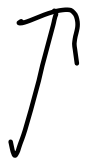


Expect Exam
results
real soon
(watch compass)

Q a set

~~$\{Q\}$~~

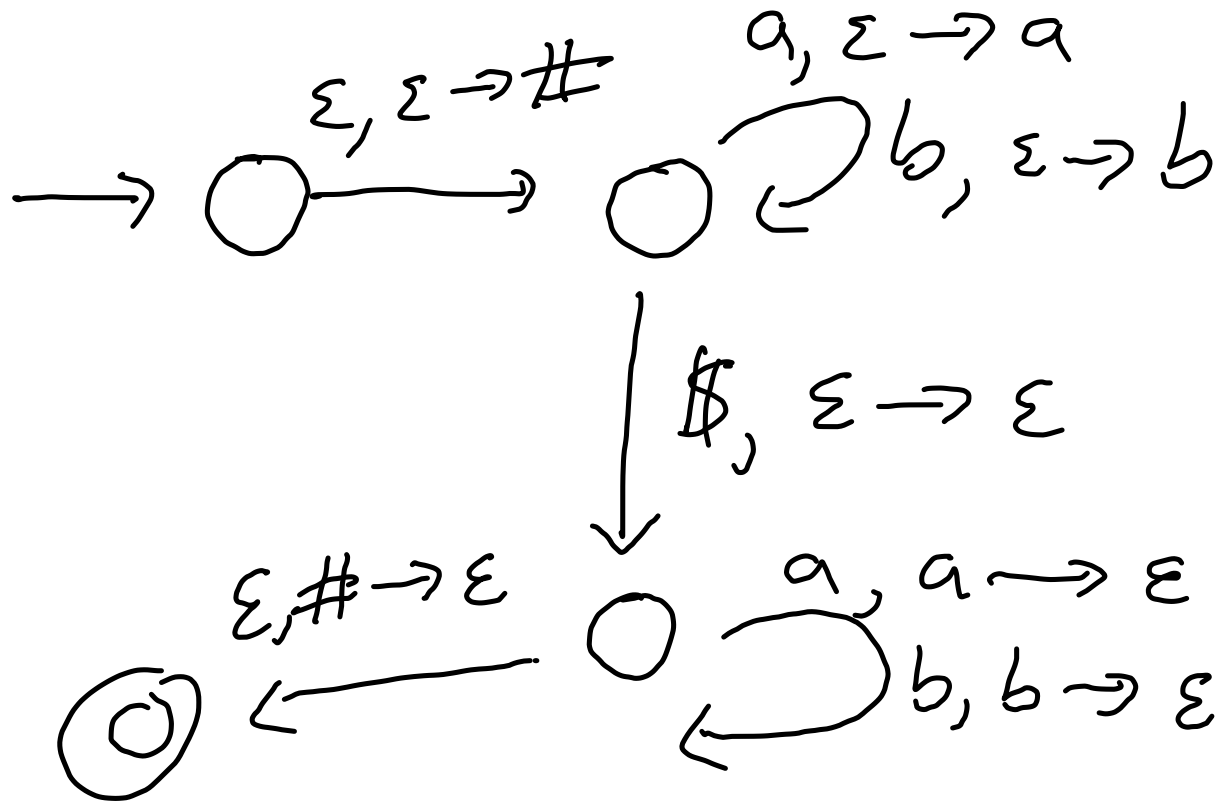
$Q \cup a$

not a set


$S(q, a) + S(q, b)$

use U

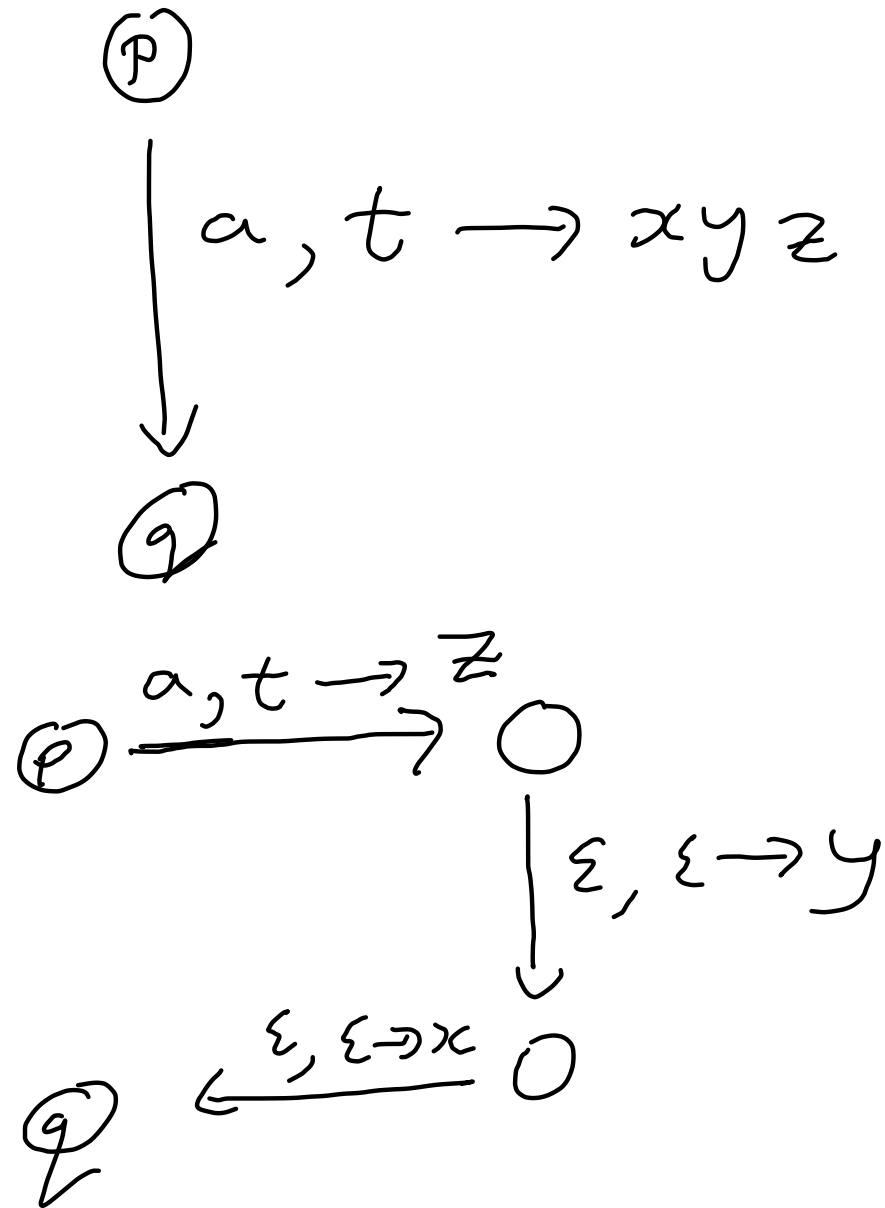

$$L = \{w \# w^r : w \in \{a, b\}^*\}$$



Variations

① PDAs That start w/
a marker Z_0 on stack

② push γ char
onto stack
in a single
move

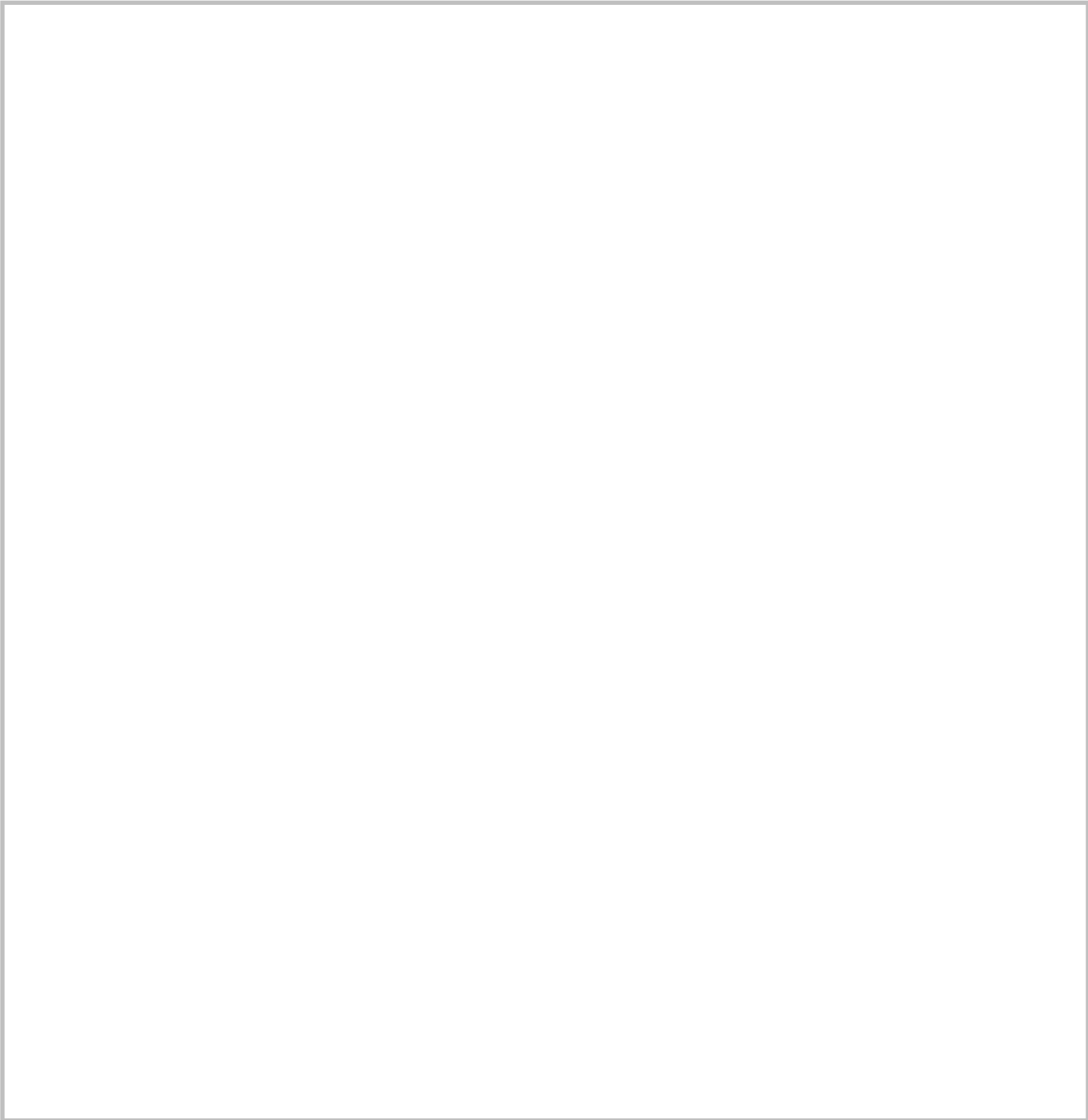


Other variants

e.g. PDA That
reads top 2 chars
on stack

(X, A)
(B, X)
(A, B)
(\$, A)
(\$, \$)

PDA's \leftrightarrow CFG's
are equivalent

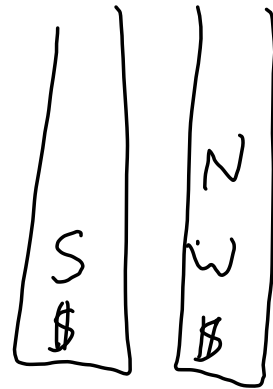


S → N W

W → V N

V → eats

N → cats / mice / oranges



.....



Stuck

interleave

— expanding S
into
terminal string
on stack

— read input char
& removing
Term from stack

S
\$

Cats eat
Oranges

N
W
\$

Cats
eat
Oranges

Cats
W
\$

Cat
eat
Oranges

W
\$

eat
Oranges

V
N
\$

eat,
Oranges

Cats
N
\$

eat
Oranges

N
\$

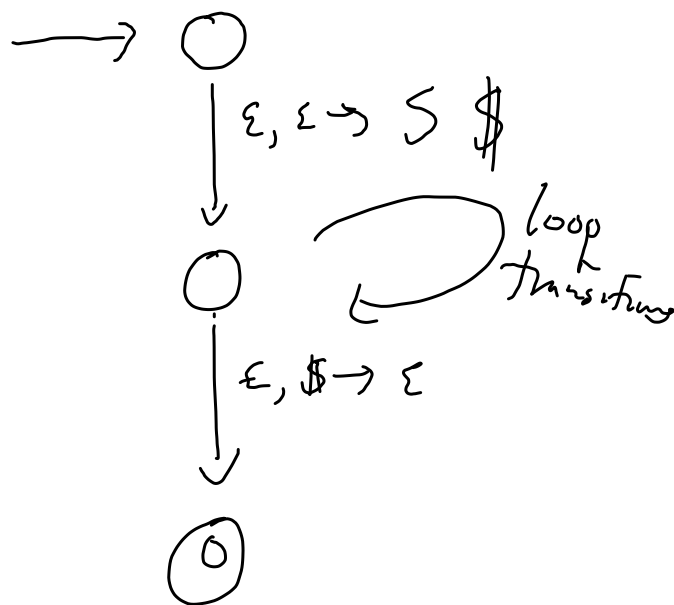
Oranges

Oranges
\$

Oranges

\$

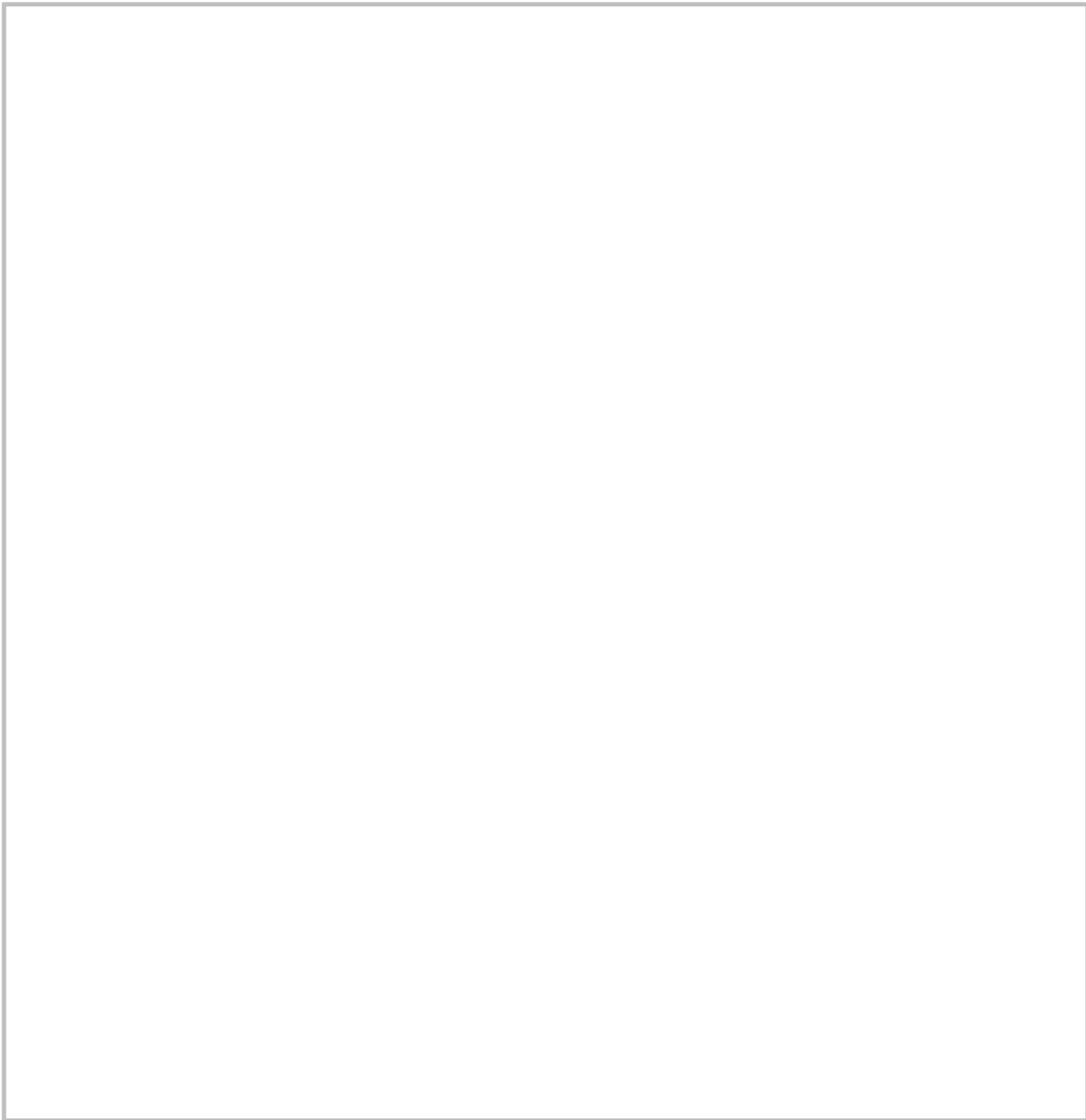
ε



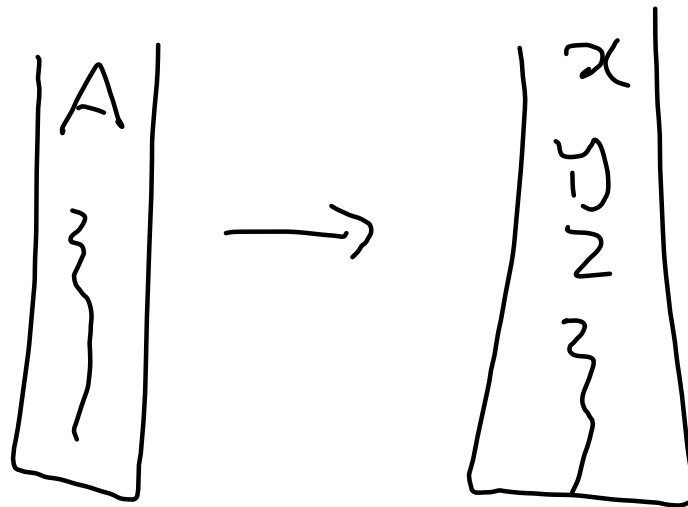
loop transitions are

- $a, a \rightarrow \epsilon$
for each character
 a in Σ
- $\epsilon, A \rightarrow w$

for every rule $A \rightarrow w$
in grammar



$$A \rightarrow xyz$$

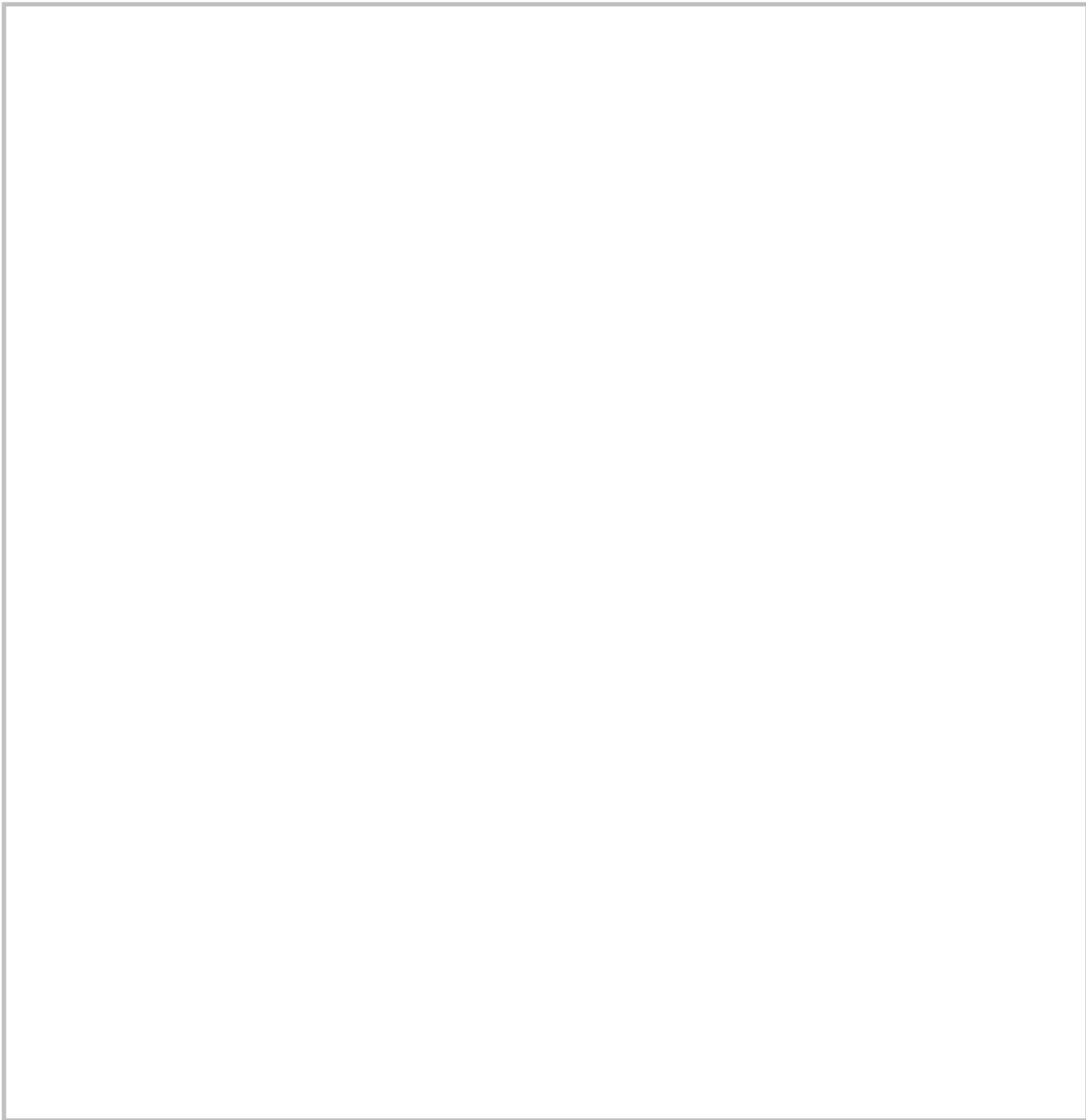


$$X \rightarrow aX \mid \dots$$

$a^* a^+$

Given a PDA
write an equivalent
CFG

- ① simplify form of PDA
- a single accept state
 - empties stack before accepting
 - each transition either pops 1 char or pushes 1 char
NOT BOTH



variables of grammar
are

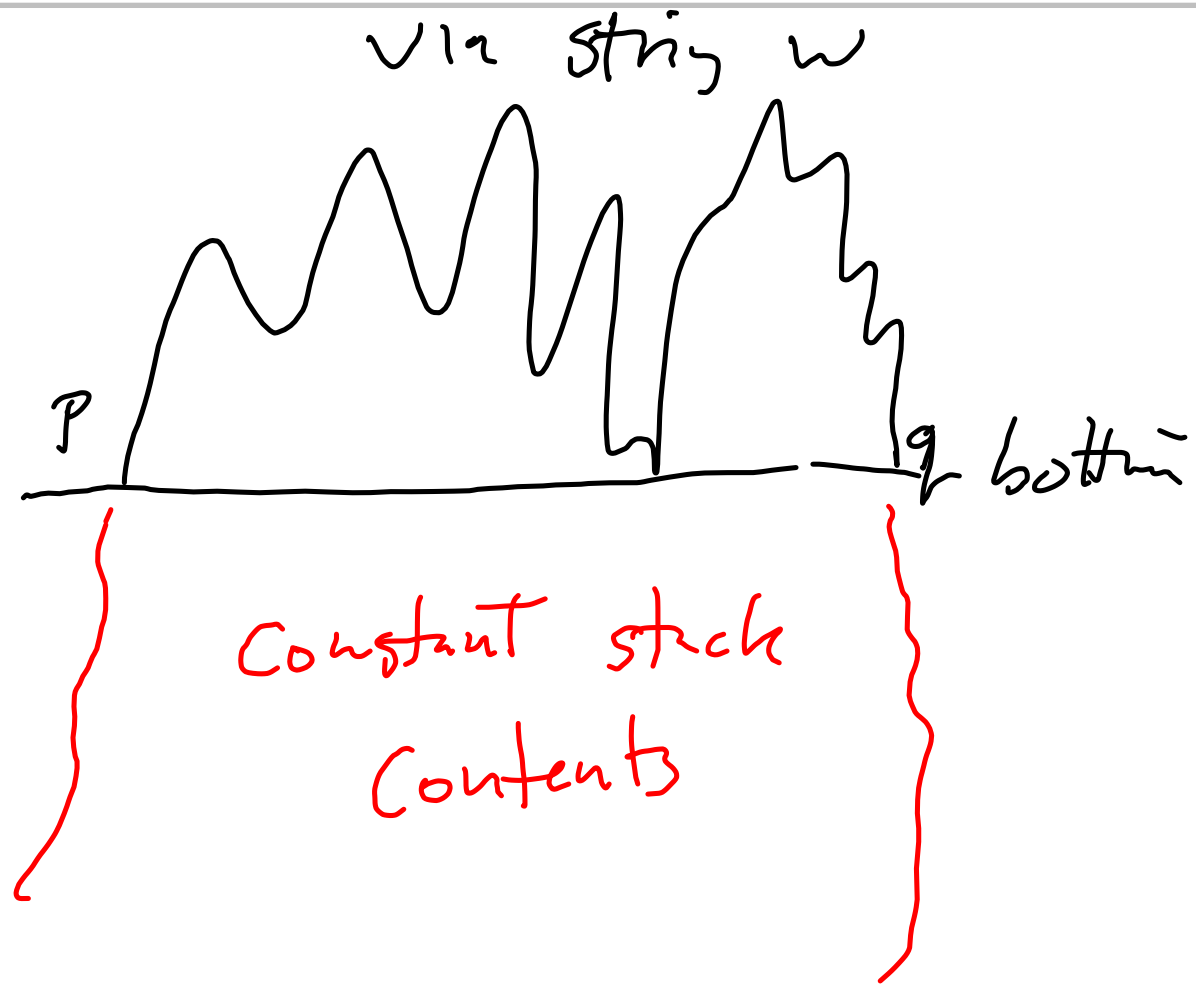
A_{pq}

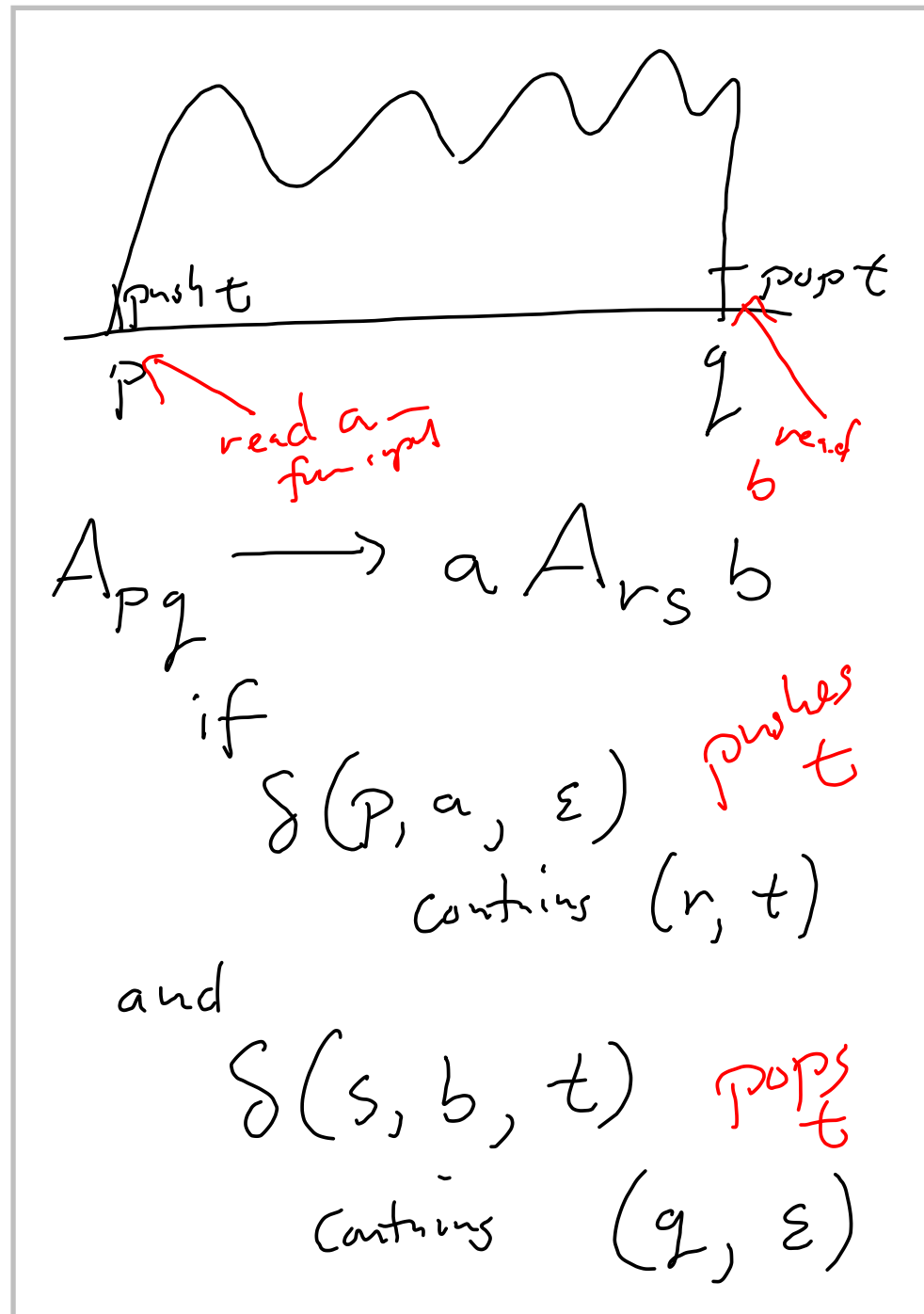
p, q are states in
the PDA

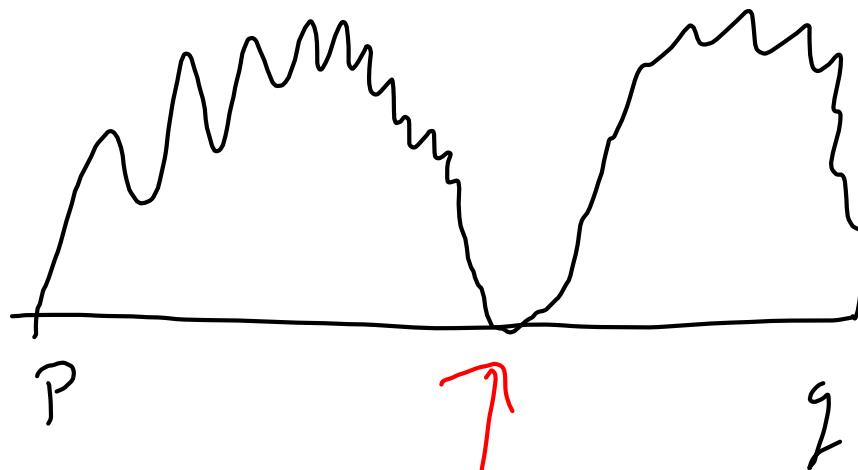
$A_{pq} \xRightarrow{*} w$ w all
terminals

Then $p \xrightarrow{w} q$
in PDA

with stack empty
at start & end
of this process







Stick down to
baseline

$$A_{Pq} \rightarrow A_{Pr} A_{rq}$$

also rules

$$A_{pp} \rightarrow \varepsilon$$

$$\left\{ \begin{array}{l} 0^i - 0^j = 0^k \quad i, j, k \geq 0 \\ \text{where } k = i - j \end{array} \right\}$$

$$" \quad 0000 - 00 = 00 \quad "$$

$$" \quad 000000 - 00000 = 0 \quad "$$

