CS 5800 Notes

based on T. A. Sudkamp[1]

1 CFLs and PDAs

By definition, L is a context-free language (CFL) if it is derived by a context-free grammar (CFG). The main theorem regarding the relationship between CFLs and push-down automata (PDAs) is

Theorem 1.1. *L is a CFL $\iff$ L is accepted by a PDA.*

Subsequently we will give a construction for the $\Longrightarrow$ part of the statement. To make it easier we will assume that the CFG is in Greibach Normal Form (GNF), which is defined as

Definition 1.1. *A CFG $G = (V, \Sigma, P, S)$ is in GNF if all its rules adhere to the following set of forms:*

$$\begin{aligned} &A \to aA_1A_2 \dots A_n \\ &A \to a \\ &S \to \lambda \end{aligned}$$

where $A_1, \dots, A_n \in V - \{S\}$ and $a \in \Sigma$.

Example 1.1. Ex. Sudkamp p. 248 #12, GNF grammar G :

$$\begin{aligned} &S \to aABA \mid aBB \\ &A \to bA \mid b \\ &B \to cB \mid c \end{aligned}$$

We construct an extended PDA that accepts $L(G)$, with two states q_0 and q_1 where q_0 is the start state and q_1 is the (only) final state, stack symbols $A, B \in V - S$, and transitions obtained as follows.

- The S rules $S \to aA_1A_2 \dots A_n$ give rise to transitions from q_0 to q_1 that process the input symbol and push the string $A_1A_2 \dots A_n$ onto the stack:

$$\delta(q_0, a, \lambda) = \{[q_1, ABA], [q_1, BB]\}$$

- All other rules, of the form $A \to bA_1A_2 \dots A_n$ yield transitions from q_1 to q_1, processing the input symbol and replacing the stack top A (left of the rule) by the string of non-terminals $A_1A_2 \dots A_n$ on the stack:

$$\delta(q_1, b, A) = \{[q_1, A], [q_1, \lambda]\}$$
$$\delta(q_1, c, B) = \{[q_1, B], [q_1, \lambda]\}$$

Another example is given in [1] Section 7.3 p. 232. The general construction of an extended PDA $M = (Q_M, \Sigma_M, \Gamma_M, \delta_M, q_0, F_M)$ from a GNF grammar $G = (V, \Sigma, P, S)$ so that $L(M) = L(G)$ is as follows:

$$\begin{aligned} Q_M &= \{q_0, q_1\} \\ \Sigma_M &= \Sigma \\ \Gamma_M &= V - \{S\} \\ F_M &= \{q_1\} \end{aligned}$$

with start state q_0 and the transition function determined as

$$\begin{aligned} &\delta(q_0, a, \lambda) = \{[q_1, w] \mid S \to aw \text{ is a rule in } P\} \\ &\delta(q_1, a, A) = \{[q_1, w] \mid A \to aw \text{ is a rule in } P,\ A \in V - \{S\}\} \\ &\delta(q_0, \lambda, \lambda) = \{[q_1, \lambda]\},\ S \to \lambda \text{ is a rule in } P \end{aligned}$$

Another construction is given for the $\Longleftarrow$ part of Theorem 1.1 (from PDA to grammar). We will not cover that construction here.

2 The pumping lemma for CFLs

We can use the pumping lemma for CFLs to show for some languages that they are not context-free. First we introduce the Chomsky Normal Form (CNF) of context-free grammars, and study the size and structure of their parse trees.

Definition 2.1. *A CFG $G = (V, \Sigma, P, S)$ is in CNF if all its rules adhere to the following set of forms:*

$$\begin{aligned} A &\to BC \\ A &\to a \\ S &\to \lambda \end{aligned}$$

where $B, C \in V - \{S\}$ and $a \in \Sigma$.

Lemma 2.1. Let G be a CFG in CNF and T a derivation tree corresponding to $A \stackrel{*}{\Rightarrow} w$ for $w \in \Sigma^*$. Then for $n \geq 1$, $\text{depth}(T) \leq n \Longrightarrow |w| \leq 2^{n-1}$.

Proof: by induction on n; since G is in CNF, the parse tree is a binary tree.
Basis: $n = 1 \Rightarrow$ the derivation is $A \Rightarrow w = \lambda$ or $A \Rightarrow w = a$ (symbol $\in \Sigma$). Thus $|w| \leq 2^0 = 1$.
Induction hypothesis (IH): Assume the property holds as stated for depth n.
Induction step: Consider T of depth $\leq n+1$. Then the derivation is of the form $A \Rightarrow BC \overset{*}{\Rightarrow} w$. The subtrees of T rooted at B or C are of depth $\leq n$, for which the IH holds. Thus for $B \overset{*}{\Rightarrow} w_1 \in \Sigma^*$ and $C \overset{*}{\Rightarrow} w_2 in \Sigma^*$, the derived strings satisfy $|w_1| \leq 2^{n-1}$ and $|w_2| \leq 2^{n-1}$, and $|w| = |w_1| + |w_2| \leq 2(2^{n-1}) = 2^n$. This is as stated in the Lemma with n replaced by $n+1$.

QED

The implication in Lemma 2.1, of the form $\mathcal{P}_1 \Longrightarrow \mathcal{P}_2$, can be written as (not $\mathcal{P}_2$) $\Longrightarrow$ (not $\mathcal{P}_1$) or $|w| > 2^{n-1} \Longrightarrow \text{depth}(T) > n$.

Corollary 2.1. *Let $S \overset{*}{\Rightarrow} w \in L(G)$ be a derivation in the CNF grammar G; then* $|w| \geq 2^n \Longrightarrow$ *depth*$(T) \geq n+1$.

Theorem 2.1. *(Pumping lemma for CFLs)*
Let L be a CFL. There is a number k, depending on L, such that any string $z \in L$ with $|z| \geq k$ can be split up as $z = uvwxy$ where

$$(i)\ |vwx| \leq k$$
$$(ii)\ |v| + |x| > 0$$
$$(iii)\ uv^iwx^iy \in L,\ \text{for } i \geq 0.$$

We will give an outline of the proof mainly to cover the pumping property (iii).
L is derived by a CFG in CNF $G = (V, \Sigma, P, S)$. As the constant k we take $k = 2^n$ where $n = \#V$ (the number of variables of the grammar). Let $z \in L$ be a string with length $|z| \geq k$ and let parse tree T correspond to the derivation $S \overset{*}{\Rightarrow} z$, as pictured in Fig. 1.

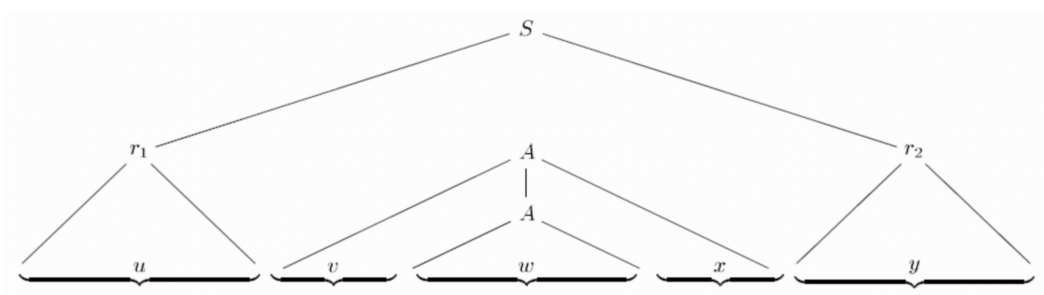

Figure 1: Parse tree for string $z \in L$ of length $|z| \geq k$

According to Corollary 2.1, $|z| \geq 2^n \Longrightarrow \text{depth}(T) \geq n+1$. Let us consider a path of maximal length $n+1 = \#V + 1$ in T. From the root to a leaf of T, this path contains $n+2$ nodes, the last one of which is a leaf (corresponding to a terminal symbol). Thus along the path there are $n+1$ nodes with non-terminals. Since only n nonterminals are available

($\#V = n$), this means that there is at least one re-occurrence of a non-terminal, say A. Fig. 1 shows the last and the one-before-last occurrence of A on the path considered in the derivation $S \overset{*}{\Rightarrow} z$, which results in the subderivations

$$
\begin{aligned}
&S \overset{*}{\Rightarrow} r_1 A r_2 \\
&r_1 \overset{*}{\Rightarrow} u \\
&r_2 \overset{*}{\Rightarrow} y \\
&A \overset{+}{\Rightarrow} vAx \\
&A \overset{*}{\Rightarrow} w
\end{aligned}
$$

The subderivation $A \overset{+}{\Rightarrow} vAx$ can be applied once or more than once; performing it i times before $A \overset{*}{\Rightarrow} w$ is applied establishes that the strings $uv^iwx^iy \in L$ for $i \geq 1$. Skipping $A \overset{+}{\Rightarrow} vAx$ but proceeding to $A \overset{*}{\Rightarrow} w$ immediately, handles $i = 0$.

Example 2.1. The language $L = \{a^ib^ic^i \mid i \geq 0\}$ is not context-free.
Proof: by contradiction; assume L is context-free, then the pumping lemma (Theorem 2.1) holds. To come to a contradiction, we choose a string $z \in L$ of length k, $|z| \geq k$, and show that for every decomposition of z that is allowed within the restrictions of the pumping lemma, there are pumped strings that are not in L.

Let $z = a^kb^kc^k$, then according to the pumping lemma, z can be decomposed as $z = uvwxy$. Since $|vwx| \leq k$, we have the following possibilities:

- vwx could be in one region (consisting of one (solid) type of symbol): $vwx \in a^*$ or $vwx \in b^*$ or $vwx \in c^*$; then the pumped string uv^2wx^2y will have an excess of that one symbol compared to the other two symbols.

- vwx could span two regions: $vwx \in a^*b^*$ or $vwx \in b^*c^*$; then

  - If v and/or x contain a boundary of regions of a-s and b-s, or b-s and c-s, the pumped string uv^2wx^2y will contain b-s before a-s, or c-s before b-s.
  - If v and x each consist of one solid symbol, then pumping will increase the numbers of two of the symbols (a-s and b-s, or b-s and c-s), not all three symbols.

In each of these cases, the pumped string $uv^2wx^2y \notin L$, which is a contradiction to the pumping lemma.

See the Sudkamp text [1] for more examples.

3 Closure properties of CFLs

Theorem 3.1. *The set of the CFLs is closed under (1) union, (2) concatenation and (3) Kleene closure.*

Construction:
Given the CFGs $G_1 = (V_1, \Sigma_1, P_1, S_1)$ for $L_1 = L(G_1)$ and $G_2 = (V_2, \Sigma_2, P_2, S_2)$ for $L_2 = L(G_2)$, we construct CFGs for $L_1 \cup L_2$, $L_1 \cdot L_2$, and ${L_1}^*$. We add a new start symbol and new rules as follows:

(1) CFG for $L_1 \cup L_2$: $G = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 \mid S_2\}, S\}$
(2) CFG for $L_1 \cdot L_2$: $G = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 S_2\}, S\}$
(3) CFG for ${L_1}^*$: $G = (V_1 \cup \{S\}, \Sigma_1, P_1 \cup \{S \to S_1 S \mid \lambda\}, S\}$

Theorem 3.2. *The set of the CFLs is not closed under intersection.*

Proof: by contradiction; assume the set of the CFLs is closed under intersection. Let $L_1 = \{a^i b^j c^j \mid i, j \geq 0\}$ and $L_2 = \{a^j b^j c^i \mid i, j \geq 0\}$. We can give CFGs for L_1 and L_2, showing that they are CFLs. The assumption of closure under intersection implies that $L_1 \cap L_2$ should be context-free; However, $L_1 \cap L_2 = \{a^i b^i c^i \mid i \geq 0\}$, which leads to a contradiction since we proved with the pumping lemma for CFLs that $\{a^i b^i c^i \mid i \geq 0\}$ is not context-free. Therefore, the assumption is invalid and the set of the CFLs is not closed under intersection.
Note: This is also called a counter example.

Theorem 3.3. *The set of the CFLs is not closed under complementation.*

Proof: by contradiction; assume the set of the CFLs is closed under complementation.
Let L_1 and L_2 be CFLs $\Longrightarrow \overline{L_1}$ and $\overline{L_2}$ are CFLs
$\Longrightarrow \overline{L_1} \cup \overline{L_2}$ is a CFL (by closure of the CFLs under union)
$\Longrightarrow \overline{L_1 \cap L_2}$ is a CFL (de Morgan's law)
$\Longrightarrow \overline{\overline{L_1 \cap L_2}}$ is a CFL (by the assumption of closure under complementation))
$\Longrightarrow L_1 \cap L_2$ is a CFL. This is a contradiction, as the CFLs are not closed under intersection (see Theorem 3.2).

References

[1] Sudkamp, T. A. *An Introduction to the Theory of Computer Science – Languages and Machines.* Pearson, Addison Wesley, 3rd edition, 2006. ISBN 0-321-32221-5.