1° / 2°

↓

DFA / RE

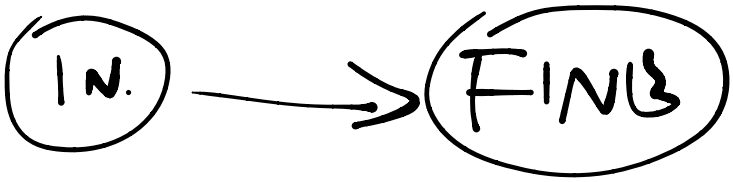
PUMPING LEMMA

TURING-RICONOSC.

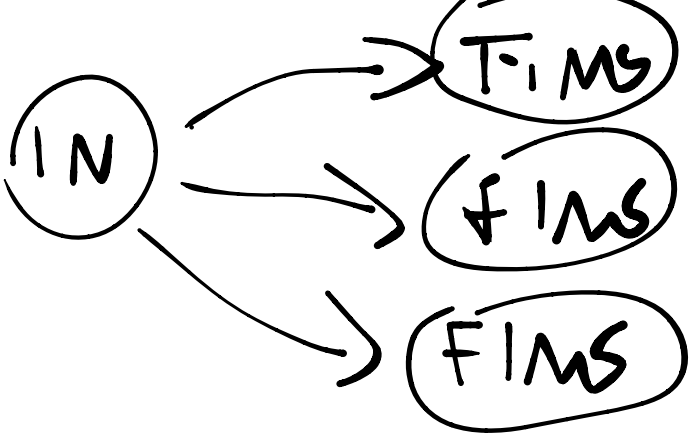
DECIDIBILITÀ

NP-HARD

DFA



NFA



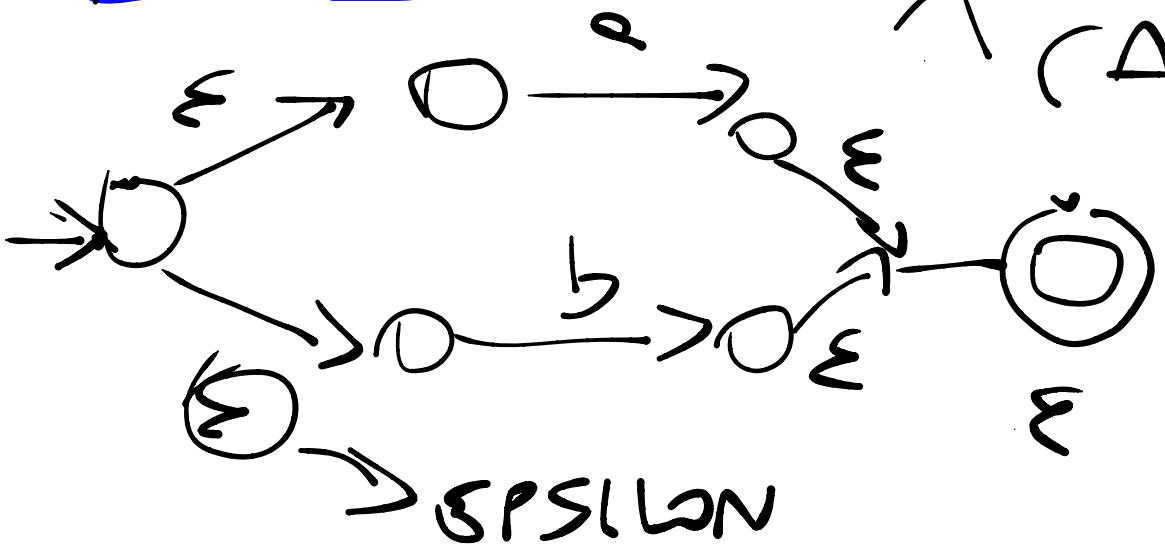
BR ->
RBG. EXPR.

$(A \cup B)^* A C^*)^+$

$A \cup B$

$(\bullet) \rightarrow$ CONC.

$(A \bullet B)$

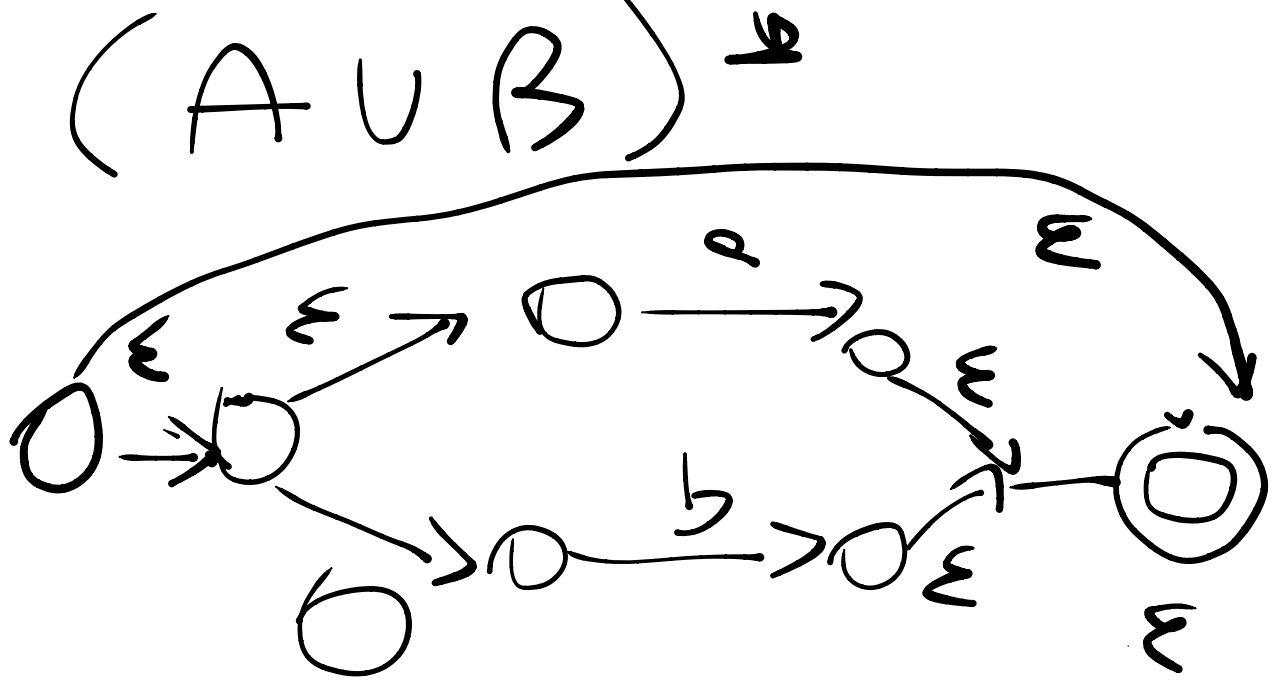


EPSILON

↓

NON CONSUMA
SIMBOLI
DI INPUT

$(A \cup B)^*$



$(A \cup B)^*$

STAR D.

(A, B) KLEENE

$A^* = AAA \ldots\ldots \infty$

DFA

↑

DETERMINISTICO

(SIAMO SICURI
CHE FINISCA)

NFA

NON-DET

(1 INIZIO, * FINI)

↓

GNFA

↑

GENERICO

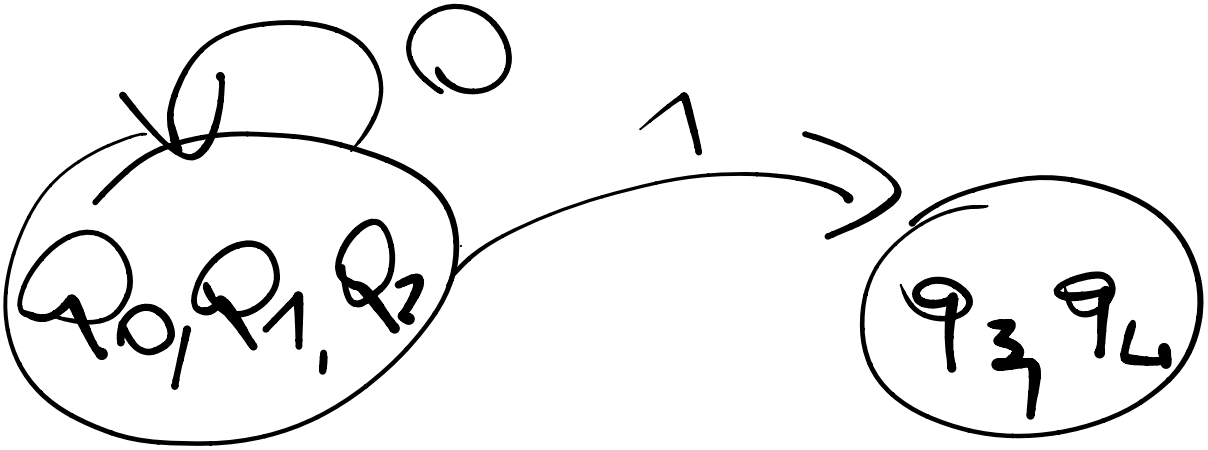
REGOLE

1. 1 STATO INIZIALE
   1 STATO FINALE
2. SCEGLI UNO STATO E LO RIMUOVI "COLLASSANDO" LE TRANSIZIONI

NFA → DFA
(GNFA)

ε - CHIUSURE
(ε - CLOSURE)

$\pi_0 = (q_0, q_1, q_2)$



↑

ε-CHIUSURA

STATI

SERVONO AD

APPLICARE

GNTA

A B

R_1 R_1 R_2

R_1, R_3

...

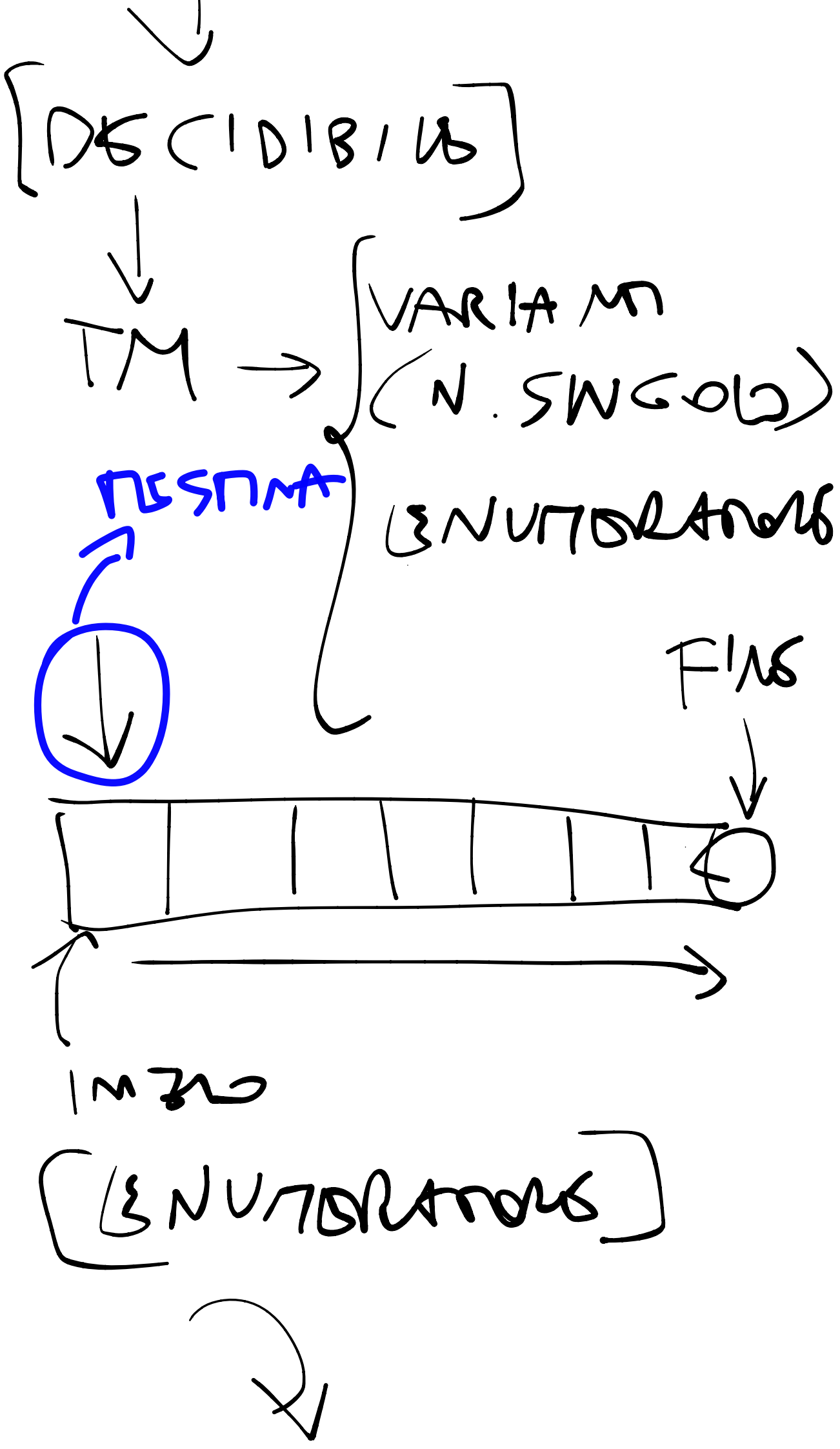
MAPPA
DOVE VANNO
TUTTI GLI STATI

↓

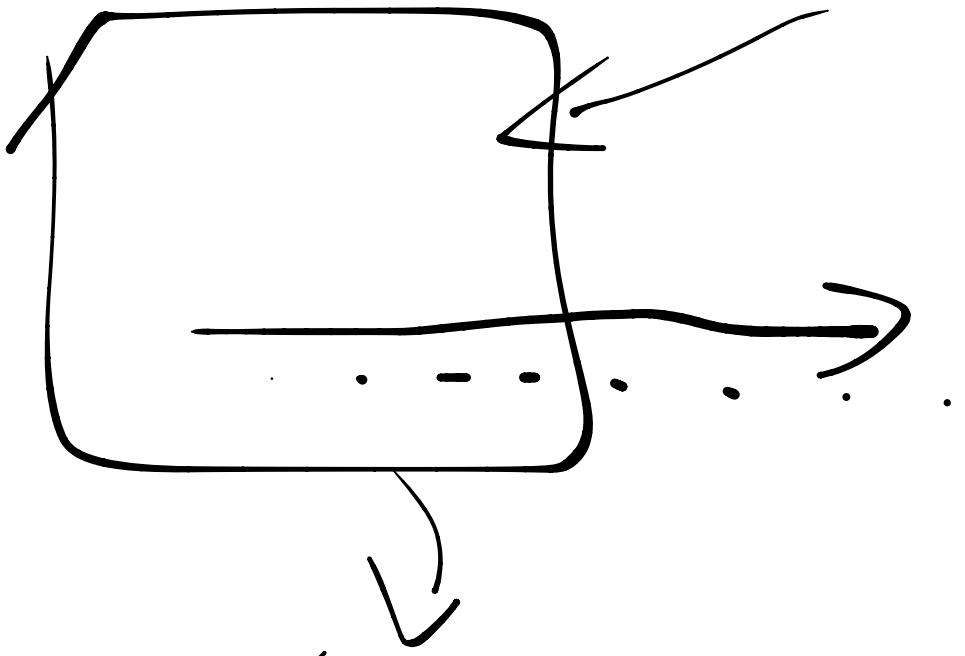
DISEGNO DI
L'AUTOMA
COMPLETO

DECIDIBILITÀ = (TM)

REGOLARITÀ = | DFA
NFA


[DECIDIBILI]
TM →
VARIANTI
(N. SINGOLI)
ENUMERATORE
RESTITUISCE
FINE
INIZIO
[ENUMERATORE]

$W = 1 \ldots\ldots n$



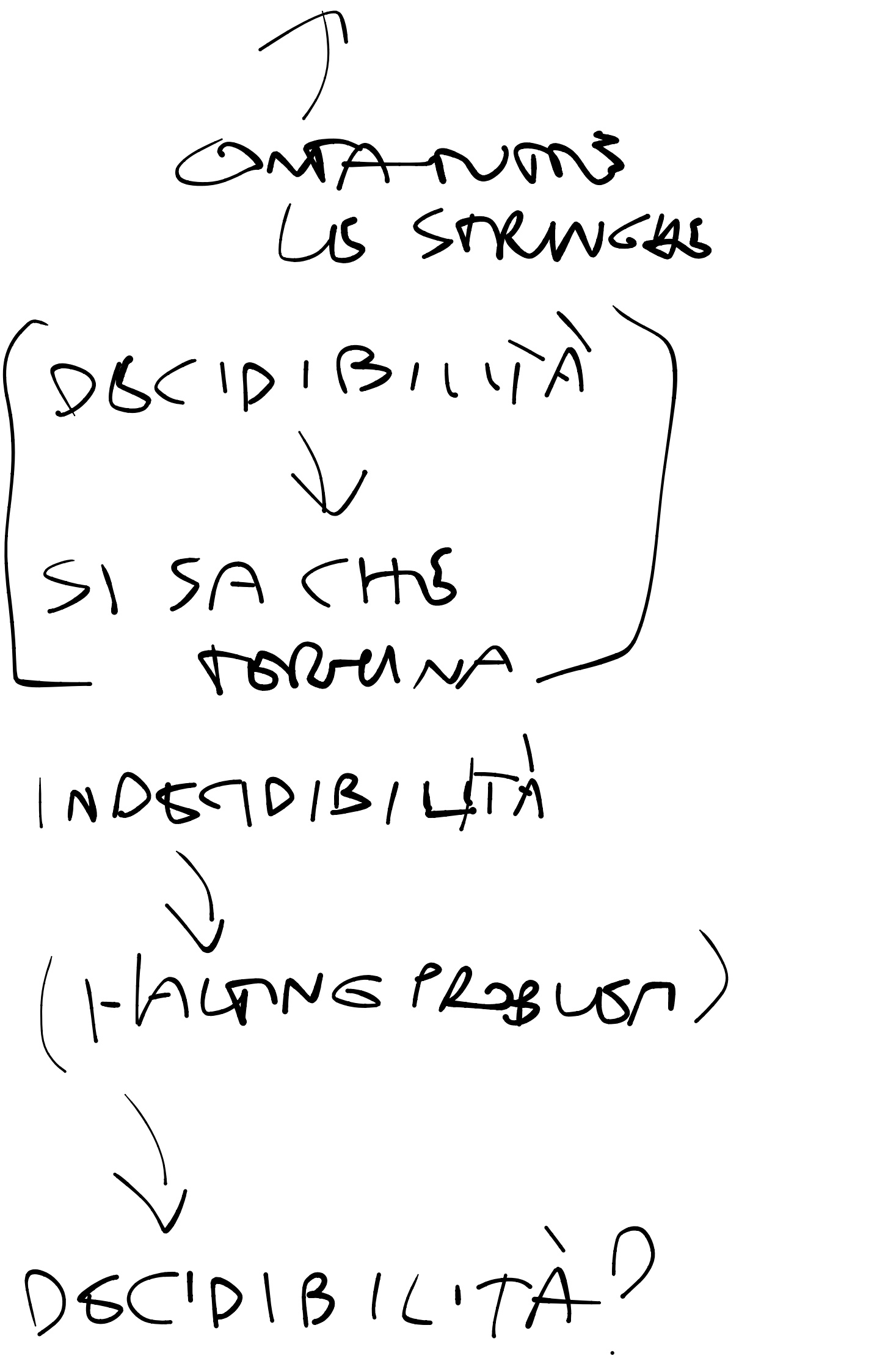
STAMPANTE

(SIPSER)

[TURING-RICONOSCIBILE]

RICONOSCIUTO DA UNA TM

(O ENUMERATORE)


CONTA TUTTE LE STRINGHE
DECIDIBILITÀ
SI SA CHE FORTUNA
INDECIDIBILITÀ
(HALTING PROBLEM)
DECIDIBILITÀ?

DESCRIZIONE TM

[TURING-RICONOSCIBILE]

=

RICORS. ENUMERABILE

$S_1 + S_2 \geq 8$

a b

$a = 3, b = 5$

$\infty + \infty = \infty$

R.E =

DIPENDE DALL'INPUT

E DALLA SUA

LUNGHEZZA

E NON SEMPRE

NOTEVOLE

$L = \{ x \mid \exists y < L$

x è sottostringa

$\{[ab]bbc\}$

L* è Turing-riconoscibile

↓

TM / E che fa L →

× SUBSTRING DI Y

E = enumeratore

- ∀x (per ogni stringa

↓

∀y (da qualche

sort selbe

(funcn'' y)

y

× × × ×

↑

[R.B]

$y \subseteq X$

↑

(sorrows usrs)

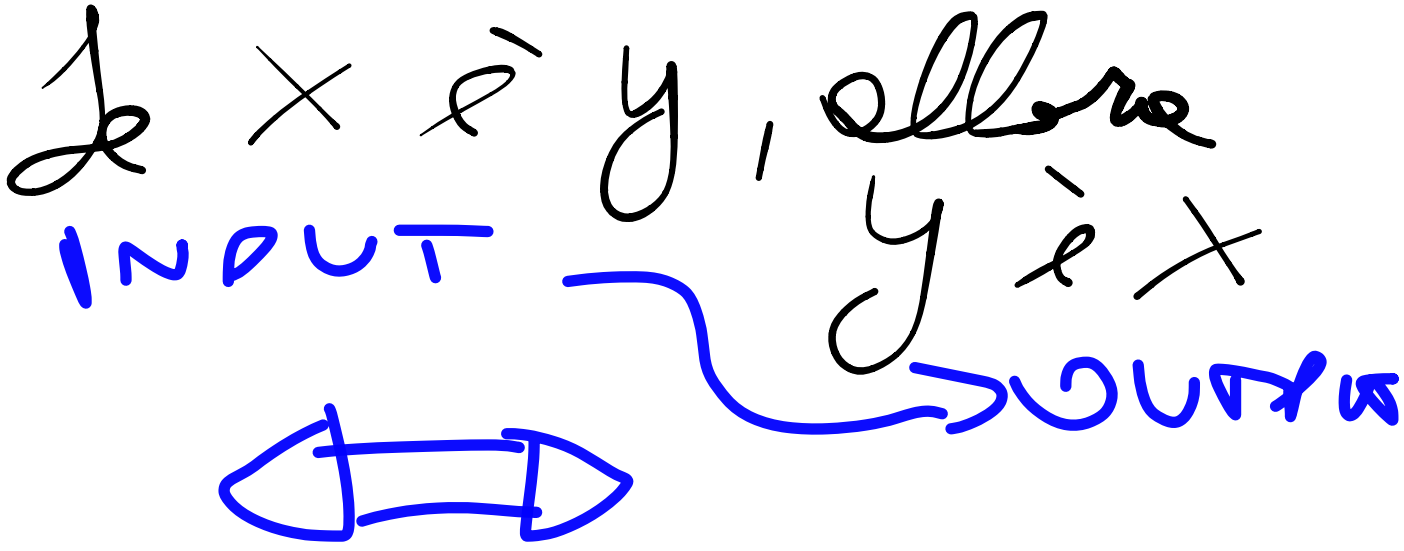
for i = 1 to X.lenght

for j = 1 to y.

)

B che enumera
(n. e) tutte le stringhe
$x \in L^*$,
da qualche parte
esiste una y
tale che B accetti

se x è y, allora
y è x



DSE

Se output, allora input!

$y \in L^*$

$\exists$ B bloche (1)

$\times$ la produtto y

---

oppure una descrizione con una TM

↓

Macchina TM

M su ogni input

$w \in L^*$

⇒ Copia ogni simbolo sul nastro

→ Verifica se $\exists y < L$

⇒ Se $\exists$, allora accetta

→ Se $\nexists$, allora rifiuta oppure in loop

[DESCRIBLOS AD AUTOLOGICS]

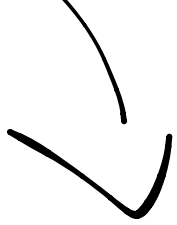
[NP / NL]

↓

Classi di problemi

$A \leq_m B$

↑ NP / NL ↓ PROBL.



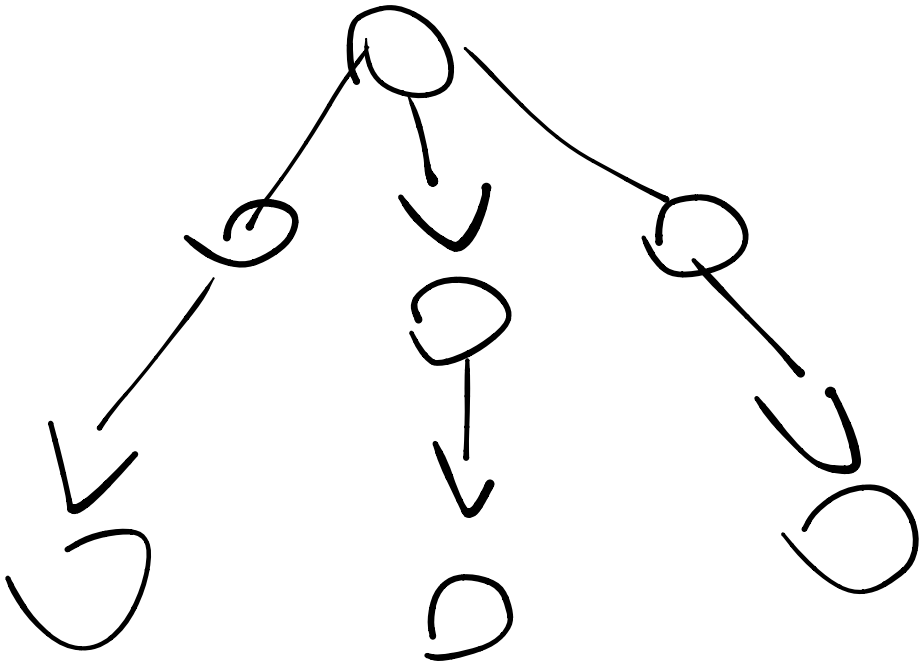
NL -> NON-LOG

DETERMINISTIC

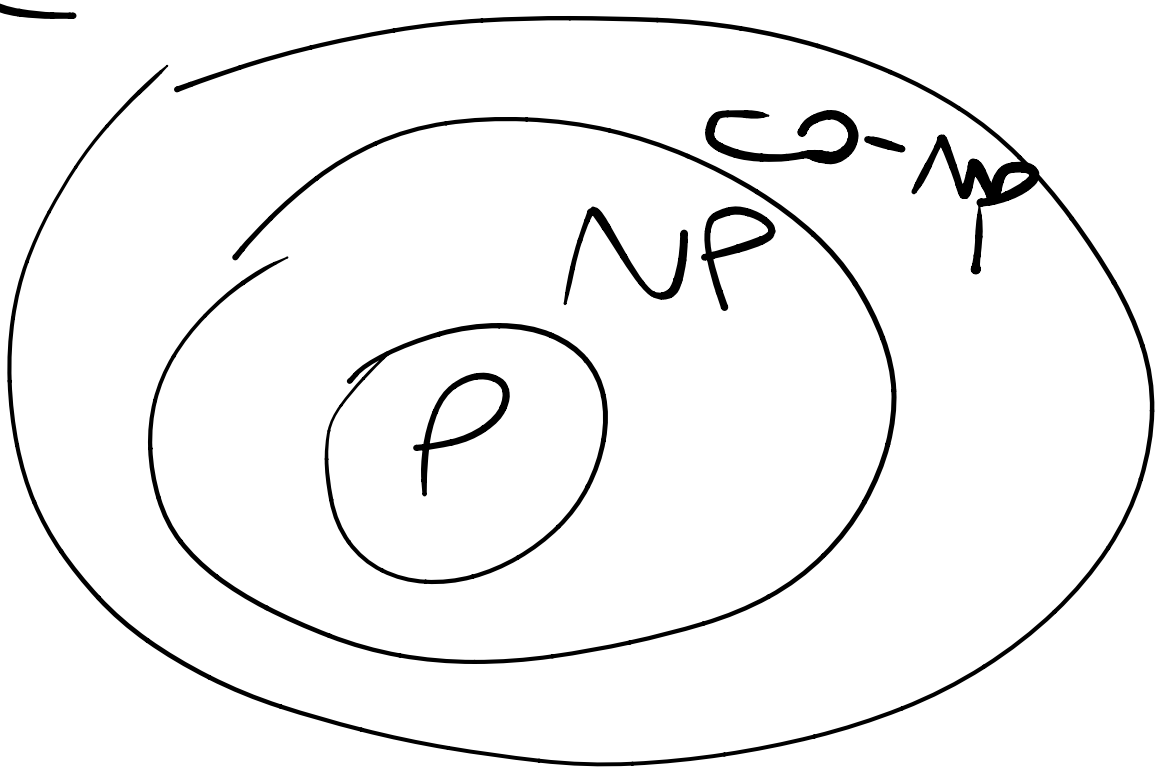
NP -> NON-POLY

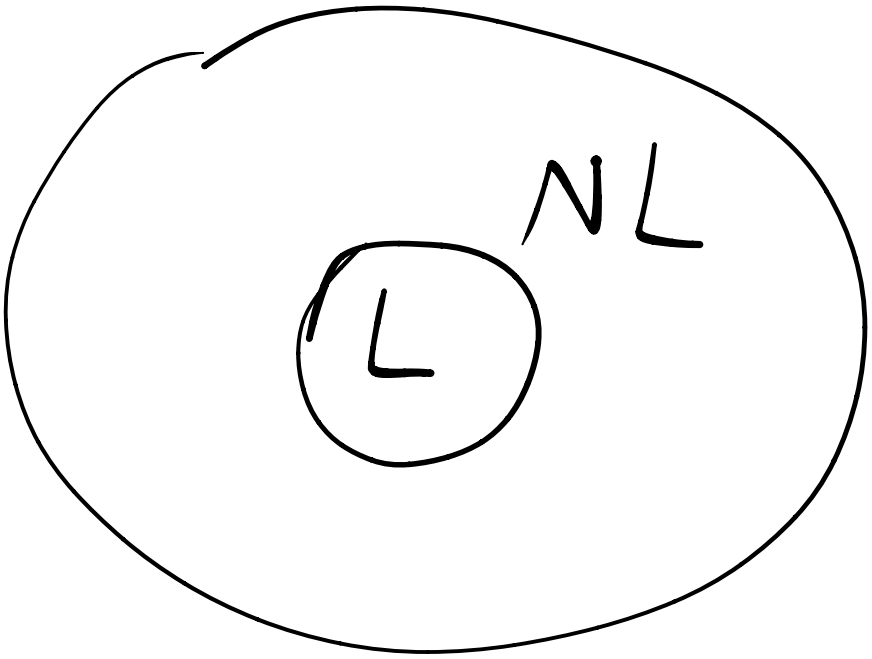
TM -> NON-DET

TM



CLASSI DI COMPLESSITÀ





NL - COMPLETO

↓

USIAMO UN
QUALCOSA DI

NL

↓

NDTM
(NON-DETERMINISTICA)

TM = MODELLI DI
CALCOLO

↓

DESCRIVERE

OGNI TIPO DI

PROBLEMA

↓

DESCRIZ. CON

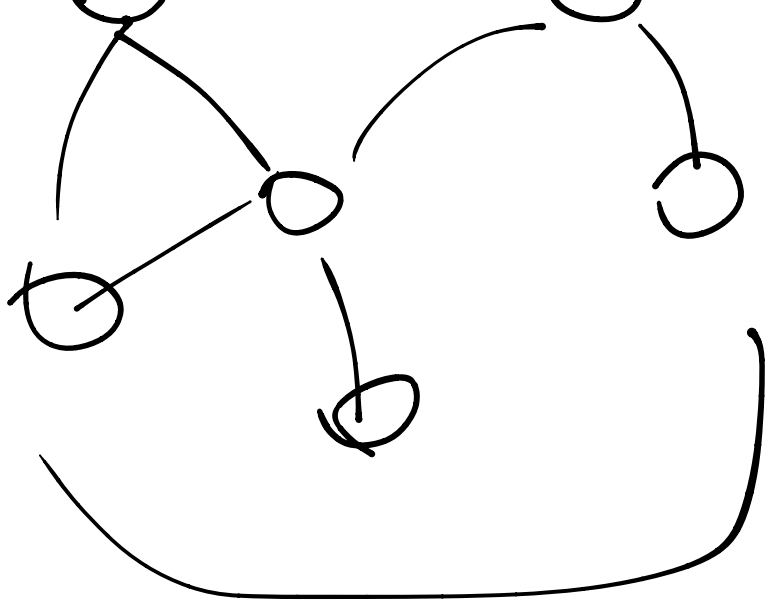
QUALCOSA IN

IL PROBLEMA

$G = (V, E)$

↑ ↑

VERTICI ARCHI



NDTM

PATH

↓

PERCORSO $\in$ P

↑

COME I RASS'
FIN LÀ
(N. DI VERTICI)

STRONGLY
CONNECTED
=
PERCORSO
COMUNE

↓

OGNI COPPIA
DI NODI HA
≥ 1 PERCORSO

NDTM → M

↓

DESCRIZIONE AD
ALTO LIVELLO

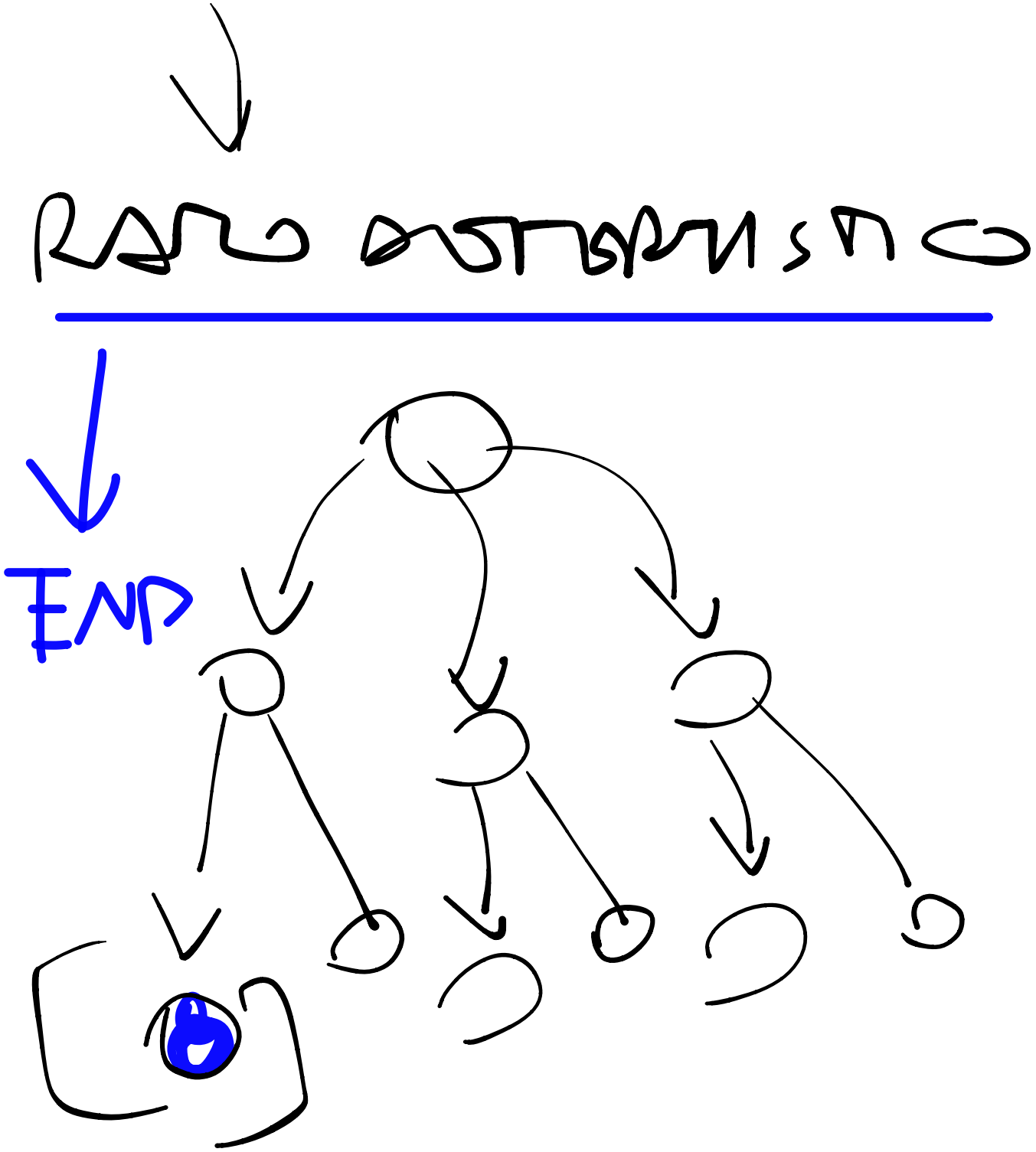
$G = (V, E)$

$\forall u, v \in G$

COPPIA DI
VERTICI

VERIFICHIAMO
"CASUALMENTE"
CHE LA CONDIZIONE

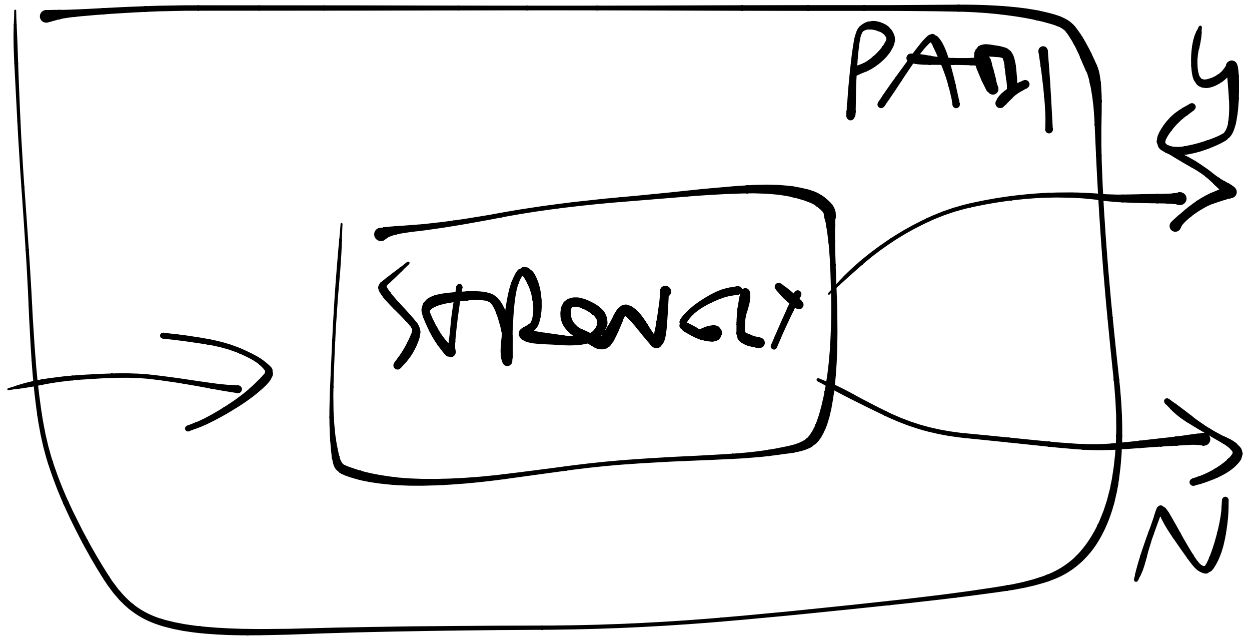
VARGA



log → PTM-NP


log(N|G|)
CARDINALITÀ
= N DI
ELEMENTI
O
lim
n→∞
più
grandi
PARTI =
NP-HARD
NL-HARD

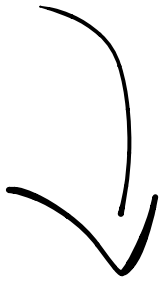
PATH $\leq_M$ STRONGLY



HARD =

Se troviamo una sol. efficiente vale per tutti i problemi dello stesso tipo

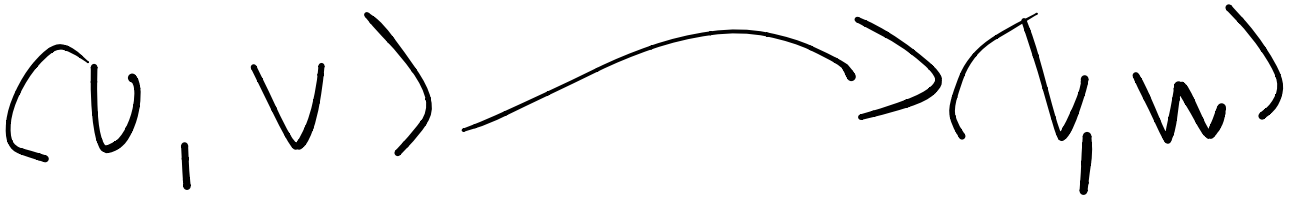
Se A riduce B,
usi B per spiegare
A



$G = (V, B)$

STRONGLY

TM → TAPE
(NASTRO)



$\exists$ riduzione

in

tempo

$A \leq_p B$