WS 2006/2007 Dr. C. Stamm / Roland Flury / Stefan Schmid

Discrete Event Systems
Solution to Exercise 5

1 Pumping Lemma revisited

a) Let us assume that L is regular and show that this results in a contradiction.

We have seen that any regular language fulfills the pumping lemma. I.e. there is a p, such that for every word $u \in L$ with $|u| \geq p$ it holds that: u can be written as $u = xyz$ with $|xy| \leq p$ and $1 \leq |y| \leq p$, such that $\forall i \geq 0 : xy^iz \in L$.

In order to obtain the contradiction, we need to show that there is at least one word $w \in L$ with $|w| \geq p$ for which it is not possible to form the string partition $w = xyz$, s.t. $|xy| \leq p$, $1 \leq |y| \leq p$, and $\forall i \geq 0 : xy^iz \in L$.

First, we need to overcome the problem that we do not know the value of p. The standard trick is to consider words whose length depends on p. E.g. consider the word $w = 1^{p^2} \in L$. For sure, $|w| \geq p$.

By the pumping lemma, we can write $w = 1^{p^2}$ as xyz. What remains to show is that there is no partition xyz that satisfies $|xy| \leq p$, $1 \leq |y| \leq p$, and $\forall i \geq 0 : xy^iz \in L$.

The expression $w = xy^iz$ can be written as $xy^iz = 1^{|x|}1^{i|y|}1^{|z|}$. Because $|w| = p^2$, we know that $|z| = p^2 - |x| - |y|$, and therefore, $xy^iz = 1^{|x|}1^{i|y|}1^{p^2-|x|-|y|} = 1^{p^2+(i-1)|y|}$.

To obtain the contradiction, we need to find an $i \geq 0$, such that $xy^iz \notin L$. For example, consider $i = 0$. Then we have $w^0 = xy^0z = 1^{p^2-|y|}$. Clearly, $|w^0| < p^2$, as $|y| \geq 1$. Note that we argue independent of the partition $w = xyz$, we do not pick a specific x and y and therefore the following holds for all possible partitions.

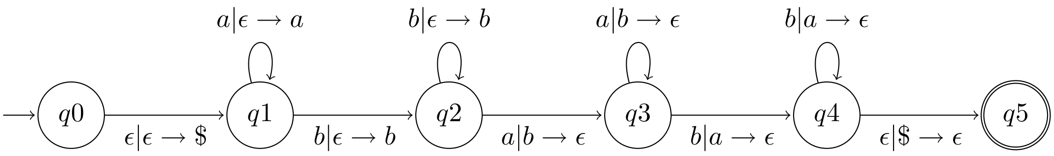
If $w^0 \in L$, then $|w^0|$ is a square number, smaller than p^2. But the next smaller square number, $(p-1)^2$, is strictly smaller than $|w^0|$: $(p-1)^2 = p^2 - 2p + 1 < p^2 - p \leq p^2 - |y| = |w^0|$, which shows that $|w^0|$ cannot be a square number. This shows that there is *no* partition for w that allows to fulfill the pumping lemma conditions. But this should be the case if L is regular. Thus, we have a contradiction, which concludes the proof.

b) Consider the alphabet $\Sigma = \{a_1, a_2, ..., a_n\}$ and the language $L = \bigcup_{i-1}^{n} a_i^*$. The language is regular, as it is the union of regular languages, and the smallest possible pumping number p for L is 1. But any DFA needs at least $n+1$ states to distinguish the n different characters of the alphabet. Thus, for the DFA, we cannot deduce any information from p about the minimum number of states.

The same argument holds for the NFA.

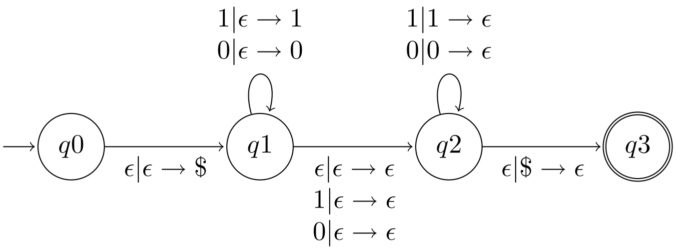
2 Push Down Automaton

a) The PDA first reads all a from the input until it reads a b. For each a it reads, it pushes an a on the stack. Then, the PDA reads all b from the input until there comes an a. Again, for each b on the input, it pushes a b on the stack. Then, the automaton reads a from the input, but only if it can pop a b from the stack. Finally, it reads b from the input as long as it can pop an a from the stack.



b) This PDA should recognize all palindromes. However, we don't know where the middle of the word to recognize is. Therefore, we have to construct a non-deterministic automaton that decides itself when the middle has been reached.

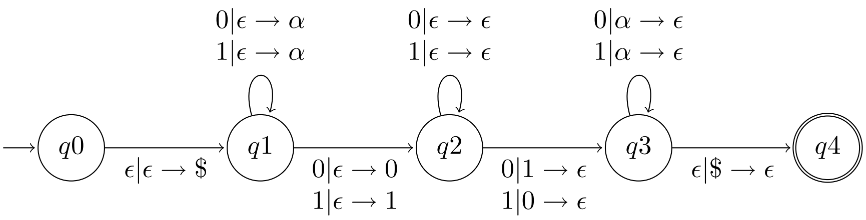
Note that we need to support words of even and odd length. Words of even length have a counter-part for each letter. However, the center letter of an odd word has no counterpart.



c) Consider the word w to be an array of symbols. If $w \in L$, there is at least one offset c, such that $w[c] \neq w[|w| - c]$. That is, there are two symbols x and y in w s.t. $x \neq y$ and the distance of x from the start of w equals the distance of y from the end of w.

The PDA reads $c-1$ symbols, and stores a token α on the stack for each read symbol. Then, it reads the c-th symbol, and puts the symbol onto the stack. Afterwards, the PDA allows to read arbitrarily many symbols from the input, and does not modify the stack. Then, when only c symbols are left on the input stream, the PDA requires that the symbol on the stack must differ to the one on the input. Finally, the PDA reads the remaining $c-1$ symbols and accepts if the stack is empty.

Note that this is again a non-deterministic PDA, as we do not know the value of c.



3 Context Free Grammars

a) If x is not a permutation of y, then x and y contain a different number of a or b.

$$\begin{array}{lcll}
S & \to & D & \textit{x and y differ in number of a} \\
 & \to & E & \textit{x and y differ in number of b} \\
D & \to & BaDaB \mid BaC\#B \mid B\#CaB & \\
E & \to & AbEbA \mid AbC\#A \mid A\#CbA & \\
B & \to & bB \mid \epsilon & \\
A & \to & aA \mid \epsilon & \\
C & \to & aC \mid bC \mid \epsilon &
\end{array}$$

b) We can distinguish 2 cases: either $|x| \neq |y|$ or there is an offset i, such that $x[i] \neq y[i]$, thinking of x and y as arrays.

$$S \quad \to \quad E \qquad |x| \neq |y|$$

$$
\begin{array}{lll}
 & \rightarrow AaC & |x| = |y| \ and\ \exists i : x[i] = a \ and\ y[i] = b \\
 & \rightarrow BbC & |x| = |y| \ and\ \exists i : x[i] = b \ and\ y[i] = a \\
E & \rightarrow DED & \\
 & \rightarrow \#DC & right\ side\ is\ longer \\
 & \rightarrow DC\# & left\ side\ is\ longer \\
D & \rightarrow a \mid b & (a|b) \\
C & \rightarrow DC \mid \epsilon & (a|b)^* \\
A & \rightarrow DAD \mid bC\# & \\
B & \rightarrow DBD \mid aC\# &
\end{array}
$$

Note that for the case $|x| = |y|$, we did not *enforce* that the two strings have equal length. But for the case they have equal length, they differ. (Thus, this grammar is ambiguous.)

4 Tandem Pumping

a) Use the tandem pumping lemma to show that the language is *not* context free. For example, consider the word $w = a^p b^{p+1} c^{p+2}$. Clearly, $w \in L$. The tandem pumping lemma requires that w can be written as $w = uvxyz$ with $|vy| \geq 1$ and $|vxy| \leq p$. For context free languages, it must hold that $uv^ixy^iz \in L \forall i \geq 0$.

The window vxy can be applied at several locations on w. If it entirely covers the a region, then either v or y is at least one a. Therefore, pumping v and y increases the number of a in the resulting word, which violates the language definition.

If the window vxy starts in the area of the a's and ends in the area of b's, then v or y contains at least an a or a b. Again, pumping v and y increases the amount of this symbol, which results in a string not contained in the language. Similarly, if vxy only covers the b region, v or y contains at least one b, which produces strings not in L while pumping.

If the window vxy starts in the b area and ends in the c area, we have several cases: a) If either v or y contains both b and c, pumping w produces words not in L. If $v \in b^+$ and $y = \epsilon$, pumping will produce words with too many b's. If $v \in b^+$ and $y \in c^+$, or if $v = \epsilon$ and $y \in c^+$, we set i to 0 to obtain an string not in L.

If the window vxy entirely covers the c region, then v or y contains at least one c. Thus, setting i to 0 removes at least one c, and the resulting string contains not enough c's to be in L.

b) This language is regular, see Figure 1. Because the set of regular languages is a subset of the context-free languages, the language is also context-free.

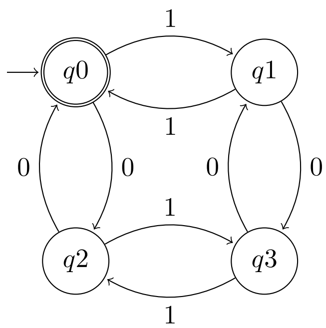


Figure 1: DFA for $L = \{x \mid x \in \{0,1\}^*$, and x contains an even number of '0' and an even number of '1'$\}$

c) Consider the word $w = 0^p1^p\#0^p1^p \in L$. If the language is context free, we can apply the tandem pumping lemma. In order to keep the property that $|x| = |y|$, we must pump the same number of symbols on the left and right of #. Thus, the only reasonable place to place the sub-string vxy is such that v lies to the left of # and y to the right of #. But because $|vxy| \leq p$, v only contains 1 and y only contains 0. Therefore, for any string that we may pump (except for $i = 1$), the number of '0's x does not equal the number of '0's in y (and similarly for the number of '1's.) Therefore, the LHS and RHS of # are not permutations and the pumped strings are not in L. Thus, L is not context free.