Che cosa posso dire su cosa fa effettivamente $D(\langle D \rangle)$? Siccome non trova una prova né per "$D(\langle D \rangle) \neq \infty$" né per "$D(\langle D \rangle) = \infty$" allora va in loop. Posso codificare l'affermazione "$D(\langle D \rangle) = \infty$" e abbiamo dimostrato:

"Se ZFC CONSISTENTE $\Longrightarrow D(\langle D \rangle) = \infty$".

"$A \rightarrow B$"

L'unico modo per evitare una contraddizione è che l'affermazione "ZFC CONSISTENTE" non sia dimostrabile in ZFC.

THM Se ZFC è CONSISTENTE, allora non solo ZFC non è COMPLETO ma la seguente affermazione non è DIMOSTRABILE:

"ZFC è CONSISTENTE".

COMPLESSITÀ

Per la prima volta iniziamo ad interrogarci sulle risorse. Dato un problema, vorremmo risolverlo usando meno risorse possibili. Es.

- Tempo ✓
- Spazio ✓
- Randomicità
- # processori (per algoritmi parallelizzabili).

Studiare quante risorse occorrono per risolvere certi problemi. Come vedremo ci sono molte questioni aperte:

- $P = NP$? Trovare soluzione è fattibile tanto quanto riconoscerne una?
- $P = NC$? Tutti gli algoritmi sono parallelizzabili.
- $P = PSPACE$? Se posso risolvere un problema usando poco spazio, posso risolverlo in poco tempo?
- $P = BPP$? Gli algoritmi efficienti randomizzati possono essere sempre resi deterministici?

Teoria della complessità di tempo.

DEF Sia $M \in TM$ un decisore. Il tempo di esecuzione di M è una funzione $t : \mathbb{N} \to \mathbb{N}$

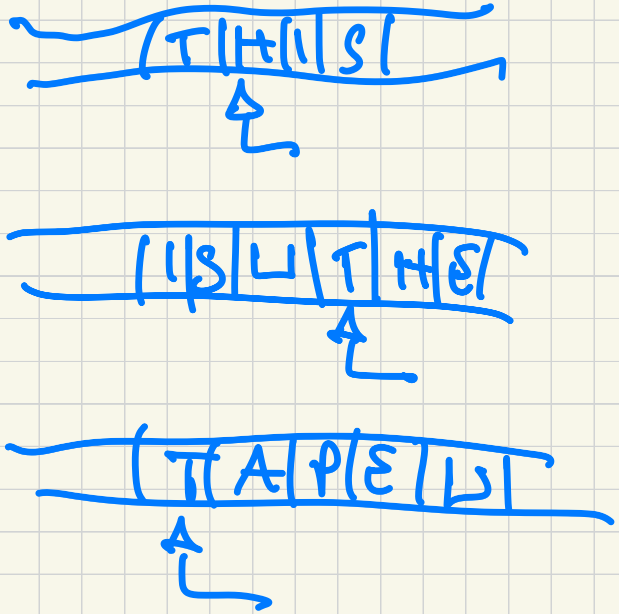
$$t(n) = \max_{\substack{x \in \Sigma^* \\ |x| = n}} \# \text{ passi di } M(x).$$

Abbiamo visto tanti "trucchi di programmazione" per le TM, adesso possiamo rivalutarli dal punto di vista della risorsa tempo.

Es.: La TM che può fare $\{L, R, S\}$. Avevamo stabilito che possiamo simulare queste TM aggiungendo stati "dummy" dove la TM standard muove a DX e poi a SX. Se la TM ha complessità di tempo $t(n)$, la nuova TM avrà $2 \cdot t(n) = O(t(n))$.

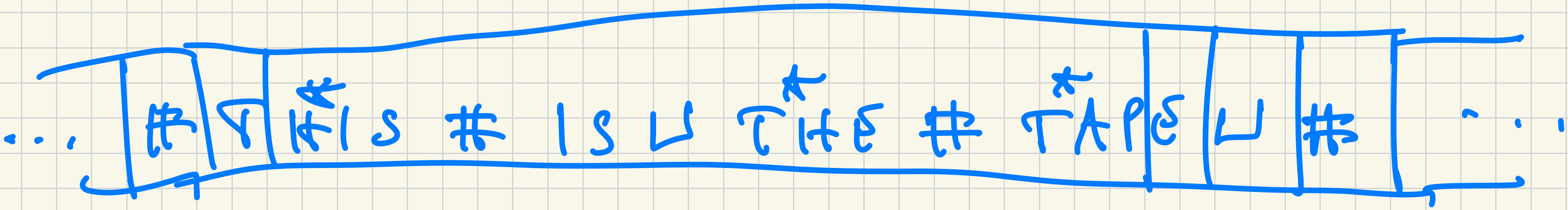
Altri esempi simili: marcare un simbolo, spostare il contenuto del nastro inserendo "$\sqcup$".

Ultimo esempio: le TM multinastro. Avevamo visto che può essere simulata usando nastro singolo.



TM M

3 nastri



A grandi linee:

*) Inizializzazione: Tempo $O(n)$

INPUT $\sqcup$ su M

$\# \overset{*}{I}NPUT \# \sqcup^* \# \sqcup^* \#$ su M'

*) Per ogni passo di computazione di M, la TM' deve leggere i simboli marcati, aggiornare i nastri ed eventualmente se necessario spostare di una posizione il contenuto del nastro.

Osservazione: Se M ha tempo $t(n)$ la lunghezza dei nastri è $\leq t(n)$. Il nastro di M' ha lunghezza $3t(n)$.

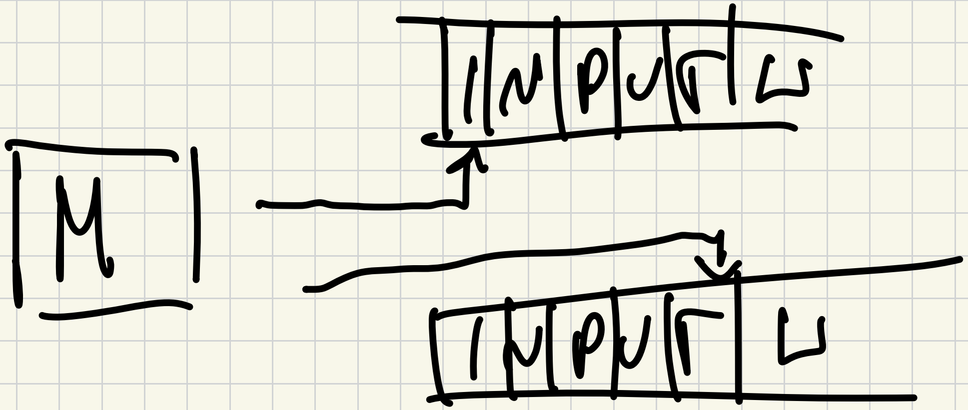
$\Rightarrow$ 1 passo di M corrisponde ad $O(t(n))$

Complessità di tempo: $O(t^2(n)) + O(n)$

$$= O(t^2(n))$$

Stessa cosa se i nastri fossero $k = O(1)$.
Questo può fare la differenza: ad es. il linguaggio PALINDROMES (stringhe palindrome su Σ^*) è decidibile in tempo $O(n)$

Su una TM a 2 nastri:



Tempo: $O(n)$

Su nastro singolo avevamo già visto una TM con tempo $O(n^2)$.

THM ('65) Ogni TM a 1 nastro per PALINDROMES ha comp. tempo $\Omega(n^2)$.

DEF Sia $t : \mathbb{N} \to \mathbb{N}$

$$DTIME(t(n)) = \{ L : \exists \text{ TM } M \text{ t.c. } L(M) = L \text{ e } M \text{ ha tempo } O(t(n)) \}$$

$PALINDROMES \in DTIME(n^2)$

$\notin DTIME(n)$

DEF. $P = \bigcup_{K \in \mathbb{N}} DTIME(n^k)$

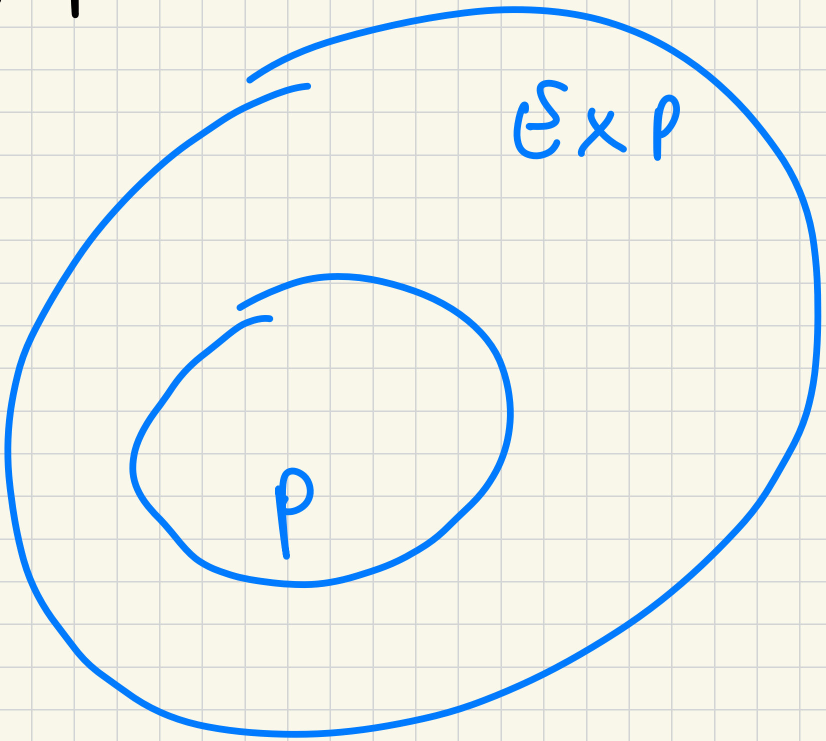
Ovvero tutti i linguaggi decidibili da TM M con comp. di tempo polinomiale $(O(n^k))$

Osserviamo che la def. di P è ROBUSTA rispetto alle varianti di TM che abbiamo studiato (e.g. k-nastri).

DEF $EXP = \bigcup_{K \in \mathbb{N}} DTIME(2^{n^k})$

Ovvero l'insieme di L decidibili con comp. di tempo esponenziale.

Fatto fondamentale: Ovviamente $P \subseteq EXP$ e $P \neq EXP$ ($\exists L$ t.c. $L \in EXP$ ma $L \notin P$)



$$\boxed{P \subsetneq EXP}$$

TEOREMA DI GERARCHIA DI TEMPO

(lo dimostreremo dopo).

Vedremo esempi di linguaggi che sono chiaramente in EXP, ma che in realtà sono in P usando algoritmi intelligenti. Ma anche esempi di linguaggi che sono in EXP ma non sembrano appartenere a P.

Linguaggi su grafi: Input $x = \langle G \rangle$ (cioè numero di nodi V e matrice di adiacenza).

Tipicamente si misura la comp. di tempo in funzione di $n = \# V$ ed $m = \# E$. Siccome $m \leq n^2$, allora un algoritmo con comp. di tempo $\text{poly}(n, m)$ è polinomiale time.

PATH = $\{ \langle G, s, t \rangle$: $s, t \in V$ e $s \rightsquigarrow t \}$.

Ovviamente, siccome un cammino $s \rightsquigarrow t$ è una sequenza $v_0 = s, v_1, v_2, \dots, v_\ell = t$ e inoltre $\ell \leq n$, il # di cammini è $O(n^n) =$
$= O(2^{n \log n})$. Ovvero PATH $\in$ EXP:
Posso provare tutti i cammini da s a t di lunghezza n e controllare un cammino costa $O(n)$: Tempo $O(n \cdot 2^n)$
Avevamo però anche visto PATH $\in$ P:
*) Marca s
*) Ripeti :

- Per ogni arco (u, v)
- Se u è marcato e v no, marca v.
- Termina quando non si marcano nuovi nodi.

*) Accetta sse t è marcato.

Tempo: $O(n \cdot m) = O(n^3)$

PATH $\in$ P.

Altro esempio: $2COL = \{ \langle G \rangle : G \text{ 2-colorabile} \}$

Posso assegnare uno tra 2 colori ai vertici in modo che nodi adiacenti abbiano colore diverso.

2COL ∈ EXP: Ci sono al più $O(2^n)$ 2-colorazioni e controllarne una costa $O(m)$. Tempo: $O(m \cdot 2^n)$.

In realtà, 2COL ∈ P:

*) Parti da un vertice e colora RED.
*) Colora tutti i suoi vicini BLU.
*) Colora i vicini dei vicini RED

....

Ripeti per ogni componente connessa di G. Se c'è una contraddizione, RIFIUTA. Se alla fine tutto è colorato, ACCETTA.

Tempo: poly $(n, m) =$ poly (n).

Ma non sempre così: Ad es. 3COL.
Ovviamente $3COL \in EXP$. Ma non sappiamo se $3COL \in P$.

THM $3COL \in DTIME(1{,}33^n)$.

Altro es.: $CLIQUE = \{ \langle G \rangle : G$ contiene triangolo$\}$.
$CLIQUE \in P$: # triangoli $O(n^3)$ e controllare se è un triangolo richiede $O(1)$
Non sappiamo se $CLIQUE \in DTIME(n^2)$. Sappiamo che $CLIQUE \in DTIME(n^{2,39})$.
Su questa linea:

- 4-CLIQUE $\in DTIME(n^4)$ ma non sappiamo se 4-CLIQUE $\in DTIME(n^3)$.
- 5-CLIQUE $\in DTIME(n^{4.7})$
- K-CLIQUE $\in DTIME(n^K)$; per $K = O(1)$ è in P, ma se K dipende da n in generale è in EXP e non sappiamo se è in P.