Il sapere tende oggi a caratterizzarsi non più come un insieme di contenuti ma come un insieme di metodi e di strategie per risolvere problemi.

L'informatica si interessa delle tecniche e dei metodi per

la rappresentazione,
l'elaborazione,
la conservazione e
la trasmissione

delle informazioni.

INFORMAZIONE + AUTO**MATICA** ⇒ **INFORMATICA**

Lo strumento privilegiato dell'informatica è il *computer*, talvolta chiamato anche *calcolatore* o *elaboratore*.

I calcolatori sono soltanto dei rapidissimi e precisi esecutori di ordini.

L'attività creativa, dominio dell'uomo, consiste nel ricercare e scoprire un elenco di ordini da far eseguire ad una macchina, come soluzione ad uno qualsiasi dei problemi che si possono effettivamente risolvere.

PROBLEMA

Un problema nasce sempre da una situazione di parziale ignoranza di alcuni aspetti della realtà e consiste nel tentativo di determinare tali aspetti sulla base di conoscenze iniziali in nostro possesso.

Risolvere un problema significa:

"ricercare ed esprimere un elenco di istruzioni che, interpretate da un esecutore, conducono da determinate informazioni iniziali ad altre finali soddisfacenti ad un criterio di verifica".

Istruzioni + esecutore ⇒ azioni

Informazioni iniziali : *dati* del problema

Informazioni finali : *obiettivo* del problema

dati ⇒ strategia risolutiva ed elaborazione ⇒ obiettivo

Esempio di problema:
“Determinare il perimetro di un quadrato avente la superficie di 100 cm^2"

Obiettivo: perimetro del quadrato
Dati: superficie del quadrato

Strategia risolutiva (o *procedimento*):
- estrarre la radice quadrata dell'area per trovare il lato del quadrato
- moltiplicare il lato per 4 per ottenere il perimetro

Elaborazione: $\sqrt{100} = 10; \quad 10 \times 4 = 40$

Il procedimento ci consente di risolvere tutta una classe di problemi (*problemi parametrici*).

In molti casi sono i dati iniziali a determinare la strategia di risoluzione; in altri è la strategia prescelta che determina le informazioni di cui si ha bisogno.

In certi problemi di tipo non deterministico una più vasta gamma di informazioni iniziali ci assicura la possibilità di prospettare più soluzioni alternative, al fine di scegliere quella ottimale.

Strategia risolutiva = **algoritmo**

Un *algoritmo* è una *sequenza ordinata di istruzioni* le quali definiscono delle azioni ben precise da compiere sui dati del problema e che, una volta eseguite fedelmente, conducono immancabilmente al risultato richiesto.

Caratteristiche di un algoritmo:

- finitezza
- non ambiguità
- generalità

Un algoritmo deve essere formulato in modo che l'*esecutore* attribuisca alle istruzioni lo stesso significato che ad esse attribuisce il progettista (*risolutore* del problema).

Comunicazione

sorgente ⇒ messaggio ⇒ destinatario

Codice

Es.:

lingua naturale
linguaggio matematico
linguaggio di programmazione

Algoritmo formulato in un *linguaggio di programmazione* ⇒ *programma*

Azioni elementari (o *passi*):
azioni non scomponibili in azioni più semplici.

Processo (o *esecuzione*):
azione composta da una sequenza di passi.

Le *istruzioni* descrivono:
– l'elenco degli oggetti da manipolare con i loro nomi;
– l'insieme delle azioni che devono essere eseguite su di essi per poter ottenere un risultato;
– l'ordine esatto con cui le azioni debbono essere eseguite se si vuol ottenere un certo risultato, nonché le condizioni che debbono essere verificate perché ad una azione ne segua una piuttosto che una cert'altra.

passo : dinamico
istruzione : statica

Legame “algoritmo — esecutore”

La precisione e il dettaglio degli ordini contenuti in un algoritmo sono concetti relativi alle capacità di chi sarà poi chiamato ad interpretare (ed eseguire) quegli ordini.

Problema: “Determinare il MCD (Massimo Comun Divisore) di due numeri naturali dati # 0"

Algoritmo MCD1:

1) si scompongano i 2 numeri in fattori primi;
2) si prendano in considerazione i soli fattori comuni;
3) dall'elenco di fattori comuni nei passi di esecuzione dell'istruzione (2) si considerino quelli con l'esponente più piccolo;
4) si moltiplichino fra di loro i fattori individuati nei passi di esecuzione dell'istruzione (3) — il risultato è il MCD cercato.

Algoritmo MCD2:

1) costruire l'insieme dei divisori del primo numero;
2) costruire l'insieme dei divisori del secondo numero;
3) costruire l'intersezione fra gli insiemi individuati nei passi di esecuzione delle istruzioni (1) e (2);
4) cercare nell'insieme costruito nei passi di esecuzione della istruzione (3) l'elemento più grande — esso è il MCD cercato.

Algoritmo MCD3 (detto ‘di Euclide’):

1) dividere il primo numero (m) per il secondo (n) — chiamare r il resto della divisione;

2) se $r = 0$ hai finito — n è il MCD cercato;

3) se $r \# 0$ si operano i seguenti cambiamenti:
   $m \Leftarrow n$,
   $n \Leftarrow r$;

4) tornare all'istruzione (1).

Tavola di traccia per l'algoritmo di Euclide con $m = 770$ e $n = 175$.

Istruzione		m	n	r
1	$r \Leftarrow m \bmod n$	770	175	70
2	$r = 0$?, no			
3	$r \neq 0$?, sì	175	70	
4	vai a 1			
1	$r \Leftarrow m \bmod n$			35
2	$r = 0$?, no			
3	$r \neq 0$?, sì	70	35	
4	vai a 1			
1	$r \Leftarrow m \bmod n$			0
2	$r = 0$?, sì	MCD = 35		

Algoritmo di Newton

Problema: "Determinare la radice quadrata *r* di un numero *a*":

$$r = \sqrt{a}$$

r' : valore approssimato r'': approssimazione migliore

$$r'' = \frac{1}{2}\left(r' + \frac{a}{r'}\right)$$

Es.: $a = 9$, $r' = 4$ (approssimazione grossolana)
(approssimazione migliore)

$$r'' = \frac{1}{2}\left(4 + \frac{a}{4}\right) = 3.125$$

Metodo di calcolo ("di Newton"):

r_0 : stima iniziale di *r* (anche grossolana, ad es., $r_0 = a$)
fino ad ottenere un valore r_n che sia sufficientemente vicino ad *r*.

Es.:

$$r_1 = \frac{1}{2}\left(r_0 + \frac{a}{r_0}\right)$$
$$r_2 = \frac{1}{2}\left(r_1 + \frac{a}{r_1}\right)$$
$$\vdots$$

$\sqrt{16}$

r_i	$r_{i+1} = ½(r_i + a/r_i)$	
$r_0 = 16$	$r_1 = ½(16+16/16)$	$= 8.5$
$r_1 = 8.5$	$r_2 = ½(8.5+16/8.5)$	$= 5.19$
$r_2 = 5.19$	$r_3 = ½(5.19+16/5.19)$	$= 4.14$
$r_3 = 4.14$	$r_4 = ½(4.14+16/4.14)$	$= 4.002$
.	.	
.	.	

La frase “fino ad ottenere un valore r_n che sia sufficientemente vicino ad r” non ci dice quando ci dobbiamo fermare e non possiamo quindi dire che il nostro metodo “si arresta in ogni caso per fornire un risultato”.

⇒ Il nostro metodo non è un algoritmo.

Possiamo sostituire la frase ambigua con la seguente:

“continuare i cicli fino a che la differenza

$$|a - r^2_{i+1}| \leq \textit{epsilon} \text{ (prefissato)}$$

oppure

$$|r_{i+1} - r_i| \leq \textit{epsilon} \text{ (che equivale alla precedente)”}$$

Più *epsilon* è piccolo, più preciso è il risultato.

Es.: $a = 23$, $r_0 = 23$, *epsilon* = 0.01

$r_0 = 23$	$r_{i+1} = \frac{1}{2}(r_i + 23/r_i)$
$r_0 = 23$	$r_1 = \frac{1}{2}(23+23/23) = 12$ $\lvert 12 - 23 \rvert \leq 0.01$? no
$r_1 = 12$	$r_2 = \frac{1}{2}(12+23/12) = 6.96$ $\lvert 6.96 - 12 \rvert \leq 0.01$? no
$r_2 = 6.96$	$r_3 = \frac{1}{2}(6.96+23/6.96) = 5.13$ $\lvert 5.13 - 6.96 \rvert \leq 0.01$? no
$r_3 = 5.13$	$r_4 = \frac{1}{2}(5.13+23/5.13) = 4.81$ $\lvert 4.81 - 5.13 \rvert \leq 0.01$? no
$r_4 = 4.81$	$r_5 = \frac{1}{2}(4.81+23/4.81) = 4.80$ $\lvert 4.80 - 4.81 \rvert \leq 0.01$? sì
$r_5 = 4.80$	

$\Rightarrow \quad \sqrt{23} \simeq 4.80 \qquad (\sqrt{23} = 4.79583...)$

L'arresto del calcolo è garantito dal verificarsi della condizione

$$|r_{i+1} - r_i| \leq \mathit{epsilon}$$

Tutti i passi sono esattamente definiti. Possiamo parlare dunque di “algoritmo di Newton”.

Requisiti di un *algoritmo* e di un *buon esecutore*:

1) finitezza della descrizione (numero finito di istruzioni)

2) non limitatezza dei dati in ingresso

3) non limitatezza dei dati in uscita

4) non limitatezza del numero di passi eseguibili

5) definitezza (a ogni passo di un algoritmo deve corrispondere una istruzione definita con precisione: le operazioni da compiere devono essere specificate rigorosamente e senza ambiguità)

6) esistenza di un limite finito alla complessità delle istruzioni eseguibili

7) disponibilità per l'esecutore di una memoria illimitata per conservare dati

8) l'esecutore opera in modo discreto (opera cioè trasformazioni successive su insiemi di dati, separate da un intervallo di tempo finito)

9) terminazione (un algoritmo deve sempre terminare dopo un numero finito di passi)

L'algoritmo di Euclide, ad esempio, soddisfa la condizione (9), in quanto ad ogni passo si ha

$$r_i < r_{i-1}$$

Dopo un numero di passi a priori sconosciuto (anche molto grande) l'algoritmo euclideo termina.

È opportuno introdurre una nozione più generale di quella di algoritmo, che ci permetta di esprimere dei procedimenti di calcolo per cui non sia imposta a priori la terminazione per tutti gli ingressi.

Con il nome di *metodi computazionali* ci si riferisce ad una classe di procedimenti di calcolo che possono in certe circostanze non terminare.

Per questi ultimi possiamo riformulare la (9).

9) Sono ammesse esecuzioni con un numero infinito di passi (l'esecutore entra in un *loop infinito*)

Gli algoritmi devono essere considerati casi speciali dei metodi computazionali: cioè un algoritmo è certamente un metodo computazionale, mentre in generale non è vero il contrario.

Necessità di una definizione *formale* (rigorosa) di algoritmo.

“Qualunque processo che possa essere eseguito da una macchina può essere descritto sotto forma di algoritmo”

e, inversamente:

“Tutti gli algoritmi costruiti fino ad oggi possono essere eseguiti da una macchina”.

Fornire una nozione rigorosa di algoritmo significa anche comprendere meglio il concetto di `esecutore automatico' (**automa**).

“La nozione formale di algoritmo e quella di esecutore automatico coincidono”.