Algoritmi, Efficenza, Problem Solving e RAM

▼ ***INDICE***

0 Introduzione

La definizione di informatica proposta dall' ACM(Association for Computing Machinery), ossia una delle principali organizzazioni scientifiche di informatici in tutto il mondo, è la seguente: " L'informatica è la scienza degli algoritmi che descrivono e trasformano l'informazione: la teoria, analisi, progetto, efficienza, realizzazione e applicazione".

Quindi l'algoritmo è un concetto fondamentale per l'informatica.

1 Algoritmi, Efficenza, Problem Solving e RAM

1.1 Che cos’è un algoritmo?

✏️ ***DEFINIZIONE DI ALGORITMO:***

Un algoritmo è una sequenza di comandi elementari ed univoci che terminano in un tempo finito ed operano su strutture dati.

Un comando si dice elementare quando non può essere scomposto in comandi più semplici, invece è univoco quando può essere interpretato in un solo modo

1.2 Strutture dati

Per risolvere i problemi abbiamo bisogno, ovviamente, di gestire i relativi dati, per questo c’è bisogno di definire le opportune strutture dati.

✏️ ***DEFINIZIONE DI STRUTTURE DATI:***

Le strutture dati sono strumenti necessari per organizzare i dati veri e propri semplificandone l’accesso e la modifica.

> ⚠️ ***ATTENZIONE:***
>
> Non esiste una struttura dati che sia adeguata per ogni problema, per cui è necessario conoscere proprietà, vantaggi e svantaggi delle principali strutture dati così da poter scegliere di volta in volta quale sia quella più adatta al problema.

Il progetto o la scelta della struttura dati da adottare nella soluzione di un problema è un aspetto fondamentale per la risoluzione del problema, al pari del progetto dell’algoritmo.

Per questo, gli algoritmi e le strutture dati fondamentali vengono sempre studiati e illustrati assieme.

1.3 Efficienza di un algoritmo

Un algoritmo per essere utilizzabile deve concludersi e produrre il suo output entro un tempo “ragionevole”.

Difatti un aspetto fondamentale nello studio degli algoritmi è la loro efficienza, ossia la stima delle loro esigenze nel tempo di esecuzione e la quantità di memoria richiesta.

Questo perché:

- I calcolatori sono molto veloci, ma non infinitamente veloci;
- La memoria è economica e abbondante, ma non è né gratuita né illimitata.

📎 ***ESEMPIO SULL'EFFICIENZA DI UN ALGORITMO:***

Immaginiamo di dover ordinare una lista di $n = 10^6$ e a nostra disposizione abbiamo 2 calcolatori:

- un calcolatore veloce, chiamato V, in grado do svolgere 10^9 operazioni al secondo;
- un calcolatore lento, chiamato L, in grado di svolgere 10^7 operazioni al secondo.

Immaginiamo anche di saper sviluppare 2 algoritmi di ordinamento, che sono:

1. L'algoritmo Insertion Sort, richiedente $2 \cdot n^2$ operazioni(più lento);
2. L'algoritmo Merge Sort, richiedente $50 \cdot n \cdot log(n)$ operazioni(più veloce).

Ci chiediamo se la maggior velocità del calcolatore V sia in grado di contro-bilanciare la maggior lentezza dell'algoritmo IS(Insertion Sort).

Proviamo quindi a calcolare il costo temporale di entrambe le scelte:

$$V(IS) = \frac{2 \cdot (10^6)^2 \ operazioni}{10^9 \ operazioni/sec} = 2000 \ sec \ \cong \ 33 \ minuti$$

$$L(MS) = \frac{5 \cdot 10^6 \cdot log(10^6) \ operazioni}{10^7 \ operazioni/sec} = 100 \ sec \ \cong \ 1.5 \ minuti$$

Notiamo quindi che, nonostante la differenza di caratteristiche hardware, la scelta dell'algoritmo è cruciale per l'efficienza.

Proviamo ad aumentare la dimensione dell'input da 10^6 a 10^7:

$$V(IS) = \frac{2 \cdot (10^7)^2 \ operazioni}{10^9 \ operazioni/sec} = 55.5 \ ore \ \cong \ 2.3 \ giorni$$

$$L(MS) = \frac{5 \cdot 10^7 \cdot log(10^7) \ operazioni}{10^7 \ operazioni/sec} \cong \ 19.5 \ minuti$$

Aumentando l'input di un solo ordine di grandezza, la differenza in termini di costi temporali dei due algoritmi è abissale.

1.4 Problem Solving

✏️ ***DEFINIZIONE DI PROBLEM SOLVING:***

Il Problem Solving è un'attività che ha lo scopo di raggiungere una soluzione a partire da una situazione iniziale.

Per approcciarsi al Problem Solving bisogna seguire vari punti, che sono:

- Analisi del problema: ossia la comprensione e la identificazione del problema;
- Esplorazione degli approcci possibili: identificazione delle metodologie di soluzione tra i metodi noti;
- Selezione di approccio: scelta dell'approccio migliore;
- Definizione dell'algoritmo risolutivo: ossia l'identificazione dei dati e la progettazione della sequenza di passi elementari da applicare su di essi;

- Riflessione critica: dopo aver risolto il problema cercare eventuali errori e possibili migliorie.

Quali sono però i problemi da risolvere?

Nel nostro caso restringiamo la nostra attenzione sui problemi computazionali, che sono problemi nel quale bisogna descrivere in modo automatico una specifica relazione tra un insieme di valori in input e il corrispondente insieme di valori in output.

Un algoritmo può essere definito corretto se per ogni istanza di un problema computazionale, termina producendo l'output corretto. In tal caso diciamo che l'algoritmo risolve il problema.

📎 ***ESEMPIO DI PROBLEMA COMPUTAZIONALE:***

Definizione del problema:

Ordinare n numeri dal più piccolo al più grande.

Input(anche detto istanza del problema):

Sequenza di n numeri $a_1, a_2, ..., a_n$;

Output:

Permutazione in $a_1^1, a_2^1, ..., a_n^1$ della sequenza di input, allora $a_1^1 \leq a_2^1, ..., \leq a_n^1$

1.5 Random Access Machine(RAM)

Per poter valutare l'efficienza di un algoritmo è necessario analizzarlo, ossia fare una stima delle risorse che esso richiede per la sua esecuzione, senza che tale analisi sia influenzata da una specifica tecnologia che col tempo poi verrà superata.

Quindi per poter valutare l'efficienza di un algoritmo si usa la Random Access Machine(RAM) che è indipendente dalle caratteristiche tecniche di uno specifico calcolatore reale.

✏️ ***DEFINIZIONE DI RAM:***

La RAM è una macchina astratta, la cui validità e potenza concettuale risiede nel fatto che non diventa obsoleta con il progredire della tecnologia.

Le caratteristiche del modello RAM sono:

- Un singolo processore che esegue le operazioni sequenzialmente;
- Esistono solo operazioni elementari e l'esecuzione di ciascuna delle quali richiede per definizione un tempo costante(es.: operazioni aritmetiche, letture, scritture, salto condizionato, ecc…);
- Esiste un limite alla dimensione di ogni valore memorizzato ed al numero complessivo di valori utilizzati, dipendente dalle dimensioni delle word in memoria.

1.6 Criterio della misura di costo uniforme

Sia d la dimensione di bit di ogni parola contenuta in memoria. Se è soddisfatta l'ipotesi che ogni dato in input sia un valore minore di:

$$k = 2^d$$

ciascuna operazione elementare sui dati del problema verrà eseguita in un tempo costante, in questo caso parliamo di misura di costo uniforme.

Tale criterio non è sempre realistico perché è possibile che un dato del problema è più grande di k e quest'ultimo dovrà comunque essere memorizzato, ed in tal caso si useranno più parole di memoria.

Di conseguenza, anche le operazioni elementari su di essa dovranno essere ripetute per tutte le parole in memoria che lo contengono e quindi il tempo non sarà più costante.

📎 ***ESEMPIO SUL CRITERIO DELLA MISURA DI COSTO UNIFORME:***

```
def PotenzaDi2(n):
  x = 1
  for i in range(n): #Calcola il valore 2^d
    x = x * 2
  return x
```

Il tempo di esecuzione totale è proporzionale ad n, poiché si tratta di un ciclo seguito n volte, dove ad ogni iterazione vengono compiute 3 operazioni, ciascuna delle quali ha costo unitario:

1. Viene incrementato il contatore relativo al ciclo for;
2. Viene calcolato $x \cdot 2$;
3. Viene assegnato il risultato del calcolo ad x.