Quiz 1, Propositional Logic

Date: September 11

1. For the proposition: \((p \rightarrow r) \rightarrow ((q \rightarrow r) \rightarrow ((p \lor q) \rightarrow r))\),

(a) (5pts) Prove it is a tautology using truth tables.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>( A ):</th>
<th>( B ):</th>
<th>( C ):</th>
<th>( D ):</th>
<th>( E ):</th>
<th>( A \rightarrow (B \rightarrow D) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>( q )</td>
<td>( r )</td>
<td>( p \rightarrow r )</td>
<td>( q \rightarrow r )</td>
<td>( p \lor q )</td>
<td>( (p \lor q) \rightarrow r )</td>
<td>( B \rightarrow D )</td>
<td></td>
</tr>
<tr>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
<tr>
<td>( T )</td>
<td>( F )</td>
<td>( F )</td>
<td>( F )</td>
<td>( T )</td>
<td>( F )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
<tr>
<td>( F )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
<tr>
<td>( F )</td>
<td>( F )</td>
<td>( F )</td>
<td>( F )</td>
<td>( T )</td>
<td>( F )</td>
<td>( F )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
<tr>
<td>( F )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( F )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
<tr>
<td>( F )</td>
<td>( F )</td>
<td>( T )</td>
<td>( T )</td>
<td>( F )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
</tbody>
</table>
(b) (5pts) Prove it is a tautology using algebra (you do not need to annotate your algebra).

\[(p \to r) \to ((q \to r) \to ((p \lor q) \to r))\]

\[\equiv (\neg p \lor r) \to ((q \to r) \to ((p \lor q) \to r))\]  
\[\text{conditional law}\]

\[\equiv (\neg p \lor r) \to (\neg q \lor r) \to ((p \lor q) \to r)\]  
\[\text{conditional law}\]

\[\equiv (\neg p \lor r) \to (\neg q \lor r) \lor (\neg (p \lor q) \lor r)\]  
\[\text{conditional law}\]

\[\equiv \neg (\neg p \lor r) \lor ((\neg q \lor r) \lor (\neg (p \lor q) \lor r))\]  
\[\text{conditional law}\]

\[\equiv (\neg p \land \neg r) \lor (\neg q \lor r) \lor (\neg (p \lor q) \lor r)\]  
\[\text{DeMorgan's law}\]

\[\equiv (p \land \neg r) \lor (\neg (q \lor r) \lor (\neg (p \lor q) \lor r))\]  
\[\text{DeMorgan's law}\]

\[\equiv (p \land \neg q) \lor ((\neg q \land \neg r) \lor (\neg (p \lor q) \lor r))\]  
\[\text{law of negation}\]

\[\equiv ((p \land \neg r) \lor (q \land \neg r)) \lor (\neg (p \lor q) \lor r)\]  
\[\text{associativity}\]

\[\equiv ((p \lor q) \land \neg r) \lor (\neg (p \lor q) \lor r)\]  
\[\text{distributivity}\]

\[\equiv \neg (\neg (p \lor q) \lor \neg r) \lor (\neg (p \lor q) \lor r)\]  
\[\text{DeMorgan's law}\]

\[\equiv \neg (\neg (p \lor q) \lor r) \lor (\neg (p \lor q) \lor r)\]  
\[\text{law of negation}\]

\[\equiv TRUE\]  
\[\text{excluded middle}\]
Quiz 2, Rules of Inference

Date: September 18

1. (10pts) Prove \((p \rightarrow r) \rightarrow ((q \rightarrow r) \rightarrow ((p \lor q) \rightarrow r))\) using inference rules.

1 \([p \rightarrow r]\) Assumption
2 \([q \rightarrow r]\) Assumption
3 \([p \lor q]\) Assumption
4 \(r\) Case analysis 3,1,2
5 \((p \lor q) \rightarrow r\) \(\rightarrow\) introduction 3,4
6 \((q \rightarrow r) \rightarrow ((p \lor q) \rightarrow r)\) \(\rightarrow\) introduction 2,5
7 \((p \rightarrow r) \rightarrow ((q \rightarrow r) \rightarrow ((p \lor q) \rightarrow r))\) \(\rightarrow\) introduction 1,6
Quiz 3, Predicate Logic
Date: September 25

1. (5pts) Assert for the array $A[1, \ldots, n]$ that the elements alternately increase and decrease.

“For a given index $i$ of $A$, $A$ increases on one side of $i$ and decreases on the other side.”:

$$((A_{i-1} < A_i) \land (A_i > A_{i+1})) \lor ((A_{i-1} > A_i) \land (A_i < A_{i+1}))$$

“For every interior element in $A$, $A$ increases on one side of the element and decreases on the other.”:

$$\forall i \in I_{2}^{n-1} : ((A_{i-1} < A_i) \land (A_i > A_{i+1})) \lor ((A_{i-1} > A_i) \land (A_i < A_{i+1}))$$

While it is not necessary to do so, note that the above can be cleverly rewritten:

$$\forall i \in I_{2}^{n-1} : (A_i - A_{i-1})(A_{i+1} - A_i) < 0$$

2. (5pts) Assert for a graph $G(V,\text{Edge}(u,v))$ that there is NOT any pair of vertices $x, y$ such that they are connected by a path of 3 edges.

“$x$ and $y$ are connected by a path of 3 edges.”:

$$\exists z \in V : \exists w \in V : (x \neq w) \land (y \neq z) \land \text{Edge}(x, z) \land \text{Edge}(z, w) \land \text{Edge}(w, y)$$

“There is no pair $x, y$ such that $x$ and $y$ are connected by a path of 3 edges.”:

$$- (\exists x \in V : \exists y \in V : \exists z \in V : \exists w \in V : (x \neq w) \land (y \neq z) \land \text{Edge}(x, z) \land \text{Edge}(z, w) \land \text{Edge}(w, y) )$$
Quiz 4, Mathematical Induction

Date: October 2

1. (5pts) Prove $2^{2n} - 1$ is divisible by 3, $n \geq 1$
   (i.e. $\exists m \in \mathbb{N} : 2^{2n} - 1 = 3m$)
   (Hint: $4 = 3 + 1$).

When $n = 1$, $2^{2n} - 1 = 2^2 - 1 = 3$, which is divisible by 3, thus proving the base case.

Assume that for some $k \geq 1$, $2^{2k} - 1$ is divisible by 3, so for some integer $m_k$,

$$2^{2k} - 1 = 3m_k$$

We would like to prove the $k + 1$ case,

$$2^{2(k+1)} - 1 = 3m_{k+1} \quad \text{(for some integer $m_{k+1}$)}$$

To do this, we begin with the left hand side, and work until we can substitute the inductive hypothesis,

$$2^{2(k+1)} - 1 = 2^{2k}2^2 - 1$$
$$= 4 \times 2^{2k} - 1$$
$$= 3 \times 2^{2k} + 2^{2k} - 1 \quad \text{(from the hint)}$$
$$= 3 \times 2^{2k} + 3m_k \quad \text{(by the inductive hypothesis)}$$
$$= 3(2^{2k} + m_k)$$
$$= 3m_{k+1} \quad \text{(if we let $m_{k+1} = 2^{2k} + m_k$)}$$

This proves the inductive conclusion, thus by mathematical induction, the theorem is proved.
2. (5pts) Let \( S_{n+1} = 2S_n + 1, \) \( n \geq 0, S_0 = 0, \)
Prove \( S_n = 2^n - 1, \) \( n \geq 0. \)

When \( n = 0, \) we have
\[
S_n = S_0 = 0 = 2^0 - 1 = 2^0 - 1 = 1 - 1 = 0
\]

Assume that for some \( k \geq 0, \)
\[
S_k = 2^k - 1
\]

We would like to prove the \( k + 1 \) case,
\[
S_{k+1} = 2^{k+1} - 1
\]

To do this, we begin with the left hand side,
\[
\begin{align*}
S_{k+1} &= 2S_k + 1 \quad \text{(from the recursive definition)} \\
&= 2(2^k - 1) + 1 \quad \text{(from the inductive hypothesis)} \\
&= 2 \times 2^k - 2 + 1 \\
&= 2^{k+1} - 1
\end{align*}
\]

This proves the inductive conclusion, thus by mathematical induction, the theorem is proved.
Quiz 5, Program Verification

Date: October 10

1. (7pts) State, prove, and use the loop invariant for the following code, assuming \( n \geq 0 \).

\[
i \leftarrow 0
\]
\[
x \leftarrow 2
\]
\[\textbf{while } i < n \textbf{ do}
\]
\[
x \leftarrow 4 \times x
\]
\[
i \leftarrow i + 1
\]
\[
x \leftarrow x/2
\]
\[
i \leftarrow 0
\]
\[
x \leftarrow 2
\]
\[\text{// } (x = 2^{i+1}) \land (i \leq n)
\]
\[\textbf{while } i < n \textbf{ do}
\]
\[\text{// } (x = 2^{i+1}) \land (i \leq n) \land (i < n)
\]
\[
x \leftarrow 4 \times x
\]
\[\text{// } (x = 4 \times 2^{i+1} = 2^{i+3}) \land (i \leq n) \land (i < n)
\]
\[
i \leftarrow i + 1
\]
\[\text{// } (x = 2^{i+2}) \land (i \leq n)
\]
\[
x \leftarrow x/2
\]
\[\text{// } (x = 2^{i+2}/2 = 2^{i+1}) \land (i \leq n)
\]
\[\text{// } (x = 2^{i+1}) \land (i \leq n) \land \neg(i < n)
\]

Note that \((i \leq n) \land \neg(i < n)\) implies \(i = n\), so \(x = 2^{n+1}\) at the end.

2. (3pts) State the loop invariant for the following code.

\[
i \leftarrow 0
\]
\[
s \leftarrow 0
\]
\[\textbf{while } i \leq n \textbf{ do}
\]
\[
s \leftarrow s + s
\]
\[
i \leftarrow i + 1
\]
\[
s \leftarrow s + 2^i
\]

Answer: \(s = i \times 2^i\)
1. (5pts) Prove \( \sum_{i=0}^{n} a^i = \frac{a^{n+1} - 1}{a-1} \), \( a \neq 1 \), \( n \geq 0 \).

When \( n = 0 \), \( \sum_{i=0}^{n} a^i = \sum_{i=0}^{0} a^i = a^0 = 1 \), and
\[
\frac{a^{n+1} - 1}{a-1} = \frac{a^{0+1} - 1}{a-1} = \frac{a-1}{a-1} = 1,
\]
which proves the base case.

Assume that for some \( k \geq 0 \), \( \sum_{i=0}^{k} a^i = \frac{a^{k+1} - 1}{a-1} \) with \( a \neq 1 \).

We would like to prove the \( k + 1 \) case, \( \sum_{i=0}^{k+1} a^i = \frac{a^{(k+1)+1} - 1}{a-1} \).

To do this, we begin with the left hand side, and work until we can substitute the inductive hypothesis,
\[
\begin{align*}
\sum_{i=0}^{k+1} a^i &= a^{k+1} + \sum_{i=0}^{k} a^i \\
&= a^{k+1} + \frac{a^{k+1} - 1}{a-1} \quad \text{(from the inductive hypothesis)} \\
&= \frac{a^{k+1}(a-1) + a^{k+1} - 1}{a-1} \\
&= \frac{a^{k+1+1} - 1}{a-1}
\end{align*}
\]

This proves the inductive conclusion, thus by mathematical induction, the theorem is proved.
2. (5pts) Let $S_{n+1} = S_n + \left(\frac{1}{2}\right)^n$, $n \geq 0$, $S_0 = 0$.
Prove $S_n = 2 - \left(\frac{1}{2}\right)^{n-1}$, $n \geq 0$.

When $n = 0$, $2 - \left(\frac{1}{2}\right)^{n-1} = 2 - \left(\frac{1}{2}\right)^{0-1} = 2 - 2 = 0 = S_0 = S_n$, which proves the base case.

Assume that for some $k \geq 0$, $S_k = 2 - \left(\frac{1}{2}\right)^{k-1}$.

We would like to prove the $k+1$ case, $S_{k+1} = 2 - \left(\frac{1}{2}\right)^{(k+1)-1} = 2 - \left(\frac{1}{2}\right)^k$.

To do this, we begin with the left hand side, and work until we can substitute the inductive hypothesis,

$$S_{k+1} = S_k + \left(\frac{1}{2}\right)^k$$

(from the recursive definition)

$$= 2 - \left(\frac{1}{2}\right)^{k-1} + \left(\frac{1}{2}\right)^k$$

(from the inductive hypothesis)

$$= 2 - 2 \times \left(\frac{1}{2}\right)^k + \left(\frac{1}{2}\right)^k$$

$$= 2 - \left(\frac{1}{2}\right)^k$$

This proves the inductive conclusion, thus by mathematical induction, the theorem is proved.
Quiz 7, Regular Expressions

Date: November 1

1. (2 pts) Write the strings of length 5 in $L(R)$, $R = (ab + b)a^*b$.

   $\{abaab, baaab\}$. At the beginning of the string, there is a choice of $ab$ or $b$, and once that choice is made, the remainder of the string is fixed, since we know that the length must be exactly five.

2. (2 pts) Simplify $((a + \Lambda)^*a + a^*)$.

   Answer: $a^*$

   $(a + \Lambda)^* = a^*$, so $(a + \Lambda)^*a = a^+$. $a^+ + a^* = a^*$ because $a^+ \subseteq a^*$.

3. (6 pts) Give a regular expression $R$, $L(R) = L$, $\Sigma = \{a, b\}$, $L = \{x \mid x \text{ contains } aa \text{ but does not contain } bb\}$.

   The set of strings which do not contain a $bb$ can written with the expression $(b + \Lambda)(a + ab)^*$, because any $b$ which is not at the beginning of the string must be preceded by an $a$.

   It follows that every string in $L$ can be expressed as follows: any string without a $bb$, followed by some $aa$, followed by any string without any string without a $bb$. In other words,

   $$((b + \Lambda)(a + ab)^*)(aa)((b + \Lambda)(a + ab)^*)$$
Quiz 8, Regular Grammars

Date: November 6

1. (6pts) Convert $a + b^*$ into a regular grammar with unit productions.

\[
P_1 = \{ S_1 \to aA_1, A_1 \to \Lambda \} \\
P_2 = \{ S_2 \to bA_2, A_2 \to \Lambda \} \\
P_3 = \{ S_3 \to \Lambda, S_3 \to S_2, S_2 \to bA_2, A_2 \to S_3 \} \\
P_4 = \{ S_4 \to S_1, S_4 \to S_3, S_1 \to aA_1, A_1 \to \Lambda, \\
S_3 \to \Lambda, S_3 \to S_2, S_2 \to bA_2, A_2 \to S_3 \} 
\]

Using $P_4$ as the final answer, the start symbol is $S_4$. 
2. (4pts) Convert into a regular grammar:
\{S \rightarrow aA, A \rightarrow B, A \rightarrow bC, C \rightarrow A, C \rightarrow bS, C \rightarrow \Lambda, B \rightarrow aB\}.

Solution:

\[
\begin{align*}
S & \rightarrow aA \\
A & \rightarrow B \\
A & \rightarrow bC \\
B & \rightarrow aB \\
C & \rightarrow \Lambda \\
C & \rightarrow bS \\
C & \rightarrow \Lambda \\
C & \rightarrow B \\
A & \rightarrow aB \\
C & \rightarrow bC \\
C & \rightarrow aB
\end{align*}
\]
Quiz 9, Regular Grammar Conversion
Date: November 13

1. (6pts) Convert \{S \to aA, S \to \Lambda, A \to bB, A \to aA, B \to bS\} into a regular expression.

First add \(S'\), \(H\), and missing loopbacks.

\[
\begin{align*}
S' & \to S \\
S & \to aA \\
A & \to bB \\
B & \to bS \\
H & \to \Lambda \\
S & \to H \\
A & \to aA \\
B & \to B \\
S & \to S \\
A & \to bB/ B \to B/ B \to bS: A \to bbS
\end{align*}
\]

To remove \(B\) then \(A\) then \(S\), begin by removing \(B\).

\[
\begin{align*}
A & \to bB/ B \to B/ B \to bS: A \to bbS
\end{align*}
\]

After removing \(B\) the remaining productions are:

\[
\begin{align*}
S' & \to S \\
S & \to aA \\
A & \to bbS \\
H & \to \Lambda \\
S & \to H \\
A & \to aA \\
S & \to S \\
\end{align*}
\]

Remove \(A\).

\[
\begin{align*}
S & \to aA/ A \to aA/ A \to bbS: S \to aa*bbS
\end{align*}
\]

After removing \(A\), the remaining productions are:

\[
\begin{align*}
S' & \to S \\
S & \to H \\
H & \to \Lambda \\
S & \to (\Lambda + aa*bb)S
\end{align*}
\]

Remove \(S\).

\[
\begin{align*}
S' & \to S/ S \to (\Lambda + aa*bb)S/ S \to H: (aa*bb)^*
\end{align*}
\]

Regular expression: \((aa*bb)^*\)
Alternately, to remove $A$ then $B$ then $S$, begin by removing $A$.

\[
S \rightarrow aA / A \rightarrow aA / A \rightarrow bB : S \rightarrow aa^*bB
\]

After removing $A$ the remaining productions are:

\[
S' \rightarrow S \\
S \rightarrow aa^*bB \\
B \rightarrow bS \\
H \rightarrow \Lambda \\
S \rightarrow H \\
B \rightarrow B \\
S \rightarrow S
\]

Remove $B$.

\[
S \rightarrow aa^*bB / B \rightarrow B / B \rightarrow bS : S \rightarrow aa^*bbS
\]

After removing $B$, the remaining productions are:

\[
S' \rightarrow S \\
S \rightarrow H \\
H \rightarrow \Lambda \\
S \rightarrow (\Lambda + aa^*bb)S
\]

Remove $S$.

\[
S' \rightarrow S / \Lambda + aa^*bb)S / S \rightarrow H : (aa^*bb)^*
\]

Regular expression: $(aa^*bb)^*$
Alternately, to remove $S$ then $A$ then $B$, begin by removing $S$.

$S' \rightarrow S / S \rightarrow S / S \rightarrow aA$  :  $S' \rightarrow aA$

$S' \rightarrow S / S \rightarrow S / S \rightarrow H$  :  $S' \rightarrow H$

$B \rightarrow bS / S \rightarrow S / S \rightarrow aA$  :  $B \rightarrow baA$

$B \rightarrow bS / S \rightarrow S / S \rightarrow H$  :  $B \rightarrow bH$

After removing $S$, the remaining productions are:

$S' \rightarrow aA$  $A \rightarrow bB$  $B \rightarrow baA$  $H \rightarrow \Lambda$

$S' \rightarrow H$  $A \rightarrow aA$  $B \rightarrow bH$

$B \rightarrow B$

Remove $A$.

$S' \rightarrow aA / A \rightarrow aA / A \rightarrow bB$  :  $S' \rightarrow aa^*bB$

$B \rightarrow baA / A \rightarrow aA / A \rightarrow bB$  :  $B \rightarrow baa^*bB$

After removing $A$, the remaining productions are:

$S' \rightarrow aa^*bB$  $B \rightarrow (\Lambda + baa^*b)B$  $H \rightarrow \Lambda$

$S' \rightarrow H$  $B \rightarrow bH$

Remove $B$.

$S' \rightarrow aa^*bB / B \rightarrow (\Lambda + baa^*b)B / B \rightarrow bH$  :  $S' \rightarrow aa^*b(baa^*b)^*bH$

Regular expression: $\Lambda + aa^*b(baa^*b)^*b$
2. (4 pts) Convert \( \{ S \to aA, S \to \Lambda, A \to bB, A \to aA, B \to bS \} \) into a deterministic regular grammar.

The clever person will note that the grammar is already a deterministic regular grammar. However, proceeding by algorithm will produce the following.

\[
V_{\{S\}} \to aV_{\{A\}} \quad V_{\{A\}} \to aV_{\{A\}} \quad V_{\{B\}} \to aV_{\emptyset} \\
V_{\{S\}} \to bV_{\emptyset} \quad V_{\{A\}} \to bV_{\{B\}} \quad V_{\{B\}} \to bV_{\{S\}}
\]

\[V_{\{S\}} \to \Lambda\]
Quiz 10, Finite Automata
Date: November 20

1. (5pts) Build a DFA for $\Sigma = \{a, b, c\}$,
$L = \{x \mid x$ contains exactly 2 $a$s and at least one $b\}$. 

$q_0$ - no $a$s or $b$s seen
$q_1$ - one $a$ and no $b$s
$q_2$ - two $a$s and no $b$s
$q_3$ - three or more $a$s (trap state; does not need to be shown)
$q_4$ - no $a$s and at least one $b$
$q_5$ - one $a$ and at least one $b$
$q_6$ - two $a$s and at least one $b$

2. (5pts) Build a DFA for $\Sigma = \{a, b\}$,
$L = \{x \mid x$ contains a $ba$ before any $bba\}$. 

$q_0$ - no $ba$ seen, and the last character was not a $b$
$q_1$ - no $ba$ seen, and the last character was a (single) $b$
$q_2$ - no $ba$ seen, and the last characters seen were $bb$
$q_3$ - $ba$ seen, prior to a $bba$
$q_4$ - $bba$ seen first (trap state; does not need to be shown)
Quiz 11, Finite Automata II
Date: November 29

1. (4pts) Give a DFA for \( L(P), P = ababbaba. \)

2. (3pts) Directly give a DFA for \( L(r), r = (a+ab)bb. \)

   It helps to note that there are only two strings in the language, \( abb \) and \( abbb \).

   Note: the intended problem was \((a+ab)^*bb\), which would result in two additional transitions:

3. (3pts) Prove \( L \) is not regular, \( L = \{ x \mid x = a^i b^j, i > j \} \).

   Let \( S \) be the set \( S = \{ a^i \mid i \geq 0 \} \). Pick any distinct pair \( x = a^i \) and \( y = a^j \) from \( S \), and without loss of generality assume that \( i > j \) (since \( x \) and \( y \) are different, then one of them has to be the longer one, so we’ll call \( x \) the longer one).

   Let \( z = b^j \), so \( z \) is the same length as \( y \) and shorter than \( x \). It follows that \( xz = a^ib^j \in L \) but \( yz = a^ib^j \notin L \). This means that no matter which distinct pair of elements are picked from \( S \), there is some string which can be appended to each of them which will allow them to be
distinguished by a state machine.

Since every string in $S$ is distinguishable by a state machine, any state machine which recognizes the language must have an infinite number of states to accommodate for each of the strings in $S$. No finite automata can have an infinite number of states, therefore the language $L$ is not regular.
Quiz 12, Context-Free Grammars
Date: December 6

1. (5pts) Give a CFG for \( \{ a^i b^j c^k \mid i \leq k \leq i + j \} \).

We see that \( k = i + n \) for some \( n \). So \( j \geq n \) and \( j = n + m \) for some \( m \). Then for as long as \( n \geq 0 \) and \( m \geq 0 \), all of the inequalities will be satisfied (because they can also be written \( i \leq i + n \leq i + n + m \)). The resulting string can be rewritten \( a^i b^{n+m} c^{i+n} = a^i (b^n)(b^m c^n) c^i \).

Let \( B \) generate strings of the form \( b^n \) and \( C \) generate strings of the form \( b^m c^n \). Then the following CFG will generate the language:

\[
S \rightarrow aSc | BC \\
B \rightarrow bB | \Lambda \\
C \rightarrow bCc | \Lambda
\]

2. (5pts) Give a CFG for \( \{ a^i b^j c^k \mid j = 2i + 3k \} \).

Since \( j = 2i + 3k \), the resulting string can be rewritten as \( a^i b^{2i+3k} c^k = (a^i b^{2i})(b^{3k} c^k) \).

Let \( A \) generate strings of the form \( a^i b^{2i} \) and \( C \) generate strings of the form \( b^{3k} c^k \). Then the following CFG will generate the language:

\[
S \rightarrow AC \\
A \rightarrow aAbb | \Lambda \\
C \rightarrow bbbCc | \Lambda
\]