Problem 4.1. (15 points) For each of the following three languages B_i over the alphabet $\{0, 1\}$, give either a PDA that accepts B_i or a CFG that generates B_i.

a. B_1 is the set of palindromes, i.e., the set of words that read the same forwards and backwards. For eample, 010 is a palindrome; 0100 is not.

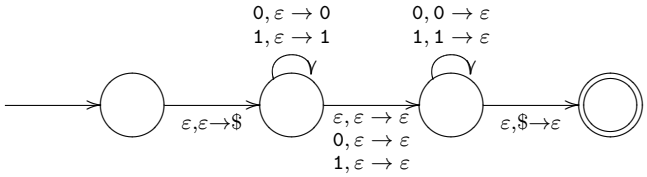
b. $B_2 = \{1^{2k}01^{3k} \mid k \geq 0\}$.

c. B_3 is the set of words that contain more 1's than 0's.

Solution:

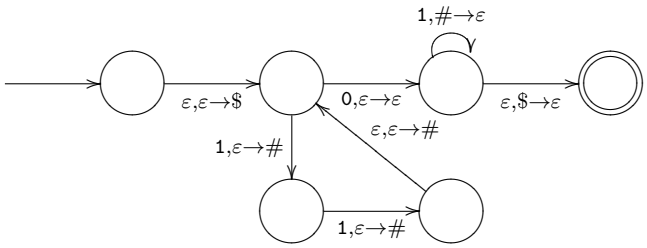
a. CFG:
$S \to \varepsilon \mid 0 \mid 1 \mid 0\ S\ 0 \mid 1\ S\ 1$.

PDA:



b. CFG:
$S \to 0 \mid 11\ S\ 1\ 1\ 1$.

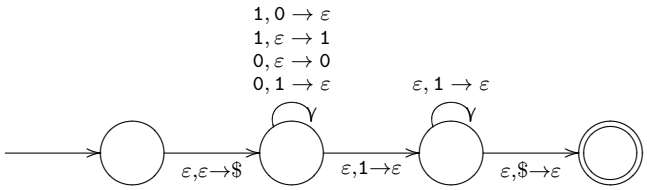
PDA:



Note that the choice of the symbol we push and pop from the stack (here #) does not matter as long as it is used consistantly for both operations. In particular, this symbol does not have to be in the *input* alphabet Σ_ε, but only in the *stack* alphabet Γ_ε.

c. CFG:
$S \to \mathtt{1}\ B \mid \mathtt{0}\ S\ S \mid S\ S$
$B \to \varepsilon \mid \mathtt{0}\ B\ \mathtt{1} \mid \mathtt{1}\ B\ \mathtt{0} \mid B\ B$

PDA:



We build a similar automaton as for the language which contains as many 0's as 1's, except that after parsing the input, we should be left with atleast one 1 in the stack. In the CFG, S denotes strings with atleast one more 1 than number of 0's. B denotes strings with equal number of 0's and 1's.

Problem 4.2. (15 points) For a word x of even length, let $\mathit{half}(x)$ be the first half of x. For a language C, let

$$\mathit{half}(C) \;=\; \{\mathit{half}(x) \mid x \in C \text{ and } |x| \text{ is even}\}.$$

a. If C is a regular language, is $\mathit{half}(C)$ necessarily regular as well? If so, then given a finite automaton for C, construct a finite automaton for $\mathit{half}(C)$. If not, then prove that $\mathit{half}(C)$ is not regular for some regular language C of your choice.

b. Show that if C is regular, then $\mathit{half}(C)$ is context-free, e.g., given a finite automaton for C, define a generic construction of a PDA that accepts $\mathit{half}(C)$.

Solution:

a. If C is regular, $\mathit{half}(C)$ is also regular. The word w is in $\mathit{half}(C)$ if there is a state p in the state set of the finite automaton accepting C such that there exists a run on w leading from the initial state to p and there exists a word of length $|w|$ in Σ^* which has a run leading from p to a final state of the automaton. We use this property in the following construction:

We consider an NFA $N = (Q, \Sigma, \delta, q_0, F)$ accepting C. For simplicity reasons, we assume that N has no ε-transitions. Therefore, if N contains ε-transitions, we need to remove them first using the algorithm covered during the lecture. We define $M = (Q', \Sigma, \delta', q_0', F')$ as follows:

$$\begin{array}{ll} Q': & Q \times \mathcal{P}(Q) \\ q_0': & (q_0, F) \\ F': & \{(q,S) \in Q' \mid q \in S\} \\ \delta': & \delta'((q,T),\ a) = \bigcup_{q' \in \delta(q,\ a)} (q',\ T') \\ & \text{where } T' = \{t \in Q \mid \text{there exists } b \in \Sigma \text{ such that } \delta(t,b) \cap T \neq \varnothing\} \end{array}$$

Intuitively, this machine simulates N both forward from the start state and backwards from the final states. The only complication is that in the backward direction, the machine simultaneously goes backwards along all arrows. If there is a state reachable forwards reading w that is also reachable backwards reading some $x \in \Sigma^*$ of length $|w|$, then M accepts. Therefore M recognizes the language $\mathit{half}(C)$.

b. We first note that, using the construction above, we are able to construct a finite automaton for $\mathit{half}(C)$ given a finite automaton for C. Since every finite automaton is also a pushdown automaton (which doesn't use the stack), anwering question *a.* also answers question *b.* If we reason on languages only, we show in question *a.* (by construction) that if C is regular, then $\mathit{half}(C)$ is also regular. Since a regular language is also context free (the set of regular languages is included in the set of context free languages), $\mathit{half}(C)$ is context free.

It is also possible to make use of the stack and build a PDA which only uses $2 \cdot |Q| + 2$ states rather than the exponential number of states introduced by the powerset in a.). We build this automaton by combining two copies of the finite automata given for C. We start by putting a stack bottom delimiter (\$) on the stack, then move to the inital state of the first copy of the automaton. This copy has been slightly modified to push a symbol on the stack each time a symbol is read on the input. There is an ε-transition from every state of this first copy to the same state in the second copy. The second copy has been modified so that it doesn't read the input at all, but pops one element from the stack on each transition which was reading the input in the original automaton. The final states of the second copy contains a transition going to the final state of the PDA if the stack is empty. Below is a formal definition of this construct. For simplicity reasons, we give it for a DFA.

Given a DFA $N = (Q, \Sigma, \delta, q_0, F)$ accepting C, we define the PDA $P = (Q', \Sigma, \Gamma', \delta', q_0', F')$ as follows:

Γ' $\{\$, \#\}$
Q': $(\{0, 1\} \times Q) \cup \{q_0', q_f\}$
F': $\{q_f\}$
δ': $\delta'(q_0', \varepsilon, \varepsilon) = ((0, q_0), \$)$
and for every $q \in Q$, we have
$\delta'((0, q), a, \varepsilon) = ((0, \delta(q, a)), \#)$ for each $a \in \Sigma$
$\delta'((0, q), \varepsilon, \varepsilon) = ((1, q), \varepsilon)$
$\delta'((1, q), \varepsilon, \#) = \{((1, \delta(q, a)), \varepsilon) \mid a \in \Sigma\}$
$\delta'((1, q), \varepsilon, \$) = (q_f, \varepsilon)$ if $q \in F$

An accepting run on this PDA would go as follows: first the entire word w is read by the first copy of the original DFA, and for each symbol read on the input, a # gets pushed on the stack. Once the word has been read, the stack contains $|w|$ symbols and the automaton moves to the second copy of the DFA, in which every transition happens with a pop from the stack. If the second copy ends in a final state with an empty stack, then there must be a word x such that $wx \in C$ and $|x| = |w|$. Therefore $w \in \mathit{half}(C)$.