COMP 36512

Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Compilers

Date: Monday 5th June 2017

Time: 09:45 - 11:45

Please answer any THREE Questions from the FIVE Questions provided

---

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. DFA minimisation
3. Best's Algorithm	c. Graphical intermediate representation
4. Bison	d. Instruction scheduling
5. Chaitin's heuristic	e. Memory management
6. Constant Folding	f. Parsing
7. Hopcroft's algorithm	g. Parsing
8. List scheduling	h. Register Allocation
9. LR(1) item	i. Register Allocation

(5 marks)

b) Give one example of: (i) a lexical analysis error; (ii) a syntax analysis error; (iii) a semantic analysis error.

(3 marks)

c) Assume that you have developed a basic compiler for a subset of the C language that generates code for certain Intel processors. Identify the components of the compiler that would need to be modified or enhanced to provide the following capabilities:

i) allow C++ (or Java) like comments, that is, //

ii) add the type complex to describe complex numbers

iii) support code generation for an Intel quad-core processor

iv) unroll loops as much as possible.

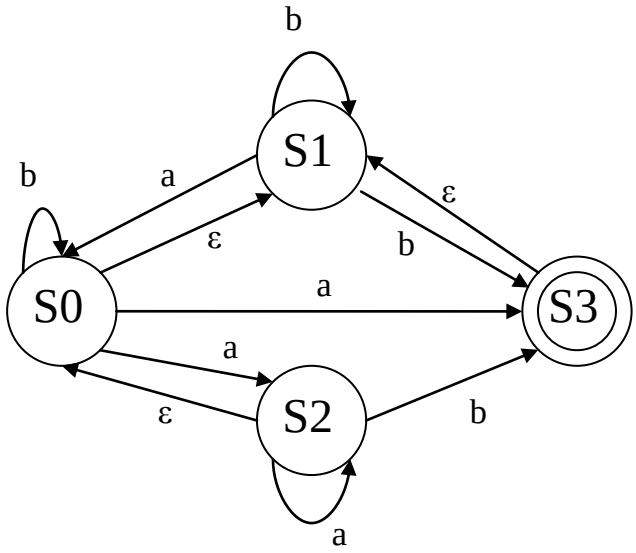
Justify your answers.

(8 marks)

d) The slice of a program with respect to a given statement, s, and a set of variables, consists of all statements of the program which might affect the value of these variables at point s. Let us assume that you are given an open-source C compiler for a certain processor and you are asked to use it as a basis to build a compiler, which, instead of generating machine code for this processor, produces slices of an input C program. Identify the compiler components that would need to be modified, could be removed, or need to be added to the C compiler you are given in order to produce the slice.

(4 marks)

2. Consider the following automaton. S0 is the start state and S3 is the final state.



a) Explain why this is a Non-Deterministic Finite Automaton (NFA) . (2 marks)

b) A string of length 3 that is clearly acceptable is aab. Find all strings of length 3 that are acceptable. (3 marks)

c) Apply the subset construction algorithm to convert this NFA to a DFA. Show the DFA. (5 marks)

d) Apply Hopcroft's algorithm to produce a minimised DFA. Show the DFA. (4 marks)

e) Use the DFA produced to write a regular expression that describes the grammar represented by the automaton above. Describe what this regular expression represents in plain English. (4 marks)

f) Explain how your answers to c), d), and e) above would change if, in the original automaton, state S2 was a final state too. Justify your answer. (2 marks)

[PTO]

3. a) Consider the following grammar for boolean expressions.

E → E or E
E → E and E
E → not E
E → (E)
E → true
E → false
E → id

(i) Show that this grammar is ambiguous.

(4 marks)

(ii) Rewrite the grammar to remove the ambiguity by introducing new non-terminal symbols. Your revised grammar must accept the same language as the original.

(8 marks)

b) Consider the grammar and the Action and Goto tables below:

1. E→E+E
2. E→E*E
3. E→(E)
4. E→id

state	ACTION						GOTO
	id	+	*	(	)	EOF	E
0	S3			S2			1
1		S4	S5			accept	
2	S3			S2			6
3		R4	R4		R4	R4	
4	S3			S2			7
5	S3			S2			8
6		S4	S5		S9		
7		R1	S5		R1	R1	
8		R2	R2		R2	R2	
9		R3	R3		R3	R3	

Show, in full detail, the steps that an LR parser, using the tables above, would follow to parse the string `(x+y)*z`. After you produce all the steps, you should also draw the resulting parse tree. For each step, your answer should show the contents of the stack, what the next input is and the action that is taken.

(8 marks)

4. a) Draw the control flow graph of the following code fragment.

```
     m=0; n=1000; s=0
L1:  if (m>n) goto L4
     r=a[m]
L2:  if (r<m) goto L3
     n=n+1
     goto L1
L3:  x=a[r]
     s=s+x
     if (s<m) goto L4
     m=s
     r=r+1
     goto L2
L4:  return
```

(4 marks)

b) i) Provide a generic approach that a C compiler might follow to generate code for a C-like `for` loop of the form:

```
for (expr1; expr2; expr3) {
  loop_body;
}
```

(4 marks)

(ii) A certain C compiler has generated executable code for the following program. When running the executable, the program falls into an infinite loop if x and y are given the values 2147483644 and 2147483647, respectively, but it works as expected if x and y are given the values 1 and 4. (in which case four dots are printed and the program terminates). What can you infer about the method used by this compiler to generate code for `for` loops?

```
main() {
int i,x,y;
scanf("%d",&x); scanf("%d",&y);
for(i=x;i<=y;i++)
  printf(".");
}
```

(4 marks)

c) Consider the following C-like code fragment.

```
n=1000;
for (i=1; i<=n; i++)  {
   t=3; if (i==1) x=t+10;
   y=x+n; a[i]=y;
}
```

What kind of code transformation techniques can a compiler apply to improve the efficiency of this code fragment? For each transformation, give the resulting version of the code after applying the transformation. Your answer should clearly show what the most optimized version you can obtain is.

(8 marks)

[PTO]

5. a) Consider the following basic block.

```
1. load r1, @x
2. load r2, @y
3. add r3, r1, r2
4. mult r4, r1, r2
5. add r5, r3, 1
6. add r6, r4, r3
7. sub r7, r6, r4
8. mult r8, r5, r7
```

(i) Explain why a processor with only 4 registers would be sufficient to execute this block without the need to do any spilling.

(4 marks)

(ii) Explain how Best's algorithm can be used for register allocation in the above basic block. Show what the basic block would look like after applying the algorithm.

(4 marks)

b) (i) Describe an efficient list scheduling algorithm for instruction scheduling on a hypothetical system with two functional units, if the cost of executing an instruction (delay latency of the instruction) on the second functional unit is one cycle more than the cost of executing the same instruction on the first functional unit.

(6 marks)

(ii) Apply your algorithm above to the basic block below and show the schedule.

```
1. load r1, @x
2. load r2, [r1+64]
3. add r3, r3, 1
4. mult r6, r6, 7
5. store r6
6. div r5, r5, 12
7. add r4, r2, r5
8. mult r5, r2, r4
9. store r4
```

You should assume that, on the fastest functional unit, the cost of a `load` and a `store` is 3 cycles, the cost of a `mult` is 2 cycles and the cost of all remaining operations is 1 cycle.

(6 marks)

END OF EXAMINATION