Student Information

Full Name : Furkan Göksel
Id Number : 2237436

Answer 1

a.

(i) $G = (\{S, T, a, b\}, \{a, b\}, R, S)$ where R as follows
R=$\{S \rightarrow aSa \mid bT,$
$T \rightarrow aTa \mid b\}$

(ii) $G = (\{S, S_1, S_2, S_a, S_b, a, b\}, \{a, b\}, R, S)$ where R as follows
$R = \{S \rightarrow S_1 \mid S_2,$
$S_1 \rightarrow aS_a a,$
$S_a \rightarrow aS_a a \mid bS_a b \mid aS_a b \mid bS_a a \mid a,$
$S_2 \rightarrow bS_b b,$
$S_b \rightarrow aS_b a \mid bS_b b \mid aS_b b \mid bS_b a \mid b\}$

(iii) 4 cases, $i > j$ (any number of c), $j < i$ (any number of c), $j > k$ (any number of a), $j < k$ (any number of a)
$G = (\{S, S_1, S_2, S_3, S_4, A, C, , a, b, c\}, \{a, b, c\}, R, S)$ where R as follows
$R = \{S \rightarrow S_1C \mid S_2C \mid AS_3 \mid AS_4,$
$S_1 \rightarrow aS_1 \mid aS_1b \mid a,$
$S_2 \rightarrow S_2b \mid aS_2b \mid b,$
$S_3 \rightarrow bS_3 \mid bS_3c \mid b,$
$S_4 \rightarrow S_4c \mid bS_4c \mid c,$
$A \rightarrow aA \mid e,$
$C \rightarrow cC \mid e\ \}$

(iv) $G = (\{S, T, F, a, b, c\}, \{a, b, c\}, R, S)$ where R as follows
$S \rightarrow FT \mid T$
$F \rightarrow aF \mid bF \mid cF \mid c$
$T \rightarrow aTa \mid bTb \mid cc \mid Fc$
(T generates ww^R (reverse part), cc terminal means only one word (think with $S \rightarrow T$ rule), Fc for case of reverse part belongs to w_1 (first string), and together with $S \rightarrow FT$ it gives general case.)

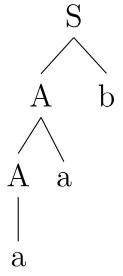
b.

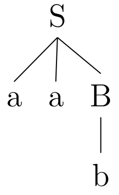
Our language have equal number of occurences of a and b. In order to prove that this grammar generates this language, we should consider each nonterminal and their behaviours. Let's get started with S it gives us either aB or bA, to make these uncomplete strings in the language, for the first case nonterminal B should give us strings that have one more b's than a's. Similarly for the second case nonterminal A should give us strings that have one more a's than b's. If this is the case then we can say that this grammar can generate the given language. Firstly when we look first rule $A \to a$ we can see that A can be directly replaced by a. And when thinking the start case which is expecting strings that one more a's than b's, this rule holds. When we look at another rule which is $A \to aS$ and thinking the start case (bA) our string will be baS and this rule provides us that left substring of the whole string is in the language, and we can start generate new substrings. For nonterminal B, it also have similar rule ($B \to b, B \to bS$) and these two rules also provides same feature strings but instead of a version it gives b version (What I mean by this is producing one more b's than a's etc.). With these 6 rules we can always generate random order strings for instance baabbaabba like, but still these rules cannot generate consecutive a's and b's such as aaabbb, and this string also in the language. And to produce such strings these two rules are in the grammar $A \to BAA$, $B \to ABB$. Think that I want to generate bbbaaa, derivation will be like this $S \to bA \to bBAA \to bBBAAA \to bbBAAA \to bbbAAA \to bbbaAA \to bbbaaA \to bbbaaa$. Also this rule holds our need in the first state (need one more a's or one more b's string in the start to be in the language). When we look that BAA, it gives us baa or when we replace A with again BAA, this will be BBAAA, or BABAA, or when we use the rule $B \to ABB$, this will be ABBAA, all in these cases we can generate one more a's than b's, and our consecutive order problem can be solved. This case is similar for $B \to ABB$ rule it will always produce one more B's than A's and also provides new order possibilities. Therefore with this grammar we can generate strings that have equal numbers of occurrences of a and b, and in any order.

Answer 2

a.

Our string is aab.
$D_1 : S \implies Ab \implies Aab \implies aab$
$D_2 : S \implies aaB \implies aab$



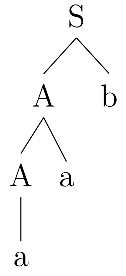


b.

This grammar generates a language such that each string ends with b. And nonterminal B causes the ambiguity. So when we remove it from grammar, our new grammar becomes unambiguous.
G({S,A,a,b},{a,b},R,S) where R as follows
R= $\{S \to Ab,\ A \to Aa \mid a\}$

c.

$D_1 : S \implies Ab \implies Aab \implies aab$



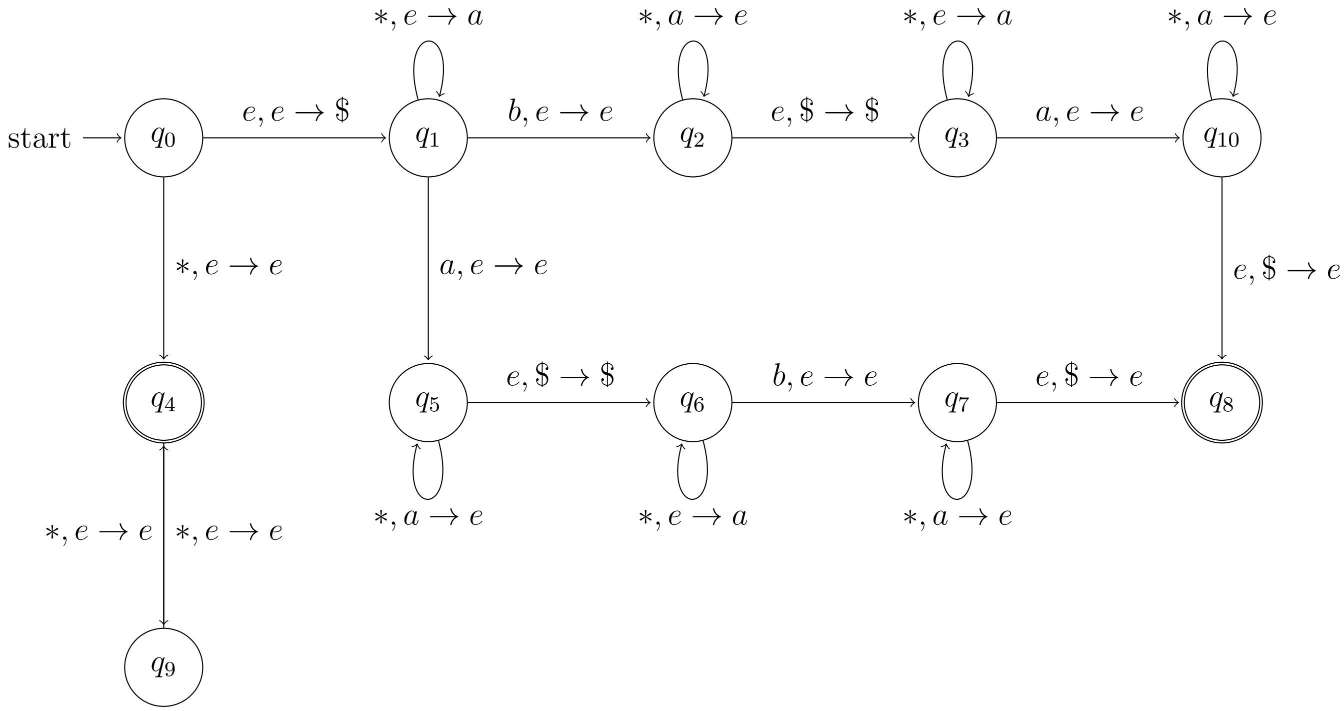
Answer 3

a.

In this PDA, we put a # symbol to indicate that stack is empty or not. Then, in q_1 and q_2 states we put a x to stock for each two a's. Then nondeterministically we go to q_3 state and in this state we pop a x from stock for each three b's, and then when the top of the stack is # which indicates that stack is empty, we go to final state. As it can be understood the flow the language generated by this PDA is
L(M) = $\{a^i b^j \in \{a,b\}^* \mid i,j \in N,\ 3i = 2j\}$

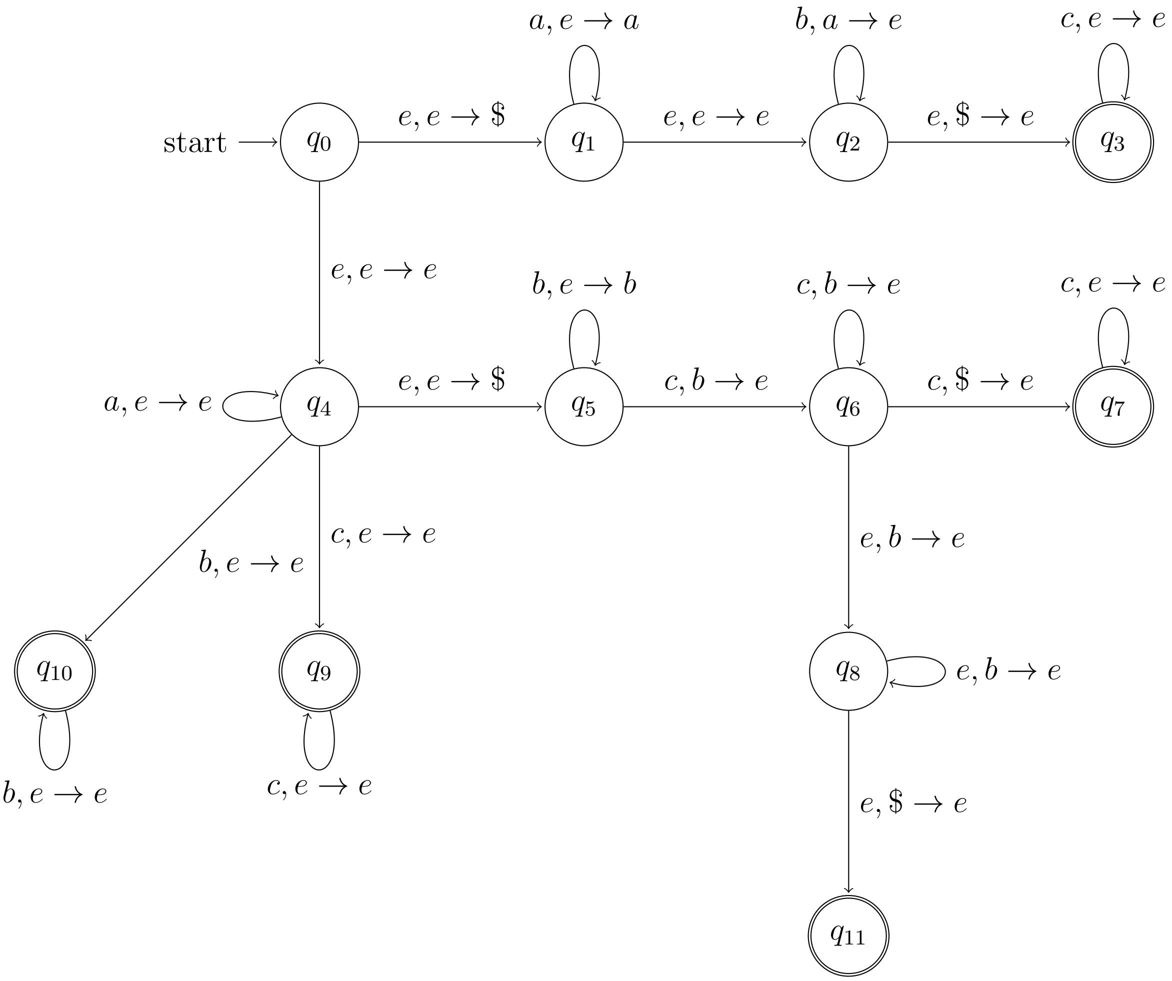
b.

When I was constructing the machine, I constructed it based on these thoughts:
1-Any odd length word is accepted,
2-For even length words, With the help of nondeterminism, when we halved the string, at least one letter of each part will be different, and it is enough to determine string is accepted or not.


start
q_0
$e, e \to \$$
q_1
$*, e \to a$
$b, e \to e$
q_2
$*, a \to e$
$e, \$ \to \$$
q_3
$*, e \to a$
$a, e \to e$
q_{10}
$*, a \to e$
$*, e \to e$
$a, e \to e$
$e, \$ \to e$
q_4
q_5
$e, \$ \to \$$
q_6
$b, e \to e$
q_7
$e, \$ \to e$
q_8
$*, a \to e$
$*, e \to a$
$*, a \to e$
$*, e \to e$
$*, e \to e$
q_9

c.

(i)



In

In the diagram, q_{10} and q_9 states are for a*b,a*bb,a*bbbb,.. or a*c,a*cc,a*ccc,.. inputs which means number of b is 0, but number of c is greater than or equal to 1 and vice versa.

(ii)

State	Input	Stack	Transition
q_0	aabcc	e	-
q_4	aabcc	e	e,e $\rightarrow$ e
q_4	abcc	e	a,e $\rightarrow$ e
q_4	bcc	e	a,e $\rightarrow$ e
q_5	bcc	\$	e,e $\rightarrow$ \$
q_5	cc	b\$	b,e $\rightarrow$ b
q_6	c	\$	c,b $\rightarrow$ e
q_7	e	e	c,\$ $\rightarrow$ e
			Accepts.

State	Input	Stack	Transition
q_0	bac	e	-
q_1	bac	\$	e,e $\rightarrow$ \$
q_2	bac	\$	e,e $\rightarrow e$
q_3	bac	e	e,\$ $\rightarrow e$
			Rejects.
q_0	bac	e	-
q_4	bac	e	e,e $\rightarrow e$
q_{10}	ac	e	b,e $\rightarrow e$
			Rejects.
q_0	bac	e	-
q_4	bac	e	e,e $\rightarrow e$
q_5	bac	\$	e,e $\rightarrow$ \$
q_5	ac	b\$	b,e $\rightarrow b$
			Rejects.

Answer 4

a.

According to Lemma 3.4.1, we construct a PDA from the given grammar with applying these rules

1-((p,e,e),(q,S))
2-((q,e,A),(q,x)) for each rule $A \rightarrow x$ in R
3-((q,a,a),(q,e)) for each $a \in \Sigma$

So,
$M = (\{p, q\}, \Sigma, V, \Delta, p, \{q\})$ where our Δ is
$\Delta = \{((p, e, e), (q, E)),$
$((q, e, E), (q, E + T)),$
$((q, e, E), (q, T)),$
$((q, e, T), (q, TxF)),$
$((q, e, T), (q, F)),$
$((q, e, F), (q, (E))),$
$((q, e, F), (q, a)),$
$((q, a, a), (q, e)),$
$((q, +, +), (q, e)),$
$((q, x, x), (q, e)),$
$((q, (, (), (q, e)),$
$((q,),)), (q, e))\}$

b.

In order to make the language recognized by M and the language recognized by M' equal, we maintain the transition' semantics.So, for the first PDA if all rules such that $((q_i, u, \beta), (q_j, \gamma)) \in \Delta, |\beta| + |\gamma| \leq 1$ are exactly in the Δ' and the states q_i and q_j's are also in the K', then for transitions such that $((q_i, u, \beta), (q_j, \gamma)) \in \Delta, |\beta| + |\gamma| > 1$ we should create intermediate states to obey the rule $|\beta| + |\gamma| \leq 1$. So we will handle these transitions such that:

1) Our β is equal to $\beta_1\beta_2\beta_3...\beta_n$ where $|\beta_i| \leq 1$, and in the transition $((q_i, u, \beta), (q_j, \gamma))$ this β will be popped from stack, so firstly we add some transitions to Δ' such that $((q_i, e, \beta_1), (q_x, e))$ and then $((q_x, e, \beta_{i+1}), (q_x, e))$ $1 \leq i \leq n-1$ and with this way all β will be popped from stack, then we can add this transition to complete the old transition $((q_x, u, e), (q_j, \gamma))$ where q_x is a new intermediate state, and if length of γ is greater than 1 than we consider the second case. With this way, we popped β from stack element by element, and these transitions obey the rule $|\beta| + |\gamma| \leq 1$.

2)Our γ is equal to $\gamma_1\gamma_2\gamma_2...\gamma_n$ where $|\gamma_i| \leq 1$, and in the transition $((q_i, u, \beta), (q_j, \gamma))$ this γ will be pushed to stack, so we add some transitions to Δ' such that $((q_i, e, e), (q_y, \gamma_n))$ and $((q_y, e, e), (q_y, \gamma_{i-1}))$ $1 < i \leq n$ and (n-1,n-2,n-3,..,2,1) , lastly to complete the transition, $((q_y, u, \beta), (q_j, e))$. With this way, we complete the transition $((q_i, u, \beta), (q_j, \gamma))$ by generating new intermediate state and add some new transitions with obeying the rule. If both of the strings' length is greater than 1 these two case can be applied by using first 1 and then 2. So, with this way all transitions in M can be transformed such transitions and can be added to M'.

Answer 5

a.

(i) Context free languages are closed under concatenation, we can prove that by introducing new grammar such that A and B are context free languages and AB = L(G) where G=(V, Σ, R, S) where $V = V_A \cup V_B \cup \{S\}$, $\Sigma = \Sigma_A \cup \Sigma_B$, R = $R_A \cup R_B \cup \{S \to S_A S_B\}$. Hence context free languages are closed under concatenation. So in this example $L_1 = a^m b^m \mid m \in N$ is context free language because we can construct a grammar for it such that $G_1 = (\{S, a, b\}, \{a, b\}, R, S)$ where R is $R = \{S \to aSb \mid e\}$ and this is similar for $L_2 = b^n a^n \mid n \in N$ $G_1 = (\{S, a, b\}, \{a, b\}, R, S)$ where R is $R = \{S \to bSa \mid e\}$. $G_2 = (\{S, a, b\}, \{a, b\}, R, S)$,So given language is concatenation of L_1 and L_2 since $a^m b^m b^n a^n = a^m b^{m+n} a^n$ and it is equal to $L_1 L_2$ (which means concatenation of two context free language) and as we proved, it is also a context free language.

(ii) Context free languages are closed under union, we can prove that by introducing new grammar such that A and B are context free languages and $A \cup B = L(G)$ where G = (V, Σ, R, S) where $V = V_A \cup V_B \cup \{S\}$, $\Sigma = \Sigma_A \cup \Sigma_B$, R = $R_A \cup R_B \cup \{S \to S_A \mid S_B\}$. Hence context free languages are closed under union. And then, complement of the given language consists of such strings; strings starts with a, strings includes at least one bb (consecutive), and strings include $ba^x ba^y$ $\mid x + 1 \neq y$ so each of this can be represented as; $a\Sigma^* \cup \Sigma^* bb \Sigma^* \cup \Sigma^* \{ba^x ba^y \mid x + 1 \neq y\} \Sigma^*$.

Regular languages are proper subset of context free languages, so they are also context free languages ($\Sigma^*, a\Sigma^*, \Sigma^*bb\Sigma^*$), also $\{ba^xba^y \mid x+1 \neq y\}$ is a context free language because we can write it's grammar like $\{ba^xba^y \mid x+1 \neq y\} = \{baa^xba^y \mid x \neq y\}$=$\{baa^xba^y \mid x > y\} \cup \{baa^xba^y \mid x < y\}$ so $G = (\{S, a, b, S_1, S_2, S_x\}, \{a, b\}, R, S)$ where
R = $\{S \rightarrow baS_x$,
$S_x \rightarrow aS_1 \mid S_2a$,
$S_1 \rightarrow aS_1 \mid aS_1a \mid b$,
$S_2 \rightarrow S_2a \mid aS_2a \mid b\}$
So $\{ba^xba^y \mid x+1 \neq y\}$ is context free language, hence this language becomes union of context free languages, and since context free languages are closed under union, given language is context free language.

b.

(i) Assume that L is Context Free Language, then let's say p is our pumping constant such that $p \geq 1$, and we choose a string in the language w such that w=$a^{p^2}b^p$ since $p^2 \leq p^2$ this string in the language and $p^2 + p^2 \geq p$ so we need to show that by picking i and for all possible decomposition of this string $uv^ixy^iz \notin L$, also these decompositions have some constraints such that $vy \neq e$ and $|vxy| \leq p$, so with this constraint there are 3 possible decompositions of this string.

Case 1:
uvxy is in a's part and z in b part. Then we can decompose the string like $z = b^p$ and $uvxy = a^{p^2}$ and when we pump v and y i times we get such a string $a^{p^2+k}b^p$ and this string is not in the language since $p^2 + k > p^2$.

Case 2:
u is in a's part and vxyz in b part. Then we can decompose the string like $u = a^{p^2}$ and $vxyz = b^p$ and when we pump v and y 0 times we get such a string $a^{p^2}b^{p-k}$ and this string is not in the language since $p^2 > (p-k)^2$.

Case 3:
v is in a part y is in b part($u = a^k$ and $y = b^m$). Since they are in vxy part, using the constraint $1 \leq k+m \leq p$. so when we pump u and v 0 times it the string becomes $a^{p^2-k}b^{p-m}$, lets check that whether this string is in the language or not.

$$\begin{aligned} (p-m)^2 &\leq (p-1)^2 \quad \text{since } m \geq 1 \\ (p-m)^2 &\leq p^2 - 2p + 1 \\ (p-m)^2 &< p^2 - k \quad \text{since } k \leq p \text{ from } |vxy| \leq p \end{aligned}$$

Hence this string is also not in the language. THerefore our first assumption is wrong and by using pumping lemma theorem this language is not context free language.

(ii) Assume that L is Context Free Language, then let's say p is our pumping constant such that $p \geq 1$, and we choose a string in the language w such that w=$a^pb^pa^pb^pa^pb^p$. We need to show that by picking i and for all possible decomposition of this string $uv^ixy^iz \notin L$, also these decompositions have some constraints such that $vy \neq e$ and $|vxy| \leq p$. So let's check all the cases.

Case1:

Assume that uvxy in the first a's part. Then the rest of the string is in z. When we pump u and v i times we get such a string $a^{p+k}b^pa^pb^pa^pb^p$ and this string is not in the language.

Case2:

Assume that vxy in the first b's part. Then first a's is in u and the rest of the string is in z. When we pump u and v i times we get such a string $a^pb^{p+k}a^pb^pa^pb^p$ and this string is not in the language.

Case3:

Assume that vxy in the second a's part. Then first a^pb^p in u and the rest of the string is in z. When we pump u and v i times we get such a string $a^pb^pa^{p+k}b^pa^pb^p$ and this string is not in the language.

Case4:

Assume that vxy in the second b's part. Then $a^pb^pa^p$ is in u and the rest of the string is in z. When we pump u and v i times we get such a string $a^pb^pa^pb^{p+k}a^pb^p$ and this string is not in the language.

Case5:

Assume that vxy in the third a's part. Then $a^pb^pa^pb^p$ is in u and the rest of the string is in z. When we pump u and v i times we get such a string $a^pb^pa^pb^pa^{p+k}b^p$ and this string is not in the language.

Case6:

Assume that vxyz in the second b's part. Then rest of the string is in u. When we pump u and v i times we get such a string $a^pb^pa^pb^pa^pb^{p+k}$ and this string is not in the language.

Case7:

Assume that uvxy in first a^pb^p part and rest of the string is in z. Then when we pump u and v i times we get $a^{p+k}b^{p+m}a^pb^pa^pb^p$ and this string is not in the language.

Case8:

Assume that vxy in second a^pb^p part and then u is in the first a^pb^p and rest of the string is in z. Then when we pump u and v i times we get $a^pb^pa^{p+k}b^{p+m}a^pb^p$ and this string is not in the language.

Case9:

Assume that vxyz in last a^pb^p part and then u is the rest of the string. Then when we pump u and v i times we get $a^pb^pa^pb^pa^{p+k}b^{p+m}$ and this string is not in the language.

vxy part cannot be in aba form because $|vxy| \leq p$ so all possible decompositions are these, and given language is not context free.

Answer 6

(i) (F)

(ii) (T)

(iii) (T)

(iv) (F)