University of Central Florida
School of Computer Science
COT 4210 Fall 2004

Prof. Rene Peralta

Homework 4 Solutions (by TA Robert Lee)

1.26. Let

$$\Sigma_2 = \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}.$$

Here, Σ_2 contains all possible columns of 0's and 1's of height two. A string of symbols in Σ_2 gives two rows of 0's and 1's. Consider each row to be a binary number and let

$$C = \{w \in \Sigma_2^* \,|\, \text{the bottom row of } w \text{ is three times the top row}\}.$$

For example, $\begin{bmatrix} 0 \\ 0 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \end{bmatrix} \in C$, but $\begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} \notin C$. Show that C is regular. You may assume the result claimed in Problem 1.24.

Solution The result from Problem 1.24 is that regular languages are closed under reversal. Consider the language C^R:

$$C^R = \{w \in \Sigma_2^* \,|\, \text{the bottom row of } w \text{ in reverse is three times the top row in reverse}\}.$$

The NFA $M_{C^R} = (\{a, b, c, d, e\}, \Sigma_2, \delta, a, \{a\})$ recognizes C^R. The transition function δ is given by

δ	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$	ϵ
a	$\{a\}$	$\emptyset$	$\emptyset$	$\{b\}$	$\emptyset$
b	$\emptyset$	$\{c\}$	$\{d\}$	$\emptyset$	$\emptyset$
c	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\{a\}$
d	$\{e\}$	$\emptyset$	$\emptyset$	$\{d\}$	$\emptyset$
e	$\emptyset$	$\{c\}$	$\{d\}$	$\emptyset$	$\emptyset$

It's important to understand why this NFA works. Think about how the sequence of digits in the top row affects the required sequence in the bottom row. Keep in mind that each digit of 1 in a binary string contributes a certain amount to the overall value of the binary number; for example, if the sixth digit is 1, it contributes 2^5 to the value. However, if a digit is 0, that position does not change the overall value of the binary number.

Consider strings with only 0's in the top row. If there is a 0 in the top row, there must also be a 0 in the bottom row in order for the string to be in the language. Furthermore, any number of these $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ dominoes will not change the status (in or out of the language) of a string. Thus we add the self-loop at state a, the only accept state, with input $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$.

Now consider what happens when a 1 appears in the top row. For any natural number i, $3 \cdot 2^i = (2+1) \cdot 2^i = 2^{i+1} + 2^i$. In other words, in the simplest example,

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in C^R,$$

because the top row has value 1 and the bottom row has value 3.

Notice that, as mentioned earlier, if we add $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ dominoes either at the beginning or end of this string, the status of the string (in this case, it is in the language) will not be affected. In the NFA, this type of string is represented by the accepting path $abca$.

However, the initial $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ can also be followed by $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$. (Confirm for yourself that it cannot be followed by $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ or $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$.) When 11 appears in the top row, what must appear in the bottom row is now 11 plus 110, which is 1001. In other words, there is a carry in the addition. In the simplest example,

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in C^R,$$

because the top row has value 3 and the bottom row has value 9. Here we begin to see the "compensation block": eventually this kind of string must end with $\begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. This is due to repeated carries in the addition in the bottom row.

Now consider what happens if there is a $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ in the third position. No matter how many of these dominoes appear, if the "compensation block" appears at the end, the string will be in the language. This type of string is represented by path abd^*eca (abusing the star notation a little) in the NFA, and they take the form

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}^* \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in C^R;$$

in the shortest example, the top row has value 7 and the bottom row has value 21.

Finally, we must consider what happens if the "compensation block" is interrupted with a $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ domino. (Confirm for yourself that it cannot be interrupted with either a $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ or a $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ domino.) In this case, the whole compensation block is once again required. This type of string is represented by path $abdedca$ in the NFA. The shortest example of such a string is

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in C^R,$$

in which the top row has value 11 and the bottom row has value 33.

1.28 Let Σ_2 be the same as in Problem 1.26. Consider the top and bottom rows to be strings of 0's and 1's and let

$$E = \{w \in \Sigma_2^* \,|\, \text{the bottom row of } w \text{ is the reverse of the top row of } w\}.$$

Show that E is not regular.

Solution Proof by contradiction. Assume E is a regular language. By the Pumping Lemma, there is a constant p associated with E.

1. Choose the string $s = \begin{bmatrix} 1 \\ 0 \end{bmatrix}^p \begin{bmatrix} 0 \\ 1 \end{bmatrix}^p$. Note that $s \in L$ because $1^p0^p = (0^p1^p)^R$, and $|s| = 2p \geq p$.

2. Choose the partition $s = xyz$ such that $x = \epsilon$, $y = \begin{bmatrix} 1 \\ 0 \end{bmatrix}^p$, $z = \begin{bmatrix} 0 \\ 1 \end{bmatrix}^p$. Note that $|y| = p \geq p$.

3. In any possible division $y = uvw$, we must have $v = \begin{bmatrix} 1 \\ 0 \end{bmatrix}^m$, where $0 < m \leq p$.

4. Choose $i = 2$. Then $xuv^iwz = xuv^2wz = \begin{bmatrix} 1 \\ 0 \end{bmatrix}^{p+m} \begin{bmatrix} 0 \\ 1 \end{bmatrix}^p$. Because $m > 0$, $1^{p+m}0^p \neq (0^{p+m}1^p)^R$; in other words, the top row is not the reverse of the bottom row. Thus $xuv^2wz \notin L$. This is a contradiction. Therefore L is not a regular language.

1.34 Let x and y be strings and let L be any language. We say that x and y are **distinguishable by *L*** if some string z exists whereby exactly one of the strings xz and yz is a member of L; otherwise, for every string z, $xz \in L$ whenever $yz \in L$ and we say that x and y are **indistinguishable by *L***. If x and y are indistinguishable by L we write $x \equiv_L y$. Show that $\equiv_L$ is an equivalence relation.

Solution We will prove that $\equiv_L$ is reflexive, symmetric, and transitive.

($\equiv_L$ is reflexive) For every string x and for every string z, $xz \in L$ whenever $xz \in L$. Thus $\forall x,\ x \equiv_L x$. Therefore $\equiv_L$ is reflexive.

($\equiv_L$ is symmetric) Assume $x \equiv_L y$. By definition of $\equiv_L$, for every string z, $xz \in L$ whenever $yz \in L$. It follows that $yz \in L$ whenever $xz \in L$. Thus $y \equiv_L x$. Therefore $\equiv_L$ is symmetric.

($\equiv_L$ is transitive) Assume $w \equiv_L x$ and $x \equiv_L y$. By definition of $\equiv_L$, for every string z, $wz \in L$ whenever $xz \in L$. Also, for every string z, $xz \in L$ whenever $yz \in L$. It follows that $wz \in L$ whenever $yz \in L$. Thus $w \equiv_L y$. Therefore $\equiv_L$ is transitive.

We have shown that $\equiv_L$ is reflexive, symmetric and transitive. Therefore $\equiv_L$ is an equivalence relation.

1.36 Let $\Sigma = \{0, 1, +, =\}$ and

$$ADD = \{x = y + z \mid x, y, z \text{ are binary integers, and } x \text{ is the sum of } y \text{ and z}\}.$$

Show that ADD is not regular.

Solution Proof by contradiction. Assume ADD is a regular language. By the Pumping Lemma, there is a constant p associated with ADD.

1. Choose the string s to be $1^p = 0^p + 1^p$. Note that $s \in ADD$ because 1^p is the sum of 0^p and 1^p, and $|s| = 3p + 2 \geq p$.

2. Choose the partition $s = xyz$ such that x is ϵ, y is 1^p, and z is$=0^p{+}1^p$. Note that $|y| = p \geq p$.

3. In any possible division $y = uvw$, the string v must be 1^m, where $0 < m \leq p$.

4. Choose $i = 2$. Then the string xuv^2wz is $1^{m+p} = 0^p + 1^p$. Because $m > 0$, 1^{m+p} is not the sum of 0^p and 1^p. Thus $xuv^2wz \notin ADD$. This is a contradiction. Therefore ADD is not a regular language.

2.6 Give context-free grammars generating the following languages.

2.6a. The set of strings over the alphabet $\{a, b\}$ with twice as many a's as b's.

Solution

$$S \rightarrow SaSaSbS \mid SaSbSaS \mid SbSaSaS \mid \epsilon$$

This rule guarantees that, if a b is added to the string, then two a's are added at the same time. The different permutations in the rule allow the string to be generated in any order.

2.6b. The complement of the language $\{a^n b^n \mid n \geq 0\}$.

Solution

$$\begin{array}{rcl} S & \rightarrow & A \mid B \mid C \\ A & \rightarrow & a \mid aA \mid aAb \\ B & \rightarrow & b \mid Bb \mid aBb \\ C & \rightarrow & DbDaD \\ D & \rightarrow & aD \mid bD \mid \epsilon \end{array}$$

The A rule generates strings of the form $a^m b^n$, where $m > n$; while the B rule generates strings of the form $a^m b^n$, where $m < n$. The C and D rules together generate all strings containing a b before an a.

2.6c. $\{w\#x \mid w^R$ is a substring of x for $w, x \in \{0,1\}^*\}$.

Solution

$$\begin{array}{rcl} S & \rightarrow & AB \\ A & \rightarrow & aAa \mid bAb \mid \#C \\ C & \rightarrow & aC \mid bC \mid \epsilon \\ B & \rightarrow & aB \mid bB \mid \epsilon \end{array}$$

The A rule generates string w to the left of the $\#$ symbol and string w^R to the right of the $\#$ symbol. The C rule generates the substring of x to the left of w^R (and to the right of the $\#$symbol), while the B rule generates the substring of x to the right of w^R.

2.6d. $\{x_1\#x_2\#\cdots\#x_k \mid k \geq 1$, each $x_i \in \{a,b\}^*$, and for some i and j, $x_i = x_j^R\}$.

Solution

$$\begin{aligned}
S &\rightarrow LPR \mid LBR \\
P &\rightarrow aPa \mid bPb \mid a \mid b \mid \epsilon \\
B &\rightarrow aBa \mid bBb \mid \#L \\
R &\rightarrow \#T \mid \epsilon \\
T &\rightarrow \#T \mid Ta \mid Tb \mid \epsilon \\
L &\rightarrow M\# \mid \epsilon \\
M &\rightarrow M\# \mid aM \mid bM \mid \epsilon
\end{aligned}$$

The strings in this language can be divided into two types: strings in which there exist i, j such that $x_i = x_j^R$ and $i = j$, in which case x_i is a palindrome; and strings in which there exist i, j such that $x_i = x_j^R$ and $i \neq j$. (Note that these two sets are not mutually exclusive.)

The P rule generates a palindrome (in other words, a single string that is the reverse of itself). The R and S rules generate additional strings to the right of the palindrome, and the L and M rules generate additional strings to the left of the palindrome.

The B rule generates two strings that are reverses of each other, call them x_i and x_i^R, separated by a $\#$ symbol. The R and S rules generate additional strings to the right of x_i and x_i^R. The L and M rules generate additional strings to the left of x_i and x_i^R, as well as in between x_i and x_i^R.

2.8 Show that the string `the girl touches the boy with the flower` has two different derivations in grammar G_2 on page 93. Describe in English the two different meanings of this sentence.

Solution First derivation:

$$
\begin{aligned}
\langle\text{SENTENCE}\rangle &\rightarrow \langle\text{NOUN-PHRASE}\rangle\langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \langle\text{CMPLX-NOUN}\rangle\langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \langle\text{ARTICLE}\rangle\langle\text{NOUN}\rangle\langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \text{the } \langle\text{NOUN}\rangle\langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \text{the girl } \langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \text{the girl } \langle\text{CMPLX-VERB}\rangle \\
&\rightarrow \text{the girl } \langle\text{VERB}\rangle\langle\text{NOUN-PHRASE}\rangle \\
&\rightarrow \text{the girl touches } \langle\text{NOUN-PHRASE}\rangle \\
&\rightarrow \text{the girl touches } \langle\text{CMPLX-NOUN}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches } \langle\text{ARTICLE}\rangle\langle\text{NOUN}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches the } \langle\text{NOUN}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches the boy } \langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches the boy } \langle\text{PREP}\rangle\langle\text{CMPLX-NOUN}\rangle \\
&\rightarrow \text{the girl touches the boy with } \langle\text{CMPLX-NOUN}\rangle \\
&\rightarrow \text{the girl touches the boy with } \langle\text{ARTICLE}\rangle\langle\text{NOUN}\rangle \\
&\rightarrow \text{the girl touches the boy with the } \langle\text{NOUN}\rangle \\
&\rightarrow \text{the girl touches the boy with the flower}
\end{aligned}
$$

This derivation corresponds to the meaning of the sentence in which the prepositional phrase "with the flower" modifies "the boy"; in other words, the boy has the flower.

Second derivation:

$$
\begin{aligned}
\langle\text{SENTENCE}\rangle &\rightarrow \langle\text{NOUN-PHRASE}\rangle\langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \langle\text{CMPLX-NOUN}\rangle\langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \langle\text{ARTICLE}\rangle\langle\text{NOUN}\rangle\langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \text{the } \langle\text{NOUN}\rangle\langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \text{the girl } \langle\text{VERB-PHRASE}\rangle \\
&\rightarrow \text{the girl } \langle\text{CMPLX-VERB}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl } \langle\text{VERB}\rangle\langle\text{NOUN-PHRASE}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches } \langle\text{NOUN-PHRASE}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches } \langle\text{CMPLX-NOUN}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches } \langle\text{ARTICLE}\rangle\langle\text{NOUN}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches the } \langle\text{NOUN}\rangle\langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches the boy } \langle\text{PREP-PHRASE}\rangle \\
&\rightarrow \text{the girl touches the boy } \langle\text{PREP}\rangle\langle\text{CMPLX-NOUN}\rangle \\
&\rightarrow \text{the girl touches the boy with } \langle\text{CMPLX-NOUN}\rangle \\
&\rightarrow \text{the girl touches the boy with } \langle\text{ARTICLE}\rangle\langle\text{NOUN}\rangle \\
&\rightarrow \text{the girl touches the boy with the } \langle\text{NOUN}\rangle \\
&\rightarrow \text{the girl touches the boy with the flower}
\end{aligned}
$$

This derivation corresponds to the meaning of the sentence in which the prepositional phrase “with the flower” modifies “touches”; in other words, the girl uses the flower to touch.

2.14. Convert the following CFG into an equivalent CFG in Chomsky normal form, using the procedure given in Theorem 2.6.

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

Solution Step 1: add new start symbol. The textbook (page 94) states that the variable on the left-hand-side of the first rule is assumed to be the start symbol.

$$\begin{aligned} S_0 &\to A \\ A &\to BAB \mid B \mid \epsilon \\ B &\to 00 \mid \epsilon \end{aligned}$$

Step 2: remove ϵ from right-hand-side of rules.

$$\begin{aligned} S_0 &\to A \\ A &\to BAB \mid AB \mid BA \mid A \mid B \mid \epsilon \\ B &\to 00 \end{aligned}$$

$$\begin{aligned} S_0 &\to A \mid \epsilon \\ A &\to BAB \mid AB \mid BA \mid A \mid B \mid BB \\ B &\to 00 \end{aligned}$$

Notice that the rule $S_0 \to \epsilon$ provides the only way to generate the empty string in a grammar in Chomsky normal form.

Step 3: remove unit rules.

$$\begin{aligned} S_0 &\rightarrow A \,|\, \epsilon \\ A &\rightarrow BAB \,|\, AB \,|\, BA \,|\, 00 \,|\, BB \\ B &\rightarrow 00 \end{aligned}$$

$$\begin{aligned} S_0 &\rightarrow BAB \,|\, AB \,|\, BA \,|\, 00 \,|\, BB \,|\, \epsilon \\ A &\rightarrow BAB \,|\, AB \,|\, BA \,|\, 00 \,|\, BB \\ B &\rightarrow 00 \end{aligned}$$

Step 4: conversion to proper form.

$$\begin{aligned} S_0 &\rightarrow BA_1 \,|\, AB \,|\, BA \,|\, B_1B_1 \,|\, BB \,|\, \epsilon \\ A &\rightarrow BA_1 \,|\, AB \,|\, BA \,|\, B_1B_1 \,|\, BB \\ B &\rightarrow B_1B_1 \\ B_1 &\rightarrow 0 \\ A_1 &\rightarrow AB \end{aligned}$$

2.17b. Use part (a) to show that the language

$$A = \{w \mid w \in \{a,b,c\}^* \text{ and contains equal numbers of } a\text{'s, } b\text{'s, and } c\text{'s}\}$$

is not a CFL.

Solution Proof by contradiction. Assume that A is a context-free language. Consider the regular language $B = a^*b^*c^*$. Notice that

$$A \cap B = \{a^n b^n c^n \mid n \geq 0\}.$$

By Example 2.20, $A \cap B$ is not context-free. However, the result of problem 2.17a states that the intersection of a context-free language and a regular language is context-free. This is a contradiction. Therefore A is not a context-free language.

2.18 Use the pumping lemma to show that the following languages are not context-free.

2.18a. $L = \{0^n 1^n 0^n 1^n \mid n \geq 0\}$

Solution Proof by contradiction. Assume L is a context-free language. By the Pumping Lemma, there is a constant p associated with L.

1. Choose the string $s = 0^p 1^p 0^p 1^p$. Note that $s \in L$ by choosing $n = p$.

2. Consider the possible divisions $s = uvxyz$, where $|vxy| \leq p$ and $|v| + |y| > 0$. There are four substrings in s, each of size p: the first 0^p, the first 1^p, the second 0^p, and the second 1^p. Because $|vxy| \leq p$, v and y can be located in at most two of these substrings (if vxy straddles a substring boundary). Because $|v| + |y| > 0$, v or y must be located in at least one of these substrings.

3. Choose $i = 2$. Pumping v and y will add at least one character to one of the four substrings of s, causing its character count to become greater than p. Furthermore, there are at least two substrings of s whose character counts cannot be affected by pumping v and y; their character counts remain at p. Thus $uv^2xy^2z \notin L$. This is a contradiction. Therefore L is not a context-free language.

2.18b. $\{0^n\#0^{2n}\#0^{3n} \mid n \geq 0\}$

Solution Proof by contradiction. Assume L is a context-free language. By the Pumping Lemma, there is a constant p associated with L.

1. Choose the string $s = 0^p\#0^{2p}\#0^{3p}$. Note that $s \in L$ by choosing $n = p$.
2. Consider the possible divisions $s = uvxyz$, where $|vxy| \leq p$ and $|v| + |y| > 0$. Either of the substrings v and y could contain a $\#$ symbol. If not, there are three substrings in s not containing a $\#$ symbol: the substring 0^p, the substring 0^{2p}, and the substring 0^{3p}. Because $|vxy| \leq p$, v and y can be located in at most two of these substrings (if vxy straddles a substring boundary).
3. **(Case 1)** v or y contains a $\#$ symbol. Choose $i = 2$. The string xv^2xy^2z will contain more than two $\#$ symbols; thus $uv^2xy^2z \notin L$.
   (Case 2) Choose $i = 2$. Pumping v and y will add at least one character (because $|v| + |y| > 0$) to at least one substring (call it substring A) of s, causing its character count to increase. Furthermore, there is at least one substring (call it substring B) of s whose character count cannot be affected by pumping v and y; its character count does not change. As a result, the ratio between the number of characters in substrings A and B changes. (For example, if substring A is originally 0^p, after pumping it becomes 0^{p+m}, where $0 < m \leq p$; if substring B is originally 0^{3p}, it does not change. The ratio between the size of substring B and substring A is no longer 3.) Thus $uv^2xy^2z \notin L$.

In both cases we have reached a contradiction. Therefore L is not a context-free language.

2.18c. $\{w\#x \mid w \text{ is a substring of } x, \text{ where } w, x \in \{a,b\}^*\}$.

Solution Proof by contradiction. Assume L is a context-free language. By the Pumping Lemma, there is a constant p associated with L.

1. Choose the string $s = a^p b^p \# a^p b^p$. Note that $s \in L$ because $a^p b^p$ is a substring of $a^p b^p$.

2. Consider the possible divisions $s = uvxyz$, where $|vxy| \leq p$ and $|v| + |y| > 0$. There are three cases: vxy is entirely within the substring to the left of the $\#$ symbol, vxy is entirely within the substring to the right of the $\#$ symbol, or vxy straddles the substring boundary.

3. **(Case 1)** vxy is entirely within the substring to the left of the $\#$ symbol. Choose $i = 2$. Because $|v| + |y| > 0$, the number of a's or the number of b's in uv^2xy^2z must be increased in the substring to the left of the $\#$ symbol; however, the number of a's and b's in the substring to the right of the $\#$ symbol remain unchanged. Because there are more characters in the substring to the left, it cannot be a substring of the substring to the right. Thus $uv^2xy^2z \notin L$.
(Case 2) vxy is entirely within the substring to the right of the $\#$ symbol. Choose $i = 0$. Because $|v| + |y| > 0$, the number of a's or the number of b's in uv^2xy^2z must be decreased in the substring to the right of the $\#$ symbol; however, the number of a's and b's in the substring to the left of the $\#$ symbol remain unchanged. Because there are more characters in the substring to the left, it cannot be a substring of the substring to the right. Thus $uv^2xy^2z \notin L$.
(Case 3) vxy straddles the substring boundary.
(Case 3a) If either v or y contains the $\#$ symbol, choose $i = 2$; then uv^2xy^2z will contain more than one $\#$ symbol, hence $uv^2xy^2z \notin L$. Thus neither v nor y can contain the $\#$ symbol. It follows that v must consist entirely of b's, and y must consist entirely of a's. Either v or y must be nonempty, because $|v| + |y| > 0$; consider these two subcases separately.
(Case 3b) v consists entirely of b's and is nonempty. Choose $i = 2$. String uv^2xy^2z now contains more b's in the substring to the left of the $\#$ symbol than in the substring to the right (because the number of b's in that substring cannot be affected by pumping). Hence the substring to the left cannot be a substring of the substring to the right. Thus $uv^2xy^2z \notin L$.
(Case 3c) y consists entirely of a's and is nonempty. Choose $i = 0$. String uv^2xy^2z now contains more a's in the substring to the left of the $\#$ symbol than in the substring to the right (because the number of a's in that substring is decreased by pumping). Hence the substring to the left cannot be a substring of the substring to the right. Thus $uv^2xy^2z \notin L$.

In all cases we have reached a contradiction. Therefore L is not a context-free language.

2.18d. $\{x_1\#x_2\#\cdots\#x_k \mid k \geq 2$, each $x_i \in \{a,b\}^*$, and for some $i \neq j$, $x_i = x_j\}$.

Solution Proof by contradiction. Assume L is a context-free language. By the Pumping Lemma, there is a constant p associated with L.

1. Choose the string $s = a^pb^p\#a^pb^p$. Note that $s \in L$ because $a^pb^p = a^pb^p$.

2. Consider the possible divisions $s = uvxyz$, where $|vxy| \leq p$ and $|v| + |y| > 0$. There are three cases: vxy is entirely within the substring to the left of the $\#$ symbol, vxy is entirely within the substring to the right of the $\#$ symbol, or vxy straddles the substring boundary.

3. **(Case 1)** vxy is entirely within the substring to the left of the $\#$ symbol. Choose $i = 2$. Because $|v| + |y| > 0$, the number of a's or the number of b's in uv^2xy^2z must be increased in the substring to the left of the $\#$ symbol; however, the number of a's and b's in the substring to the right of the $\#$ symbol remain unchanged. Because there are more characters in the substring to the left, it cannot be equal to the substring to the right. Thus $uv^2xy^2z \notin L$.
   (Case 2) vxy is entirely within the substring to the right of the $\#$ symbol. Choose $i = 0$. Because $|v| + |y| > 0$, the number of a's or the number of b's in uv^2xy^2z must be decreased in the substring to the right of the $\#$ symbol; however, the number of a's and b's in the substring to the left of the $\#$ symbol remain unchanged. Because there are more characters in the substring to the left, it cannot be equal to the substring to the right. Thus $uv^2xy^2z \notin L$.
   (Case 3) vxy straddles the substring boundary.
   (Case 3a) If either v or y contains the $\#$ symbol, choose $i = 2$; then uv^2xy^2z will contain more than one $\#$ symbol, hence $uv^2xy^2z \notin L$. Thus neither v nor y can contain the $\#$ symbol. It follows that v must consist entirely of b's, and y must consist entirely of a's. Either v or y must be nonempty, because $|v| + |y| > 0$; consider these two subcases separately.
   (Case 3b) v consists entirely of b's and is nonempty. Choose $i = 2$. String uv^2xy^2z now contains more b's in the substring to the left of the $\#$ symbol than in the substring to the right (because the number of b's in that substring cannot be affected by pumping). Hence the substring to the left cannot be equal to the substring to the right. Thus $uv^2xy^2z \notin L$.
   (Case 3c) y consists entirely of a's and is nonempty. Choose $i = 0$. String uv^2xy^2z now contains more a's in the substring to the left of the $\#$ symbol than in the substring to the right (because the number of a's in that substring is decreased by pumping). Hence the substring to the left cannot be equal to the substring to the right. Thus $uv^2xy^2z \notin L$.

In all cases we have reached a contradiction. Therefore L is not a context-free language.