Boolean Expressions

$E \rightarrow E$ or $E \mid E$ and $E \mid$ not $E \mid (E) \mid$ id relop id $\mid$ true $\mid$ False

Translated in 2 ways,

a) numerical representation

b) control flow representation.

Numerical representation

1 → true, 0 → false. Expressions will be evaluated from left to right.

a or b and not c } three-address code

t_1 = not c

t_2 = b and t_1

t_3 = a or t_2

relational expression – if a < b then 1 else 0.

} three-address code

100 : if a < b goto 103

101 : t := 0

102 : goto 104

103 : t := 1

104 :

Translation scheme using numerical representation for booleans.

$E \rightarrow E_1$ or E_2 { E.place := newtemp;
emit (E.place ':=' E_1.place 'or' E_2.place)}.

$E \rightarrow E_1$ and E_2

$E \rightarrow$ not E_1 { E.place := newtemp;
emit (E.place ':=' 'not' E_1.place)}

$E \rightarrow (E_1)$ { E.place := E_1.place }

$E \rightarrow id_1$ relop id_2 { E.place := new temp;
emit ('if' id_1.place relop.op id_2.place 'goto' nextstat+3);
emit (E.place ':=' '0');
emit ('goto' nextstat+2);
emit (E.place ':=' '1')}

$E \rightarrow$ true { E.place := newtemp;
emit (E.place ':=' 1)}

control flow representation

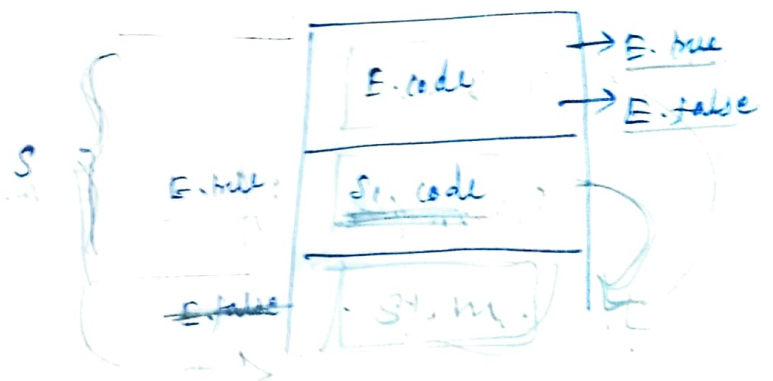
$S \rightarrow$ if E then S_1

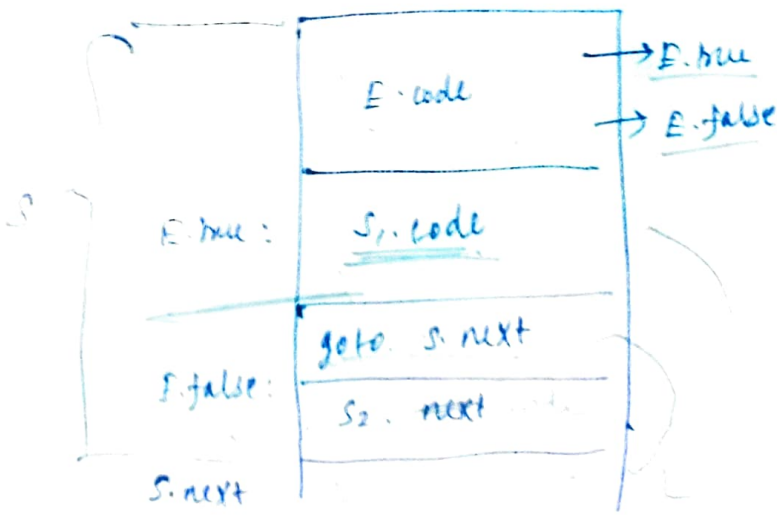
if E then S_1 else S_2

while E do S_1

Syntax-directed definition for flow of control statements.

production	semantic rules
$S \rightarrow$ if E then S_1	E.true := newlabel; E.false := S.next; S_1.next := S.next; S.code := E.code || gen(E.true ':') || S_1.code





$S \rightarrow$ if E then S_1 else S_2

E.true := newlabel;

E.false := newlabel;

S_1.next := S.next;

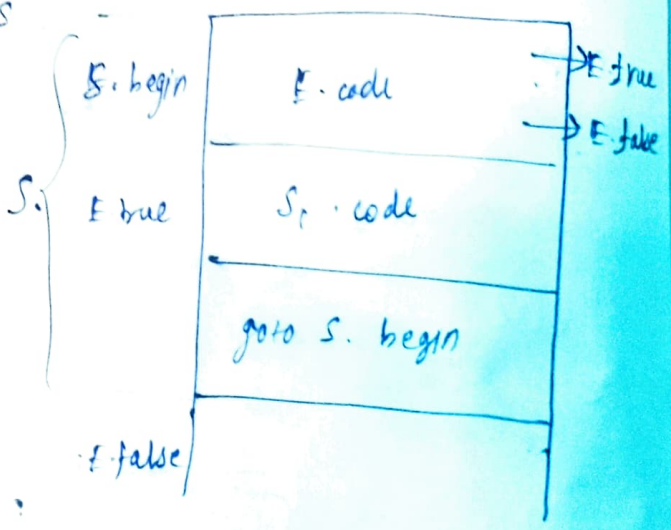
S_2.next := S.next;

S.code := E.code || gen(E.true ':')

S_1.code || gen('goto' S.next)

|| gen(E.false ':') || S_2.code

$S \rightarrow$ while E do S



S.begin := newlabel;

E.true := newlabel;

E.false := S.next;

S_1.next := S.begin;

S.code := gen(S.begin ':') || E.code

gen(E.true ':') || S_1.code

gen('goto' S.begin)

Control Flow Translation of Boolean Expression:

SDD to produce three address code for booleans.

$E \rightarrow E_1$ or E_2

0 0	0
0 1	1
1 0	1
1 1	1

$E_1.true := E.true$

$E_1.false := newlabel$

$E_2.true := E.true$

$E_2.false := E.false;$

$E.code := E_1.code \,||\, gen(E_1.false:) \,||\, E_2.code.$

$E \rightarrow E_1$ and E_2

0 0	0
0 1	0
1 0	0
1 1	1

$E.true := newlabel;$

$E_1.false := E.false;$

$E_2.true := E.true;$

$E_2.false := E.false;$

$E.code := E_1.code \,||\, gen(E_1.true:) \,||\, E_2.code.$

$E \rightarrow not\ E_1$

$E_1.true := E.false$

$E_1.false := E.true;$

$E.code := E_1.code.$

$E \rightarrow (E_1)$

$E_1.true := E.true;$

$E_1.false := E.false;$

$E.code := E_1.code$

$E \longrightarrow id_1\ relop\ id_2$ E.code := gen('if' id_1.place relop.op id_2.place 'goto' E.true) || gen('goto' E.false)

$E \longrightarrow$ true E.code := gen('goto' E.true)

$E \longrightarrow$ false E.code := gen('goto' E.false).