1

Membership Queries and Context-Free Languages

(Based on Joint Work with Ryo Yoshinaka)

Makoto Kanazawa, Hosei University

2

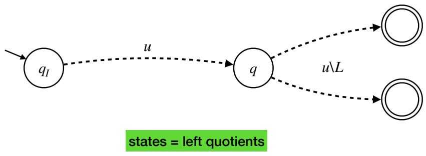
regular language	CFL
DFA	CFG
states	nonterminals
left quotients	???

This talk is about a new way of classifying context-free languages motivated by learning from (positive data and) membership queries. There are several different answers you can put in place of “???” in this chart, and this gives rise to several distinct classes of context-free languages.

3

Regular Languages

- Every regular language has a canonical **minimal DFA**.



- **Left quotient** of a language $L \subseteq \Sigma^*$ by a string $u \in \Sigma^*$:

$$u\backslash L = \{ x \in \Sigma^* \mid ux \in L \}$$

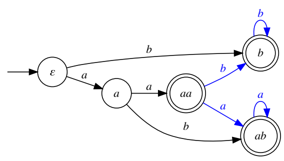
- A language L is regular if and only if $\{ u\backslash L \mid u \in \Sigma^* \}$ is finite.

In the case of regular languages, everything is very clear-cut and well-understood.
There is a canonical minimal DFA for every regular language *L*.
The states of this DFA correspond to the (nonempty) left quotients of *L*.
If a string *u* takes you from the initial state to a state *q*, then the “future” of *q* (the set of strings that take you from *q* to some final state) is the left quotient of *L* by *u*.
(If *L* is represented by a regular expression r, then the regular expression for *u* \ *L* is sometimes called the derivative of *r* by *u*.)

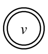
4

Example

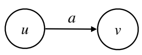
$L = aba^* \cup bb^* \cup aa(a^* \cup b^*)$



$\varepsilon\backslash L = L$
$a\backslash L = ba^* \cup a(a^* \cup b^*)$
$b\backslash L = b^*$
$aa\backslash L = a^* \cup b^*$
$ab\backslash L = a^*$



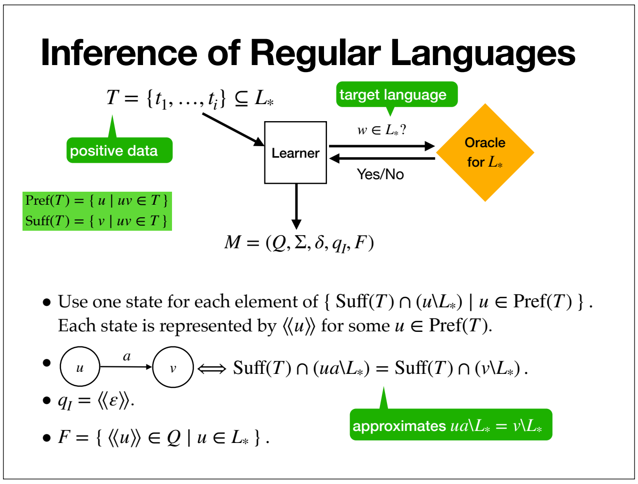
$\iff v \in L$
$\iff \varepsilon \in v\backslash L$
$\iff (v\backslash L) \cap \{\varepsilon\} = \{\varepsilon\}$



$\iff ua\backslash L = v\backslash L$
$\iff (u\backslash L) \cap a\Sigma^* = a(v\backslash L)$

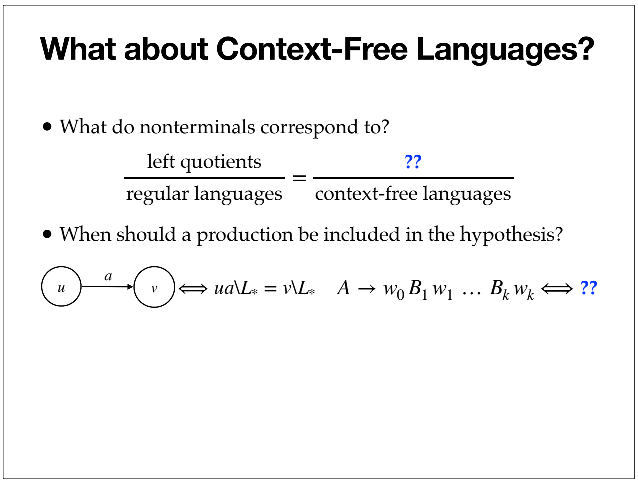
$bb\backslash L = b^* = b\backslash L$
$aaa\backslash L = a^* = ab\backslash L$
$aab\backslash L = b^* = b\backslash L$
$aba\backslash L = a^* = ab\backslash L$

The minimal DFA for *L* has five states, each corresponding to a distinct left quotient.
A state is final iff *ε* (the empty string) is in the left quotient associated with that state.
There is an arrow labeled *a* from a state corresponding to a left quotient *u* \ *L* and a state corresponding to *ua* \ *L*.
There is one more left quotient, namely the empty set, which would correspond to a dead state. We consider minimal DFAs without dead states.



5

In learning from positive data and membership queries, the learner receives positive examples one by one, and each time makes a polynomial number of membership queries before outputting a hypothesis.
When the target language L_* is regular, the task of the learner is to find the minimal DFA for L_*.
A left quotient is approximated by a subset of Suff(T), the set of suffixes of the strings in T.
Equations involving left quotients are approximated using these finite sets, and decided by making appropriate membership queries.



6

What would a similar picture be in the case of context-free languages?
Two questions have to be answered.
Let's focus on the second question first. We can avoid answering the first question by restricting ourselves to CFGs with just one nonterminal.

Easy Case: Grammars with Just One Nonterminal

$S \to \varepsilon$
$S \to aSbS$

Dyck language: $D_1 = \{ x \in \{a,b\}^* \mid |x|_a = |x|_b \wedge \forall uv(uv = x \to |u|_a \geq |u|_b) \}$

($|x|_a$: number of occurrences of a in x)

$$\pi: \quad S \to w_0 S w_1 \dots S w_k \qquad (w_i \in \Sigma^*)$$

When should π be in the hypothesized grammar?

$$\pi \text{ is } \textbf{valid} \overset{\text{def}}{\iff} L_* \supseteq w_0 L_* w_1 \dots L_* w_k$$

Why is this reasonable?

approximated by $L_* \supseteq w_0 (\mathrm{Sub}(T) \cap L_*) w_1 \dots (\mathrm{Sub}(T) \cap L_*) w_k$

$\mathrm{Sub}(T) = \{ x \mid uxv \in T \}$

7

Let's bypass the problem of how to deal with nonterminals and consider the special class of CFGs whose start symbol is the only nonterminal. An example of such a CFG is a grammar for the Dyck language (over a single pair of parentheses). Suppose that the learner is contemplating a candidate production $S \to w_0\ S\ w_1 \dots S\ w_k$. It should be included in the hypothesis grammar when it is *valid*.

$$\pi: \quad S \to w_0 S w_1 \dots S w_k \qquad (w_i \in \Sigma^*)$$

$$\pi \text{ is } \textbf{valid} \overset{\text{def}}{\iff} L_* \supseteq w_0 L_* w_1 \dots L_* w_k$$

- If π is not valid, π can't be in a correct grammar for L_*.

  If $x_1, \dots, x_k \in L_*$, $w_0 x_1 w_1 \dots x_k w_k \notin L_*$, and π is in G, then $L_* \not\subseteq L(G)$.

$$S \Rightarrow w_0 S w_1 \dots S w_k$$
$$\Rightarrow^* w_0 x_1 w_1 \dots x_k w_k$$

- If all productions in G are valid, then $L(G) \subseteq L_*$.

8

In order to understand the second bullet point, it is useful to recall some basic facts about context-free grammars in general.

Context-Free Grammars as Monotone Operators

9

$$
\begin{aligned}
S &\to aD_1bS \mid aA \mid bU\\
D_1 &\to \varepsilon \mid aD_1bD_1\\
A &\to \varepsilon \mid aD_1bA \mid aA\\
U &\to \varepsilon \mid Ua \mid Ub
\end{aligned}
$$

abbreviates three productions: $S \to aD_1bS,\ S \to aA,\ S \to bU$

$\overline{D_1} = \{\, x \in \{a,b\}^* \mid |x|_a \neq |x|_b \vee \exists uv(uv = x \wedge |u|_a < |u|_b) \,\}$

$L_G(B) = \{\, x \in \Sigma^* \mid B \Rightarrow_G^* x \,\}$

$(L_G(S), L_G(D_1), L_G(A), L_G(U))$ is the **least fixed point** of the operator $\Phi_G\colon (\mathscr{P}(\{a,b\}^*))^4 \to (\mathscr{P}(\{a,b\}^*))^4$:

$$
\Phi_G \begin{bmatrix} X_S \\ X_{D_1} \\ X_A \\ X_U \end{bmatrix} = \begin{bmatrix} aX_{D_1}bX_S \cup aX_A \cup bX_U \\ \varepsilon \cup aX_{D_1}bX_{D_1} \\ \varepsilon \cup aX_{D_1}bX_A \cup aX_A \\ \varepsilon \cup X_Ua \cup X_Ub \end{bmatrix}
$$

Let's look at context-free grammars with more than one nonterminal, for example, this grammar for the complement of the Dyck language.
Productions with the same left-hand side nonterminal are often collected together.
A CFG is associated with an operator on tuples of string sets.
Nonterminals are interpreted as sets, and the vertical bar is interpreted as union.
The languages of the nonterminals are the components of the least fixed point of this operator.

Pre-fixed Points of Context-Free Grammars

10

$G = (N, \Sigma, P, S)$

Φ_G: associated operator

- $(X_B)_{B\in N}$ is a **pre-fixed point** of $\Phi_G \overset{\text{def}}{\iff} \Phi_G((X_B)_{B\in N}) \subseteq (X_B)_{B\in N}$ (componentwise inclusion)
- $(L_G(B))_{B\in N}$ is the least pre-fixed point of Φ_G.
- $(X_B)_{B\in N}$ is a pre-fixed point of Φ_G if and only if for every production $A \to w_0\, B_1\, w_1\, \ldots\, B_k\, w_k$ in P,

$$X_A \supseteq w_0\, X_{B_1}\, w_1\, \ldots\, X_{B_k}\, w_k.$$

- If G has a pre-fixed point $(X_B)_{B\in N}$ with $X_S = L_*$, then $L(G) \subseteq L_*$.

The validity of the productions corresponds to *pre-fixed* points.
Least fixed points coincide with least pre-fixed points.
The advantage of pre-fixed points is that you can look at individual productions in isolation.

11

Easy Case: Grammars with Just One Nonterminal

$$\pi: \quad S \to w_0\, S\, w_1\, \dots\, S\, w_k \qquad (w_i \in \Sigma^*)$$

$$\pi \text{ is } \textbf{valid} \overset{\text{def}}{\iff} L_* \supseteq w_0\, L_*\, w_1\, \dots\, L_*\, w_k$$

- All productions of G are valid $\iff L_*$ is a pre-fixed point of Φ_G.
- If all productions in G are valid, then $L(G) \subseteq L_*$.
  $\because L_*$ is a pre-fixed point of G.
- If $L(G) = L_*$, all productions in G are valid.
  $\because L_* = L(G)$ is the least pre-fixed point of G_*.

12

Validity in the General Case

$$\pi: \quad A \to w_0\, B_1\, w_1\, \dots\, B_k\, w_k$$

$$\pi \text{ is } \textbf{valid} \overset{\text{def}}{\iff} [\![A]\!]^{L_*} \supseteq w_0\, [\![B_1]\!]^{L_*}\, w_1\, \dots\, [\![B_k]\!]^{L_*}\, w_k$$

- Each nonterminal B hypothesized by the learner should "denote" a set $[\![B]\!]^{L_*}$ relative to the target language L_*, independently of the rest of the hypothesized grammar.
- $[\![S]\!]^{L_*} = L_*$.
- Membership in $[\![B]\!]^{L_*}$ should **reduce in polynomial time** to membership in L_*.
- This reduction must be uniform across different target languages.

We defined validity for productions involving the start nonterminal only, which should "denote" the target language. In the general case, each nonterminal has a denotation determined by the target language. When you test a candidate production for validity, you don't know what other productions are in the grammar you hypothesize. So you don't know what strings are derived from each nonterminal.

Validity

$$\pi: \quad A \to w_0 B_1 w_1 \ \ldots \ B_k w_k$$

$$\pi \text{ is } \textbf{valid} \overset{\text{def}}{\iff} [\![A]\!]^{L_*} \supseteq w_0 [\![B_1]\!]^{L_*} w_1 \ \ldots \ [\![B_k]\!]^{L_*} w_k$$

All productions of $G = (N, \Sigma, P, S)$ are valid

$\iff ([\![B]\!]^{L_*})_{B \in N}$ is a pre-fixed point of G

$\implies L(G) \subseteq L_*$

13

When all productions in your hypothesis grammar are valid, you never overgenerate, because the denotations of the nonterminals form a pre-fixed point.

What Should Nonterminals Denote?

$$G = (N, \Sigma, P, S)$$

$$\frac{\text{left quotients}}{\text{regular languages}} = \frac{??}{\text{context-free languages}}$$

- The set of terminal strings derived from a nonterminal is included in some **quotient** of the language of the grammar:

$$u \backslash L / v = \{ \, x \mid uxv \in L \, \}$$

$$S \Rightarrow_G^* uAv \text{ implies } L_G(A) \subseteq u \backslash L(G)/v$$

$L_G(A) = \{ \, x \in \Sigma^* \mid A \Rightarrow_G^* x \, \}$
$L(G) = L_G(S)$

- Is there anything further that can be said in general??

14

Left quotients played an important role in the case of regular languages.
A left quotient corresponds to a state of the minimal DFA, and membership in it can be determined by a membership query.
Can you use quotients instead of left quotients for context-free languages?

Simplest Class: Nonterminals Denote Quotients

15

- The learner uses pairs of strings as nonterminals.

$$[\![\langle\!\langle u, v\rangle\!\rangle]\!]^{L_*} = u\backslash L_*/v = \{ x \in \Sigma^* \mid uxv \in L_* \}$$

- $S = \langle\!\langle \varepsilon, \varepsilon\rangle\!\rangle$.

$\mathrm{Sub}(T) = \{ x \mid uxv \in T \}$

- Hypothesize production $A \to w_0\, B_1\, w_1\, \ldots\, B_k\, w_k$ iff

$$[\![A]\!]^{L_*} \supseteq w_0\, (\mathrm{Sub}(T) \cap [\![B_1]\!]^{L_*})\, w_1\, \ldots\, (\mathrm{Sub}(T) \cap [\![B_k]\!]^{L_*})\, w_k.$$

approximates $[\![A]\!]^{L_*} \supseteq w_0\, [\![B_1]\!]^{L_*}\, w_1\, \ldots\, [\![B_k]\!]^{L_*}\, w_k$

- L_* has infinitely many quotients unless it is regular, so the learner must stop creating new nonterminals.

We start with a very simple instantiation of this idea, where nonterminals denote quotients of L_*. The strings u, v used to represent nonterminals are drawn from positive data, similarly to the case of the regular languages.

CFGs with the Quotient Property

16

- $G = (N, \Sigma, P, S)$ has the **quotient property** $\overset{\text{def}}{\iff}$ G has a pre-fixed point $(X_B)_{B \in N}$ with $X_S = L(G)$ such that $X_B \in \mathcal{Q}(L(G))$ for all $B \in N$.

$\mathcal{Q}(L) = \{ u\backslash L/v \mid u, v \in \Sigma^* \}$

- X_B "usually" coincides with $L_G(B)$, but doesn't have to be.
- CFGs with the quotient property (with a bound on the length of the right-hand side of productions) can be learned from positive data and membership queries.

When using quotients as denotations of nonterminals, learning is successful if the target language has the *quotient property*.

17

Examples of CFGs with the Quotient Property

- CFGs with just one nonterminal.

$$X_S = L(S) = \varepsilon \backslash L(G)/\varepsilon$$

- Right-linear grammars corresponding to minimal DFAs of regular languages.
- $D_1' = aD_1b$.

the set of Dyck primes

$$S \to aD_1b$$
$$D_1 \to \varepsilon \mid SD_1$$

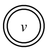
$$S = \varepsilon \backslash D_1'/\varepsilon \ (= D_1'),$$
$$D_1 = a \backslash D_1'/b$$

Left quotients are quotients.
You need at least two nonterminals to generate the set of Dyck primes.
Here I'm writing *B* for the set of strings derived from *B*.

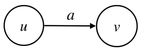
18

Right-Linear Grammars

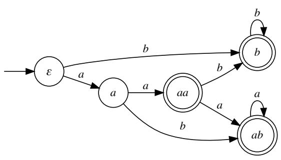
$[\![\langle\!\langle u \rangle\!\rangle]\!]^{L_*} = u \backslash L_*$



$\langle\!\langle v \rangle\!\rangle \to \varepsilon$

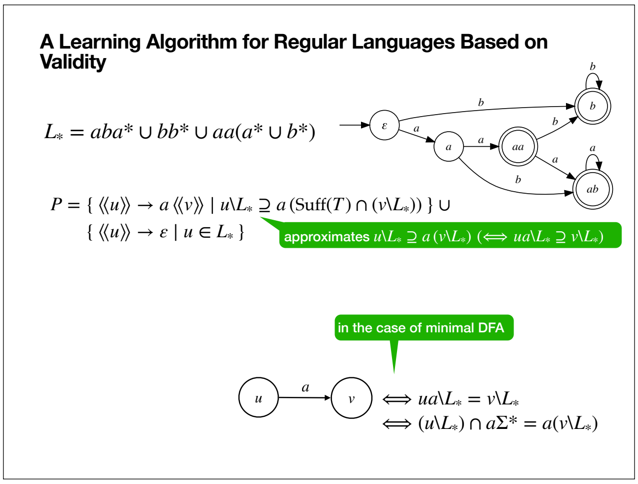


$\langle\!\langle u \rangle\!\rangle \to a\,\langle\!\langle v \rangle\!\rangle$



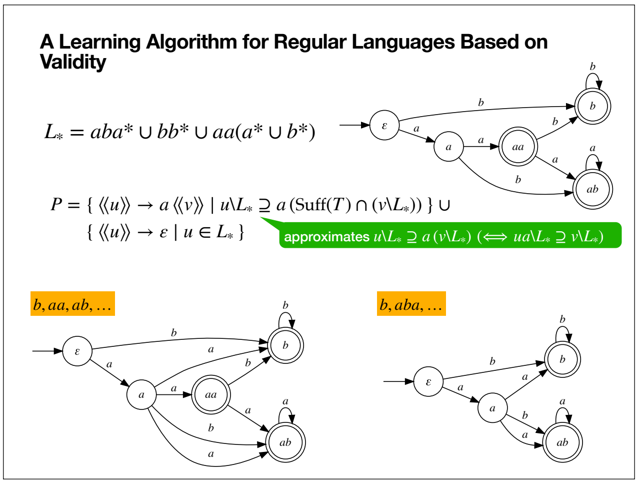
$\langle\!\langle b \rangle\!\rangle \to \varepsilon$
$\langle\!\langle aa \rangle\!\rangle \to \varepsilon$
$\langle\!\langle ab \rangle\!\rangle \to \varepsilon$
$\langle\!\langle \varepsilon \rangle\!\rangle \to a\,\langle\!\langle a \rangle\!\rangle$
$\langle\!\langle \varepsilon \rangle\!\rangle \to b\,\langle\!\langle b \rangle\!\rangle$
$\langle\!\langle a \rangle\!\rangle \to a\,\langle\!\langle aa \rangle\!\rangle$
$\langle\!\langle a \rangle\!\rangle \to b\,\langle\!\langle ab \rangle\!\rangle$
$\langle\!\langle \varepsilon \rangle\!\rangle \to a\,\langle\!\langle a \rangle\!\rangle$
$\langle\!\langle b \rangle\!\rangle \to b\,\langle\!\langle b \rangle\!\rangle$
$\langle\!\langle aa \rangle\!\rangle \to a\,\langle\!\langle ab \rangle\!\rangle$
$\langle\!\langle aa \rangle\!\rangle \to b\,\langle\!\langle b \rangle\!\rangle$
$\langle\!\langle ab \rangle\!\rangle \to a\,\langle\!\langle ab \rangle\!\rangle$

19



Instead of a condition that approximates a certain identity, use a condition that approximates an inclusion.

20



Hypothesized right-linear grammars correspond to NFA.

A Larger Class: Nonterminals Denote Finite Intersections of Quotients

- $G = (N, \Sigma, P, S)$ has the **intersection closure property** $\overset{\text{def}}{\iff}$ G has a pre-fixed point $(X_B)_{B \in N}$ with $X_S = L(G)$ such that X_B is in the intersection closure of $\mathcal{Q}(L(G))$ for all $B \in N$.

$\mathcal{Q}(L) = \{ u \backslash L / v \mid u, v \in \Sigma^* \}$

- CFGs with the intersection closure property (with a bound on some global parameters) can be learned from positive data and membership queries.

$$[\![\langle\!\langle u_1, v_1 \rangle\!\rangle \cap \ldots \cap \langle\!\langle u_l, v_l \rangle\!\rangle]\!]^{L_*} = (u_1 \backslash L_* / v_1) \cap \ldots \cap (u_l \backslash L_* / v_l)$$

21

CFGs with the quotient property cover a very small subclass of the context-free languages.
The intersection closure property is also known as the **(very weak) finite context property** (Kanazawa and Yoshinaka 2017).

CFGs with the Intersection Closure Property

$$L = \{ a^n b^n \mid n \geq 0 \} \cup \{ a^n b^{2n} \mid n \geq 0 \}.$$

$$S \to T \mid U$$
$$T \to \varepsilon \mid aTb$$
$$U \to \varepsilon \mid aUbb$$

$$S = \varepsilon \backslash L / \varepsilon \ (= L)$$
$$T = \varepsilon \backslash L / \varepsilon \cap a \backslash L / b$$
$$U = \varepsilon \backslash L / \varepsilon \cap a \backslash L / bb$$

- L does not have a grammar with the quotient property.
- CFGs with the intersection closure property cover only a small subclass of the CFLs.

22

23

Γ-closure

- Γ: finite set of operations on $\mathscr{P}(\Sigma^*)$ (of variable arity)
- For $\mathscr{L} \subseteq \mathscr{P}(\Sigma^*)$,

$$\Gamma(\mathscr{L}) = \{\, f(L_1, \ldots, L_m) \mid f\colon (\mathscr{P}(\Sigma^*))^m \to \mathscr{P}(\Sigma^*), f \in \Gamma, L_1, \ldots, L_m \in \mathscr{L} \,\}$$
$$\Gamma^0(\mathscr{L}) = \mathscr{L}$$
$$\Gamma^{n+1}(\mathscr{L}) = \mathscr{L} \cup \Gamma(\Gamma^n(\mathscr{L}))$$

Γ-closure of $\mathscr{Q}(L)$

"Γ-expression" over query atoms

- Sets in $\bigcup_{t \geq 0} \Gamma^t(\mathscr{Q}(L))$ can be represented by expressions built from $\langle\!\langle u, v \rangle\!\rangle$ and symbols for operations in Γ.

query atoms

Now let's look at more general classes of representations (used as nonterminals).

24

Γ-closure Property

- A CFG $G = (N, \Sigma, P, S)$ has the **Γ^t-property** $\overset{\text{def}}{\iff}$ G has a pre-fixed point $(X_B)_{B \in N}$ with $X_S = L(G)$ such that $X_B \in \Gamma^t(\mathscr{Q}(L(G)))$ for all $B \in N$.
- G has the **Γ-closure property** $\overset{\text{def}}{\iff}$ G has the Γ^t-property for some t.

25

Learning CFGs with the Γ-Closure Property

- The learner uses Γ-expressions over query atoms as nonterminals.
- $S = \langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle$.
- Hypothesize production $A \to w_0\, B_1\, w_1\, \dots\, B_k\, w_k$ iff
  $[\![A]\!]^{L_*} \supseteq w_0\, (\mathrm{Sub}(T) \cap [\![B_1]\!]^{L_*})\, w_1\, \dots\, (\mathrm{Sub}(T) \cap [\![B_k]\!]^{L_*})\, w_k$.

  approximates $[\![A]\!]^{L_*} \supseteq w_0\, [\![B_1]\!]^{L_*}\, w_1\, \dots\, [\![B_k]\!]^{L_*}\, w_k$
- The algorithm works when Γ-expressions translate into polynomial-time reductions.

26

Extended Regular Closure

$$\Gamma = \{\cap, \overline{\cdot}, \cup\} \cup \{\varnothing, \varepsilon\} \cup \Sigma \cup \{\text{concatenation}, *\}$$

extended regular expression over query atoms

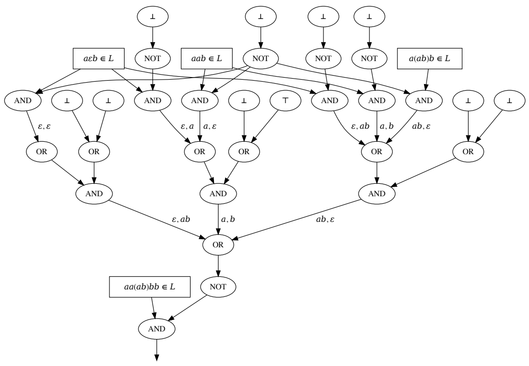
$$\begin{aligned}
&[\![\langle\!\langle aa, bb \rangle\!\rangle \cap \overline{(\langle\!\langle a, b \rangle\!\rangle\, \overline{\varnothing})\, (a \cup b)}]\!]^L \\
&\quad = (aa\backslash L/bb) \cap (\{a,b\}^* - ((a\backslash L/b)(\{a,b\}^* - \varnothing))(\{a\} \cup \{b\})) \\
&\quad = (aa\backslash L/bb) \cap (\{a,b\}^* - (a\backslash L/b)\{a,b\}^*\{a,b\}) \\
&\quad = \{\, x \mid x \in aa\backslash L/bb \wedge \text{no proper prefix of } x \text{ is in } a\backslash L/b \,\}
\end{aligned}$$

- If e is an extended regular expression over query atoms, then $[\![e]\!]^L$ **reduces in polynomial time** to L.

$$[\![e]\!]^L \le_{tt}^{P} L$$

The learning algorithm using Γ-expressions (expressions that stand for sets belonging to the Γ-closure) as nonterminals works when Γ-expressions translate into polynomial-time reductions. When Γ consists of the Boolean and regular operations, we get polynomial-time truth-table reduction.

27



$$ab \in \left[\!\left[\langle\!\langle aa, bb \rangle\!\rangle \cap \overline{(\langle\!\langle a, b \rangle\!\rangle\, \overline{\varnothing})\,(a \cup b)} \right]\!\right]^L$$

The Boolean circuit for the truth function the reduction uses for this particular input. The circuit for "$x \in [[e]]^L$" depends on e and x, but not on L.

28

CFLs having Grammars with the Γ-Closure Property

- What context-free languages can be targeted by learning algorithms using Γ-expressions as nonterminals for various choices of $\Gamma \subseteq \{\cap, \overline{\cdot}, \cup\} \cup \{\varnothing, \varepsilon\} \cup \Sigma \cup \{\text{concatenation}, {}^*\}$?

A Grammar with the Extended Regular Closure Property

29

$$\begin{array}{l} S \to aD_1bS \mid aA \mid bU \\ D_1 \to \varepsilon \mid aD_1bD_1 \\ A \to \varepsilon \mid aD_1bA \mid aA \\ U \to \varepsilon \mid Ua \mid Ub \end{array}$$

$$\begin{aligned} \overline{D_1} &= \{\, x \in \{a,b\}^* \mid \\ &\qquad |x|_a \neq |x|_b \vee \exists uv(uv = x \wedge |u|_a < |u|_b) \,\} \\ &= \{\, x \in \{a,b\}^* \mid \\ &\qquad |x|_a > |x|_b \vee \exists uv(uv = x \wedge |u|_a < |u|_b) \,\} \end{aligned}$$

$$\begin{aligned} A &= \{\, x \in \{a,b\}^* \mid \forall uv(x = uv \to |u|_a \geq |u|_b) \,\} \\ &= \{\, x \in \{a,b\}^* \mid \text{no prefix of } x \text{ is in } D_1b \,\} \\ &= \overline{D_1b\{a,b\}^*} \\ &= [\![\overline{\overline{\langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle}\, b\, (a \cup b)^*}]\!]^{\overline{D_1}} \end{aligned}$$

Our example grammar has the extended regular closure property.

Boolean Closure Property

30

$$\begin{array}{l} S \to D_1bD_1 \mid D_1aD_1 \mid D_1bS \mid SaD_1 \\ D_1 \to \varepsilon \mid aD_1bD_1 \end{array}$$

$$\overline{D_1} = \{\, x \in \{a,b\}^* \mid \exists mn(m + n > 0 \wedge \mathrm{nf}(x) = b^m a^n) \,\}$$

normal form of x under the rewriting $ab \to \varepsilon$

$$\begin{aligned} S &\Rightarrow^* D_1\, b\, \ldots\, D_1\, b\, D_1\, a\, D_1\, \ldots\, a\, D_1 \\ &\Rightarrow^* x_1\, b\, \ldots\, x_m\, b\, y\, a\, z_1\, \ldots\, a\, z_n \end{aligned}$$

$$\begin{aligned} D_1 &= [\![\overline{\langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle}]\!]^{\overline{D_1}} \\ &= [\![\overline{\langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle} \cap \langle\!\langle b, \varepsilon \rangle\!\rangle]\!]^{\overline{D_1}} \end{aligned}$$

In fact, the same language has a grammar with the Γ-closure property with Γ = {¬}.
For technical reasons, the learner needs a positive occurrence of a query atom.

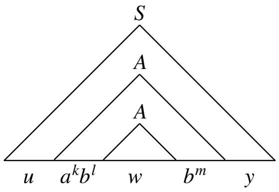
Theorem. $\overline{D_1}$ does not have a grammar with the intersection closure property.

Γ-closure property with $\Gamma = \{ \cap \}$

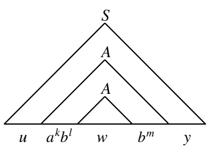
Suppose $\overline{D_1} = L(G)$. Applying Ogden's (1968) theorem to

$$a^{p!+p}\boxed{b^p}$$

with sufficiently large p, with the last p positions marked, we get a derivation tree



with $ua^kb^lwb^my = a^{p!+p}b^p$ and $k \geq l + m$ and $m > 0$.



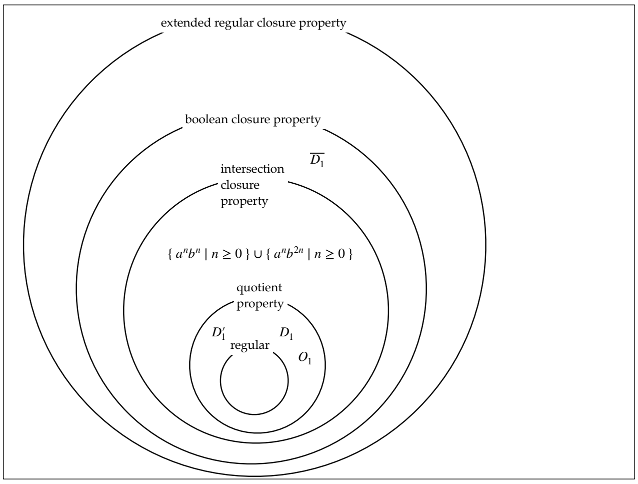
Since $S \Rightarrow^* u(a^kb^l)^nA(b^m)^ny$, any pre-fixed point $(X_B)_{B \in N}$ of G with $X_S = \overline{D_1}$ must have

$$u(a^kb^l)^nX_A(b^m)^ny \subseteq \overline{D_1} \text{ for all } n \geq 0.$$

Let $a^j = u$ and $b^r = y$. Then $b^{j+n(k-l)}a^{nm+r} \notin X_A$ for all $n \geq 0$. This means that

$$b^*a^* - X_A \text{ is infinite.}$$

But for any strings s and t, it holds that $b^*a^* - (s\backslash\overline{D_1}/t) = b^*a^* \cap (s\backslash D_1/t)$ is finite. So X_A cannot be a finite intersection of quotients of $\overline{D_1}$.



33

A Grammar with the Extended Regular Closure Property

$$
\begin{aligned}
S &\to aA \mid aD_1bS \mid bB \mid bD_1^RaS \\
A &\to \varepsilon \mid aA \mid aD_1bA \\
D_1 &\to \varepsilon \mid aD_1bD_1 \\
B &\to \varepsilon \mid bB \mid bD_1^RaB \\
D_1^R &\to \varepsilon \mid bD_1^RaD_1^R
\end{aligned}
$$

$$\overline{O_1} = \{ x \in \{a,b\}^* \mid |x|_a \neq |x|_b \}$$

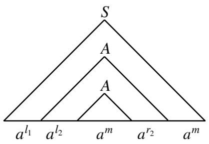
$$
\begin{aligned}
A &= \{ x \in \{a,b\}^* \mid \forall uv(x = uv \to |u|_a \geq |u|_b) \} \\
&= \{ x \in \{a,b\}^* \mid \neg\exists uv(x = uv \land |u|_a + 1 = |u|_b) \} \\
&= \overline{O_1 b \{a,b\}^*} \\
&= [\![\overline{\overline{\langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle}\, b\, (a \cup b)^*}]\!]^{\overline{O_1}} \\
D_1 &= A \cap O_1 \\
&= [\![\overline{\overline{\langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle}\, b\, (a \cup b)^*} \cap \overline{\langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle}]\!]^{\overline{O_1}}
\end{aligned}
$$

34

35

Theorem. $\overline{O_1}$ does not have a grammar with the Boolean closure property.

The pumping lemma applied to a long string a^p gives



with $l_2 + r_2 > 0$. If $(X_B)_{B \in N}$ is a pre-fixed point with $X_S = \overline{O}_1$, then

$$\{ a^{nl_2+m+nr_2} \mid n \geq 0 \} \subseteq X_A \subseteq \bigcap_{n \geq 0} a^{l_1+nl_2} \backslash \overline{O_1} / a^{nr_2+r_1}$$

It follows that $\{ |x|_a - |x|_b \mid x \in X_A \}$ is both infinite and co-infinite.

But $\{ |x|_a - |x|_b \mid x \in u \backslash \overline{O_1} / v \} = \mathbb{Z} - \{ -(|uv|_a - |uv|_b) \}$ is a co-finite set.

36

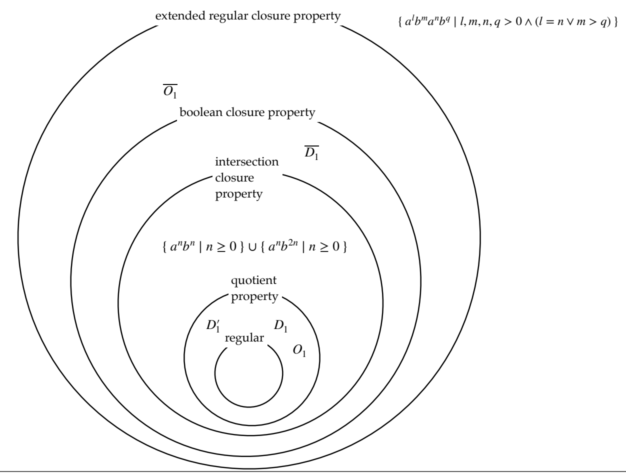
A CFL That Has No Grammar with the Extended Regular Closure Property

$$L = \{ a^l b^m a^n b^q \mid l, m, n, q > 0 \wedge (l = n \vee m > q) \}$$

- L is **inherently ambiguous**.
- L does not have a grammar with the extended regular closure property.

Question. Are there any CFLs that are not inherently ambiguous that have no grammar with the extended regular closure property?

37



38

Star-Free Closure Property

$$
\begin{aligned}
S &\to aA \mid aD_1bS \mid bB \mid bD_1^R aS \\
A &\to \varepsilon \mid aA \mid aD_1bA \\
D_1 &\to \varepsilon \mid aD_1bD_1 \\
B &\to \varepsilon \mid bB \mid bD_1^R aB \\
D_1^R &\to \varepsilon \mid bD_1^R aD_1^R
\end{aligned}
$$

$$\overline{O_1} = \{ x \in \{a,b\}^* \mid |x|_a \neq |x|_b \}$$

$$
\begin{aligned}
A &= \{ x \in \{a,b\}^* \mid \forall uv(x = uv \to |u|_a \geq |u|_b) \} \\
&= \{ x \in \{a,b\}^* \mid \neg \exists uv(x = uv \land |u|_a + 1 = |u|_b) \} \\
&= \overline{O_1 b \{a,b\}^*} \\
&= [\![\overline{\langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle\, b\, (a \cup b)^*}]\!]^{\overline{O_1}} \\
&= [\![\overline{\langle\!\langle \varepsilon, \varepsilon \rangle\!\rangle\, b\, \overline{\varnothing}}]\!]^{\overline{O_1}}
\end{aligned}
$$

star-free expression over query atoms

There is an interesting intermediate class between the CFLs having grammars with the extended closure property and the CFLs having grammars with the Boolean closure property.
A star-free expression is an extended regular expression that does not contain Kleene star (*).

39

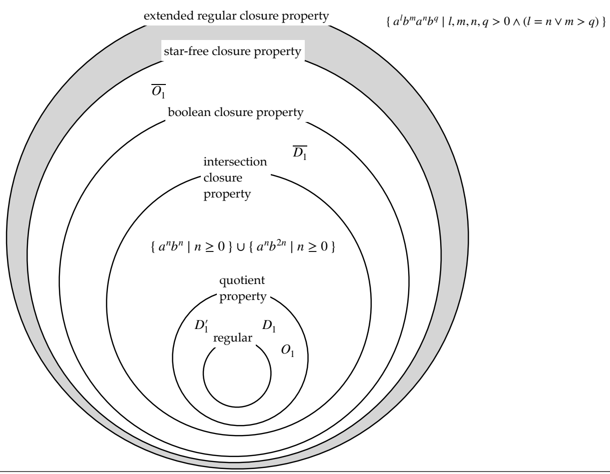
Extended Regular Closure vs. Star-Free Closure

Question. Are there any CFLs that have a grammar with the extended regular closure property but have no grammar with the star-free closure property?

$$L = \{ a^n b^m c^l \mid (n \text{ is odd} \wedge n > m) \vee (n \text{ is even} \wedge n > l) \}$$

L has a grammar with the extended regular closure property, but we do not know whether it has a grammar with the star-free closure property. Note that (aa)* is not a star-free regular language (it has no star-free expression).

40



Simple Deterministic Grammars

41

- A CFG $G = (N, \Sigma, P, S)$ in Greibach normal form is **simple deterministic** $\overset{\text{def}}{\Longleftrightarrow}$ $\{A \to a\,\beta, A \to a\,\gamma\} \subseteq P$ implies $\beta = \gamma$.
- **Theorem** (Ishizaka 1990). If $S \Rightarrow^* v\,A\,y$ and $A \Rightarrow^* x$, then

$$L_G(A) = (v\backslash L(G)/y') \cap \overline{(v\backslash L(G)/y')\Sigma^+}$$
$$= [\![\langle\!\langle v, y'\rangle\!\rangle \cap \overline{\langle\!\langle v, y'\rangle\!\rangle\overline{\varepsilon}}]\!]^{L(G)}$$

  where y' is the shortest suffix of y such that $vxy' \in L(G)$.
- Simple deterministic grammars have the star-free closure property.
- What about larger classes like $LL(k)$?

Simple deterministic grammars are almost the same as what Aho and Ullman (1972) called "simple LL(1)". Ishizaka (1990) showed that simple deterministic grammars are learnable from "extended" equivalence queries and membership queries.

References

42

Hiroki Ishizaka. 1990. Polynomial time learnability of simple deterministic languages. *Machine Learning* **5**, 151–164.

Makoto Kanazawa and Ryo Yoshinaka. 2017. The strong, weak, and very weak finite context and kernel properties. LATA 2017.

Makoto Kanazawa and Ryo Yoshinaka. 2021. A hierarchy of context-free languages learnable from positive data and membership queries. ICGI 2021.

Makoto Kanazawa and Ryo Yoshinaka. 2023. Extending distributional learning from positive data and membership queries. ICGI 2023.

Makoto Kanazawa. 2023. Learning context-free grammars from positive data and membership queries. WoLLIC 2023.