Elementi di calcolabilità e complessità

Matteo Giorgi

a.a. 2020

Note (._.)

Queste note sono state prese dal sottoscritto durante le lezioni del corso di *Elementi di calcolabilità e complessità* tenuto dal prof. Pierpaolo Degano con l'ausilio del dott. Giulio Masetti durante l'anno accademico 2019-2020.

Il materiale usato per la stesura, oltre alle note distribuite dal professore, comprende, in maniera più o meno estensiva, i seguenti testi:

- Introduction to the theory of computation - Sipser
- Computability, introduction to recursive function theory - Cutland
- Introduzione alla complessità computazionale - Bernasconi, Codenotti
- Linguaggi, modelli, complessità - Ausiello, d'Amore, Gambosi

---

Pierpaolo Degano

- ufficio 285, dipartimento di Informatica
- `di.unipi.it/~degano`
- `degano@unipi.di.it`
- 050 221 2757

Giulio Masetti

- ufficio I-25, edificio B, CNR
- `giulio.masetti@isti.cnr.it`

Matteo Giorgi (me)

- `matteogiorgi.github.io`
- `matteo.giorgi@protonmail.com`

Lezione 01 2020.02.17

Idea intuitiva di algoritmo

Il punto di partenza della teoria della calcolabilità è l'esigenza di formalizzare l'idea intuitiva di *funzione calcolabile* da un algoritmo, ovvero di *funzione algoritmica*[1].

Si può dire che un algoritmo sia un procedimento di calcolo che consenta di pervenire alla soluzione di un problema, numerico o simbolico, mediante una sequenza finita di operazioni, completamente e univocamente determinate: una serie di istruzioni la cui esecuzione consenta di trasformare l'insieme finito di dati simbolici che descrivono il problema, nella soluzione del problema stesso.

Caratteristiche distintive

1. l'insieme delle istruzioni che definisce l'algoritmo è finito
2. l'insieme delle informazioni che rappresentano il problema
    - ha cardinalità finita
    - ha un effetto limitato su dati discreti
    - è descritta da dati finiti
3. il procedimento di calcolo (o *computazione*) è suddiviso in passi discreti e non fa uso di dispositivi analogici
4. ogni passo di computazione dipende solo dai precedenti e da una porzione finita dei dati in modo deterministico (è determinato senza ambiguità - non è soggetto ad alcuna distribuzione probabilistica non banale)
5. non c'è limite al numero di passi necessari all'esecuzione di un algoritmo, nè alla memoria richiesta per contenere i dati iniziali, intermedi e finali[2]

In conclusione, una procedura algoritmica riceve in ingresso una descrizione finita dei dati del problema e restituisce, dopo un tempo finito, una descrizione finita del

[1] la nozione di funzione algoritmica sarà poi formalizzata dal concetto di *funzione ricorsiva*

[2] è utile aggiungere una ulteriore caratteristica a quelle elencate dal prof: *esiste un agente di calcolo in grado di eseguire le istruzioni (un calcolatore)*

risultato. La sua natura deterministica fa sì che l'algoritmo fornisca sempre lo stesso risultato ogni volta che riceve in ingresso gli stessi dati: così facendo l'algoritmo stabilisce una relazione funzionale tra l'insieme dei dati e quello dei risultati.

Macchina di Turing

Esattamente come *funzioni ricorsive* e λ-*calcolo*, le Macchine di Turing sono un modello di calcolo. Nella sua versione più tradizionale una *MdT* si presenta come un dispositivo che accede ad un nastro potenzialmente illimitato diviso in celle, ciascuna contenente un simbolo appartenente ad un dato alfabeto Σ (alfabeto della macchina), comprendente i simboli # e $\triangleright$ che denotano rispettivamente l'assenza di informazione nella cella e la marca di inizio stringa.

- la MdT opera tramite un *cursore*, che può scorrere sul nastro in entrambe le direzioni e scrivere i caratteri σ_i appartenenti all'alfabeto Σ
- gli *stati* q_i della macchina appartengono all'insieme finito degli stati Q e identificano istante per istante le informazioni contenute nella computazione

  $\boxed{\triangleright\ a\ b\ \dots\ a\ \underline{a}\ b\ \#\ \#}$

- il meccanismo che fa evolvere la computazione della macchina è detta *funzione di transizione*[3] δ e consente, partendo dallo stato q_x e dal carattere σ_x presente sulla cella puntata dalla testina, di portare la macchina in un'altro stato q_y, scrivere un carattere σ_y su tale cella precedentemente occupata da σ_x ed eventualmente spostare la testina ($\Lleftarrow$, $\Rrightarrow$, $\equiv$)

Def. una Macchina di Turing è definita formalmente come la quadrupla

$$M = (Q, \Sigma, \delta, q_0)$$

potrebbe essere conveniente modificare la definizione di MdT estendendo la quadrupla ad una quintupla togliendo gli stati di accettazione e rifiuto dall'insieme Q, aggiungendoli in un insieme a parte: $M = (Q \backslash \{q_A, q_R\}, \Sigma, \delta, q_0, \{q_A, q_R\})$. *Così facendo evito che la macchina vada in stallo.*

- Q è l'insieme finito degli stati q_i che non contiene lo stato di accettazione h
- Σ è l'insieme finito dei simboli (alfabeto della macchina) al quale non appartengono i simboli di spostamento $\Lleftarrow$, $\Rrightarrow$, $\equiv$
- $\delta \subseteq (Q \times \Sigma) \times (Q \cup h) \times \Sigma \times \{\Lleftarrow, \Rrightarrow, \equiv\}$ è la relazione di transizione, tale che il carattere puntato dal cursore non possa mai trovarsi a sinistra di $\triangleright$
- $q_0 \in Q$ è lo stato iniziale

[3]fintanto che verranno trattare MdT deterministiche non è errato considerare δ una funzione piuttosto che una relazione come invece è specificato nella definizione formale

Computazione, configurazione e transizione

Alla luce della definizione formale di MdT, è importante analizzare meglio la relazione di transizione δ, definita come la quadrupla $((q_x, \sigma_x), q_y, \sigma_y, D)$, dove

- (q_x, σ_x) è la coppia stato di partenza, carattere puntato dal cursore
- q_y è lo stato di arrivo
- σ_y è il carattere scritto nella cella precedentemente occupata da σ_x
- D è lo spostamento del cursore ($\Lleftarrow$, $\Rrightarrow$, $\equiv$)

Adesso è facilmete intuibile che, una macchina che si trova nella medesima configurazione di partenza e compie il medesimo spostamento del cursore, si troverà nella medesima configurazione di arrivo.

Questa constatazione permette di restringere la relazione δ in modo che sia una funzione rispetto ai suoi primi due argomenti, ovvero che $\delta(q_x, \sigma_x) = (q_y, \sigma_y, D)$.

Lezione 02 2020.02.18

Hello

Tristique sed, semper in, hendrerit non, sem. Vivamus dignissim massa in ipsum. Morbi fringilla ullamcorper ligula. Nunc turpis. Mauris vitae sapien. Nunc luctus bibendum velit.

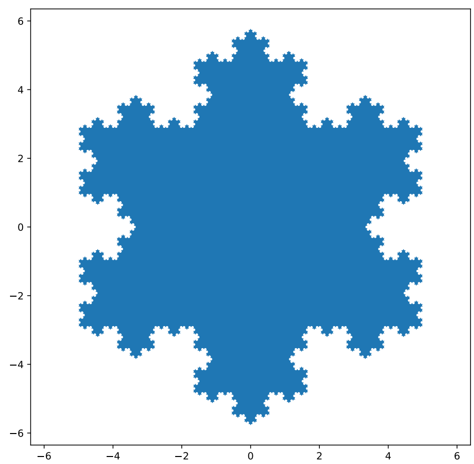

Figura 1: Hello

Morbi faucibus volutpat sapien. Nam ac mauris at justo adipiscing facilisis. Nunc et velit. Donec auctor, nulla id laoreet volutpat, pede erat feugiat ante, auctor facilisis dui augue non turpis. Suspendisse mattis metus et justo. Aliquam erat volutpat. Suspendisse potenti. Nam hendrerit lorem commodo metus laoreet ullamcorper. Proin vel nunc a felis sollicitudin pretium. Maecenas in metus at mi mollis posuere. Quisque ac quam sed massa adipiscing rutrum. Vestibulum ipsum. Phasellus porta sapien. Maecenas venenatis tellus vel.