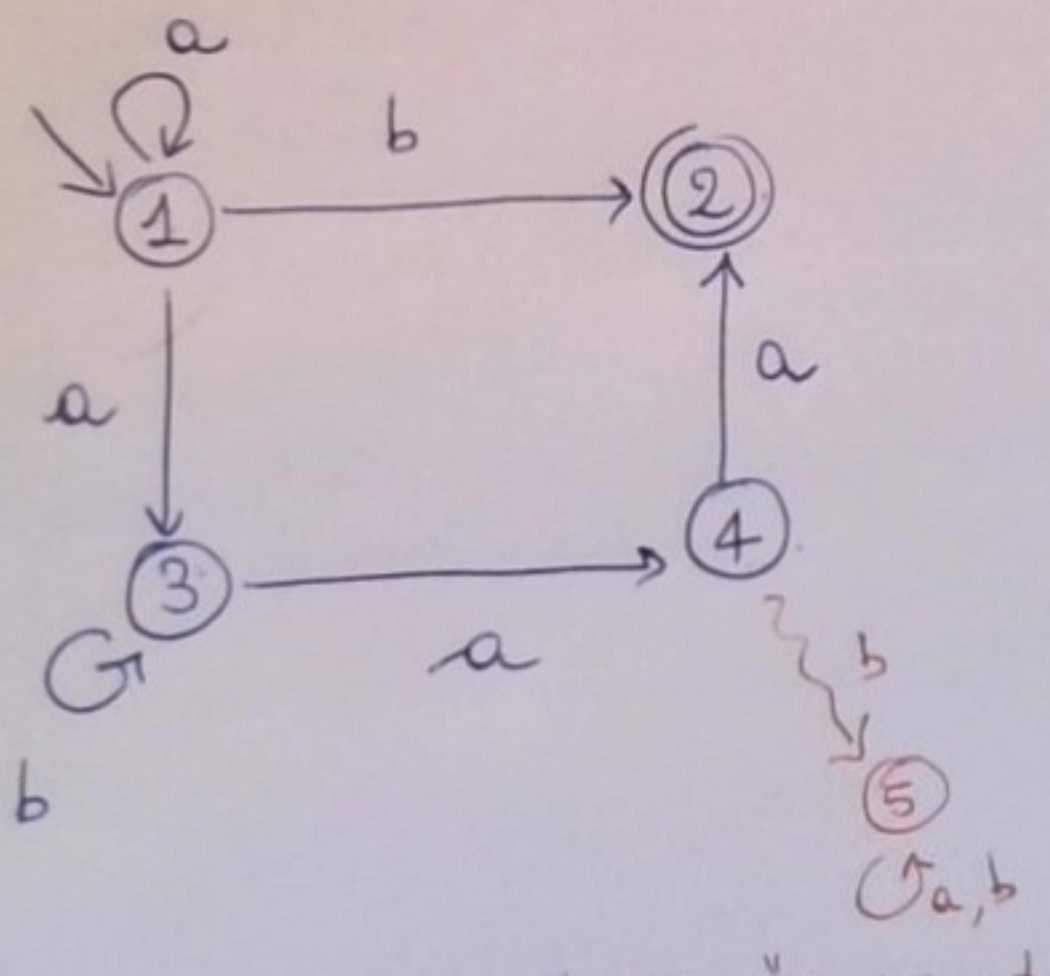
Non-determinismo e Grammatiche generative

Es Data l'ASFND A, trovarne uno

equivalente A' che sia deterministico

$\mathcal{L}(A) = \mathcal{L}(A')$

A :



pozzo "nascosto"

non è deterministico:

$\tau(1, a) = \{1, 3\}$

non è completo:

$\tau(4, b) = \phi$

Sappiamo che esiste di sicuro A' <u>det.</u> tc $\mathcal{L}(A) = \mathcal{L}(A')$,

però A' potrebbe avere "molti" più stati di A.

"insieme delle parti"

A' si può costruire tramite la "power-set construction"

gli stati di A' hanno per ~~stato~~ nome i sottoinsiemi dell'insieme degli stati di A.

Oss $\mathcal{L}(A) = a^*b + a^+b^*aa$

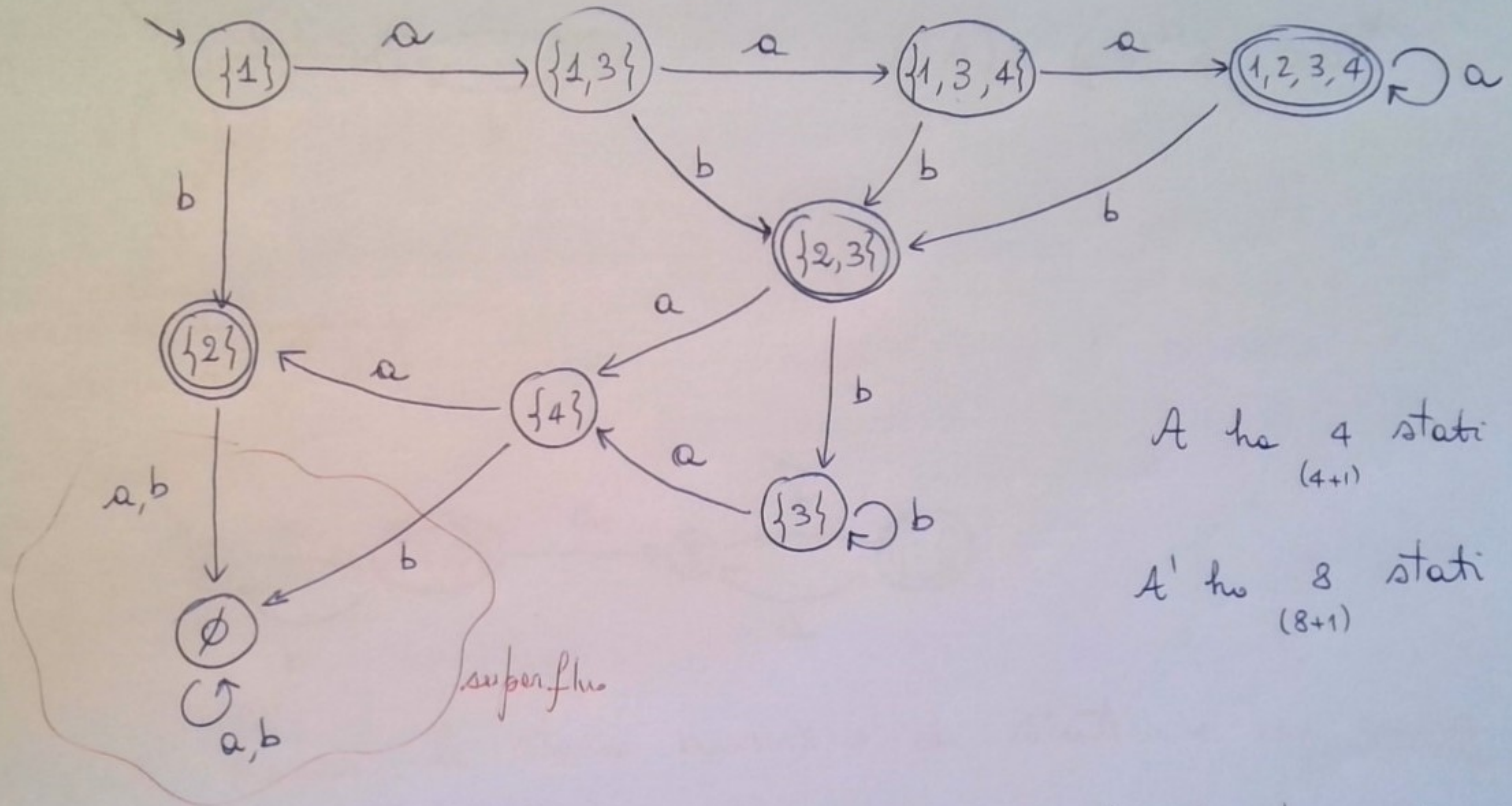
A è non-det.: una stessa stringa può "etichettare" percorsi diversi, ad es.

aab: $\searrow 1 \xrightarrow{a} 1 \xrightarrow{a} 3 \xrightarrow{b} 3$ sequenza iniziale ma non finale

$\searrow 1 \xrightarrow{a} 1 \xrightarrow{a} 1 \xrightarrow{b} 2$ sequenza iniziale e finale

Allora $aab \in \mathcal{L}(A)$ poiché esiste almeno una sequenza iniziale e finale etichettata da aab.

A'



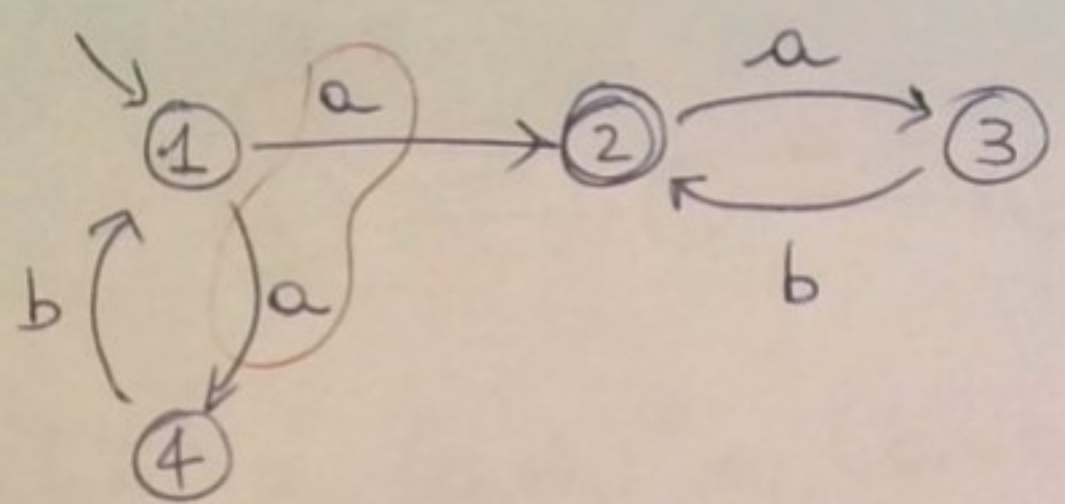
A ha 4 (4+1) stati

A' ha 8 (8+1) stati

Non è detto che A' sia, tra gli ASF<u>D</u> equivalenti ad A, quello con meno stati.

<u>Es</u> Idem con

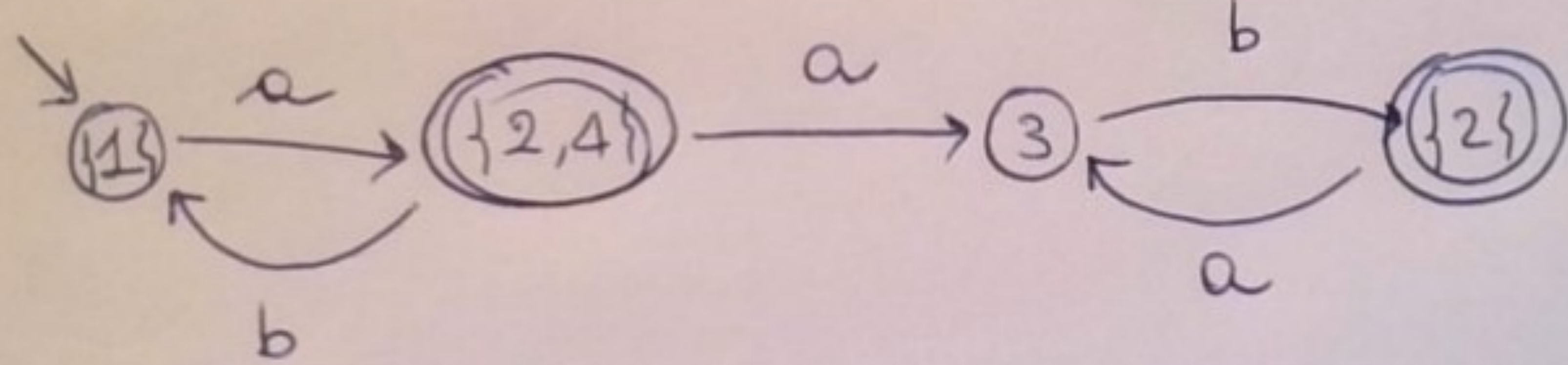
A



$\mathcal{L}(A) = (ab)^* a (ab)^*$

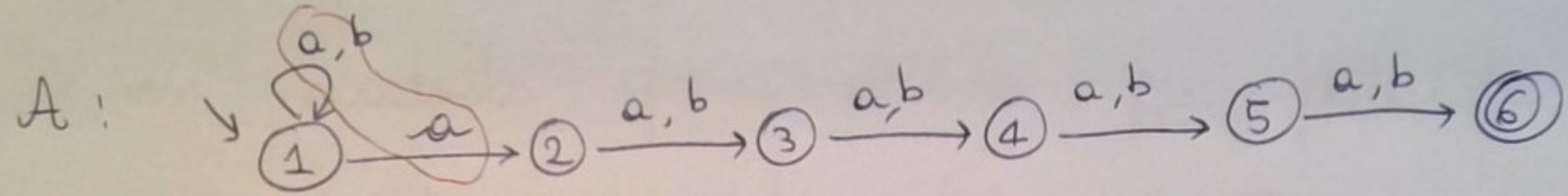
(evito di aggiungere pozzi in A e in A')

A'



A e A' hanno lo stesso numero di stati: è un <u>caso</u>.

Compito Provate con



$\mathcal{L}(A) = (a+b)^* a\, (a+b)^4$ cioè le stringhe la cui quintultima lettera è 'a'.

Oss Gli automi a pila non deterministici sono un modello operazionale più potente degli automi a pila deterministici

Cioè, ci sono linguaggi riconosciuti da APND che non sono riconosciuti da nessun APD.

Sia $\Sigma = \{a, b\}$ e siano $L, L' \subseteq \{a, b\}^*$ due linguaggi.

Supponiamo che $L \subseteq L'$ e che non esista alcun APD che riconosce L, ma che ce ne sia uno non det.

Cosa si può dire di L'? NIENTE

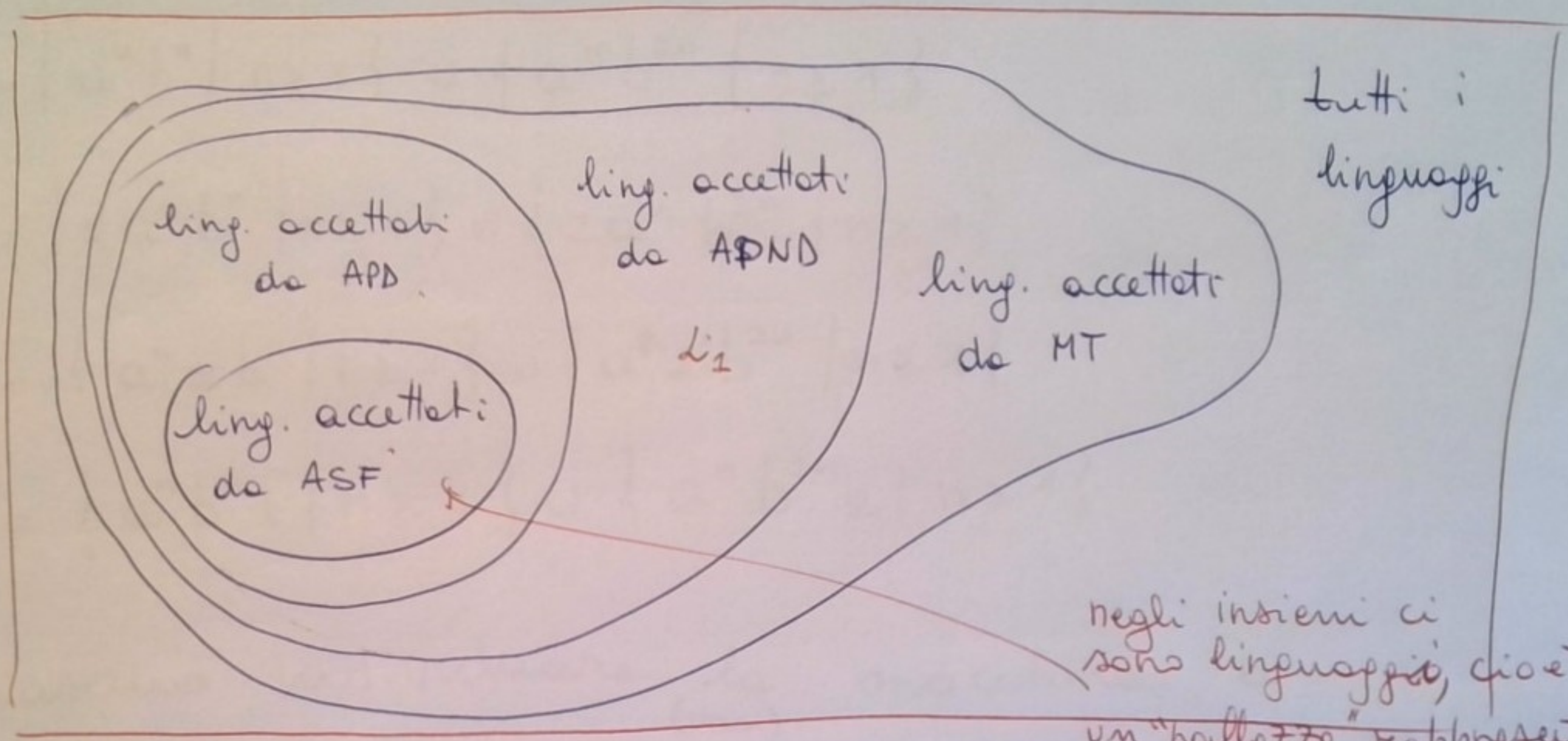
$\mathcal{L}'$ potrebbe essere Σ^*, che è regolare.
Oppure $\mathcal{L}'$ potrebbe un linguaggio molto complesso per il quale ci vuole una MT.

In generale, ~~ogni~~ ogni linguaggio $\mathcal{L}$ su Σ^* è tc

$$\phi \subseteq \mathcal{L} \subseteq \Sigma^*$$

e ϕ e Σ^* sono regolari. Quindi <u>ogni</u> linguaggio <u>contiene</u> ed è <u>contenuto</u> in un linguaggio regolare, indipendentemente da quanto è "complicato".

Sui libri a volte si trova questo diagramma



<u>NON</u> significa che se $L \subseteq L'$ e L' è accettato da un APND, allora L è di sicuro accettato da un APND.

<u>Es</u> Consideriamo i linguaggi

$$L_1 = \{a^n b^n \mid n \geq 1\} \cup \{a^n b^{2n} \mid n \geq 1\}$$

$$L_2 = \{1 a^n b^n \mid n \geq 1\} \cup \{2 a^n b^{2n} \mid n \geq 1\}$$

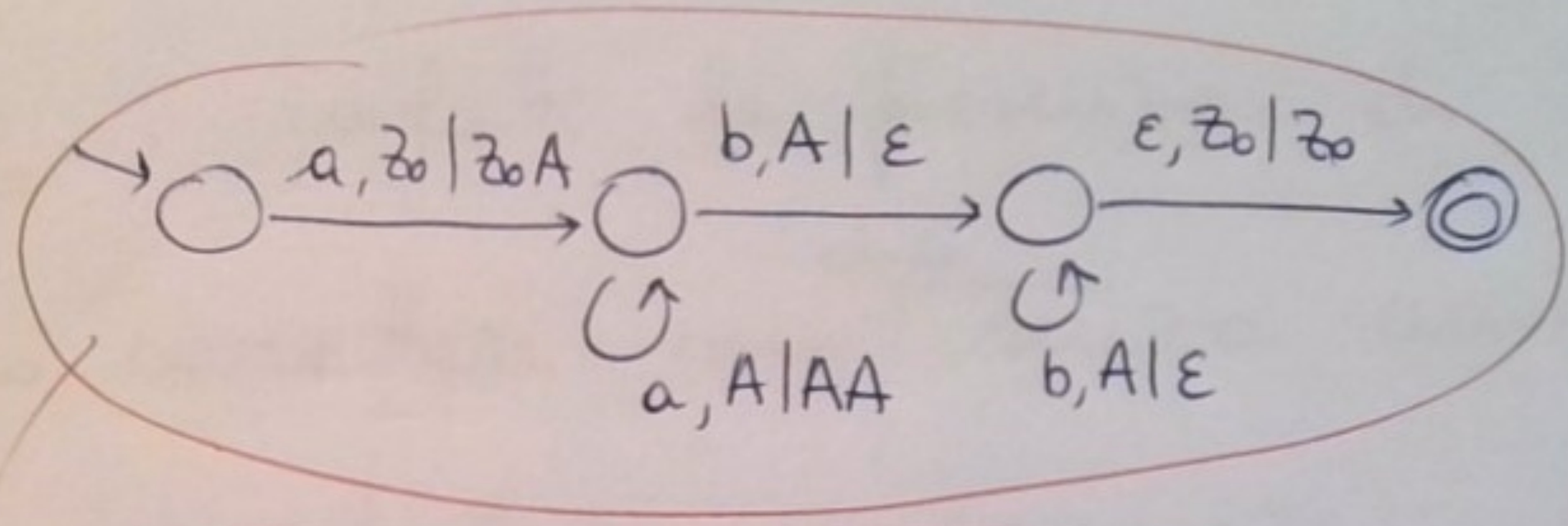
$$L_3 = \{a^n 1 b^n \mid n \geq 1\} \cup \{a^n 2 b^{2n} \mid n \geq 1\}$$

$$L_4 = \{a^n b^n 1 \mid n \geq 1\} \cup \{a^n b^{2n} 2 \mid n \geq 1\}$$

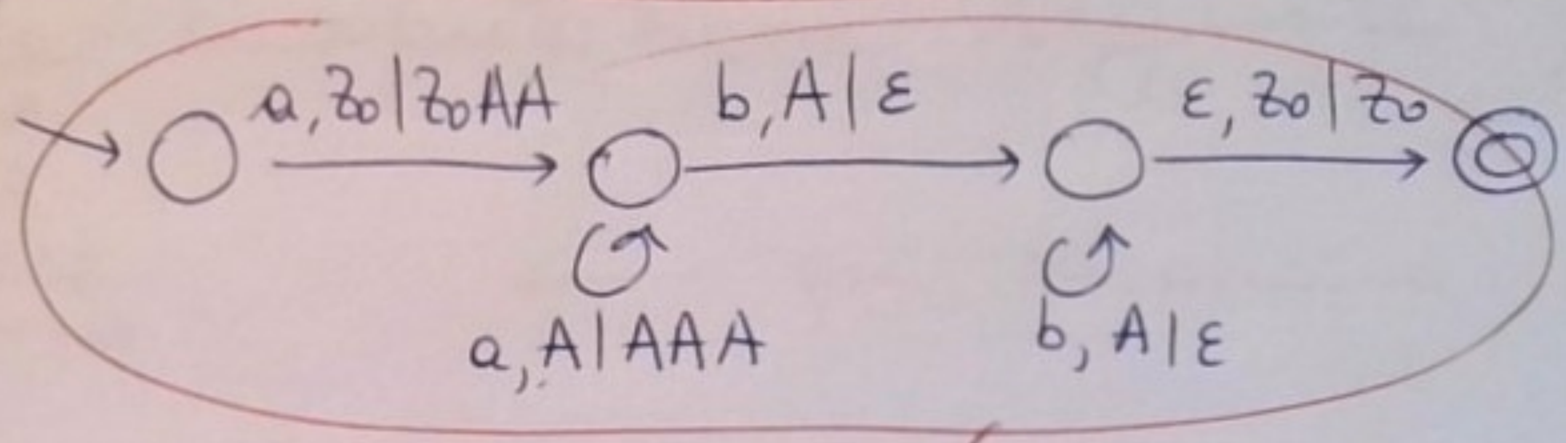
Per ciascuno individuare la (una) macchina a "potenza minima" che lo riconosce.

$\mathcal{L}_1$: è unione di <u>due</u> linguaggi che sappiamo essere riconosciuti da AP<u>D</u>

$\{a^n b^n \mid n \geq 1\}$:

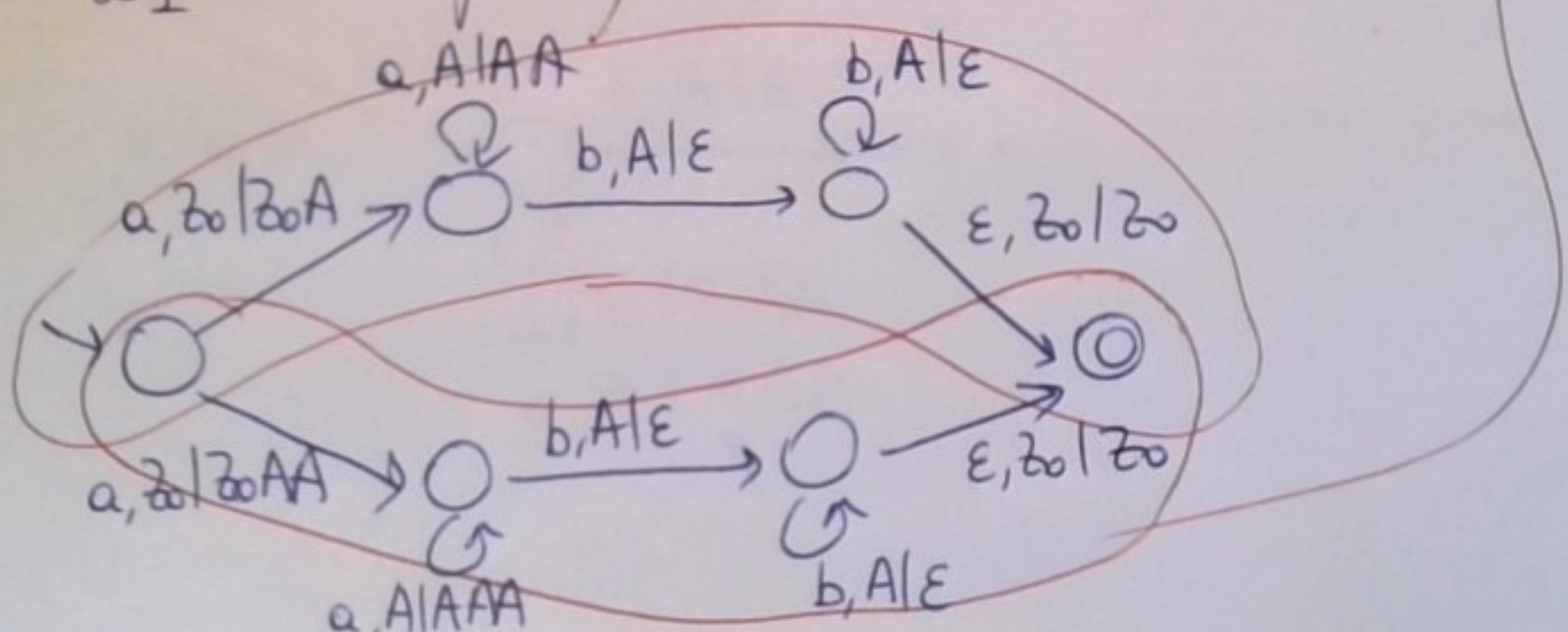


$\{a^n b^{2n} \mid n \geq 1\}$:



tutti e due deterministici

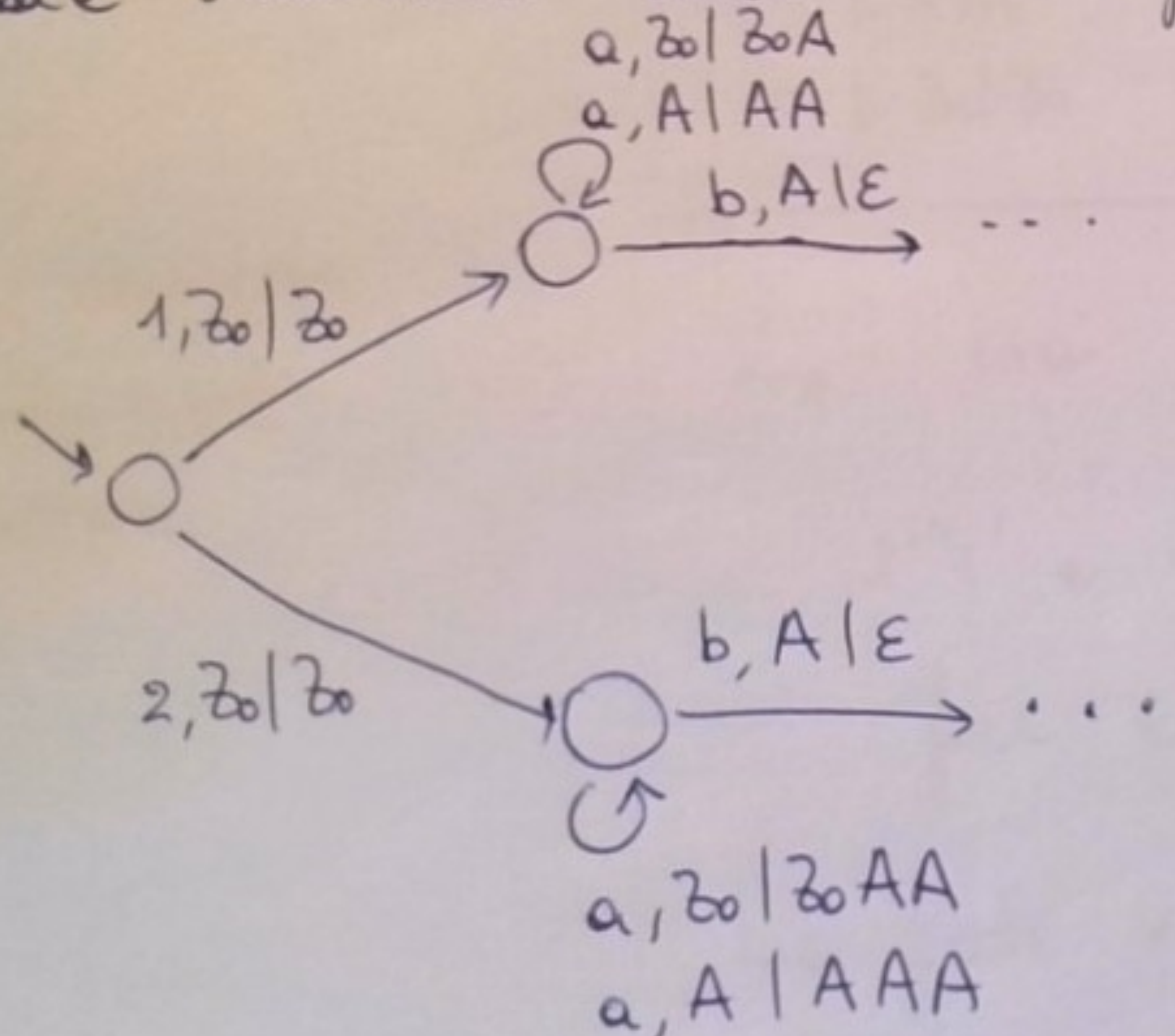
Allora $\mathcal{L}_1$ si può riconoscere con un AP<u>ND</u> :



Per L_2 vale "più o meno" lo stesso che per L_1,

cioè si <u>può</u> riconoscere L_2 con un <u>APND</u>

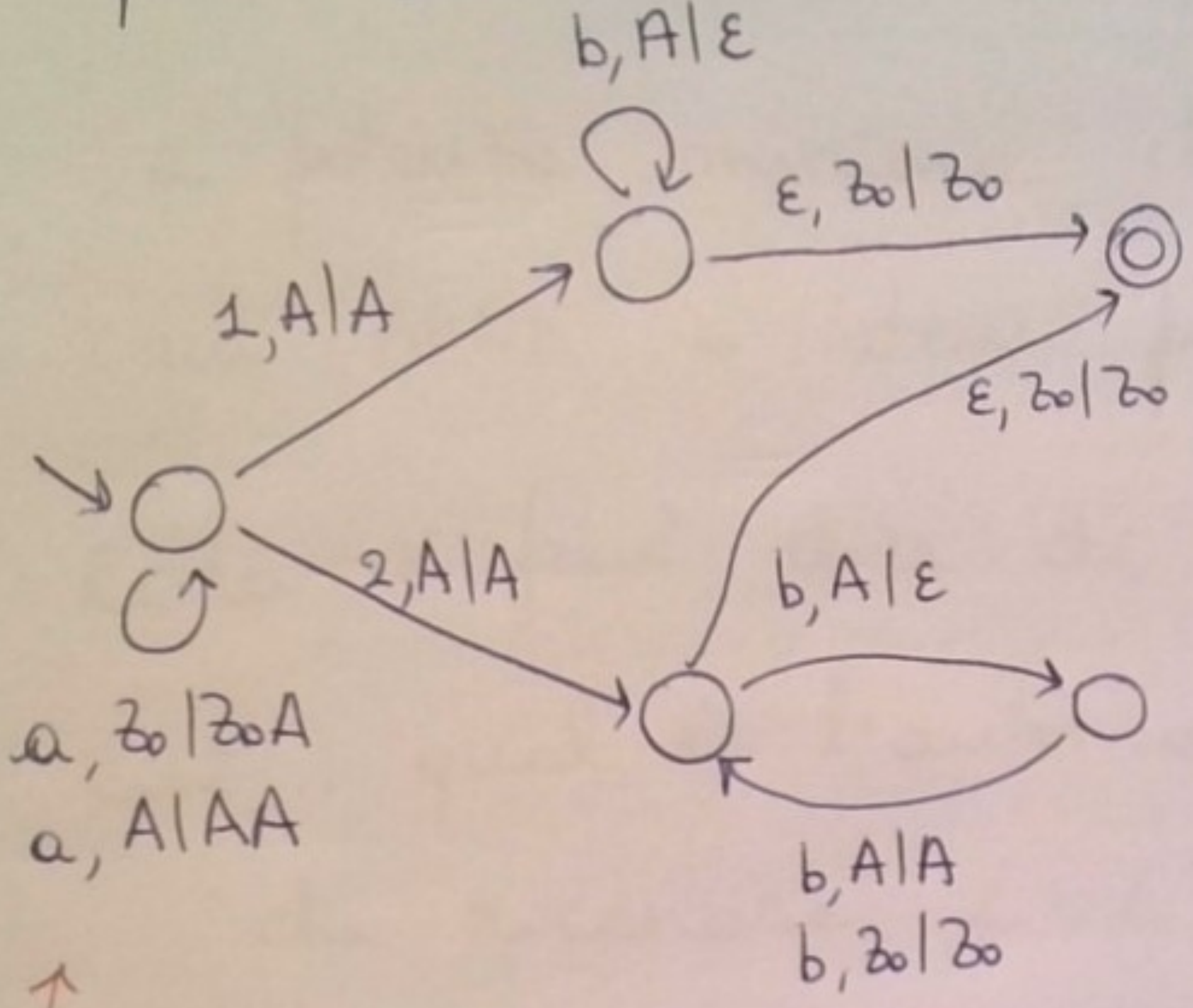
Ma <u>basta</u> un <u>APD</u>, infatti la presenza di '1' o '2' come primo carattere mi dice senza ambiguità quale ramo della computazione seguire.



← qui riconosco $1a^n b^n$

← qui $2a^n b^{2n}$

Idea per $\mathcal{L}_3$:



impilo sempre un solo nuovo carattere

$a^n b^n c^m d^m$

$(a+b)^n (c+d)^n$

è ancora deterministico

$\{a^n b^m c^m d^n \mid n, m \geq 0\}$

$\cup$

$\{a^n b^n c^n d^n \mid n \geq 0\}$

$\mathcal{L}_4$: ora serve proprio un APN$\underline{\underline{D}}$,

l''1' e il '2' arrivano troppo tardi

per essere utili.

Lo fate voi.

Es Siano $L, L' \subseteq \{a,b\}^*$ t.c. l'automa a potenza minima che riconosce L è un APND e idem per L'.

Cosa si può dire di $L \cup L'$? e di $L \cap L'$?

Cioè: qual è l'automa a potenza minima che riconosce $L \cup L'$? e $L \cap L'$?

Di sicuro per $L \cup L'$ si può fare con un AP<u>ND</u>, ma si può fare con qualcosa di più semplice?

Potrebbe bastare un ASF: ad esempio se L' è il complementare di L si ha

$L \cup L' = \Sigma^*$ e $L \cap L' = \phi$ che sono entrambi regolari

<u>Grammatiche</u> : sono un modello <u>NON</u> operazionale

inerentemente <u>non</u> deterministico

$G = (V_N, V_T, P, S)$

V_N, V_T sono insiemi di simboli, $V_N \cap V_T = \emptyset$, e $S \in V_N$.

P è un insieme di "produzioni", cioè regole di riscrittura del tipo

$$\alpha \longrightarrow \beta \quad \text{dove} \quad \alpha \in V_N^+ , \; \beta \in (V_N \cup V_T)^*$$

In base al tipo di produzione, classifichiamo le grammatiche.

I non terminali servono a "simulare" gli stati di A, quindi uso S per lo stato 1, e poi ancora 2, 3, 4.

$V_T = \{a, b\}$, $V_N = \{S, 2, 3, 4\}$, P?

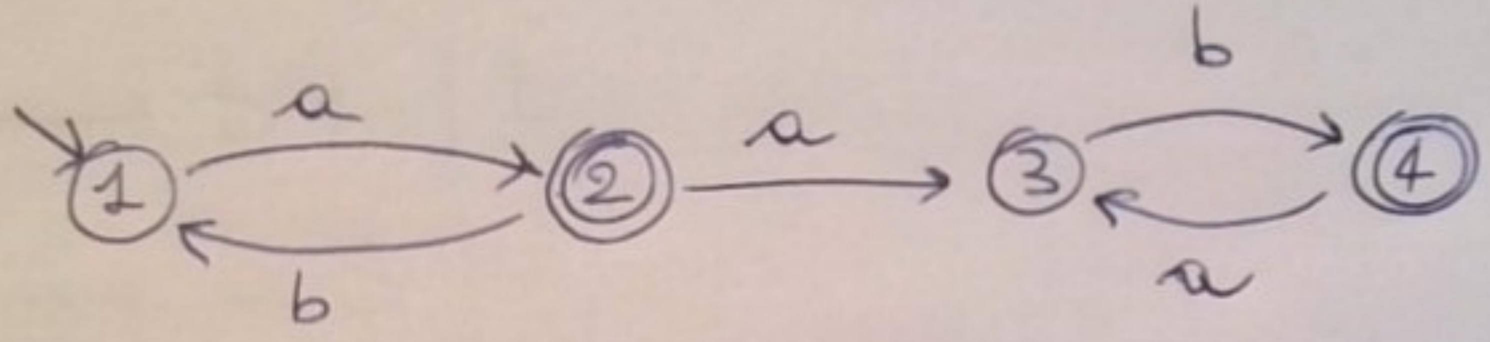
Con P simulo le transizioni:

se ho $q \xrightarrow{a} q'$ avrò la produzione $q \longrightarrow aq'$

Se q' è finale avrò anche la produzione $q \longrightarrow a$.

__Es__ Dato l'ASF A, trovare una grammatica $\mathcal{G}$ equivalente, cioè tc $\mathcal{L}(A) = \mathcal{L}(\mathcal{G})$.

A :



$$\mathcal{L}(A) = (ab)^* a + (ab)^* a \, a \, b \, (ab)^*$$

$$= (ab)^* a + (ab)^* a (ab)^+ = (ab)^* a \left(\varepsilon + (ab)^+ \right) = (ab)^* a (ab)^*$$

Che grammatica cerco? Basta una <u>regolare</u>.

Come terminali <u>devo</u> prendere a e b.

Tra i non terminali ci vuole S, e poi?

Le produzioni sono:

$$S \longrightarrow a2 \mid a$$
$$2 \longrightarrow a3 \mid bS$$
$$3 \longrightarrow b4 \mid b$$
$$4 \longrightarrow a3$$

Esempi: $S \rightarrow a.$

$$S \rightarrow a2 \rightarrow abS \rightarrow aba$$

$$S \rightarrow a\boxed{2} \rightarrow aa\boxed{3} \rightarrow aab\boxed{4} \rightarrow aaba\boxed{3} \rightarrow aabab.$$

...

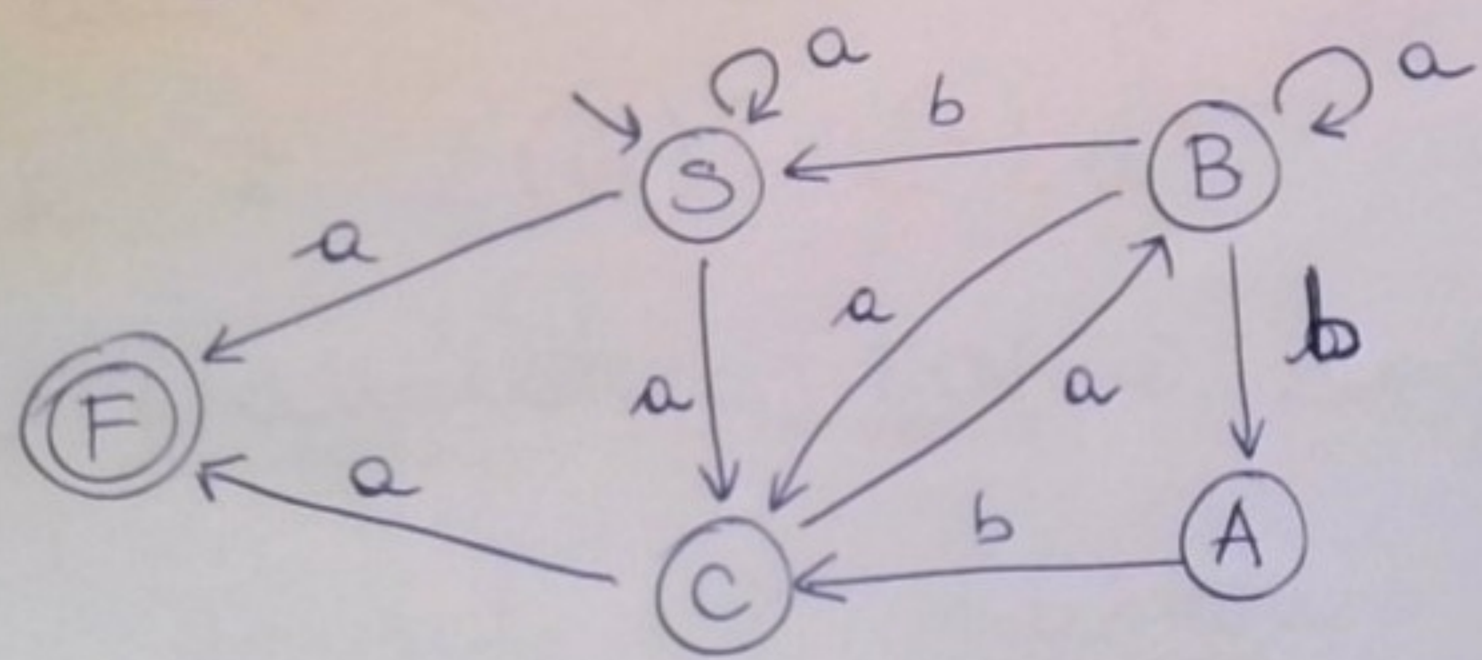
Es Data un gram. regolare, trovare un ASF equivalente

$$G = \{ V_N = \{S, A, B, C\}, V_T = \{a, b\}, P, S)$$

con $P:$

$$S \longrightarrow a \mid aS \mid aC$$
$$A \longrightarrow bC$$
$$B \longrightarrow aC \mid aB \mid bS \mid bA$$
$$C \longrightarrow a \mid aB$$

Idea: S: stati iniziali, A, B, C altri stati; uno stato extra finale, F.



non viene deterministico;
determinizzatelo
per compito

<u>Es</u> Sia G la grammatica con

$V_T = \{a, b\}$, $V_N = \{S, X\}$ e produzioni

$$S \longrightarrow Sa \mid aS \mid bX \mid b \qquad\qquad S \to aS \to aSa \to aba$$

$$SX \longrightarrow XXSS \mid S$$

$$XS \longrightarrow abba$$

- Che tipo di grammatica è? tipo/generale/non ristretta
- Qual è un automa a potenza minima che accetta lo stesso linguaggio generato da G? ASF
- Chi è $L(G)$? a^*ba^*

La grammatica NON è regolare, il linguaggio generato è regolare.