generisan na osnovu apstraktnog sintaksnog stabla

$a:=b*(d+c)+b*(d+c)$

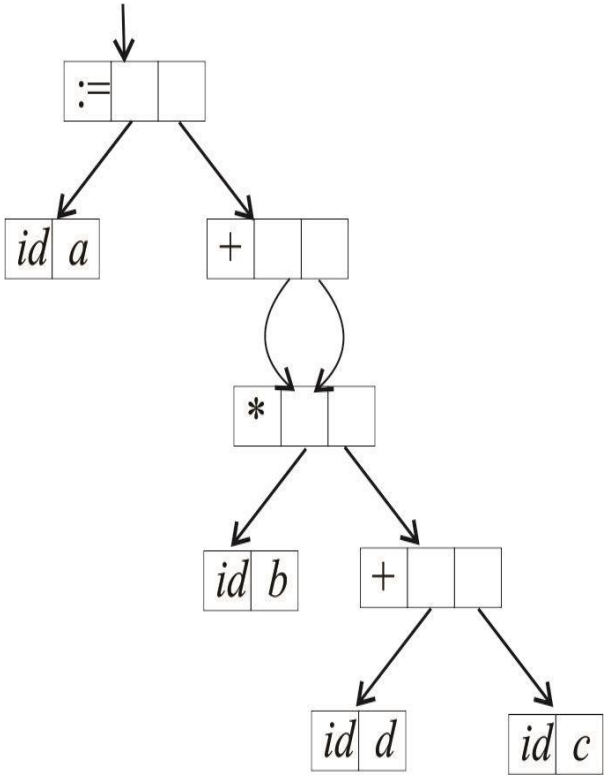
$t1 := d + c$
$t2 := b * t1$
$t3 := d + c$
$t4 := b * t3$
$t5 := t2 + t4$
$a := t5$



Troadresni medjukod generisan na osnovu sintaksnog dijagrama

$a:=b*(d+c)+b*(d+c)$

$t1 := d + c$
$t2 := b * t1$
$t3 := t2 + t2$
$a := t3$



Načini za memorisanje troadresnog međukoda

- Pomoću uređenih četvorki (*Quadruples*)
- Pomoću uređenih trojki (*Triples*)
- Pomoću uređenih trojki sa indirektnim ardesiranjem (*Indirect triples*) – umesto naredbi pamte se pokazivači na naredbe – pri promeni redosleda instrukcija menjaju se samo pokazivači

Memorisanje troadresnog međukoda pomoću uređenih četvorki (*Quadruples-a)*

Naredba	Quadruples	Opis
x := y binop z	Op | x | y | z	binop je binarni operator: +,-,*,/, and, or <, <=, =, >=, >,<> Op ∈ {PLUS, MINUS, TIMES, DIV, LT, LE, EQ, GE, GT, NE}
x := unop y	Op | x | y | -	unop je unarni operator: – , NOT Op ∈{NEG, NOT}
x := y	ASSIGN | x | y | -	Kopiranje vrednosti y u x
L:	LABEL | L | - | -	Oznaka (labela)
goto L	GOTO | L | - | -	Naredba bezuslovnog skoka
if x goto L	IF | L | x | -	Naredba uslovnog skoka

Memorisanje troadresnog međukoda pomoću uređenih četvorki (*Quadruples-a)*

Naredba	Quadruples	Opis
param x	PAR | - | *x* | -	Definisanje parametara potprograma
call p,n	CALL | - | *p* | *n*	Poziv potprograma p sa n argumenata
x := fcall f,n	FCALL | *x* | *f* | *n*	Poziv funkcijskog potprograma
Return	RET | - | - | -	Povratak iz potprograma
return x	FRET | - | *x* | -	Povratak iz funkcijskog potprograma kada se kao rezultat vraća x
x := y[k]	LDAR | *x* | *y* | *k*	Preuzimanje vrednosti promenljive sa indeksom
x[k] := y	STAR | *x* | *k* | *y*	Dodeljivanje vrednosti promenljivoj sa indeksom

Memorisanje troadresnog međukoda - Primer

Međukod

$t1 := d + c$
$t2 := b * t1$
$t3 := d + c$
$t4 := b * t3$
$t5 := t2 + t4$
$a := t5$

Međukod memorisan pomoću Quadriples-a

	op	rez	arg1	arg2
0	+	t1	D	C
1	*	t2	B	t1
2	+	t3	D	C
3	*	t4	B	t3
4	+	t5	t2	t4
5	=	a	t5	

Međukod memorisan pomoću Triples-a

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

Memorisanje troadresnog međukoda - Primer

Međukod

$t1 := d + c$
$t2 := b * t1$
$t3 := d + c$
$t4 := b * t3$
$t5 := t2 + t4$
$a := t5$

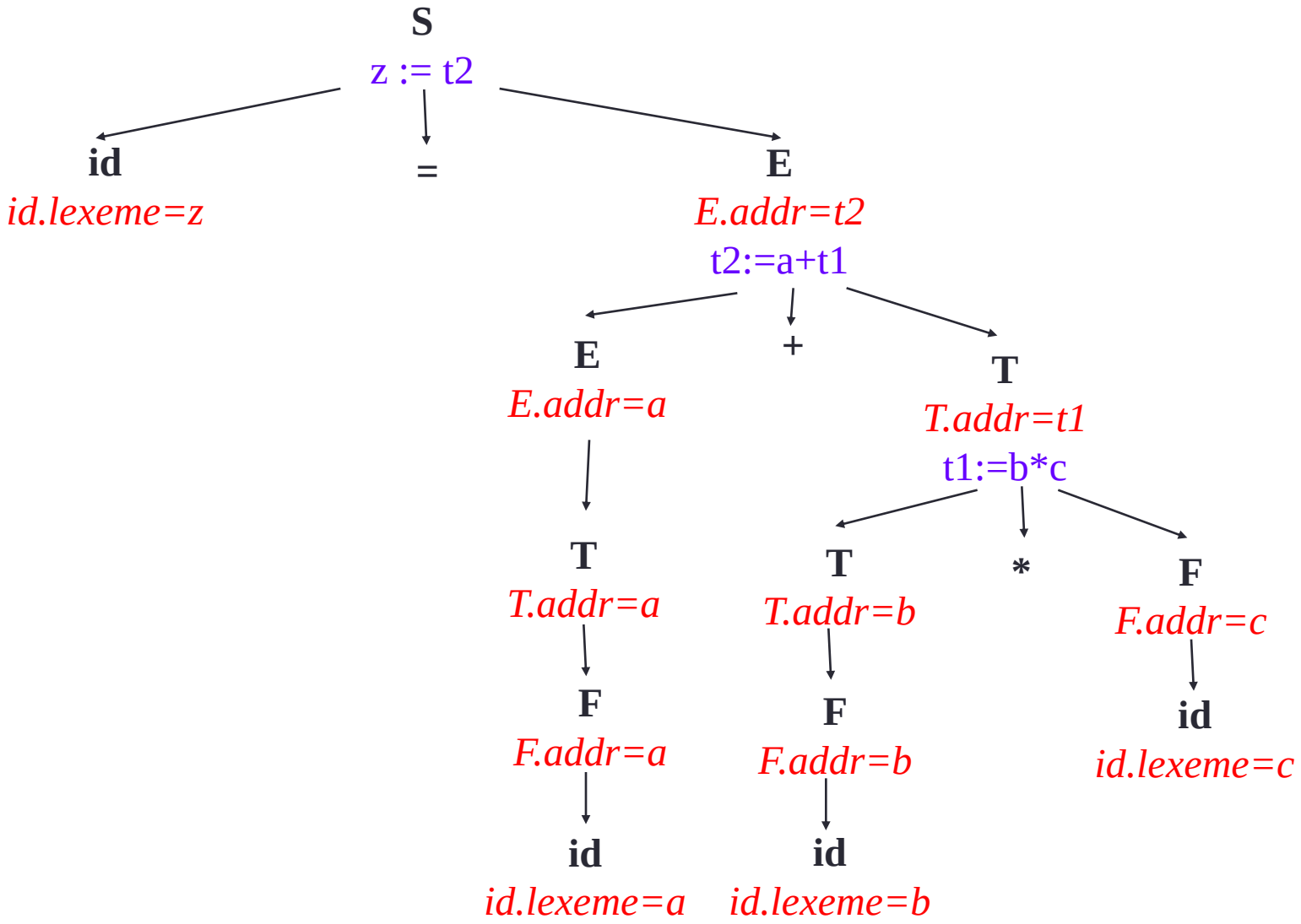
Međukod memorisan pomoću Indirect triples-a

	Naredbe		Op	arg1	arg2
35	(0)		0	+	d	**c**
36	(1)		1	*	b	**(0)**
37	(2)		2	+	d	**c**
38	(3)		3	*	b	**(2)**
39	(4)		4	+	(1)	**(3)**
40	(5)		5	=	a	**(4)**
	...			...		

Semantička pravila za generisanje međukoda naredbe dodeljivanja

Pravila gramatike	Semantička pravila
S →id = E;	{gen(top.get(id.lexeme) '=' E.addr)}
$E \rightarrow E_1 + T$	{E.addr = new Temp ()} {gen(E.addr '=' E1.addr '+' T.addr)}
E→ T	{E.addr = T.addr}
$T \rightarrow T_1 * F$	{E.addr = new Temp ()} {gen(E.addr '=' T1.addr '*' F.addr)}
T→F	{T.addr = F.addr}
F→ (E)	{F.addr = E.addr}
F→ id	{F.addr = top.get(id.lexeme)}
F→cons	{F.addr = top.get(cons.lexval)}

Notirano sintaksno stablo za generisanje troadresnog međukoda nardbe z = a + b * c



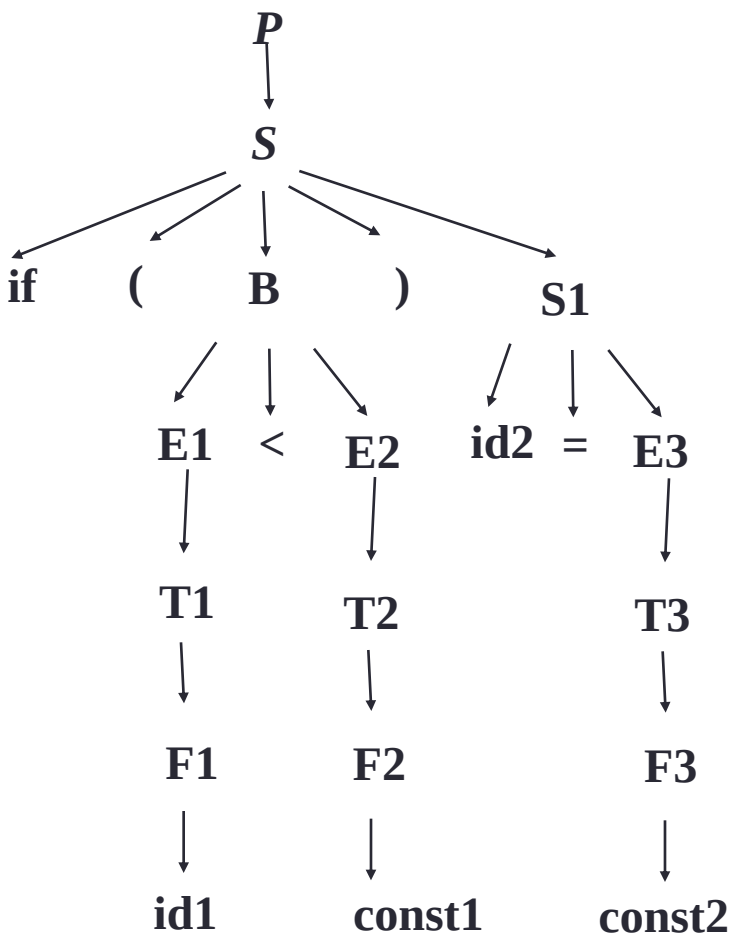
Semantička pravila za generisanje međukoda upravljačkih struktura

Pravila gramatike	Semantička pravila
P→ S	S.next = newlabel() P.code = S.code || label(S.next)
S→ assign	S.code = assign.code
S→if (B) S1	B.true = newlabel() B.false = S1.next = S. next S.code = B.code || label (B.true) || S1.code
S→ if (B) S1 else S2	B.true = newlabel() B.false = newlabel() S1.next = S2.next = S.next S.code = B.code ||label(B.true) || S1.code ||gen('goto' S.next) || label (B.false) || S2.code
S→ while (B) S1	begin = newlabel() B.true = newlabel() B.false = S.next S1.next = begin S.code = label(begin) || B.code || label(B.true)|| S1.code || gen('goto' begin)
S→ S1 S2	S1.next = newlabel() S2.next = S.next S.code = S1.code || label(S1.next) || S2.code

Semantička pravila za generisanje međukoda logičkih izraza

Pravila gramatike	Semantička pravila
B→B1 || B2	B1.true = B.true B1.false = newlabel() B2.true = B.true B2.false = B.false B.code = B1.code || label(B1.false) || B2.code
B→B1 && B2	B1.true=newlabel() B1.false = B.false B2.true = B.true B2.false = B.false B.code = B1.code || label(B1.true) || B2.code
B→! B1	B1.true = B.false B1.false = B.true B.code = B1.code
B→E1 rel E2	B.code = E1.code || E2.code ||gen('if' E1.addr rel. E2.addr 'goto' B.true) ||gen ('goto' B.false)
B→true	B.code = gen('goto', B.true)
B→false	B.code = gen('goto', B.false)

Generisanje troadresnog međukoda nardbe *if (x<100) x=0;*



Čvor	Atributi i kod
F_1	F_1.addr = x
T_1	T_1.addr= x
E_1	E_1.addr = x
F_2	F_2.addr =100
T_1	T_2.addr= 100
E_1	E_2.addr = 100
B	B.code = if x<100 goto *B.true* goto *B.false*
F_3	F_3.addr = 0
T_3	T_3.addr = 0
E_3	E_3.addr = 0
S_1	S_1.code = ‘x := 0’
S	B.True = Lab1 B.False=S1.Next=S.Next S.Code = if x<100 goto Lab1 goto *B.false* Lab1 x:=0
P	S.Next = Lab2 B.False=S1.Next=S.Next P.Code = if x<100 goto Lab1 goto Lab2 Lab1 x:=0 Lab2