<table>
<thead>
<tr>
<th>Language Class</th>
<th>Generation</th>
<th>Recognition</th>
<th>Properties</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>reg. expr.</td>
<td>DFA/NFA</td>
<td>p.l. closure properties decision algo</td>
<td>can't handle a*b/c^k</td>
</tr>
<tr>
<td>Context Free</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Set of rules

1. \( S \rightarrow a \ b \)
2. \( S \rightarrow a S b \)
3. \( S \rightarrow \epsilon \)
   \( S \rightarrow a S b \rightarrow a a S b b \rightarrow a a a S b b b \rightarrow a a a a b b b b \)

apply repeatedly one of the rules

Any sequence of substitution must begin with the special variable \( S \)

\( G : \{ V = S, S = S \} \)
\( T = \{ a, b, \epsilon \} \)
\( P = 1, 2, 3 \) 

Convention: Capital letters for variables, small case for terminals

\( AB \rightarrow ABB \)

Context Free Grammar (CFG), the production rules have exactly one symbol on the LHS
Grammar \( G = (V, T, P, S) \)
- Set of variables that appear in RHS
- Set of variables
- Set of rules / productions
- Set of alphabet / terminals

Let: Equal number of a's and b's
\( a^i b^i \), \( aabab, aababa \)

Is \( L_eq \) regular?

Context Free Language (CFL): all languages that can be generated using CFG

Is \( L_eq \) CFL?

\( S, \{a, b, \epsilon\}, S, S \rightarrow \epsilon | ba | ab | la | bl | ba S | ab S \)
The given text contains a grammar for a formal language, along with a claim and its proof by induction.

**Grammar:**

\[ V = \{ \epsilon, A, B \} \quad T = \{ a, b \} \]

- \( S \rightarrow aB \mid bA \mid \epsilon \)
- \( A \rightarrow a \mid bAA \mid aS \)
- \( B \rightarrow b \mid aBB \mid bS \)

**Claim 1:**

\( S \rightarrow \omega \) iff \( \omega \) has equal number of a's and b's for \( |\omega| \geq 1 \).

**Claim 2:**

- \( A \rightarrow \omega \) iff \( \omega \) has one more a than b.
- \( B \rightarrow \omega \) iff \( \omega \) has one more b than a.

**Proof by Induction on \( |\omega| \):**

- **Base Case:** \( |\omega| = 1 \)
  - \( S \): no strings of length 1, so true.
  - \( A \): \( A \rightarrow a \) only string of length 1.
  - \( B \): \( B \rightarrow b \)

- **Induction Hypothesis:** For all \( |\omega| \leq k-1 \)

Consider any string \( |\omega| = k \)

- \( S \rightarrow a \omega \rightarrow b \omega \), If \( \omega \) has equal number of a's and b's then \( S \rightarrow \omega \)

- \( S \rightarrow aB \rightarrow a \omega \) if the B generates all strings \( \leq \) length \( k-1 \) with one extra b.
So $B \xRightarrow{} w_i$

Conversely if $S \xRightarrow{} w$ then $w = a w_i \overset{\text{extra } b}{\underset{\text{extra } a}{\Rightarrow}}$

Let $S \rightarrow aB$ So $B \xRightarrow{} w_i$

from I.H. $w_i$ has an extra $b$

Prove it for all three assert $A, B, S$ and their converse

$$A \rightarrow a \mid b A A \mid a S$$

If $A \xRightarrow{} w$ $|w| = k$ then $w$ has one more $a$ than $b$

Let $w = b w_i$, $A \rightarrow b A A \xRightarrow{} b w_i', A \overset{a}{\Rightarrow} w_i'$

Sub $A \rightarrow w_i', A \rightarrow w_i''$ $|w_i'| \leq k-1, |w_i''|$

From I.H. $w_i', w_i''$ will have $|w_i''|$

So overall one more $a$ than $b$

If $w$ has one more $a$ than $b$ then $A \xRightarrow{} w$

$|w| = k$, $w = a w_i$, $w_i \notin \Sigma^{*} a\Sigma^{*} b$ and $b$’s

$A \rightarrow a S \xRightarrow{} a w_i$, where $S \xRightarrow{} w_i$
\[ W = b W_1 \quad w_1 \text{ has } 2 \text{ extra } a's \text{ than } b's \]

\[ W_1 = \underbrace{x_1 x_2} \overbrace{x_3 \ldots x_{k-1}}^A A \]

Difference between \#a's and \#b's for each position of the string \( w \),

\[ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 2 \]

\[ baababa \]

**H.W. Problem.** Design a CFG for strings over \( a, b \) s.t.

\[ \#a's = 2 \times \#b's \]

**Different ways of writing CFG**

**Membership problem**

Given a CFG \( G = (V, T, S, P) \) and a string \( w \in T^* \), does \( S \rightarrow w \) Derivation True

```
S
 b
 \_ \_ \_
A \_ A
\_ \_ \_
```

\( \text{0 0 0 0 0 0} \) Terminals
Canonical form of CFG

Chomsky Normal Form

A → BC
A → a

Greibach Normal Form

A → a BCD
A → a

Claim: Any given CFG can be transformed into an equivalent CNF or GNF