Student Information

Full Name : Utku Gungor
Id Number : 2237477

Answer 1

a.

(i) The CFG that generate this language is G = (K,Σ,Δ,E) , where

$K = \{E,A\}$,

$\Sigma = \{a,b\}$,

$\Delta = \{E \rightarrow aEa \mid bA$,

$A \rightarrow aAa \mid b \}$. $\diamondsuit$

(ii) The CFG that generate this language is G = (K,Σ,Δ,E) , where

$K = \{E,X,Y\}$,

$\Sigma = \{a,b\}$,

$\Delta = \{E \rightarrow aXa \mid bYb$,

$X \rightarrow aXa \mid bXb \mid aXb \mid bXa \mid aaa \mid aab \mid baa \mid bab$,

$Y \rightarrow bYb \mid aYa \mid aYb \mid bYa \mid aba \mid abb \mid bba \mid bbb \}$. $\diamondsuit$

(iii) The CFG that generate this language is G = (K,Σ,Δ,E) , where

$K = \{E,X,Y,Z,P,R,S\}$,

$\Sigma = \{a,b,c\}$,

$\Delta = \{E \rightarrow aXZ \mid YbZ \mid PbR \mid PSc$,

$X \rightarrow aXb \mid aX \mid \varepsilon$,

$Y \rightarrow aYb \mid Yb \mid \varepsilon$,

$Z \rightarrow cZ \mid \varepsilon$,

$P \rightarrow aP \mid \varepsilon$,

$R \rightarrow bRc \mid bR \mid \varepsilon$,

$S \rightarrow bSc \mid Sc \mid \varepsilon \}$. $\diamondsuit$

(iv)

b.

First of all , it is obvious that all strings over this language are non-empty since start symbol S has two transitions which include at least one a or one b . Now , we will check whether these strings have equal number of a's and b's or not by induction .

Our inductive hypotesis is that the number of a's and b's in strings that are generated by the language S is equal , strings that are generated by A have one more a's than number of b's and similarly strings that are generated by B have one more b's than number of a's .

Base Case :

S has two transitions , aB and bA . Both B and A have three transitions and the shortest string that we can obtain is 'ab' or 'ba' in which the number of a's and b's is equal to each other . So , there is no way to generate a string with length 1 .

Inductive Step :

We can assume that our inductive hypotesis is valid for every string with length l or smaller than l . Now we will analyze that strings generated by S with length l+1 .

Choose the transition S $\rightarrow$ aB . According to the inductive hypotesis ; strings that are generated by B have one more b's than number of a's . And we have one 'a' independent from B . It follows that number of a's and b's becomes equal .

Just like the previous one , when we choose the transition S $\rightarrow$ bA , According to the inductive hypotesis ; strings that are generated by A have one more a's than number of b's . And we have one 'b' independent from B . It follows that number of a's and b's becomes equal .

Our inductive hypotesis holds for all cases that we analyze .

As we have shown inductively , the language L(G) is the set of all non-empty strings over {a,b} in which numbers of occurrences ofa and b are equal . $\diamondsuit$

Answer 2

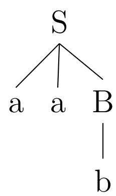
a.

I choose the string 'aab' which has two leftmost derivations .Derivation for the first tree :
$D_1 = \mathrm{S} \Rightarrow \mathrm{Ab} \Rightarrow \mathrm{Aab} \Rightarrow \mathrm{aab}$. $\diamondsuit$
Derivation for the second tree :
$D_2 = \mathrm{S} \Rightarrow \mathrm{aaB} \Rightarrow \mathrm{aab}$. $\diamondsuit$

b.

The equivalent unambiguous CFG I constructed is G = (K,Σ,Δ,S) , where
K = {S,A,B,M} ,
Δ = {S → Ab | aaB ,
A → Ma | M ,
M → ε ,
B → b } . ♢

c.

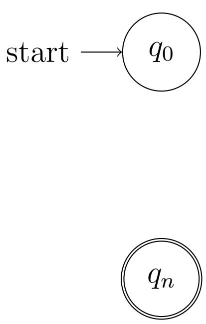


Answer 3

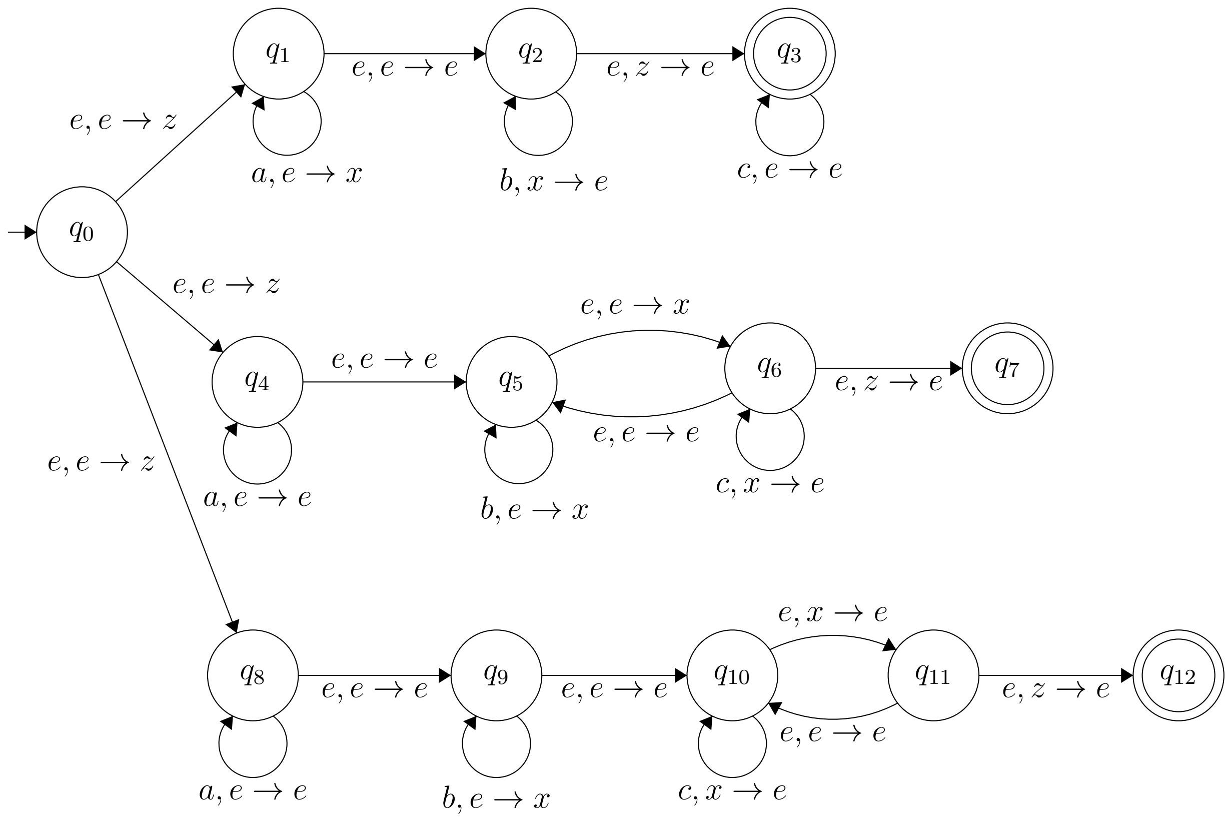
a.

Let's analyze this PDA and stack used in this PDA . For every two a's , we push a x into the stack . For the triangular part of the PDA , we pop an x for every three b's . But it is also possible that we don't take a's and go straight from the state q_1 , this follows that we don't take any b's . So the language that is generated by this PDA is:
K = $\{a^{2k}b^{3k}$ | k $\geq$ 0}

b.



c.


q_0
q_1
q_2
q_3
q_4
q_5
q_6
q_7
q_8
q_9
q_{10}
q_{11}
q_{12}
$e, e \to z$
$e, e \to z$
$e, e \to z$
$a, e \to x$
$e, e \to e$
$b, x \to e$
$e, z \to e$
$c, e \to e$
$a, e \to e$
$e, e \to e$
$b, e \to x$
$e, e \to x$
$e, e \to e$
$c, x \to e$
$e, z \to e$
$a, e \to e$
$e, e \to e$
$b, e \to x$
$e, e \to e$
$c, x \to e$
$e, x \to e$
$e, e \to e$
$e, z \to e$

(i)

(ii)

State	Input	Stack	Transition
q_0	aabcc	e	-
q_4	aabcc	z	(q_0,e,e),(q_4,z)
q_4	abcc	z	(q_4,a,e),(q_4,e)
q_4	bcc	z	(q_4,a,e),(q_4,e)
q_5	bcc	z	(q_4,e,e),(q_5,e)
q_5	cc	xz	(q_5,b,e),(q_5,x)
q_6	cc	xxz	(q_5,e,e),(q_6,x)
q_6	c	xz	(q_6,c,x),(q_6,e)
q_6	e	z	(q_6,c,x),(q_6,e)
q_7	e	e	(q_6,e,z),(q_7,e)
			Accepts.

State	Input	Stack	Transition
First Way :			
q_0	bac	e	-
q_1	bac	z	(q_0,e,e),(q_1,z)
q_2	bac	z	(q_1,e,e),(q_2,e)
q_2	ac	xz	(q_2,b,x),(q_2,e)
			Rejects.(not possible to choose a after b)
Second Way :			
q_0	bac	e	-
q_4	bac	z	(q_0,e,e),(q_4,z)
q_5	bac	z	(q_4,e,e),(q_5,e)
q_5	ac	xz	(q_5,b,x),(q_5,e)
			Rejects.(not possible to choose a after b)
Third Way :			
q_0	bac	e	-
q_8	bac	z	(q_0,e,e),(q_8,z)
q_9	bac	z	(q_8,e,e),(q_9,e)
q_9	ac	xz	(q_9,b,x),(q_9,e)
			Rejects.(not possible to choose a after b)

Answer 4

a.

The pushdown automaton that corresponds to the grammar G is :
P = ({p,q} , Σ , V , Δ , p , {q}) , where
Δ = {((p,e,e) , (q,E)),
((q,e,E) , (q,E+T)),
((q,e,E) , (q,T)),
((q,e,T) , (q,TxF)),
((q,e,T) , (q,F)),
((q,e,F) , (q,a)),
((q,e,F) , (q,(E))),
((q,a,a) , (q,e))
((q,+,+) , (q,e))
((q,x,x) , (q,e))
((q,(,() , (q,e)),
((q,),)) , (q,e))} . ♢

b.

We have the transition representation $(q_i, \upsilon, \beta)(q_j, \gamma)$. There are three ways to use these transitions :

1) We can push one item , pop anything (empty) . $\rightarrow \gamma = 1$, $\beta = 0$. $|\gamma| + |\beta| = 1$,
2) We can push anything (empty) , pop one item . $\rightarrow \gamma = 0$, $\beta = 1$. $|\gamma| + |\beta| = 1$,
3) We can push anything (empty) , pop anything (empty) . $\rightarrow \gamma = 0$, $\beta = 0$. $|\gamma| + |\beta| = 0$,

As it can obviously be seen in every case , $|\gamma| + |\beta| \leq 1$. $\diamondsuit$

Answer 5

a.

(i) We have the language { $a^m b^{m+n} a^n$ $\in$ {a,b}* | m,n $\in$ N } . It is obviously equal to $a^m b^m b^n a^n$ which is again equal to a concatenation of two languages : $a^m b^m$ and $b^n a^n$. If both parts are context-free languages , the whole part will be a context-free language since context-free languages are closed under concatenation according to closure properties . We already know that the language $a^n b^n$ is context-free from the text book and the lectures . Since both parts are in the same form , i.e. $a^n b^n$, they are both context-free . According to the closure properties , it follows that the language { $a^m b^{m+n} a^n$ $\in$ {a,b}* | m,n $\in$ N } is a context-free language . $\diamondsuit$

(ii) The language { a,b }* - L is exactly equal to the complement of L ,over the alphabet {a,b} of course . When we try to design a pushdown automaton that generates L , we conclude that there is no possible way to construct this PDA . It follows that the language L is not context-free . According to the closure properties , context-free languages are not closed under complementation . Therefore , since L is not a context-free language , the complement of L , { a,b }* - L is a context-free language . $\diamondsuit$

b.

(i) Let's use L_1 as the name of this language and p as the parameter of pumping lemma . First of all , we assume that L_1 is a context-free language . We can choose the string $a^{p^2} b^p$ $\in L_1$ for pumping.

Now , we will analyze the possible positions of vxy . We must satisfy the conditions for pumping lemma : $a^{p^2} b^p$ = uvxyz , |vxy|$\leq$ p and |vy|$\geq$1 . Let's analyze the different cases :

1) v belongs to a^{p^2} and y belongs to b^p :

We have v = a^s and y = b^t and as we have the conditions |vxy|$\leq$ p and |vy|$\geq$1 , there must be the equation 1$\leq$s+t$\leq$p . In contrast , if we pump this string , the equation 1$\leq$s+t$\leq$p will not hold anymore since s+t becomes larger and larger when we compare it with p . And the resulting string will not satisfy the conditions to be included in L_1 .

2) v and y belong to a^{p^2} :

When we try to pump v and y continuously , the string that we will get have the type $a^{p^2+r}b^p$ where r $\geq$ 1 .The resulting string will clearly not be in the language L_1 .

3) v and y belong to b^p :

Similar to the previous case , when we try to pump down v and y continuously , the string that we will get have the type $a^{p^2}b^{p-r}$ where r $\geq$ 1 .The resulting string will clearly not be in the language L_1 .

4) v belongs to a^{p^2} and b^p at the same time :

In this case , when we pump the string , it is obvious that the order of a's and b's will be mixed and the string will not be included in L_1 . Same result is valid for y , too .

Since we have reached contradictions in every case , the language L_1 is not a context-free language . $\diamondsuit$

(ii) Let's use L_2 as the name of this language and assume another language K = $L_2 \cap$ ab*ab*ab* . If our language L_2 is context-free , this follows that K is context-free , too. Now, our purpose is to show that K is not context-free . Let's consider w = ab^nab^nab^n . According to pumping lemma , we need to have ab^nab^nab^n = uvxyz ,|vxy|$\leq$ p and |vy|$\geq$1 where p is pumping length . Clearly , v and y parts will not include any a's since if they include a's , after pumping the string , there will be more than 3 a's which makes the string not belong to the language K . So v any y will be in one of the three b parts . Let's consider possible cases :

1) If they are included in the same b part (first,second or third), after we pump the string , that b part will have more than b's than other two b parts . This follows that the string does not belong to K .

2) If they are included in two different b parts (first-second or second-third) , similar to the previous case , when we pump the string , two parts will have the same number of b's but the remaining part will have less b's than those two . This follows that the string does not belong to K.

Since we have shown that the language K is not context-free , we conclude that the language L_2 is not context-free . $\diamondsuit$

Answer 6

(i) (T/F)? F

(ii) (T/F)? T

(iii) (T/F)? T

(iv) (T/F)? F