Capitolo 1

Introduzione

1.1 Argomenti

Pensiamo a un punto lontano nel tempo 2150: necessità di tenere traccia di n numeri di telefono, risparmiando spazio scrivendoli per intero.
Idea di rubrica: non scrivere il numero interamente, ma un programma più breve del numero di telefono. Questa sembra una memorizzazione efficiente.
Tuttavia questo non è sempre vero:

$$size(n) <= size(P) \tag{1.1}$$

Causato dall'esistenza dei numeri casuali, ovvero esistono programmi di dimensione superiore al numero stesso.
I numeri casuali sono in numero infinto e non è possibile realizzare un programma che determini se un numero è casuale.

Conclusione: non tutto si può risolvere per mezzo di un calcolatore, questi problemi sono ben formalizzati. Alcuni esempi sono:

- *Haiting Input* (problema della non determinazione di un programma dato uno specifico input);
- Correttezza dei programmi.

L'obiettivo del corso è:

- Quali sono i problemi risolvibili con una procedura effettiva;
- Cosa è una procedura effettiva;
- Cosa vuol dire che un problema è risolto;
- Problemi risolvibili e non risolvibili.

Considerano sempre che le risorse non sono un problema.

1.2 Storicamente

L'informatica inizia a trattare i problemi di computabilità ancora prima di nascere.

Dijkstra parla di "informatica scienza delle procedure".
L'informatica è figlia della logica intesa come studio dei maccanismi del ragionamento (da dalle premesse si riesce a trarre delle conclusioni).
Si parla di schemi di ragionamento che portano a delle varità sempre vere:
Ogni uomo è morale $\longrightarrow$ Socrate è un uomo $\longrightarrow$ Socrate è un uomo

Lullus cosruisce il primo esempio di meccanismo, la ruola lulliana. Crea frasi su cui l'uomo può ragionare.

Leibniz idea una lingua artificiale per mettere in relazione dei concetti (lingua per rappresentare le relazioni tra i pensieri umani). In questo modo viene favorita una lingua universale che permette l'apprendimento e lo sviluppo del sapere:

- *Caracteristica Universalis*: linguaggio universale che permette di esprimere nozione e concetto. Usa simboli, nozioni di base e grammatica.
- *Calculus ratiocinator*: ragionamento attraverso una manipolazione di simboli.

Questo permette di avere un metodo matematico che permette di risolvere le controversie. Un esempio pratico di questa idea è la *Staffelwalze*, la prima macchina calcolatrice che computa le quattro operazioni aritmetiche.

Boole riprende le idee di *Leibniz*. vuole interpretare con un'algebra (formalismo simbolico) i meccanismi della logica: algebra booleana che fornisce uno strumento di congiunzione tra i concetti. Per farlo tutto viene regolamentato da leggi $\in$ 0,1 che consentendo la creazione di leggi universalmente accettate.

Baddage sviluppa idealmente delle macchine *general purpose* per il calcolo dei polinomi e a schede perforate programmabile con memoria e unità aritmetica. Quest'ultima ha permesso lo sviluppo di ADA.

Frege inventa la logica del primo ordine e si occupa di togliere la circolarità presente all'interno della Logica di *Boole*. Introduce i quantificatori esiste e per ogni. Tuttavia *Russel* individua un paradosso nei suoi ragionamenti $\{x|x \notin x\}$.

Hilbert ricerca un sistema matematico formale composto da regole e assiomi, che si dimostri consistenze e solido, senza alcun paradosso. In modo da ottenere un algoritmo in grado di definire se un teorema è conseguenza diretta dei suoi assimi.

Godel dimostra l'impossibilità delle aspettative di Hilbert. Esiste sempre qualcosa di vero impossibile da dimostrare. Inotre se T è assiomatizzabile non è possibile dimostrarne la consistenza interna.

Turing definisce il concetto di algoritmo e si inventa una macchina programmabile con funzione calcolabile per dimostrare che non tutti i teoremi sono risolvibili. In questo modo *Turing* dimostra l'esistenza di una macchina universale che funge da interprete (esecuzione di programmi sui dati di *input*). Tuttavia la realizzazione del primo calcolatore programmabile fallisce a causa di limiti di risorse.

Von Neumann realizza il primo progetto di macchina calcolatrice dall'idea della macchina di *Turing* universale realizzando il *computer* digitale moderno.

Capitolo 2

Algoritmo e calcolabilità

"Gli algoritmi sono pochi, le cose da calcolare tante."

Un algoritmo o procedura è una sequenza di passi elementari (meccanici, senza intelligenza) che permettono di ottenere un certo risultato.

Passi elementari: somma di due cifre.
172+ 43 = 215
È perciò un procedimento guidato deterministico $f : \{input\} \longrightarrow \{output\}$ dove f è calcolata dall'algoritmo.

La funzione f è calcolabile se esiste un algoritmo che la calcola. Attenzione se esiste, non se la conosciamo.

Esempi di funzioni calcolabili:

- $f(x, y) = x + y$
  La somma è calcolabile.
- $f(x, y) = \begin{cases} 1 & \texttt{x primo} \\ 0 & \texttt{x non primi} \end{cases}$
  Individuazione numeri primi è calcolabile.
- $f(x) = x^{\text{mo}}$
  È difficile vanno calcolati tutti mo, ma è calcolabile.
- $g(n) = n^{\text{ma}}$ cifra di π
  È difficile vanno calcolati tutti mo, ma è calcolabile.
- $h(n) = \begin{cases} 1 & \texttt{se in } \pi\texttt{ci sono esattamente 5 cifre consecutive} \\ 0 & \texttt{altrimenti} \end{cases}$
  Le cifre di π vanno calcolate un pò alla volta. Se π è normale allora la costante è 1, altrimenti non lo so.
- $h'(n) = \begin{cases} 1 & \texttt{se esistono almeno n cifre 5 consecutive in } \pi \\ 0 & \texttt{altrimenti} \end{cases}$
  Vale come sopra, se π è normale allora la costante è 1, altrimenti non

posso dire se è calcolabile.
Inoltre posso esprimere $k = \sup$ {n | n 5 consecutivi in π } =

1. k finito;
2. ∞ se π normale.

Ovvero:

1. $h'(n) = \begin{cases} 1 & \texttt{se n <= k} \\ 0 & \texttt{altrimenti} \end{cases}$
2. $h'(n) = 1 \ \forall \texttt{n}$