1. For the PDA defined by $(Q,\Sigma,\Gamma,\delta,q_0,F)$, which of the following defines the mapping associated with the transition function δ?
(3 points)
   1. $Q\times\Sigma_\varepsilon\times F \rightarrow Q\times\Gamma_\varepsilon$
   2. $Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon \rightarrow Q\times\Gamma_\varepsilon$
   3. ✓ $Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon \rightarrow P(Q\times\Gamma_\varepsilon)$
   4. $Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon \rightarrow P(Q\times\Sigma_\varepsilon)$
   5. $Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon \rightarrow P(\Sigma_\varepsilon\times\Gamma_\varepsilon)$

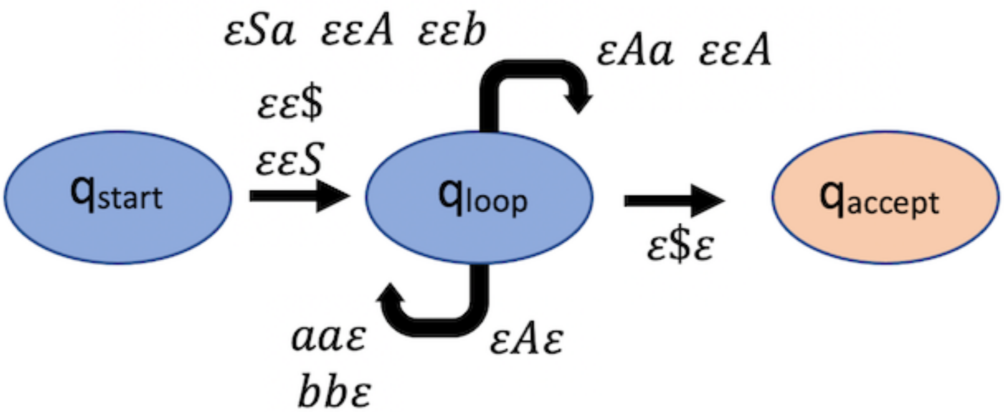
2. Which of the following notations for a PDA stack operation defines a transition without reading input and without popping any symbol from the stack?
(3 points)

1. 1 0 ε
2. 0 ε 0
3. ε ε $
4. 0 $ ε

3. Which of the following notations for a PDA stack pushes a symbol onto the stack?
(3 points)

1. 1 0 ε
2. ✓ 0 ε 0
3. ε $ ε
4. 0 $ ε

1. Consider the PDA below that recognizes strings generated by the CFG G with rules S→bAa; A→Aa|ε.

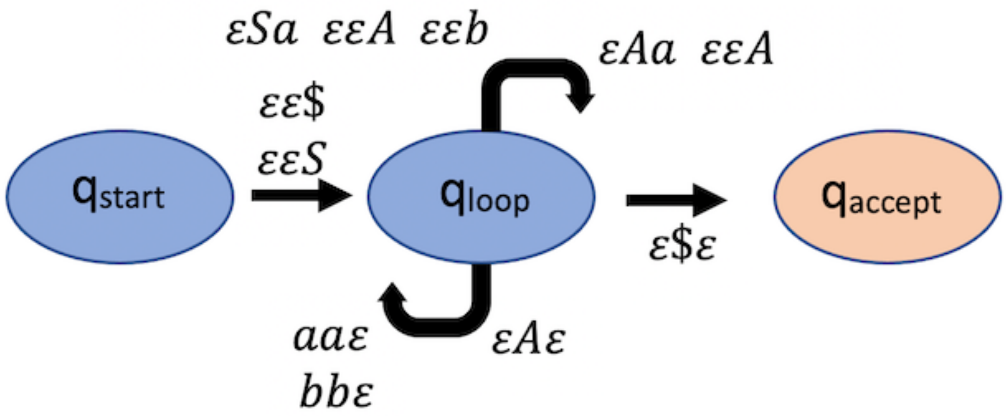


Which of the following is the correct list of stack elements (leftmost is top) when the **b** is matched from the input **baa**.

(3 points)

✓ 1. b A a $
2. $ a b A
3. A a b $
4. b a A $

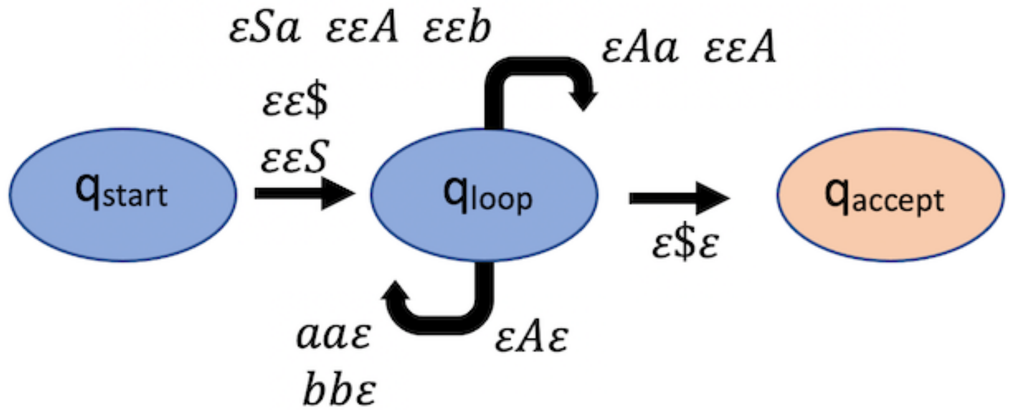
2. Consider the PDA below that recognizes strings generated by the CFG G with rules S→bAa; A→Aa|ε.



Which of the following is the correct list of stack elements (leftmost is top) when the first **a** is matched from the input **baa**.
(3 points)

1. b A a $
2. a a A $
3. a A A $
4. ✓ a a $

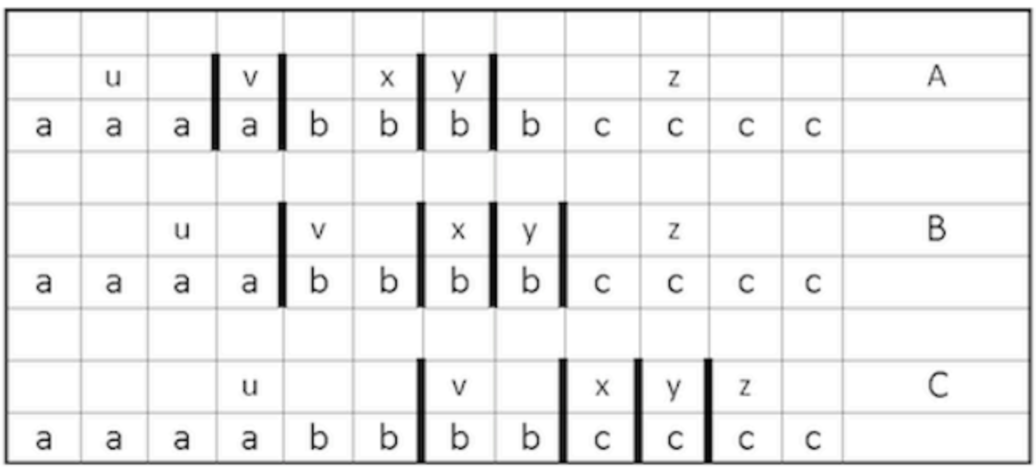
3. Consider the PDA below that recognizes strings generated by the CFG G with rules S→bAa; A→Aa|ε.



Which of the following is the correct list of stack elements (leftmost is top) when the second **a** is matched from the input **baa**. *(3 points)*

1. b a a $
2. a $ ✓
3. A a $
4. a A $

1. Consider $C=\{a^i b^j c^k \mid 0 \le i \le j \le k\}$ with pumping length 4. For which of the partitions of $s=a^4b^4c^4=uvxyz$ would the string uv^2xy^2z be in C?



(3 points)

1. A
2. B
3. C
4. None ✓

2. Consider $C=\{a^i b^j c^k \mid 0 \le i \le j \le k\}$ with pumping length 4. For which of the partitions of $s=a^4b^4c^4 = uvxyz$ would the string uv^0xy^0z be in C?

	u		v		x	y			z			A
a	a	a	a	b	b	b	b	c	c	c	c	
		u		v		x	y		z			B
a	a	a	a	b	b	b	b	c	c	c	c	
			u			v		x	y	z		C
a	a	a	a	b	b	b	b	c	c	c	c	

(3 points)

✓ 1. A

2. B

3. C

4. None

3. Consider C={$a^i b^j c^k$|0≤i≤j≤k} with pumping length 4. For which of the partitions of $s=a^4b^4c^4$ =uvxyz would the string uvxyz be in C?

	u		v		x	y			z			A
a	a	a	a	b	b	b	b	c	c	c	c	
		u		v		x	y		z			B
a	a	a	a	b	b	b	b	c	c	c	c	
			u			v		x	y	z		C
a	a	a	a	b	b	b	b	c	c	c	c	

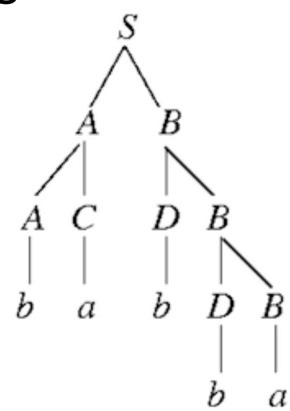
(3 points)

1. A
2. B
3. C
4. All of them ✓

1. Which of the following is a CFL? Assume $\Sigma=\{0,1\}$.
(3 points)
   A. {1^n0^n1^n|n≥0}
   B. {0^n10^n10^n|n≥0}
   ✓ C. {0^n10^n|n≥0}
   D. {0^n1^n0^n|n≥0}

ANS: C
PTS: 3

2. Suppose G is a CFG in CNF form with Σ={a,b} and the following rules (R): S→AB, A→AC|b, B→DB|a, C→a, D→b. Assume that S is the start variable. Below is the 5-level parse tree for the string **babba** using G:



How many levels would the parse tree for **baba** have?

(3 points)

A. 6

B. 5

✓ C. 4

D. 3

ANS: C
PTS: 3

3. Given a grammar in CNF, what is the minimum number of levels that a parse tree can have when a string of length 2^n is derived? Assume that the tree is full, that is, every nonterminal node has two nonterminal children except those on the next-to-lowest level in which case each nonterminal has only one terminal as a child.
(3 points)

✓ A. n+2

B. n+1

C. n

D. n/2

ANS: A
PTS: 3

1. Which step of CNF conversion for CFGs was executed below?

Before:	After:
$S_0 \rightarrow S$	$S_0 \rightarrow S$
$S \rightarrow AS \mid ASB \mid SB \mid S$	$S \rightarrow AS \mid ASB \mid SB \mid S$
$A \rightarrow aAS \mid aS \mid a$	$A \rightarrow aAS \mid aS \mid a$
$B \rightarrow SbS \mid A \mid bb$	$B \rightarrow SbS \mid bb \mid aAS \mid aS \mid a$

(3 points)

A. Step 1 (New start variable)

B. Step 2 (Remove ε rules)

✓ C. Step 3 (Remove unit rules)

D. Step 4 (Final conversion)

2. Which step of CNF conversion for CFGs was executed below?

(Before)	(After)
$S_0 \rightarrow S$	$S_0 \rightarrow S$
$S \rightarrow ASB \mid SB$	$S \rightarrow AS \mid ASB \mid SB \mid S$
$A \rightarrow aAS \mid aS \mid a$	$A \rightarrow aAS \mid aS \mid a$
$B \rightarrow SbS \mid A \mid \varepsilon \mid bb$	$B \rightarrow SbS \mid A \mid bb$

(3 points)

A. Step 1 (New start variable)

✓ B. Step 2 (Remove ε rules)

C. Step 3 (Remove unit rules)

D. Step 4 (Final conversion)

3. How many right-hand-sides are not in proper CNF form before Step 4 is applied?

$$S_0 \rightarrow AS|ASB|SB$$
$$S \rightarrow AS|ASB|SB$$
$$A \rightarrow aAS|aS|a$$
$$B \rightarrow SbS|bb|aAS|a$$

(3 points)

A. 5

B. 6

✓ C. 7

D. 8

1. Which of the following is the correct updated rule for replacing the nonterminal S after the A -> ε rule is removed from the following CFG: S -> AS|bA, A -> B|ε, B -> b ?

(3 points)

A. S -> AS | bA | A

B. S -> AS | b

C. S -> S | b

✓ D. S -> AS | S | bA | b

2. Which of the following is the correct updated rule for replacing the nonterminal S after the B -> ε rule is removed from the following CFG: S -> ASB|BAB, B -> b|ε, A -> Aa|ε ?

(3 points)

✓ A. S -> ASB | BAB | AS | AB | BA | A

B. S -> ASB | BAB | AS | AB | BA

C. S -> ASB | BAB | AB | BA

D. S -> ASB | BAB | AS | S | AB | BA

3. Which of the following is the correct conversion of the grammar rule S -> SAA | bA to CNF?

(3 points)

A. S -> A_1A|BA, A_1->AS, B->b

B. S -> A_1A|bA, A_1->AS

✓ C. S -> A_1A|BA, A_1->SA, B->b

D. S -> A_1|AB, A_1->SA, B->b

Theorem: Any CFL is generated by a CFG in CNF.

Proof. **Stage 1** – Add a new start symbol S_0 and rule $S_0 \to S$, where S was the original start variable (do not want S in the *rhs* of any rule).

Stage 2 – Eliminate all ϵ-rules.
Repeat ... (until all ϵ-rules are removed):

1. Eliminate the ϵ-rule $A \to \epsilon$, where A is not the start variable.

2. For each occurrence of A on the *rhs* of a rule, add a new rule with that occurrence of A deleted.

3. Replace the rule $B \to A$ (if present) by $B \to A \mid \epsilon$ unless the rule $B \to \epsilon$ has not been previously eliminated.

Example: To delete $A \to \epsilon$, replace $B \to uAv$ by $B \to uAv \mid uv$; replace $B \to uAvAw$ by $B \to uAvAw \mid uvAw \mid uAvw \mid uvw$.

Stage 3 – Remove all unit rules.
Repeat ... (until all unit rules are removed):

1. Remove a unit rule $A \to B$.

2. For each rule $B \to u$ that appears, add the rule $A \to u$, unless it was a previously-removed unit rule (u can be a string of variables and terminals).

Stage 4 – Convert all remaining rules.
Repeat ... (until no rules of the form $A \to u_1 u_2 \ldots u_k$ with $k \geq 3$ remain):

1. Replace a rule $A \to u_1 u_2 \ldots u_k$, $k \geq 3$, where each u_i, $1 \leq i \leq k$, is a variable or a terminal, by $A \to u_1 A_1$, $A_1 \to u_2 A_2, \ldots, A_{k-2} \to u_{k-1} u_k$, where $A_1, A_2, \ldots, A_{k-2}$ are **new variables**.

2. If $k \geq 2$, replace any terminal u_k with a new variable U_i and add the rule $U_i \to u_k$.

$\square$

Step 1 – new start variable.

$S_0 \to S$
$S \to ASA \mid ab$
$A \to B \mid S$
$B \to b \mid \epsilon$

Step 2 – remove ϵ rules.

Remove $B \to \epsilon$

$S_0 \to S$
$S \to ASA \mid aB$
$A \to B \mid S$
$B \to b$

Remove $A \to \epsilon$

$S_0 \to S$
$S \to ASA \mid aB \mid a \mid AS \mid SA \mid S$
$A \to B \mid S$
$B \to b$

Step 3 – remove unit rules.

Remove $S \to S$

$S_0 \to S$
$S \to ASA \mid aB \mid a \mid SA \mid AS$
$A \to B \mid S$
$B \to b$

Remove $S_0 \to S$

$S_0 \to ASA \mid aB \mid a \mid SA \mid AS$
$S \to ASA \mid aB \mid a \mid SA \mid AS$
$A \to B \mid S$
$B \to b$

Remove $A \to B$

$S_0 \to ASA \mid aB \mid a \mid SA \mid AS$
$S \to ASA \mid aB \mid a \mid SA \mid AS$
$A \to S \mid b$
$B \to b$

Remove $A \to S$

$S_0 \to ASA \mid aB \mid a \mid SA \mid AS$
$S \to ASA \mid aB \mid a \mid SA \mid AS$
$A \to b \mid ASA \mid aB \mid a \mid SA \mid AS$
$B \to b$

Step 4 – convert remaining rules.

$S_0 \to AA_1 \mid UB \mid a \mid SA \mid AS$

$S \to AA_1 \mid UB \mid a \mid SA \mid AS$

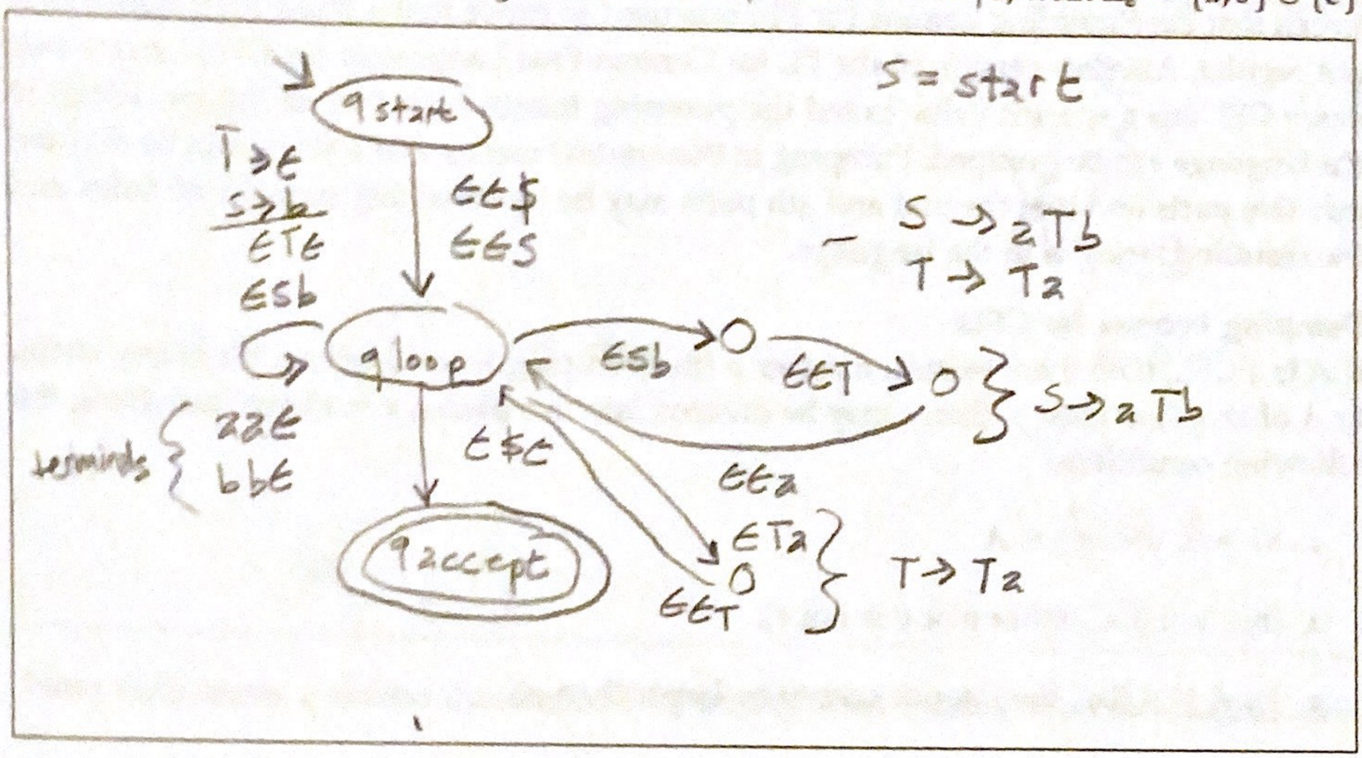
$A \to b \mid AA_1 \mid UB \mid a \mid SA \mid AS$

$A_1 \to SA$

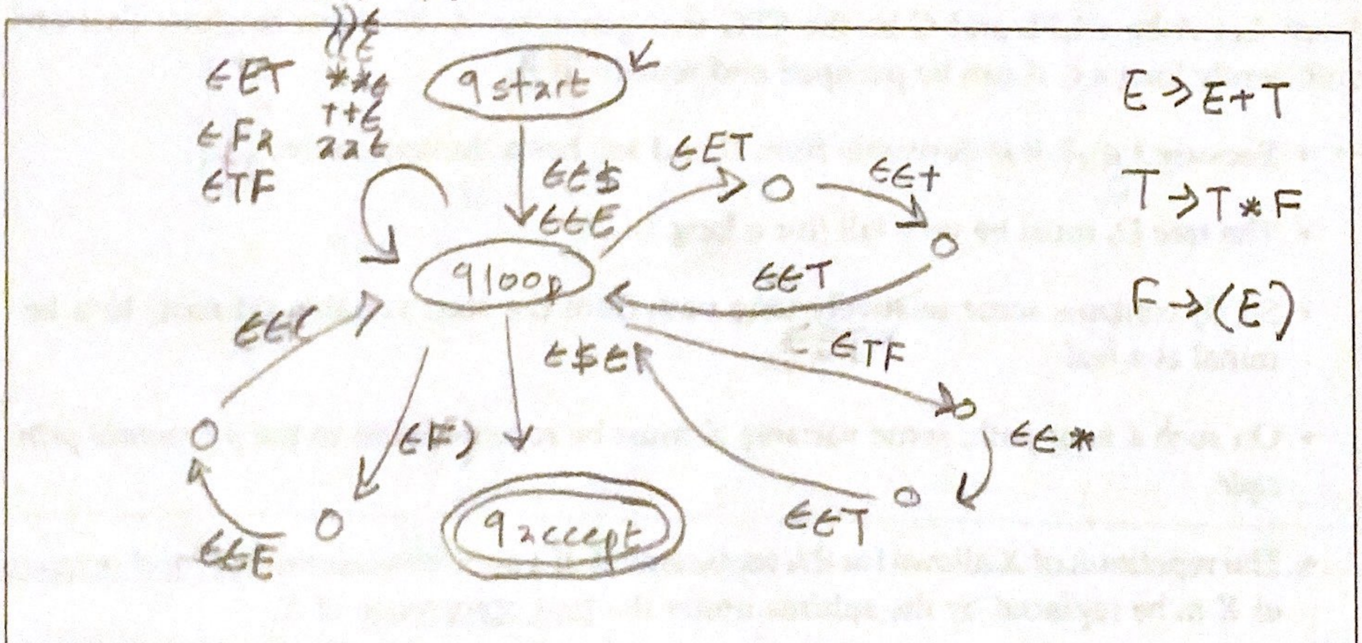
$U \to a$

$B \to b$

Example: Let's draw the state diagram for the PDA that would recognize the language generated by the CFG G_1 having rules $S \to aTb \mid b$ and $T \to Ta \mid \epsilon$, with $\Sigma_\epsilon = \{a,b\} \cup \{\epsilon\}$.



Example: Let's draw the state diagram for the PDA that would recognize the language generated by the CFG G_2 having rules $E \to E + T \mid T$, $T \to T * F \mid F$, and $F \to (E) \mid a$, with $\Sigma_\epsilon = \{a, +, *, (,)\} \cup \{\epsilon\}$.



CS312

Homework #6 Answer Key

1 CNF Step 1

Perform step one of converting the following CFG into CNF by adding a new start state S. $V = \{A, B\}$, $\Sigma = \{0, 1, \epsilon\}$, $S = A$, $R =$

$$
\begin{aligned}
A &\rightarrow BAB \mid B \mid 1 \mid \epsilon \\
B &\rightarrow 00 \mid \epsilon
\end{aligned}
$$

Answer

$V = \{S, A, B\}$, $\Sigma = \{0, 1\}$, $S = S$, $R =$

$$
\begin{aligned}
S &\rightarrow A \\
A &\rightarrow BAB \mid B \mid 1 \mid \epsilon \\
B &\rightarrow 00 \mid \epsilon
\end{aligned}
$$

2 CNF Step 2

Perform step two of converting the following CFG's into CNF by removing ϵ rules. **No points off if $C \rightarrow C$ rule left in for part (a)**.

2.a

$V = \{S, A, B, C\}$, $\Sigma = \{a, b, c\}$, $S = S$, $R =$

$$
\begin{aligned}
S &\rightarrow A \\
A &\rightarrow AaB \\
B &\rightarrow b \mid C \mid \epsilon \\
C &\rightarrow CC \mid c \mid \epsilon
\end{aligned}
$$

Answer

$V = \{S, A, B, C\}$, $\Sigma = \{a, b, c\}$, $S = S$, $R =$

$$\begin{aligned}
S &\rightarrow A \\
A &\rightarrow AaB \mid Aa \\
B &\rightarrow b \mid C \\
C &\rightarrow CC \mid c
\end{aligned}$$

2.b

$V = \{S, A, B\}$, $\Sigma = \{a, b, \epsilon\}$, $S = S$, $R =$

$$\begin{aligned}
S &\rightarrow A \\
A &\rightarrow AA \mid AB \mid B \mid a \\
B &\rightarrow BB \mid b \mid \epsilon
\end{aligned}$$

Answer

$V = \{S, A, B\}$, $\Sigma = \{a, b\}$, $S = S$, $R =$

$$\begin{aligned}
S &\rightarrow A \mid \epsilon \\
A &\rightarrow AA \mid AB \mid B \mid a \\
B &\rightarrow BB \mid b
\end{aligned}$$

3 CNF Step 3

Perform step three of converting the following CFG's into CNF by removing unit rules. **No points off if $A \to BC$ rule left in for part (b)**.

3.a

$V = \{S, A, B\}$, $\Sigma = \{a, b\}$, $S = S$, $R =$

$$\begin{aligned} S &\to A \\ A &\to AA \mid AB \mid A \mid B \mid aB \\ B &\to BB \mid Bb \mid b \end{aligned}$$

Answer

$V = \{S, A, B\}$, $\Sigma = \{a, b\}$, $S = S$, $R =$

$$\begin{aligned} S &\to AA \mid AB \mid BB \mid Bb \mid b \mid aB \\ A &\to AA \mid AB \mid BB \mid Bb \mid b \mid aB \\ B &\to BB \mid Bb \mid b \end{aligned}$$

3.b

$V = \{S, A, B, C, D\}$, $\Sigma = \{a, b, c\}$, $S = S$, $R =$

$$\begin{aligned} S &\to A \mid \epsilon \\ A &\to BC \\ B &\to BD \mid bb \\ C &\to CD \mid cc \\ D &\to B \mid C \end{aligned}$$

Answer

$V = \{S, B, C, D\}$, $\Sigma = \{a, b, c\}$, $S = S$, $R =$

$$\begin{aligned} S &\rightarrow BC \mid \epsilon \\ B &\rightarrow BD \mid bb \\ C &\rightarrow CD \mid cc \\ D &\rightarrow BD \mid bb \mid CD \mid cc \end{aligned}$$

4 CNF Step 4

Perform step four of converting the following CFG into CNF by removing remaining rules.

$V = \{S, A, B\}$, $\Sigma = \{a, b\}$, $S = S$, $R =$

$$\begin{aligned} S &\rightarrow AAB \mid aBb \mid ABB \mid Ab \\ A &\rightarrow AAB \mid aBb \mid ABB \mid Ab \\ B &\rightarrow BB \mid Bb \mid b \end{aligned}$$

Sample Answer

$V = \{S, A, B\}$, $\Sigma = \{a, b\}$, $S = S$, $R =$

$$\begin{aligned} S &\rightarrow AE \mid FC \mid EB \mid AC \\ A &\rightarrow AE \mid FC \mid EB \mid AC \\ B &\rightarrow BB \mid BC \mid b \\ C &\rightarrow b \\ D &\rightarrow a \\ E &\rightarrow AB \\ F &\rightarrow DB \end{aligned}$$

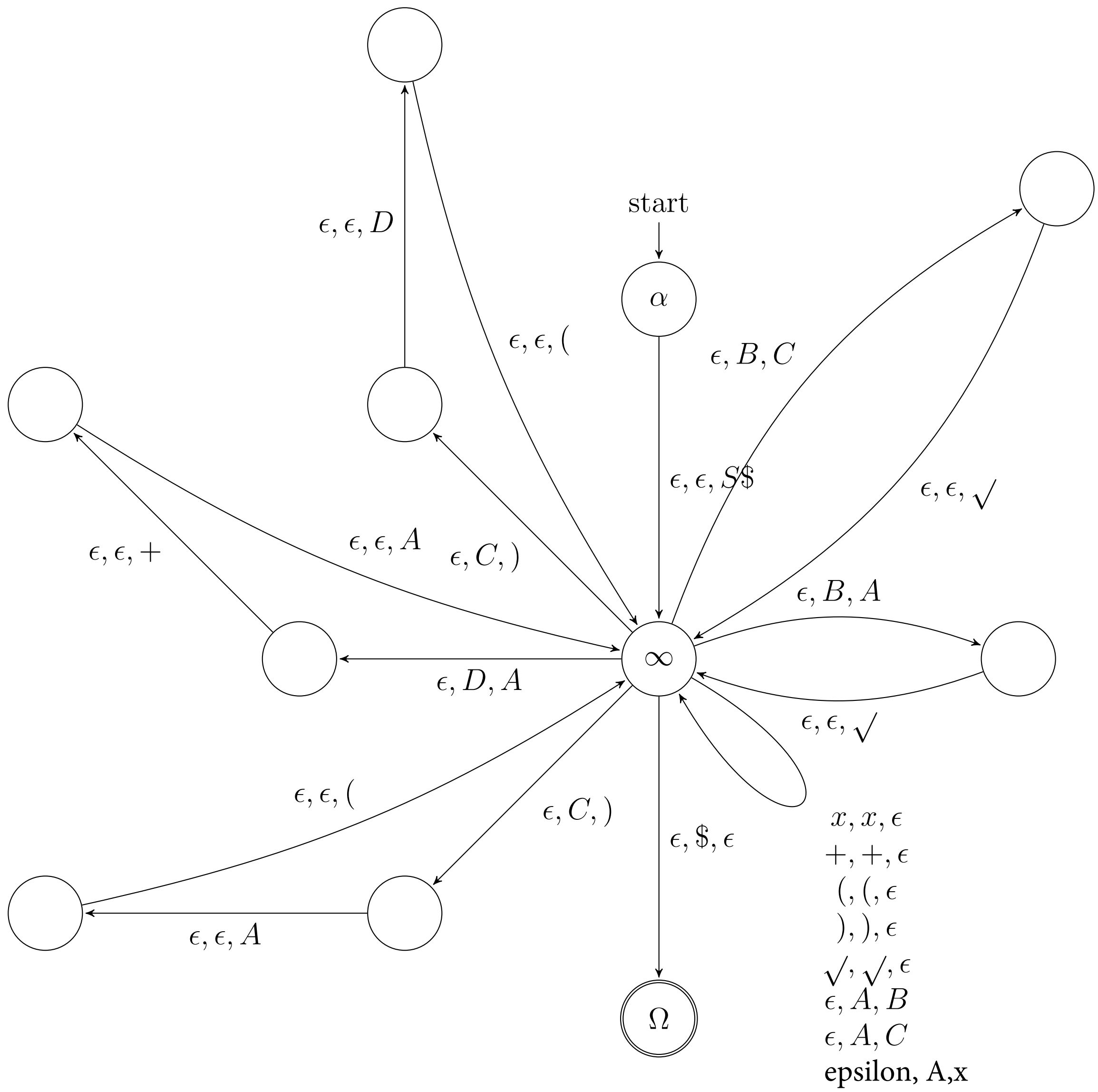
5 CFG to PDA Conversion

Using the technique that was covered in class, convert the following CFG to a PDA:

$V = \{A, B, C, D\}$,$\Sigma_\epsilon = \{x, \surd, +, (,)\} \cup \{\epsilon\}$, $S = A$,$R =$

$$
\begin{aligned}
A &\rightarrow B \mid C \mid x \\
B &\rightarrow \surd C \mid \surd A \\
C &\rightarrow (D) \mid (A) \\
D &\rightarrow A + A
\end{aligned}
$$

Answer



start
α
∞
Ω
ϵ, ϵ, D
$\epsilon, \epsilon, ($
ϵ, B, C
$\epsilon, \epsilon, S\$$
$\epsilon, \epsilon, \surd$
$\epsilon, \epsilon, +$
ϵ, ϵ, A
$\epsilon, C,)$
ϵ, B, A
ϵ, D, A
$\epsilon, \epsilon, \surd$
$\epsilon, \epsilon, ($
$\epsilon, C,)$
$\epsilon, \$, \epsilon$
ϵ, ϵ, A
x, x, ϵ
$+, +, \epsilon$
$(, (, \epsilon$
$),), \epsilon$
$\surd, \surd, \epsilon$
ϵ, A, B
ϵ, A, C
epsilon, A,x

COSC 312

Homework #5 Answer Key

1

Produce a context-free grammar (CFG) for each of the following languages, assuming $\Sigma = \{0, 1\}$: **(showing just the rules suffices)**

1.a

$\{w \mid w$ starts and ends with different symbols$\}$
Example Answer: $(S \to 0A1 \mid 1A0; A \to 0A \mid 1A \mid \epsilon)$

1.b

$\{w \mid$ the length of w is an integer multiplier of 3$\}$
Example Answer: $(S \to 0T \mid 1T \mid \epsilon; T \to 0U \mid 1U; U \to 0S \mid 1S)$

1.c

$\{ww^R \mid$ i.e., a word followed by that word reversed$\}$
Example Answer: $(S \to 0S0 \mid 1S1 \mid \epsilon)$

2

Let $G = (\{S, A, B, C, D, Z\}, (0, 1), R, S)$,
where $R = \{S \to A \mid C \mid Z; A \to 01B \mid 0A \mid \epsilon; B \to 1B \mid 10A; C \to 10D \mid 1C \mid \epsilon; D \to 01C \mid 0D; Z \to 0Z1 \mid \epsilon\}$.

2.a

Describe the language L (in English) that is generated by the CFG G.

Answer: L(G) is the language of either an equal number of disjoint 01 and 10 substrings or the language of n 0's followed by n 1's. By disjoint we mean 0110 rather than 010.

2.b

Prove that the language L generated by the CFG G is **not** regular. **Hint**: use the fact that regular languages are closed under union and prove that one component of the language is not regular by the P

Answer:

Prove that the language $L(G)$ is not a regular language.

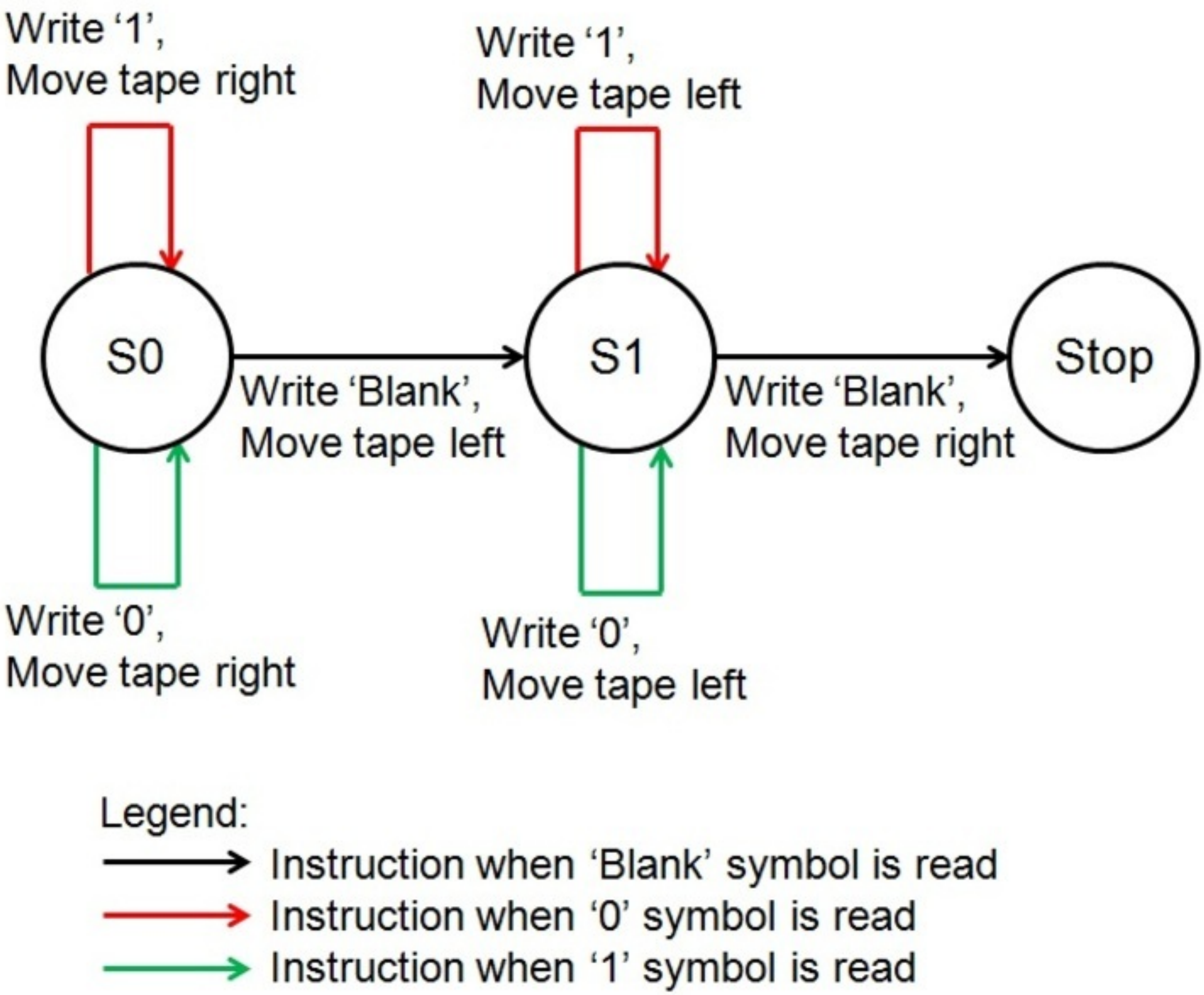
Proof. 1. Let L(G) be the language generated by the CFG G.

2. Assume L(G_1) is the language generated by the rule set $\{S \to Z; Z \to 0Z1 \mid \epsilon\}$
3. Assume L(G_2) is the language generated by the rule set $\{S \to A \mid C; A \to 01B \mid 0A \mid \epsilon; B \to 1B \mid 10A; C \to 10D \mid 1C \mid \epsilon; D \to 01C \mid 0D\}$
4. If we can show that either L(G_1) or L(G_2) is not regular, then L(G) would not be regular by the union closure of regular languages.
5. Assume L(G_1) is regular.
6. Let p be the pumping length of L(G_1).
7. Suppose we choose $s = 0^p1^p \in L(G_1)$ so that $|0^p1^p| > p$
8. By the pumping lemma, s can be partitioned as $s = xyz$ such that for all $i \geq 0, xy^iz \in L(G_1)$.
9. By condition 3 of the pumping lemma, s must be divided so that $|xy| \leq p$. Therefore y must contain only one or more 0's $\forall s = xyz$.
10. Let $s' = xy^2z \in L(G_1)$ by choosing $i = 2$.
11. $s' = 0^{p+|y|}1^p \notin L(G_1)$

Therefore, by this contradiction $L(G_1)$ is not regular, and subsequently $L(G)$ must not be regular due to the union closure of regular languages. $\square$

Example: Let's revisit our TM M_1 that tests the membership of the language $L_1 = \{w\#w \mid w \in \{0,1\}^*\}$.

S_0	If symbol read is a 0 or 1, replace it by x and remember the symbol as a; if the symbol is a # go to S_5; else **reject**. $\delta(S_0, 0) = (S_1(0), x, R)$, $\delta(S_0, 1) = (S_1(1), x, R)$, $\delta(S_0, \#) = (S_5, \#, R)$
$S_1(a)$	Move right until a # is found; if no # is found before *blank*, **reject**. $\delta(S_1(a), 0) = (S_1(a), 0, R)$, $\delta(S_1(a), 1) = (S_1(a), 1, R)$ $\delta(S_1(a), \#) = (S_2(a), \#, R)$ // change state
$S_2(a)$	Move right until a 0 or 1 is found; if current symbol is the same as a, replace if by x; else **reject**. $\delta(S_2(a), x) = (S_2(a), x, R)$, $\delta(S_2(0), 0) = (S_3, x, L)$ $\delta(S_2(1), 1) = (S_3, x, L)$ // change state
S_3	Move left until a # is found. $\delta(S_3, a) = (S_3, a, L)$, where $a \in \{0, 1, x\}$ $\delta(S_3, \#) = (S_4, \#, L)$ // change state
S_4	Move left until an x is found and go to S_0 $\delta(S_4, a) = (S_4, a, L)$, where $a \in \{0, 1\}$ $\delta(S_4, x) = (S_0, x, R)$ // change state
S_5	Move right until a 0, 1, or *blank* is found; **accept** if current symbol is a *blank*; **reject** if current symbol is 0 or 1. $\delta(S_5, a) = (S_5, a, R)$, where $a \in \{0, 1, x\}$ // stay in S_5 $\delta(S_5, blank) = (S_0, blank, L)$ // accept


Write ‘1’,
Move tape right
Write ‘0’,
Move tape right
S0
Write ‘Blank’,
Move tape left
S1
Write ‘1’,
Move tape left
Write ‘0’,
Move tape left
Write ‘Blank’,
Move tape right
Stop
Legend:
Instruction when ‘Blank’ symbol is read
Instruction when ‘0’ symbol is read
Instruction when ‘1’ symbol is read