Polynomial-Time Identification in the Limit of k, l-Substitutable Context-Free Languages

Ryo Yoshinaka

Hokkaido University

Japan

1. Introduction

Substitutable Context-Free Languages

- Clark and Eyraud (2005, 2007)
- Substitutability: for any strings $x_1, y_1, z_1, x_2, y_2, z_2$,

$$x_1 y_1 z_1, x_1 y_2 z_1$$

Substitutable Context-Free Languages

- Clark and Eyraud (2005, 2007)
- Substitutability: for any strings $x_1, y_1, z_1, x_2, y_2, z_2$,

$$x_1 y_1 z_1, x_1 y_2 z_1, x_2 y_1 z_2 \in L \quad \Longrightarrow \quad x_2 y_2 z_2 \in L$$

Substitutable Context-Free Languages

- Clark and Eyraud (2005, 2007)
- Substitutability: for any strings $x_1, y_1, z_1, x_2, y_2, z_2$,

$$x_1 y_1 z_1, x_1 y_2 z_1, x_2 y_1 z_2 \in L \quad \Longrightarrow \quad x_2 y_2 z_2 \in L$$

- Example:
  - John loves Mary, John hates Mary

Substitutable Context-Free Languages

- Clark and Eyraud (2005, 2007)
- Substitutability: for any strings $x_1, y_1, z_1, x_2, y_2, z_2$,

$$x_1 y_1 z_1, x_1 y_2 z_1, x_2 y_1 z_2 \in L \quad \Longrightarrow \quad x_2 y_2 z_2 \in L$$

- Example:
  - John loves Mary, John hates Mary, a boy loves chocolate $\Longrightarrow$ a boy hates chocolate

Substitutable Context-Free Languages

- Clark and Eyraud (2005, 2007)
- Substitutability: for any strings $x_1, y_1, z_1, x_2, y_2, z_2$,

$$x_1y_1z_1, x_1y_2z_1, x_2y_1z_2 \in L \quad \Longrightarrow \quad x_2y_2z_2 \in L$$

- Example:
  - John loves Mary, John hates Mary, a boy loves chocolate $\Longrightarrow$ a boy hates chocolate
  - the man died, the man ordered dinner, the man who was hungry died

Substitutable Context-Free Languages

- Clark and Eyraud (2005, 2007)
- Substitutability: for any strings $x_1, y_1, z_1, x_2, y_2, z_2$,

$$x_1 y_1 z_1, x_1 y_2 z_1, x_2 y_1 z_2 \in L \implies x_2 y_2 z_2 \in L$$

- Example:
  - John loves Mary, John hates Mary, a boy loves chocolate
    $\implies$ a boy hates chocolate
  - the man died, the man ordered dinner,
    the man who was hungry died
    $\implies$ the man who was hungry ordered dinner

Substitutable Context-Free Languages

- Clark and Eyraud (2005, 2007)
- Substitutability: for any strings $x_1, y_1, z_1, x_2, y_2, z_2$,

$$x_1 y_1 z_1, x_1 y_2 z_1, x_2 y_1 z_2 \in L \quad \Longrightarrow \quad x_2 y_2 z_2 \in L$$

- Example:
  - ♦ John loves Mary, John hates Mary, a boy loves chocolate
    $\Longrightarrow$ a boy hates chocolate
  - ♦ the man died, the man ordered dinner,
    the man who was hungry died
    $\Longrightarrow$ the man who was hungry ordered dinner
  - ♦ a boy was hungry
    $\Longrightarrow$ the man who loves chocolate ordered dinner

Substitutable Context-Free Languages

- Natural languages

Substitutable Context-Free Languages

- Natural languages
- Subclass of CFLs that is
  efficiently identifiable in the limit from positive data

Substitutable Context-Free Languages

- Natural languages
- Subclass of CFLs that is efficiently identifiable in the limit from positive data
- Analogy with *reversible languages* (Angluin 1982)
  - Zero-reversible languages:

$$x_1 y_1, x_1 y_2, x_2 y_1 \in L \quad \Longrightarrow \quad x_2 y_2 \in L$$

  - Substitutable languages:

$$x_1 y_1 z_1, x_1 y_2 z_1, x_2 y_1 z_2 \in L \quad \Longrightarrow \quad x_2 y_2 z_2 \in L$$

Reversibility Hierarchy and Substitutability

- k-reversible languages:

$$x_1 v y_1, x_1 v y_2, x_2 v y_1 \in L \quad \Longrightarrow \quad x_2 v y_2 \in L$$

if $|v| = k$.

Reversibility Hierarchy and Substitutability

- k-reversible languages:

$$x_1 v y_1, x_1 v y_2, x_2 v y_1 \in L \quad \Longrightarrow \quad x_2 v y_2 \in L$$

  if $|v| = k$.

- Substitutable languages:

$$x_1 y_1 z_1, x_1 y_2 z_1, x_2 y_1 z_2 \in L \quad \Longrightarrow \quad x_2 y_2 z_2 \in L$$

Reversibility Hierarchy and Substitutability

- k-reversible languages:

$$x_1 v y_1, x_1 v y_2, x_2 v y_1 \in L \quad \Longrightarrow \quad x_2 v y_2 \in L$$

  if $|v| = k$.

- Substitutable languages:

$$x_1 y_1 z_1, x_1 y_2 z_1, x_2 y_1 z_2 \in L \quad \Longrightarrow \quad x_2 y_2 z_2 \in L$$

- k-substitutable languages?

This Talk

- k, l-substitutability for $k, l \geq 0$

This Talk

- k, l-substitutability for $k, l \geq 0$
- Polynomial-time identifiability of k, l-substitutable context-free languages (k, l-SCFLs)

This Talk

- k, l-substitutability for $k, l \geq 0$
- Polynomial-time identifiability of k, l-substitutable context-free languages (k, l-SCFLs)

— Outline —

1. Introduction
2. Priliminaries
3. Learning Algorithm for k, l-SCFLs
4. Summary and Future Work

2. Preliminaries

λ-free Context-Free Grammars

- Σ: alphabet, λ: empty string,
  - v, u, w, x, y, z: strings over Σ
- $L \subseteq \Sigma^*$: language

λ-free Context-Free Grammars

- Σ: alphabet, λ: empty string,
  - v, u, w, x, y, z: strings over Σ
- $L \subseteq \Sigma^*$: language
- CFG $G = \langle \Sigma, V, P, S \rangle$
  - Σ: terminal symbols
  - V: nonterminal symbols ($V \cap \Sigma = \varnothing$)
  - $P \subseteq V \times (\Sigma \cup V)^+$: production rules ($\lambda$-free)
  - $S \in V$: start symbol
  - $L(G) = \{ w \in \Sigma^* \mid S \Rightarrow_G^* w \}$
  - $||G|| = \sum_{A \to \alpha \in P} |A\alpha|$

Identification in the Limit from Positive Data

Strings from the target : $w_1 \quad w_2 \quad w_3 \quad w_4 \quad \ldots \quad \in L(G_*)$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$\mathcal{A}$: Learner

Conjectures by the learner :

- learning algorithm: $\mathcal{A}$
- learning target: $L(G_*)$
- positive data: $\langle w_1, w_2, \ldots \rangle$ such that $L(G_*) = \{ w_i \mid i \in \mathbb{N} \}$
- conjecture: $G_m = \mathcal{A}(w_1, \ldots, w_m)$

Identification in the Limit from Positive Data

Strings from the target : $w_1 \quad w_2 \quad w_3 \quad w_4 \quad \ldots \quad \in L(G_*)$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$\mathcal{A}$: Learner

Conjectures by the learner :

- learning algorithm: $\mathcal{A}$
- learning target: $L(G_*)$
- positive data: $\langle w_1, w_2, \ldots \rangle$ such that $L(G_*) = \{ w_i \mid i \in \mathbb{N} \}$
- conjecture: $G_m = \mathcal{A}(w_1, \ldots, w_m)$

Identification in the Limit from Positive Data

Strings from the target : w_1 w_2 w_3 w_4 $\ldots$ $\in L(G_*)$

$\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$

$\mathcal{A}$: Learner

$\downarrow$

Conjectures by the learner : G_1

- learning algorithm: $\mathcal{A}$
- learning target: $L(G_*)$
- positive data: $\langle w_1, w_2, \ldots \rangle$ such that $L(G_*) = \{ w_i \mid i \in \mathbb{N} \}$
- conjecture: $G_m = \mathcal{A}(w_1, \ldots, w_m)$

Identification in the Limit from Positive Data

Strings from the target : $w_1 \quad w_2 \quad w_3 \quad w_4 \quad \ldots \quad \in L(G_*)$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$\mathcal{A}$: Learner

$\downarrow$

Conjectures by the learner : G_1

- learning algorithm: $\mathcal{A}$
- learning target: $L(G_*)$
- positive data: $\langle w_1, w_2, \ldots \rangle$ such that $L(G_*) = \{ w_i \mid i \in \mathbb{N} \}$
- conjecture: $G_m = \mathcal{A}(w_1, \ldots, w_m)$

Identification in the Limit from Positive Data

Strings from the target : $w_1 \quad w_2 \quad w_3 \quad w_4 \quad \ldots \in L(G_*)$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$\mathcal{A}$: Learner

$\downarrow \quad \downarrow$

Conjectures by the learner : $G_1 \quad G_2$

- learning algorithm: $\mathcal{A}$
- learning target: $L(G_*)$
- positive data: $\langle w_1, w_2, \ldots \rangle$ such that $L(G_*) = \{ w_i \mid i \in \mathbb{N} \}$
- conjecture: $G_m = \mathcal{A}(w_1, \ldots, w_m)$

Identification in the Limit from Positive Data

Strings from the target : $w_1 \quad w_2 \quad w_3 \quad w_4 \quad \ldots \quad \in L(G_*)$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$\mathcal{A}$: Learner

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

Conjectures by the learner : $G_1 \quad G_2 \quad G_3 \quad G_4 \quad \ldots$

- learning algorithm: $\mathcal{A}$
- learning target: $L(G_*)$
- positive data: $\langle w_1, w_2, \ldots \rangle$ such that $L(G_*) = \{\, w_i \mid i \in \mathbb{N} \,\}$
- conjecture: $G_m = \mathcal{A}(w_1, \ldots, w_m)$

Identification in the Limit from Positive Data

Strings from the target : $w_1 \quad w_2 \quad w_3 \quad w_4 \quad \ldots \quad \in L(G_*)$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$\mathcal{A}$: Learner

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

Conjectures by the learner : $G_1 \quad G_2 \quad G_3 \quad G_4 \quad \ldots$

- learning algorithm: $\mathcal{A}$
- learning target: $L(G_*)$
- positive data: $\langle w_1, w_2, \ldots \rangle$ such that $L(G_*) = \{ w_i \mid i \in \mathbb{N} \}$
- conjecture: $G_m = \mathcal{A}(w_1, \ldots, w_m)$
- convergence: $\exists n_0 \in \mathbb{N} \, \forall m > n_0 \, [G_m = G_{n_0}]$
- $\mathcal{A}$ *identifies* $\mathbb{G}$ *in the limit from positive data* iff
  for any positive data of any $G_* \in \mathbb{G}$,
  $\mathcal{A}$ converges to a grammar G_{n_0} with $L(G_{n_0}) = L(G_*)$.

Identification in the Limit from Positive Data

Strings from the target : $w_1 \quad w_2 \quad w_3 \quad w_4 \quad \ldots \quad \in L(G_*)$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$\mathcal{A}$: Learner

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

Conjectures by the learner : $G_1 \quad G_2 \quad G_3 \quad G_4 \quad \ldots$

— Polynomial Identification —

- Polynomial Time
  - Computation of G_m is done in poly-time in $\sum_{i=1}^{m} |w_i|$
- Polynomial Characteristic Set $K_{G_*} = \{x_1, \ldots, x_k\} \subseteq L(G_*)$:
  - whenever $K_{G_*} \subseteq \{w_1, \ldots, w_m\}$, $\mathcal{A}$ converges to $G_m = G_{n_0}$
  - $|K_{G_*}| = k$ is bounded by a polynomial in $||G_*||$

k, l-Substitutable Languages

- k, l-substitutability: for any strings $x_1, z_1, x_2, z_2, y_1, y_2, v, u \in \Sigma^*$,

$$x_1 v y_1 u z_1, x_1 v y_2 u z_1, x_2 v y_1 u z_2 \in L \implies x_2 v y_2 u z_2 \in L$$

  if $|v| = k$, $|u| = l$ and $v y_1 u, v y_2 u \neq \lambda$.

k, l-Substitutable Languages

- k, l-substitutability: for any strings $x_1, z_1, x_2, z_2, y_1, y_2, v, u \in \Sigma^*$,

$$x_1 v y_1 u z_1, x_1 v y_2 u z_1, x_2 v y_1 u z_2 \in L \quad \Longrightarrow \quad x_2 v y_2 u z_2 \in L$$

  if $|v| = k$, $|u| = l$ and $v y_1 u, v y_2 u \neq \lambda$.

- $0, 0$-substitutability $=$ substitutability by Clark and Eyraud (2007)

k, l-Substitutable Languages

- k, l-substitutability: for any strings $x_1, z_1, x_2, z_2, y_1, y_2, v, u \in \Sigma^*$,

$$x_1 v y_1 u z_1, x_1 v y_2 u z_1, x_2 v y_1 u z_2 \in L \quad \Longrightarrow \quad x_2 v y_2 u z_2 \in L$$

  if $|v| = k$, $|u| = l$ and $v y_1 u, v y_2 u \neq \lambda$.
- $0, 0$-substitutability $=$ substitutability by Clark and Eyraud (2007)
- (k, l)-$\mathcal{SUB} \subseteq (m, n)$-$\mathcal{SUB}$ iff $k \leq m$ and $l \leq n$.

k, l-Substitutable Languages

- k, l-substitutability: for any strings $x_1, z_1, x_2, z_2, y_1, y_2, v, u \in \Sigma^*$,

$$x_1 v y_1 u z_1, x_1 v y_2 u z_1, x_2 v y_1 u z_2 \in L \quad \Longrightarrow \quad x_2 v y_2 u z_2 \in L$$

  if $|v| = k$, $|u| = l$ and $v y_1 u, v y_2 u \neq \lambda$.
- $0, 0$-substitutability $=$ substitutability by Clark and Eyraud (2007)
- (k, l)-$\mathcal{SUB} \subseteq (m, n)$-$\mathcal{SUB}$ iff $k \leq m$ and $l \leq n$.
- k-reversible languages:

$$x_1 v y_1, x_1 v y_2, x_2 v y_1 \in L \quad \Longrightarrow \quad x_2 v y_2 \in L$$

  if $|v| = k$.

3. Learning Algorithm for k, l-Substitutable Context-Free Languages

Learning Target

- Our learning target is the class of languages that are
  - k, l-substitutable and
  - context-free

  for each k, l.
- Hereafter we fix nonnegative integers k and l with $\langle k, l \rangle \neq \langle 0, 0 \rangle$.

Learning Algorithm k, l-SGL

let $\hat{G}$ be a CFG generating the empty language;
for $n = 1, 2, \ldots$ **do**
read the next string w_n;
if $w_n \notin \mathcal{L}(\hat{G})$
then for $K = \{w_1, \ldots, w_n\}$, let $\hat{G} = \langle \Sigma, V_K, P_K, S \rangle$ where

$$V_K = \{ [y] \mid xyz \in K,\ y \neq \lambda \} \cup \{S\},$$

$$\begin{aligned} P_K = \{ S \rightarrow [w] \mid w \in K \} & \\ \cup \{ [xy] \rightarrow [x][y] \mid [xy], [x], [y] \in V_K \} & \\ \cup \{ [a] \rightarrow a \mid a \in \Sigma \} & \\ \cup \{ [vyu] \rightarrow [vy'u] \mid xvyuz, xvy'uz \in K,\ |v| = k,\ |u| = l \}; & \end{aligned}$$

end if
output $\hat{G}$;
end for

Example Run

Let $k = l = 1$ and $K = \{ab, aabb\}$.

Example Run

Let $k = l = 1$ and $K = \{ab, aabb\}$.

$$V_K = \{S, [a], [b], [aa], [ab], [bb], [aab], [abb], [aabb]\}$$

$$\begin{aligned} P_K = \{ & S \to [ab],\ S \to [aabb], \\ & [aabb] \to [aab][b],\ [aabb] \to [aa][bb],\ [aabb] \to [a][abb], \\ & [aab] \to [aa][b],\ [aab] \to [a][ab],\ [abb] \to [ab][b],\ [abb] \to [a][bb], \\ & [aa] \to [a][a],\ [ab] \to [a][b],\ [bb] \to [b][b], \\ & [a] \to a,\ [b] \to b, \\ & [aabb] \to [ab],\ [ab] \to [aabb]\ \} \end{aligned}$$

$$S \Rightarrow [aabb] \Rightarrow [a][abb] \Rightarrow [a][ab][b] \Rightarrow [a][aabb][b] \stackrel{*}{\Rightarrow} aaabbb$$

$$L(\hat{G}) = \{\, a^n b^n \mid n \geq 1 \,\}$$

Convergence

G_*: target grammar generating a k, l-SCFL

$\hat{G}_K$: output on $K \subseteq L(G_*)$

- $L(\hat{G}_K) \subseteq L(G_*)$ always.

Convergence

G_*: target grammar generating a k, l-SCFL

$\hat{G}_K$: output on $K \subseteq L(G_*)$

- $L(\hat{G}_K) \subseteq L(G_*)$ always.
- $L(\hat{G}_K)$ is not necessarily k, l-substitutable.
  - Example: $k = 1$, $l = 0$, $K = \{a, ab, abbc\}$

$$abbc \in L(G_*) \longrightarrow abc \in L(G_*) \longrightarrow ac \in L(G_*)$$

Convergence

G_*: target grammar generating a k, l-SCFL
$\hat{G}_K$: output on $K \subseteq L(G_*)$

- $L(\hat{G}_K) \subseteq L(G_*)$ always.
- $L(\hat{G}_K)$ is not necessarily k, l-substitutable.
  - Example: $k = 1, l = 0, K = \{a, ab, abbc\}$

$$abbc \in L(G_*) \longrightarrow abc \in L(G_*) \longrightarrow ac \in L(G_*)$$

$$S \underset{\hat{G}_K}{\Rightarrow} [abbc] \Rightarrow [ab][bc] \Rightarrow [a][bc] \not\Rightarrow [ab][c] \Rightarrow [a][c]$$

Convergence

G_*: target grammar generating a k, l-SCFL
$\hat{G}_K$: output on $K \subseteq L(G_*)$

- $L(\hat{G}_K) \subseteq L(G_*)$ always.
- $L(\hat{G}_K)$ is not necessarily k, l-substitutable.
  - Example: $k = 1$, $l = 0$, $K = \{a, ab, abbc\}$

$$abbc \in L(G_*) \longrightarrow abc \in L(G_*) \longrightarrow ac \in L(G_*)$$

$$S \underset{\hat{G}_K}{\Rightarrow} [abbc] \Rightarrow [ab][bc] \Rightarrow [a][bc] \not\Rightarrow [ab][c] \Rightarrow [a][c]$$

- We will present a characteristic set K_{G_*} of a k, l-SCFL $L(G_*)$:

$$K_{G_*} \subseteq K \implies L(G_*) \subseteq L(\hat{G}_K).$$

k, l-Greibach Normal Form

- $G = \langle \Sigma, V, P, S \rangle$ is in k, l-GNF if every rule has the form either
  - $A \longrightarrow w$ with $w \in \Sigma^{\leq k+l} - \{\lambda\}$ or
  - $A \longrightarrow x\beta z$ with $x \in \Sigma^k$, $z \in \Sigma^l$, $\beta \in V^+$.
- $1, 0$-GNF $=$ Standard Greibach normal form
- $1, 1$-GNF $=$ Double Greibach normal form

Theorem:
Every CFG has an equivalent CFG in k, l-GNF $G = \langle \Sigma, V, P, S \rangle$ of polynomial size with $P \subseteq V \times (\Sigma^{\leq k+l} \cup \Sigma^k V^{\leq 2} \Sigma^l)$.

k, l-Greibach Normal Form

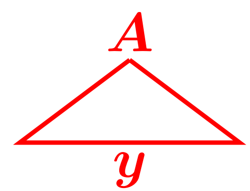
- $G = \langle \Sigma, V, P, S \rangle$ is in k, l-GNF if every rule has the form either
  - $A \longrightarrow w$ with $w \in \Sigma^{\leq k+l} - \{\lambda\}$ or
  - $A \longrightarrow x\beta z$ with $x \in \Sigma^k$, $z \in \Sigma^l$, $\beta \in V^+$.
- $1, 0$-GNF $=$ Standard Greibach normal form
- $1, 1$-GNF $=$ Double Greibach normal form

Theorem:
Every CFG has an equivalent CFG in k, l-GNF $G = \langle \Sigma, V, P, S \rangle$ of polynomial size with $P \subseteq V \times (\Sigma^{\leq k+l} \cup \Sigma^k V^{\leq 2} \Sigma^l)$.

$\longrightarrow$ Learning target is in k, l-GNF.

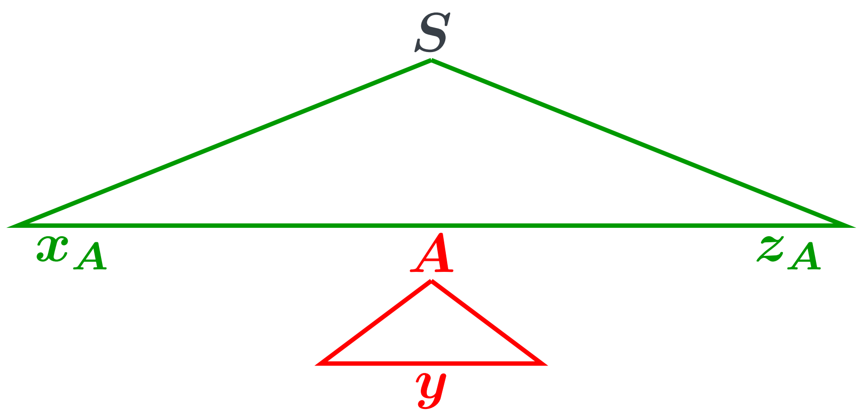
Characteristic Set (1)

Let $G = \langle \Sigma, V, P, S \rangle$ be in k, l-GNF.
For each rule $A \rightarrow y$ with $y \in \Sigma^*$,



Characteristic Set (1)

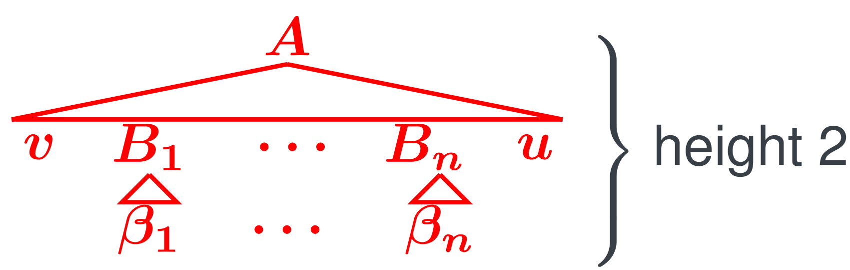
Let $G = \langle \Sigma, V, P, S \rangle$ be in k, l-GNF.
For each rule $A \to y$ with $y \in \Sigma^*$,



$x_A y z_A \in K_G$, where $x_A, z_A \in \Sigma^*$ are shortest.

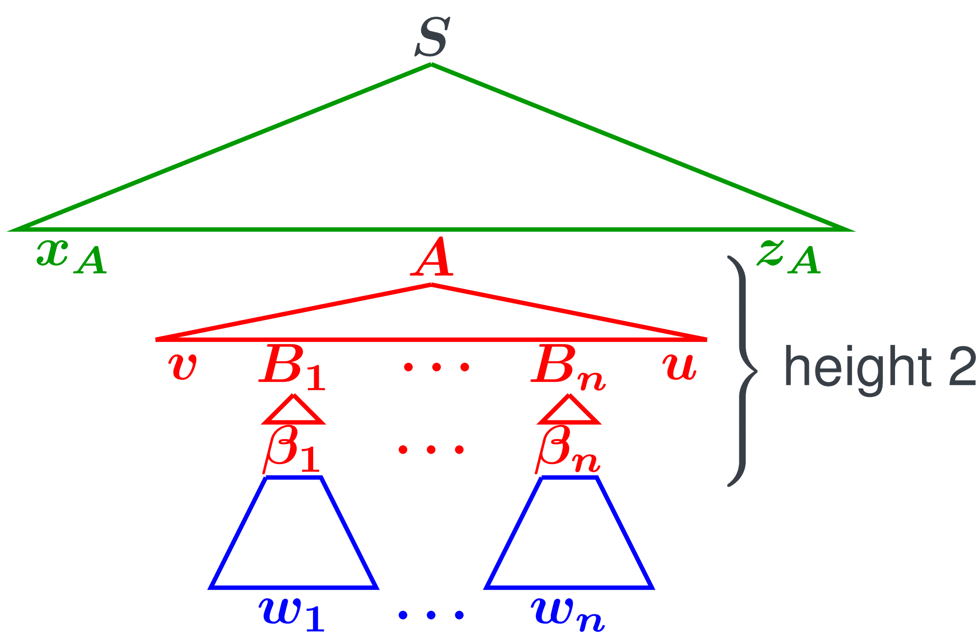
Characteristic Set (2)

Let $G = \langle \Sigma, V, P, S \rangle$ be in k, l-GNF.
For each tuple of rules $A \rightarrow vB_1 \ldots B_n u$ and $B_i \rightarrow \beta_i$ for $i = 1 \ldots n$ which forms a tree of height 2,



Characteristic Set (2)

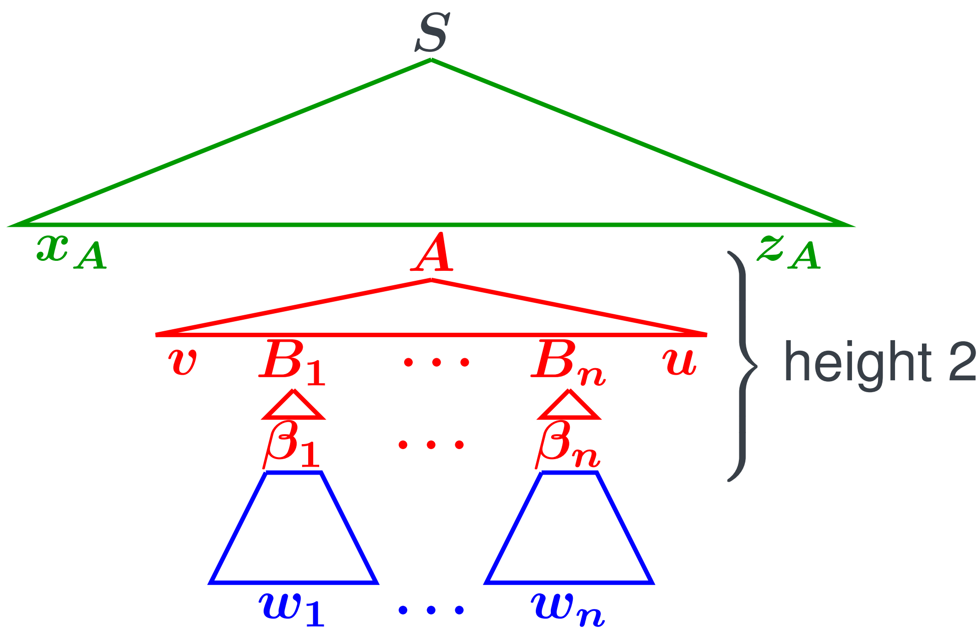
Let $G = \langle \Sigma, V, P, S \rangle$ be in k, l-GNF.
For each tuple of rules $A \rightarrow vB_1 \ldots B_n u$ and $B_i \rightarrow \beta_i$ for $i = 1 \ldots n$ which forms a tree of height 2,



$x_A v w_1 \ldots w_n u z_A \in K_G$, where $x_A, z_A, w_1, \ldots, w_n \in \Sigma^*$ are shortest.

Characteristic Set (2)

Let $G = \langle \Sigma, V, P, S \rangle$ be in k, l-GNF.
For each tuple of rules $A \rightarrow vB_1 \ldots B_n u$ and $B_i \rightarrow \beta_i$ for $i = 1 \ldots n$ which forms a tree of height 2,



$x_A v w_1 \ldots w_n u z_A \in K_G$, where $x_A, z_A, w_1, \ldots, w_n \in \Sigma^*$ are shortest.

- $|K_G| \leq |P|^{n+1}$ where $n = \max\{\, |\beta| \mid A \rightarrow v\beta u \in P \,\} \leq 2$,

Example

$k = l = 1.$

$G: \; S \to aSc, \; S \to b$

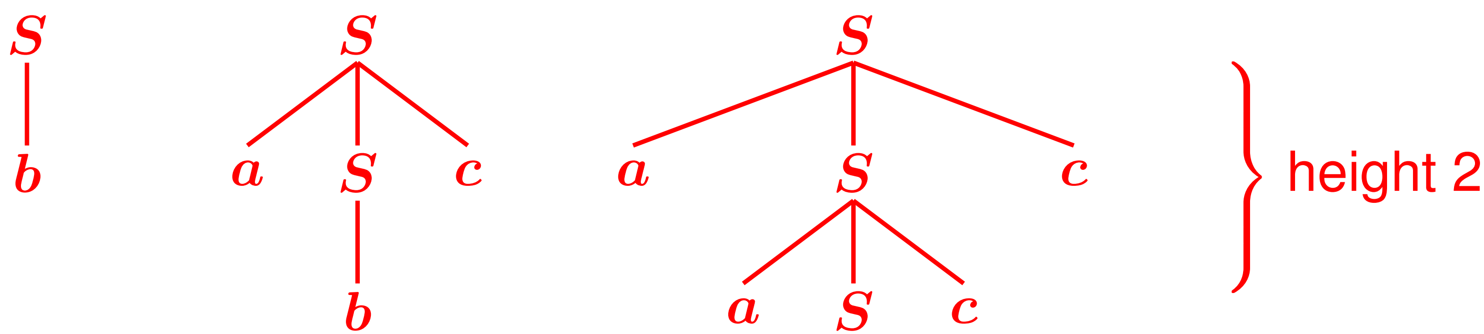
$L(G) = \{ a^n b c^n \mid n \geq 0 \}$

Example

$k = l = 1.$

$G: \; S \to aSc, \; S \to b$

$L(G) = \{ a^n b c^n \mid n \geq 0 \}$

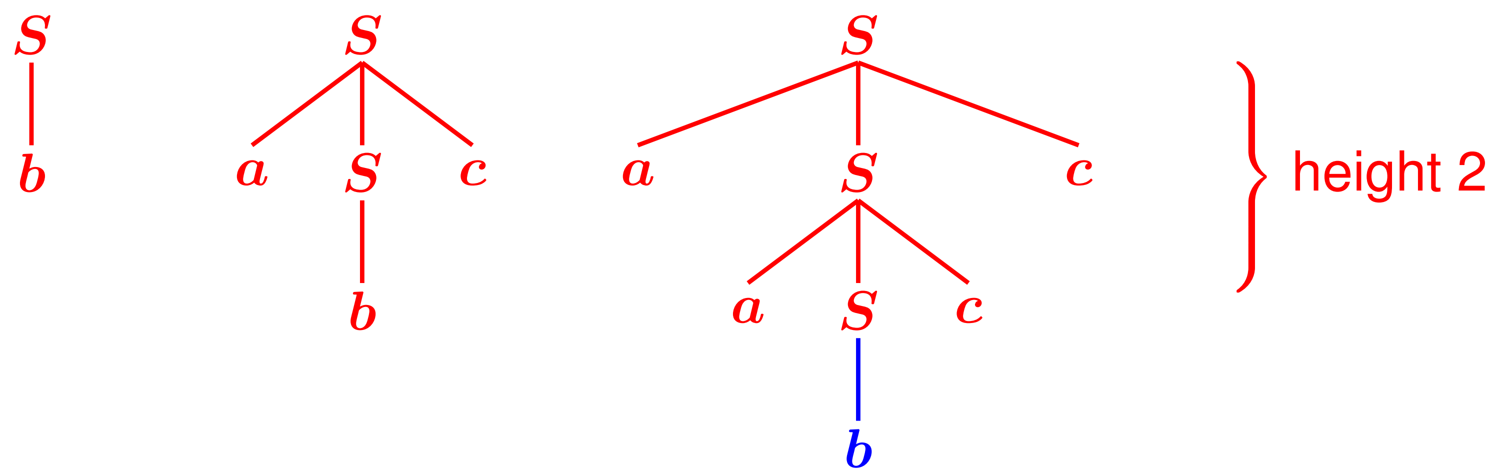


Example

$k = l = 1.$

$G: \; S \to aSc, \; S \to b$

$L(G) = \{ a^n b c^n \mid n \geq 0 \}$



$K_G = \{ b, \; abc, \; aabcc \}$

$$S \underset{\hat{G}_{K_G}}{\Rightarrow} [abc] \Rightarrow [aabcc] \overset{*}{\Rightarrow} [a][abc][c] \Rightarrow [a][aabcc][c] \overset{*}{\Rightarrow} \cdots$$

$aabcc \in K_G$ is necessary, as $\{b, \; abc\}$ is a $1, 1$-SCFL too.

Proof

$$K_A = \{ y \in \Sigma^* \mid A \to y \in P \}$$
$$\cup \{ v\widetilde{\beta}_1 \ldots \widetilde{\beta}_n u \in \Sigma^* \mid A \to vB_1 \ldots B_n u, B_i \to \beta_i \in P \}$$
where $\widetilde{\alpha} = \min\{ w \in \Sigma^* \mid \alpha \overset{*}{\underset{G}{\Rightarrow}} w \}$ for $\alpha \in (\Sigma \cup V)^*$,

$$\left(K_G = \bigcup_{A \in V} x_A K_A z_A \right).$$

Lemma.
Let $\mathcal{A}(K_G) = \hat{G}$. $\forall A \to vB_1 \ldots B_n u \in P$, $\forall w_i \in K_{B_i}$, $\exists w \in K_A$:

$$[w] \overset{*}{\underset{\hat{G}}{\Rightarrow}} v[w_1] \ldots [w_n]u.$$

Proof

Lemma.
Let $\mathcal{A}(K_G) = \hat{G}$. $\forall A \longrightarrow vB_1 \ldots B_n u \in P$, $\forall w_i \in K_{B_i}$, $\exists w \in K_A$:

$$[w] \overset{*}{\underset{\hat{G}}{\Rightarrow}} v[w_1] \ldots [w_n]u.$$

Proof. Let β_i be such that $B_i \Rightarrow_G \beta_i \overset{*}{\Rightarrow} w_i$. $v\widetilde{\beta}_1 \ldots \widetilde{\beta}_n u \in K_A$. Let

$$J = \{\, i \mid w_i \neq \widetilde{\beta}_i \,\}.$$

For each $i \in J$, we have $\widetilde{\beta}_i = v_i y'_i u_i$ and $w_i = v_i y_i u_i$ for some $v_i \in \Sigma^k$, $u_i \in \Sigma^l$ and $y_i, y'_i \in \Sigma^*$. Then $x_i v_i y'_i u_i z_i, x_i v_i y_i u_i z_i \in K_G$ for minimal x_i, z_i such that $S \Rightarrow^*_G x_i B_i z_i$. $[v_i y'_i u_i] \longrightarrow [v_i y_i u_i] \in P_K$. Thus

$$[v\widetilde{\beta}_1 \ldots \widetilde{\beta}_n u] \overset{*}{\underset{\hat{G}}{\Rightarrow}} v[\widetilde{\beta}_1] \ldots [\widetilde{\beta}_n]u \overset{*}{\Rightarrow} v[w_1] \ldots [w_n]u$$

QED.

Polynomial Identifiability of k, l-SCFLs

Proposition:
For any CFG G_* generating a k, l-substitutable languages,
and any input K such that $K_{G_*} \subseteq K$,
we have $L(\hat{G}_K) = L(G_*)$ for $\hat{G}_K$ the output by the algorithm on K.

Polynomial Identifiability of k, l-SCFLs

Proposition:
For any CFG G_* generating a k, l-substitutable languages,
and any input K such that $K_{G_*} \subseteq K$,
we have $L(\hat{G}_K) = L(G_*)$ for $\hat{G}_K$ the output by the algorithm on K.

- Computation of $\hat{G}_K$: Poly-time in $\|K\| = \sum_{w \in K} |w|$,
- $|K_{G_*}| \leq |P|^3$,

4. Summary and Future Work

Summary and Future Work

Summary

- k, l-substitutable CFLs $\Longleftrightarrow$ k-reversible regular languages
- Identifiable in the limit from positive data with polynomial-time and data

Summary and Future Work

Summary

- k, l-substitutable CFLs $\Longleftrightarrow$ k-reversible regular languages
- Identifiable in the limit from positive data with polynomial-time and data

Future Work

- Grammatical characterization of k, l-SCFLs (k, l-GNF ?)
- Results on k-reversible regular languages
  $\Longrightarrow$ Analogous results on k, l-SCFLs
  - k-reversible closure of a finite language is always regular, but k, l-substitutable closure of a finite language can be non-context-free.

Thank You