B.Tech Third Year End Semester Examination

Department: Computer Science and Engineering

Course Name: Compilers **Code: CS 346**

Full Marks-100 **Time: 3 hours**

Answer ALL the Questions

Make reasonable assumptions as and whenever necessary. You can attempt the questions in any sequence, but the answers to all the components of any particular question should appear together. Marks will be deducted if this is not followed properly.

1. Consider the following code fragment (1-18 are the instruction numbers)

   1: i = 0; 2: a = 1; 3: b = i * 4; 4: c = a + b; 5: L1: if i > n goto L2; 6: c = a + 1; 7: i = i +1; 8: b = i * 4; 9: if c <= p goto L3; 10: e = 1; 11: c = e – b; 12: a = e; 13: goto L4;; 14: L3: d = 2; 15: c = d + b; 16: a = d; 17: L4: goto L1; 18: L2: return

a) Determine the live ranges for the variables i, a, b, c, d, e, p (***use the instruction numbers to denote the live range***)
b) Draw the interference graph using the simplest definition of interference where the edges (or, webs) include the live range in terms of the line numbers where an access to a variable (either a read or a write operation) occurs.
c) Determine the minimum number of registers needed (without spilling) using the graph coloring algorithm.
d) Assuming that you were short of one register, which register(s) would you spill and at which points in the program would you insert the spill code? Explain.

20

2. (a). Rewrite the following SDT: A→A {a} B | A B {b} | 0; B→B {c} A | B A {d} | 1

so that the underlying grammar becomes non-left recursive. Here a, b, c and d are actions, and 0 and 1 are terminals.
(b). Generate three-address code for the following instructions assuming that all array elements are integers taking four bytes each.

a. X= a[i]+1
b. X=a/(b+c)-d*(e+f)

10+10

3. (a). Consider the following set of basic blocks:

B1={d1:a=1, d2: b=2}, **B2**={d3:c=a+b, d4:d=c-a}, **B3**={d5:d=b+d}, **B4**={d6:d=a+b, B7:e=e+1}, **B5**={d8:b=a+b, d9:e=c-a}, **B6**: {d10:a=b*d, d11:b=a-d}

The CFG has the following set of edges: B1→B2; B2→B3; B3→B4, B5; B4→B3; B5→B6, B2.

(96)

Compute the **GEN**, **KILL**, **IN** and **OUT** sets for the reaching definition analysis.

(b). Explain with examples "Peephole optimization" and "Constant propagation" with respect to code optimization.

14+6

4. What is meant by backpatching? Design a syntax directed translation scheme for backpatching (use **attributes**: *truelist, falselist, instr* etc. and **functions**: *makelist, merge, backpatch* etc.) the following boolean expressions during bottom-up evaluation.

$E \rightarrow E_1$ or E_2| E_1 and E_2 | not E_1| (E_1) | id_1 relop id_2| true| false

Translate the expression (a==b || c==d) && e==f. Show the values of *truelist* and *falselist* for each sub-expression.

3+17

5. Choose the correct answer **10*2=20**

(i). In a bottom-up evaluation of a syntax directed definition, inherited attributes can (a) always be evaluated; (b) be evaluated if the definition is L-attributed; (c) be evaluated only if the definition has synthesized attributes; (d) never be evaluated.

(ii). Which of the following is NOT an advantage of using shared, dynamically linked libraries as opposed to using statically linked libraries?
(a) Smaller sizes of executable; (b) Lesser overall page fault rate in the system; (c) Faster program start-up; (d) Existing programs need not be re-linked to take advantage of newer versions of libraries.

(iii). Consider a program P that consists of two source modules M1 and M2 contained in two different files. If M1 contains a reference to a function defined in M2, the reference will be resolved at (a). Edit-time; (b). Compile-time; (c). Link-time and (d). Load-time.

(iv). Consider the grammar rule $E \rightarrow E_1 - E_2$ for arithmetic expressions. The code generated is targeted to a CPU having a single user register. The subtraction operation requires the first operand to be in the register. If E_1 and E_2 do not have any common sub expression, in order to get the shortest possible code : (a) E_1 should be evaluated first ; (b) E_2 should be evaluated first; (c) Evaluation of E_1 and E_2 should necessarily be interleaved; (d) Order of evaluation of E_1 and E_2 is of no consequence.

(v). Some code optimizations are carried out on the intermediate code because (a) They enhance the portability of the complier to other target processors; (b) Program analysis is name accurate on intermediate code than on machine code; (c) The information from data flow analysis cannot otherwise be used for optimization ; (d) The information from the front end cannot otherwise be used for optimization.

(vi). An LALR(1) parser for a grammar can have shift-reduce (S-R) conflicts if and only if
(a) The SLR(1) parser for G has S-R conflicts ; (b) The LR(1) parser for G has S-R conflicts ; (c) The LR(0) parser for G has S-R conflicts ; (d) The LALR(1) parser for G has reduce-reduce conflicts

(vii). In a compiler, keywords of a language are recognized during (a) parsing of the program; (b) the code generation; (c) the lexical analysis of the program; (d) dataflow analysis

(viii). Which of the following statements is FALSE? (a) In statically typed language, each variable in a program has a fixed type; (b) In up-typed languages, values do not have any types; (c) In dynamically typed languages, variables have no types; (d) In all statically typed languages, each variable in a program is associated with values of only a single type during the execution of the program

(ix). Consider the translation scheme shown below

$S \rightarrow TR$; $R \rightarrow + T$ {print ('+');}$R \mid \varepsilon$; $T \rightarrow$ num {print (num.val);}

Here "num" is a token that represents an integer and "num.val" represents the corresponding integer value. For an input string '9 + 5+ 2', this translation scheme will print

(a). 9+5+2; (b). 95+2+; (c). 952++; (d). ++952

(x). The grammar $A \rightarrow AA \mid (A) \mid \varepsilon$ is not suitable for predictive-parsing because the grammar is

(a). ambiguous; (b) left-recursive ; (c) right-recurisve; (d) an operator-grammar

(97)