CD Assignment

CS21 B2045
T. Lakshmi Srinivas

(1) Consider following grammer.

$S \to S*E$ $\quad E \to F$

$S \to E$ $\quad F \to id$

$E \to F+E$

Consider following LR(0) items corresponding to grammer above.

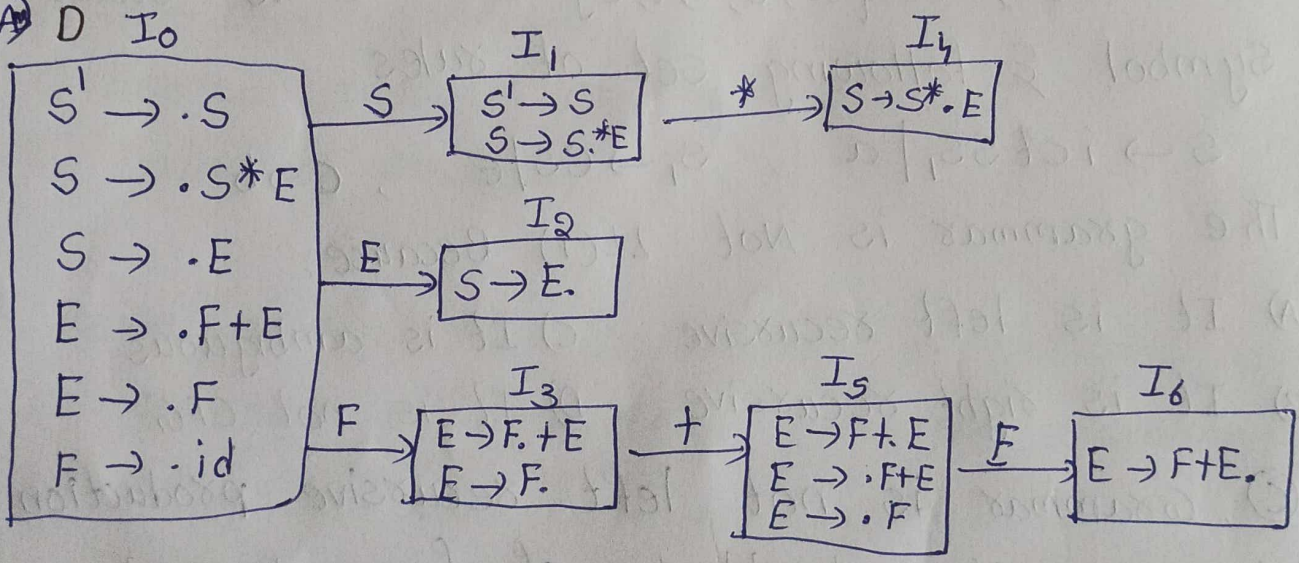
(i) $S \to S*.E$ (ii) $E \to F.+E$ (iii) $E \to F.+E$

Given items above, which two of them will appear in same set in canonical sets of items for grammer?

(A) (i) & (ii) (B) (i) & (iii)

(C) (ii) & (iii) (D) None of the above.

A D I_0



we can see that given things belong to (I_3, I_4, I_5).

CS21B2045
T. Lakshmi Srinivas

2) P: Every regular grammer is LL(1)
Q: Every regular sets has LR(1) grammer
which of the following is True?

A) Both P & Q are true.
B) P is true & Q is false
C) P is false & Q is true.
D) Both P & Q are false.

Ans (C)

A Regular grammar can also be ambiguous, we have regular grammar which is unambiguous so it can parse by LR Parser.

3) Consider grammar with non-terminals $N = \{S, C, S_1\}$ terminals, $T = \{a, b, i, t, e\}$ with S as start Symbol & following set of rules.

$S \rightarrow ictSS_1 | a$, $S_1 \rightarrow eS | \epsilon$, $C \rightarrow b$

The grammar is Not LL(1) Because:

A) It is left recursive
B) It is right recursive
C) It is ambiguous
D) It is not CFG

Ans (C), Grammar is not left recursive production because it should be of form $A \rightarrow Ab$.
→ Right recursive not related to LL(1).
→ Grammar is context free.

Lexical analysis.

CS21B2045
T. Lakshmi Srinivas

Q) The number of tokens in the given C statement.

```
Printf("Pt=%d, &Pt=%x", Pt, &Pt);
```

A) 10 tokens

In C we have 6 tokens.

Identifiers, Constants, keywords, operators, string literals & other separators.

```
Printf, C, " Pt=%d, &Pt=%x"
, Pt , & Pt ) ;
```

5) In any compiler, we can recognize keywords of a language during —

A) Parsing of the Program
B) The code generation.
C) Lexical analysis of the program.
D) Dataflow analysis.

Sol) (C)

Lexical analysis is the method of changing over a grouping of characters into an arrangement of tokens. A token can be a keyword.

CS21B2045.

6) The output of any lexical analyzer is?

A) A parse tree.

B) Machine code.

C) Intermediate code.

D) A stream of tokens.

Sol) (D)

The lexical analysis produces a stream of tokens as output, consisting of identifiers, keywords, separators, operators & literals.

7) The lexical analyzer uses the given patterns for recognizing three tokens T_1, T_2, & T_3, over the alphabets a, b, c. T_1: a?(b|c)*a & T_2: b?(a|c)*b & T_3: c?(b|a)*c. If the analyzer processes the string "bbaacabc", which of the below is the sequence of tokens it outputs?

A) $T_1\ T_2\ T_3$

B) $T_1\ T_1\ T_3$

C) $T_2\ T_1\ T_3$

D) $T_3\ T_3$

Sol) (D), we can rewrite the token as: T_1: (b+c)*a and T_2: (a+c)*b + b(a+c)*b and T3: (b+a)*c + c(b+a)*c.

The given string is "bbaacabc" the longest matching prefix is "bbaac" Hence $T_3\ T_3$.

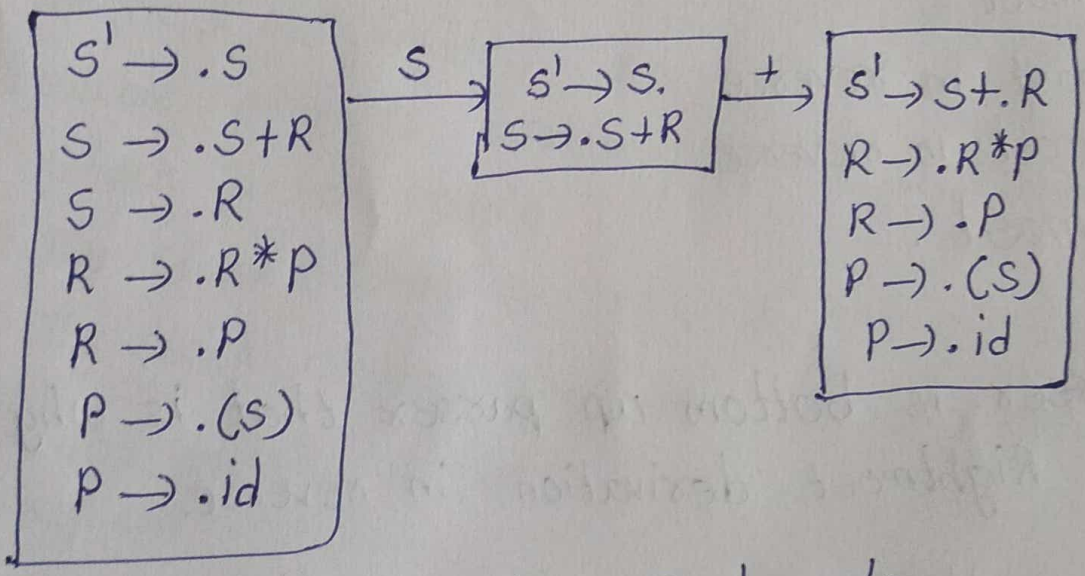
CS21B2045

8) Consider the augmented grammar with {+, *, (,), id} as set of terminals.

$S' \rightarrow S$
$S \rightarrow S+R \mid R$
$R \rightarrow R*P \mid P$
$P \rightarrow (S) \mid id$

If I_0 is the set of two LR(0) items {[S' → S.], [S → S.+R]}, then goto (closure(I_0), +) contains exactly __ items.

8) 5.



Goto (I_0, +) will have 5 elements.

9) Consider the following grammar.

$S \rightarrow aSB \mid d$
$B \rightarrow b$

The no. of reduction steps taken by a bottom-up parser while accepting the string aaadbbb is.

CS21B2045

Sol) $S \rightarrow aSB$

$\rightarrow aaSBB$ [$S \rightarrow aSB$]

$\rightarrow aaaSBBB$ [$S \rightarrow aSB$]

$\rightarrow aaadBBB$ [$S \rightarrow d$]

$\rightarrow aaadbBB$ [$B \rightarrow b$]

$\rightarrow aaadbbB$ [$B \rightarrow b$]

$\rightarrow aaadbbb$ [$B \rightarrow b$]

Total 7 steps required.

10) Which one of the following kinds of derivation is used by LR parsers?

A) Rightmost

B) Leftmost in reverse

C) Rightmost in reverse

D) Leftmost.

A) (C)

LR parser is bottom up parser that is why it uses Rightmost derivation in reverse.

(11) Consider the grammar given below:

$S \rightarrow Aa$

$A \rightarrow BD$

$B \rightarrow b/\epsilon$

$D \rightarrow d/\epsilon$

Let a, b, d & $ be indexed as follows:

a	b	d	$
3	2	1	0

Compute the Follow set of the non-terminal B & write the index values for the symbols in the Follow set in the descending order.

CS21B2045

A) Correct answer is 31 to 31

Follow(B) = First(D) ∪ Follow(A)

∵ when D is ϵ then Follow(B) = Follow(A)]

∴ Follow(B) = {d} ∪ {a} = {a, d}

As a=3 & d=1 then descending order=31

(12) Which one of the following is TRUE at any valid state in shift-reduce parsing?

A) Viable prefixes appear only at the bottom of the stack & not inside.

B) viable prefixes appear only at the top of the stack & not inside.

C) The stack contains only a set of viable prefixes

D) The stack never contains viable prefixes.

Ans © The stack contains only a set of viable prefixes.

(13) Consider the grammar defined by the following production rules, with two operators * and +

S → T*P , T → u | T*u

P → Q+P | Q , Q → id , u → id.

Which one of the following is True?

Ans + is right associative while * is left associative.

CS21B2045

Semantic Analysis

(14) what is true about syntax directed definitions.

A) Syntax Dir. Def + Semantic rules = CFG

B) Syntax Directed Def + CFG = Semantic rules.

C) CFG + Semant rules = Syntax Directed Definitions.

D) None of the above.

A) (C)

CFG + Semantic rules = Syntax Directed defn.

(15) which of the following error is expected to recognize by Semantic analyzer.

A) Type mismatch

B) Undeclared variable.

C) Reserved identifier misuse

D) all of the above.

A) (D)

we have mentioned some of the semantics errors that the semantic analyzer is expected to recognize: Type mismatch. Undeclared variable. Reserved identifier misuse. Multiple declaration of variable in a scope. Accessing an out of scope variable. Actual and formal parameter mismatch.

CS21B2045

16) write syntax directed definitions for following grammar to add type of each identifier in semantic analysis.

D → TL
T → int
T → real
L → L, id
L → id

Ans

Prod. Rule.	Semantic action
D → TL	L.in = T.type
T → int	T.type = integer
T → real	T.type = real
L → L, id	L.in = Lin Enter_type(id, entry, L.in)
L → id	Enter_type(id, entry, L.in)

(17) what is meant by compositional semantics
A) Determining the meaning
B) Logical connectives
C) Semantics.
D) All of the above.

Ans (A), Compositional semantics is the process of determining the meaning of P^*Q from $P, Q \in \Sigma^*$.

(18) Incompatable types work with the
a) Syntax tree (b) semantic analyser (c) code optimizer
d) Lexical analyser.

Ans (b) semantic analyser.

CS21B2045

Intermediate code generation

(19) The least no. of temporary variables require to create a 3 address code in static single assignment form for expression.

"$a = b*d - c + b*e - c$" is

A) 3 B) 4 C) 5 D) 6

Ans (B)

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

$temp_1 = b*d$

$temp_2 = b*e$

$temp_3 = temp_1 + temp_2$

$temp_4 = temp_3 - c$

$a = temp_4 - c$

∴ No. of temporary variables is 4.

(20) Consider the following intermediate program in three address code

$p = a - b$

$q = p*c$

$p = u*v$

$q = p+q$

which one of the following corresponds to a static single assignment form of the above code?

CS21B2045

(A)
$P_1 = a - b$
$q_1 = P_1 * c$
$P_1 = u * v$
$q_1 = P_1 + q_1$

(B)
$P_3 = a - b$
$q_4 = P_3 * c$
$P_4 = u * v$
$q_3 = P_4 + q_4$

(C)
$P_1 = a - b$
$q_1 = P_2 * c$
$P_3 = u * v$
$q_2 = P_4 + q_3$

(D)
$P_1 = a - b$
$q_1 = P_1 * c$
$P_2 = u * v$
$q_3 = P + q$

Ans (B)
$P_1 = a - b$
$q_1 = P_1 * c$
$P_2 = u * v$
$q_2 = P_2 + q_1$

(21) Consider the intermediate code given below.

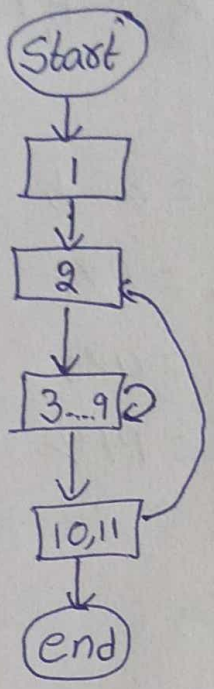
(i) $i = 1$
(ii) $j = 1$
(iii) $t_1 = 5 * i$
(iv) $t_2 = t_1 + j$
(v) $t_3 = 4 * t_2$
(vi) $t_4 = t_3$
(vii) $a[t_4] = -1$
(viii) $j = j + 1$
(ix) if $j \leq 5$ goto (3)
(x) $i = i + 1$
(xi) if $i < 5$ goto (2)

CS21B2045

The no. of nodes & edges in the control-flow-graph constructed for the above code, respectively are

A) 5 & 7 B) 6 & 7 C) 5 & 5 D) 7 & 8

Ans (B)



6 nodes & 7 edges

(22) which is/are not Intermediate representation of Source program?

a) Three address code b) AST c) CFG d) Symbol table

Ans d), Symbol table.

(23) which languages necessarily need heap allocation in the runtime environment?

A) Those that Support recursion

B) Those that use dynamic Scoping

C) Those that allow dynamic data structures.

D) Those that use global variables.

Ans (C) Those that allow dynamic data structures

24) The attributes of 3 arithmetic operator in some programming languages are given below.

operator	Precedence	Associativity
+	High	Left
-	medium	Right
*	Low	Left

The value of expression 2-5+1-7*3 in this is.

Ans

2-5+1-7*3

= 2-4-7*3

= 2-(-3)*3

= 2+3*3

= 5*3

= 15

25) For a C program accessing X[i][j][k] the following intermediate code is generated by a compiler. Assume that the size of an integer is 32 bits and size of character is 8 bits.

$t_0 = i * 1024$

$t_1 = j * 32$

$t_2 = k * 4$

$t_3 = t_1 + t_0$

$t_4 = t_3 + t_2$

$t_5 = X[t_4]$

which is correct?

Ans

$t_5 = X[t_4]$

$= X[i*1024 + j*32 + k*4]$

X is declared as int.[32][32][8]

26) consider u*v+a-b*c. which of the following corresponds to SSA from the expressions.

A) $x_1 = a-b$ $y_1 = P*C$ $x_2 = u*v$ $y_2 = P+q$

B) $x_1 = a-b$ $y_1 = x_2*C$ $x_3 = u*v$ $y_2 = x_4+y_3$

C) $x_1 = a-b$ $y_2 = x_1*C$ $x_2 = u*v$ $y_3 = x_2+y_2$

D) $P = a-b$ $q = P*C$ $P = u*v$ $q = P+q$

Ans (C) as SSA 1) a var can't be used more than once in LHS. a 2) A var should be initialised atmost once.

Code-optimization

27) Substitution of values for names is done in

A) Local optimization B) Loop optimization.

C) constant Folding D) Strength Reduction.

Ans (C), Constant Folding.

28) Consider the following ANSI C code segment:

```
z = x + 3 + y->f1 + y->f2
for (i=0; i<200; i=i+2)
{
    if (z>i)
    {
        p = p+x+3;
        q = q+y->f1;
    }else
    {
        p = p+y->f2;
        q = q+x+3;
    }
}
```

Assume that the variable y points to a struct containing two fields f_1 & f_2, and the local variables x, y, z, p, q & i are allotted registers. CSE optimization is applied on the code.

A) 403 & 102

B) 203 & 2

C) 303 & 102

D) 303 & 2

Ans) whether we take if or else block we get 2 additions the loop runs exactly 100 times, so total additions 303

we only do two de-reference outside the for-loop

so, total de-references = 2

29) which of the following code replacements is an illustration of operator strength reduction?

A) Replace P+P by 2*P or replace 3+4 by 7.

B) Replace P*32 by P<<5

C) Replace P*0 by 0

D) Replace (P<<4) - P by P*15

Ans B) Replace P*32 by P<<5

30) In compiler technology reduction in strength means.

A) Replacing run time compilation by complile time.

B) Removing loop invariant computation

C) Removing common sub-expression.

D) Replacing costly operation with a cheaper one.

Ans D.

31) match the following.

I	II
a) Lexical analyzer	i) Graph coloring
b) Parsing	ii) DFA minimization.
c) Register allocation	iii) Post-order traversal
d) Expression evaluation	iv) Production tree.

a → ii b → iv

c → i d → iii

CS21B2045

32) which optimization technique is used to reduce the multiple jumps?

A) Latter optimization technique

B) Peephole optimization technique.

C) Local optimization technique

D) Code optimization technique.

Ans B) Peephole optimization technique

(33) A Linker reads four modules whose length are 200, 800, 600, 500 respectively. If they are loaded in that order, what are the relocation constants:

A) 0, 200, 500, 600 (B) 0, 200, 1000, 1600.

(C) 200, 500, 600, 800 (D) 200, 700, 1300, 2100.

Ans (B)

module	Relocation Base.	Limit length
1	0	200
2	200	800
3	1000	600
4	1600	500.

CS21B2045

(34) which of the following is False about peephole optimization?

a) It's applied to small part of code
b) It can be used to optimise intermediate code
c) To get best out of this, it has to be applied repeatedly.
d) It can be applied to code that isn't contiguous.

Ans (d).

Target code Generator.

35) which languages necessarily need heap allocation in runtime environment?

A) Those that support recursion.
B) Those that use dynamic scoping.
C) Those that allow dynamic data variables.
D) Those that use global variables.

Ans (C).

(36) In a resident-OS computer, which of the following systems must reside in main memory under all situation?

A) assembler B) Linker C) Loader D) compiler.

Ans C) Loader. is permanently resident in memory, although some O.S.

(37) which of the following comparisons b/w static & dynamic type checking incorrect.

A) Dynamic type checking slows down the execution.
B) Dynamic type checking offers more flexibility to programers
C) DTC ~~offers~~ may cause failure in runtime
D) unlike static type checking dynamic type checking is done during compilation.

Ans D).

(38) S_1: Sequence of procedure calls corresponds to preorder traversal of activation tree.

S_2: Sequence of procedure returns corresponds to post order traversal of activation tree.

which is True?

a) $S_1 \rightarrow$ True, $S_2 \rightarrow$ False b) $S_1 \rightarrow$ False, $S_2 \rightarrow$ True

c) $S_1 \rightarrow$ True, $S_2 \rightarrow$ True c) $S_1 \rightarrow$ False, $S_2 \rightarrow$ False.

Ans (c) performing procedure calls → parent calls → Pre order
In returning procedure calls → return child first → Postorder

39) one of the purposes of using intermediate code in compilers is to.

A) make passing & semantic analysis simpler.
B) Improve error recovery & error reporting

C) Increasing chance of using machine Independent code optimiser in other compilers.

D) Improve the register Allocation.

Ans (C)

40) A computer has only 2 registers & 3 opcodes

Case-1:

$t_1 = a + b$

$t_2 = c + d$

$t_3 = e - t_2$

$t_4 = t_1 - t_2$

Case-2:

$t_2 = c + d$

$t_3 = e - t_2$

$t_1 = a + b$

$t_4 = t_1 - t_2$

Final value of computation has to reside in memory. which one is better in memory access & how many Mov instruction.

A) Case 2, 2 B) Case 2, 3 C) Case 1, 2 D) Case 1, 3

Ans (D) case 1, 3

Case-1:

```
Mov a, t1
add b, t1
Mov c, t2
add d, t2
Sub e, t2
Sub t1, t2
Mov t2, a
```

3 moves

Case-2: => 4 mov operations

```
Mov c, t2
Add d, t2
Mov t2, t3
Sub e, t2
Mov a, t1
Add b, t1
Sub t1, t3
Mov c, t3.
```