Big picture

- All languages
- Decidable

  Turing machines
- NP
- P
- Context-free

  Context-free grammars, push-down automata
- Regular

  Automata, non-deterministic automata,

  regular expressions

Recall:

Theorem: $L := \{0^n 1^n : n \geq 0\}$ is not regular

But it is often needed to recognize this language

Example: Programming language syntax have matching brackets, not regular.

Next: Introduce context-free languages

Why study context-free languages

- Practice with more powerful model
- Programming languages: Syntax of C++, Java, etc. is specified by context-free grammar
- Other reasons: human language has structures that can be modeled as context-free language
  English is not a regular language

Example: Context-free grammar G, $\Sigma = \{0,1\}$

$S \to 0\ S\ 1$

$S \to \varepsilon$

Two substitution rules (a.k.a. productions) $\to$

Variables = {S}, Terminals = {0,1}

Derivation of 0011 in grammar:

$S \Rightarrow 0S1 \Rightarrow 00S11 \Rightarrow 0011$

$L(G) = \{0^n\ 1^n : n \geq 0\}$

Example: Context-free grammar G, $\Sigma = \{0,1\}$

$S \rightarrow A$

$S \rightarrow B$

$A \rightarrow 0 A 1$

$A \rightarrow \varepsilon$

$B \rightarrow 1 B 0$

$B \rightarrow \varepsilon$

$$L(G) = L(A) \cup L(B)$$
$$= \{0^n 1^n : n \geq 0\} \cup \{1^n 0^n : n \geq 0\}$$

Next: A convention to write this more compactly

Example: Context-free grammar G, $\Sigma = \{0,1\}$

$S \rightarrow A \mid B$

$A \rightarrow 0\ A\ 1 \mid \varepsilon$

$B \rightarrow 1\ B\ 0 \mid \varepsilon$

Convention: Write $A \rightarrow w|w'$ for

$A \rightarrow w$ and $A \rightarrow w'$

Definition: A context-free grammar (CFG) G is a 4 tuple (V, Σ, R, S) where

- V is a finite set of variables
- Σ is a finite set of terminals $(V \cap \Sigma = \emptyset)$
- R is a finite set of rules, where each rule is

  $A \rightarrow w$ $\qquad A \in V, w \in (V \cup \Sigma)^*$
- $S \in V$ is the start variable

Example

The language $L = \{a^m b^n : m > n\}$

is described by the CFG $G = (V, \Sigma, R, S)$

where:

$V = \{S, T\}$

$\Sigma = \{a, b\}$

$R = \{ S \rightarrow aS \mid aT$

$T \rightarrow aTb \mid \varepsilon \}$

Derive aaab:

$S \rightarrow$?

Example

The language $L = \{a^m b^n : m > n\}$

is described by the CFG $G = (V, \Sigma, R, S)$

where:

$V = \{S, T\}$

$\Sigma = \{a, b\}$

$R = \{ \ S \rightarrow aS \mid aT$

$T \rightarrow aTb \mid \varepsilon \ \}$

Derive aaab:

$S \rightarrow aS$

$\rightarrow$?

Example

The language $L = \{a^m b^n : m > n\}$

is described by the CFG $G = (V, \Sigma, R, S)$

where:

$V = \{S, T\}$

$\Sigma = \{a, b\}$

$R = \{\ S \rightarrow aS \mid aT$

$T \rightarrow aTb \mid \varepsilon\ \}$

Derive aaab:

$S \rightarrow aS$

$\rightarrow aaT$

$\rightarrow$?

Example

The language $L = \{a^m b^n : m > n\}$

is described by the CFG $G = (V, \Sigma, R, S)$

where:

$V = \{S, T\}$

$\Sigma = \{a, b\}$

$R = \{ S \rightarrow aS \mid aT$

$T \rightarrow aTb \mid \varepsilon \}$

Derive aaab:

$S \rightarrow aS$

$\rightarrow aaT$

$\rightarrow aaaTb$

$\rightarrow$?

Example

The language $L = \{a^m b^n : m > n\}$

is described by the CFG $G = (V, \Sigma, R, S)$

where:

$V = \{S, T\}$

$\Sigma = \{a, b\}$

$R = \{\ S \rightarrow aS \mid aT$

$T \rightarrow aTb \mid \varepsilon\ \}$

Derive aaab:

$S \rightarrow aS$

$\rightarrow aaT$

$\rightarrow aaaTb$

$\rightarrow aaab$

Definition: Let $G = (V, \Sigma, R, S)$ be a CFG

we write $uAv \Rightarrow uwv$ and say uAv yields uwv

if $A \rightarrow w$ is a rule

We say u derives v, written $u \Rightarrow^* v$, if

- $u = v$, or
- $\exists\ u_1, u_2, \ldots, u_k \quad k \geq 1$:

$u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow \ldots \Rightarrow u_k = v$

The language of the grammar is $L(G) = \{w : S \Rightarrow^* w\}$

Definition: A language L is context-free if

$\exists$ CFG G : L(G) = L

Example:

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : \ x,y \text{ in } \{0,1\}^* \ \ |x| \neq |y| \}$

$$
\begin{aligned}
G = \quad & S \rightarrow BL \\
& S \rightarrow RB \\
& L \rightarrow BL \mid A \\
& R \rightarrow RB \mid A \\
& A \rightarrow BAB \mid \# \\
& B \rightarrow 0 \mid 1
\end{aligned}
$$

Remark: $B \Rightarrow^* ?$

To understand, explain what each piece does!

Example:

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : \ x,y \text{ in } \{0,1\}^* \ |x| \neq |y| \}$

$G =$ $S \rightarrow BL$

$S \rightarrow RB$

$L \rightarrow BL \mid A$

$R \rightarrow RB \mid A$

$A \rightarrow BAB \mid \#$ Remark: $A \Rightarrow^* ?$

$B \rightarrow 0 \mid 1$ Remark: $B \Rightarrow^* 0$, $B \Rightarrow^* 1$

Example:

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : \; x,y \text{ in } \{0,1\}^* \; |x| \neq |y| \}$

G = $S \rightarrow BL$

$S \rightarrow RB$

$L \rightarrow BL \mid A$

$R \rightarrow RB \mid A$ Remark: $R \Rightarrow^* ?$

$A \rightarrow BAB \mid \#$ Remark: $A \Rightarrow^* x\#y : |x|=|y|$

$B \rightarrow 0 \mid 1$ Remark: $B \Rightarrow^* 0, \; B \Rightarrow^* 1$

Example:

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : \ x,y \text{ in } \{0,1\}^* \ \ |x| \neq |y| \}$

$G = \quad S \rightarrow BL$

$S \rightarrow RB$

$L \rightarrow BL \mid A$ Remark: $L \Rightarrow^* ?$

$R \rightarrow RB \mid A$ Remark: $R \Rightarrow^* x\#y : |x| \leq |y|$

$A \rightarrow BAB \mid \#$ Remark: $A \Rightarrow^* x\#y : |x|=|y|$

$B \rightarrow 0 \mid 1$ Remark: $B \Rightarrow^* 0, \ B \Rightarrow^* 1$

Example:

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : \ x,y \text{ in } \{0,1\}^* \ \ |x| \neq |y| \}$

G = $S \to BL$

$S \to RB$ Remark: $RB \Rightarrow^* ?$

$L \to BL \mid A$ Remark: $L \Rightarrow^* x\#y : \ |x| \geq |y|$

$R \to RB \mid A$ Remark: $R \Rightarrow^* x\#y : |x| \leq |y|$

$A \to BAB \mid \#$ Remark: $A \Rightarrow^* x\#y : |x|=|y|$

$B \to 0 \mid 1$ Remark: $B \Rightarrow^* 0, \ B \Rightarrow^* 1$

Example:

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : x,y \text{ in } \{0,1\}^* \ |x| \neq |y| \}$

G =	$S \rightarrow BL$	Remark: $BL \Rightarrow^*$?
	$S \rightarrow RB$	Remark: $RB \Rightarrow^* x\#y : |x| < |y|$
	$L \rightarrow BL \mid A$	Remark: $L \Rightarrow^* x\#y : |x| \geq |y|$
	$R \rightarrow RB \mid A$	Remark: $R \Rightarrow^* x\#y : |x| \leq |y|$
	$A \rightarrow BAB \mid \#$	Remark: $A \Rightarrow^* x\#y : |x|=|y|$
	$B \rightarrow 0 \mid 1$	Remark: $B \Rightarrow^* 0,\ B \Rightarrow^* 1$

Example:

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : \; x,y \text{ in } \{0,1\}^* \; |x| \neq |y| \}$

G =	$S \rightarrow BL$	Remark: $BL \Rightarrow^* x\#y : \; |x| > |y|$
	$S \rightarrow RB$	Remark: $RB \Rightarrow^* x\#y : \; |x| < |y|$
	$L \rightarrow BL \mid A$	Remark: $L \Rightarrow^* x\#y : \; |x| \geq |y|$
	$R \rightarrow RB \mid A$	Remark: $R \Rightarrow^* x\#y : |x| \leq |y|$
	$A \rightarrow BAB \mid \#$	Remark: $A \Rightarrow^* x\#y : |x|=|y|$
	$B \rightarrow 0 \mid 1$	Remark: $B \Rightarrow^* 0, \; B \Rightarrow^* 1$

$L(G) = L$

Example: CFG for expressions in programming languages

Task: recognize strings like 0 + 0 + 1 x (1 + 0)

$S \rightarrow S+S \mid S \text{ x } S \mid (S) \mid 0 \mid 1$

$S \rightarrow S + S \rightarrow 0 + S \rightarrow 0 + S + S \rightarrow 0 + 0 + S$

$\rightarrow 0 + 0 + S \text{ x } S \rightarrow 0 + 0 + 1 \text{ x } S$

$\rightarrow 0 + 0 + 1 \text{ x } (S) \rightarrow 0 + 0 + 1 \text{ x } (S + S)$

$\rightarrow 0 + 0 + 1 \text{ x } (1 + S) \rightarrow 0 + 0 + 1 \text{ x } (1 + 0)$

We have seen: CFG, definition, and examples

Next: Ambiguity

- Ambiguity: Some string may have multiple derivations in a CFG

- Ambiguity is a problem for compilers:

Compilers use derivation to give meaning to strings.

Example: meaning of 1+0x0 $\in \Sigma^*$ is its value, 1 $\in \mathbb{N}$

If there are two different derivations,

the value may not be well defined.

Example: The string 1+0x0 has two derivations in

$S \rightarrow S+S \mid S \times S \mid (S) \mid 0 \mid 1$

<u>One derivation:</u>

$S \rightarrow S+S \rightarrow 1+S \rightarrow 1+S\times S \rightarrow 1+0\times S \rightarrow 1+0\times 0$

<u>Another derivation:</u>

$S \rightarrow$?

Example: The string 1+0x0 has two derivations in

$S \rightarrow S+S \mid S \times S \mid (S) \mid 0 \mid 1$

One derivation:

$S \rightarrow S+S \rightarrow 1+S \rightarrow 1+SxS \rightarrow 1+0xS \rightarrow 1+0x0$

Another derivation:

$S \rightarrow SxS \rightarrow$?

Example: The string 1+0x0 has two derivations in

S → S+S | S x S | (S) | 0 | 1

One derivation:

S → S+S → 1+S → 1+SxS → 1+0xS → 1+0x0

Another derivation:

S → SxS → Sx0 → ?

Example: The string 1+0x0 has two derivations in

$S \rightarrow S+S \mid S \times S \mid (S) \mid 0 \mid 1$

<u>One derivation:</u>

$S \rightarrow S+S \rightarrow 1+S \rightarrow 1+SxS \rightarrow 1+0xS \rightarrow 1+0x0$

<u>Another derivation:</u>

$S \rightarrow SxS \rightarrow Sx0 \rightarrow S+Sx0 \rightarrow$?

Example: The string 1+0x0 has two derivations in

$S \rightarrow S+S \mid S \text{ x } S \mid (S) \mid 0 \mid 1$

<u>One derivation:</u>

$S \rightarrow S+S \rightarrow 1+S \rightarrow 1+SxS \rightarrow 1+0xS \rightarrow 1+0x0$

<u>Another derivation:</u>

$S \rightarrow SxS \rightarrow Sx0 \rightarrow S+Sx0 \rightarrow S+0x0 \rightarrow$?

Example: The string 1+0x0 has two derivations in

S → S+S | S x S | (S) | 0 | 1

One derivation:

S → S+S → 1+S → 1+SxS → 1+0xS → 1+0x0

Another derivation:

S → SxS → Sx0 → S+Sx0 → S+0x0 → 1+0x0

We now want to define CFG with no ambiguity

Definition: A derivation is leftmost if at every step the leftmost variable is expanded

Example: the 1st previous derivation was leftmost

$$S \rightarrow S+S \rightarrow 1+S \rightarrow 1+SxS \rightarrow 1+0xS \rightarrow 1+0x0$$

Definition: A CFG G is un-ambiguous if no string has two different leftmost derivations.

Example

The CFG $S \rightarrow S+S \mid S \times S \mid (S) \mid 0 \mid 1$

is ambiguous because 1+0x0 has two distinct leftmost derivations

<u>One leftmost derivation:</u>

$S \rightarrow S+S \rightarrow 1+S \rightarrow 1+SxS \rightarrow 1+0xS \rightarrow 1+0x0$

<u>Another leftmost derivation:</u>

$S \rightarrow SxS \rightarrow S+SxS \rightarrow 1+SxS \rightarrow 1+0xS \rightarrow 1+0x0$

Example Instead of using CFG

$S \rightarrow S+S \mid S \times S \mid (S) \mid 0 \mid 1$

we may use un-ambiguous grammar

$S \rightarrow S + T \mid T$

$T \rightarrow T \times F \mid F$

$F \rightarrow 0 \mid 1 \mid (S)$

Unique leftmost derivation of 1+0x0:

$S \rightarrow$?

Example Instead of using CFG

$S \to S+S \mid S \times S \mid (\ S\ ) \mid 0 \mid 1$

we may use un-ambiguous grammar

$S \to S + T \mid T$

$T \to T \times F \mid F$

$F \to 0 \mid 1 \mid (S)$

<u>Unique leftmost derivation of 1+0x0:</u>

$S \to S + T \to$?

Example Instead of using CFG

$S \to S{+}S \mid S \times S \mid (S) \mid 0 \mid 1$

we may use un-ambiguous grammar

$S \to S + T \mid T$

$T \to T \times F \mid F$

$F \to 0 \mid 1 \mid (S)$

<u>Unique leftmost derivation of 1+0x0:</u>

$S \to S + T \to T + T \to$?

Example Instead of using CFG

$S \to S+S \mid S \times S \mid (S) \mid 0 \mid 1$

we may use un-ambiguous grammar

$S \to S + T \mid T$

$T \to T \times F \mid F$

$F \to 0 \mid 1 \mid (S)$

Unique leftmost derivation of 1+0x0:

$S \to S + T \to T + T \to F + T \to$?

Example Instead of using CFG

$S \to S+S \mid S \times S \mid (S) \mid 0 \mid 1$

we may use un-ambiguous grammar

$S \to S + T \mid T$

$T \to T \times F \mid F$

$F \to 0 \mid 1 \mid (S)$

<u>Unique leftmost derivation of 1+0x0:</u>

$S \to S + T \to T + T \to F + T \to 1 + T$

$\to$?

Example Instead of using CFG

$S \to S+S \mid S \times S \mid (S) \mid 0 \mid 1$

we may use un-ambiguous grammar

$S \to S + T \mid T$

$T \to T \times F \mid F$

$F \to 0 \mid 1 \mid (S)$

<u>Unique leftmost derivation of 1+0x0:</u>

$S \to S + T \to T + T \to F + T \to 1 + T$

$\to 1 + T \times F \to$?

Example Instead of using CFG

$S \to S+S \mid S \times S \mid (S) \mid 0 \mid 1$

we may use un-ambiguous grammar

$S \to S + T \mid T$

$T \to T \times F \mid F$

$F \to 0 \mid 1 \mid (S)$

Unique leftmost derivation of 1+0x0:

$S \to S + T \to T + T \to F + T \to 1 + T$

$\to 1 + T \times F \to 1 + 0 \times F \to 1 + 0 \times 0$

Actual Java specification grammar snippet

Cumbersome but un-ambiguous

MultiplicativeExpression:

UnaryExpression

*MultiplicativeExpression * UnaryExpression*

MultiplicativeExpression / UnaryExpression

MultiplicativeExpression % UnaryExpression

AdditiveExpression:

MultiplicativeExpression

AdditiveExpression + MultiplicativeExpression

AdditiveExpression - MultiplicativeExpression

Next: understand power of context-free languages

Study closure under not, U, o, *

Recall from regular langues: If A, B are regular then

not A is regular ?

A U B is regular ?

A o B is regular ?

A* is regular ?

Next: understand power of context-free languages

Study closure under not, U, o, *

Recall from regular langues: If A, B are regular then

not A regular

A U B regular

A o B regular

A* regular

Suppose A, B are context-free:

$A = L(G_A)$ for CFG $G_A=(V_A, \Sigma, R_A, S_A)$

$B = L(G_B)$ for CFG $G_B=(V_B, \Sigma, R_B, S_B)$

What about

A U B $\quad S \to$?

A o B

A*

Suppose A, B are context-free:

$A = L(G_A)$ for CFG $G_A=(V_A, \Sigma, R_A, S_A)$

$B = L(G_B)$ for CFG $G_B=(V_B, \Sigma, R_B, S_B)$

What about

A U B $\quad S \rightarrow S_A|S_B$ $\quad$ Context-free

A o B $\quad S \rightarrow$?

A*

Suppose A, B are context-free:

$A = L(G_A)$ for CFG $G_A=(V_A, \Sigma, R_A, S_A)$

$B = L(G_B)$ for CFG $G_B=(V_B, \Sigma, R_B, S_B)$

What about

A U B	$S \rightarrow S_A \mid S_B$	Context-free
A o B	$S \rightarrow S_A\, S_B$	Context-free
A*	$S \rightarrow$?	

Suppose A, B are context-free:

$A = L(G_A)$ for CFG $G_A=(V_A, \Sigma, R_A, S_A)$

$B = L(G_B)$ for CFG $G_B=(V_B, \Sigma, R_B, S_B)$

What about

A U B	$S \rightarrow S_A \mid S_B$	Context-free
A o B	$S \rightarrow S_A S_B$	Context-free
A*	$S \rightarrow S S_A \mid \varepsilon$	Context-free

Above all context-free!

In general, (not A) is NOT context-free

Suppose A, B are context-free:

$A = L(G_A)$ for CFG $G_A=(V_A, \Sigma, R_A, S_A)$

$B = L(G_B)$ for CFG $G_B=(V_B, \Sigma, R_B, S_B)$

What about

A U B	$S \rightarrow S_A \mid S_B$	Context-free
A o B	$S \rightarrow S_A S_B$	Context-free
A*	$S \rightarrow S S_A \mid \varepsilon$	Context-free

Above also shows regular ⇨ context-free

Context-free languages contain regular languages

Example: Context Free UNION

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : x,y \text{ in } \{0,1\}^* \; |x| \neq |y| \text{ OR } x = y^R \}$

y^R is the reverse of y:

$001^R = 100$

$11010^R = 01011$

$1^R = 1$

Example: Context Free UNION

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : x,y \text{ in } \{0,1\}^*$
$|x| \neq |y| \text{ OR } x = y^R \}$

Write $L = L_1 \cup L_2$, where

$L_1 = \{ x\#y : |x| \neq |y| \}$ $\quad$ $L_2 = \{ x\#y : x = y^R \}$

Example: Context Free UNION

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : \; x,y \text{ in } \{0,1\}^*$
$|x| \neq |y| \text{ OR } x = y^R \}$

Write $L = L_1 \cup L_2$, where

$L_1 = \{ x\#y : |x| \neq |y| \}$ $\quad$ $L_2 = \{ x\#y : x = y^R \}$

$G_1 =$ $S_1 \rightarrow BL \mid RB$

$L \rightarrow BL \mid A$ $\quad$ Remark: $L \Rightarrow^* x\#y : |x| \geq |y|$

$R \rightarrow RB \mid A$ $\quad$ Remark: $R \Rightarrow^* x\#y : |x| \leq |y|$

$A \rightarrow BAB \mid \#$ $\quad$ Remark: $A \Rightarrow^* x\#y : |x|=|y|$

$B \rightarrow 0 \mid 1$

Example: Context Free UNION

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : x,y \text{ in } \{0,1\}^*$
$|x| \neq |y| \text{ OR } x = y^R \}$

Write $L = L_1 \cup L_2$, where

$L_1 = \{ x\#y : |x| \neq |y| \}$

$L_2 = \{ x\#y : x = y^R \}$

$G_1 =$
$S_1 \rightarrow BL \mid RB$
$L \rightarrow BL \mid A$
$R \rightarrow RB \mid A$
$A \rightarrow BAB \mid \#$
$B \rightarrow 0 \mid 1$

$G_2 = S_2 \rightarrow 0S_20 \mid 1S_21 \mid \#$

Example: Context Free UNION

$\Sigma = \{0,1,\#\}$

Give a CFG for $L = \{ x\#y : x,y \text{ in } \{0,1\}^*$
$|x| \neq |y|$ OR $x = y^R \}$

Write $L = L_1 \cup L_2$, where

$L_1 = \{ x\#y : |x| \neq |y| \}$

$L_2 = \{ x\#y : x = y^R \}$

$G_1 = \quad S_1 \to BL \mid RB$

$L \to BL \mid A$

$R \to RB \mid A$

$A \to BAB \mid \#$

$B \to 0 \mid 1$

$G_2 = \quad S_2 \to 0S_20 \mid 1S_21 \mid \#$

Let $G = S \to S_1 \mid S_2$

Then, $L(G_1) = L_1$ & $L(G_2) = L_2$

$\Rightarrow L(G) = L_1 \cup L_2 = L$

Example: Context Free CONCATENATION

Give a CFG for $L = \{ 0^m 1^m 0^n 1^n : m \text{ even and } n \text{ odd} \}$

Example: Context Free CONCATENATION

Give a CFG for $L = \{ 0^m 1^m 0^n 1^n : m \text{ even and } n \text{ odd}\}$

Write $L = L_1 \circ L_2$, where

$L_1 = \{ 0^m 1^m : m \text{ even}\}$

$L_2 = \{ 0^n 1^n : n \text{ odd}\}$

Example: Context Free CONCATENATION

Give a CFG for $L = \{ 0^m 1^m 0^n 1^n : m \text{ even and } n \text{ odd}\}$

Write $L = L_1 \circ L_2$, where

$L_1 = \{ 0^m 1^m : m \text{ even}\}$ $L_2 = \{ 0^n 1^n : n \text{ odd}\}$

$G_1 = S_1 \rightarrow 00S_1 11 \mid \varepsilon$

Example: Context Free CONCATENATION

Give a CFG for $L = \{ 0^m 1^m 0^n 1^n : m \text{ even and } n \text{ odd} \}$

Write $L = L_1 \circ L_2$, where

$L_1 = \{ 0^m 1^m : m \text{ even} \}$ $L_2 = \{ 0^n 1^n : n \text{ odd} \}$

$G_1 = S_1 \rightarrow 00S_1 11 \mid \varepsilon$ $G_2 = S_2 \rightarrow 00S_2 11 \mid 01$

Example: Context Free CONCATENATION

Give a CFG for $L = \{ 0^m 1^m 0^n 1^n : m \text{ even and } n \text{ odd}\}$

Write $L = L_1 \circ L_2$, where

$L_1 = \{ 0^m 1^m : m \text{ even}\}$ $L_2 = \{ 0^n 1^n : n \text{ odd}\}$

$G_1 = S_1 \to 00S_1 11 \mid \varepsilon$ $G_2 = S_2 \to 00S_2 11 \mid 01$

Let $G = S \to S_1 S_2$

Then, $L(G_1) = L_1$ & $L(G_2) = L_2$

$\Rightarrow L(G) = L_1 \circ L_2 = L$

Example: Context Free STAR

Give a CFG for

$L = \{ w \text{ in } \{0,1\}^* : w = w_1 w_2 \cdots w_k , k \geq 0$

where each w_i is a palindrome $\}$

- A string w is a palindrome if $w = w^R$

  That is, w reads the same forwards and backwards

- Example: 00100, 1001, and 0 are palindromes;

  0011, 01 are not

Example: Context Free STAR

Give a CFG for

$L = \{ w \text{ in } \{0,1\}^* : w = w_1 w_2 \cdots w_k , k \geq 0$

where each w_i is a palindrome $\}$

Write $L = L_1^*$, where $L_1 = \{w : w \text{ is a palindrome}\}$

Note: In fact, $L = \{0,1\}^*$, but we will not use that.

Example: Context Free STAR

Give a CFG for

$L = \{ w \text{ in } \{0,1\}^* : w = w_1 w_2 \cdots w_k , k \geq 0$

where each w_i is a palindrome $\}$

Write $L = L_1^*$, where $L_1 = \{w : w \text{ is a palindrome}\}$

$G_1 = S_1 \rightarrow 0S_10 \mid 1S_11 \mid 0 \mid 1 \mid \varepsilon$

Example: Context Free STAR

Give a CFG for

$L = \{ w \text{ in } \{0,1\}^* : \ w = w_1 w_2 \cdots w_k , k \geq 0$

where each w_i is a palindrome $\}$

Write $L = L_1^*$, where $L_1 = \{w : w \text{ is a palindrome}\}$

$G_1 = \ S_1 \to 0S_10 \ | \ 1S_11 \ | \ 0 \ | \ 1 \ | \ \varepsilon$

Let $G = S \to SS_1 \mid \varepsilon$. Then, $L(G_1) = L_1$

$\Rightarrow L(G) = L_1^* = L$.

Beyond regular

A string and its reversal with C in middle:

$S \rightarrow 0S0 \mid 1S1 \mid C$

Example: $S \Rightarrow^* 0001C1000$

More generally, to get strings of the form $A^k\ C\ B^k$

use rules: $S \rightarrow A\ S\ B \mid C$

Example: $\Sigma = \{0,1,\#\}$, w^R is reverse of w

L = {w # x : w^R is a substring of x}

Useful to rewrite L as:

Example: $\Sigma = \{0,1,\#\}$, w^R is reverse of w

$L = \{w \# x : w^R$ is a substring of $x\}$

$= \{ w\#x\, w^R\, y : w,x,y \in \{0,1\}^* \}$

G :=

$S \rightarrow CB$

$C \rightarrow 0C0 \mid 1C1 \mid \#B$

$B \rightarrow 0\,B \mid 1\,B \mid \varepsilon$ Remark: $B \Rightarrow^*$?

Example: $\Sigma = \{0,1,\#\}$, w^R is reverse of w

$L = \{w \# x : w^R \text{ is a substring of } x\}$

$= \{ w\#x\, w^R\, y : w,x,y \in \{0,1\}^* \}$

G :=

$S \rightarrow CB$

$C \rightarrow 0C0 \mid 1C1 \mid \#B$ Remark: $C \Rightarrow^*$?

$B \rightarrow 0\,B \mid 1\,B \mid \varepsilon$ Remark: $B \Rightarrow^* \{0,1\}^*$

Example: $\Sigma = \{0,1,\#\}$, w^R is reverse of w

$L = \{w \# x : w^R$ is a substring of $x\}$

$= \{ w\#x\, w^R\, y : w,x,y \in \{0,1\}^* \}$

G :=

$S \rightarrow CB$

$C \rightarrow 0C0 \mid 1C1 \mid \#B$ Remark: $C \Rightarrow^* w\#\{0,1\}^* w^R$

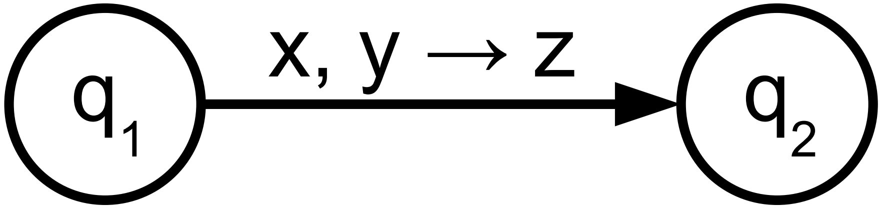
$B \rightarrow 0\,B \mid 1\,B \mid \varepsilon$ Remark: $B \Rightarrow^* \{0,1\}^*$

$L(G) = L$

CFG vs. automata

CFG ⇔ non-deterministic pushdown automata (PDA)

A PDA is simply an NFA with a stack.



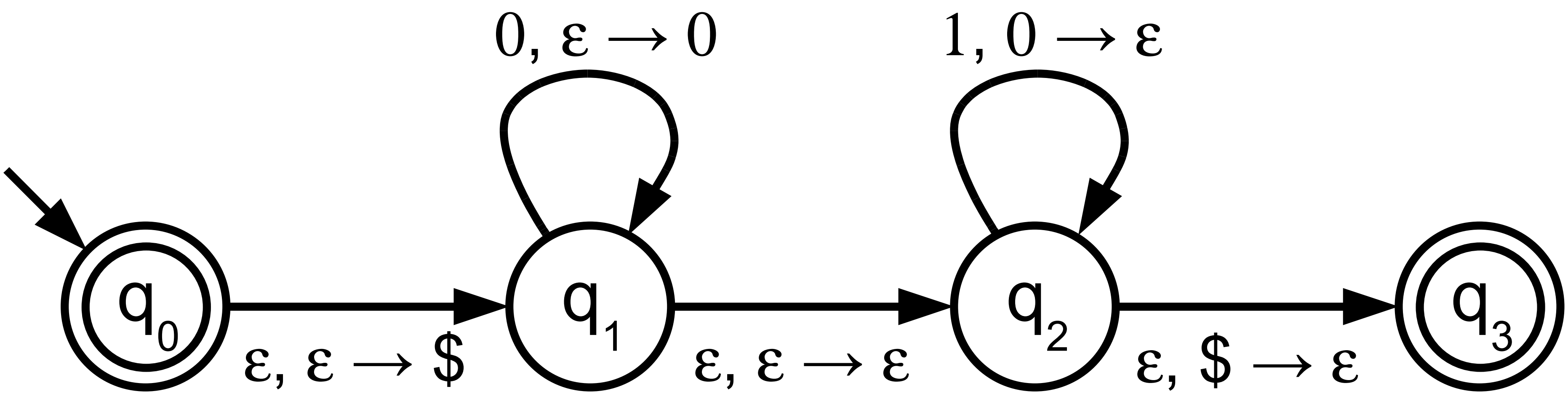
This means: "read x from the input;

pop y off the stack;

push z onto the stack"

Any of x,y,z may be ε.

Example: PDA for $L = \{0^n1^n : n \geq 0\}$



The $ is a special symbol to recognize end of stack

Idea:

q_1 : read and push 0s onto stack until no more

q_2 : read 1s and match with 0s popped from stack

Unlike the case for regular automata,
non-deterministic PDA are strictly more powerful
than deterministic PDA.

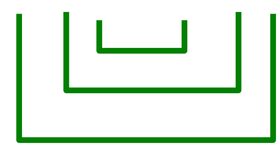
Compilers must work with deterministic PDA,
an important subclass of context-free languages

Non-context-free languages

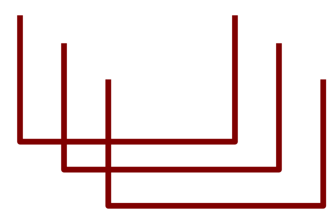
Intuition: If L involves regular expressions and/or nested matchings then probably context-free.

If not, probably not.

$\{ 0^n 1^n : n \geq 0 \}$ CF : 000 111 nested



$\{w\,w : w \in \Sigma^*\}$ not CF: 1101 1101 not nested



$\{0^n 1^n 2^n : n \geq 0\}$ not CF: 00 11 22 not nested

Non-context-free languages

There is a pumping lemma for context-free languages.

Similar to the one for regular, but simultaneously "pump" string in two parts:

$w = u\ v^i\ x\ y^i\ z$

Context-free pumping lemma:

L is CF language $\Rightarrow$

$\exists\, p \geq 0$

$\forall\, w \in L, |w| \geq p$

$\exists\, u,v,x,y,z :$

$w = uvxyz,\ |vy| > 0,\ |vxy| \leq p$

$\forall\, i \geq 0 : uv^ixy^iz \in L$

Context-free pumping lemma:

L is CF language $\Rightarrow$

$\exists\ p \geq 0$

$\forall\ w \in L,\ |w| \geq p$

$\exists\ u,v,x,y,z :$

$w = uvxyz,\ |vy| > 0,\ |vxy| \leq p$

$\forall\ i \geq 0 : uv^ixy^iz \in L$

Proof idea:

Let G be CFG : L(G) = L

If $w \in L$ is very long, derivation repeats a variable V

(like repeat states in regular P.L.)

vxy = piece of w that V derives: $V \Rightarrow^* vxy$

Because V repeated once, can repeat it again

Context-free pumping lemma:

L is CF language $\Rightarrow$

$\exists\, p \geq 0$ **A**

$\forall\, w \in L, |w| \geq p$

$\exists\, u,v,x,y,z$:

$w = uvxyz, \; |vy| > 0, \; |vxy| \leq p$

$\forall\, i \geq 0 : uv^ixy^iz \in L$

Useful to prove L NOT context-free.

Use contrapositive:

L context-free language $\Rightarrow$ A

same as

(not A) $\Rightarrow$ L not context-free

Context-free pumping lemma (contrapositive)

not A $\forall$ p $\geq$0 $\exists$ w $\in$ L, $|w| \geq p$ $\forall$ u,v,x,y,z : w = uvxyz, $|vy| > 0$, $|vxy| \leq p$ $\exists$ i $\geq$ 0 : $uv^ixy^iz \notin L$	$\Rightarrow$ L not context-free

To prove L not context-free it is enough to prove not A

Not A is the stuff in the box.

Context-free pumping lemma (contrapositive)

$\forall$ p $\geq$ 0
$\exists$ w $\in$ L, |w| $\geq$ p
$\forall$ u,v,x,y,z :
w = uvxyz, |vy| > 0, |vxy| $\leq$ p
$\exists$ i $\geq$ 0 : $uv^ixy^iz \notin L$

$\Rightarrow$ L not context-free

Adversary picks p,

You pick w $\in$ L of length $\geq$ p,

Adversary decomposes w = uvxyz, |vy| > 0, |vxy| $\leq$ p

You pick i $\geq$ 0

Finally, you win if $uv^ixy^iz \notin L$

Theorem: $L := \{a^n b^n c^n : n \geq 0\}$ is not context-free.

Proof:

Adversary moves p

You move $w := a^p b^p c^p$

Adversary moves u,v,x,y,z

You move i := 2

You must show $uvvxyyz \notin L$:

vy misses at least one symbol in $\Sigma = \{a,b,c\}$

since ?

$\forall$ $p \geq 0$

$\exists$ $w \in L$, $|w| \geq p$

$\forall$ u,v,x,y,z : w = uvxyz,

$|vy| > 0$, $|vxy| \leq p$

$\exists$ $i \geq 0$: $uv^ixy^iz \notin L$

Theorem: $L := \{a^n b^n c^n : n \geq 0\}$ is not context-free.

Proof:

Adversary moves p

You move $w := a^p b^p c^p$

Adversary moves u,v,x,y,z

You move i := 2

You must show uvvxyyz $\notin$ L:

vy misses at least one symbol in $\Sigma = \{a,b,c\}$

since between as and cs there are p bs, and $|vy| \leq p$

so uvvxyyz ????

$\forall$ p $\geq$0

$\exists$ w $\in$ L, $|w| \geq p$

$\forall$ u,v,x,y,z : w = uvxyz,

$|vy| > 0$, $|vxy| \leq p$

$\exists$ $i \geq 0 : uv^ixy^iz \notin L$

Theorem: $L := \{a^n b^n c^n : n \geq 0\}$ is not context-free.

Proof:

Adversary moves p

You move $w := a^p b^p c^p$

Adversary moves u,v,x,y,z

You move i := 2

You must show uvvxyyz ∉ L:

vy misses at least one symbol in ∑ = {a,b,c}

since between as and cs there are p bs, and |vy| ≤ p

so uvvxyyz has too few of that symbol, so ∉ L

DONE

∀ p ≥0

∃ w ∈ L, |w| ≥ p

∀ u,v,x,y,z : w = uvxyz, |vy|> 0, |vxy|≤ p

∃ i ≥ 0 : $uv^ixy^iz \notin L$

Theorem: $L := \{a^i b^j c^k : 0 \le i \le j \le k\}$ is not context-free.

Proof:

Adversary moves p

You move $w := a^p b^p c^p$

Adversary moves u,v,x,y,z

$\forall$ $p \geq 0$

$\exists$ $w \in L$, $|w| \geq p$

$\forall$ u,v,x,y,z : $w = uvxyz$,

$|vy| > 0$, $|vxy| \le p$

$\exists$ $i \geq 0$: $uv^ixy^iz \notin L$

So far, same as $\{a^n b^n c^n : n \geq 0\}$.

But now we need a few cases.

Our choice of i depends on u,v,x,y,z

Theorem: L := $\{a^i b^j c^k : 0 \leq i \leq j \leq k\}$ is not context-free.

Proof (cont.):

You have $w = a^p b^p c^p$, with $w = uvxyz$, $|vy| > 0$, $|vxy| \leq p$.

You must pick $i \geq 0$ and show $uv^ixy^iz \notin L$.

If no a's in vy: ?

Theorem: $L := \{a^i b^j c^k : 0 \leq i \leq j \leq k\}$ is not context-free.

Proof (cont.):

You have $w = a^p b^p c^p$, with $w = uvxyz$, $|vy| > 0$, $|vxy| \leq p$.

You must pick $i \geq 0$ and show $uv^i xy^i z \notin L$.

If no a's in vy: $uv^0 xy^0 z$ has fewer b's or c's than a's.

If no c's in vy: ?

Theorem: $L := \{a^i\, b^j\, c^k : 0 \leq i \leq j \leq k\}$ is not context-free.

Proof (cont.):

You have $w = a^p b^p c^p$, with $w = uvxyz$, $|vy| > 0$, $|vxy| \leq p$.

You must pick $i \geq 0$ and show $uv^i xy^i z \notin L$.

If no a's in vy: uv^0xy^0z has fewer b's or c's than a's.

If no c's in vy: uv^2xy^2z has more a's or b's than c's.

If no b's in vy:

?

Theorem: $L := \{a^i b^j c^k : 0 \leq i \leq j \leq k\}$ is not context-free.

Proof (cont.):

You have $w = a^p b^p c^p$, with $w = uvxyz$, $|vy| > 0$, $|vxy| \leq p$.

You must pick $i \geq 0$ and show $uv^ixy^iz \notin L$.

If no a's in vy: uv^0xy^0z has fewer b's or c's than a's.

If no c's in vy: uv^2xy^2z has more a's or b's than c's.

If no b's in vy:

You fall in a previous case, since $|vxy| \leq p$

DONE

Theorem: $L := \{s\,s : s \in \{0,1\}^*\}$ is not context-free.

Proof:

Adversary moves p

You move $w := 0^p\, 1^p\, 0^p\, 1^p$

$\forall\ p \geq 0$

$\exists\ w \in L,\ |w| \geq p$

$\forall\ u,v,x,y,z :\ w = uvxyz,$

$|vy| > 0,\ |vxy| \leq p$

$\exists\ i \geq 0 : uv^i x y^i z \notin L$

Note: To prove L not regular we moved $w = 0^p\, 1\, 0^p\, 1$

That move does not work for context-free!

Theorem: $L := \{s\, s : s \in \{0,1\}^* \}$ is not context-free.

Proof:

Adversary moves p

You move $w := 0^p\, 1^p\, 0^p\, 1^p$

Adversary moves u,v,x,y,z

Three cases:

vxy in 1st half of w: ?

$\forall\, p \geq 0$

$\exists\, w \in L, |w| \geq p$

$\forall\, u,v,x,y,z : w = uvxyz,$

$|vy| > 0, |vxy| \leq p$

$\exists\, i \geq 0 : uv^ixy^iz \notin L$

Theorem: $L := \{s\,s : s \in \{0,1\}^*\}$ is not context-free.

Proof:

Adversary moves p

You move $w := 0^p\, 1^p\, 0^p\, 1^p$

Adversary moves u,v,x,y,z

Three cases:

$\forall$ $p \geq 0$

$\exists$ $w \in L$, $|w| \geq p$

$\forall$ u,v,x,y,z : $w = uvxyz$,

$|vy| > 0$, $|vxy| \leq p$

$\exists$ $i \geq 0 : uv^ixy^iz \notin L$

vxy in 1st half of w: 2nd half of uv^2xy^2z starts with 1,

but uv^2xy^2z still starts with 0.

vxy in 2nd half of w: ?

Theorem: $L := \{s\,s : s \in \{0,1\}^*\}$ is not context-free.

Proof:

Adversary moves p

You move $w := 0^p\, 1^p\, 0^p\, 1^p$

Adversary moves u,v,x,y,z

Three cases:

$\forall$ $p \geq 0$

$\exists$ $w \in L$, $|w| \geq p$

$\forall$ u,v,x,y,z : $w = uvxyz$,

$|vy| > 0$, $|vxy| \leq p$

$\exists$ $i \geq 0$: $uv^ixy^iz \notin L$

vxy in 1st half of w: 2nd half of uv^2xy^2z starts with 1,

but uv^2xy^2z still starts with 0.

vxy in 2nd half of w: 1st half of uv^2xy^2z ends with 0,

but uv^2xy^2z still ends with 1.

vxy touches midpoint: ?

Theorem: $L := \{s\,s : s \in \{0,1\}^*\}$ is not context-free.

Proof:

$\forall\ p \geq 0$
$\exists\ w \in L, \lvert w \rvert \geq p$
$\forall\ u,v,x,y,z :\ w = uvxyz,$
$\lvert vy \rvert > 0, \lvert vxy \rvert \leq p$
$\exists\ i \geq 0 : uv^ixy^iz \notin L$

Adversary moves p

You move $w := 0^p\,1^p\,0^p\,1^p$

Adversary moves u,v,x,y,z

Three cases:

vxy in 1st half of w: 2nd half of uv^2xy^2z starts with 1,

but uv^2xy^2z still starts with 0.

vxy in 2nd half of w: 1st half of uv^2xy^2z ends with 0,

but uv^2xy^2z still ends with 1.

vxy touches midpoint:

$uv^0xy^0z = 0^p\,1^i\,0^j\,1^p$ with either $i < p$ or $j < p$. DONE

L := { w ∈ {a,b}* : w has same number of a and b}

Grammar for L

??

L := { w ∈ {a,b}* : w has same number of a and b}

Grammar for L

$S \to \varepsilon \mid SS \mid aSb \mid bSa$

Not clear why this works.

It requires a proof.

Proofs by induction

Let P(n) be any claim

To prove "$\forall$ n ≥ 0, P(n) is true" it suffices to prove

Base case: P(0) is true

Induction step: $\forall$ n : (($\forall$ i < n, P(i)) => P(n))

Induction hypothesis

You can replace "0" by any fixed value

Example: $P(n) = \sum_{i=0}^{n} i = n(n+1)/2$

Claim: $\forall\ n \geq 0, P(n)$

Proof by induction:

Base case: $P(0)$

$0 = 0(1)/2 = 0$ is true

Induction step: $\forall\ n : ((\forall\ i < n, P(i)) \Rightarrow P(n))$

$\sum_{i=0}^{n} i =$??

Example: $P(n) = \sum_{i=0}^{n} i = n(n+1)/2$

Claim: $\forall\ n \geq 0, P(n)$

Proof by induction:

Base case: $P(0)$

$$0 = 0(1)/2 = 0 \text{ is true}$$

Induction step: $\forall\ n : (\ (\forall\ i < n, P(i)\ ) \Rightarrow\ P(n)\ )$

$$\sum_{i=0}^{n} i = \sum_{i=0}^{n-1} i + n = (n-1)n/2 + n = n(n+1)/2$$

$L := \{ w \in \{a,b\}^* :$ w has same number of a and b$\}$

$S \to \varepsilon \mid SS \mid aSb \mid bSa$

Claim: For any $w \in \{a,b\}^*$, $S \to^* w$ if and only if $w \in L$

Proof of "only if": Suppose $S \to^* w$. Must show $w \in L$.

This fact is self-evident.

We show a proof by induction nevertheless,

as a warm-up for the other direction,

which is not self-evident.

$L := \{ w \in \{a,b\}^* : w$ has same number of a and b$\}$

$S \to \varepsilon \mid SS \mid aSb \mid bSa$

Claim: For any $w \in \{a,b\}^*$, $S \to^* w$ if and only if $w \in L$

Proof of “only if”: Suppose $S \to^* w$. Must show $w \in L$.

Let P(n) = any $w \in \{S,a,b\}^*$ such that $S \to^* w$ in n steps

has same number of a and b.

Base case (n=1): ??

L := { w ∈ $\{a,b\}^*$: w has same number of a and b}

$S \to \varepsilon \mid SS \mid aSb \mid bSa$

Claim: For any w ∈ $\{a,b\}^*$, $S \to^* w$ if and only if w ∈ L

Proof of “only if”: Suppose $S \to^* w$. Must show w ∈ L.

Let P(n) = any w ∈ $\{S,a,b\}^*$ such that $S \to^* w$ in n steps

has same number of a and b.

Base case (n=1): ε, SS, aSb, bSa have same number.

Induction step: Suppose $S \to^* w' \to w$

where $S \to^* w'$ in n-1 steps.

By induction hypothesis, ??

$L := \{ w \in \{a,b\}^* : w$ has same number of a and b$\}$

$S \rightarrow \varepsilon \mid SS \mid aSb \mid bSa$

Claim: For any $w \in \{a,b\}^*$, $S \rightarrow^* w$ if and only if $w \in L$

Proof of "only if": Suppose $S \rightarrow^* w$. Must show $w \in L$.

Let $P(n)$ = any $w \in \{S,a,b\}^*$ such that $S \rightarrow^* w$ in n steps

has same number of a and b.

Base case (n=1): ε, SS, aSb, bSa have same number.

Induction step: Suppose $S \rightarrow^* w' \rightarrow w$

where $S \rightarrow^* w'$ in n-1 steps.

By induction hypothesis, w' has same number of a, b.

Since any rule adds same number of a and b, w has too.

$L := \{ w \in \{a,b\}^* : w$ has same number of a and b$\}$

$S \rightarrow \varepsilon \mid SS \mid aSb \mid bSa$

Claim: For any $w \in \{a,b\}^*$, $S \rightarrow^* w$ if and only if $w \in L$

Proof of "if": Suppose $w \in L$. Must show $S \rightarrow^* w$

Let $P(n) = \forall\ w \in \{S,a,b\}^*, |w| = n, S \rightarrow^* w$.

Base case: $w = \varepsilon$. Use rule ??

$L := \{ w \in \{a,b\}^* : w$ has same number of a and b$\}$

$S \rightarrow \varepsilon \mid SS \mid aSb \mid bSa$

Claim: For any $w \in \{a,b\}^*$, $S \rightarrow^* w$ if and only if $w \in L$

Proof of "if": Suppose $w \in L$. Must show $S \rightarrow^* w$

Let $P(n) = \forall\ w \in \{S,a,b\}^*, |w| = n, S \rightarrow^* w$.

Base case: $w = \varepsilon$. Use rule $S \rightarrow \varepsilon$

Induction step: Let $|w| = n$.

This step is more complicated, and is the "creative step" of this proof.

$S \rightarrow \varepsilon \mid SS \mid aSb \mid bSa$

Induction step: Let $|w| = n$.

Define c_i := number of a - number of b in $w_1 w_2 \ldots w_i$

$c_0 = 0$ $c_n =$??

$S \rightarrow \varepsilon \mid SS \mid aSb \mid bSa$

Induction step: Let $|w| = n$.

Define c_i := number of a - number of b in $w_1\, w_2 \ldots w_i$

$c_0 = 0 \quad c_n = 0$

If $\exists\ 0 < i < n : c_i = 0$

then w = ??

$S \to \varepsilon \mid SS \mid aSb \mid bSa$

Induction step: Let $|w| = n$.

Define c_i := number of a - number of b in $w_1 w_2 \ldots w_i$

$c_0 = 0$ $c_n = 0$

If $\exists\ 0 < i < n : c_i = 0$

then $w = w' w''$, where $w', w'' \in L$,

and $|w'| < n$, $|w''| < n$

By induction hypothesis. $S \to^* w'$, $S \to^* w''$.

Hence $S \to SS \to^* w' S \to w' w'' = w$

$S \to \varepsilon \mid SS \mid aSb \mid bSa$

Induction step: Let $|w| = n$.

Define c_i := number of a - number of b in $w_1 w_2 \ldots w_i$

$c_0 = 0 \quad c_n = 0$

If $\forall\ 0 < i < n : c_i > 0$

then w = ??

$S \rightarrow \varepsilon \mid SS \mid aSb \mid bSa$

Induction step: Let $|w| = n$.

Define c_i := number of a - number of b in $w_1 \, w_2 \ldots w_i$

$c_0 = 0 \quad c_n = 0$

If $\forall \; 0 < i < n : c_i > 0$

then $w = a\,w'\,b$, where $w' \in L$ and $|w'| < n$

By induction hypothesis. $S \rightarrow^* w'$

Hence $S \rightarrow aSb \rightarrow^* a\,w'\,b = w$

$S \rightarrow \varepsilon \mid SS \mid aSb \mid bSa$

Induction step: Let $|w| = n$.

Define c_i := number of a - number of b in $w_1 w_2 \ldots w_i$

$c_0 = 0$ $c_n = 0$

If $\forall\ 0 < i < n : c_i < 0$

then w = ??

$S \to \varepsilon \mid SS \mid aSb \mid bSa$

Induction step: Let $|w| = n$.

Define c_i := number of a - number of b in $w_1 w_2 \ldots w_i$

$c_0 = 0 \quad c_n = 0$

If $\forall\ 0 < i < n : c_i < 0$

then $w = b\, w'\, a$, where $w' \in L$ and $|w'| < n$

By induction hypothesis. $S \to^* w'$

Hence $S \to bSa \to^* b\, w'\, a = w$

$S \rightarrow \varepsilon \mid SS \mid aSb \mid bSa$

Induction step: Let $|w| = n$.

Define c_i := number of a - number of b in $w_1 w_2 \ldots w_i$

$c_0 = 0$ $c_n = 0$

These three cover all cases, because two consecutive c_i differ by 1. So the c_i cannot change sign without going through 0

DONE