PUMPING LEMMAs FOR CFL AND RL

These are Only Necessary Conditions:

- The Pumping Lemma for CFL (PL-CFL) is a *necessary* condition for CFLs, i.e., if L is a CFL then it satisfies PL-CFL.
- Similarly, for Pumping Lemma for RL (PL-RL), i.e., if L is a RL, then it satisfies PL-RL.

PL-RL is a more restrictive (special) form of PL-CFL:

- Since each RL is also a CFL, each RL also satisfies PL-CFL.
- Since a CFL may not be a RL, a CFL may not satisfies PL-RL.

Main Uses:

- Show that a language L is not regular by showing that it does not satisfy PL-RL.
  - $L_{a^n b^n}$ does not satisfy PL-RL (and hence not an RL).
  - $L_{has\text{-}11}$ satisfies RL-PL (and hence satisfies RL-CFL).
- Show that a language L is not context-free by showing that it does not satisfy PL-CFL.
  - $L_{a^n b^n c^n}$ does not satisfy PL-CFL and hence not a CFL.
  - $L_{a^n b^n}$ satisfies PL-CFL.

Question:

•? Which pumping-lemmas will be satisfied by L_{sym}?

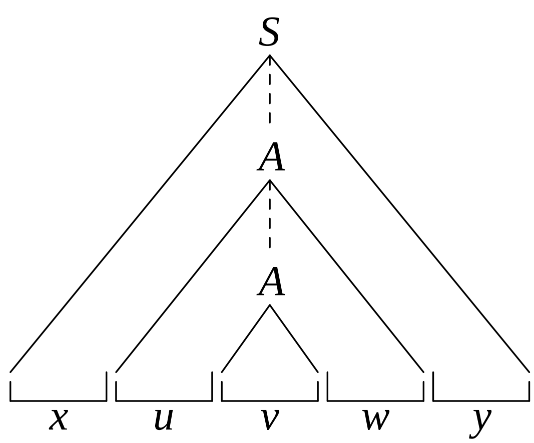
•? Which pumping-lemmas will be satisfied by the language of special binary multiplications $\{10^m \times 10^n = 10^{m+n}: m, n \geq 0\}$?

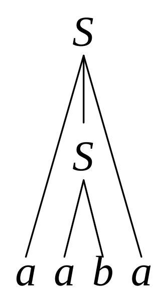
•? How about $\{x \times y = z$: where $x, y, z \in 1(0+1)^*$ and binaryNum(z) equals the product of binaryNum(x) and binaryNum(y)$\}$?

PUMPING LEMMA FOR CFL

Observations on CFG:

- We can eliminate all rules of the form $A \rightarrow B$ from the grammar.
- A parse-tree of depth d can derive a string of length $\leq m^d$, where m = max. length of the right side of a rule.
- If $L = L(G)$ is infinite, then there are arbitrarily long strings in L and hence parse-trees of arbitrarily large depth.
- If $|V(G)| = n$, then a parse-tree of depth $> n$ will have some variable A repeating on a path from the root.
- This means we can derive from A a string of the form uAw, where $uw \in T^+$. Such an A may be called a *recursive* variable.





$A = S$; $x = y = \lambda$
$u = a$, $v = ab$, $w = b$

Some Important Consequences:

- Replacing the upper A-subtree by the lower A-subtree gives $xvy \in L$.
- Replacing the lower A-subtree by the upper A-subtree gives $xuuvwwy \in L$. Likewise, $xu^k vw^k y \in L$ for $k \geq 1$.
- No recursion anywhere in the lower A-subtree means $|v| \leq m^n$.
- No recursion in the upper A-subtree, save the one shown, means $|uw| \leq m^n$.

PUMPING LEMMA FOR CFL

Pumping Lemma (PL-CFL).

- For each CFL L, there exist an integer $N > 0$ (which may depend on L) such that every $s \in L$ of length $|s| \geq N$ can be written as $s = xuvwy$ with the following properties:

  (1) $0 < |uw| < |uvw| \leq N$ ($v \neq \lambda$ and at least one of u and $w \neq \lambda$).
  (2) For all $k \geq 0$, $xu^k vw^k y \in L$.
  (3) Either or both of x, y may be λ.

- The decomposition $s = xuvwy$ may depend on L.
- The location of uvw in s may depend on s and L, and cannot be chosen arbitrarily.
- The pair $\langle u, w\rangle$ is called the *pump*; a pump is two sided if $u \neq \lambda \neq w$.
- Fuiding a pump includes the part v, the context of the pump.

Example 1. $N = 4$ works for PL-CFL for $L = \{a^n b^n: n \geq 1\}$.

- The smallest string s of length ≥ 3 is $s = aabb$. Any pump uw must satisfy the following conditions in order for $xu^k vw^k y \in L$.

  (i) $\#(a, uw) = \#(b, uw)$.

  (ii) Each of u and w should consists of only a's or only b's in order to avoid mixing of a's and b's in $xu^k vw^k y$ for $k > 1$.

- From (i)-(ii), we get $u = a^m$ and $w = b^m$ for some $m \geq 1$.
- $u = a^2$ and $w = b^2$ does not work because $s = aabb = \lambda.u.\lambda.w.\lambda$ is a bad (becasue $v = \lambda$) and only decompostion; also, $xvy = \lambda \notin L$.
- $u = a$ and $w = b$ works. For any $s = a^n b^n$, $n \geq 2$, the decomposition $s = a^{n-2}. a. ab. b. b^{n-2}$ satisfies the conditions in PL-CFL.
- $N = 2$ does not work; there is no pump in $s = ab \in L$.

MORE EXAMPLES OF PUMP IN CFL

- For $L_{a^n b^n}$, $N = 3$ also works, with a slightly different decomposition.

  $$a^n b^n = a^{n-1}.\, a.\, b.\, b.\, b^{n-2}, \text{ with } u = a \text{ and } v = w = b.$$

  This decompostion is related to the following CFG for $L_{a^n b^n}$:

  $$S \to aB,\ B \to aBb \mid b.$$

  Another similar decompostion is $a^n b^n = a^{n-2}.\, a.\, a.\, b.\, b^{n-1}$, with $u = a = v$ and $w = b$.

- For $L_{a^m b^n} = \{a^m b^n\colon m \geq n \geq 1\}$, the smallest string in the language is ab and $N = 4$ works.

  $$\begin{aligned} a^m b &= a^{m-1}.\, a.\, b.\, \lambda.\, \lambda \text{ for } m > 1 \\ a^m b^n &= a^{m-1}.\, a.\, b.\, \lambda.\, b^{n-1}, \text{ when } m > n \\ a^m b^m &= a^{m-1}.\, a.\, ab.\, b.\, b^{m-1},\ m \geq 2 \end{aligned}$$

  This corresponds to the following CFG for $L_{a^m b^n}$:

  $$S \to ab \mid aSb \mid aAb,\ A \to aA \mid a$$

- For $L_{a^m b^n c^{m+n}}$, the samllest string in the language is $abcc$ and $N = 6$ works (there is no string of length 5 in the language).

  $$\begin{aligned} a^m bc^{m+1} &= a^{m-1}.\, a.\, b.\, c.\, c^m\ (m > 1) \\ a^m b^n c^{n+1} &= a^m b^{n-2}.\, b.\, bc.\, c.\, c^{m+n-2}, \text{ for } n > 1 \end{aligned}$$

NON-CFL LANGUAGE

- If a language L does not satisfy PL-CFL, i.e., there is no N for which the pumping conditions (1)-(3) hold for all string $s \in L$ with $|s| \geq N$, then L is not CFL (hence not a regular language either).

Example 2. $L = \{a^n b^n c^n: n \geq 1\}$ is not a CFL.

- We first show that $N = 6$ does not work; the same argument shows that no N works, i.e., L does not satisfy PL-CFL and hence L is not a CFL.
- Let $s = aabbcc \in L$, $|s| \geq 6$. If possible, let $s = xuvwy$ be a proper decomposition that satisfies the conditions in PL-CFL. Then,
  - (i) The number of a's, b's, and c's are the same in uw.
  - (ii) Each of u and w is made of just one symbol from $\{a, b, c\}$.
- The condition (ii) means that u should consists of a's and w should consist of b's, but then (i) cannot be satisfied.
- Thus, there is no decomposition $s = xuvwy$ as desired.

Question:

•? Show that the language of binary multipications of the form $2^m \times 2^n = 2^{m+n}$, i.e, the language $\{10^m \times 10^n = 10^{m+n}: m, n \geq 0\}$ satisfies PL-CFL. Does this mean this language is a CFL?

•? Show that $\{x \times y = z$: where $x, y, z \in 1(0+1)^*$ and binaryNum(z) equals the product of binaryNum(x) and binaryNum(y)$\}$ does not satisfy PL-CFL. What does that say about this language? (Hint: consider multiplication of numbers of the form 2^m and $2^{2m} - 2^m$.)

PUMPING LEMMA FOR REGULAR LANGUAGES

Pumping Lemma (PL-RL).

- For each regular language L, there exists an integer $N > 0$ (which may depend on L) such that every $s \in L$ of length $|s| \geq N$ can be written as $s = xuy$ with the following properties:

  (1) $0 < |u| \leq N$ (actually, one can say that $0 < |u| \leq |xu| \leq N$)
  (2) For all $k \geq 0$, $xu^k y \in L$.

- The pump u can depend on s and on L. The pump u relates to a cycle (loop) in the FSA or NFSA for L. Thus, N can be taken to be the minimum number of states in (N)FSA for L.

Notes:

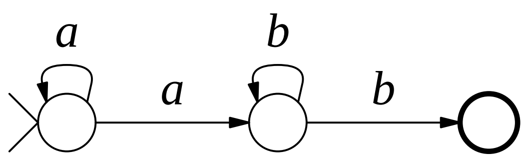
- The coditions (1)-(2) above are obtained by putting $w = \lambda$ in the conditions (1)-(2) for the pumping lemma for CFL.
- Unlike CFL, we can assure that the pump u is not far from the beginning of the string s.
- Since the reverse of a regular language is also regular, we also get a pump close to the end of s. Thus, for $|s| \geq 2N$, there will be a pump which is towards the beginning of s and a disjoint pump (without any overlap with the pump on the left) towards the end of s.
- One can actually get a regular pump on any part of a large string s in a regular language in the following sense. For any string $s = xyz \in L$, where $|s| \geq |y| \geq N$, we can write $y = uvw$ such that $0 < |v| \leq N$ and $xuv^k wz \in L$ for all $k \geq 0$.

Similarities between PL-CFL and PL-RL:

- If $N = N_0$ works for the PL-CFL for an L, then any $N > N_0$ also works for that L. The same is true for PL-RL.

EXAMPLE OF PUMPS IN AN RL

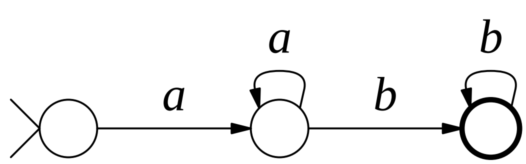
- Let $L = a^+b^+ = \{ab, aab, abb, aaab, aabb, abbb$i$, \cdots\}$.
  - Here, $N = 3$ works and there are two kinds of pumps depending on $s \in L$ as shown below. (N must be larger than the length of the smallest string in L.)
  - For $s = ab^n$ and $n \geq 2$, $s = a.\, b.\, b^{n-1}$ is a valid decomposition.
  - For $s = a^m b^n$ and $m \geq 2$, $s = \lambda.\, a.\, a^{m-1}b^n$ is a valid decomposition.



Each pump corresponds to a cycle
or loop in this NFSA for a^+b^+.

  - The valid decompostions look slightly different in terms of the (min-state) FSA for a^+b^+.

For $s = ab^n$ and $n \geq 2$: $s = ab.\, b.\, b^{n-2}$.
For $s = a^m b^n$ and $m \geq 2$: $s = a.\, a.\, a^{m-2}b^n$.



Each pump corresponds to a cycle
or loop in this FSA for a^+b^+.

- There are many other valid decomposition of the form $s = xuy$, with $|u| \leq N$, if we do not insist on $|xu| \leq N$.
- It is easy to see that a^+b^+ satisfi es PL-CFL, and that $L_{a^n b^n}$ does not satisfy PL-RL.

EXERCISE.

1. Find the smallest N which satisfies PL-CFL for $L_{bal\text{-}par}$. Repeat the exercise for L_{sym}.

2. Find the smallest N which satisfies PL-CFL for the following language $L_{m \geq n} = \{a^m b^n \colon m \geq n \geq 1\}$. Note that the pumps look different for different $s \in L_{m \geq n}$. Repeat the exercise for $L_{m \neq n} = \{a^m b^n \colon m \neq n,\ m \geq 1 \text{ and } n \geq 1\}$. (Do you notice any thing special about how the pumps change whether $m > n$ or $m < n$?)

3. Show that the language $L_{m,n,m+n} = \{a^m b^n c^{m+n} \colon m,\ n \geq 1\}$ satisfies PL-CFL. (You will need different pumps depending on whether n is large or small; you need to describe the nature of the pump in each situation.)

4. Consider the languages $L_{m,m,n} = \{a^m b^m c^n \colon m \geq 1,\ n \geq 1\}$ and $L_{m,n,n} = \{a^m b^n c^n \colon m \geq 1,\ n \geq 1\}$. For $s = a^2 b^2 c^2 \in L_{m,m,n} \cap L_{m,n,n}$, compare the pumps for s computed with respect to $L_{m,m,n}$ and $L_{m,n,n}$, respectively. After generalizing the observation to $a^j b^j c^j$ (why do we need to generalize it to $j > 2$), argue that $L_{m,m,n} \cap L_{m,n,n} = L_{n,n,n} = \{a^n b^n c^n \colon n \geq 1\}$ is not context-free.

5. Show that the binary additions presented as a language over the alphabet $\{0, 1, +, =\}$ is not a CFL.

6. Does the strings of the form $10^n{+}0^n1{=}10^{n-1}1$ satisfy CFL-pumping lemma? How about the strings of the form $10^n{+}1{=}10^{n-1}1$?

7. Show that the binary multiplication language over the alphabet of binary triplets $\{t_0,\ t_1,\ \cdots,\ t_7\}$ does not satisfy CFL-pumping lemma. (Hint: exploit the special role of t_6 which cannot be part of any pump.)

8. What is wrong with the following statement for the pumping lemma for CFL:

   There exists an integer $N \geq 1$ such that every string of the form $xzy \in L$, with $0 < |z| \leq N$, one can decompose z as $z = uvw$ such that $|uw| > 0$, $|v| > 0$, and $xu^k v w^k y \in L$ for all k

≥ 0.

Give an example of CFL that does not satisfy the above statement.

9. What is wrong with the following statement for the condition that L does not satisfy the Pumping Lemma for CFL?

   L has strings of the form $|xuvwy| \geq N$, $N \geq 1$, such that $uw \neq \lambda \neq v$ and $|uvw| \leq N$ such that $xu^k vw^k y \notin L$ for all $k \neq 1$.

   Give a correct form of the above.

10. Show that $L_{bal\text{-}sym}$, the balanced parenthetical strings which are symmetric, do not form a context-free language; $L_{bal\text{-}sym} = \{ab, aabb, abab, aaabbb, aababb, ababab, \cdots\} = L_{bal} \cap L_{sym}$.

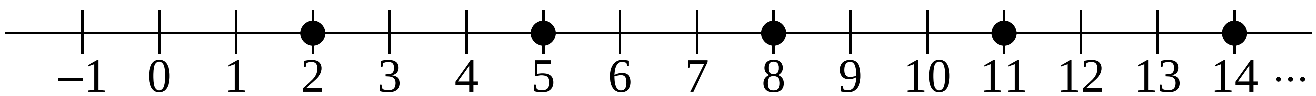
11. Show that none of the languages $\{a^k b^m c^n \colon k \geq m \geq n \geq 1\}$ and $\{a^m b^n c^{m+n} \colon m \geq n \geq 1\}$ satisfi es the pumping lemma for CFL.

SEMI-LINEAR SETS

Semi-linear Set on line: More general than arithmatic progression.

- Simple form: $\{m + k.\, n : k \geq 0\}$, where m, n are fixed integers ≥ 0.
- More general: $\{m + k_1.\, n_1 + k_2.\, n_2 + \cdots + k_p.\, n_p :$ each $k_i \geq 0\}$, where m and n_i's are fixed integers ≥ 0.

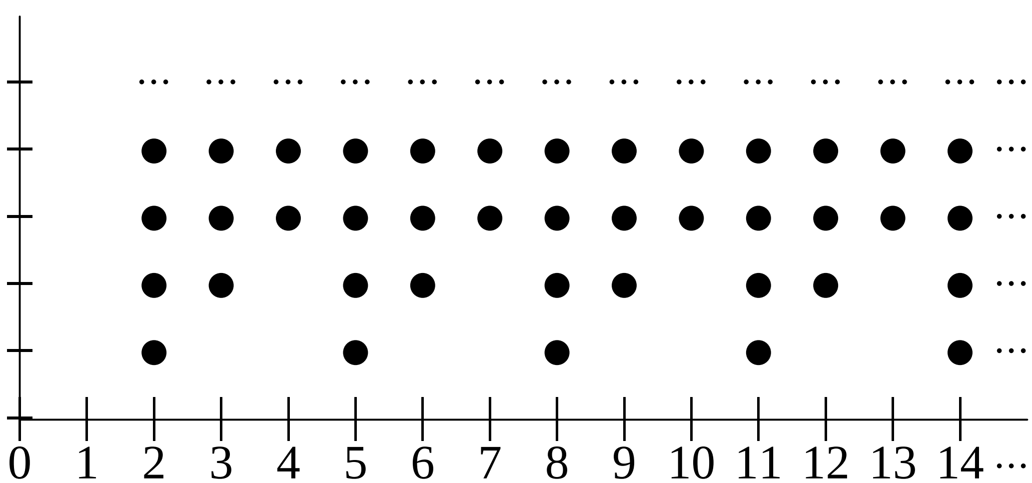
Example. $m = 2$, $n = 3$, and $p = 1$ give the set $\{2, 5, 8, 11, 14, \cdots\}$.



Semi-linear Set on the Plane:

- $\{m + k_1.\, n + k_2.\, n_2 + \cdots + k_p.\, n_p :$ each $k_i \geq 0\}$, where $m = (m_1, m_2)$ and $n_i = (n_{i1}, n_{i2})$'s are fixed integer vectors with coordinates ≥ 0.

Example. For $m = (2,1)$, $n_1 = (3,0)$, $n_2 = (1,1)$, $n_3 = (0,1)$, and $p = 3$ give the set shown below.



Generalization to Dimensions $\geq$ 3: Similar.

SEMI-LINEAR SETS AND CFLs

CountSet(L): Let $\Sigma = \{a_1, a_2, \cdots, a_n\}$, the alphabet of L.

- CountVector(x) = (#(a_1, x), #(a_2, x), $\cdots$, #(a_n, x)), for $x \in L$.
- CountSet(L) = {CountVector(x): $x \in L$}.

Example. Each of the following is a semi-linear set.

- For $L = L_{a^n b^n}$, CountSet(L) = {(1,1), (2,2), (3,3), $\cdots$}.
- For $L = L_{bal}$, CountSet(L) = {(1,1), (2,2), (3,3), $\cdots$}.
- For $L = L_{\#a=\#b}$, CountSet(L) = {(1,1), (2,2), (3,3), $\cdots$}.
- For $L = L_{a^{n+1} b^n}$, CountSet(L) = {(2,1), (3,2), (4,3), $\cdots$}.

Parikh's Mapping:

- $x \to$ CountVector(x), a many-to-one mapping from strings to non-negative integer-vectors.
- $L \to$ CountSet(L), a many-to-one mapping from languages to sets of non-negative integer-vectors.

Theorem (Parikh, 1966):

- For each CFL L, CountSet(L) is a finite union of semi-linear sets.

Question:

•? Why do we need "union" in the above theorem?

•? If L_1 and L_2 are two languages with the same alphabet and both CountSet(L_1) and CountSet(L_2) are semi-linear, then is CountSet($L_1 L_2$) also semi-linear? How about CountSet($L_1 \cup L_2$) and CountSet(L_1^*)? How about CountSet(L) if L is a finite language?