

The Pumping Lemma for CFL

Let G be a Context-Free Grammar.

There is an integer N such that

$$\omega \in L(G), \quad |\omega| \geq N$$

can be written:

$$\omega = UVXYZ$$

where

$$V \neq \lambda \text{ or } Y \neq \lambda \quad (\text{i.e., } |VY| \geq 1)$$

$$\text{and } |VXY| \leq N$$

$$\text{and Forall } i, \quad UV^iXY^iZ \in L(G)$$

Slide Lecture 13 –375

PROOF:

Let G be a CNF grammar.

If $\omega \in L(G)$ and w is sufficiently long
Then the parse must contain a “long” path.

More precisely,
if $|w| \geq 2^i$ then w 's parse tree
must contain a path of at least $i+1$ in length.

Slide Lecture 13 –376

Assume we have K nonterminals and let $N = 2^K$.
If $\omega \in L(G)$ and $|\omega| \geq N$, $N = 2^K$
then the parse tree contains a path of length $\geq K + 1$

This path contains $K+2$ vertices:
that is, $K+1$ nonterminals and 1 terminal at the leaf.

Slide Lecture 13 –377

Since there are only K nonterminals,
AT LEAST ONE NONTERMINAL MUST APPEAR *TWICE*
along some path.

In fact, some nonterminal
must appear at least twice
near the bottom of the parse tree:

I.E., some nonterminal
must appear twice within $K+1$ nonterminal vertices
counting up from the bottom of the parse tree.

Slide Lecture 13 –378

More formally,

1. Two vertices, v_1 and v_2 have the same (nonterminal) label.
2. Let v_1 be the vertex closer to the root.
3. The path from v_1 to a leaf is at most $K+1$ in length.

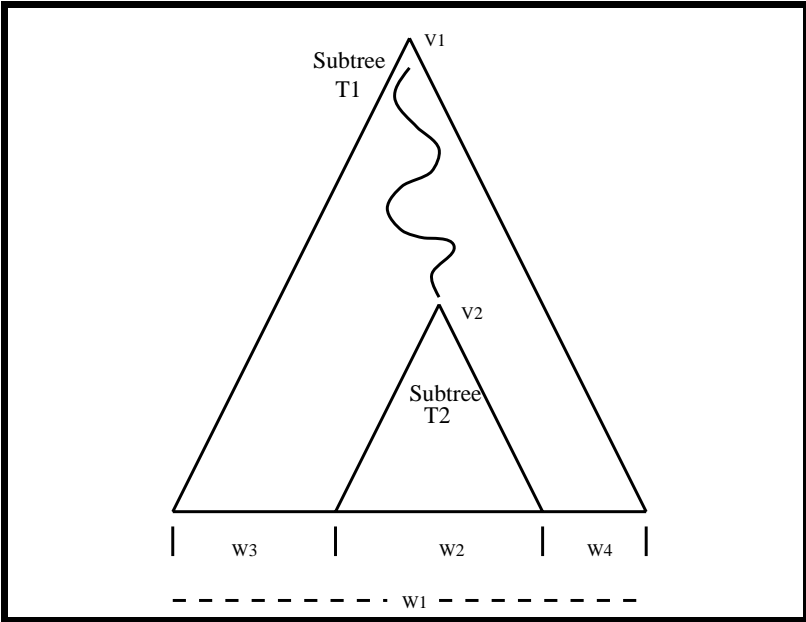
Slide Lecture 13 –379

We can now talk about a subtree T_1
rooted at v_1 which represents
the derivation of some subword ω_1 ;
Property 3 above shows that $|\omega_1| \leq 2^K$.

Slide Lecture 13 –380

Let T_2 be a subtree rooted at node v_2
and let ω_2 be the yield of T_2 .
Subtree T_1 has a yield of $\omega_1 = \omega_3\omega_2\omega_4$.

Slide Lecture 13 -381



Slide Lecture 13 -382

Both ω_3 and ω_4 cannot be λ since the first production of T_1 must be of the form:

$$A \rightarrow BC$$

And T_2 must be a subtree of T_1 generated under B or C.

Slide Lecture 13 –383

Therefore there exist nonterminal D which appears at v_1 and v_2 which generates the following:

$$D \xrightarrow{*} \omega_3 D \omega_4$$

and

$$D \xrightarrow{*} \omega_2$$

Slide Lecture 13 –384

By substitution:

$$D \xrightarrow{*} \omega_3 D \omega_4$$

$$D \xrightarrow{*} \omega_3 \omega_3 D \omega_4 \omega_4$$

and in general

$$D \xrightarrow{*} (\omega_3)^i D (\omega_4)^i$$

Now the yield of the parse tree before reaching T_1 must also be considered. There is some U and Z , such that

$$\omega = U\omega_1 Z = U\omega_3\omega_2\omega_4 Z$$

Slide Lecture 13 –385

But, as we have already shown by substitution for ω_1 ,

$$\text{If } U\omega_3\omega_2\omega_4 Z \in L(G)$$

$$\text{Then } U(\omega_3)^i \omega_2 (\omega_4)^i Z \in L(G)$$

Slide Lecture 13 –386

Put another way, for a sufficiently long string
(one that must repeat some nonterminal)
there exists (U, V, X, Y, Z) such that

$$\text{If } \omega = UVXYZ \in L(G)$$

$$\text{Then } UV^iXY^iZ \in L(G)$$

$$\text{Where } |VY| \geq 1 \text{ And } |VXY| \leq N = 2^K.$$

QED.

Slide Lecture 13 –387

EXAMPLE 1.

$$\text{Let } L = \{a^t b^t c^t \mid t \geq 1\}$$

Show this is not Context Free.

Let N be the constant of the pumping lemma.

$$\omega = a^N b^N c^N \in L.$$

Slide Lecture 13 –388

Assume the Language is Context Free.

Rewrite ω as UVXYZ.

Where $|VY| \geq 1$ And $|VXY| \leq N = 2^K$.

Where K is the number of nonterminals in the CFG.

Slide Lecture 13 –389

Since $|VXY| \leq N = 2^K$,

VY cannot contain instances of both a's and c's, since the rightmost "a" is N+1 positions away from the leftmost "c."

The only possible cases are:

1. VY could be all a's.
2. VY could be all b's.
3. VY could be all c's.
4. VY could be made up of a's and b's.
5. VY could be made up of b's and c's.

Slide Lecture 13 –390

For each of the 5 cases,

$$UV^iXY^iZ \notin L(G) \text{ for } i \neq 1$$

Thus $L = \{a^tb^tc^t | t \geq 1\}$

is not Context Free.

Slide Lecture 13 –391

We also note that the language

$$L = \{xcx | x \in \{a, b\}^*\}$$

is not context free.

This means that languages such as Pascal that require variables be declared before they are used are not strictly context free.

In other words, this requirement of the language cannot be expressed in BNF notation, which is context free.

Slide Lecture 13 –392

TURING MACHINES

- We have looked at FINITE AUTOMATA, PUSHDOWN AUTOMATA. But still cannot recognize simple languages.
- Example: $a^n b^n c^n | n \geq 0$
- In 1936 Alan Turing Proposed what is now known as the **TURING MACHINE.**

Slide Lecture 13 –393

It is:

1. Extremely Simple
2. Recognizes all languages that can be recognized by any known machine (despite many subsequent models).

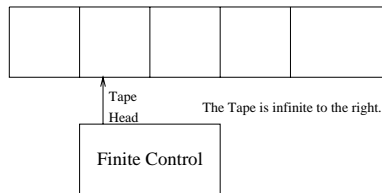
Resulting Conjecture

The Turing Machine is
the most general computing model
which can be finitely described.

Slide Lecture 13 –394

BASIC MODEL

A Turing Machine is a finite state machine extended with a Read/Write tape with discrete cells capable of holding designated tape symbols.



Slide Lecture 13 –395

Each move of a TM depends on the symbol under the tape head and the current state of the finite control.

A *MOVE* consists of :

1. Determining the next state of the finite control.
2. Writing a symbol on the tape cell that is currently addressed by the tape head .
3. Moving the head Left, Stationary, or Right.

Slide Lecture 13 –396

Initially the leftmost N cells
contain the input sequence.

All other cells contain the blank symbol
(denoted $\#$, or Δ by Martin)

Initially the tape head is positioned
under the leftmost cell.

(Sometimes $\#$ is also used to mark
the leftmost cell, and thus to bracket the input.)

Slide Lecture 13 –397

An Accepting Computation

If the TM enters the special *Halting State*, h it halts.

Hung Computation

If the TM attempts to move
to the left of the leftmost symbol it *Hangs*
which means it ceases to operate without halting.

Slide Lecture 13 –398

More formally, A TM can be defined as: $TM = (K, \Sigma, \delta, S)$
where

1. K : finite set of state not containing h
2. Σ : Alphabet of symbols, including $\#$, but not L or R
3. $S \in K$, the Initial State
4. δ : Function mapping
 $K \times \Sigma \rightarrow (K \cup h) \times (\Sigma \cup L, R)$

(Martin includes an *Input* alphabet and a *Tape* alphabet, but we don't have to distinguish between the two.)

Slide Lecture 13 –399

If $q \in K$ and $a \in \Sigma$ and $\delta(q, a) = (p, b)$ then M will go from state q to p while scanning symbol a , and either

1. Rewrite a with b if $b \in \Sigma$ or
2. Move the tape head left (if $b = L$) or right (if $b = R$)

The operation of the machine is deterministic
and it will stop only if it

1. Enters the halting state, h or
2. Attempts to move off the left end of the tape.

Slide Lecture 13 –400

Example 1:

Let $M = (K, \Sigma, \delta, S)$ where

$K = \{q_0, q_1\}$, $\Sigma = \{a, \#\}$, $S = q_0$

The function δ is given as follows:

Current State	INPUT	
	a	#
q_0	$(q_1, \#)$	$(h, \#)$
q_1	(q_0, a)	(q_0, R)

Slide Lecture 13 –401

This Machine simply scans the sequence left to right, *erasing* the *a*'s (converting them to #), and halting when finished.

Note that

$\delta(q_1, a) \rightarrow (q_0, a)$ is never used. It is included to make δ a complete function.

δ is often defined as a partial function.

Slide Lecture 13 –402

Example 2:

Let $M = (K, \Sigma, \delta, S)$ where

$K = q_0$, $\Sigma = \{a, \#\}$, $S = q_0$

The function δ is given as follows:

Current State	INPUT	
	a	#
q_0	(q_0, R)	$(h, \#)$

Slide Lecture 13 –403

The Machine in example 2 scans the a 's from left to right, and halts when it encounters $\#$.

If the symbol being scanned is not a or $\#$ the machine *hangs*.

Slide Lecture 13 –404

CONFIGURATIONS:

A configuration captures an instantaneous description of a TM.

This description specifies:

1. Current State
2. Tape Contents
3. Head Position

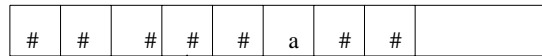
Slide Lecture 13 –405

Since all but a finite portion of the tape will be blank, tape contents can be represented by a finite string. We decompose this string into 3 parts.

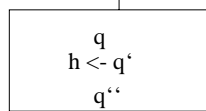
1. The (possibly empty) string to the left of the cell under the tape head
2. The symbol being scanned
3. The (possibly empty) string to the right of the cell under the tape head not containing # as the rightmost symbol

Slide Lecture 13 –406

Example (h, ###, #, #a)

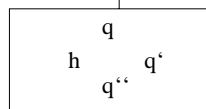


A Halting Configuration



Slide Lecture 13 -407

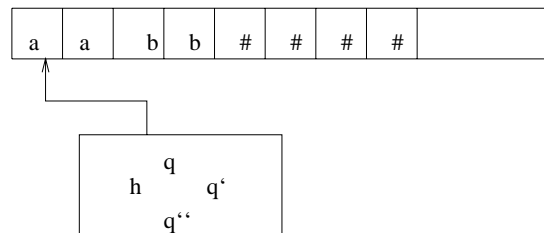
Example (q, #a#, #,)



Slide Lecture 13 -408

A configuration then is an element of $K \cup h \dots$ A state
 $\times \Sigma^*$ Left Hand String
 $\times \Sigma$ Scanned Symbol
 $\times \Sigma^*((\Sigma - \#) \cup \lambda)$

Example (q, .a,abb)



Slide Lecture 13 –409

All configurations yield
 exactly one configuration in a single step except
halting and *hanging* configurations.

As usual \vdash_M^* is the reflexive transitive closure of \vdash_M .

C_1 yields configuration of C_2 if $C_1 \vdash_M^* C_2$.

A computation (of length n) is a sequence of configurations
 $C_0 \vdash_M C_1 \vdash_M \dots C_n$ where $n \geq 0$.

Slide Lecture 13 –410

For TM's we define "Yields in one step" as follows:
 Let (q_1, W_1, a_1, u_1) and (q_2, W_2, a_2, u_2) be configurations.
 $(q_1, W_1, a_1, u_1) \vdash (q_2, W_2, a_2, u_2)$
 iff for some $b \in (\Sigma \cup L, R)$ $\delta(q_1, a_1) = (q_2, b)$
 and one of the three cases holds:

1. $b \in \Sigma$, $W_1 = W_2$, $u_1 = u_2$ and $a_2 = b$

2. $b = L$, $W_1 = W_2 a_2$ and either

$$u_2 = a_1 u_1 \text{ or } u_2 = \lambda$$

3. $b = R$, $W_2 = W_1 a_1$ and either

$$u_1 = a_2 u_2 \text{ or } u_1 = u_2 = \lambda, \quad a_2 = \#$$

Slide Lecture 13 -411

Examples:

Let W & $u \in \Sigma^*$
 where u doesn't end with $\#$ and $\Sigma = a, b$.

Case 1: $\delta(q_1, a) = (q_2, b)$
 $(q_1, W \underline{a} u) \vdash (q_2, W \underline{b} u)$

"b" is written over "a"

Case 2: $\delta(q_1, a) = (q_2, L)$
 case(a) $(q_1, W b \underline{a} u) \vdash_M (q_2, W \underline{b} a u)$
 case(b) $(q_1, W b \underline{\#}) \vdash_M (q_2, W \underline{b})$

A move left operation

Slide Lecture 13 -412

Case 3: $\delta(q_1, a) = (q_2, R)$

(a) $(q_1, W\underline{a}bu) \vdash_M (q_2, Wabu)$

(b) $(q_1, W\underline{a}) \vdash_M (q_2, W\underline{a}\#)$

A move right operation

If $\delta(q_1, a) = (q_2, b)$ and $b = L$ and if $W_1 = \lambda$ then
 (q_1, W_1, a, u) is a *hanging* configuration.