Polynomial Identification in the Limit of Substitutable Context-free Languages

Alexander Clark and Rémi Eyraud

Review: Substitutability

Definition: Context

The context of a substring u of a string s is the pair of strings l, r such that $lur = s$.
Example: the context of a in abc is (λ, bc)
$C_L(u)$ is the set of contexts for string u in all strings in L.
Example: If $L = \{abc, cab\}$, $C_L(a) = \{(\lambda, bc), (c, b)\}$

Definition: Substitutable language

A substitutable language L is a set of strings such that any two strings that share one context share all contexts.

Review: Substitutability

Definition: Weak substitutability

Two strings u and v are weakly substitutable with respect to language L (written $u \doteq_L v$) if there exist $l, r \in \Sigma^*$ such that $lur \in L$ and $lvr \in L$.

Definition: Syntactic congruence

Two strings u and v are syntactically congruent with respect to language L (written $u \equiv_L v$) iff $\forall l, r \in \Sigma^*$ $lur \in L$ iff $lvr \in L$

Review: Substitutability

Definition: Substitutable language (redux)

A language L is substitutable iff $\forall u, v \in L$, $u \doteq_L v \implies u \equiv_L v$

Review: Identifying Substitutable Languages

Is $L = \{a, aa\}$ substitutable?

Review: Identifying Substitutable Languages

Is $L = \{a, aa\}$ substitutable?

a and aa both share the context (λ, λ), so they must share all contexts for L to be substitutable.

Review: Identifying Substitutable Languages

Is $L = \{a, aa\}$ substitutable?

a and aa both share the context (λ, λ), so they must share all contexts for L to be substitutable.

a also has the contexts (λ, a) and (a, λ), which aa does not have.

Review: Identifying Substitutable Languages

Is $L = \{a, aa\}$ substitutable?

a and aa both share the context (λ, λ), so they must share all contexts for L to be substitutable.

a also has the contexts (λ, a) and (a, λ), which aa does not have.

L is not substitutable.

Review: Identifying Substitutable Languages

Is $L = a^+$ substitutable?

Review: Identifying Substitutable Languages

Is $L = a^+$ substitutable?

Every pair of strings $u = a^{i \geq 1}$ and $v = a^{j \geq 1}$ appears in the context (λ, λ), so u and v must share all contexts for L to be substitutable.

Review: Identifying Substitutable Languages

Is $L = a^+$ substitutable?

Every pair of strings $u = a^{i \geq 1}$ and $v = a^{j \geq 1}$ appears in the context (λ, λ), so u and v must share all contexts for L to be substitutable.

For all $n \geq 0$, a^n will appear to the left and to the right of any u and v in L.

Review: Identifying Substitutable Languages

Is $L = a^+$ substitutable?

Every pair of strings $u = a^{i \geq 1}$ and $v = a^{j \geq 1}$ appears in the context (λ, λ), so u and v must share all contexts for L to be substitutable.

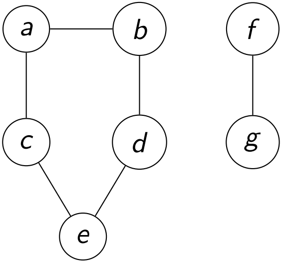
For all $n \geq 0$, a^n will appear to the left and to the right of any u and v in L.

L is substitutable.

Review: Components

Definition: Component

A set of vertices that are connected without any edges to vertices outside the set.



$$\lceil a \rceil = \lceil b \rceil = \lceil e \rceil \neq \lceil f \rceil = \lceil g \rceil$$

Contribution

This paper provides an algorithm for learning a subclass of context-free grammars.

The algorithm learns from the distributions of strings without needing to learn constituency.

It learns in Gold's IIL paradigm from only positive data in polynomial time.

This paper shows that a few assumptions allow CFGs to be learned if they have syntactic congruence.

Substitution graphs

Definition: Substitution graph

Given sample set of strings S, substitution graph $SG(S) = (V, E)$, where:
$V = \{u \in \Sigma^+ : \exists l, r \in \Sigma^*, lur \in S\}$
$E = \{(u, v) \in \Sigma^+ \times \Sigma^+ : u \doteq_S v\}$

A substitution graph is a data structure representing the substitutability of each string in the language.

The vertices are all of the substrings in the sample and the edges connect each pair of strings that are weakly substitutable w.r.t. S.

Each string can be replaced with another string in the same component without affecting language membership.

Substitution graphs

What does the substitution graph of $S = \{a, aa\}$ look like?

Substitution graphs

What does the substitution graph of $S = \{a, aa\}$ look like?

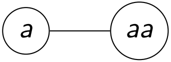
$Sub(S) = \{a, aa\}$
$a \doteq_S aa$

Substitution graphs

What does the substitution graph of $S = \{a, aa\}$ look like?

$Sub(S) = \{a, aa\}$
$a \doteq_S aa$

$V = \{a, aa\}$
$E = \{(a, aa)\}$



Substitution graphs

What does the substitution graph of $S = \{a, ab, abb\}$ look like?

Substitution graphs

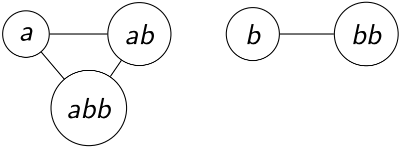
What does the substitution graph of $S = \{a, ab, abb\}$ look like?

$Sub(S) = \{a, b, ab, bb, abb\}$
$C_S(a) = \{(\lambda, \lambda), (\lambda, b), (\lambda, bb)\}$
$C_S(b) = \{(a, \lambda), (a, b), (ab, \lambda)\}$
$C_S(ab) = \{(\lambda, \lambda), (\lambda, b)\}$
$C_S(bb) = \{(a, \lambda)\}$
$C_S(abb) = \{(\lambda, \lambda)\}$

Substitution graphs

What does the substitution graph of $S = \{a, ab, abb\}$ look like?

$Sub(S) = \{a, b, ab, bb, abb\}$
$C_S(a) = \{(\lambda, \lambda), (\lambda, b), (\lambda, bb)\}$
$C_S(b) = \{(a, \lambda), (a, b), (ab, \lambda)\}$
$C_S(ab) = \{(\lambda, \lambda), (\lambda, b)\}$
$C_S(bb) = \{(a, \lambda)\}$
$C_S(abb) = \{(\lambda, \lambda)\}$



Constructing the grammar

Definition: Context-free grammar

A context-free grammar G is a tuple $\langle \Sigma, V, P, S \rangle$ where
Σ is an alphabet
V is a finite set of non-terminal variables in our production rules
$P \subseteq V \times (\Sigma \cup V)^{+}$ is a set of production rules
$S \in V$ is the starting symbol

Constructing the grammar

Algorithm 1 Learn a grammar G from a substitution graph SG

Let Σ be the alphabet used in SG
Compute the set of components $\hat{V}$
Store map $V \to \hat{V}$ where $u \mapsto \lceil u \rceil$
Let $\hat{S}$ be the component corresponding to the context (λ, λ)
$\hat{P} = \{\}$
for $u \in V$ **do**
  if $|u| > 1$ **then**
    for v, w s.t. $u = vw$ **do**
      $\hat{P} \leftarrow \hat{P} \cup (\lceil u \rceil \to \lceil v \rceil \lceil w \rceil)$
    end for
  else
    $\hat{P} \leftarrow \hat{P} \cup (\lceil u \rceil \to u)$
  end if
end for
output $\hat{G} = \langle \Sigma, \hat{V}, \hat{P}, \hat{S} \rangle$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

Let Σ be the alphabet SG | $\Sigma = \{a\}$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

Let Σ be the alphabet SG — $\Sigma = \{a\}$

Compute the set of components $\hat{V}$ — $\hat{V} = \{\lceil a \rceil, \lceil aa \rceil\}$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

Let Σ be the alphabet SG	$\Sigma = \{a\}$
Compute the set of components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil aa \rceil\}$
Store map $V \to \hat{V}$ where $u \mapsto \lceil u \rceil$	$a \mapsto \lceil a \rceil, aa \mapsto \lceil aa \rceil$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

Let Σ be the alphabet SG	$\Sigma = \{a\}$
Compute the set of components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil aa \rceil\}$
Store map $V \to \hat{V}$ where $u \mapsto \lceil u \rceil$	$a \mapsto \lceil a \rceil, aa \mapsto \lceil aa \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil aa \rceil$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

Let Σ be the alphabet SG	$\Sigma = \{a\}$
Compute the set of components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil aa \rceil\}$
Store map $V \to \hat{V}$ where $u \mapsto \lceil u \rceil$	$a \mapsto \lceil a \rceil, aa \mapsto \lceil aa \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil aa \rceil$
$u = a$	$\lvert a \rvert \not> 1$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$	
Let Σ be the alphabet SG	$\Sigma = \{a\}$
Compute the set of components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil aa \rceil\}$
Store map $V \to \hat{V}$ where $u \mapsto \lceil u \rceil$	$a \mapsto \lceil a \rceil, aa \mapsto \lceil aa \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil aa \rceil$
$u = a$	$\|a\| \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	$\hat{P} = \{\lceil a \rceil \to a\}$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

Let Σ be the alphabet SG	$\Sigma = \{a\}$
Compute the set of components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil aa \rceil\}$
Store map $V \to \hat{V}$ where $u \mapsto \lceil u \rceil$	$a \mapsto \lceil a \rceil, aa \mapsto \lceil aa \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil aa \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	$\hat{P} = \{\lceil a \rceil \to a\}$
$u = aa$	$\lvert aa \rvert > 1$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$	
Let Σ be the alphabet SG	$\Sigma = \{a\}$
Compute the set of components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil aa \rceil\}$
Store map $V \to \hat{V}$ where $u \mapsto \lceil u \rceil$	$a \mapsto \lceil a \rceil, aa \mapsto \lceil aa \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil aa \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	$\hat{P} = \{\lceil a \rceil \to a\}$
$u = aa$	$\lvert aa \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil aa \rceil \to \lceil a \rceil \lceil a \rceil)$	$\hat{P} = \{\lceil a \rceil \to a, \lceil aa \rceil \to \lceil a \rceil \lceil a \rceil\}$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

Let Σ be the alphabet SG	$\Sigma = \{a\}$
Compute the set of components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil aa \rceil\}$
Store map $V \to \hat{V}$ where $u \mapsto \lceil u \rceil$	$a \mapsto \lceil a \rceil, aa \mapsto \lceil aa \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil aa \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	$\hat{P} = \{\lceil a \rceil \to a\}$
$u = aa$	$\lvert aa \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil aa \rceil \to \lceil a \rceil \lceil a \rceil)$	$\hat{P} = \{\lceil a \rceil \to a, \lceil aa \rceil \to \lceil a \rceil \lceil a \rceil\}$
output $\hat{G} = \langle \Sigma, \hat{V}, \hat{P}, \hat{S} \rangle$	

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

$G = \langle$

$\Sigma = \{a\},$

$\hat{V} = \{\hat{S}\},$

$\hat{P} = \{\lceil a \rceil \to a, \lceil aa \rceil \to \lceil a \rceil \lceil a \rceil\}$

$\hat{S} = \lceil a \rceil = \lceil aa \rceil$

$\rangle$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

$G = \langle$

$\Sigma = \{a\}$,

$\hat{V} = \{\hat{S}\}$,

$\hat{P} = \{\lceil a \rceil \rightarrow a, \lceil aa \rceil \rightarrow \lceil a \rceil \lceil a \rceil\}$

$\hat{S} = \lceil a \rceil = \lceil aa \rceil$

$\rangle$

What language does this grammar produce?

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

$G = \langle$

$\Sigma = \{a\},$

$\hat{V} = \{\hat{S}\},$

$\hat{P} = \{\lceil a \rceil \to a, \lceil aa \rceil \to \lceil a \rceil \lceil a \rceil\}$

$\hat{S} = \lceil a \rceil = \lceil aa \rceil$

$\rangle$

What language does this grammar produce?

$\hat{S} \to a$

$\hat{S} \to \hat{S}\hat{S}$

Constructing the grammar

Learning a grammar from $S = \{a, aa\}$

$G = \langle$

$\Sigma = \{a\}$,

$\hat{V} = \{\hat{S}\}$,

$\hat{P} = \{\lceil a \rceil \rightarrow a, \lceil aa \rceil \rightarrow \lceil a \rceil \lceil a \rceil\}$

$\hat{S} = \lceil a \rceil = \lceil aa \rceil$

$\rangle$

What language does this grammar produce?

$\hat{S} \rightarrow a$

$\hat{S} \rightarrow \hat{S}\hat{S}$

$L = a^+$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet | $\Sigma = \{a, b\}$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \rightarrow \hat{V}$	$a \mapsto \lceil a \rceil, \ldots, abb \mapsto \lceil abb \rceil$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \rightarrow \hat{V}$	$a \mapsto \lceil a \rceil, ..., abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, \ldots, abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\|a\| \not> 1$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, \ldots, abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \rightarrow \hat{V}$	$a \mapsto \lceil a \rceil, ..., abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \rightarrow a)$	
$u = b$	$\lvert b \rvert \not> 1$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, \ldots, abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	
$u = b$	$\lvert b \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil b \rceil \to b)$	

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, \ldots, abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	
$u = b$	$\lvert b \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil b \rceil \to b)$	
$u = ab$	$\lvert ab \rvert > 1$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, \ldots, abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	
$u = b$	$\lvert b \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil b \rceil \to b)$	
$u = ab$	$\lvert ab \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil ab \rceil \to \lceil a \rceil \lceil b \rceil)$	

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, ..., abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	
$u = b$	$\lvert b \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil b \rceil \to b)$	
$u = ab$	$\lvert ab \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil ab \rceil \to \lceil a \rceil \lceil b \rceil)$	
$u = bb$	$\lvert bb \rvert > 1$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, \ldots, abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	
$u = b$	$\lvert b \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil b \rceil \to b)$	
$u = ab$	$\lvert ab \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil ab \rceil \to \lceil a \rceil \lceil b \rceil)$	
$u = bb$	$\lvert bb \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil bb \rceil \to \lceil b \rceil \lceil b \rceil)$	

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, ..., abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	
$u = b$	$\lvert b \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil b \rceil \to b)$	
$u = ab$	$\lvert ab \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil ab \rceil \to \lceil a \rceil \lceil b \rceil)$	
$u = bb$	$\lvert bb \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil bb \rceil \to \lceil b \rceil \lceil b \rceil)$	
$u = abb$	$\lvert abb \rvert > 1$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

Let Σ be the alphabet	$\Sigma = \{a, b\}$
Compute components $\hat{V}$	$\hat{V} = \{\lceil a \rceil, \lceil b \rceil, \lceil ab \rceil, \lceil bb \rceil, \lceil abb \rceil\}$
Store map $V \to \hat{V}$	$a \mapsto \lceil a \rceil, ..., abb \mapsto \lceil abb \rceil$
Set $\hat{S}$ to the component for (λ, λ)	$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$
$u = a$	$\lvert a \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil a \rceil \to a)$	
$u = b$	$\lvert b \rvert \not> 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil b \rceil \to b)$	
$u = ab$	$\lvert ab \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil ab \rceil \to \lceil a \rceil \lceil b \rceil)$	
$u = bb$	$\lvert bb \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil bb \rceil \to \lceil b \rceil \lceil b \rceil)$	
$u = abb$	$\lvert abb \rvert > 1$
$\hat{P} \leftarrow \hat{P} \cup (\lceil abb \rceil \to \lceil a \rceil \lceil bb \rceil)$	
$\hat{P} \leftarrow \hat{P} \cup (\lceil abb \rceil \to \lceil ab \rceil \lceil b \rceil)$	

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

$$
\begin{aligned}
G = \langle & \\
& \Sigma = \{a, b\}, \\
& \hat{V} = \{\hat{S}, \hat{B}\}, \\
& \hat{P} = \{\lceil a \rceil \to a, \\
& \qquad \lceil b \rceil \to b, \\
& \qquad \lceil ab \rceil \to \lceil a \rceil \lceil b \rceil, \\
& \qquad \lceil bb \rceil \to \lceil b \rceil \lceil b \rceil, \\
& \qquad \lceil abb \rceil \to \lceil a \rceil \lceil bb \rceil, \\
& \qquad \lceil abb \rceil \to \lceil ab \rceil \lceil b \rceil\}, \\
& \hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil \\
\rangle &
\end{aligned}
$$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

$$G = \langle$$
$$\Sigma = \{a, b\},$$
$$\hat{V} = \{\hat{S}, \hat{B}\},$$
$$\begin{aligned}\hat{P} = \{ \lceil a \rceil &\rightarrow a, \\ \lceil b \rceil &\rightarrow b, \\ \lceil ab \rceil &\rightarrow \lceil a \rceil \lceil b \rceil, \\ \lceil bb \rceil &\rightarrow \lceil b \rceil \lceil b \rceil, \\ \lceil abb \rceil &\rightarrow \lceil a \rceil \lceil bb \rceil, \\ \lceil abb \rceil &\rightarrow \lceil ab \rceil \lceil b \rceil \},\end{aligned}$$
$$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$$
$$\rangle$$

What language does this grammar produce?

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

$G = \langle$

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

$\hat{V} = \{\hat{S}, \hat{B}\}$,

$\hat{P} = \{\lceil a \rceil \rightarrow a$,

$\lceil b \rceil \rightarrow b$,

$\lceil ab \rceil \rightarrow \lceil a \rceil \lceil b \rceil$,

$\lceil bb \rceil \rightarrow \lceil b \rceil \lceil b \rceil$,

$\lceil abb \rceil \rightarrow \lceil a \rceil \lceil bb \rceil$,

$\lceil abb \rceil \rightarrow \lceil ab \rceil \lceil b \rceil\}$,

$\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil$

$\rangle$

What language does this grammar produce?

$\hat{S} \rightarrow a, \hat{B} \rightarrow b, \hat{S} \rightarrow \hat{S}\hat{B}, \hat{B} \rightarrow \hat{B}\hat{B}$

Constructing the grammar

Learning a grammar from $S = \{a, ab, abb\}$

$$
\begin{aligned}
G = \langle \\
&\Sigma = \{a, b\}, \\
&\hat{V} = \{\hat{S}, \hat{B}\}, \\
&\hat{P} = \{\lceil a \rceil \to a, \\
&\qquad \lceil b \rceil \to b, \\
&\qquad \lceil ab \rceil \to \lceil a \rceil \lceil b \rceil, \\
&\qquad \lceil bb \rceil \to \lceil b \rceil \lceil b \rceil, \\
&\qquad \lceil abb \rceil \to \lceil a \rceil \lceil bb \rceil, \\
&\qquad \lceil abb \rceil \to \lceil ab \rceil \lceil b \rceil\}, \\
&\hat{S} = \lceil a \rceil = \lceil ab \rceil = \lceil abb \rceil \\
\rangle
\end{aligned}
$$

What language does this grammar produce?

$\hat{S} \to a, \hat{B} \to b, \hat{S} \to \hat{S}\hat{B}, \hat{B} \to \hat{B}\hat{B}$

$L = ab^*$

Constructing the grammar

The language $L = \{a^n cb^n\}$ can be represented by the rules $\{S \rightarrow aSb, S \rightarrow c\}$.
How can we learn it with this algorithm and a large representative sample?

Constructing the grammar

The language $L = \{a^n c b^n\}$ can be represented by the rules $\{S \rightarrow aSb, S \rightarrow c\}$.

How can we learn it with this algorithm and a large representative sample?

The words share the empty context and contexts with equal amounts of a and b on each side.

The other substrings don't share any contexts, so each will be in a separate component.

The rules learned from all the components are:

$C_{j+k} \rightarrow C_j B_k$ for all $k > 0, j \in \mathbb{Z}$

$C_{j-k} \rightarrow A_k C_j$ for all $k > 0, j \in \mathbb{Z}$

$C_0 \rightarrow c$

$A_{i+j} \rightarrow A_i A_j$ for all $i, j > 0$

$A_1 \rightarrow a$

$B_{i+j} \rightarrow B_i B_j$ for all $i, j > 0$

$B_1 \rightarrow b$

Global algorithm

Algorithm 2 Learn sequence of CFGs $G_1, G_2, \ldots$ from sample S

$G =$ grammar generating the empty language
while *true* **do**
  read next string s_n in S
  if $s_n \notin L(G)$ **then**
    create substitution graph SG from $\{s_1, \ldots, s_n\}$
    create grammar G from SG ▷ Algorithm 1
  end if
  output G
end while

Efficiency

We have sample $S = \{w_1, ..., w_n\}$.
Each sample w_i has $\frac{|w_i|^2 + |w_i|}{2}$ non-empty substrings.
Let $N = \sum |w_i|$ and $L = max|w_i|$.
N^2 is an upper bound on $|V|$.
Finding weakly substitutable pairs of substrings can be done in time less than L^2.
Computing edges can be done in time less than $L^2 n^2$.
Finding the components can be done in $|V| + |E|$ time.
The total number of rules is bound by LN^2 and each is constructed in constant time.
The global algorithm runs the SG learning algorithm at most n times.

Conclusion

This paper introduces an algorithm that runs in time efficient with respect to the size of the input.

CFGs for substitutable languages are learned through the distributions of terminals instead of through constituency.

This algorithm is inadequate for learning natural language, but it demonstrates that the assumption that children need examples of a specific construction to produce it is false.