EASTERN UNIVERSITY, SRI LANKA

DEPARTMENT OF MATHEMATICS

SPECIAL DEGREE EXAMINATION IN COMPUTER SCIENCE -

2009/2010 (Sep./Oct., 2011)

CS 408: Compiler Design

Answer all questions

This paper has 4 questions in a total of 3 pages

e allowed: Three Hours

A finite automaton is, in the abstract sense, a machine that has a finite number of states and a finite number of transitions among them.

(a) State the purpose of finite automaton in compiler design. [10%]

(b) Describe the following DFA in words briefly, and give a regular expression to represent it. There are four states A,B,C,D; A is the start state and D is the only final state; the alphabet consists of 1 and 0, and the transitions are written as triplets:

(from state, alphabet symbol, to state):

(A,1,A) (A,0,B) (B,1,C) (B,0,B)
(C,0,D) (C,1,A) (D,0,B) (D,1,C) [20%]

(c) Explain how an NFA can be converted into a DFA with relevant detail. [15%]

(d) Explain how a DFA can be converted to be with minimal set of states. [10%]

(e) Give the NFA construction for the regular expression, ~~(0*1+)*0*~~, $(0^*1^+)^*0^*$ and then find a minimised DFA for the same language of the expression. [25%]

(f) Even though NFA and DFA have the same constructs, explain why it would be difficult to directly construct an equivalent DFA for a given regular expression. [10%]

(g) Discuss about the expressive power of regular expression, NFA and DFA. [10%]

. The syntax analyser of a compiler recombine the tokens identified in the lexical analysis phase of a compiler into a syntax tree which reflects the structure of the input text with respect to the grammar of the programming language.

(a) Explain the purpose and structure of context-free grammars and describe its part in the syntax analysis phase of a compiler. [10%]

(b) Using a simple example explain how the syntax analyser constructs the syntax tree through derivations. [10%]

(c) State what ambiguity is in the context of syntax analysis and and the problems arisen in syntax analysis due to ambiguity. [10%]

(d) State the causes for ambiguity in a context-free grammar and explain how ambiguity can be removed. [20%]

(c) Consider the following operators:

- The operator $\uparrow$ is used to find x to the power of y. (Example: $7 \uparrow 3 = 7^3$)
- The $\downarrow$ is used to find x to the power of $1/y$. (Example $7 \downarrow 3 = \sqrt[3]{7}$
- The operator $\odot$ is used to find the greatest common divisor of x and y. (Exam $21 \odot 15 = 3$)

here the $\odot$ operator has a lower precedence than the other two operators. The po operators have the same precedence level. Answer the following questions:

i. Write a simple context free grammar for the operators stated above.

ii. Describe the associativity of the operators using suitable examples.

iii. Find out whether the grammar you have given in part i. is ambiguous or not, w the aid of a suitable example.

iv. If the grammar is ambiguous then rewrite the grammar to remove the ambiguity it is not ambiguous then explain how it is so.

v. Apply the unambiguous grammar to construct the syntax tree for the input $6 \uparrow 2 \odot 16 \downarrow 2$.

3. A predictive parser is a recursive descent parser that does not require backtracking.

(a) Explain the process of predictive parsing.

(b) With a suitable example show how *First* and *Nullable* helps predictive parsing.

(c) Using suitable examples describe situations where the First and Nullable fail their purp in predictive parsing.

(d) With a suitable example explain how *Follow* helps the predictive parsing.

(e) State clearly the LL(1) parsing method.

(f) Consider the following grammar with four terminals: `=`, `+`, `*`, `int`.

`S` $\rightarrow$ `B + B`

`A` $\rightarrow$ `*`

`B` $\rightarrow$ ϵ

`B` $\rightarrow$ `int B B`

`B` $\rightarrow$ `A =`

Answer the following:

i. Construct the Nullable, First and Follow sets.

ii. Construct the LL(1) parsing table.

(g) Explain what is meant by conflict in LL(1) parsing and the causes for conflicts.

(h) Discuss the causes for conflicts and ways to remove it in the LL(1) parsing table you ha constructed in part f(ii).

LR parser is a parser that reads input from left to right and produces a rightmost derivation and SLR parser is a simple form of LR parser.

(a) There are a number of actions associated with an SLR parser. List those actions and describe each. [10%]

(b) State the steps involved in constructing the SLR parsing table from a DFA. [15%]

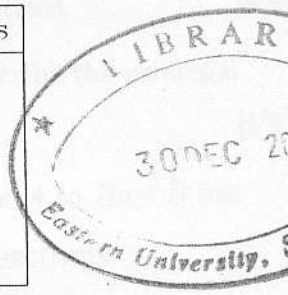
(c) Explain briefly how a given input string of tokens can be parsed using SLR parser. [10%]

(d) Consider the following grammar, NFA fragments and the DFA transitions for the grammar:

Grammar		
S′	→	S
S	→	Sa
S	→	b
S	→	e a

NFA fragments
(1, S, 2)
(3, S, 4, a, 5)
(6, b, 7)
(8, e, 9, a, 10)

DFA transitions
(I_0, S, I_1)
(I_1, a, I_5)
(I_0, b, I_2)
(I_0, e, I_3)
(I_3, a, I_4)



In each of the NFA fragments,the numbers represent the NFA states and the symbols `S`, `a`, `b`, `e` are the input tokens which make the transitions among NFA states.

In each DFA transition, I_0, I_1, I_2, I_3 & I_4 are the DFA states and the symbols `S`, `a`, `b`, `e` are the tokens which make the transitions among DFA states.

i. Draw the NFA and DFA in a graphical form suitable to construct the SLR parsing table from the given information. [10%]

ii. Construct the SLR parsing table. [20%]

iii. Parse the the string "`e a a`" showing all sequence of actions and the state of the stack. [10%]

iv. When a parsing error is encountered (*i.e.*, the parser cannot move further from some point using the allowed operations) then the following actions could be performed:

(α) Pop and discard elements of the stack one at a time until the stack is empty.

(β) Shift the token `e` on to the stack.

(γ) Discard tokens of the input one at a time until one is found such that an operation can be performed.

(δ) Resume normal execution of the parser.

Show the sequence of actions and the state of the stack for parsing the input "`bacadfa`" using the parsing table constructed in part ii. and the rules given above. Be sure to show all discard actions (for both stack and input). [25%]