UMB CS 420
Non-CFLs
Wednesday, March 22, 2023
Announcements

• HW 6
  • Due Sunday 3/26 11:pm EDT

Quiz Preview

• The **Pumping Lemma for CFLs** states that:
  • all strings in a CFL that are longer than the pumping length can be split into 5 substrings \( uvxyz \) ...
  • ... where repeating some of these substrings (together) results in a "pumped" string that is still in the language.
  • Which are the substrings that can be pumped (together) in this way?
Flashback: Pumping Lemma for Regular Langs

- **Pumping Lemma** describes how strings repeat

- Regular language strings repeat using **Kleene star** operation
  - 3 substrings \(xyz\) are independent!

- A non-regular language:
  \[
  \{0^n1^n | n \geq 0\}
  \]

  Kleene star can’t express this pattern: 2nd part depends on (length of) 1st part

- \(Q\): How do CFLs repeat?
Repetition and Dependency in CFLs

Parts before/after repetition point are linked

\[ A \rightarrow 0A1 \]
\[ A \rightarrow B \]
\[ B \rightarrow \# \]

\[ \{0^n\#1^n | n \geq 0\} \]

Repetition

\[ A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111 \]
How Do Strings in CFLs Repeat?

• Strings in regular languages repeat states

• Strings in CFLs repeat subtrees in the parse tree

NFA can take loop transition any number of times, to process repeated $y$ in input

One repeated subtree means that it can be repeated any number of times

5 substrings

Linked parts

Linked parts repeat together
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 \( i \geq 0 \), \( uv^ixy^iz \in A \),
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 \( i \geq 0 \), \( xy^iz \in A \),
2. \( |y| > 0 \), and
3. \( |xy| \leq p \).

Two pumpable parts.
But they must be pumped together!
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
- This language 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: string \( \geq \) length \( p \) that is **not** splittable into \( uvxyz \) where \( v \) and \( y \) are pumpable

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 \( i \geq 0, \ uv^ixy^i z \in A \)
2. \( |vy| > 0 \), and
3. \( |vxy| \leq p \).

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

---

\[ p \ as \quad p \ bs \quad p \ bs \]
\[ a \ ... \ b \ ... \ c \ ... \]
Want to prove: $a^n b^n c^n$ is not a CFL

**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** $= a^p b^p c^p$
  - Contradiction if: string $\geq$ length $p$ that is not splittable into $uvxyz$ where $v$ and $y$ are pumppable

- **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

So $a^n b^n c^n$ is not a CFL (justification: contrapositive of CFL pumping lemma)
Another Non-CFL $D = \{ww | w \in \{0,1\}^*\}$

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

\[
\begin{align*}
0^p 1 \\
\{000 \ldots 000\} & \ {0} & \ {1} & \ {000 \ldots 0001} \\
\text{u} & \ {v} & \ {x} & \ {y} & \ {z}
\end{align*}
\]

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

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

- Need another counterexample string \( s \):
  - If \( vyx \) is contained in first or second half, then any pumping will break the match.
  - \( 0^p 1^p 0^p 1^p \)
  - 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 $\rightarrow$ `<TAG>CONTENT</TAG>`
  - Where TAG is any string

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

• **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 2nd pass with a more powerful machine...
Next: A More Powerful Machine ...

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

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

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!
In-class quiz 3/22

See gradescope