Solutions Exam 150819

Here we only give a brief explanation of the solution. Your solution should in general be more elaborated than these ones.

1. We will use course-of-value induction on the length of the derivation (number of steps) $T \Rightarrow^* w$.

   Base case: $T \Rightarrow w$, hence the rule applied should have been $T \to \mathsf{Lf}$.

   Here, $\#_{\mathsf{Nd}}(w) = 0$ and $\#_{\mathsf{Lf}}(w) = 1$. Hence $\#_{\mathsf{Nd}}(w) < \#_{\mathsf{Lf}}(w)$.

   Step case: IH: if $T \Rightarrow^* w$ in at most $n > 0$ steps then $\#_{\mathsf{Nd}}(w) < \#_{\mathsf{Lf}}(w)$.

   Let $T \Rightarrow^* w'$ in $n + 1$ steps with $n > 0$.

   Since $n > 0$ then the rule applied should have been $T \to \mathsf{Nd}\ T\ T$.

   It should also be that $w' = \mathsf{Nd}\ w_1\ w_2$ with both $T \Rightarrow^* w_1$ and $T \Rightarrow^* w_2$ in less than n steps. Hence the IH applies to both derivations and we have $\#_{\mathsf{Nd}}(w_1) < \#_{\mathsf{Lf}}(w_1)$ and $\#_{\mathsf{Nd}}(w_2) < \#_{\mathsf{Lf}}(w_2)$. In other words, $\#_{\mathsf{Nd}}(w_1) + 1 \leqslant \#_{\mathsf{Lf}}(w_1)$ and $\#_{\mathsf{Nd}}(w_2) + 1 \leqslant \#_{\mathsf{Lf}}(w_2)$.

   Then $\#_{\mathsf{Nd}}(w_1) + \#_{\mathsf{Nd}}(w_2) + 2 \leqslant \#_{\mathsf{Lf}}(w_1) + \#_{\mathsf{Lf}}(w_2)$ and $\#_{\mathsf{Nd}}(w_1) + \#_{\mathsf{Nd}}(w_2) + 1 < \#_{\mathsf{Lf}}(w_1) + \#_{\mathsf{Lf}}(w_2)$. That is, $\#_{\mathsf{Nd}}(w') < \#_{\mathsf{Lf}}(w')$.

2. (a)

	0	1
$\to^* A$	B	A
B	C	B
C	A	C

   (b)

	0	1
$\to D$	E	D
E	E	F
F	G	D
$^*G$	G	G

   (c)

	0	1
$\to^* AD$	BE	AD
$^*AE$	BE	AF
$^*AF$	BG	AD
$^*AG$	BG	AG
BD	CE	BD
BE	CE	BF
BF	CG	BD
$^*BG$	CG	BG
CD	AE	CD
CE	AE	CF
CF	AG	CD
$^*CG$	AG	CG

3. (a)

	0	1
$\to q_0$	$\emptyset$	$\{q_1\}$
q_1	$\{q_2\}$	$\emptyset$
q_2	$\{q_3\}$	$\{q_2, q_5\}$
q_3	$\emptyset$	$\{q_4\}$
q_4	$\{q_2, q_3\}$	$\{q_5\}$
q_5	$\{q_6\}$	$\emptyset$
$^*q_6$	$\emptyset$	$\emptyset$

(b)

	0	1
$\to q_0$	q	q_1
q_1	q_2	q
q_2	q_3	q_2q_5
q_3	q	q_4
q_4	q_2q_3	q_5
q_5	q_6	q
$^*q_6$	q	q
q_2q_5	q_3q_6	q_2q_5
q_2q_3	q_3	$q_2q_4q_5$
$^*q_3q_6$	q	q_4
$q_2q_4q_5$	$q_2q_3q_6$	q_2q_5
$^*q_2q_3q_6$	q_3	$q_2q_4q_5$
q	q	q

4. I will solve equations:

$$E_0 = 0E_1 + 1E_3 \qquad E_0 = 0E_1 + 10E_2 + 11E_1 = (0+11)E_1 + 10E_2$$
$$E_1 = 0E_3 + 1E_2 \qquad E_1 = 00E_2 + 01E_1 + 1E_2 = 01E_1 + (00+1)E_2 = (01)^*(00+1)E_2$$
$$E_2 = 0E_1 + 1E_3 + \epsilon \qquad E_2 = 0E_1 + 10E_2 + 11E_1 + \epsilon = (0+11)E_1 + 10E_2 + \epsilon$$
$$E_3 = 0E_2 + 1E_1$$

$$E_0 = ((0+11)(01)^*(00+1) + 10)E_2$$
$$E_2 = ((0+11)(01)^*(00+1) + 10)E_2 + \epsilon = ((0+11)(01)^*(00+1) + 10)^*$$

Hence

$$E_0 = ((0+11)(01)^*(00+1) + 10)((0+11)(01)^*(00+1) + 10)^* = ((0+11)(01)^*(00+1) + 10)^+$$

5. q_3 is not reachable so we eliminate it before constructing the table.

	q_0	q_1	q_2	q_4	q_5	q_6
q_7	X	X	X	X	X	X
q_6	X	X	X	X		
q_5	X	X	X	X		
q_4	X	X	X			
q_2	X	X				
q_1						

The resulting automaton is (remember that equivalence of states is transitive):

	0	1
$\rightarrow q_0q_1$	q_7	q_0q_1
q_2	q_4	q_5q_6
q_4	q_5q_6	q_5q_6
q_5q_6	q_5q_6	q_5q_6
q_7	q_2	q_2

6. (a) RE represent languages so one could prove that both languages contain the same set of words by double inclusion.

   One could transform one of the RE into the other using known algebraic laws on RE.

   One could also transform the REs into DFAs and then minimize both DFAs and see if they are the same (the minimim DFA is unique modulo the name of the states).

   Alternative one could transform the REs into DFAs and check if both the initial states are equivalent.

   (b) i. (2.5pts) Yes.
   Inclusion from left to right: The first expression genetares cero or more 1's and then any sequence of 0's and 1's, even the empty one. The second expression can also do this if in each iteration of the external closure ($*$) one generates as many 0's or 1's as needed in order to generate the expression the word in question.
   Inclusion from right to left: Observe that by taking the 1^* as ϵ the rest of the first expression can generate any sequence of 0's and 1's, so in particular can generate anything that the second expression can generate.

   ii. (1.5pts) No. The first expression can generate only a but the smallest non-empty word the second expression can generate is ab.

7. (a)

$$\begin{array}{ll} S \rightarrow PD \mid AQ \mid AD & A \rightarrow aAb \mid \epsilon \\ P \rightarrow aPc \mid aAc & D \rightarrow cDd \mid \epsilon \\ Q \rightarrow bQd \mid bDd & \end{array}$$

   (b) A generates as many a's as b's, B generates as many b's as d's, C generates as many a's as c's and D generates as many c's as d's.

   There are three possible cases: either $n > m$ (and hence $l < k$), $n < m$ (and hence $l > k$) or $n = m$ (and hence $(l = k)$.

   The first case is generated by starting with the rules $S \rightarrow PD$. P will put the extra $n - m$ a's and c's (at least one of each) and the continue with as many a's and b's (if any) via A. D will put the rest of the c's and d's (if any).

   The second case is generated by starting with the rules $S \rightarrow AQ$. Q will put the extra $m - n$ b's and d's (at least one of each) and the continue with as many c's and d's (if any). A will put the rest of the a's and b's.

   The third case is generating with the rules $S \rightarrow AD$.

   (c) The leftmost derivation is

$$S \Rightarrow PD \Rightarrow aPcD \Rightarrow aaAccD \Rightarrow aaaAbccD \Rightarrow aaabccD \Rightarrow aaabcccDd \Rightarrow aaabcccd$$

   (d) No, the grammar is not ambiguous. Observe that the three starting rules are mutually exclusive. In all other rules, the options are also mutually exclusive: either we do a recursive derivation or we move to move to generating a different part of the word.

8. (a) Slide 23, lecture 12.

   (b) Assume then language is context-free. So the Pumping lemma applies.
   Let n be the constant given by the lemma.
   Let $w = 0^n1^n20^n1^n$ which is long enough and belongs to the language.
   Then $w = xuyvz$ with $uv \neq \epsilon$ and $|uyv| \leqslant n$.
   If uyv takes place completely before (after) the 2, then whatever is pump/remove there will not be pump/remove after (before) the 2 and hence the word will not have the form $s2s$.
   If 2 is part of uyv than uyv should be 1^i20^j for i, j such that $i + j + 1 \leqslant n$. Even here the resulting word is not of the form $s2s$ either because we pump/remove the 2, or because we pump/remove 0's (and/or 1's) but we do not pum/remove them in the first (second) half of the word.

9.

$\{S, B\}$				
$\{S, A, C\}$	$\emptyset$			
$\{S, A, C\}$	$\emptyset$	$\{S\}$		
$\emptyset$	$\{B\}$	$\emptyset$	$\{S\}$	
$\{A\}$	$\{A\}$	$\{C\}$	$\{B\}$	$\{A\}$
a	a	c	b	a

   S belongs to the upper-most set, which means that the word is generated by the grammar since S is the starting symbol of the grammar.

10. (a) See lecture 14 slide 17.

   (b) Let $\Sigma = \{0, 1, 2\}$.
   Let $M = (\{q_0, ..., q_6, q_y\}, \Sigma, \delta, q_0, \square, \{q_y\})$, with δ is as follows:

$\delta(q_0, 0) = (q_1, 0, R)$	the word starts with 0 which is good;
$\delta(q_1, 0) = (q_1, 0, R)$	we move forwards through the 0's;
$\delta(q_1, 1) = (q_2, 1, R)$	we found the first set of 1's;
$\delta(q_2, 1) = (q_2, 1, R)$	we move forwards through the 1's;
$\delta(q_2, 2) = (q_3, 2, R)$	we found the first 2;
$\delta(q_3, 2) = (q_3, 2, R)$	we move forwards through the 2's;
$\delta(q_3, 1) = (q_4, 1, R)$	we found the second set of 1's;
$\delta(q_4, 1) = (q_4, 1, R)$	we move forwards through the 1's;
$\delta(q_4, 0) = (q_5, 0, R)$	we found the first 0 in the end;
$\delta(q_5, 0) = (q_5, 0, R)$	we move forwards through the 0's;
$\delta(q_5, \square) = (q_y, \square, R)$	the word is of the correct form;
$\delta(q_1, 2) = (q_6, 2, R)$	there are no 1's but we found the first 2;
$\delta(q_6, 2) = (q_6, 2, R)$	we move forwards through the 2's;
$\delta(q_6, 0) = (q_5, 0, R)$	we found the first 0 in the end; we need to see that the rest is just 0's, q_5 will do it!

   Whenever the tapes is not of the correct form the machine will halt in a state other than q_y indicating that the input is not correct.