SFWR ENG 2FA3. Solution to the Assignment #5

Total = 197, 100%\(\approx\)160

The solutions below are often very detailed on purpose. Such level of details is not required from students’ solutions. Some questions have more than one solution. If you think your solution has been marked wrongly, write a short memo stating where marking is wrong and what you think is right, and resubmit to me during class, office hours, or just slip under the door to my office.

1.[6] Exercise 7.5 (b), (i), (o) (pages 135-137 of the Gries-Schneider textbook).

(b).[3]

\[0 = \langle \text{Axiom Left zero} \rangle \]
\[1 = \langle \text{Axiom Left one} \rangle \]
\[10 = \langle \text{Axiom Left zero} \rangle \]
\[11 = \langle \text{Axiom Left zero} \rangle \]

(i).[3] 1011 ≻ (2 \cdot 1 + 0) 11 ≻ (2 \cdot 2 + 1) 11 ≻ (2 \cdot 5 + 1) ≻ (11) ≻ (12) ≻ (12 + 1) = 13

(o).[1] Show that the expression \(x1 = \infty\) is unsatisfiable.

There are no axioms of the form: \(x = \ldots, 1 = \ldots\) and \(x1 = \ldots\), so there is nowhere to start.

2.[2] Exercise 8.1 (c) (page 155 of the Gries-Schneider textbook).

\(e(a(c(x)), a(u))\): Since \(x : C\) then \(c(x)\) is type A, so \(a(c(x))\) is type B; since \(u : A\) then \(a(u)\) is also type B. Since \(e : B \times B \rightarrow E\), then \(e(a(c(x)), a(u))\) is type E. So the expression is correct.

3.[2] Exercise 8.6 (b) (page 155 of the Gries-Schneider textbook).

Solution:
\(\forall(i \mid 0 \leq i < n + 1 : b[i] = 0)\)
\(= (0 \leq i < n + 1 \iff 0 \leq i < n \lor i = n)\)
\(\forall(i \mid 0 \leq i < n \lor i = n : b[i] = 0)\)
\(= (\text{Axiom Range Split and } (i = n) \land (0 \leq i, n) \equiv false)\)
\(\forall(i \mid 0 \leq i < n : b[i] = 0) \land \forall(i \mid i = n : b[i] = 0)\)
\(= (\text{Axiom One-point Rule})\)
\(\forall(i \mid 0 \leq i < n : b[i] = 0) \land b[n] = 0\)
4.5 Exercise 9.5 (page 174 of the Gries-Schneider textbook).
Prove \( \forall (x \mid R : P \equiv Q) \implies (\forall (x \mid R : P) \equiv \forall (x \mid R : Q)) \).

Assume that \( \forall (x \mid R : P \equiv Q) \).

\[
\forall (x \mid R : P) = \forall (x \mid R : Q)
\]

\( \forall (x \mid R : P) \) means that for each \( x \in R \), \( P(x) = True \) of and only if \( Q(x) = True \), so we can replace \( P \) with \( Q \) as long as \( x \in R \).

Hence \( \forall (x \mid R : P \equiv Q) \implies (\forall (x \mid R : P) \equiv \forall (x \mid R : Q)) \).

5.5 Exercise 9.24 (page 175 of the Gries-Schneider textbook).
Prove \( \exists (x \mid R : P) \implies \exists (x \mid Q \lor R : P) \).

First note that the disjunction \( \lor \) is idempotent as \( p \lor p = p \).

\[
\exists (x \mid R : P) \implies \exists (x \mid Q \lor R : P)
\]

6.3 Exercise 9.29 (e) (page 175 of the Gries-Schneider textbook).
Every positive integer is smaller than the absolute value of some negative integer.

\[
\forall (x \mid x \in \mathbb{Z} : x > 0) \implies \exists (y \mid y \in \mathbb{Z} : y < 0 \land x < abs(y))
\]

7.3 Exercise 9.31(b) (page 176 of the Gries-Schneider text).
Define suitable predicates and function and then formalize the following sentence: 
Broadcasts made by a process are received by all processes in the order sent.

Let \( Processes \) be the set of appropriate processes and for every process \( P \in Processes \), let \( Br(P) \) be the set of all broadcasts made by \( P \). For every two broadcasts \( x, y \in Br(P) \), we will write \( s(x) <_P s(y) \) if \( x \) was sent earlier than \( r \), and \( r(x) <_Q r(y) \) if \( x \) was received earlier than \( y \) by a process \( Q \). Now we can write:

\[
\forall (P, x, y \mid P \in Processes \land x, y \in Pr(P) : s(x) <_P s(y) \implies \forall (Q \mid Q \in Processes \setminus \{P\} : r(x) <_Q r(y)))
\]
8. Exercise 10.1 (g) (page 191 of the Gries-Schneider textbook).

It is not the case that all zeros of $b[0..n-1]$ are in $b[j..k]$.

\[ \neg (\forall i : 0 \leq i < n : b[i] = 0 \implies j \leq i \leq k) \]

9. Exercise 10.6 (h) (page 192 of the Gries-Schneider text).

Integer array $s[0..n]$ contains the grade of each student on a homework, where a negative number means that no grade was handed in. All the grades handed in turns out to be different. Find the average grade (from these handed in).

Let $one$ be the constant function for integers given by $one(i) = 1$ for all integers $i$.

Define $\text{sum} = \sum(i : 0 \leq i < n \land s[i] \geq 0 : s[i])$, and $\text{count} = \sum(i : 0 \leq i < n \land s[i] \geq 0 : one(i))$. Then

\[ \{ n \geq 0 \} \]

\[ \text{average} := ? \]

\[ \{(\text{count} = 0 \implies \text{average} = 0) \land (\text{count} > 0 \implies \text{average} = \text{sum}/\text{count}\} \]

10. Exercise 10.7 (d) (page 192 of the Gries-Schneider textbook).

Reverse $b$, e.g. change $(3,2,5,5,1)$ into $(1,5,5,2,3)$.

Assume the array is $b[0..n-1]$.

\[ \{ n > 0 \} \]

\[ \text{reverseb} := ? \]

\[ \{(i : 0 \leq j < n : \text{reverseb}[i] = b[n-i-1]\} \]

11. Exercise 10.9 (page 193 of the Gries-Schneider textbook).

Calculate and simplify the weakest precondition for the following (where $x$ and $y$ are boolean variables)

\[ \{ ? \} \quad x := x \neq y; \quad y := x \neq y; \quad x := x \neq y; \quad \{(x \equiv X) \land (y \equiv Y)\} \]

The simplest solution is by dividing the problem into four specific cases: $X = 0 \land Y = 0$, $X = 0 \land Y = 1$, $X = 1 \land Y = 0$ and $X = 1 \land Y = 1$.

Case $X = 0 \land Y = 0$.

\[ \{(x = 0) \land (y = 0)\}\{x := x \neq y; y := x \neq y; x := x \neq y; \{(x \equiv X) \land (y \equiv Y)\} \]

\[ = \text{ textual substitution } \]

\[ = \{(x \neq y) = 0 \land (y = 0)\}\{y := x \neq y; x := x \neq y; \{(x \equiv X) \land (y \equiv Y)\} \]

\[ = \text{ since } (x \neq y) = 0 \text{ implies } x = y \]

\[ = \{(x = 0) \land (y = 0)\}\{y := x \neq y; x := x \neq y; \{(x \equiv X) \land (y \equiv Y)\} \]

\[ = \text{ textual substitution } \]

\[ = \{(x = 0) \land ((x \neq y) = 0)\}\{x := x \neq y; \{(x \equiv X) \land (y \equiv Y)\} \]

\[ = \text{ since } (x \neq y) = 0 \text{ implies } x = y \]
\( \{(x = 0) \wedge (y = 0)\}[x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)

\( \{(x \neq y = 0) \wedge (y = 0)\} \)

= \( \langle \text{since } (x \neq y) = 0 \text{ implies } x = y \rangle \)

\( \{(x = 0) \wedge (y = 0)\} \)

Case \( X = 0 \wedge Y = 1 \).

\( \{(x = 0) \wedge (y = 1)\}[x := x \neq y][y := x \neq y][x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)

\( \{(x \neq y = 0) \wedge (y = 1)\}[y := x \neq y][x := x \neq y] \)

= \( \langle \text{since } (x \neq y) = 0 \text{ implies } x = y \rangle \)

\( \{(x = 1) \wedge (y = 1)\}[y := x \neq y][x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)

\( \{(x = 1) \wedge ((x \neq y) = 1)\}[x := x \neq y] \)

= \( \langle \text{since } (x \neq y) = 1 \text{ implies } y = \neg x \rangle \)

\( \{(x = 1) \wedge (y = 0)\}[x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)

\( \{(x \neq y = 1) \wedge (y = 0)\} \)

= \( \langle \text{since } (x \neq y) = 1 \text{ implies } x = \neg y \rangle \)

\( \{(x = 1) \wedge (y = 0)\} \)

Case \( X = 1 \wedge Y = 0 \).

\( \{(x = 1) \wedge (y = 0)\}[x := x \neq y][y := x \neq y][x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)

\( \{(x \neq y = 1) \wedge (y = 0)\}[y := x \neq y][x := x \neq y] \)

= \( \langle \text{since } (x \neq y) = 1 \text{ implies } x = \neg y \rangle \)

\( \{(x = 1) \wedge (y = 0)\}[y := x \neq y][x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)

\( \{(x = 1) \wedge ((x \neq y) = 0)\}[x := x \neq y] \)

= \( \langle \text{since } (x \neq y) = 0 \text{ implies } y = x \rangle \)

\( \{(x = 1) \wedge (y = 1)\}[x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)

\( \{(x \neq y = 1) \wedge (y = 1)\} \)

= \( \langle \text{since } (x \neq y) = 1 \text{ implies } x = \neg y \rangle \)

\( \{(x = 0) \wedge (y = 1)\} \)

Case \( X = 1 \wedge Y = 1 \).

\( \{(x = 1) \wedge (y = 1)\}[x := x \neq y][y := x \neq y][x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)

\( \{(x \neq y = 1) \wedge (y = 1)\}[y := x \neq y][x := x \neq y] \)

= \( \langle \text{since } (x \neq y) = 1 \text{ implies } x = \neg y \rangle \)

\( \{(x = 0) \wedge (y = 1)\}[y := x \neq y][x := x \neq y] \)

= \( \langle \text{textual substitution} \rangle \)
\{ (x = 0) \land ((x \not\equiv y) = 1) \}\{ x := x \not\equiv y \} \\
= \langle \text{since } (x \not\equiv y) = 1 \text{ implies } y = \neg x \rangle \\
\{ (x = 0) \land (y = 1) \}\{ x := x \not\equiv y \} \\
= \langle \text{textual substitution} \rangle \\
\{ ((x \not\equiv y) = 0) \land (y = 1) \} \\
= \langle \text{since } (x \not\equiv y) = 0 \text{ implies } x = y \rangle \\
\{ ((x = 1) \land (y = 1)) \}

Hence in all cases \{ ? \} = \{ (x \equiv Y) \land (y \equiv X) \}.

12. [3] Show the postcondition \( R \) for the following program:

\{ y = 3 \}
\begin{align*}
x &:= 2; \\
z &:= x + y; \\
\text{if } y > 0 & \text{ then } x := z + y \\
\text{else } & y := x + z
\end{align*}
\{ R \}

\( R = (x = 8 \land y = 3 \land z = 5) \).

13. [3] Show the postcondition \( R \) for the following program:

\{ z = 0 \land y = 5 \}
\begin{align*}
\text{for } i = 1 \text{ to } 5 \text{ do} \\
y &:= y \ast z; \\
z &:= z \ast b[i]
\end{align*}
\{ R \}

After unfolding the loop we have:

\{ z = 0 \land y = 5 \}
\begin{align*}
y &:= y \ast z; \\
z &:= z \ast b[1]; \\
y &:= y \ast z; \\
z &:= z \ast b[2]; \\
y &:= y \ast z; \\
z &:= z \ast b[3]; \\
y &:= y \ast z; \\
z &:= z \ast b[4]; \\
y &:= y \ast z; \\
z &:= z \ast b[5];
\end{align*}
\{ R \}

Since initially \( z = 0 \), it is easy to show that \( R = (z = 0 \land y = 5) \).
Exercise 12.4 (page 243 of the Gries-Schneider textbook), questions (d) and (f).

(d) Prove by induction that for \( n \geq 0 \), \( \sum (i \mid 0 \leq i < n : 3^i) = (3^n - 1)/2 \).

For \( n = 0 \), the range of the sum is empty, so \( \sum (i \mid 0 \leq i < 0 : 3^i) = 0 = (3^0 - 1)/2 \).

Suppose it holds for \( n \) and consider \( n + 1 \):
\[
\sum (i \mid 0 \leq i < n + 1 : 3^i) = \sum (i \mid 0 \leq i < n : 3^i) + 3^n = \frac{1}{2} \cdot (3^n - 1 + 2 \cdot 3^n) = \frac{1}{2} \cdot (3 \cdot 3^n - 1) = (3^{n+1} - 1)/2.
\]

(f) Prove by induction that for \( n \geq 0 \), \( \sum (i \mid 1 \leq i \leq n : 2^i) = (n-1) \cdot 2^{n+1} + 2 \).

For \( n = 0 \), the range of the sum is empty, so \( \sum (i \mid 1 \leq i \leq 0 : 2^i) = 0 = -1 \cdot 2^1 + 2 \).

Suppose it holds for \( n \) and consider \( n + 1 \):
\[
\sum (i \mid 1 \leq i \leq n + 1 : 2^i) = \sum (i \mid 1 \leq i \leq n : 2^i) + (n + 1) \cdot 2^{n+1} = (n-1) \cdot 2^{n+1} + 2 + (n+1) \cdot 2^{n+1} = 2 \cdot n \cdot 2^{n+1} + 2 = n \cdot 2^{n+2} + 2 = ((n-1) + 1) \cdot 2^{(n+1)+1} + 2.
\]

Exercise 12.34 (page 246 of the Gries-Schneider textbook).

Prove by induction that \( n! > 2^{n-1} \) for \( n \geq 3 \).

For \( n = 3 \), \( n! = 1 \cdot 2 \cdot 3 = 6 \) while \( 2^3 = 8 \), so it does hold.

Suppose it holds for \( n \) and consider \( n + 1 \):
\[
(n+1)! = n! \cdot (n+1) > 2^{n-1} \cdot (n+1) > 2^{n-1} \cdot 2 = 2^n = 2^{(n+1)-1}
\]

Give state diagrams of Deterministic Finite Automata accepting the following sets. The alphabet is always \{0,1\}.

(a) \{ \( x \mid x \) has length at least 3 and its third symbol is 0\}
(b) [5] \( \{x \mid x \text{ contains at least three 1's}\} \)

(c) [5] \( \{x \mid x \text{ contains at least two 0's and at most one 1}\} \)

17. [16] Each of the following languages is the intersection of two simpler languages. Construct Deterministic Finite Automata for simpler languages and then combine them using the construction for the intersection of two regular sets. The alphabet is \( \{a, b\} \).

(a) [8] \( \{x \mid x \text{ has at least three } a \text{'s and at least two } b \text{'s}\} \). At least three \( a \)'s At least two \( b \)'s

\[ \text{At least three } a \text{'s} \]

\[ \text{At least two } b \text{'s} \]
Intersection

(b) \( [8] \) \( \{x \mid x \text{ has exactly two } a\text{'s and at least two } b\text{'s}\} \).

Exactly two \( a\)’s    At least two \( b\)’s
18. [10] Each of the following languages is the complement of a simpler language. Construct Deterministic Finite Automata for simpler languages, then use it to give a state diagram of the language given (construction of the complement of a language). The alphabet is \( \{a, b\} \).

(a) [5] \( \{x \mid x \text{ does not contain the substring } ab \} \).

(b) [5] \( \{x \mid x \text{ is any string not in } a^*b^* \} \).

19. [4] Give a Nondeterministic Finite Automaton with \( \Sigma = \{0, 1\} \) that accepts \( \{x \mid x \text{ ends with } 00 \} \) and the automaton has three states.

20. [10] Prove that every Nondeterministic Finite Automaton can be converted to an equivalent one that has a single final state.

The idea is illustrated by the following example:
Formal definition is the following.

Let $M = (Q, \Sigma, \Delta, s_0, F)$ be a non-deterministic automaton such that $L(M) = A$ and $|F| > 1$.

Define a non-deterministic automaton $M_1 = (Q_1, \Sigma, \Delta_1, s_0, \{s_F\})$,

where $s_F \notin Q$, $Q_1 = Q \cup \{s_F\}$, and $\Delta_1 : Q_1 \times \Sigma \rightarrow 2^{Q_1}$ is defined as follows:

$$\Delta_1(p, a) = \Delta(p, a) \cup \{s_F | \Delta(p, a) \in F\},$$

for all $p \in Q$ and $a \in \Sigma$, and $\Delta_1(s_F, a) = \emptyset$ for all $a \in \Sigma$.

21.[5] Convert the following nondeterministic automaton into equivalent deterministic one.
Solution:

22. [16] Give a regular expression that is equivalent to the following automaton:

(a) [8] the automaton from Question 21.

Hence the answer is: \((a + ab^* (a+b))(aab^* (a+b))^*\).
(b) [8] the automaton below.

Solution:

Hence the answer is: $a^*b(ba^*)^*$.

23. [10] Give regular expressions generating the languages from Question 17.

(a) [5] $L = \{x \mid x \text{ has at least three } a's \text{ and at least two } b's\}$.

Let start with $L' = \{x \mid x \text{ has exactly three } a's \text{ and exactly two } b's\}$.

The regular expression for $L'$ is the following:

$$aaabb + aabba + abbaa + bbaaa + aabab + ababa + babaa + abaab + baaab + baaba$$

We now just have to allow additional $a$'s and $b$'s:

$$aaaa'b^*b + aaaa'bb + aaab + baaa'bb + babaa'b + babab + baba + bab + bab + bba + baaa + bbb$$

(b) [5] $L = \{x \mid x \text{ has exactly two } a's \text{ and at least two } b's\}$.

Let start with $L' = \{x \mid x \text{ has exactly two } a's \text{ and exactly two } b's\}$.

The regular expression for $L'$ is the following:

$$aabb + abba + bbaa + abab + baba + baab$$

We now just have to allow additional $b$'s:

$$aabb + abbb + bbb^*a + bba + bba + abba + baab$$
24. [10] Give an Nondeterministic Finite Automaton accepting the following languages:

(a) [5] \((01 + 001 + 010)^*\)

(b) [5] \((0 + 1)^*000(0 + 1)^*\)

25. [8] Find the minimum state finite automaton equivalent to the one below.

After deleting all useless/unreachable states we get the following automaton:

The above automaton is minimum states as:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
26.[13] Give context-free grammars generating the following languages:

(a) \{a^n b^m \mid n \neq m \}

Note that \{a^n b^m \mid n \neq m \} = \{a^n b^m \mid n > m \} \cup \{a^n b^m \mid n < m \}

The following grammar generates \{a^n b^m \mid n > m \}:

\begin{align*}
S_1 & \rightarrow AB \\
A & \rightarrow aA \mid a \\
B & \rightarrow aBb \mid \varepsilon
\end{align*}

The following grammar generates \{a^n b^m \mid n < m \}:

\begin{align*}
S_2 & \rightarrow CD \\
C & \rightarrow aCb \mid \varepsilon \\
D & \rightarrow bD \mid b
\end{align*}

Hence \{a^n b^m \mid n \neq m \} is given by the following grammar:

\begin{align*}
S & \rightarrow S_1 \mid S_2 \\
S_1 & \rightarrow AB \\
A & \rightarrow aA \mid a \\
B & \rightarrow aBb \mid \varepsilon \\
S_2 & \rightarrow CD \\
C & \rightarrow aCb \mid \varepsilon \\
D & \rightarrow bD \mid b
\end{align*}

(b) [5] The language of all regular expressions over the alphabet \{a, b, c\}.

\begin{align*}
S & \rightarrow a \mid b \mid c \mid \varepsilon \mid (S + S) \mid (SS) \mid (S)^*
\end{align*}

27.[5] Convert the below grammar into Chomsky Normal Form.

\begin{align*}
S & \rightarrow ASA \mid A \mid \varepsilon \\
A & \rightarrow 00 \mid \varepsilon
\end{align*}

Let \(L\) be the languages generated by this grammar. Since \(\varepsilon \in L\), in the textbook version this grammar cannot be fully represented in Chomsky Normal Form, so let us consider a grammar that generates \(L \setminus \{\varepsilon\}\).

After removal of all \(\varepsilon\)-productions, we get:
After removal all unit productions we get:

\[ S \rightarrow ASA \mid A \]
\[ A \rightarrow 00 \]

A Chomsky Normal Form is:

\[ S \rightarrow AXSA \mid X0X0 \]
\[ XSA \rightarrow SA \]
\[ A \rightarrow X0 \]
\[ X0 \rightarrow 0 \]

28. [10] Construct a pushdown automaton accepting:

\[ \{a^n b^m \mid n > 2m\} \]

When \( a \) is read it is pushed on the stack, however the very first \( a \) is not pushed. When \( b \) is read it pops two stack elements. We can only once switched from reading \( a \) to \( b \) and the is no switch back (two separate states). We accept with final states when there there are some \( a \)'s on the stack, or just \( \perp \).

Formally:

\[ M = (Q, \Sigma, \Gamma, \delta, s, \perp, F), \text{ where} \]
\[ Q = \{s,q_1,q_2,q_3\}, \Sigma = \{a,b\}, \Gamma = \{a, \perp\}, F = \{q_2\}, \text{ and the relation } \delta \text{ is the following:} \]

\[ \{(s,a,\perp),(q_1,\perp)\}, \quad \leftarrow \text{the first } a \text{ is just read (to enforce } '>') \]
\[ (q_1,a,\perp),(q_1,a, \perp), \quad \leftarrow \text{the second } a \text{ is pushed on the stack} \]
\[ (q_1,a,a),(q_1,aa), \quad \leftarrow \text{remaining } a \text{'s are pushed on the stack} \]
\[ (q_1,b,a),(q_2,\epsilon), \quad \leftarrow \text{first } b \text{ pops } a \text{ from the stack for the first time} \]
\[ (q_2,\epsilon,a),(q_3,\epsilon), \quad \leftarrow \text{ } b \text{ pops } a \text{ from the stack for the second time} \]
\[ (q_3,b,a),(q_2,\epsilon), \quad \leftarrow b \text{ (but not first) pops } a \text{ from the stack for the first time} \]
\[ (q_1,a,a),(q_2,\epsilon), \quad \leftarrow \text{deals with the case when } m = 0 \]

Remember, this automaton is non deterministic!

29. [8] Show that the following language is not context-free:

\[ \{a^i b^j c^k \mid 0 < i < j < k\} \]

The proof is very similar to the proof for \( \{a^n b^n c^n \mid 0 < n\} \)

Solution:
Assume $L = \{a^ib^jc^k \mid 0 < i < j < k\}$ is context free, then pumping lemma holds.

Let $p$ be the number in pumping lemma. We take $z = a^pb^{p+1}c^{p+2}$. Clearly, $|z| \geq p$ and $z \in L$. Therefore, $z = uvwxy$ and $|vx| \geq 1$, $|vwxy| \leq p$, and for all $i \geq 0$, $uv^iwx^iy \in L$.

Since $|vwxy| \leq p$ we have two separate cases, $z = vwxy$ contains no $c$ and $z = vwxy$ contains no $a$.

If $z = vwxy$ contains no $c$, then all $c$’s are in $y$, so for each $i$, $\#_c(uv^iwx^iy) = p + 2$.

Since $|vx| \geq 1$, $vy$ must contain at least one symbol and this must be either $a$ or $b$. Suppose that it is $a$. Then consider $t = uv^{p+3}wx^{p+3}y$. Clearly $\#_a(t) > p + 2$ while $\#_c(t) = \#_c(y) = p + 2$, so $t \notin L$. Similarly if $vy$ contains at least one $b$. Hence the case that $z = vwxy$ contains no $c$ leads to a contradiction.

If $z = vwxy$ contains no $a$, then all $a$’s are in $u$, so for each $i$, $\#_a(uv^iwx^iy) = p$.

Since $|vx| \geq 1$, $vy$ must contain at least one symbol and this must be either $b$ or $c$. Suppose that it is $b$. Now we consider $t = uv^0wx^0y$. Clearly $\#_b(t) < p + 1 \leq p$ while $\#_a(t) = \#_a(u) = p$, so $t \notin L$.

Suppose that $vx$ contains only $c$, i.e. no $b$. Consider again $t = uv^0wx^0y$. Now we have $\#_b(t) = p + 1$ while $\#_c(t) < p + 1 \leq p + 1$, so $t \notin L$. A contradiction again, so the language is not context-free.

Notice the importance of the clever choice of $z$!