PUSH - DOWN AUTOMATON.

Introdurremo un nuovo modello di computazione: il PDA. Mostreremo che è equivalente alle CFG. La differenza con NFA: La presenza di una pila. Possiamo scrivere simboli sulla pila e poi leggerli; possiamo anche manipolare la pila:

- "TOP": leggo il simbolo in cima senza rimuoverlo.
- "POP": Elimino il simbolo in cima.
- "PUSH": Inserisco nuovo simbolo in cima e spingo in giù gli altri.

Ci saranno due alfabeti: input Σ e pila Γ.
La funzione di transizione:

$$q_1 \xrightarrow{a;\, b \to c} q_2$$

b, c potrebbero essere ε.

Il dominio di δ sarà $Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon$. Il codominio sarà $\mathcal{P}(Q \times \Gamma_\varepsilon)$.

DEF (PDA) Un PDA è una tupla $(Q, \Sigma, \Gamma, \delta, q_0, F)$ dove Q, Σ, q_0, F sono come per gli NFA/DFA, mentre:

- Γ è l'alfabeto finito di pila
- $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \to \mathcal{P}(Q \times \Gamma_\varepsilon)$.

Vediamo le transizioni: $(q, c) \in \delta(p, a, b)$ dove $q, p \in Q$, $a \in \Sigma_\varepsilon$, $b, c \in \Gamma_\varepsilon$:

- $a, b, c \neq \varepsilon$: PDA legge a, b è sul top della pila, lo stato è p e si muove in q rimpiazzando b con c.
- $a, c \neq \varepsilon$, $b = \varepsilon$: sto facendo PUSH.
- $a, b \neq \varepsilon$, $c = \varepsilon$: sto facendo POP.
- $a \neq \varepsilon$, $b, c = \varepsilon$: sto facendo TOP.
- $a = \varepsilon$: Non determinismo come nei NFA.

Accettazione: Come computa una PDA? Sia $w = w_1 \dots w_m$ l'input con $w_i \in \Sigma$

Attraverserò stati $r_0 \dots r_m$ e ci saranno stringhe $s_0 \dots s_m$ t.c.:

- $r_0 = q_0$ ed $s_0 = \varepsilon$ (pila vuota).
- $\forall\, i = 0, \dots, m$ $(r_{i+1}, a) \in \delta(r_i, w_{i+1}, b)$
  dove $s_i = bt$ e $s_{i+1} = at$ con $a, b \in \Gamma_\varepsilon$.
  $t \in \Gamma^*$
- $r_m \in F$

Relazione di transizione:

$$(p, ax, by) \vdash (q, x, cy)$$

se $(q, c) \in \delta(p, a, b)$.

$p, q \in Q$; $b, c \in \Gamma_\varepsilon$, $a \in \Sigma_\varepsilon$, $x \in \Sigma^*$, $y \in \Gamma^*$

la sua chiusura $\vdash^*$ definisce la relazione di transizione estesa come per gli NFA.

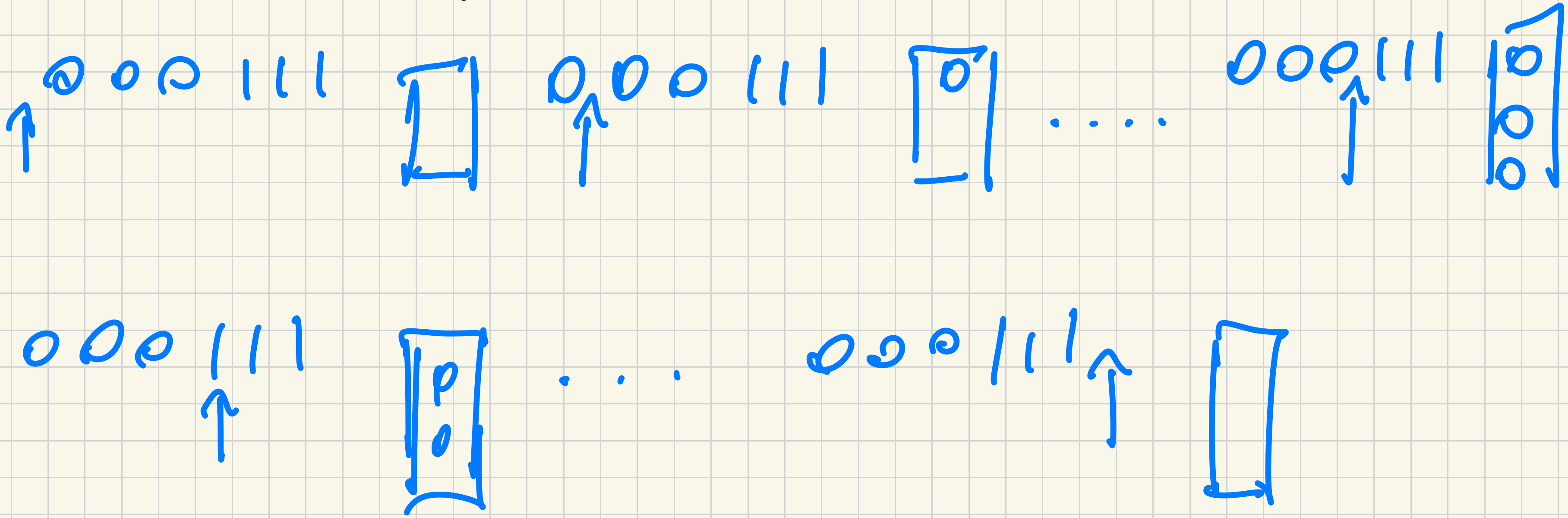
Ovvero per PDA P:

$$L(P) = \{ w \in \Sigma^* : (q_0, w, \varepsilon) \vdash_P^* (q, \varepsilon, \gamma) \quad q \in F;\ \gamma \in \Gamma^* \}$$

Definizione (equivalente) ma più ordinata è quella in cui sostituisco (q, ε, γ) con $(q, \varepsilon, \varepsilon)$ per $q \in F$.

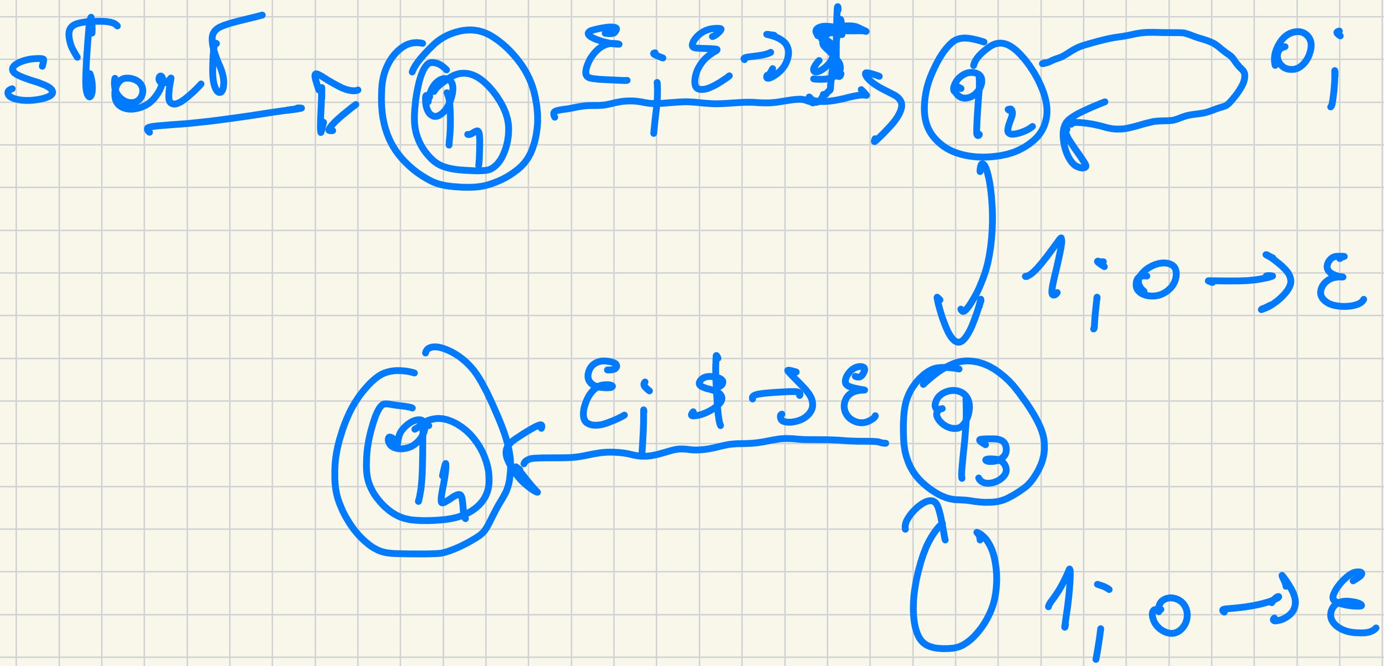
Esempio: Riconoscere $L = \{ 0^n 1^n : n \geq 0 \}$ con un PDA.

Idea: Supponiamo $w = 000111$. Per ogni 0 che leggo aggiungo 0 nello stack. Per ogni 1 che leggo faccio pop di 0 nello stack.



$Q = \{q_1, q_2, q_3, q_4\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{0, \$\}$

$F = \{q_1, q_4\}$.



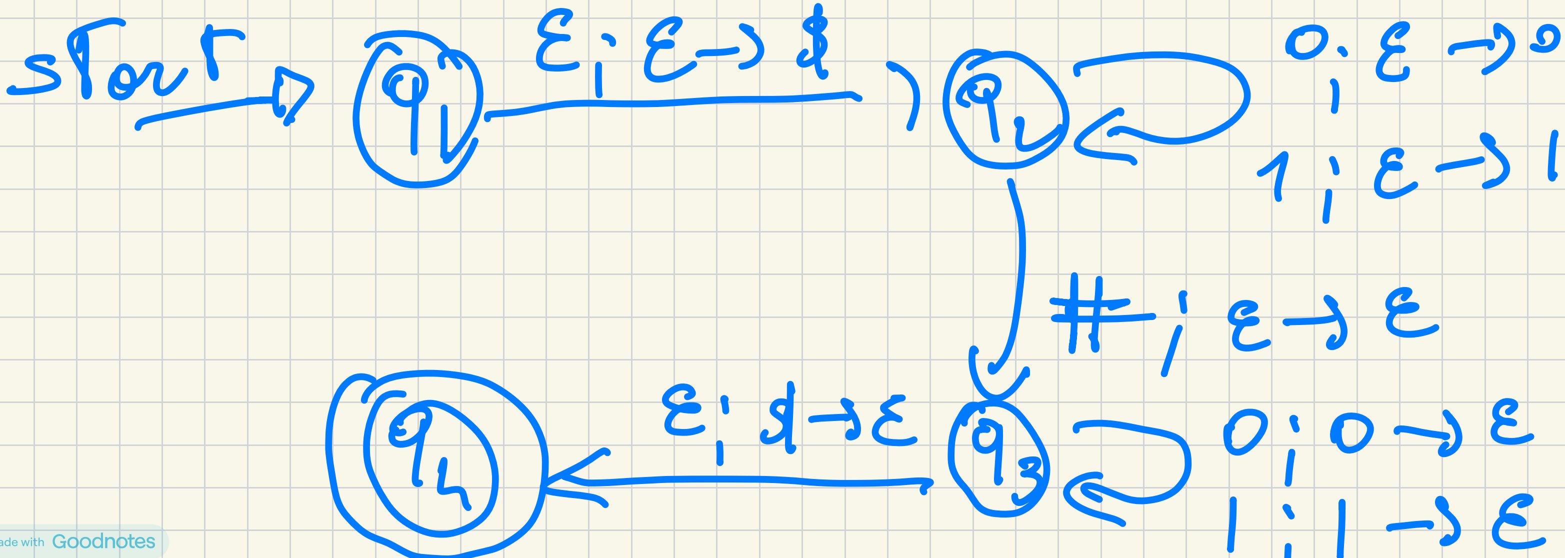
000111

000010

00011

0001111

Esempio: $L = \{ w \# w^R : w \in \{0,1\}^* \}$.



1100#1100

0
0
1
1
$

Equivalenza tra PDA e CFG.

TEO. Un linguaggio è ACONTESTUALE se e solo se esiste un PDA che lo riconosce.

LEMMA $(\Rightarrow)$.

DIM. Sia $L = L(G)$ un linguaggio ACONTESTUALE.

per qualche CFG G. Devo costruire PDA P t.c. P accetta w sse $w \in L(G)$.

Idea: P userà il non determinismo per derivare w usando tutte le possibili derivazioni usando le regole di G. Poi proverà a fare MATCH dei terminali nella pila con l'input.

Vediamo un piccolo esempio. Il PDA:

$$S \to V_1 \, a \, V_2$$

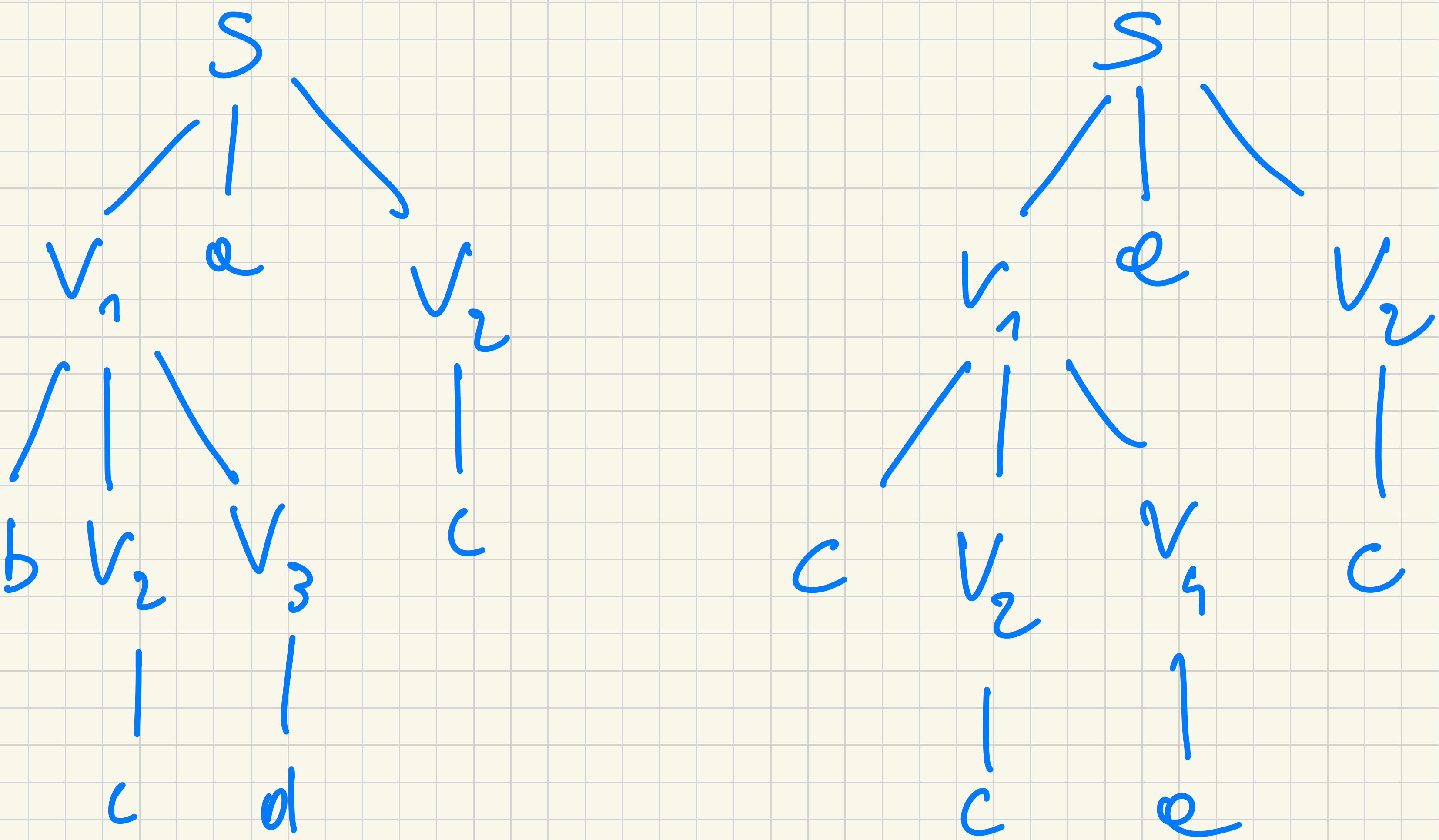
$$V_1 \to b \, V_2 \, V_3$$

$$V_1 \to c \, V_2 \, V_4$$

...

$$\begin{array}{|c|} S \\ \$ \\ \hline \end{array} \longrightarrow \begin{array}{|c|} V_1 \\ a \\ V_2 \\ \hline \end{array}$$

e così via sostituisco V_1 e V_2.



Ci possono essere più alberi di derivazione, per cui useremo il NON DETERMINISMO.

Pseudocodice di P:

- Inserisce $ nella pila
- Ripete:

*) Se in cima alla pila c'è VARIABILE A, sceglie non-det. una delle regole $A \to \dots$ in G e sostituisce A con l'espressione a DX della regola.

*) Se in cima c'è un TERMINALE a, lo tira fuori e lo confronta con il prossimo carattere di input w.
Se sono uguali continua; altrimenti

rifiuta.

*) Se un cammino c'è e accetta è perché che abbia letto tutto w.

Più formalmente, $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$

Notazione: $(r, u) \in \delta(q, a, s)$ quando

(i) s è in cima alla pila.

(ii) q è lo stato in cui P è

(iii) P legge a e va in r inserendo nello stack $u = u_1 \dots u_\ell$ (è una stringa).

Questo è WLOG:

$$q \xrightarrow{a;\ s \to xyz} r$$

$$q \xrightarrow{a;\ s \to z} q_1 \xrightarrow{\varepsilon;\ \varepsilon \to y} q_2 \xrightarrow{\varepsilon;\ \varepsilon \to x} r$$

$$Q = \{q_{start},\ q_{loop},\ q_{acc}\} \cup E$$

↳ stati intermedi

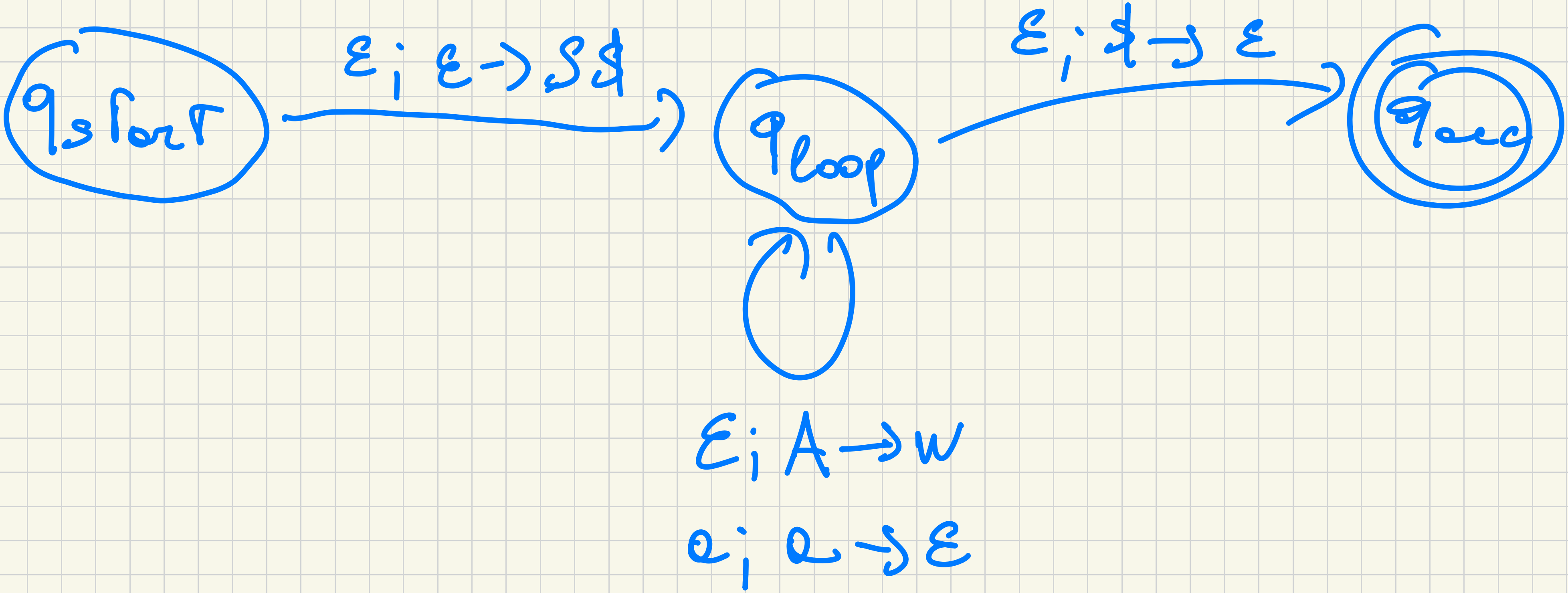
Le Transizioni:

*) $\delta(q_{start}, \varepsilon, \varepsilon) = \{(q_{loop}, S\$)\}$

*) $\delta(q_{loop}, \varepsilon, A) = \{(q_{loop}, w) : w \text{ t.c. } A \to w \in G\}$

*) $\delta(q_{loop}, a, a) = \{(q_{loop}, \varepsilon)\}$

*) $\delta(q_{loop}, \varepsilon, \$) = \{(q_{acc}, \varepsilon)\}$

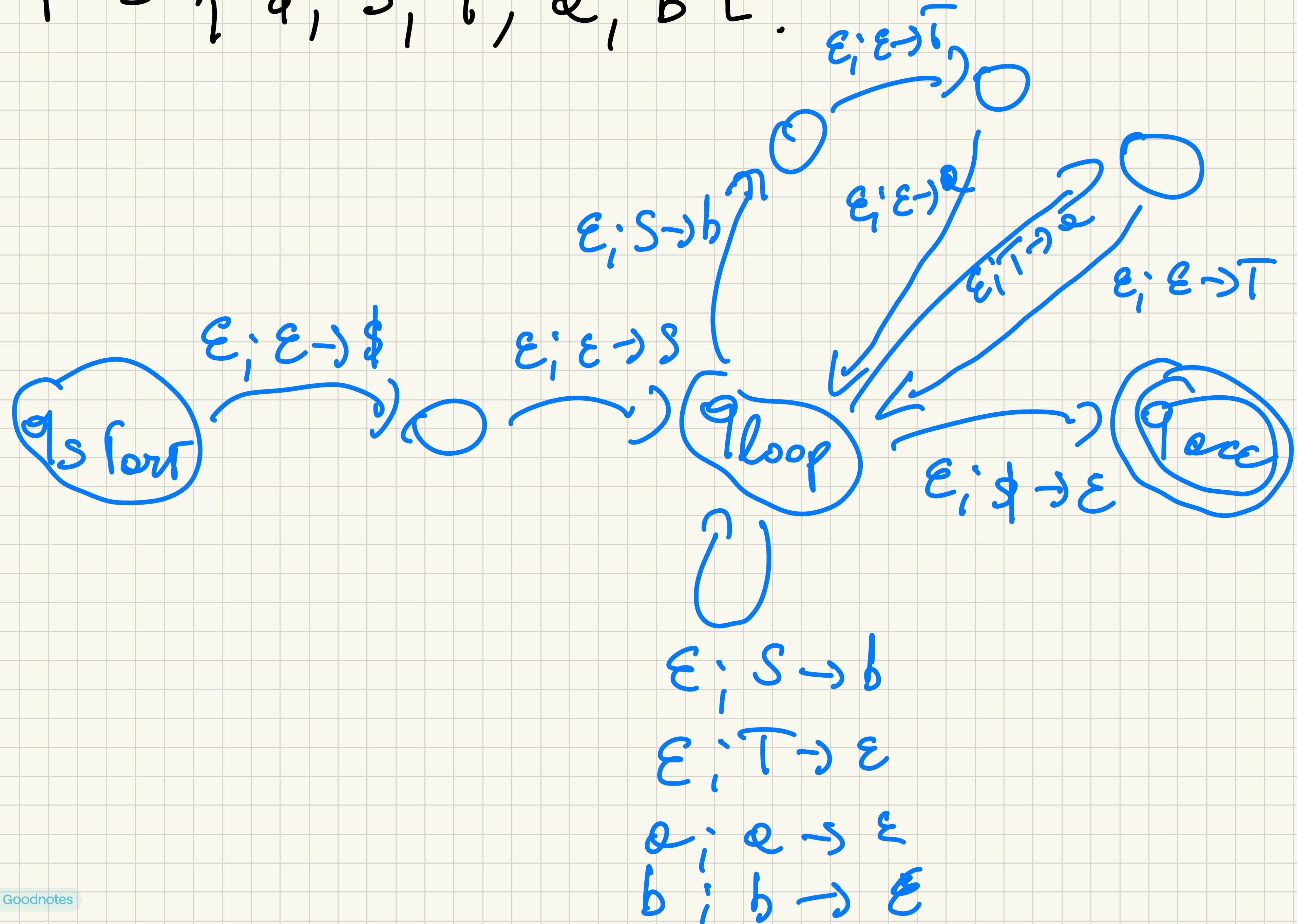


ESEMPIO. Sia G:

$$S \to aTb \mid b$$

$$T \to Ta \mid \varepsilon$$

Il PDA equivalente avrà $\Sigma = \{a, b\}$
$\Gamma = \{\$, S, T, a, b\}$.



LEMMA ($\Longleftarrow$)

Dato PDA P devo costruire grammatica CFG G t.c. $L(G) = L(P)$.

Idea: Per ciascuna coppia di stati p, q in P, avrò VARIABILE A_{pq} che genera tutte le stringhe che portano P da p a q con pila vuota.

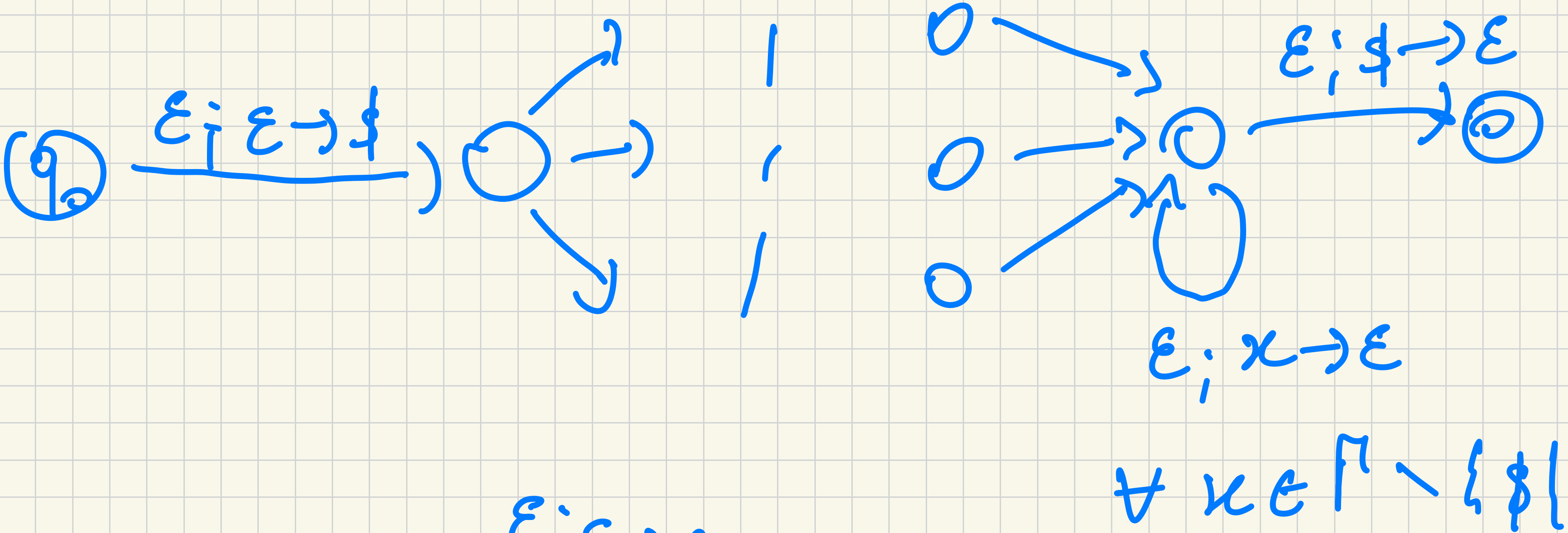
Forma canonica:

1) Il PDA inizia con pila vuota e finisce con pila vuota.

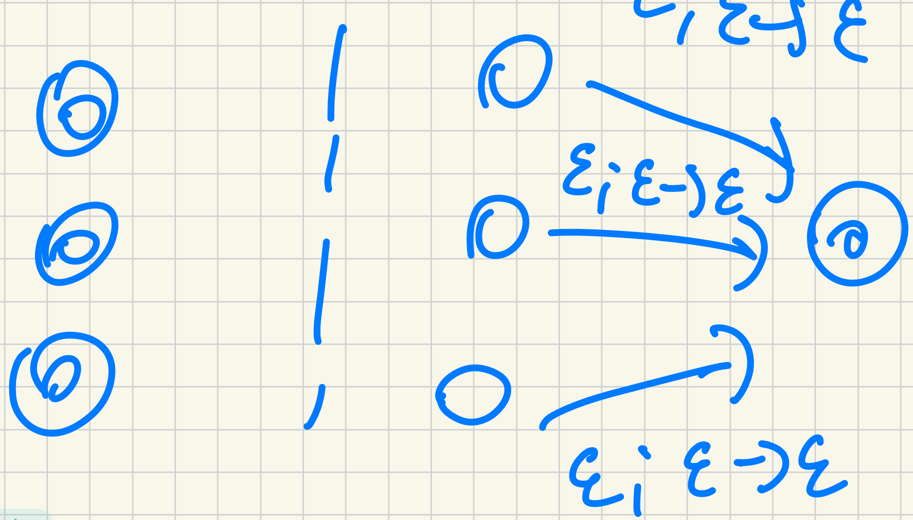
2) Il PDA ha un solo stato accettante

3) Il PDA elimina oppure inserisce nella pila (ma non entrambe allo stesso tempo).

1)



2)



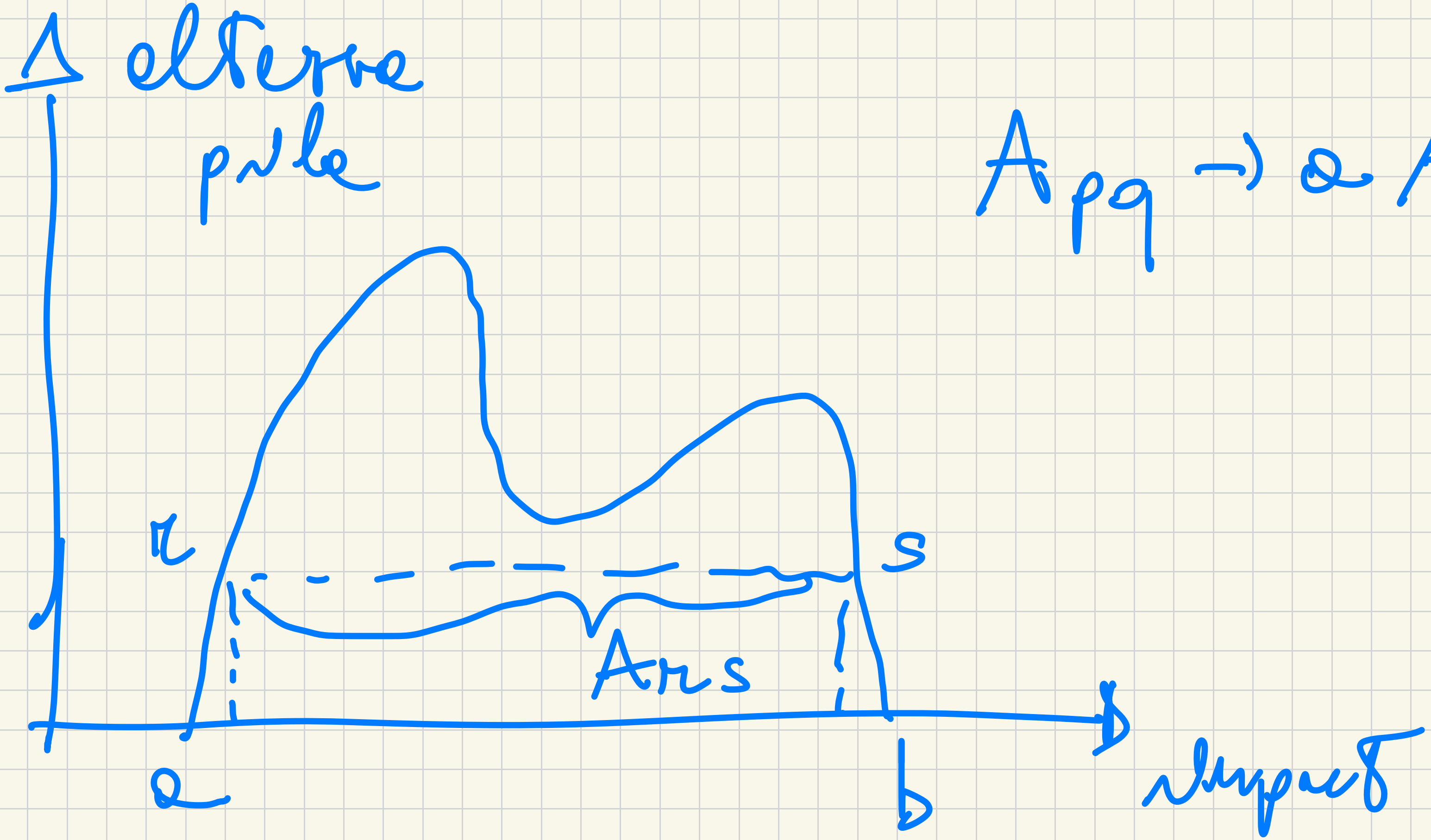
3) $\bigcirc \xrightarrow{a;\, x \to y} \bigcirc \quad \Big| \quad \bigcirc \xrightarrow{a;\, x \to \varepsilon} \bigcirc \xrightarrow{\varepsilon;\, \varepsilon \to y} \bigcirc$

Le variabili A_{pq} generano le stringhe che fanno andare P da p a q con pila vuota.

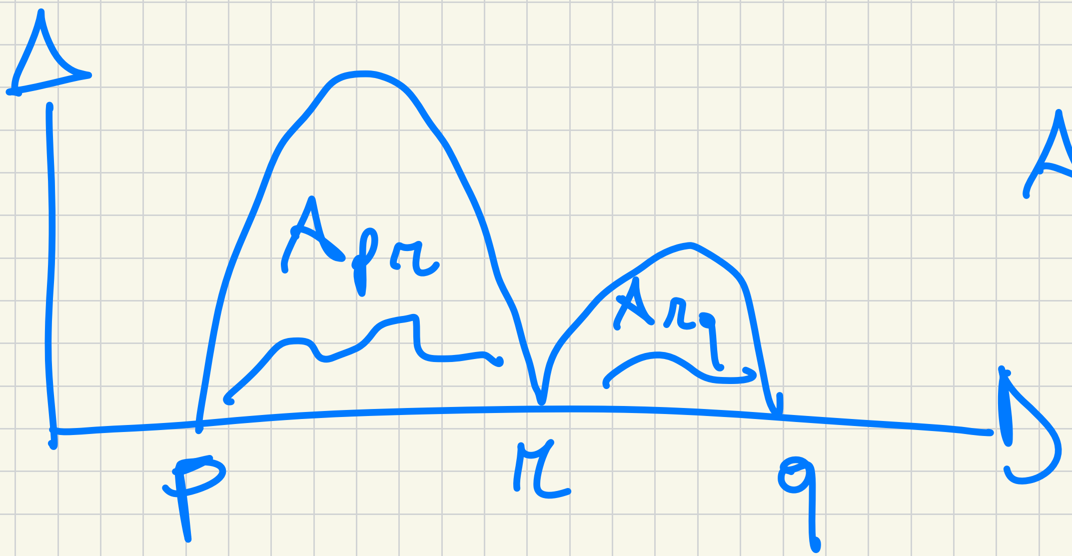
$$A_{pq} \to a A_{rs} b$$

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

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



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



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