School of Computing and Information Systems
COMP30026 Models of Computation
Week 9: Pushdown Automata and Context-Free Languages

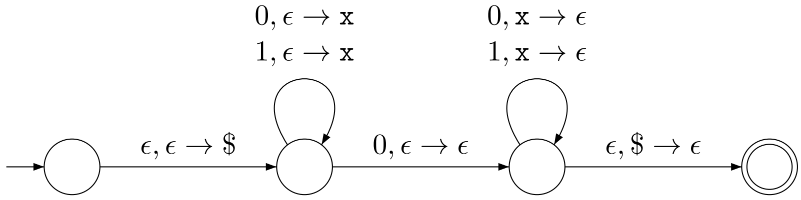
Homework problems

P9.1 Construct a pushdown automaton for

$$\{w \in \{\texttt{0}, \texttt{1}\}^* \mid \text{the length of } w \text{ is odd and its middle symbol is } \texttt{0}\}.$$

What's the minimum number of states you can achieve?

Solution:



P9.2 Take any context-free grammar in any problem set or tutorial, and construct a pushdown automata which recognises the language of that grammar. Try using the CFG to PDA conversion from the lectures, otherwise you can try to understand what the language is, and construct a PDA from scratch. Repeat this a few times for practice.

P9.3 We have seen that the set of context-free languages is not closed under intersection. However, it *is* closed under intersection with regular languages. That is, if L is context-free and R is regular then $L \cap R$ is context-free.

We can show this if we can show how to construct a push-down automaton P' for $L \cap R$ from a push-down automaton P for L and a DFA D for R. The idea is that we can do something similar to what we did in T8.3 when we built "product automata", that is, DFAs for languages $R_1 \cap R_2$ where R_1 and R_2 were regular languages. If P has state set Q_P and D has state set Q_D, then P' will have state set $Q_P \times Q_D$.

More precisely, let $P = (Q_P, \Sigma, \Gamma, \delta_P, q_P, F_P)$ and let $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$. Recall the types of the transition functions:

$$\begin{aligned} \delta_P &: (Q_P \times \Sigma_\epsilon \times \Gamma_\epsilon) \to \mathcal{P}(Q_P \times \Gamma_\epsilon) \\ \delta_D &: (Q_D \times \Sigma) \to Q_D \end{aligned}$$

We construct P' with the following components: $P' = (Q_P \times Q_D, \Sigma, \Gamma, \delta, (q_P, q_D), F_P \times F_D)$. Discuss how P' can be constructed from P and D. Then give a formal definition of δ, the transition function for P'.

Solution: For the case $v \neq \epsilon$ we define

$$\delta((q_p, q_d), v, x) = \{((r_p, r_d), y) \mid (r_p, y) \in \delta_P(q_p, v, x) \wedge r_d = \delta_D(q_d, v)\}$$

But we must also allow transitions that don't consume input, so:

$$\delta((q_p, q_d), \epsilon, x) = \{((r_p, q_d), y) \mid (r_p, y) \in \delta_P(q_p, \epsilon, x)\}$$

P9.4 Give a context-free grammar for $\{\texttt{a}^i\texttt{b}^j\texttt{c}^k \mid i = j \vee j = k \text{ where } i, j, k \geq 0\}$. Is your grammar ambiguous? Why or why not?

P9.5 Consider the context-free grammar $G = (\{S, A, B\}, \{\texttt{a}, \texttt{b}\}, R, S)$ with rules R:

$$\begin{aligned} S &\to A\ B\ A \\ A &\to \texttt{a}\ A \mid \epsilon \\ B &\to \texttt{b}\ B \mid \epsilon \end{aligned}$$

(a) Show that G is ambiguous.

(b) The language generated by G is regular; give a regular expression for $L(G)$.

(c) Give an unambiguous context-free grammar, equivalent to G. *Hint:* As an intermediate step, you may want to build a DFA for $L(G)$.

Solution: We are looking at the context-free grammar G:

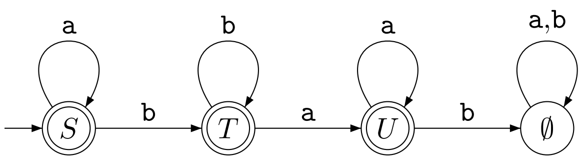
$$\begin{array}{lcl} S & \to & A\ B\ A \\ A & \to & \mathtt{a}\ A \mid \epsilon \\ B & \to & \mathtt{b}\ B \mid \epsilon \end{array}$$

(a) The grammar is ambiguous. For example, `a` has two different leftmost derivations:

$$\begin{array}{l} S \Rightarrow A\ B\ A \Rightarrow B\ A \Rightarrow A \Rightarrow \mathtt{a}\ A \Rightarrow \mathtt{a} \\ S \Rightarrow A\ B\ A \Rightarrow \mathtt{a}\ A\ B\ A \Rightarrow \mathtt{a}\ B\ A \Rightarrow \mathtt{a}\ A \Rightarrow \mathtt{a} \end{array}$$

(b) $L(G) = \mathtt{a}^*\mathtt{b}^*\mathtt{a}^*$.

(c) To find an unambiguous equivalent context-free grammar it helps to build a DFA for $\mathtt{a}^*\mathtt{b}^*\mathtt{a}^*$. (If this is too hard, we can always construct an NFA, which is easy, and then translate the NFA to a DFA using the subset construction method, which is also easy.) Below is the DFA we end up with. The states are named S, T, and U to suggest how they can be made to correspond to variables in a context-free grammar. The DFA translates easily to the grammar on the right. The resulting grammar is a so-called *regular* grammar, and it is easy to see that it is unambiguous—there is never a choice of rule to use.



$$\begin{array}{lcl} S & \to & \epsilon \mid \mathtt{a}\ S \mid \mathtt{b}\ T \\ T & \to & \epsilon \mid \mathtt{b}\ T \mid \mathtt{a}\ U \\ U & \to & \epsilon \mid \mathtt{a}\ U \end{array}$$