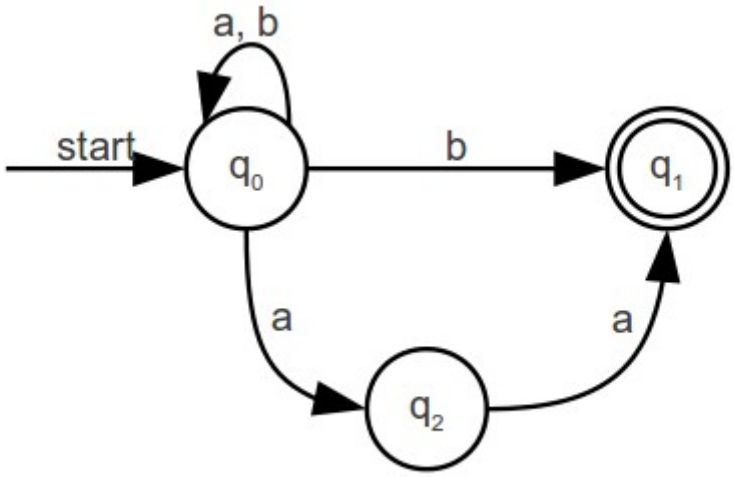
Cairo University
Faculty of Computers & Information
Computer Science Department
Year 2014 – 2015
Second Term

Compilers - CS416
Mid-Term Exam
28/03/2015

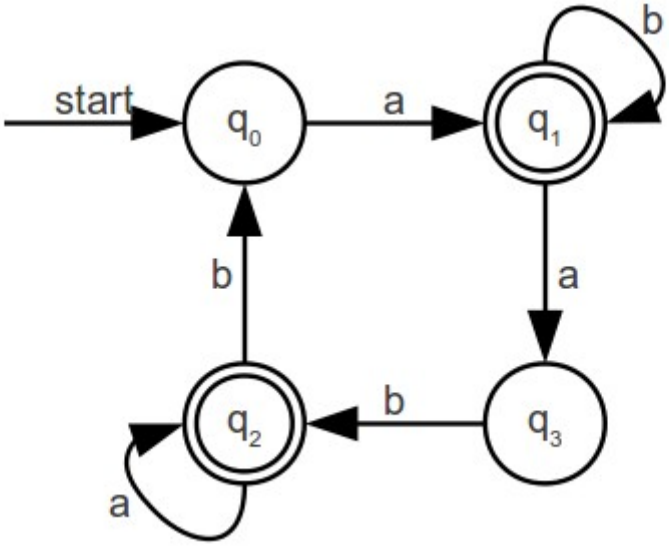
You have 60 minutes to complete this exam. You are not allowed to use any books, notes, calculators, or electronic devices. Write your answers carefully. Incomplete, unintelligible, or illegible answers will receive little or no credit. There are a total of 50 points on this exam.

1- [10 points] Use the subset construction to convert the following NFAs into DFAs. You can assume that the alphabet is just the two letters a and b . Remember that in a DFA, every state must have a transition defined on every symbol.

i.

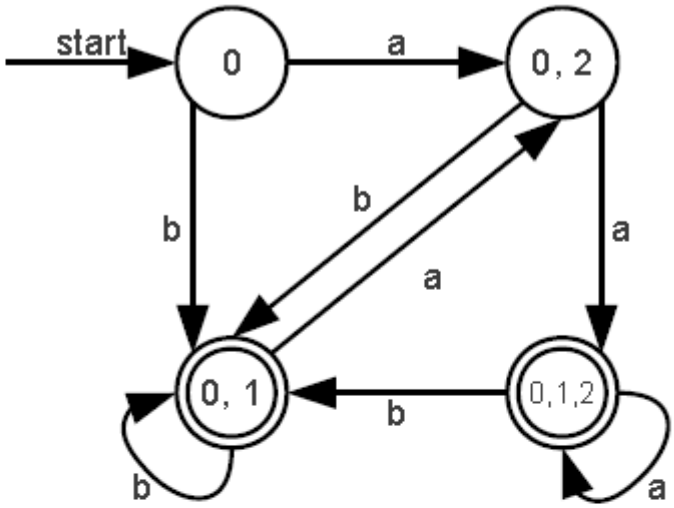


ii.

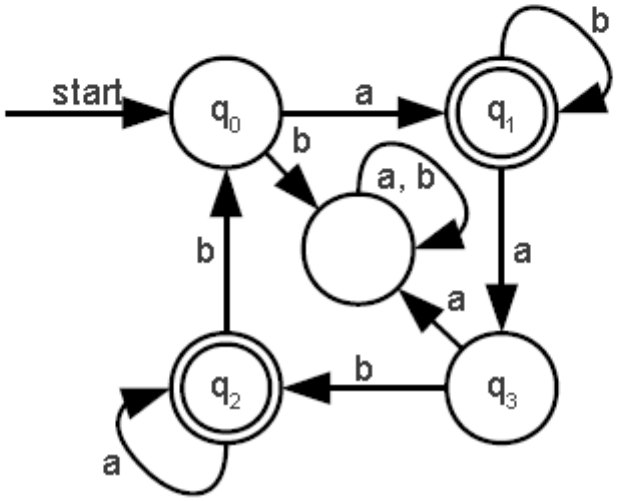


Answer:

i.



ii.

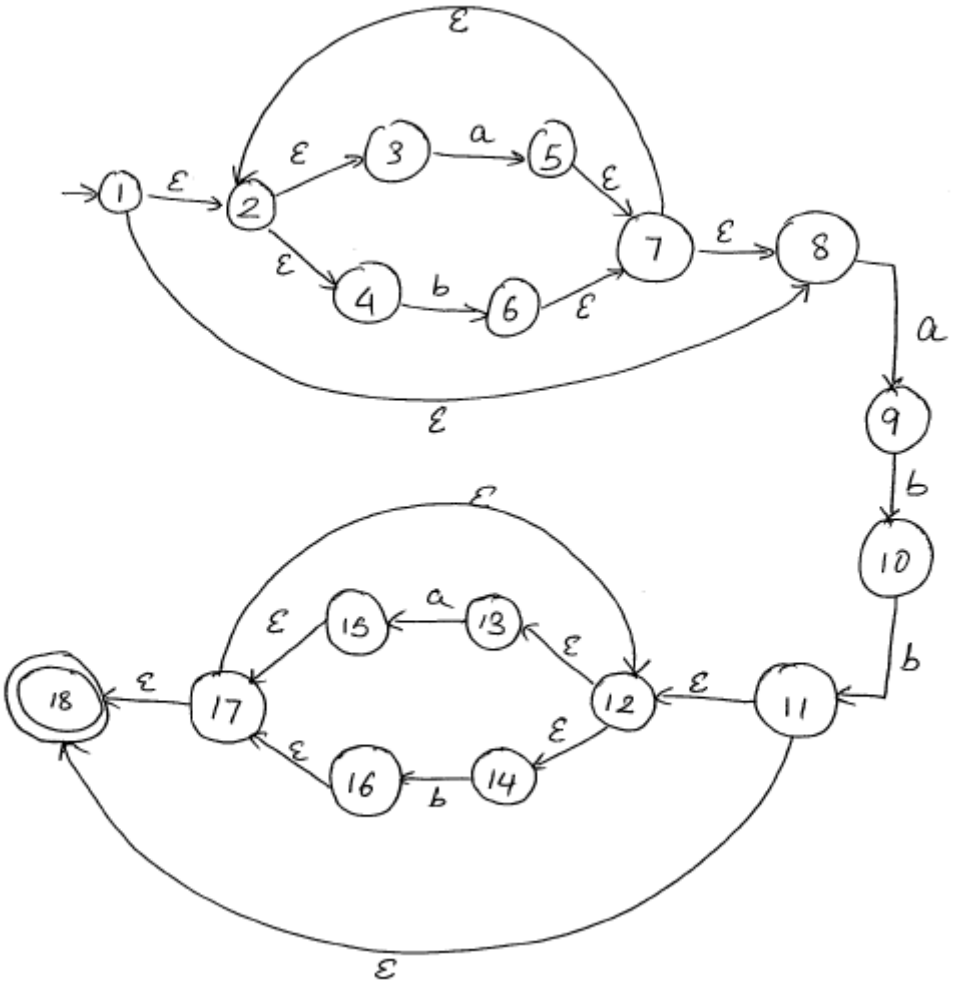


2- [5 points] A compiler can be decomposed into front end and back end.
(a) What are the main components of a compiler front-end ?
Lexical analyzer, parser, semantic analyzer, IR code generation
(b) What are the main components of a compiler back-end ?
(IR and machine code) optimization, Code generation.

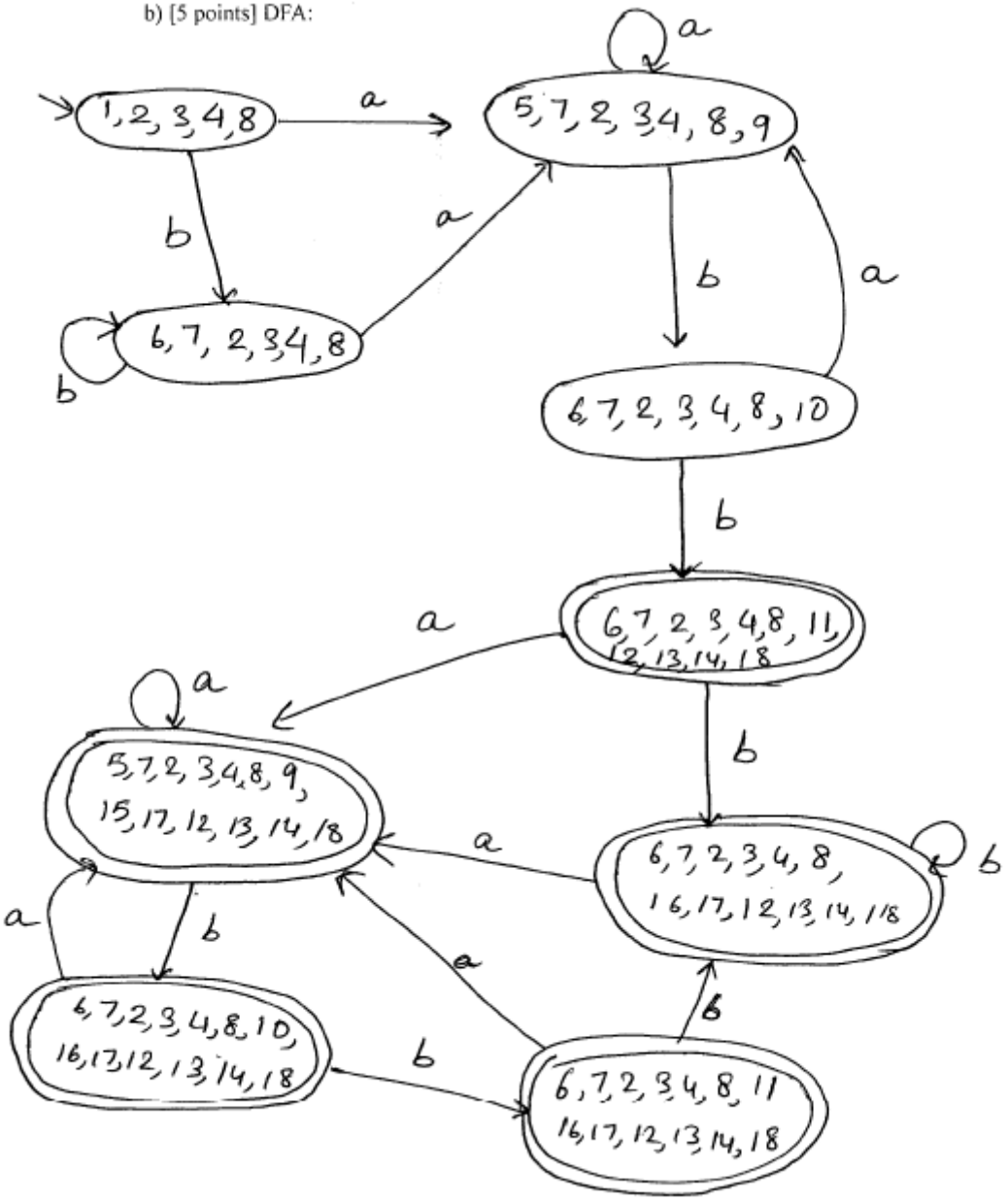
3- [10 points] Convert the regular expression (a | b)* a b b (a | b)* to deterministic finite automata by first creating a non-deterministic finite state machine (NFA) and then converting the NFA into a deterministic machine.

Solution attached

a) [5 points] NFA:



b) [5 points] DFA:

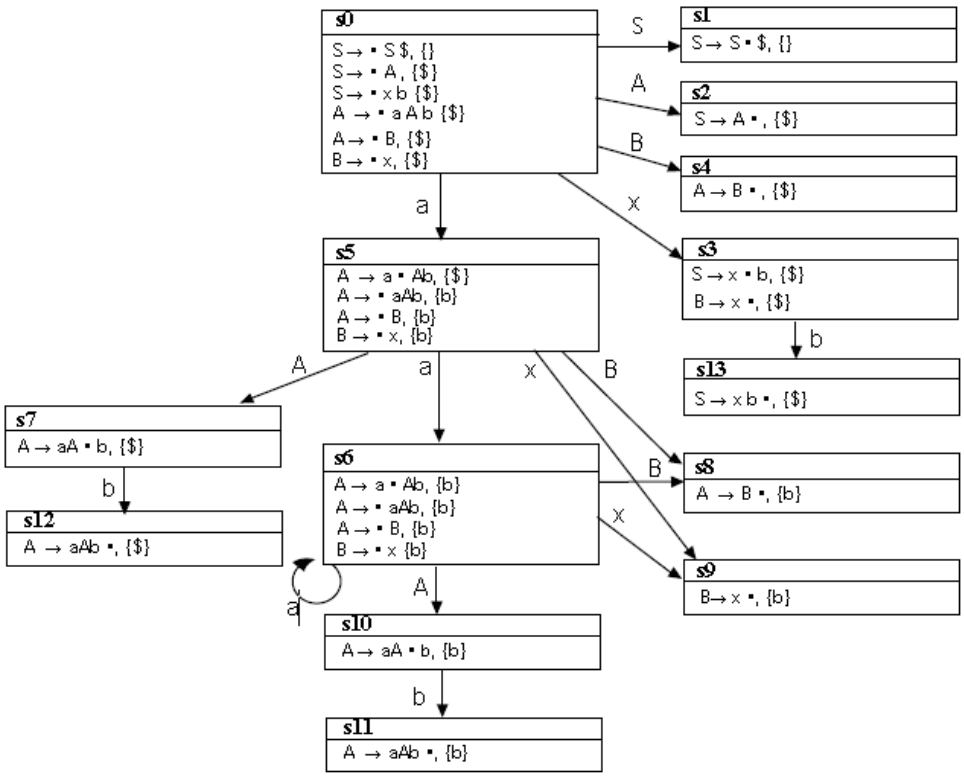


4- Given the following CFG grammar = ({S,A,B}, S, {a, b, x), P) with P:
S → A
S → xb
A → aAb
A → B
B → x

For this grammar answer the following questions:
a) [4 points] Compute the set of LR(1) items for this grammar and the corresponding DFA. Do not forget to augment the grammar with the default initial production S' → S$ as the production (0).
b) [3 points] Construct the corresponding LR parsing table.
c) [3 points] Would this grammar be LR(0)? Why or why not?
d) [3 points] Show the stack contents, the input and the rules used during parsing for the input string w = axb$

Solution attached

a) The set of LR(1) items and the corresponding DFA are shown below:

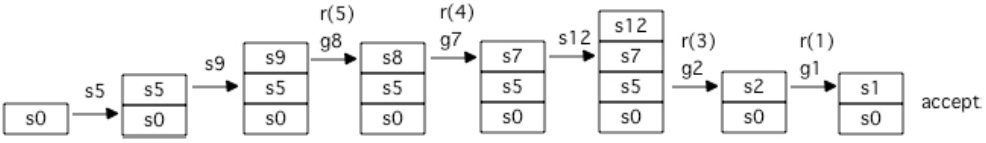


b) The parsing table is as shown below:

State	terminals				goto		
	a	b	x	$	S	A	B
0	s5		s3		g1	g2	g4
1				acc			
2				r(1)			
3		s13		r(5)			
4				r(4)			
5	s6		s9			g7	g8
6	s6						
7		s12					
8		r(4)					
9		r(5)					
10		s11				g10	
11		r(3)					
12				r(3)			
13				r(2)			

c) If we were to compute the set of LR(0) items for state s3 we would have a shift-reduce conflict as on "b" as in this method for building the parse table we would have a shift action and for all the input tokens but the second item in the s3 set of items would suggest a reduction. This grammar is thus not LR(0) parseable.

d) The stack contents for the input string w = axb$.



5- Given the following CFG grammar G = ({SL}, S, {a, “(“,”)”,“,”}, P) with P:
S → (L) | a
L → L, S | S
a) [3 points] Is this grammar suitable to be parsed using the recursive descendent parsing method? Justify and
modify the grammar if needed.
b) [3 points] Compute the FIRST and FOLLOW set of non-terminal symbols of the grammar resulting from your
answer in a)
c) [3 points] Construct the corresponding parsing table using the predictive parsing LL method.
d) [3 points] Show the stack contents, the input and the rules used during parsing for the input string w = (a,a)

Solution attached

a) No because it is left-recursive. You can expand L using a production with L as the left-most symbol without consuming any of the input terminal symbols. To eliminate this left recursion we add another non-terminal symbol, L’ and productions as follows:

S → (L) | a
L → S L’
L’ → , S L’ | ε

b) FIRST(S) = { (, a }, FIRST(L) = { (, a } and FIRST(L’) = { , , ε}
FOLLOW(L) = {) }, FOLLOW(S) = { , ,) , $ }, FOLLOW(L’) = {) }

	(	)	a	,	$
S	S→(L)		S → a		
L	L→S L'		L→ S L'		
L'		L'→ε		L'→ , S L'	

d) The stack and input are as shown below using the predictive, table-drven parsing algorithm:

STACK	INPUT	RULE/OUTPUT
$S	(a,a)$	
$) L (	(a,a)$	S → (L)
$) L	a,a)$	
$) L' S	a,a)$	L → S L'
$) L' a	a,a)$	S → a
$) L'	,a)$	
$) L' S ,	,a)$	L' → , S L'
$) L' S	a)$	
$) L' a	a)$	S → a
$) L'	)$	
$)	)$	L' → ε
$	$	

Best wishes
Dr. Nouh Sabri