CompSci 162
Spring 2023 Lecture 11:
Equivalence of CFGs and PDAs

Every language accepted by PDA is CF

For all pairs of states p, q, create A_{pq} : all strings that $p \rightsquigarrow q$, empty stack at start and end

rule to generate

Simplifying Assumptions:

1. Only one accept state
2. PDA empties stack before accepting
3. Each transition: exactly one push XOR one pop

If you're unsure why this is okay, think of this: what if I added this requirement an hour before homework is due? How could you modify a PDA to fit this?

Let's create A_{pq}

- What is the first move? The last move?

First is push, last pop

- Does first symbol pushed remain until end?

$\in \Gamma$ here

If yes?:

w/ stack net no change, never empty

$A_{pq} \to a A_{rs} b$

$A_{pp} \to \varepsilon$ all p

$A_{pq} \to A_{pr} A_{rq}$

If no:

$\exists r$ where stack empties

for Any/all where $p \rightsquigarrow r$ and $r \rightsquigarrow q$ empty the stack

Proof of Correctness

Claim: if rule A_{pq} generates x, then x can bring PDA from p with an empty stack to q with an empty stack.
Proof: by induction on how many steps.

Basis: $A_{pp} \to \varepsilon$ is only

Inductive Hypothesis: true for up to k steps.
What if I have $k+1$ steps?

What is the first step?

how many steps?
K-1 or fewer

Suppose first step is $A_{pq} \to a A_{rs} b$

What if first step was $A_{pq} \to \underbrace{A_{pr}}_{y} \underbrace{A_{rq}}_{z}$?

$x = yz$

Proof: A_{pq} generates all such strings

Induction on number of steps in computation.
Basis:

Inductive Hypothesis: suppose true for computation of length at most k.
What if I have a $k+1$ step computation?

The $k + 1$ step computation

Is the stack empty only at start and end?