Last time: given a CFG, how to make a PDA that accepts the lang gen. by the CFG

Key idea: Modify the CFG so that, in a leftmost deriv, we always have letters → vars. Then have the stack of the PDA keep the vars.

Today: Given a PDA, how to make a CFG that generates the lang accepted by the PDA.

Key idea: Variables in CFG will look like \[
\text{state, stack, symbol, state} \]
- this variable can generate any string that is accepted by the PDA going from left state to right state consuming the stack symbol from stack.

\[
\text{one variable}
\]
Given a PDA - we'll make it an equivalent PDA so that
1) it has one final state, and when you enter this final state, there is nothing on the stack.
2) we'll assume for ease of description that every transition either decreases stack size by one, or increases stack size by one.

The start variable for our CFG is $q_0, z, q_f$  (where $q_0 =$ start state, $z =$ bottom of stack symbol, $q_f =$ final state)

Given a transition

$(q, l, s) \rightarrow (q', \lambda)$,
we have a production

$q_0, s, q' \rightarrow l$
\[(q, l, s) \rightarrow (q', tu)\]

we have transitions productions

\[
\begin{array}{c}
q, s, p \\
\rightarrow\ l \ [q', t, p'] \ [p', u, p]
\end{array}
\]

for every state \(p'\) and every state \(p\).

Note - this generates \(n^2\) productions, where \(n=\#\text{states in PDA}\).

Most of these productions will be useless - but we know how to get rid of useless productions and can do that afterwards.

Example: The PDA that accepts \(\{a^n b^n | n \geq 1\}\)

It's transitions are

\[
\begin{align*}
(q_0, a, z) & \rightarrow (q_1, Az) \\
(q_1, a, A) & \rightarrow (q_1, AA) \\
(q_1, b, A) & \rightarrow (q_2, A) \\
(q_2, b, A) & \rightarrow (q_2, \lambda) \\
(q_2, \lambda, z) & \rightarrow (q_f, \lambda)
\end{align*}
\]
The CFG has $4 \times 2 \times 4 = 32$ variables. Its productions are:

- $q_1, A, q_2 \rightarrow b$
- $q_2, A, q_2 \rightarrow b$
- $q_2, Z, q_f \rightarrow \lambda$

Additionally, there are 8 more productions.
\[ q, A \rightarrow \lambda \]

16 of these

Actually, most of these are useless.