Fondamenti di Informatica

per la Sicurezza

a.a. 2005/06

Teoria della computazione

Stefano Ferrari

Università degli Studi di Milano
Dipartimento di Tecnologie
dell'Informazione

Aforisma

Computer Science is no more about computers

than astronomy is about telescopes.

Edsger Dijkstra

Scienza informatica

Ogni disciplina scientifica richiede un inquadramento teorico che ne definisca:

- gli scopi;
- i limiti;
- le potenzialità.

La nascita dell'informatica risale agli anni '30.

Dagli anni '40 la tecnologia ha prodotto strumenti di calcolo sempre più potenti.

È una mera (e fortunata) coincidenza.

Scopi e potenzialità dell'informatica

L'informatica descrive:

- la codifica
- l'analisi
- la manipolazione
- la trasmissione

dell'informazione a prescindere dal particolare strumento di calcolo.

I principi dell'informatica saranno validi anche per i calcolatori che soppianteranno gli attuali.

Storia della matematica

Semplificando e senza pretese di completezza:

- Greci:
  - numeri: entità di interesse pratico (pitagorici a parte);
  - geometria: entità intuitive e assiomi ragionevoli;
- Descartes, Newton e tanti altri (1600–1700):
  - geometria analitica;
  - calcolo differenziale e integrale;
- Lobacevskij, Cantor, Boole (1700–1900):
  - geometrie non euclidee;
  - assiomatizzazione degli insiemi;
  - infinito.

Quadro storico del 1900

- **Hilbert** (1879)
  Programma formalista della matematica:
  - una teoria matematica formata da un insieme di assiomi e di regole di deduzione;
  - una teoria coerente e completa;
  - sviluppo meccanico della matematica.
- **Russel** (1902)
  Paradosso che mina la coerenza della teoria degli insiemi (sui quali è costruito tutto il resto!):
  - l'insieme degli insiemi che non hanno se stessi come elemento, ha se stesso come elemento?

Incompletezza

- **Gödel** (1931)

  Creare un sistema matematico completo e coerente è impossibile:

  - è possibile formulare asserzioni che non possono essere dimostrate né vere, né false;
  - non si può esser certi che gli assiomi dell'aritmetica non portino a contraddizioni.

In altre parole: non si può ridurre la matematica al calcolo automatico di teoremi a partire da assiomi e regole di deduzione.

Ma allora cosa è calcolabile?

Teoria computazionale

- **Turing**, **Church** (1936)

  Formalizzazione del processo di calcolo:

  - macchina calcolatrice ideale;
  - sistemi formali di calcolo;
  - esistenza di funzioni non calcolabili.

Tesi di Church

Tesi di Church-Turing

La classe delle funzioni intuitivamente calcolabili coincide con quella delle funzioni calcolabili dalla macchina di Turing.

È un'asserzione praticamente indimostrabile!

Ciò è dovuto all'impossibilità di formalizzare il concetto di "calcolabilità intuitiva".

È possibile dimostrarne la falsità, però nessuno c'è ancora riuscito.

Viene comunemente utilizzata come ipotesi.

Formulazioni equivalenti

- **λ-calcolo** (Church, 1936);
- **macchina di Turing** (Turing, 1936);
- **funzioni ricorsive parziali** (Kleene, 1936);
- **macchina di Post** (Post, 1936);
- **calcolo dei combinatori** (Schofinkel, 1924; Curry, 1958);
- **algoritmi di Markov** (Markov, 1960).

Calcolabilità e algoritmi

Ci sono diversi gradi di calcolabilità:

- funzioni facili da calcolare;
- funzioni difficili da calcolare;
- funzioni che non si sa come calcolare:

$$f(x) = \begin{cases} 1 & \text{esistono almeno } x \text{ “5” in } \pi \\ 0 & \text{altrimenti} \end{cases}$$

- funzioni che non si sa se si possono calcolare:

$$g(x) = \begin{cases} 1 & \text{esistono esattamente } x \text{ “5” in } \pi \\ 0 & \text{altrimenti} \end{cases}$$

Algoritmi e funzioni

Un algoritmo è una sequenza univoca di istruzioni che manipola i dati in ingresso e fornisce i dati in uscita.

Come tale, è rappresentabile come una funzione.

Non è restrittivo limitare i dati in ingresso e in uscita ai soli numeri naturali.

Quindi un algoritmo, A, realizza una funzione su $\mathbb{N}$, $f_A \colon \mathbb{N} \longrightarrow \mathbb{N}$.

Relazione algoritmo-funzione

Una funzione f associa ad un valore x un valore $f(x)$.

Un algoritmo che calcola f è un metodo per trovare $f(x)$ dato x.

È una relazione simile a quella che lega un teorema e una sua dimostrazione:

- possono esserci più algoritmi che calcolano la stessa funzione;
- se non conosciamo un algoritmo di calcolo, non è detto che la funzione non sia calcolabile.

Numerabilità degli algoritmi

Gli algoritmi sono composti da una successione finita di istruzioni.

Ci sono un numero finito di istruzioni.

È quindi possibile assegnare un codice ad ogni algoritmo.

Indicheremo con P_n, l'n-esimo algoritmo.

Funzioni calcolabili

Non tutte le funzioni su $\mathbb{N}$ sono calcolabili.

Definiamo la funzione $h\colon \mathbb{N} \longrightarrow \mathbb{N}$ come:

1. input k;
2. trova P_k;
3. output $P_k(k) + 1$.

Non esiste j tale che P_j calcola $h(\cdot)$!

Infatti $P_j(j)$ varrebbe $P_j(j) + 1$: impossibile.

Alcune considerazioni

Abbiamo definito una funzione su $\mathbb{N}$ che non è possibile calcolare.

Tale funzione sembrerebbe ben definita:

- ogni singolo passo è matematicamente ben definito;
- ogni singolo passo è intuitivamente calcolabile.

La conclusione è:

- non tutto ciò che possiamo definire matematicamente è calcolabile;
- serve qualche definizione più formale di ciò che è intuitivamente calcolabile.

Teoria della computazione

La teoria della computazione studia le funzioni astrattamente calcolabili, senza preoccuparsi dell’efficienza dell’algoritmo che le computa.

Nasce negli anni ’30, a seguito dell’esigenza, sorta nell’ambito degli studi di logica:

- di fornire un equivalente rigoroso al concetto di algoritmo;
- di indagare le possibilità ed i limiti dei metodi di calcolo effettivi.

Caratteristiche della computazione (1)

Il processo di calcolo è definito dalle seguenti caratteristiche:

1. un algoritmo è di lunghezza finita;
2. esiste un esecutore che può effettuare le istruzioni dell’algoritmo;
3. il calcolo è deterministico;
4. l’esecuzione avviene per passi discreti;
5. l’esecutore dispone di una memoria ausiliaria per immagazzinare i risultati intermedi dell’elaborazione;

Caratteristiche della computazione (2)

6. non c'è limite alla lunghezza dei dati;
7. non c'è limite alla quantità di memoria ausiliaria;
8. le istruzioni hanno una complessità finita;
9. l'esecuzione termina dopo un numero di passi finito, ma illimitato;
10. l'esecuzione può non terminare.

Nota: la caratteristica 10 dice che una funzione calcolabile può essere non definita per alcuni elementi del dominio.