

Automi e Linguaggi Formali

Parte 17 – Complessità di tempo

Davide Bresolin
Ultimo aggiornamento: 22 maggio 2024



UNIVERSITÀ
DEGLI STUDI
DI PADOVA

- 1 Complessità di tempo
- 2 Relazioni di complessità tra modelli

Consideriamo il linguaggio $A = \{0^k 1^k \mid k \geq 0\}$

- Che tipo di linguaggio è?
- È decidibile?
- Quanto tempo serve ad una TM a nastro singolo per decidere questo linguaggio?

- Sia M una TM **deterministica** che **si ferma** su tutti gli input.
- Il **tempo di esecuzione** (o **complessità di tempo**) di M è la funzione $f : \mathbb{N} \mapsto \mathbb{N}$ tale che $f(n)$ è il numero massimo di passi che M utilizza su un input di lunghezza n .
- Se $f(n)$ è il tempo di esecuzione di M , diciamo che M è una TM **di tempo** $f(n)$.
- Useremo n per rappresentare la **lunghezza dell'input**.
- Ci interesseremo dell'**analisi del caso peggior**.

- **Analisi asintotica**: valuta il tempo di esecuzione su input grandi
- Considera solo il **termine di ordine maggiore** e **ignora i coefficienti**

Definition

Date due funzioni f, g , diciamo che $f(n) = O(g(n))$ se esistono interi positivi c, n_0 tali che per ogni $n \geq n_0$:

$$f(n) \leq c g(n).$$

$g(n)$ è un **limite superiore asintotico** per $f(n)$

Analizziamo questa TM per $A = \{0^k 1^k \mid k \geq 0\}$:

M_1 = “Su input w :

- 1 Scorre il nastro e **rifiuta** se trova uno 0 a destra di un 1
- 2 Ripete finché il nastro contiene almeno uno 0 ed un 1:
- 3 Scorre il nastro cancellando uno 0 ed un 1
- 4 Se rimane almeno uno 0 dopo che ogni 1 è stato cancellato, o se rimane almeno un 1 dopo che ogni 0 è stato cancellato, **rifiuta**. Altrimenti, se non rimangono né 0 né 1 sul nastro, **accetta**”

Definition

Sia $t : \mathbb{N} \mapsto \mathbb{N}$ una funzione.

La **classe di complessità di tempo** $\text{TIME}(t(n))$ è l'insieme di tutti i linguaggi che sono **decisi** da una **TM in tempo** $O(t(n))$.

- Questo è diverso dalle classi di linguaggi discusse in precedenza, che si concentravano sulla **computabilità**.
- Questa classificazione si concentra sul **tempo necessario** per decidere il linguaggio.

- Dall'analisi che abbiamo fatto, sappiamo che

$$A = \{0^k 1^k \mid k \geq 0\}$$

appartiene alla classe di complessità di tempo $\text{TIME}(n^2)$,
perché M_1 decide A in tempo $O(n^2)$

MA

- Esiste una macchina che decide A in modo **asintoticamente più veloce?**

Idea: cancelliamo metà degli 0 e metà degli 1 ad ogni scansione

M_2 = “Su input w :

- 1 Scorre il nastro e **rifiuta** se trova uno 0 a destra di un 1
- 2 Ripete finché il nastro contiene almeno uno 0 ed un 1:
 - 3 Scorre il nastro e controlla se il numero totale di 0 e 1 rimasti è pari o dispari. Se è dispari, **rifiuta**.
 - 4 Scorre il nastro, cancellando prima ogni secondo 0 a partire dal primo 0, poi cancellando ogni secondo 1 a partire dal primo 1.
- 5 Se nessuno 0 e nessun 1 rimangono sul nastro, **accetta**.
Altrimenti **rifiuta**.”

Possiamo fare ancora meglio?



- Possiamo trovare una TM che decide A in $O(n)$?
- **Problema:** non esiste una TM a nastro singolo che è in grado di decidere A in tempo $O(n)$.
- Possiamo farlo se la TM ha un secondo nastro
- **Importante:** la complessità di tempo dipende dal modello di calcolo.
- La **tesi di Church-Turing** implica che tutti i modelli di calcolo “ragionevoli” siano equivalenti.
- **In pratica:** discuteremo quanto questa differenza sia importante (o meno) per il nostro sistema di classificazione nelle prossime lezioni.

Idea: usiamo il secondo nastro per contare 0 e 1

M_2 = "Su input w :

- 1** Scorre il nastro 1 e **rifiuta** se trova uno 0 a destra di un 1
- 2** Scorre i simboli 0 sul nastro 1 fino al primo 1.
Contemporaneamente, copia ogni 0 sul nastro 2.
- 3** Scorre i simboli 1 sul nastro 1 fino alla fine dell'input. Per ogni 1 letto sul nastro 1, cancella uno 0 sul nastro 2. Se ogni 0 è stato cancellato prima di aver letto tutti gli 1, **rifiuta**.
- 4** Se tutti gli 0 sono stati cancellati, **accetta**. Se rimane qualche 0, **rifiuta**."

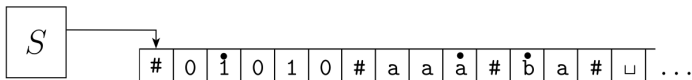
1 Complessità di tempo

2 Relazioni di complessità tra modelli

Theorem

Sia $t(n)$ una funzione tale che $t(n) \leq n$. Ogni TM *multinastro* di tempo $t(n)$ ammette una TM *equivalente* a *nastro singolo* di tempo $O(t^2(n))$.

- Ricordiamo la dimostrazione di come convertire una TM da multinastro a nastro singolo.



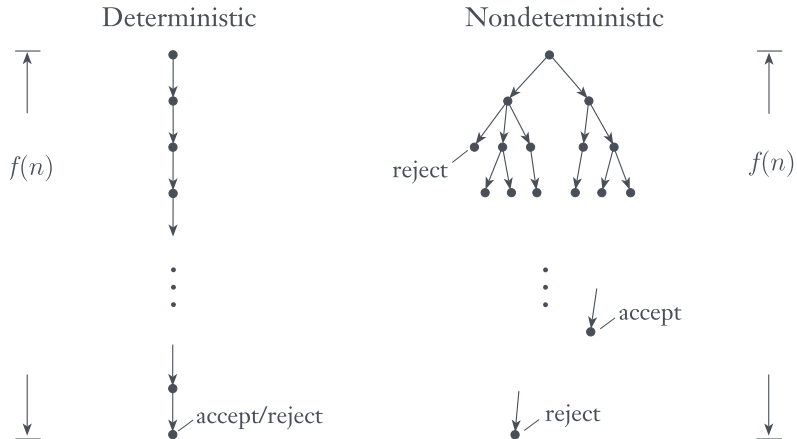
- Dobbiamo determinare quanto tempo ci vuole per simulare ogni passo della macchina multinastro sulla TM a nastro singolo.

Definition

Sia N una TM non deterministica che è anche un **decisore**. Il **tempo di esecuzione** di N è la funzione $f : \mathbb{N} \mapsto \mathbb{N}$ tale che $f(n)$ è il massimo numero di passi che N usa per ognuno dei rami di computazione, su input di lunghezza n .

- **N.B.:** la definizione di tempo di esecuzione per le TM non deterministiche non è destinato a corrispondere ad un qualche dispositivo di calcolo reale. È uno strumento teorico che utilizziamo per comprendere i problemi computazionali.

TM non deterministic



Theorem

Sia $t(n)$ una funzione tale che $t(n) \leq n$. Ogni TM *non deterministica* di tempo $t(n)$ ammette una TM equivalente a *nastro singolo* di tempo $2^{O(t(n))}$.

Dimostrazione

- Sia N una TM non deterministica in tempo $t(n)$.
- Costruiamo una TM deterministica D che simula N .
- D è una TM *multinastro*. Qual è la complessità quando viene convertita in una TM a nastro singolo?