CFL Pumping Lemma

Friday, March 29, 2024
Announcements

• HW 6
  • Due Monday 4/1 12pm noon
Pumping Lemma for CFLS

Pumping lemma for context-free languages

If $A$ is a context-free language, then there is a number $p$ (the pumping length) where, if $s$ is any string in $A$ of length at least $p$, then $s$ may be divided into five pieces $s = uvxyz$ satisfying the conditions:

1. for each $i \geq 0$, $uv^i xy^i z \in A$,
2. $|vy| > 0$, and
3. $|vxy| \leq p$.

Pumping lemma

If $A$ is a regular language, then there is a number $p$ (the pumping length) where if $s$ is any string in $A$ of length at least $p$, then $s$ may be divided into three pieces, $s = xyz$, satisfying the conditions:

1. for each $i \geq 0$, $xy^i z \in A$,
2. $|y| > 0$, and
3. $|xy| \leq p$.

Two pumpable parts.
But they must be pumped together!

Two pumpable parts, pumped together

One pumpable part
A Non CFL example

\[ \text{language } B = \{ a^n b^n c^n \mid n \geq 0 \} \text{ is not context free} \]

Intuition

• Strings in CFLs can have **two parts** that are “pumped” together
• Language \( B \) requires **three parts** to be “pumped” together
• So it’s not a CFL!

Proof?
Want to prove: \(a^n b^n c^n\) is not a CFL

Proof (by contradiction):

- **Assume**: \(a^n b^n c^n\) is a CFL
  - So it must satisfy the pumping lemma for CFLs
  - I.e., all strings \(\geq\) length \(p\) are pumpable
- **Counterexample** = \(a^p b^p c^p\)

Now we must find a contradiction ...

Contradiction if:
- A string in the language \(\checkmark\)
- \(\geq\) length \(p\) \(\checkmark\)
- Is not splittable into \(uvxyz\) where \(v\) and \(y\) are pumpable

Reminder: CFL Pumping lemma says: all strings \(a^n b^n c^n \geq\) length \(p\) are splittable into \(uvxyz\) where \(v\) and \(y\) are pumpable
Possible Splits

Proof (by contradiction):

- **Assume**: $a^n b^n c^n$ is a CFL
  - So it must satisfy the pumping lemma for CFLs
  - i.e., all strings $\geq$ length $p$ are pumppable
- **Counterexample** = $a^p b^p c^p$

- **Possible Splits** (using condition #3: $|vxy| \leq p$)
  - $vxy$ is all $a$s
  - $vxy$ is all $b$s
  - $vxy$ is all $c$s
  - $vxy$ has $a$s and $b$s
  - $vxy$ has $b$s and $c$s
  - (Because of condition #3, $vxy$ cannot have $a$s, $b$s, and $c$s)

So $a^n b^n c^n$ is not a CFL

**Contradiction if**:
- A string in the language
  - $\geq$ length $p$
  - Is not splittable into $uvxyz$ where $u$ and $y$ are pumppable

**$a^p b^p c^p$ cannot be split** into $uvxyz$ where $v$ and $y$ are pumppable!
Another Non-CFL \( D = \{ww \mid w \in \{0,1\}^*\} \)

Be careful when choosing counterexample \( s: 0^p 10^p 1 \)
This \( s \) can be pumped according to CFL pumping lemma:

\[
\begin{array}{c}
\underbrace{000 \ldots 000} & 0 & 1 & \underbrace{000 \ldots 0001} \\
\text{u} & \text{v} & \text{x} & \text{y} & \text{z}
\end{array}
\]

Pumping \( v \) and \( y \) (together) produces string still in \( D \) ...

\[
...\text{just like pumping lemma says (no contradiction)!}
\]

\[ \checkmark 1. \text{for each } i \geq 0, uv^i xy^i z \in A, \]
\[ \checkmark 2. |vy| > 0, \text{ and} \]
\[ \checkmark 3. |vxy| \leq p. \]

So this attempt to prove that the language is not a CFL failed.
(It doesn’t prove that the language is a CFL!)
Another Non-CFL \( D = \{ww \mid w \in \{0,1\}^*\} \)

- Need another counterexample string \( s \):
  - If \( vyx \) is contained in first or second half, then any pumping will break the match.
  - \( 0^p1^p0^p1^p \)
  - e.g., \( 0^p1^{p-1}100^{p-1}1^p \)
  - So \( vyx \) must straddle the middle
  - But any pumping still breaks the match because order is wrong

- CFL Pumping Lemma conditions:
  1. for each \( i \geq 0 \), \( uv^i xy^i z \in A \),
  2. \( |vy| > 0 \), and
  3. \( |vxy| \leq p \).

Now we have proven that this language is not a CFL!
A Practical Non-CFL

• **XML**
  - ELEMENT → <TAG>CONTENT</TAG>
  - Where TAG is any string

• XML also looks like this **non-CFL**: \( D = \{ww | w \in \{0,1\}^*\} \)

• This means XML is **not context-free**!
  - **Note**: HTML is context-free because ...
  - ... there are only a **finite** number of tags,
  - so they can be embedded into a **finite** number of rules.

**In practice:**
• XML is **parsed** as a CFL, with a CFG
• Then matching tags checked in a 2\(^{nd}\) pass with a more powerful machine...
Next: A More Powerful Machine ...

\[ M_1 \] accepts its input if it is in language: \[ B = \{ w\#w | w \in \{0,1\}^* \} \]

\[ M_1 = \text{“On input string } w: \]

1. Zig-zag across the tape to corresponding positions on either side of the \# symbol to check whether these positions contain the same symbol. If they do not, or if no \# is found, reject. Cross off symbols as they are checked to keep track of which symbols correspond.

Infinite memory (initial contents are the input string)

Can move to, and read/write from arbitrary memory locations!