COMP 36512

Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Compilers

Date: Monday 4th June 2018

Time: 09:45 - 11:45

Please answer all Questions.

This is a CLOSED book examination

The use of electronic calculators is NOT permitted

[PTO]

1. a) For each item in column one, choose the best match for column two. Each item in column two should be used only once.

Column 1	Column 2
1. Abstract Syntax Tree	a. Code optimisation
2. Activation Record	b. Context Sensitive Analysis
3. Attribute Grammars	c. DFA minimisation
4. Chaitin's heuristic	d. Instruction scheduling
5. Copy propagation	e. Intermediate representation
6. Hopcroft's algorithm	f. Intermediate representation
7. List scheduling	g. Memory management
8. LR(1) item	h. Parsing
9. Static Single Assignment (SSA)	i. Register Allocation

(5 marks)

b) Explain briefly what a symbol table is. Your answer should mention when in compilation symbol table entries are normally created.

(3 marks)

c) Explain whether it is possible to write regular expressions to recognise:

(i) any even integer

(ii) any binary number which contains as many 0s as 1s

(iii) any real number which contains the same number of digits in both the integer part and the decimal part (e.g., 13.74 is such a number, but 3.14 is not)

(iv) any real number with an exponential part (e.g., 3.45E2)

(v) any string of lowercase letters in which the letters are in ascending lexicographic order (for example, amrz or delos are such strings but anno or agora are not) Justify your answers.

(5 marks)

d) An IP address consists of four integers each with a value between 0 and 255 (including 0 and 255), which are separated by a dot. For example, 34.0.123.54 is a valid IP address but 52.256.1.7 is not a valid IP address. Write a regular expression to recognise a valid IP address.

(4 marks)

e) Given a Non-Deterministic Finite Automaton (NFA) with several final states explain how to convert this to an NFA with only one final state.

(3 marks)

2. a) Consider the following context-free grammar for variable declarations in a Pascal like language:

1. Goal → Var_Decl
2. Var_Decl → var Decl_List
3. Decl_List → Decl ; Decl_List
4. Decl_List → Decl ;
5. Decl → Id_List : Id_Type
6. Id_List → Id_List , identifier_name
7. Id_List → identifier_name
8. Id_Type → integer
9. Id_Type → real
10. Id_Type → boolean
11. Id_Type → char

i) Derive a leftmost derivation for the string

```
var x,y:integer; z:boolean;
```

and show the corresponding parse tree. (6 marks)

ii) Transform this grammar so that it can be used to construct a top-down predictive parser with one symbol of lookahead. (6 marks)

b) Draw the control flow graph of the following code fragment.

```
L1:  if (m>n) goto L2
     m=m+a[n]
     n=n-1
     goto L1
L2:  if (s<m) goto L3
     s=s-a[n]
     n=n+1
     m=m+1
     goto L2
L3:  return
```

(4 marks)

c) Show how the expression `x-2*y` may be translated into an abstract syntax tree, one-address code, two-address code, and three-address code. (4 marks)

[PTO]

3. a) A certain C compiler performs the following optimisation:

```
/* before optimisation */     /*after optimisation */
for (i=1;i<1000;i++)            for (i=1;i<1000;i++)
  for(j=1;j<1000;j++)             for(j=1;j<1000;j++)
     a[j][i]+=1;                     a[i][j]+=1;
```

Why is this an optimisation? What would stop the compiler from applying this optimisation if the assignment statement of the original code was as follows?

```
a[j][i]+=a[j-1][i+1];
```

(5 marks)

b) The following piece of code makes use of 7 variables.

```
1. a = 1
2. b = a + 3
3. c = a + b
4. d = b + 5
5. e = c + 7
6. f = e + d
7. return f
```

Draw an interference graph and apply a graph colouring algorithm of your choice to find the smallest number of variables that may be used to write this piece of code. Show the resulting code. State any assumptions you make.

(6 marks)

c) Apply list scheduling to generate a schedule for the following basic block on a processor that can issue up to two instructions per cycle. Your answer should show the precedence graph, explain how you set priorities and show the schedule. Assume that all instructions have a latency of 1 cycle except `mult` that has a latency of 2 cycles.

```
1. mult r1, r1, 12
2. mov  r2, 1
3. mult r3, r3, 7
4. add  r4, r1, r2
5. add  r5, r2, 19
6. add  r6, r2, r3
7. add  r7, r5, 6
8. add  r8, r5, 4
```

(5 marks)

d) A critical aspect of list scheduling is the mechanism for setting priorities and breaking ties when several operations with the same priority are ready at the same cycle. Using as an example the schedule generated in c) above, demonstrate how different mechanisms for setting priorities and breaking ties can lead to different schedules.

(4 marks)

END OF EXAMINATION