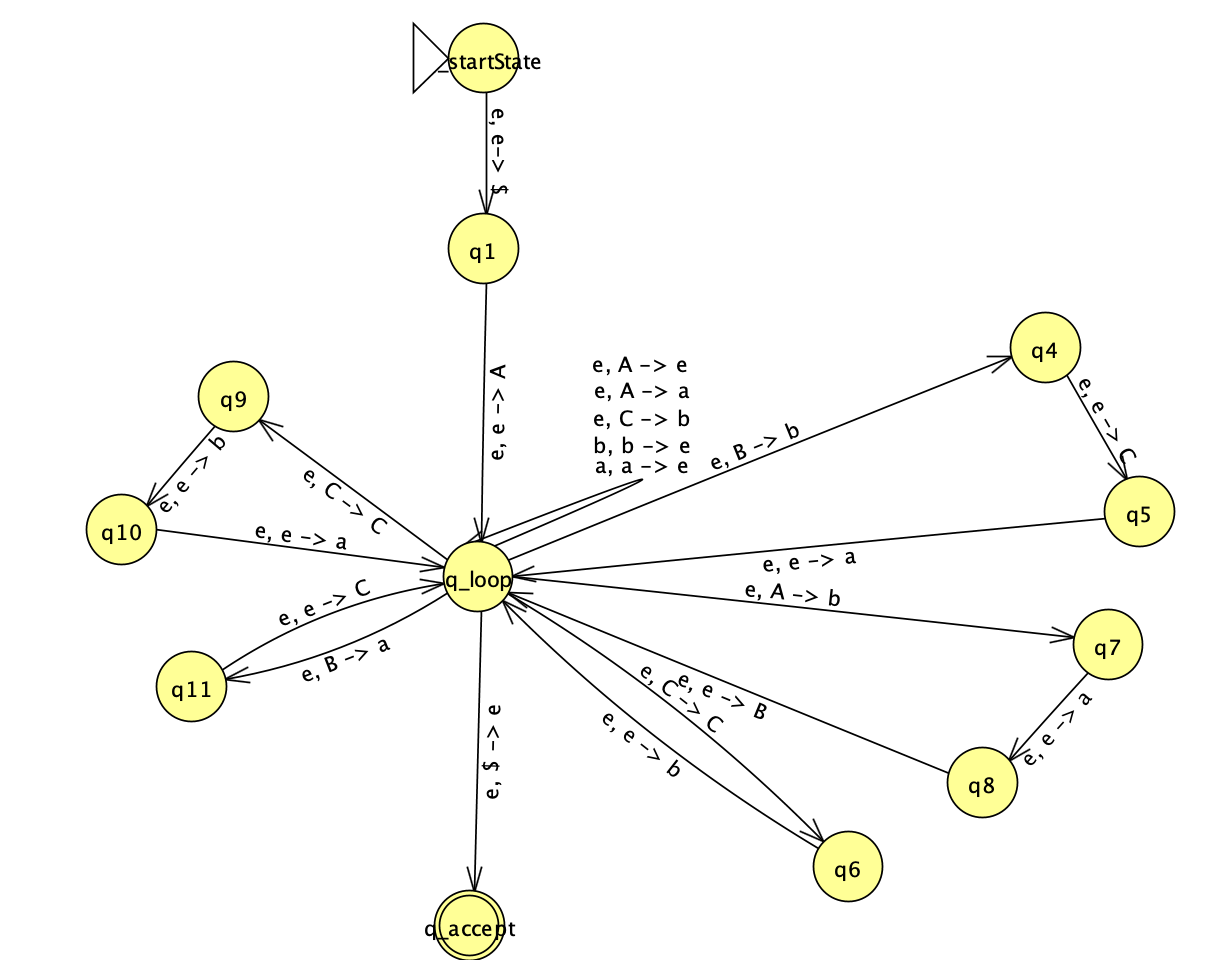
Polina Gannotskaya

Devin Smith

**Context Free Grammar to PDA**

**Algorithm**

1. Create q\_startState
2. Create q1, place $ on stack when transitioning from q\_startState to q1
3. Create q\_loop
4. Create a “petal” using each of the rules:
   1. Iterate backwards through each individual rule, pushing each variable to the stack
   2. Once the end of a rule is reached, return to q\_loop and go to the next rule
5. Create a petal that pops each character off of the stack
6. Create q\_accept, pop $ off the stack when transitioning from q\_loop to q\_accept



**Example Data**

**Input:**

{<A>, <B>, <C>}

{a, b}

<A> :: a | <B> a b | e

<B> :: a <C> b | <C> a

<C> :: b <C> | b | a b <C>

{<A>}

**Output:**

PDA:

q\_startState -> q1 : e, e-> $

q1 -> q\_loop : e, e-> <A>

q\_loop:

e,<A> ->a

|

e,<A> ->b

e, e ->a

e, e -><B>

|

e,<A> ->e

|

e,<B> ->b

e, e -><C>

e, e ->a

|

e,<B> ->a

e, e -><C>

|

e,<C> -><C>

e, e ->b

|

e,<C> ->b

|

e,<C> -><C>

e, e ->b

e, e ->a

|

a, a -> e

b, b -> e

q\_loop -> q\_accept : $, $->e