

# Parikh's Theorem for CFL's

Deepak D'Souza

Department of Computer Science and Automation  
Indian Institute of Science, Bangalore.

16 October 2019

# Outline

- 1 Parikh map
- 2 Parikh's theorem
- 3 Proof

# Rohit Parikh



## Parikh map of a string

- Let  $A = \{a_1, \dots, a_n\}$  be a finite alphabet.
- The **Parikh map** of a string  $w \in A^*$  is a vector in  $\mathbb{N}^n$  given by:

$$\psi(w) = (\#_{a_1}(w), \#_{a_2}(w), \dots, \#_{a_n}(w)).$$

- For example if  $A = \{a, b\}$ , then  $\psi(baabb) = (2, 3)$ .
- Parikh map is also called the “letter-count” of a string.
- Extend the map to languages  $L$  over  $A$ :

$$\psi(L) = \{\psi(w) \mid w \in L\}.$$

- What is  $\psi(\{a^n b^n \mid n \geq 0\})$ ?

## Parikh map of a string

- Let  $A = \{a_1, \dots, a_n\}$  be a finite alphabet.
- The **Parikh map** of a string  $w \in A^*$  is a vector in  $\mathbb{N}^n$  given by:

$$\psi(w) = (\#_{a_1}(w), \#_{a_2}(w), \dots, \#_{a_n}(w)).$$

- For example if  $A = \{a, b\}$ , then  $\psi(baabb) = (2, 3)$ .
- Parikh map is also called the “letter-count” of a string.
- Extend the map to languages  $L$  over  $A$ :

$$\psi(L) = \{\psi(w) \mid w \in L\}.$$

- What is  $\psi(\{a^n b^n \mid n \geq 0\})$ ?
  - $\{(n, n) \mid n \geq 0\}$ .
- What is  $\psi(\{w \in \{a, b\}^* \mid \#_a(w) \leq \#_b(w)\})$ ?

## Parikh map of a string

- Let  $A = \{a_1, \dots, a_n\}$  be a finite alphabet.
- The **Parikh map** of a string  $w \in A^*$  is a vector in  $\mathbb{N}^n$  given by:

$$\psi(w) = (\#_{a_1}(w), \#_{a_2}(w), \dots, \#_{a_n}(w)).$$

- For example if  $A = \{a, b\}$ , then  $\psi(baabb) = (2, 3)$ .
- Parikh map is also called the “letter-count” of a string.
- Extend the map to languages  $L$  over  $A$ :

$$\psi(L) = \{\psi(w) \mid w \in L\}.$$

- What is  $\psi(\{a^n b^n \mid n \geq 0\})$ ?
  - $\{(n, n) \mid n \geq 0\}$ .
- What is  $\psi(\{w \in \{a, b\}^* \mid \#_a(w) \leq \#_b(w)\})$ ?
  - $\{(m, n) \mid m \leq n\}$ .

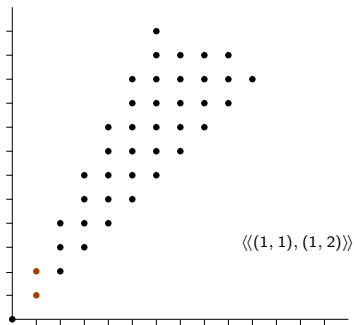
# Semi-linear sets of vectors

- The set of vectors **generated** by a set of vectors  $u_1, \dots, u_k$  in  $\mathbb{N}^n$ , denoted  $\langle\langle u_1, \dots, u_k \rangle\rangle$ , is the set

$$\{d_1 \cdot u_1 + d_2 \cdot u_2 + \dots + d_k \cdot u_k \mid d_i \in \mathbb{N}\}.$$

- A subset  $X$  of  $\mathbb{N}^n$  is called **linear** if there exist vectors  $u_0, u_1, \dots, u_k$  such that

$$X = u_0 + \langle\langle u_1, u_2, \dots, u_k \rangle\rangle.$$



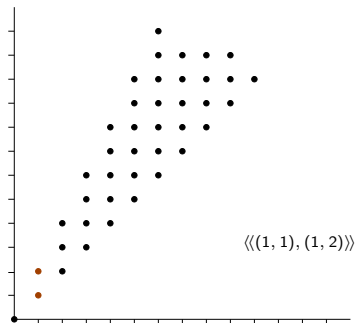
# Semi-linear sets of vectors

- The set of vectors **generated** by a set of vectors  $u_1, \dots, u_k$  in  $\mathbb{N}^n$ , denoted  $\langle\langle u_1, \dots, u_k \rangle\rangle$ , is the set

$$\{d_1 \cdot u_1 + d_2 \cdot u_2 + \dots + d_k \cdot u_k \mid d_i \in \mathbb{N}\}.$$

- A subset  $X$  of  $\mathbb{N}^n$  is called **linear** if there exist vectors  $u_0, u_1, \dots, u_k$  such that

$$X = u_0 + \langle\langle u_1, u_2, \dots, u_k \rangle\rangle.$$



- A set of vectors is called **semi-linear** if it is a finite union of linear sets.



# Parikh's Theorem for CFL's

## Theorem (Parikh)

*The Parikh map of a CFL is a semi-linear set. That is, if  $L$  is a CFL then  $\psi(L)$  is semi-linear.*

Some corollaries:

- Every CFL is “letter-equivalent” to a regular language.
  - For example:  $\psi(\{a^n b^n \mid n \geq 0\})$

# Parikh's Theorem for CFL's

## Theorem (Parikh)

*The Parikh map of a CFL is a semi-linear set. That is, if  $L$  is a CFL then  $\psi(L)$  is semi-linear.*

Some corollaries:

- Every CFL is “letter-equivalent” to a regular language.
  - For example:  $\psi(\{a^n b^n \mid n \geq 0\}) = \psi((ab)^*)$ .
- Lengths of a CFL forms an ultimately periodic set.

# Parikh's Theorem for CFL's

## Theorem (Parikh)

*The Parikh map of a CFL is a semi-linear set. That is, if  $L$  is a CFL then  $\psi(L)$  is semi-linear.*

Some corollaries:

- Every CFL is “letter-equivalent” to a regular language.
  - For example:  $\psi(\{a^n b^n \mid n \geq 0\}) = \psi((ab)^*)$ .
- Lengths of a CFL forms an ultimately periodic set.
- CFL's over a single-letter alphabet are regular.

# Parikh's Theorem for CFL's

## Theorem (Parikh)

*The Parikh map of a CFL is a semi-linear set. That is, if  $L$  is a CFL then  $\psi(L)$  is semi-linear.*

Some corollaries:

- Every CFL is “letter-equivalent” to a regular language.
  - For example:  $\psi(\{a^n b^n \mid n \geq 0\}) = \psi((ab)^*)$ .
- Lengths of a CFL forms an ultimately periodic set.
- CFL's over a single-letter alphabet are regular.

Is Parikh's theorem a sufficient condition for context-freeness as well?

# Parikh's Theorem for CFL's

## Theorem (Parikh)

*The Parikh map of a CFL is a semi-linear set. That is, if  $L$  is a CFL then  $\psi(L)$  is semi-linear.*

Some corollaries:

- Every CFL is “letter-equivalent” to a regular language.
  - For example:  $\psi(\{a^n b^n \mid n \geq 0\}) = \psi((ab)^*)$ .
- Lengths of a CFL forms an ultimately periodic set.
- CFL's over a single-letter alphabet are regular.

Is Parikh's theorem a sufficient condition for context-freeness as well?

- No, since  $\psi(\{a^n b^n c^n \mid n \geq 0\}) = \{(n, n, n) \mid n \geq 0\}$  is semi-linear.

# Running example

## CFG $G_1$

$$S \rightarrow XC \mid AY$$

$$X \rightarrow aXb \mid ab$$

$$Y \rightarrow bYc \mid bc$$

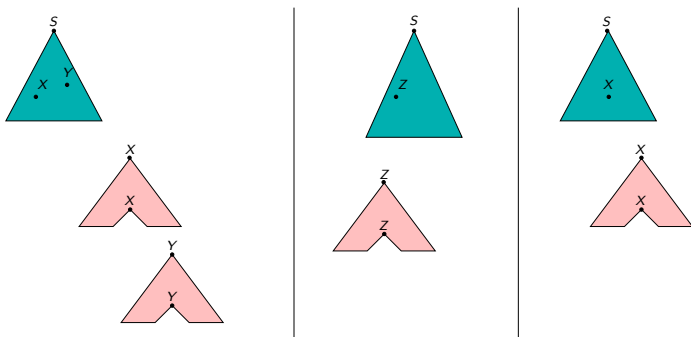
$$A \rightarrow aA \mid a$$

$$C \rightarrow cC \mid c$$

- What is the language generated?
- What is the Parikh image of this language?
- Write it as a semi-linear set.

# Idea of proof

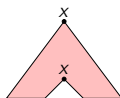
- Partition parse trees into finite number of blocks
- Each block is represented by a “minimal” parse tree and associated “basic pumps”.
- Argue that set of strings derived in each block gives rise to a linear set.



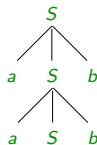
# Proof: Pumps

Let us fix a CFG  $G = (N, A, S, P)$  in CNF form.

- A **pump** is a derivation tree  $s$  which has at least two nodes, and  $yield(s) = x \cdot root(s) \cdot y$ , for some terminal strings  $x, y$ .

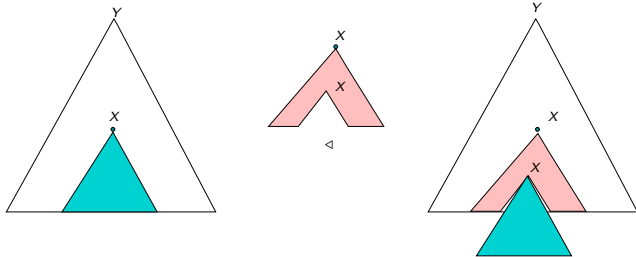


- Example pumps for grammar  $S \rightarrow aSb \mid SS \mid \epsilon$ :

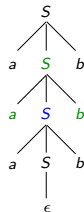
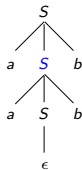
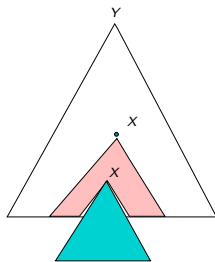
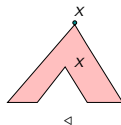
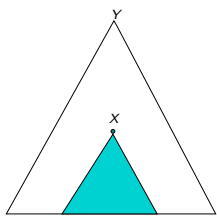




# Growing and shrinking with pumps

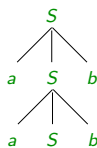


# Growing and shrinking with pumps



# Basic Pumps

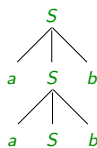
A pump is **basic** if it is  $\leftarrow$ -minimal. Thus a pump  $s$  is a basic pump if it cannot be shrunk by some pump and still remain a pump.



- First pump is basic but second is not.
- How many pumps are there?

# Basic Pumps

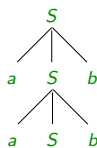
A pump is **basic** if it is  $\leftarrow$ -minimal. Thus a pump  $s$  is a basic pump if it cannot be shrunk by some pump and still remain a pump.



- First pump is basic but second is not.
- How many pumps are there? Infinitely many.
- How many basic pumps are there?

# Basic Pumps

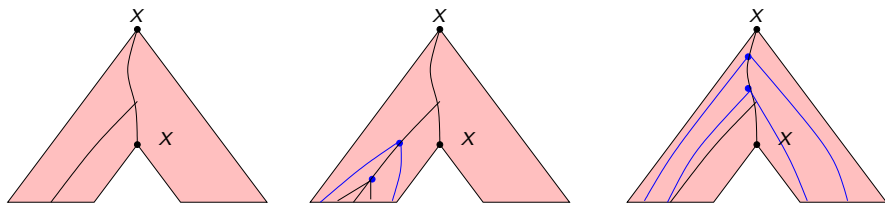
A pump is **basic** if it is  $\leftarrow$ -minimal. Thus a pump  $s$  is a basic pump if it cannot be shrunk by some pump and still remain a pump.



- First pump is basic but second is not.
- How many pumps are there? Infinitely many.
- How many basic pumps are there? Finitely many since their height is bounded by  $2|N|$  (See argument on next slide).
- Let  $p$  be the number of basic pumps.

# Basic pumps height bounded by $2N$

Consider the longest path from root to leaf in a basic pump.  
The number of nodes on it is bounded by  $2|N| + 1$ .



## $\leq$ relation on parse trees

- Let  $s$  and  $t$  be derivation trees of terminal strings starting from start symbol  $S$ .
- Then we say  $s \leq t$  iff  $t$  can be grown from  $s$  by basic pumps whose non-terminals are contained in those of  $s$  (thus the pumps do not introduce any new non-terminals, and  $s$  and  $t$  have the same set of non-terminal nodes).
- A parse tree  $s$  is thus  $\leq$ -minimal if it does not contain a basic pump that can be cut out **without** reducing the set of non-terminals that occur in  $s$ .
- $\leq$ -minimal trees can be seen to be finite in number: their height is bounded by  $(p + 1)(|N| + 1)$ .
- Ex: What are the  $\leq$ -minimal parse trees for grammar  $G_1$ ?

## Overall strategy of Proof

- Begin with the  $\leq$ -minimal derivation trees, say  $s_1, \dots, s_k$ .
- Associate with each  $s_i$  the set of basic pumps whose non-terminals are **contained in** that of  $s_i$ .
- Argue that the set of derivation trees obtained by starting with  $s_i$  and growing using the associated basic pumps, (let us call this the “bucket” of parse trees associated with  $s_i$ ) gives rise to a set of strings whose Parikh map is a linear set.

### Exercise

Describe the “buckets” for  $G_1$ , and the corresponding linear sets.