Lecture 9 – Closure properties of context-free languages, Dyck languages

NTIN071 Automata and Grammars

Jakub Bulín (KTIML MFF UK)

Spring 2024

** Adapted from the Czech-lecture slides by Marta Vomlelová with gratitude. The translation, some modifications, and all errors are mine.*

Recap of Lecture 8

- Pushdown automata accept exactly context-free languages (constructions: CFG to PDA and PDA to CFG)
- A deterministic pushdown automaton (DPDA)
- DPDA recognize a proper subclass of context-free languages, accepts by empty stack iff prefix-free and accepts by final state (Deterministic PDA + acceptance by empty stack does not even cover regular languages!)
- Deterministic PDA have unambiguous grammars
- The landscape of languages
- Converting between representations of context-free languages
- Undecidable problems about context-free languages (preview)

2.12 Closure properties of context-free languages

Closed under union, concatenation, iteration, reverse

Theorem

If $L, L' \subseteq \Sigma^*$ are context-free, then so are $L \cup L'$, $L.L'$, L^*, L^+, L^R.

Proof: Let G, G' be CFG generating L, L' such that $V \cap V' = \emptyset$. Take a new start symbol $S_{new} \notin V \cup V'$.

- union $L \cup L'$: add the rule $S_{new} \to S_1 \mid S_2$
- concatenation $L.L'$: add $S_{new} \to S_1 S_2$
- iteration L^*: add $S_{new} \to SS_{new} \mid \epsilon$
- positive iteration L^+: add $S_{new} \to SS_{new} \mid S$
- reverse L^R: reverse the bodies of all production rules (i.e., $A \to \beta$ becomes $A \to \beta^R$) $\square$

Not closed under intesection

Example

$L = \{0^n1^n2^n \mid n \geq 1\} = \{0^n1^n2^i \mid n, i \geq 1\} \cap \{0^i1^n2^n \mid n, i \geq 1\}$

L is not context-free, even though both operands of the intersection are context-free:

$L_1 = \{0^n1^n2^i \mid n, i \geq 1\}$ generated by $G = (\{S, A, B\}, \{0, 1\}, \mathcal{P}, S)$ with production rules

$$\mathcal{P} = \{S \to AB, A \to 0A1 \mid \epsilon, B \to 2B \mid \epsilon\}$$

$L_2 = \{0^n1^n2^i \mid n, i \geq 1\}$ generated similarly using production rules

$$\mathcal{P} = \{S \to AB, A \to 0A \mid \epsilon, B \to 1B2 \mid \epsilon\}$$

Simulating two PDAs in parallel

Regular languages are closed under intersection, because we can simulate two DFAs in parallel. Why not PDAs?

- the FA units can be merged (same as for DFAs)
- reading input can be merged (one automaton can wait)
- but two stacks cannot be simulated on one stack!

In fact, 'PDAs with two stacks' are equivalent to Turing machines, can recognize any recursively enumerable language $L \in \mathcal{L}_0$.

⋮

But what if one of the PDAs does not really use its stack?

Intersection of a context-free and a regular language

Theorem

Let L be a context-free language and R a regular language. Then $L \cap R$ is context free. Moreover, if L is deterministic, so is $L \cap R$.

Proof: Let $L = L(P)$ for a PDA $P = (Q_1, \Sigma, \Gamma, \delta_1, q_1, Z_0, F_1)$ and $R = L(A)$ for a DFA $A = (Q_2, \Sigma, \delta_2, q_2, F_2)$. Construct a PDA

$$M = (Q_1 \times Q_2, \Sigma, \Gamma, \delta, (q_1, q_2), Z_0, F_1 \times F_2)$$

where we have a transition $\delta((q_1, q_2), a, X) \ni ((r_1, r_2), \gamma)$ iff either

(i) $a \neq \epsilon$ and $(r_1, \gamma) \in \delta_1(q_1, a, X)$ and $r_2 = \delta_2(q_2, a)$, or

(ii) $a = \epsilon$ and $(r_1, \gamma) \in \delta_1(q_1, \epsilon, X)$ and $r_2 = q_2$

In (i) both automata read input, in (ii) P works on its stack while A waits. Clearly, $L(M) = L(P) \cap L(A)$ (P and R run in parallel). $\square$

An application: proving non-context-freeness

Example

$L = \{0^i 1^j 2^k 3^\ell \mid i = 0 \text{ or } j = k = \ell\}$ is not context-free.

By contradiction, assume L is context-free.

The language $L_1 = \{01^j 2^k 3^\ell \mid i, j, k \geq 0\}$ is regular (e.g. a regular grammar $\{S \to 0B, B \to 1B \mid C, C \to 2C \mid D, D \to 3D \mid \epsilon\}$.

But $L \cap L_1 = \{01^i 2^i 3^i \mid i \geq 0\}$ is not context-free, a contradiction with the previous theorem. $\square$

In fact, L is a context-sensitive language:

$S \to \epsilon \mid 0 \mid 0A \mid B_1 \mid C_1 \mid D_1$
$A \to 0 \mid 0A \mid P,\ P \to 1PCD \mid 1CD$
$B_1 \to 1 \mid 1B_1 \mid C_1$
$C_1 \to 2 \mid 2C_1 \mid D_1$
$D_1 \to 3 \mid 3D_1$

$DC \to CD$ rewrite as context-sensitive rules
$DC \to XC,\ XC \to XY,$
$XY \to CY,\ CY \to CD$

$1C \to 12$
$2C \to 22$
$2D \to 23$
$3D \to 33$

Difference and complement

Theorem

The class of the context-free languages is not closed under difference, nor complement.

Proof: $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$, closure under complement would imply closure under intersection. For difference, use $\overline{L} = \Sigma^2 - L$. $\square$

NB: PDA is non-deterministic, switching accepting/non-accepting states does not work.

Proposition

If L is context-free and R regular, then $L - R$ is context-free.

Proof: $L - R = L \cap \overline{R}$, and $\overline{R}$ is also regular. $\square$

Substitution and homomorphism

Recall the definitions

A (string) substitution is a mapping $\sigma\colon \Sigma^* \to \mathcal{P}(Y^*)$ where

- Σ and Y are finite alphabets, $Y = \bigcup_{x \in \Sigma} Y_x$
- for each $x \in \Sigma$, $\sigma(x)$ is a language over Y_x
- $\sigma(\epsilon) = \{\epsilon\}$ and $\sigma(u.v) = \sigma(u).\sigma(v)$

For a language $L \subseteq \Sigma^*$, $\sigma(L) = \bigcup_{w \in L} \sigma(w) \subseteq Y^*$.

A (string) homomorphism is defined similarly but each letter is mapped to a single word, $h\colon \Sigma^* \to Y^*$ where $h(x) \in Y_x^*$ for $x \in \Sigma$, $h(\epsilon) = \epsilon$ and $h(u.v) = h(u).h(v)$. Then $h(L) = \{h(w) \mid w \in L\}$.

The inverse homomorphism applied to a language $L' \subseteq Y^*$:

$$h^{-1}(L') = \{w \in \Sigma^* \mid h(w) \in L'\}$$

Example: substitution

Example

Consider $G = (\{E\}, \{a, +, (,)\}, \{E \to E + E \mid (E) \mid a\}, E)$. Let us have the following substitution:

- $\sigma(a) = L(G_a)$, where

$$G_a = (\{I\}, \{a, b, 0, 1\}, \{I \to I0 \mid I1 \mid Ia \mid Ib \mid a \mid b\}, I)$$

- $\sigma(+) = \{-, *, :, \div, \mod\}$
- $\sigma(() = \{(\}$
- $\sigma()) = \{)\}$

Take $(a + a) + a \in L(G)$. Note that $(a + a) + a \notin \sigma(L(G))$, because $+ \notin \sigma(+)$. But e.g. $(a001 - bba) * b1 \in \sigma((a + a) + a) \subseteq \sigma(L(G))$

What if we modify the definition: $\sigma(() = \{(, [\}, \sigma()) = \{),]\}$?

Example: homomorphism

Example

$G = (\{E\}, \{a, +, (,)\},$
$\{E \to E + E \mid (E) \mid a\}, E)$

- $h(a) = \epsilon$
- $h(+) = \epsilon$
- $h(() = \mathit{left}$
- $h()) = \mathit{right}$

- $h((a + a) + a) = \mathit{leftright}$,
- $h^{-1}(\mathit{leftright}) \ni (a + +)a$.

Example

$G = (\{E\}, \{a, +, (,)\},$
$\{E \to a + E \mid (E) \mid a\}, E)$

- $h_2(a) = a$
- $h_2(+) = +$
- $h(() = \epsilon$
- $h()) = \epsilon$

Are the following regular?

1. $L(G)$
2. $h(L(G))$
3. $h^{-1}(h(L(G)))$
4. $h^{-1}(h(L(G))) = L(G)$

Closure under substitution and homomorphism

Theorem

Let $L \subseteq \Sigma^*$ be a context-free language.

(i) *If σ is a substitution on Σ such that $\sigma(a)$ is context-free for all $a \in \Sigma$, then $\sigma(L)$ is context-free.*

(ii) *If h is a homomorphism on Σ, then $h(L)$ is context-free.*

Proof: TODO

2.13 Dyck languages

Dyck languages

A characterization of context-free languages

The proof