Notes on Pumping Lemma

Finite Automata Theory and Formal Languages – TMV027/DIT321

Ana Bove, March 5th 2018

In the course we see two different versions of the Pumping lemmas, one for regular languages and one for context-free languages. In what follows we explain how to use these lemmas.

1 Pumping Lemma for Regular Languages

We can use a variety of tools in order to show that a certain language is regular. For example, we can give a finite automaton that recognises the language, a regular expression that generates the language, or use closure properties to show that the language is regular.

But how to formally show that a language is NOT regular? We could still use closure properties, but this is not always easy. Formally showing that there is no possible automaton that could recognise the language nor a possible regular expression that could generate it is even more difficult. Instead we can try to use the *Pumping lemma for regular languages* which says:

> *Let $\mathcal{L}$ be a regular language. Then, there exists a constant n—which depends on $\mathcal{L}$—such that for every string $w \in \mathcal{L}$ with $|w| \geqslant n$, it is possible to break w into three strings x, y and z such that $w = xyz$ and*
>
> 1. $|xy| \leqslant n$;
> 2. $y \neq \epsilon$;
> 3. $\forall k \geqslant 0.\ xy^k z \in \mathcal{L}$.

Note that the lemma states certain things to happen whenever a language is regular. Note also that the string y to be "pump" is always placed within the first n symbols in the word.

When we want to prove that a certain language $\mathcal{L}$ is NOT regular, we start by assuming the language is regular. Then, the Pumping lemma for regular languages must apply. We reason now using the information provided by the lemma until we arrive to a contradiction. Since the only assumption we have made in the reasoning was that the language $\mathcal{L}$ was regular, then this must the source of the problem and thus, we can conclude that $\mathcal{L}$ is NOT regular.

More concretely, if we want to prove that $\mathcal{L}$ is not regular we proceed as follows:

1. We assume that $\mathcal{L}$ is regular; hence the Pumping lemma for regular languages must apply.

2. The Pumping lemma tells us that there must exist a constant depending on $\mathcal{L}$; let us give a name to this constant, say n, so we can easily refer to it. Only now we can start using this constant!

3. Now we know that certain things must hold whenever we have a word in the language which is longer than the constant n. Let us continue by picking a concrete word w in the language which is larger than n. Observe that it needs to be clear that indeed $|w| \geqslant n$, otherwise this will need to be shown. In practice, since we do not really know the value of n, w will probably need to use n as part of its definition in order to actually guarantee that $|w| \geqslant n$!

4. From the lemma we know that this word w can be split into three parts x, y and z. Observe that we do NOT know exactly how these parts look like, only that $w = xyz$, $|xy| \leqslant n$ and $y \neq \epsilon$, which is the information provided by the lemma!

5. According to the Pumping lemma it must also be the case that for all $k \geqslant 0$ then $xy^kz \in \mathcal{L}$. That is, if we "pump" the part y as many times as we wish, then the word xy^kz must still belong to the language $\mathcal{L}$. This means that if we find at least ONE value k for which xy^kz does NOT belong to $\mathcal{L}$, then we arrive to a contradiction since xy^kz must be in $\mathcal{L}$ for ANY k!

Let us now look at the use of the Pumping lemma for regular languages in more detail with the help of some examples. In what follows, $\#_0(w)$ is the number of occurrences of the symbol 0 in the word w and $\#_1(w)$ is the number of occurrences of 1 in w.

1.1 $\mathcal{L}_1 = \{0^i1^i2^i \mid i > 0\}$ is not a Regular Language

Let us assume the language $\mathcal{L}_1$ is regular. Then the Pumping lemma for regular languages applies for $\mathcal{L}_1$.

Let n be the constant given by the Pumping lemma.

Let $w = 0^n1^n2^n$. Clearly $w \in \mathcal{L}_1$ and $|w| \geqslant n$.

By the lemma we know that $w = xyz$ with $|xy| \leqslant n$ and $y \neq \epsilon$.

Given our choice of w, and given that xy is located at the beginning of the word and that it is not longer than n, then xy should contain only 0's.

Since $y \neq \epsilon$ then y should contain at least one 0 and since xy should contain only 0's then y will also contain only 0's!

Then for $k = 2$, the word $xy^2z = xyyz$ will contain more 0's than 1's and 2's (recall that y will contain at least one 0!) and hence xy^2z will not belong to $\mathcal{L}_1$, which contradicts the Pumping lemma. (Actually this will be the case for any $k > 1$!)

Therefore $\mathcal{L}_1$ cannot be regular.

Observe that even $k = 0$ would have help us showing that the language is not regular. Here however, the word $xy^0z = xz$ will contain less 0's than 1's and 2's. Still xy^0z will not belong to $\mathcal{L}_1$, which would also contradict the Pumping lemma.

1.2 $\mathcal{L}_2 = \{w \in \{0,1\}^* \mid \#_1(w) = \#_0(w)\}$ is not a Regular Language

Let us assume the language $\mathcal{L}_2$ is regular. Then the Pumping lemma for regular languages applies for $\mathcal{L}_2$.

Let n be the constant given by the Pumping lemma.

Let $w = 1^n 0^n$. Clearly $w \in \mathcal{L}_2$ and $|w| \geqslant n$.

By the lemma we know that $w = xyz$ with $|xy| \leqslant n$ and $y \neq \epsilon$.

Given our choice of w, and given that xy is located at the beginning of the word and that it is not longer than n, then xy should contain only 1's.

Since $y \neq \epsilon$ then y should contain at least one 1 and since xy should contain only 1's then y will also contain only 1's!

Let $x = 1^j$ for $j \geqslant 0$, $y = 1^i$ for $i \geqslant 1$ (hence $y \neq \epsilon$!) and $z = 1^{n-j-i}0^n$. That is, $w = 1^j 1^i 1^{n-j-i} 0^n = 1^n 0^n$ as expected.

Then for $k = 2$, the word $xy^2z = xyyz = 1^j 1^{2i} 1^{n-j-i} 0^n = 1^{n+i} 0^n$ will contain more 1's than 0's since $n + i > n$ when $i \geqslant 1$. Hence xy^2z will not belong to $\mathcal{L}_2$, which contradicts the Pumping lemma. (Actually this will be the case for any $k > 1$!)

Therefore $\mathcal{L}_2$ cannot be regular.

Discussion note: In this exercise, we could have chosen for example w to be the word $(01)^n$. This word is also such that $w \in \mathcal{L}_2$ and $|w| \geqslant n$.

Observe that this particular choice of word does not really allow us to continue reasoning as desired. By the Pumpung lemma we know that xy belongs to the first half of the word (the length of w is $2n$) and we know that $y \neq \epsilon$. But these two facts do not give enough information to have any meaningful reasoning arriving to a contradiction. Recall that we really do not know how the actual parts x, y and z look like. It could perfectly be the case that $x = \epsilon$, $y = 01$ and $z = (01)^{n-1}$. Here, for any $k \geqslant 0$ we have that $xy^kz \in \mathcal{L}_2$!

So the choice of the word plays an important role when reasoning about the Pumping lemma for certain languages.

1.3 $\mathcal{L}_3 = \{w \mid \#_0(w) > 2 \times \#_1(w)\}$ is not a Regular Language

Let us assume the language $\mathcal{L}_3$ is regular. Then the Pumping lemma for regular languages applies for $\mathcal{L}_3$.

Let n be the constant given by the Pumping lemma.

Let $w = 0^{2n+1}1^n$. Clearly $w \in \mathcal{L}_3$ and $|w| \geqslant n$.

By the lemma we know that $w = xyz$ with $|xy| \leqslant n$ and $y \neq \epsilon$.

Given our choice of w, and given that xy is located at the beginning of the word and that it is not longer than n, then xy should contain only 0's.

Since $y \neq \epsilon$ then y should contain at least one 0 and since xy should contain only 0's then y will also contain only 0's!

Then for $k = 0$, the word $xy^0z = xz$ will contain at most $2n$ 0's (recall that $y \neq \epsilon$ so we are removing at least one 0 when we remove the part y!) and n 1's. Hence $\#_0(w) \leqslant 2 \times \#_1(w)$ and then w will not belong to $\mathcal{L}_3$, which contradicts the Pumping lemma. (Observe that this is the only value of k which allows us to arrive to a contradiction for this particular word!)

Therefore $\mathcal{L}_3$ cannot be regular.

2 Pumping Lemma for Context-Free Languages

The procedure is similar when we work with context-free languages. In order to show that a language is context-free we can give a context-free grammar that generates the language, a push-down automaton that recognises it, or use closure properties to show

that the language is context-free. But if we want to show that a language is NOT context-free, the *Pumping lemma for context-free languages* is a very useful tool. The lemma says:

> *Let $\mathcal{L}$ be a context-free language. Then, there exists a constant n—which depends on $\mathcal{L}$—such that for every $w \in \mathcal{L}$ with $|w| \geqslant n$, it is possible to break w into five strings x, u, y, v and z such that $w = xuyvz$ and*
>
> 1. *$|uyv| \leqslant n$;*
> 2. *$uv \neq \epsilon$, that is, either u or v is not empty;*
> 3. *$\forall k \geqslant 0.\ xu^k yv^k z \in \mathcal{L}$.*

Here as well the lemma states certain things to happen whenever a language is context-free. The way we use the lemma is also similar to that in the previous section. If we want to prove that a language $\mathcal{L}$ is not context-free we start by assuming it is, and then we reason using the information provided by the Pumping lemma (which must apply if indeed the language is context-free!) until we arrive to a contradiction. Since the only assumption we have made in the reasoning was that the language $\mathcal{L}$ was context-free, then this must the source of the problem and thus, we can conclude that $\mathcal{L}$ is NOT context-free.

An important difference, however, is that we now have no information about where in the word the parts that can be "pumped" are located! The lemma tells us that given a word w in the language which is longer than the constant provided by the lemma, the word can be divided into five parts x, u, y, v and z such that $w = xuyvz$. We know that uyv is not longer than the constant provided by the lemma but we do not know where in the word w the part uyv is located. So here we need to reason for ALL possible locations of uyv within the word w.

So, if we want to prove that $\mathcal{L}$ is not context-free we proceed as follows:

1. We assume that $\mathcal{L}$ is context-free; hence the Pumping lemma for context-free languages must apply.

2. The Pumping lemma tells us that there must exist a constant depending on $\mathcal{L}$; let us give a name to this constant, say n, so we can easily refer to it. Only now we can start using this constant!

3. Now we know that certain things must hold whenever we have a word in the language which is longer than the constant n. Let us continue by picking a concrete word w in the language which is larger than n. Observe that it needs to be clear that indeed $|w| \geqslant n$, otherwise this will need to be shown. In practice, since we do not really know the value of n, w will probably need to use n as part of its definition in order to actually guarantee that $|w| \geqslant n$!

4. From the lemma we know that this word w can be split into five parts x, u, y, v and z. Observe that we do NOT know exactly how these parts look like, only that $w = xuyvz$, $|uyv| \leqslant n$ and $uv \neq \epsilon$, which is the information provided by the lemma! In particular, we do not know how x looks like, and hence we do not know where exactly uyv is placed within the word!

5. According to the Pumping lemma it must also be the case that for all $k \geqslant 0$ then $xu^k yv^k z \in \mathcal{L}$. That is, if we "pump" the parts u and v as many times as we wish (but the same amount of times for both parts), then the word $xu^k yv^k z$ must still belong to the language $\mathcal{L}$. This means that if we find at least ONE value k for which $xu^k yv^k z$ does NOT belong to $\mathcal{L}$ for ANY possible location of uyv within w, then we arrive to a contradiction since $xu^k yv^k z$ must be in $\mathcal{L}$ for ANY k and ANY location of uyv within w! Observe however that the value of k which makes $xu^k yv^k z$ not belonging to $\mathcal{L}$ does not need to be the same of each of the possible locations of uyv within w.

Let us now look at the use of the Pumping lemma for context-free languages in more detail with the help of some examples.

2.1 $\mathcal{L}_1 = \{0^i 1^i 2^i \mid i \geqslant 0\}$ is not a Context-free Language

Let us assume the language $\mathcal{L}_1$ is context-free. Then the Pumping lemma for context-free languages applies for $\mathcal{L}_1$.

Let n be the constant given by the Pumping lemma.

Let $w = 0^n 1^n 2^n$. Clearly $w \in \mathcal{L}_1$ and $|w| \geqslant n$.

By the lemma we know that $w = xuyvz$ with $|uyv| \leqslant n$. We know also that $uv \neq \epsilon$, and hence we know that either u or v could be empty but not both.

Since $|uyv| \leqslant n$ then we know that there is one number $p \in \{0, 1, 2\}$ that *does not* occur in uyv and hence neither in uv.

Since $uv \neq \epsilon$ then we know that there is another number $q \in \{0, 1, 2\}$ that *does* occur in uv.

Observe that $p \neq q$ because a number cannot occur AND not occur in uv at the same time!

Then the number q (occurring in uv) has more occurrences than the number p (not occurring in uv) in xu^2yv^2z, whatever q and p are. Hence xu^2yv^2z does not belong to $\mathcal{L}_1$. Therefore $\mathcal{L}_1$ cannot be context-free.

2.2 $\mathcal{L}_4 = \{a^i b^{2i} c^i \mid i > 0\}$ is not a Context-free Language

Let us assume the language $\mathcal{L}_4$ is context-free. Then the Pumping lemma for context-free languages applies for $\mathcal{L}_4$.

Let n be the constant stated by the Pumping lemma.

Let $w = a^n b^{2n} c^n$. Clearly $w \in \mathcal{L}_4$ and $|w| \geqslant n$.

By the lemma we know that $w = xuyvz$ with $|uyv| \leqslant n$. We know also that $uv \neq \epsilon$, and hence we know that either u or v could be empty but not both.

We have five different possibilities for the location of uyv within w (recall we do not know where exactly uyv is located in the word):

1. uyv consists only of a's: then for any $k > 1$ the word $xu^k yv^k z$ will have more a's than c's (because $uv \neq \epsilon$), and hence it will not belong to $\mathcal{L}_4$;

2. uyv consists only of b's: then for any $k > 1$ the word $xu^k yv^k z$ will have more b's than a's and c's together (again because $uv \neq \epsilon$), and hence it will not belong to $\mathcal{L}_4$;

3. uyv consists only of c's: then for any $k > 1$ the word $xu^k yv^k z$ will have more c's than a's (because $uv \neq \epsilon$), and hence it will not belong to $\mathcal{L}_4$;

4. uyv consists of a's and b's: recall that we know that $uv \neq \epsilon$ but we do not really know if both u and v are non-empty. So, for any $k > 1$, when pumping u and v we could actually only be pumping a's (if u is not empty and contains only a's, and v is empty), only pumping b's (if u is empty, and v is not empty and contains only b's), or we could be pumping both a's and b's (if neither u nor v are empty, and/or when at least one of them contains both a's and b's). We do know however that we are pumping at least one a or one b, and that we are not pumping any c's. So we know that xu^kyv^kz will not belong to $\mathcal{L}_4$ because we would have more a's than c's (whenever we pump a's), or more b's than a's and c's together (whenever we pump b'a but no a's). Observe that in the case u and/or v contain occurrences of both a's and b's, xu^kyv^kz will not even have the symbols in the right order when $k > 1$;

5. uyv consists of b's and c's: this is similar to the case before, only that for any $k > 1$ we will be pumping at least one b or one c, but we will not pump any a's. So we know that xu^kyv^kz will not belong to $\mathcal{L}_4$ because we would have more c's than a's (whenever we pump c's), or more b's than a's and c's together (whenever we pump b's but no c's). Even here, the order of the symbols in xu^kyv^kz might not be the right one if either u or v contain occurrences of both b's and c's when $k > 1$.

So, for each of these five possibilities there exists at least one k for which xu^kyv^kz does not belong to $\mathcal{L}_4$. Therefore $\mathcal{L}_4$ cannot be context-free.

Observe that in this example, uyv cannot consists of a's, b's AND c's. This is due to the fact that $|uyv| \leqslant n$ and also that w has $2n$ b's between the a's and the c's. Hence no part of the word containing at most n symbols could consists of all three letters.

2.3 $\mathcal{L}_5 = \{s2s \mid s \in \{0,1\}^*\}$ is not a Context-free Language

Let us assume the language $\mathcal{L}_5$ is context-free. Then the Pumping lemma for context-free languages applies for $\mathcal{L}_5$.

Let n be the constant given by the Pumping lemma.

Let $w = 0^n1^n20^n1^n$. Clearly $w \in \mathcal{L}_5$ and $|w| \geqslant n$.

By the lemma we know that $w = xuyvz$ with $|uyv| \leqslant n$. We know also that $uv \neq \epsilon$, and hence we know that either u or v could be empty but not both.

We have three different possibilities for the location of uyv within w.

If uyv takes place completely before the 2, then for $k = 0$ and since $uv \neq \epsilon$, whatever part of the word is removed before the 2 will not be removed after the 2. Hence xu^kyv^kz will not have the form $s2s$ and therefore it will not belong to $\mathcal{L}_5$.

We have a similar reasoning if uyv takes place completely after the 2.

If 2 is part of uyv then uyv should be of the form 1^i20^j for i, j such that $i + j + 1 \leqslant n$. Even here, for $k = 0$ and since $uv \neq \epsilon$, the resulting word is not of the form $s2s$ either because we remove the 2 (if 2 is part of uv), or because we remove 1's (right before the 2) and/or we remove 0's (right after the 2) but we do not remove 1's at the end of the word nor 0's at the beginning of the word. Hence xu^kyv^kz will not have the form $s2s$ and therefore it will not belong to $\mathcal{L}_5$.

So, for each of these three possibilities xu^0yv^0z does not belong to $\mathcal{L}_5$. Therefore $\mathcal{L}_5$ cannot be context-free.