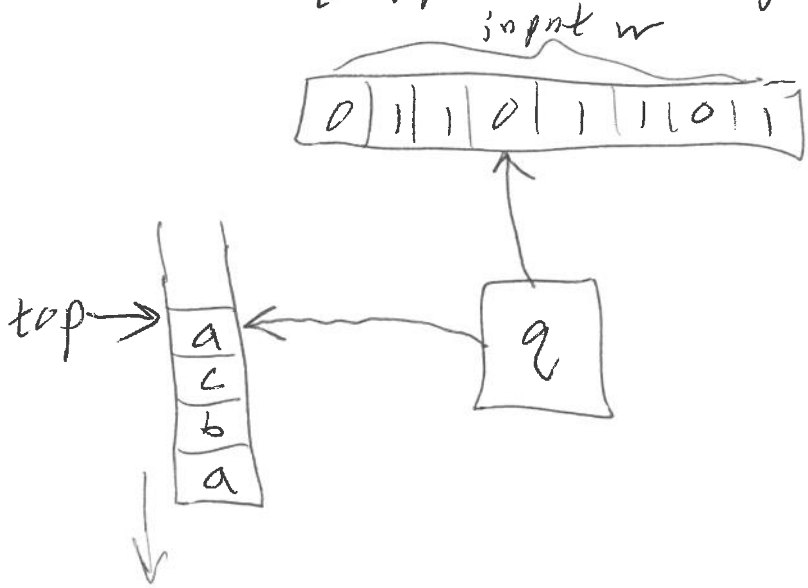
CSCE 355
3/17/2025

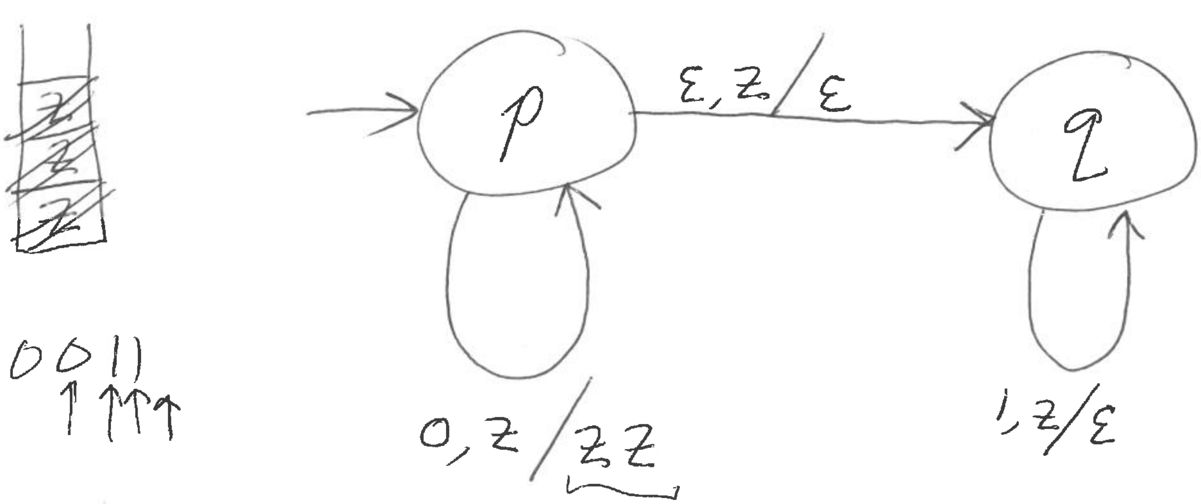
Pushdown Automata (PDAs) ①

A pushdown automaton (PDA) is a finite-state automaton equipped with a private stack.



<u>Ex</u>: $L = \{ 0^n 1^n : n \geq 0 \}$

A 2-state PDA recognizing L:



Stack empty here means "accept"

replace the top of stack with this

011 — rejected (can't read the whole input) ②

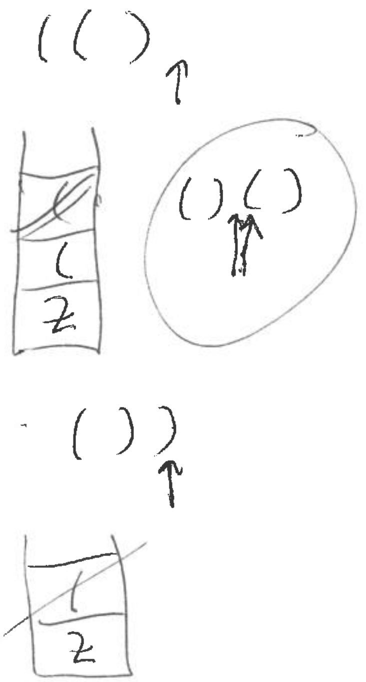
001 — rejected (can't empty the stack)

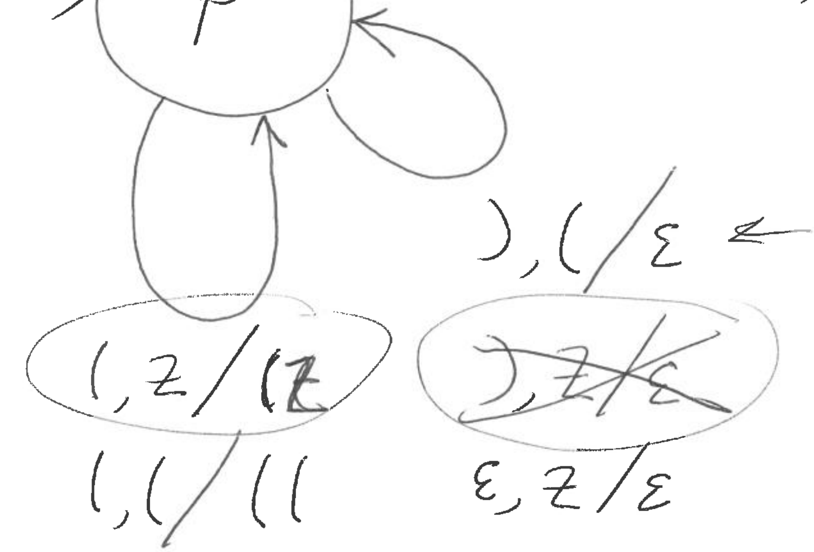
Ex: ~~Nested Blam~~ Balance parentheses

$\Sigma = \{ '(' , ')' \}$

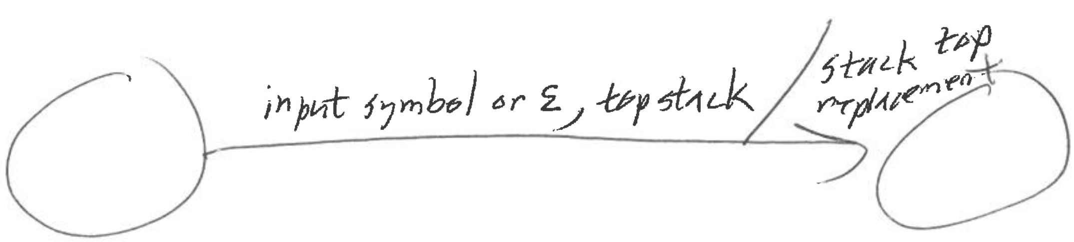
$B = \{ w \in \Sigma^* : w \text{ is well-balanced} \}$

stack alphabet $= \{ Z, '(' \}$





In ~~kno~~ any transition diagram



Ex: $\Sigma = \{0,1\}$

$L = \{w \in \Sigma^* : w$ has equal number of zeroes and ones$\}$

stack alphabet $= \{Z, +, -\}$

stack is either

$++\cdots+Z$] more 1s than 0s read so far

or

$--\cdots-Z$] more 0s than 1s

or

Z] equal number of 0s, 1s

$0, Z / -Z$ ("push −")

$0, - / --$ " "

$1, Z / +Z$

$1, + / ++$

$0, + / \varepsilon$

$1, - / \varepsilon$

$\varepsilon, Z / \varepsilon$

Formalities

Def: A pushdown automaton (PDA) is a tuple $\langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$ where

Q is a finite set (elements are states)

Σ and Γ are alphabets

(the input and stack alphabets, respectively)

$q_0 \in Q$ (the start state)

$Z_0 \in \Gamma$ (the bottom stack marker)

$F \subseteq Q$ (the set of accepting states)

$$\delta : \underbrace{Q}_{\text{current state}} \times \underbrace{(\Sigma \cup \{\varepsilon\})}_{\text{input to consume}} \times \underbrace{\Gamma}_{\text{current stack top}} \longrightarrow \underbrace{2^{Q \times \Gamma^*}}_{\substack{\text{some set of pairs of the} \\ \text{form } (q, \gamma) \\ q \in Q \\ \gamma \in \Gamma^*}}$$

$$q \xrightarrow{a,\, X \,/\, \gamma} r$$

means $(r, \gamma) \in \delta(q, a, X)$

$(q, r \in Q,\ X \in \Gamma,\ a \in \Sigma \cup \{\varepsilon\},\ \gamma \in \Gamma^*)$.

<u>Def</u>: Let $P = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$ be a PDA. A <u>configuration</u> or <u>instantaneous description</u> of P is a triple $\langle q, w, \gamma \rangle$ where $q \in Q$, $w \in \Sigma^*$, and $\gamma \in \Gamma^*$.

<u>Idea</u>: $\langle q, w, \gamma \rangle$ gives complete information about the possible future behavior of P during a computation:

q — the current state of P

w — unread portion of the input

γ — the current stack contents (top to bottom)

1st symbol of γ is the current stack top.

Initial config
~~Given~~ P as above and input string x, the <u>initial configuration</u> of P on input x is

$$\langle q_0, x, Z_0 \rangle.$$

<u>Transitions</u>: Given a configuration $\cancel{\langle q, w, x\rangle}$ $\langle q, w, \gamma \rangle$

Say that $\langle q, w, \gamma \rangle$ <u>leads to</u> configuration $\langle r, x, \beta \rangle$ in one step if:

$w = ax$ for some $a \in \Sigma \cup \{\varepsilon\}$

$\gamma = y\gamma'$ some $y \in \Gamma$, $\gamma' \in \Gamma^*$

$\beta = \alpha\gamma'$ " $\alpha \in \Gamma^*$

such that

$$\langle r, \alpha \rangle \in \delta(q, a, y).$$

Then we write $\langle q, w, \gamma \rangle \Rightarrow \langle r, x, \beta \rangle$

or $\langle q, w, \gamma \rangle \vdash \langle r, x, \beta \rangle$

Say that $\langle r, x, \beta \rangle$ is a <u>successor</u> to $\langle q, w, \gamma \rangle$.

<u>Def</u>: Let P be as above, and $w \in \Sigma^*$ be any input string.

1) <u>P accepts w via empty stack</u> if there is a finite sequence of configurations

$$\langle q_0, w, Z_0 \rangle \vdash \cdots \vdash \langle q, \varepsilon, \varepsilon \rangle \quad \text{for some } q \in Q$$

2) <u>P accepts w via (final) accepting state</u> if there is a finite sequence

$$\langle q_0, w, Z_0 \rangle \vdash \cdots \vdash \langle q, \varepsilon, \gamma \rangle$$

where $q \in F$ and $\gamma \in \Gamma^*$.

$L(P) := \{ w \in \Sigma^* : P$ accepts w by ~~finite~~ final state$\}$

$N(P) := \{ w \in \Sigma^* : P$ accepts w via empty stack$\}$