

Answer all questions in the exam book. You may keep the exam questions after you are done.

I. [56] Consider the following grammar G:

1. $S \rightarrow X \perp$
2. $X \rightarrow A 1 B$
3. $X \rightarrow 2$
4. $A \rightarrow 2$
5. $B \rightarrow A$

[Note: Original exam had additional rule $B \rightarrow 2$, which makes grammar ambiguous and not LR(1) or LALR(1).]

(a) LL(0)

Not LL(0) since more than 1 string can be generated, and LL(0) grammars generate languages containing single strings:

$$S \rightarrow X\perp \rightarrow 2\perp$$

$$S \rightarrow A1B\perp \rightarrow 21B\perp \rightarrow 212\perp$$

(b) LL(1)

$$X \rightarrow A1B \text{ and } X \rightarrow 2$$

$$\text{First}(A1B) = \{2\}$$

$$\text{First}(2) = \{2\}$$

$$\text{First}(A1B) \cap \text{First}(2) = \{2\} \neq \emptyset$$

So not LL(1)

(c) LR(0)

S0	$[S \rightarrow .X\perp]$	X - S1
	$[X \rightarrow .A1B]$	A - S2
	$[X \rightarrow .2]$	2 - S3
	$[A \rightarrow .2]$	
S1	$[S \rightarrow X.\perp]$	\perp - S4
S2	$[X \rightarrow A.1B]$	1 - S5
S3	$[X \rightarrow 2.]$	
	$[A \rightarrow 2.]$	

Reduce/reduce conflict in S3, so not LR(0).

(e) SLR(1)

Continuing with LR(0) table of (c) to check for SLR(1). Look at follow sets in S3:

$$\text{Follow}(X) = \{\perp\}$$

$$\text{Follow}(A) = \{1, \perp\}$$

$$\text{Follow}(A) \cap \text{Follow}(X) = \{\perp\}$$

Follow sets do not resolve ambiguity, so still have reduce/reduce conflict. So not SLR(1).

(d) LR(1)

S0	$[S \rightarrow .X\perp,]$	X - S1
	$[X \rightarrow .A1B, \perp]$	A - S2
	$[X \rightarrow .2, \perp]$	2 - S3
	$[A \rightarrow .2, 1]$	
S1	$[S \rightarrow X.\perp,]$	\perp - S4
S2	$[X \rightarrow A.1B, \perp]$	1 - S5
S3	$[X \rightarrow 2., \perp]$	\perp - Reduce 3
	$[A \rightarrow 2., 1]$	1 - Reduce 4
S4	$[S \rightarrow X\perp.,]$	Reduce 1
S5	$[X \rightarrow A1B., \perp]$	B - S6
	$[B \rightarrow .A, \perp]$	A - S7
	$[A \rightarrow .2, \perp]$	2 - S8
S6	$[X \rightarrow A1B., \perp]$	\perp - Reduce 2
S7	$[B \rightarrow A., \perp]$	\perp - Reduce 5
S8	$[A \rightarrow 2., \perp]$	\perp - Reduce 4

(f) LALR(1)

All the core items in each of the 9 states are different, so no 2 are possible to merge. But "All core items that are the same can be merged" so by default the grammar is LALR(1).

(g) Operator precedence

1. Grammar IS an operator grammar, so it might be operator precedence.
2. Parsing tables:

	FIRST	LAST
S	2	\perp
X	2	2
A	2	2
B	2	2

$$2 = \text{LAST}(A) > 1$$

$$1 < \text{FIRST}(B) = 2$$

$$2 = \text{LAST}(X) > \perp$$

	1	2	\perp
1		<	
2	>		>
\perp			

II. [11] What is k and EXPLAIN WHY for each of the following:

(a) What is the minimal value of k for which a Polish postfix grammar is LR(k)?

$k=0$. (Grammar is LR(0). When see operator, can make reduction. (Alternatively, show that the grammar: $S \rightarrow S S a \mid S b$ where a and b are operators is LR(0).)

(b) What is the minimal value of k for which a Polish prefix grammar is LR(k)?

$k=0$. (Grammar is LR(0). When see appropriate number of operands, can make reduction. (Alternatively, show that the grammar: $S \rightarrow a S S \mid b S$ where a and b are operators is LR(0).)

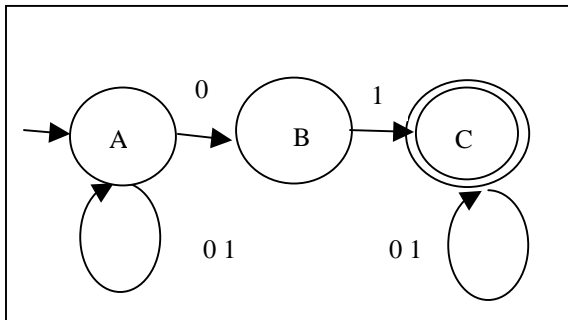
(c) What is the minimal value of k for which a grammar for miniscript is LR(k)?

$k=0$. (Grammar is LR(0). Miniscript is just Polish postfix.

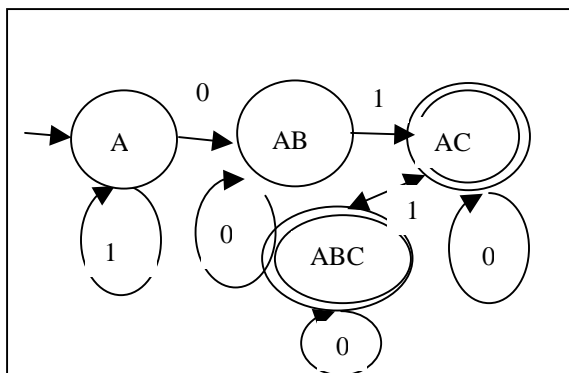
III. [15] For the regular expression $(0 \mid 1)^* 01 (0 \mid 1)^*$

(a) Give the minimal state DFA that recognizes the same set?

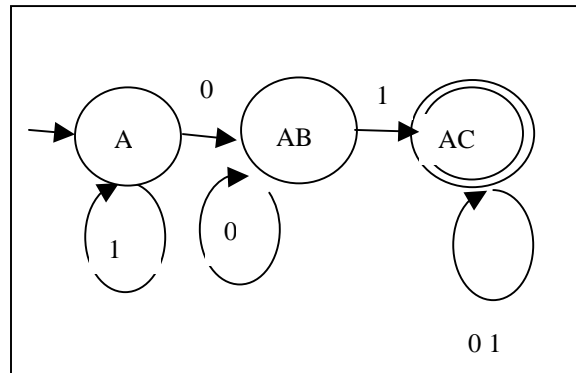
NDFA is:



DFA:



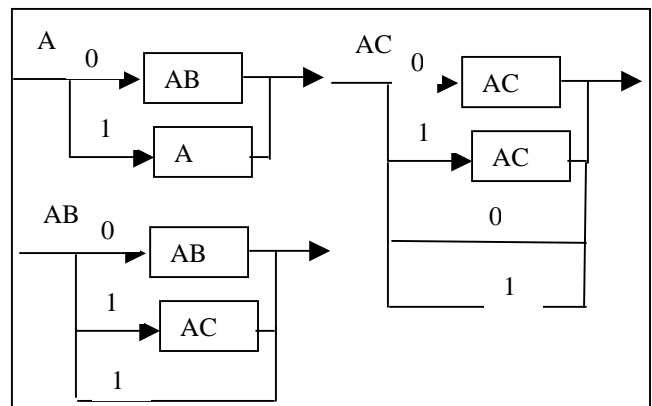
Minimized DFA



(b) Give the regular grammar that recognizes the same set?

$\langle A \rangle ::= 1 \langle A \rangle \mid 0 \langle AB \rangle$
 $\langle AB \rangle ::= 0 \langle AB \rangle \mid 1 \langle AC \rangle \mid 1$
 $\langle AC \rangle ::= 0 \langle AC \rangle \mid 1 \langle AC \rangle \mid 0 \mid 1$

(c) Give the syntax diagrams for the regular grammar in (b) above.



IV. [18] Answer each of the following:

(a) If S is a regular set, show that S^r is regular. (S^r means S -reversed. That is, $abc \in S$ if and only if $cba \in S^r$)

1. S is regular, so there exists a DFA M that recognizes S .
2. Construct a NDFA N with a new start state A and a final state F .
3. Reverse all the arrows in the original machine M .
4. Add epsilon arcs from A to all the final states of M .
5. Add epsilon arcs from the start state of M to F .

This machine now works “backwards.” It will accept a string if and only if the original machine M accepts the reverse string.

(b) Consider S^r (from part (a) above). Is $(S \cap S^r)$ regular, context free, or context sensitive?(Choose the smallest class.) Prove your answer.

$X \in (S \cap S^r)$ means $X = X^r$. So the first character of X is the first character of X^r , which is the last character of X . That is, X must be a palindrome. We already know that it takes a push down automata to recognize palindromes, so S^r must be context free and not regular.

(c) Why is the following not LR(k) for any k?
 $\{x^a y^b z^c \mid a=b \text{ or } b=c, a>0, b>0, c>0\}$

This is the set $x^n y^n z^m \cup x^m y^n z^n$. When $m=n$ we get the set $x^n y^n z^n$. This is inherently ambiguous. That is, any parse will not know whether you need to check for $a=b$ or check if $b=c$. Either one of these cases can easily be checked by a push down automaton. When $a=b$ and $b=c$, the parse can go either way. But LR(k) means a deterministic parse, which this is not. So grammar is context free, but not LR(k) for any k.