COMPLESSITÀ DI SPAZIO

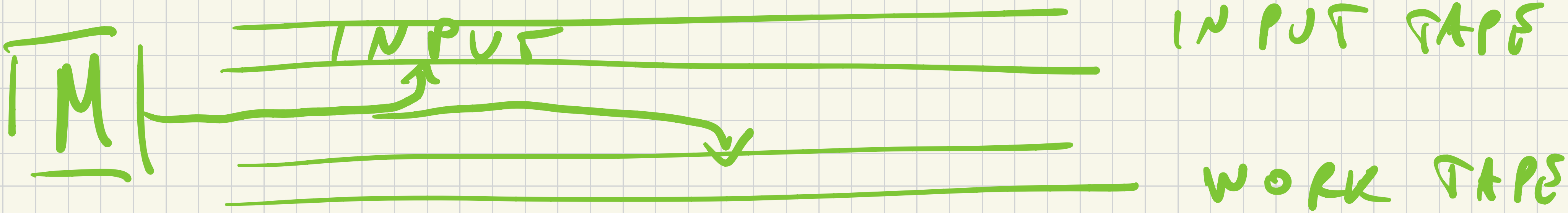
Vogliamo studiare efficienza delle TM in termini di spazio. Differenza fondamentale: lo spazio si può riusare.

DEF La complessità di spazio di un decisore M è una funzione $S: \mathbb{N} \to \mathbb{N}$ t.c.

$$S(n) = \max_{\substack{x \\ |x| = n}} \left\{ \# \text{ celle nastro scritte da } M(x) \right\}$$

Siccome l'input ha dimensione n, non

vogliamo essere penalizzati per dover leggere input. Cambiamo il modello di TM:



Il nastro di input è READ-ONLY.

Questo cambia le cose: Ad es. consideriamo le TM multinastro. Per la complessità di tempo la riduzione genera un overhead da $T(n)$ a $O(T^2(n))$. Per lo spazio non importa, solo da $S(n)$ a $O(S(n))$.

DEF $SPACE(f(n)) =$
$= \{ L :$ ∃TM M con complessità di spazio $O(f(n))$ t.c. $L(M) = L \}$

Stesse cose per $NSPACE(f(n))$, basta sostituire la TM M con NTM N.

Intuizione:

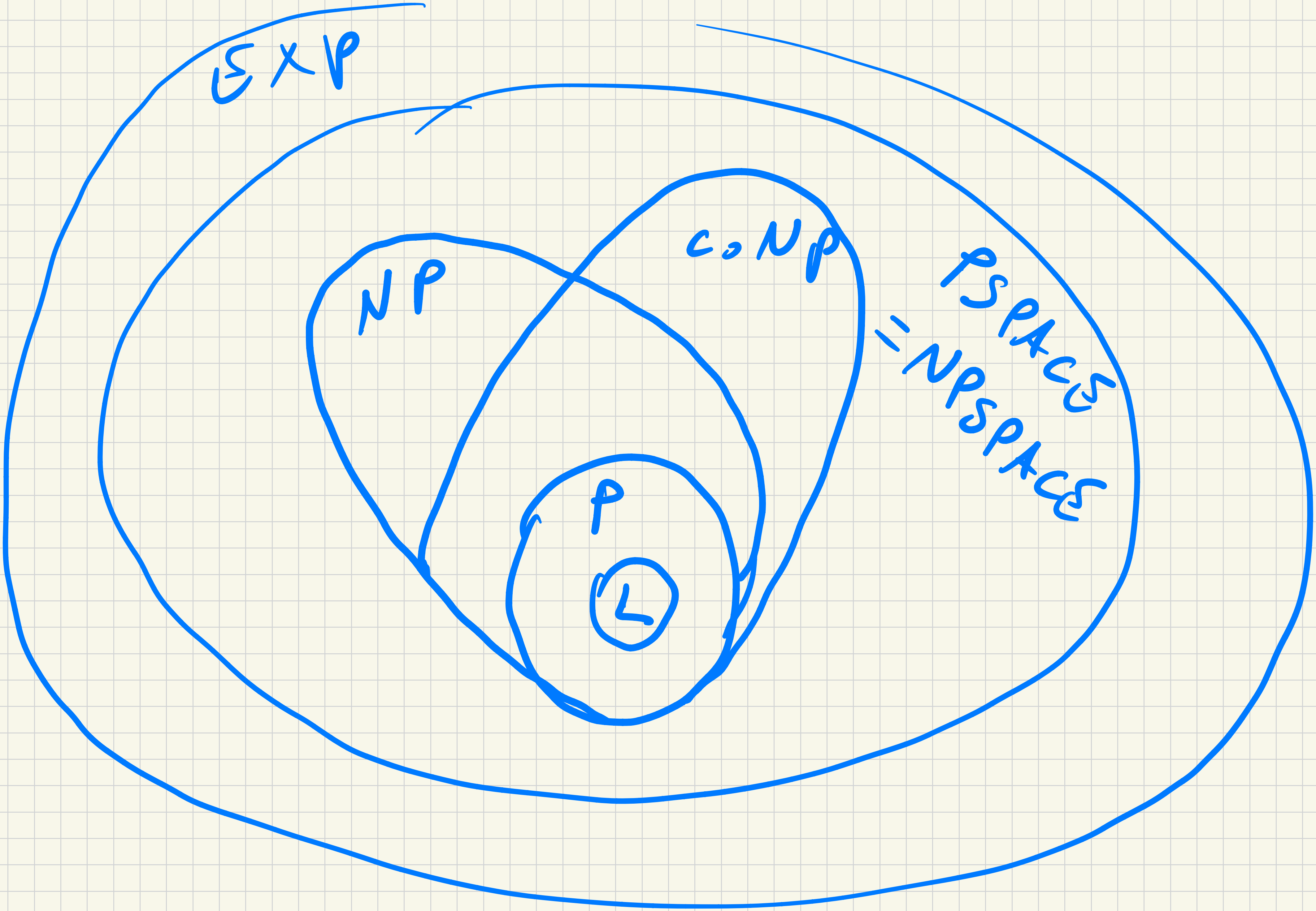
- Per tempo, la complessità è almeno lineare in n. Devo almeno leggere x.
- Per spazio, la complessità è almeno

log n. Devo almeno poter contare la lunghezza di x.

Le principali classi di complessità:

- $L = SPACE(\log n)$ e $NL = NSPACE(\log n)$
- $PSPACE = \bigcup_k SPACE(n^k)$ e
  $NPSPACE = \bigcup_k NSPACE(n^k)$
- EXPSPACE e NEXPSPACE.

La figura di riferimento (ipotetica):


EXP
NP
coNP
PSPACE
=NPSPACE
P
L

Iniziamo con degli esempi:

- $A = \{0^n 1^n : n \in \mathbb{N}\}$. Avevamo visto una TM che decide A in spazio $O(n)$. In realtà $A \in L$. Uso un contatore per contare gli 0 e a partire dal primo 1 lo decremento e accetto sse alla fine il contatore arriva a zero.

- PALINDROMES $\in L$. Pseudocodice:

  * Su input x, determino $|x| = n$.

  * For $i = 1, 2, \dots, n$

* RIFIUTO se $x_i \neq x_{n+1-i}$

* ACCETTO.

Alcuni dettagli implementativi:

- Nastro di lavoro 1 conto fino a n,
- Nastro di lavoro 2 ci metto i.
- Nastro di lavoro 3 ci metto $n+1-i$.

Per fare questo ad es. copio n dal nastro 1, lo incremento +1, poi copio i su nastro di lavoro 4 e decremento nastro 3 e nastro 4 fino a che nastro 4 contiene 0.

- Infine controllo $x_i \neq x_{n+1-i}$. Per fare questo copio i e $n+1-i$ su nastro 5 e 6. Decremento e sposto testina su nastro input.

Altro esempio: $c = a \cdot b$. Posso fare

$$c = \underbrace{a + a + a \ldots\ldots + a}_{b \text{ volte}}$$

Detto questo, non tutti i problemi sono in L.
Ci sono problemi che sappiamo essere in PSPACE ma non crediamo siano in L. Esempi:

3SAT, CIRCUIT-EVAL, PATH.

TEO $TIME(f(n)) \subseteq SPACE(f(n))$.

Dim. Una TM con tempo $O(f(n))$ può scrivere al più $O(f(n))$ celle. $\blacksquare$

COR. $P \subseteq PSPACE$; $NP \subseteq NPSPACE$.

TEO. $NP \subseteq PSPACE$.

Dim. Segue da un risultato che vedremo più avanti (Teorema di Savitch). Ma si può anche mostrare direttamente. Supponiamo che $A \in NP$ se esiste poly-time V t.c. $\forall x$: $x \in L$ sse $\exists y$ t.c.

$$V(x, y) = 1$$

Siccome V è poly time, $|y| = p(n)$ per un polinomio p. Per decidere A in spazio polinomiale:

- Memorizzo candidato y su nastro lavoro #1.
- Simulo $V(x, y)$ su nastro lavoro #2.
- Se $V(x, y)$ accetta, accetto. Altrimenti passo al candidato y successivo e simulo $V(x, y)$ riusando stesso spazio. $\blacksquare$

Ma in un certo senso è anche vero che lo spazio limita il tempo:

TEO Per ogni $f(n) \geq \log n$, $SPACE(f(n)) \subseteq DTIME(2^{O(f(n))})$.

COR $L \subseteq P$ e $PSPACE \subseteq EXP$.

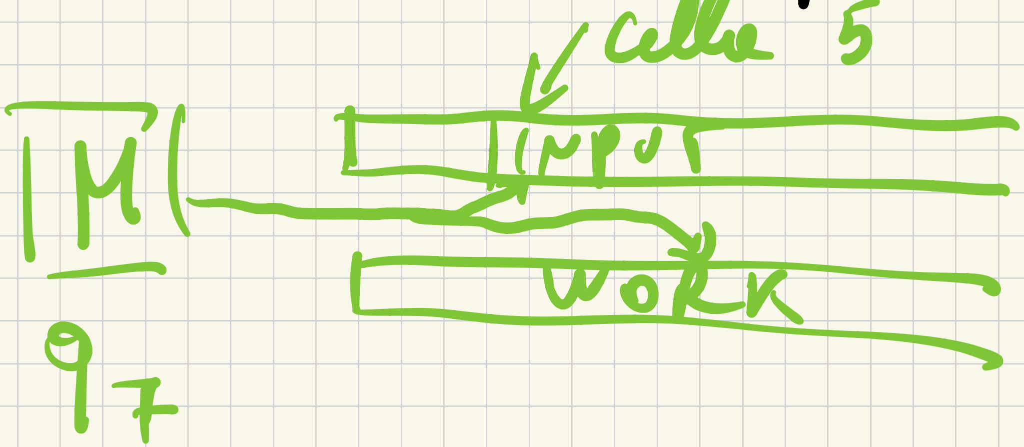
DIM. Intuizione è che $2^{O(f(n))}$ è bound sul # configurazioni di una TM con complessità di spazio $f(n)$.

Sia M decisore per A in spazio $O(f(n))$.

Ricordiamo le configurazioni:

- Contenuto nastro di lavoro.

- Stato
- Testine: input e lavoro.



Lo rappresento con una stringa:

$$WO\,|q_7|\,RK\ ;\ 5$$

Osservazione: Siccome M è un decisore, non può ripetere una configurazione.

$\Rightarrow$ Running time $\leq$ # configurazioni:

$$\#\text{ config.} \leq |\Gamma|^{f(n)} \cdot (|Q| \cdot f(n)) \cdot n$$

$|Q|$, $|\Gamma|$ sono costanti (indip. da n)

e $n \leq 2^{f(n)}$:

$\#$ config. $\leq 2^{O(f(n))}$. $\blacksquare$

Notare che:

$L \subseteq P \subseteq PSPACE \subseteq EXP$.

<u>T.H.T.</u> $P \neq EXP$

$\Rightarrow$ o $P \neq PSPACE$ o $PSPACE \neq EXP$

o tutte e due.

S.H.T. lo vedremo insieme al T.H.T.

Cominciamo a lavorare nella direzione di dimostrare il teorema di Savitch, ovvero $PSPACE = NPSPACE$.

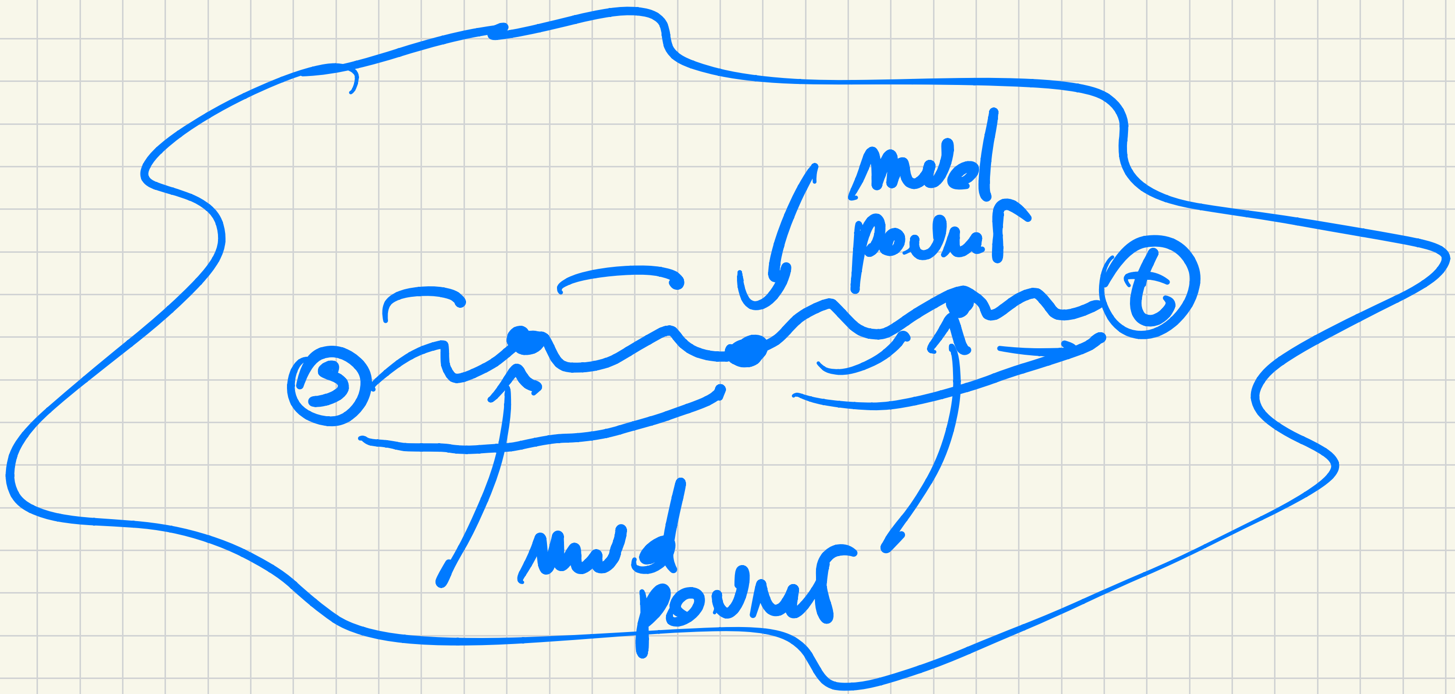
La tecnica è basata su PATH.

TEO $PATH \in SPACE(\log^2 n)$.

Dim. Dato $G = (V, E)$ e $s, t \in V$ devo stabilire esistenza $s \rightsquigarrow t$. Abbiamo studiato algoritmo di marcatura (PATHCP) che però non è efficiente in spazio.

Idea: Riusare lo spazio a discapito del tempo.

Supponiamo che G sia nella forma elenco nodi $1, 2, \ldots, n$ e matrice di adiacenza.



Considero la funzione:

$PATH?(x, y, k)$: Ritorna sì sse

$\exists$ $x \rightsquigarrow y$ con lunghezza $\leq 2^k$.

Fatto questo, lancerò: $PATH?(s, t, \lceil \log_2 n \rceil)$

perché cammino $s \rightsquigarrow t$ se esiste

ha lunghezza $\leq n$.

<u>$PATH?(x, y, k)$</u>:

- Se $k = 0$, ritorna sì sse $(x, y) \in E$ oppure $x = y$.

- Se $K > 0$:
  * (# Cerco MIDPOINT #.) Per ogni $w \in V$ se PATH?$(x, w, K-1)$ $\wedge$ PATH?$(w, y, K-1)$ allora ritorna SI. Altrimenti ritorna No.

La correttezza è immediata : $\exists$ $s \rightsquigarrow t$

$\exists$ midpoint w.

Complessità di spazio: Devo memorizzare n, s, t, x, y. Ma per ogni w posso riutilizzare lo spazio. Anche k devo memorizzare.

$\Rightarrow$ Per ogni passo della ricorsione

$O(\log n)$ spazio

$\Rightarrow$ Profondità ricorsione $O(\log n)$

$\Rightarrow$ $O(\log^2 n)$ complessità spazio