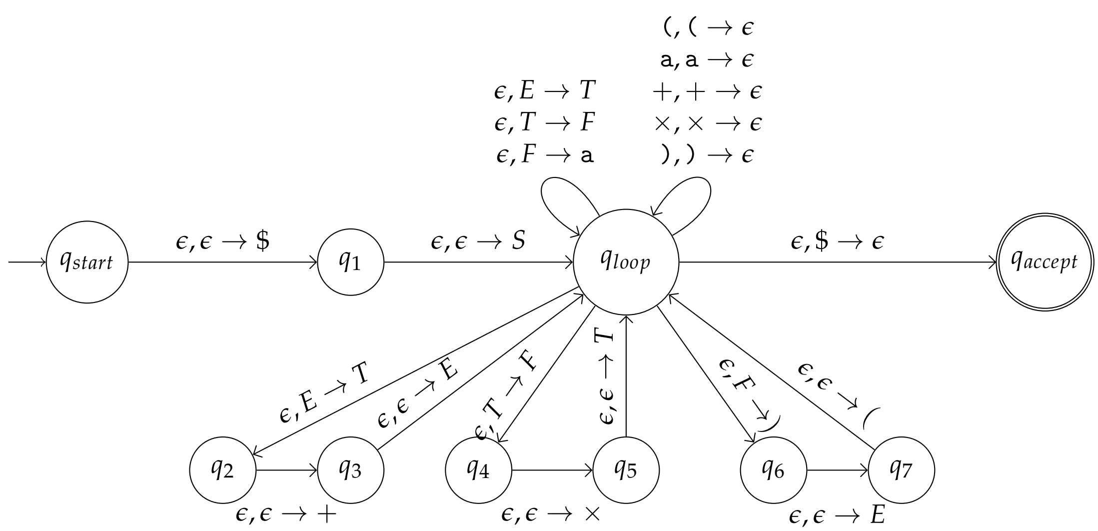
Homework 8

CMPS130 Computational Models, Spring 2015

2.11

$$
\begin{aligned}
M &= (Q, \Sigma, \Gamma, \delta, q_{start}, F) \\
Q &= \{q_{start}, q_{loop}, q_{accept}, q_1, q_2, q_3, q_4, q_5, q_6, q_7\} \\
\Sigma &= \{\texttt{(}, \texttt{a}, +, \times, \texttt{)}\} \\
\Gamma &= \{\texttt{(}, \texttt{a}, +, \times, \texttt{)}, E, T, F, \$\} \\
F &= \{q_{accept}\}
\end{aligned}
$$

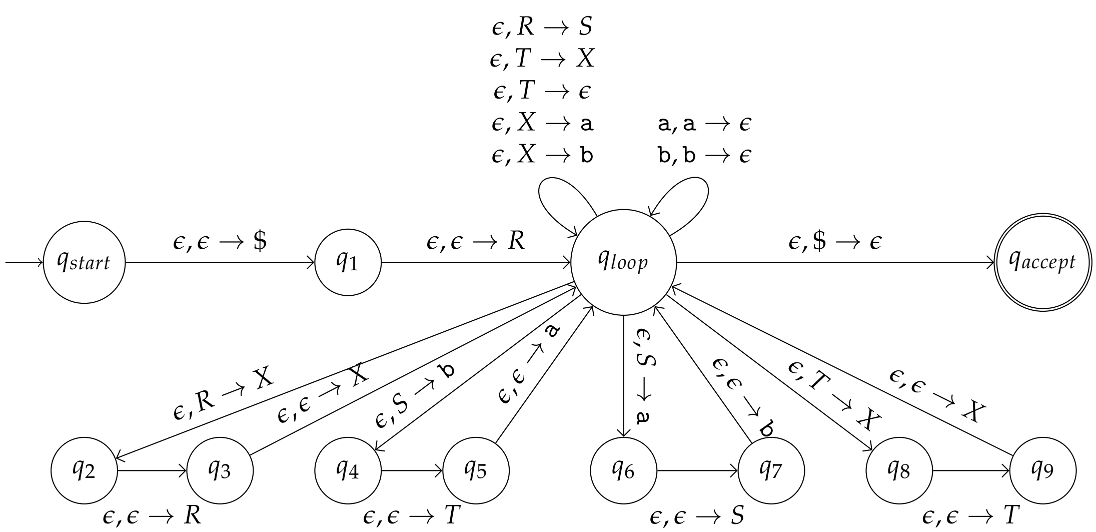
start	$\delta(q_{start}, \epsilon, \epsilon) = \{(q_1, \$)\}$	$\delta(q_1, \epsilon, \epsilon) = \{(q_{loop}, E)\}$	
$E \to E + T$	$\delta(q_{loop}, \epsilon, E) = \{(q_2, T)\}$	$\delta(q_2, \epsilon, \epsilon) = \{(q_3, +)\}$	$\delta(q_3, \epsilon, \epsilon) = \{(q_{loop}, E)\}$
$E \to T$	$\delta(q_{loop}, \epsilon, E) = \{(q_{loop}, T)\}$		
$T \to T \times F$	$\delta(q_{loop}, \epsilon, T) = \{(q_4, F)\}$	$\delta(q_4, \epsilon, \epsilon) = \{(q_5, \times)\}$	$\delta(q_5, \epsilon, \epsilon) = \{(q_{loop}, T)\}$
$T \to F$	$\delta(q_{loop}, \epsilon, T) = \{(q_{loop}, F)\}$		
$F \to \texttt{(}\ E\ \texttt{)}$	$\delta(q_{loop}, \epsilon, F) = \{(q_6, \texttt{)})\}$	$\delta(q_6, \epsilon, \epsilon) = \{(q_7, E)\}$	$\delta(q_7, \epsilon, \epsilon) = \{(q_{loop}, \texttt{(})\}$
$F \to \texttt{a}$	$\delta(q_{loop}, \epsilon, F) = \{(q_{loop}, \texttt{a})\}$		
consume `(`	$\delta(q_{loop}, \texttt{(}, \texttt{(}) = \{(q_{loop}, \epsilon)\}$		
consume `a`	$\delta(q_{loop}, \texttt{a}, \texttt{a}) = \{(q_{loop}, \epsilon)\}$		
consume $+$	$\delta(q_{loop}, +, +) = \{(q_{loop}, \epsilon)\}$		
consume $\times$	$\delta(q_{loop}, \times, \times) = \{(q_{loop}, \epsilon)\}$		
consume `)`	$\delta(q_{loop}, \texttt{)}, \texttt{)}) = \{(q_{loop}, \epsilon)\}$		
accept	$\delta(q_{loop}, \epsilon, \$) = \{(q_{accept}, \epsilon)\}$		



2.12

$$
\begin{aligned}
M &= (Q, \Sigma, \Gamma, \delta, q_{start}, F) \\
Q &= \{q_{start}, q_{loop}, q_{accept}, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9\} \\
\Sigma &= \{\texttt{a}, \texttt{b}\} \\
\Gamma &= \{\texttt{a}, \texttt{b}, R, S, T, X, \$\} \\
F &= \{q_{accept}\}
\end{aligned}
$$

start	$\delta(q_{start}, \epsilon, \epsilon) = \{(q_1, \$)\}$	$\delta(q_1, \epsilon, \epsilon) = \{(q_{loop}, R)\}$	
$R \to XRX$	$\delta(q_{loop}, \epsilon, R) = \{(q_2, X)\}$	$\delta(q_2, \epsilon, \epsilon) = \{(q_3, R)\}$	$\delta(q_3, \epsilon, \epsilon) = \{(q_{loop}, X)\}$
$R \to S$	$\delta(q_{loop}, \epsilon, R) = \{(q_{loop}, S)\}$		
$S \to \texttt{a}T\texttt{b}$	$\delta(q_{loop}, \epsilon, S) = \{(q_4, \texttt{b})\}$	$\delta(q_4, \epsilon, \epsilon) = \{(q_5, S)\}$	$\delta(q_5, \epsilon, \epsilon) = \{(q_{loop}, \texttt{a})\}$
$S \to \texttt{b}T\texttt{a}$	$\delta(q_{loop}, \epsilon, S) = \{(q_6, \texttt{a})\}$	$\delta(q_6, \epsilon, \epsilon) = \{(q_7, S)\}$	$\delta(q_7, \epsilon, \epsilon) = \{(q_{loop}, \texttt{b})\}$
$T \to X\ T\ X$	$\delta(q_{loop}, \epsilon, T) = \{(q_8, X)\}$	$\delta(q_8, \epsilon, \epsilon) = \{(q_9, T)\}$	$\delta(q_9, \epsilon, \epsilon) = \{(q_{loop}, X)\}$
$T \to X$	$\delta(q_{loop}, \epsilon, T) = \{(q_{loop}, X)\}$		
$T \to \epsilon$	$\delta(q_{loop}, \epsilon, T) = \{(q_{loop}, \epsilon)\}$		
$X \to \texttt{a}$	$\delta(q_{loop}, \epsilon, X) = \{(q_{loop}, \texttt{a})\}$		
$X \to \texttt{b}$	$\delta(q_{loop}, \epsilon, X) = \{(q_{loop}, \texttt{b})\}$		
consume a	$\delta(q_{loop}, \texttt{a}, \texttt{a}) = \{(q_{loop}, \epsilon)\}$		
consume b	$\delta(q_{loop}, \texttt{b}, \texttt{b}) = \{(q_{loop}, \epsilon)\}$		
accept	$\delta(q_{loop}, \epsilon, \$) = \{(q_{accept}, \epsilon)\}$		



2.14

Original grammar:

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

Adding rules for ϵ-production $A \to \epsilon$:

$$\begin{aligned} A &\to BAB \mid B \mid \epsilon \mid BB \\ B &\to 00 \mid \epsilon \end{aligned}$$

Adding rules for ϵ-production $B \to \epsilon$:

$$\begin{aligned} A &\to BAB \mid B \mid \epsilon \mid BB \mid AB \mid BA \\ B &\to 00 \mid \epsilon \end{aligned}$$

Adding rules for unit production $A \to B$:

$$\begin{aligned} A &\to BAB \mid B \mid \epsilon \mid BB \mid AB \mid BA \mid 00 \\ B &\to 00 \mid \epsilon \end{aligned}$$

No more productions to add for ϵ- and unit production. Remove all ϵ- and unit productions except for $A \to \epsilon$ (because A is the starting symbol):

$$\begin{aligned} A &\to BAB \mid BB \mid AB \mid BA \mid 00 \mid \epsilon \\ B &\to 00 \end{aligned}$$

Use new non-terminal N for terminal 0 used in rules:

$$\begin{aligned} A &\to BAB \mid BB \mid AB \mid BA \mid NN \mid \epsilon \\ B &\to NN \\ N &\to 0 \end{aligned}$$

Use new non-terminal A_2 to break up rules with more than two non-terminals:

$$\begin{aligned} A &\to BA_2 \mid BB \mid AB \mid BA \mid NN \mid \epsilon \\ A_2 &\to AB \\ B &\to NN \\ N &\to 0 \end{aligned}$$

The resulting grammar is in Chomsky-Normal-Form (CNF).

2.30

a.

Proof by Contradiction: Assuming that $\{0^n1^n0^n1^n \mid n \geq 0\}$ is context-free, then the pumping lemma holds. Therefore, there exists a pumping length $p \geq 1$, such that any string longer than p can be pumped.

The string $s = 0^p1^p0^p1^p$ is longer than p, and therefore can be split into $uvxyz$ such that $|vxy| \leq p$, $|vy| > 0$ and uv^ixy^iz is in the language.

Since $|vxy| \leq p$, the string vxy contains either completely out of 0's or 1's or spans one change of symbols, so is either of the shape 0^*1^* or 1^*0^*. In each of these cases, the pumped string $uvvxyyz$ ($i = 2$) will be $0^a1^b0^c1^d$ where no more than two of the variables a, b, c or d are greater than p and is therefore not in the language. Due to these contradictions, the pumping lemma does not hold and therefore $\{0^n1^n0^n1^n \mid n \geq 0\}$ cannot be context-free.

b.

Proof by Contradiction: Assuming that $\{0^n\#0^{2n}\#0^{3n} \mid n \geq 0\}$ is context-free, then the pumping lemma holds. Therefore, there exists a pumping length $p \geq 1$, such that any string longer than p can be pumped.

The string $s = 0^p\#0^{2p}\#0^{3p}$ is longer than p, and therefore can be split into $uvxyz$ such that $|vxy| \leq p$, $|vy| > 0$ and uv^ixy^iz is in the language.

Neither v nor y can contain #, otherwise $uvvxyyz$ ($i = 2$) would contain more than two #'s. The pumped string $uvvxyyz$ ($i = 2$) will be $0^a\#0^b\#0^c$ where one of the 0 sequences a, b or c will not be pumped, so the resulting string is not in the language. Due to these contradictions, the pumping lemma does not hold and therefore $\{0^n\#0^{2n}\#0^{3n} \mid n \geq 0\}$ cannot be context-free.

c.

Proof by Contradiction: Assuming that $\{w\#t \mid w \text{is a substring of } t, \quad w,t \in \{a,b\}^*\}$ is context-free, then the pumping lemma holds. Therefore, there exists a pumping length $p \geq 1$, such that any string longer than p can be pumped.

The string $s = a^pb^p\#a^pb^p$ is longer than p, and therefore can be split into $uvxyz$ such that $|vxy| \leq p$, $|vy| > 0$ and uv^ixy^iz is in the language.

Neither v nor y can contain #, otherwise $uvvxyyz$ ($i = 2$) would contain more than one #. If v is on the left of # and y is either also on the left or empty, then $uvvxyyz$ ($i = 2$) would result in a longer w for the same t and not a substring anymore. If v is on the right and y also on the right or empty, then uxz ($i = 0$) would result in a shorter t for the same w which would not be a substring anymore. Finally, if both v and y are non-empty and on different sides of the #, then v consists of b's and y of a's since $|vxy| \leq p$, so $uvvxyyz$ ($i = 2$) would not be in the language. Due to these contradictions, the pumping lemma does not hold and therefore $\{w\#t \mid w \text{is a substring of } t, \quad w,t \in \{a,b\}^*\}$ cannot be context-free.

d.

Proof by Contradiction: Assuming that $\{t_1\#t_2\#...\#t_k \mid k \geq 2,$ each $t_i \in \{\mathtt{a},\mathtt{b}\}^*$, and $t_i = t_j$ for some $i \neq j\}$ is context-free, then the pumping lemma holds. Therefore, there exists a pumping length $p \geq 1$, such that any string longer than p can be pumped.

The string $s = \mathtt{a}^p\mathtt{b}^p\#\mathtt{a}^p\mathtt{b}^p$ is longer than p, and therefore can be split into $uvxyz$ such that $|vxy| \leq p$, $|vy| > 0$ and uv^ixy^iz is in the language.

Neither v nor y can contain #, otherwise uxz ($i = 0$) would contain no #'s. If v and y are on the same side or one of them is empty, then $uvvxyyz$ ($i = 2$) would result in t_1 and t_2 to have different lengths. If both v and y are non-empty and on different sides of the #, then v consists of b's and y of a's since $|vxy| \leq p$, so $uvvxyyz$ ($i = 2$) would not be in the language. Due to these contradictions, the pumping lemma does not hold and therefore $\{t_1\#t_2\#...\#t_k \mid k \geq 2,$ each $t_i \in \{\mathtt{a},\mathtt{b}\}^*$, and $t_i = t_j$ for some $i \neq j\}$ cannot be context-free.

2.31

Proof by Contradiction: Assuming that

$$B = \{w \in \{\mathtt{0},\mathtt{1}\}^* \mid w = w^R \text{ , and } w \text{ has an equal number of } \mathtt{0}'s \text{ and } \mathtt{1}'s.\}$$

is context-free, then the pumping lemma holds. Therefore, there exists a pumping length $p \geq 1$, such that any string longer than p can be pumped.

The string $s = \mathtt{0}^p\mathtt{1}^{2p}\mathtt{0}^p$ is longer than p, and therefore can be split into $uvxyz$ such that $|vxy| \leq p$, $|vy| > 0$ and uv^ixy^iz is in the language.

Here, v and y together need to have the same number of 0's and 1's, otherwise $uvvxyyz$ ($i = 2$) would not be in the language. Due to $|vxy| \leq p$, v and y can include 1's either from the left or the right sequence of 1's but not from both. However, in that case, $uvvxyyz$ ($i = 2$) would increase the number of 1's only on one side, so the resulting string would not be a palindrome anymore. Due to these contradictions, the pumping lemma does not hold and therefore $\{t_1\#t_2\#...\#t_k \mid k \geq 2,$ each $t_i \in \{\mathtt{a},\mathtt{b}\}^*$, and $t_i = t_j$ for some $i \neq j\}$ cannot be context-free.

2.32

Proof by Contradiction: Assuming that

$$C = \{w \in \{\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4}\}^* \mid w \text{ has an equal number of 1's and 2's, and of 3's and 4's}\}$$

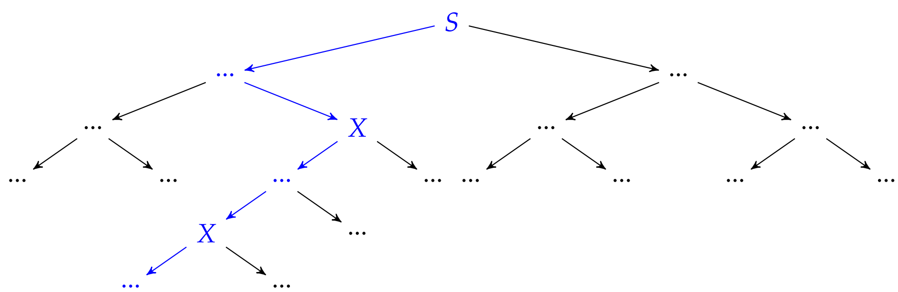
is context-free, then the pumping lemma holds. Therefore, there exists a pumping length $p \geq 1$, such that any string longer than p can be pumped.

The string $s = \mathtt{1}^p\mathtt{3}^p\mathtt{2}^p\mathtt{4}^p$ is longer than p, and therefore can be split into $uvxyz$ such that $|vxy| \leq p$, $|vy| > 0$ and uv^ixy^iz is in the language.

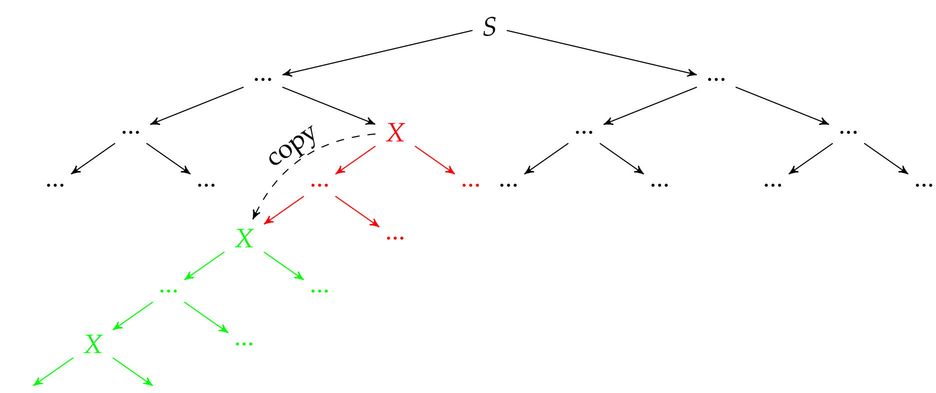
Due to $|vxy| \leq p$, v and y together can include either 1's and 3's, 3's and 2's, 2's and 4's or a just single kind of symbol. In each case, the pumped string $uvvxyyz$ ($i = 2$) has either a different number of 1's and 2's or a different number of 3's and 4's. Due to these contradictions, the pumping lemma does not hold and therefore $\{t_1\#t_2\#...\#t_k \mid k \geq 2,$ each $t_i \in \{\mathtt{a},\mathtt{b}\}^*$, and $t_i = t_j$ for some $i \neq j\}$ cannot be context-free.

2.35

Proof by Construction: Let s be a string generated with more than 2^b steps in a CNF grammar G with b variables. Since G is in Chomsky Normal Form, the derivation for s is a binary tree in which all leaf nodes are terminal symbols and all other nodes are non-terminals which are part of a production. Therefore, if there are at least $2^b + 1$ production steps, the tree has at least $2^b + 1$ non-leaf nodes. A full binary tree with n nodes has a height h of at least $h \geq \lceil \log_2(n+1) \rceil$. With $n = 2^b + 1$, it follows that $h \geq b + 1$. Since a full binary tree is the most compact shape, any non-complete tree will have an even larger height. The height h corresponds to the length of the longest path in the derivation tree of s. So, with $b + 1$ productions in the longest path and b symbols, it follows from the pigeonhole principle that one symbol X was used at least twice in a path of the derivation tree.



It is now possible to replace the whole subtree rooted at the second X with the (bigger) subtree rooted at the first X, yielding a larger derivation tree and thereby a longer string.



This process can be repeated indefinitely, yielding bigger and bigger derivation trees and longer and longer strings. Since all these derivation trees are valid productions, there are infinitely many different strings in the language $L(G)$.

CFG to PDA

Homework 7 included a grammar for the language $\{x \in \{a,b\}^* | x \neq ww \text{ for some } w \in \{a,b\}^*\}$. We can use the known method of converting that grammar to the following PDA:

$$
\begin{aligned}
M &= (Q, \Sigma, \Gamma, \delta, q_{start}, F) \\
Q &= \{q_{start}, q_{loop}, q_{accept}, q_1, q_2, q_3, q_4, q_5, q_6, q_7\} \\
\Sigma &= \{\mathtt{a}, \mathtt{b}\} \\
\Gamma &= \{\mathtt{a}, \mathtt{b}, S, X, Y, Z, \$\} \\
F &= \{q_{accept}\}
\end{aligned}
$$

start	$\delta(q_{start}, \epsilon, \epsilon) = \{(q_1, \$)\}$	$\delta(q_1, \epsilon, \epsilon) = \{(q_{loop}, S)\}$	
$S \to X$	$\delta(q_{loop}, \epsilon, S) = \{(q_{loop}, X)\}$		
$S \to Y$	$\delta(q_{loop}, \epsilon, S) = \{(q_{loop}, Y)\}$		
$S \to XY$	$\delta(q_{loop}, \epsilon, S) = \{(q_2, Y)\}$	$\delta(q_2, \epsilon, \epsilon) = \{(q_{loop}, X)\}$	
$S \to YX$	$\delta(q_{loop}, \epsilon, S) = \{(q_3, X)\}$	$\delta(q_3, \epsilon, \epsilon) = \{(q_{loop}, Y)\}$	
$X \to ZXZ$	$\delta(q_{loop}, \epsilon, X) = \{(q_4, Z)\}$	$\delta(q_4, \epsilon, \epsilon) = \{(q_5, X)\}$	$\delta(q_5, \epsilon, \epsilon) = \{(q_{loop}, Z)\}$
$X \to \mathtt{a}$	$\delta(q_{loop}, \epsilon, X) = \{(q_{loop}, \mathtt{a})\}$		
$Y \to ZYZ$	$\delta(q_{loop}, \epsilon, Y) = \{(q_6, Z)\}$	$\delta(q_6, \epsilon, \epsilon) = \{(q_7, Y)\}$	$\delta(q_7, \epsilon, \epsilon) = \{(q_{loop}, Z)\}$
$Y \to \mathtt{b}$	$\delta(q_{loop}, \epsilon, Y) = \{(q_{loop}, \mathtt{b})\}$		
$Z \to \mathtt{a}$	$\delta(q_{loop}, \epsilon, Z) = \{(q_{loop}, \mathtt{a})\}$		
$Z \to \mathtt{b}$	$\delta(q_{loop}, \epsilon, Z) = \{(q_{loop}, \mathtt{b})\}$		
consume a	$\delta(q_{loop}, \mathtt{a}, \mathtt{a}) = \{(q_{loop}, \epsilon)\}$		
consume b	$\delta(q_{loop}, \mathtt{b}, \mathtt{b}) = \{(q_{loop}, \epsilon)\}$		
accept	$\delta(q_{loop}, \epsilon, \$) = \{(q_{accept}, \epsilon)\}$