Elementi di Informatica

di

Luca Tallini

Anno Accademico 2002/2003

1 Che cosa è un computer

In questi appunti, dal titolo “Elementi di Informatica”, intendiamo spiegare i concetti basilari di Informatica, ovvero l’Informatica che ogni persona colta deve sapere. In questa sede, **per Informatica intendiamo la scienza che si occupa della teoria e pratica di cos’è, come è fatto, cosa si può fare e cosa si fa con un Computer.** Tale scienza, nata negli Stati Uniti intorno agli anni 1950, in lingua inglese prende il nome di “Computer Science” che significa appunto Scienza dei Computer.

Dato ciò, pensiamo sia naturale iniziare questi appunti con il dire che cosa sia un Computer. Ed è appunto questo lo scopo di questo capitolo. Come vedremo definire precisamente un Computer è una cosa non semplice che ha dato molto da pensare ai ricercatori da molto tempo a questa parte. **Intuitivamente, si può dire che un Computer sia un entità che esegua un lavoro in maniera “automatica”.** Ma che cosa significa esattamente “automatico” ? Ad esempio, si può ben dire che eseguire la ricetta di una pietanza possa essere fatto in maniera automatica. Se, per esempio, tale ricetta è quella degli spaghetti al pomodoro per 4 persone basta eseguire “alla cieca” le seguenti istruzioni nell’ordine.

I1 Prendere una pentola,
I2 mettere mezzo bicchiere di olio di oliva nella pentola,
I3 mettere due agli spezzettati nella pentola,
I4 mettere la pentola sul fuoco fino a che l’aglio è dorato,
I5 aggiungere 500g di pomodori nella pentola, e cuocere per altri 15 minuti,
I6 aggiungere 8 foglie di basilico nella pentola, e cuocere per altri 5 minuti.
I7 Prendere un’altra pentola più grande,
I8 riempirla di acqua,
I9 metterci un pugno di sale,
I10 mettere la pentola sul fuoco fino a che l’acqua non bolle,
I11 buttare 500g di spaghetti nella pentola, e cuocere per altri 9 minuti,
I12 scolare gli spaghetti.
I13 Mischiare gli spaghetti con il sugo nella prima pentola e servire.

Una qualsiasi persona, animale o cosa che realizzi una ricetta di cucina, possiamo dire che sia un’entità che esegue un lavoro (quello di cucinare la pietanza) in maniera automatica. È possibile dire la stessa cosa se il lavoro è quello di dipingere un quadro ? Ovvero, è possibile dire che una persona che dipinge un quadro lo fa in maniera automatica ? Su questo ci sono dei forti dubbi. Se ciò fosse vero, Leonardo da Vinci avrebbe dipinto la Gioconda, od ogni altro quadro gli fosse venuto in mente, in maniera automatica. Si noti quindi che il problema di definire esattamente che cosa sia un Computer equivale al problema di definire esattamente quello che si intende per automatico e, tra tutti i lavori, quali possono essere fatti automaticamente e quali no. In questa sezione ci occupiamo di risolvere questo problema. Cominciamo a dare la seguente definizione.

Definizione di Computer: un Computer è una macchina che computa, ovvero che esegue un tipo di lavoro in maniera automatica, ovvero che esegue un *algoritmo* specificato in un linguaggio che può essere capito dalla macchina.

Partendo da questa definizione, si vede subito che il concetto di Computer è strettamente legato a quello di algoritmo, e chiarito quest’ultimo, è chiarito il concetto di Computer e di automatico. Infatti, eseguire un lavoro automaticamente significa proprio eseguire il lavoro mediante un algoritmo. La parola algoritmo denota un concetto che è stato definito rigorosamente nella prima metà del ventesimo secolo ed esprime quello che noi tutti intendiamo intuitivamente per algoritmo. Ovvero, “un algoritmo è una insieme di istruzioni, scritte, date, ... , codificate in un certo

linguaggio, che definisce come si deve effettuare l'esecuzione di un certo lavoro". La definizione rigorosa alla quale si è pervenuti oggidì può essere data al seguente modo.

Definizione di algoritmo: un algoritmo è una particolare *macchina di Turing* (da Alan Turing (1912-1954) matematico e logico inglese) oppure un programma della *macchina di von Neumann* (da John von Neumann (1903-1957) matematico statunitense di origine ungherese).

La macchina di Turing e la macchina di von Neumann sono due modelli di calcolo (i.e.; modi di definire e/o eseguire algoritmi) specifici. In breve, si definisce il concetto di algoritmo come tutto ciò che può essere eseguito dai due tipi di macchine specifiche su menzionate. Vogliamo dire che, tra tutti i modelli di calcolo oggi esistenti, questi due giocano un ruolo importante. Il modello di calcolo descritto dalla macchina di Turing (introdotto nel 1936) è importante perché è il primo modello di calcolo che sia stato definito ed è tutt'ora usato come definizione teorica di algoritmo. Il modello di calcolo descritto dalla macchina di von Neumann (introdotto nel 1946) è importante perché traduce in pratica il concetto di algoritmo. In altre parole, **tutti** i piú moderni computer sono strutturati come una macchina di von Neumann. Definiremo la macchina di von Neumann nella Sezione 2.1.2.

2 Come è fatto un computer

In questo capitolo ci occuperemo di descrivere la struttura di un moderno computer. È possibile suddividere le varie componenti di tale struttura in due categorie: le componenti materiali, palpabili, concrete, fisiche e le componenti immateriali, impalpabili, concettuali, metafisiche. L'insieme delle componenti del primo tipo è comunemente chiamato *hardware*. Ad esempio, la tastiera è una componente hardware di un computer. L'insieme delle componenti del secondo tipo è chiamato *software.* Ad esempio, il programma Microsoft Word è una componente software di un computer. Nella prossima sezione ci occuperemo dell'hardware di un computer. Nella sezione 2.2 ci occuperemo del software di un computer.

2.1 Le componenti fisiche di un computer: l'hardware

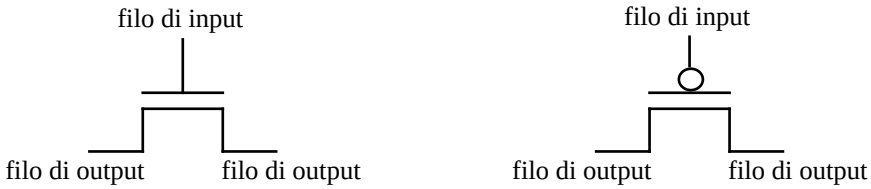
In questa sezione ci occuperemo di come è strutturato e come funziona l'hardware di un computer. In particolare, nella Sezione 2.1.1 ci occuperemo del "punto di incontro" tra hardware e software, ovvero di come è possibile "far ragionare" l'hardware mediante software. Nella Sezione 2.1.2 descriveremo la macchina di von Neumann descrivendo cosí la struttura generale di tutti i moderni computer e chiarendo i concetti di automatico, algoritmo e di macchina introdotti nel Capitolo 0. Nella Sezione 2.1.5 descriveremo i "cinque sensi" di un computer, ovvero descriveremo tutti gli apparati, oggidì più comuni, che permettono ad un computer di interagire con il mondo esterno ed in particolare con l'*utente* (la persona che usa il computer).

2.1.1 Il linguaggio binario e sua rappresentazione in hardware

Come noi umani, quando ragioniamo, pensiamo in una lingua composta da determinate parole in un certo alfabeto (tale lingua sarà l'italiano se siamo italiani, l'inglese se siamo inglesi, il cinese se siamo cinesi, ecc.), così un computer "ragiona" usando varie lingue tutte composte da parole nell'alfabeto binario costituito dalle sole due lettere 0 e 1. Nel caso dei computer, una lingua si chiama *codice* e una lettera di una parola di codice si chiama *bit*. Più specificatamente, **1 bit** è una lettera o variabile binaria (che quindi può assumere esclusivamente il valore 0 o 1) e **rappresenta l'unità di misura base della quantità di dati**. Le altre unità di misura per la quantità di dati fondamentali sono:

1 byte	= 8 bits		= 2^3 bits,
1Kilo byte	= 1024 bytes	= 2^{10} bytes	= 2^{13} bits,
1 Mega byte	= 1024 Kilo bytes	= 2^{10} Kilo bytes	= 2^{23} bits,
1 Giga byte	= 1024 Mega bytes	= 2^{10} Mega bytes	= 2^{33} bits,
1 Tera byte	= 1024 Giga bytes	= 2^{10} Giga bytes	= 2^{43} bits.

Figura 1: simboli schematici per i transistors.



transistor di tipo NPN transistor di tipo PNP

Di solito gli esseri umani rappresentano i numeri naturali nel sistema decimale ovvero come somme di potenze di 10 a coefficienti 0,1,..., 9. Ad esempio la rappresentazione decimale del numero 89 è $89 = \underline{8}\cdot 10^1 + \underline{9}\cdot 10^0$. La cifra più a destra è il numero di unità decimali di cui è composto il numero, la cifra appena a sinistra di questa è il numero di decine decimali, e così via per le centinaia, migliaia, eccetera. Viceversa, una sequenza finita di cifre decimali (ovvero, simboli da 0 a 9) può essere pensata come la rappresentazione decimale di un numero: quello il cui numero di unità decimali è pari alla cifra più a destra della sequenza, il cui numero di decine decimali è pari alla cifra appena a sinistra delle unità, e così via per le centinaia, migliaia, eccetera. Siccome l'alfabeto usato dai computer è quello binario, in essi ogni numero naturale è rappresentato nel sistema binario; ovvero come somme di potenze di 2 a coefficienti 0 e 1. Ad esempio, la rappresentazione binaria del numero 89 è:

$$\begin{aligned} 89 &= \underline{1}\cdot 2^6 + (89 - 64) = \underline{1}\cdot 2^6 + 25 \\ &= \underline{1}\cdot 2^6 + \underline{0}\cdot 2^5 + 25 \\ &= \underline{1}\cdot 2^6 + \underline{0}\cdot 2^5 + \underline{1}\cdot 2^4 + (25-16) = \underline{1}\cdot 2^6 + \underline{0}\cdot 2^5 + \underline{1}\cdot 2^4 + 9 \\ &= \underline{1}\cdot 2^6 + \underline{0}\cdot 2^5 + \underline{1}\cdot 2^4 + \underline{1}\cdot 2^3 + (9-8) = \underline{1}\cdot 2^6 + \underline{0}\cdot 2^5 + \underline{1}\cdot 2^4 + \underline{1}\cdot 2^3 + 1 \\ &= \underline{1}\cdot 2^6 + \underline{0}\cdot 2^5 + \underline{1}\cdot 2^4 + \underline{1}\cdot 2^3 + \underline{0}\cdot 2^2 + \underline{0}\cdot 2^1 + \underline{1}\cdot 2^0 \\ &= 1011001. \end{aligned}$$

Si noti che il funzionamento del sistema binario è esattamente lo stesso del sistema decimale, l'unica cosa che cambia è la base: nel sistema binario i numeri vengono espressi come somme di potenze di 2 anziché di 10. Come nel caso decimale, una sequenza finita di cifre binarie (ovvero, di simboli 0 e 1)) può essere pensata come la rappresentazione (binaria) di un numero. Ad esempio, la sequenza 1001011 rappresenta il numero

$$\begin{aligned} 1001011 &= \underline{1}\cdot 2^6 + \underline{0}\cdot 2^5 + \underline{0}\cdot 2^4 + \underline{1}\cdot 2^3 + \underline{0}\cdot 2^2 + \underline{1}\cdot 2^1 + \underline{1}\cdot 2^0 \\ &= \underline{1}\cdot 64 + \underline{0}\cdot 32 + \underline{0}\cdot 16 + \underline{1}\cdot 8 + \underline{0}\cdot 4 + \underline{1}\cdot 2 + \underline{1}\cdot 1 \\ &= 64 + 8 + 2 + 1 \\ &= 75. \end{aligned}$$

Per questi motivi, è anche possibile assumere che un computer ragioni pensando numeri naturali in quanto ragionare pensando numeri naturali o sequenze binarie sono due cose equivalenti.
Ogni componente hardware di un computer è costituito essenzialmente *da circuiti elettrici* ovvero da *transistors* (fatti di materiale semiconduttore) connessi tra loro da fili di metallo (alluminio o rame) detti *fili elettrici*. I fili elettrici memorizzano le lettere binarie costituenti una parola di codice oppure trasportano le lettere di una parola di codice da un transistor all'altro. I transistors permettono di manipolare (cambiare o meno) le lettere binarie memorizzate e/o trasportate dai fili. Ora, le lettere binarie sono rappresentate nei fili da due stati fisici particolari dei fili stessi: per convenzione, il filo trasporta e/o memorizza la lettera 0 se il suo potenziale elettrico è $GND = 0$ volt e trasporta e/o memorizza la lettera 1 se il suo potenziale elettrico è V_{DD} (oggidì, il valore di potenziale elettrico che comunemente si usa per rappresentare la lettera binaria 1 è circa $V_{DD} = 3$

volt, cinque anni fa era V_{DD} = 5 volt e fra cinque anni sarà V_{DD} = 2,5 volt o meno. Si cerca di ridurre tale valore per questioni di velocità e consumo di energia). Si noti che è proprio questo il punto di contatto tra hardware e software: l'hardware è costituito dai fili elettrici ed il software dallo stato fisico (specificatamente, dal valore di potenziale) dei fili elettrici.
Menzioniamo che, i circuiti elettrici in cui per convenzione hanno senso solo un numero finito di stati fisici (due nel nostro caso), sono detti *circuiti digitali*, mentre quelli per cui ha senso ogni stato fisico (che non faccia scoppiare la macchina) sono detti *circuiti analogici*. L'hardware di un computer è per lo più costituito da circuiti digitali binari (ovvero a due stati).

2.1.1.1 I transistors

I transistors opportunamente connessi tra loro manipolano (cambiano o meno) le lettere binarie (ovvero lo stato fisico) memorizzate e/o trasportate dai fili elettrici. Questa manipolazione avviene come descritto nella Sezione 2.1.1.2. In questa sezione diciamo che un transistor è un apparato a cui sono connessi tre fili: uno detto di input e due detti di output. Ora, se nel filo di input c'è 0 (oppure 1) i due fili di output non sono connessi tra loro. Se nel filo di input c'è 1 (oppure 0) i due fili di output sono connessi tra loro. Si può quindi dire che un transistor sia un interruttore elettrico controllato elettronicamente (invece che con un pulsante come gli usuali interruttori di lampadine che sono in ogni stanza di ogni casa). Ci sono quindi due tipi di transistors: quelli che sconnettono i due fili di output se nel filo di input c'è 0 e quelli che sconnettono i due fili di output se nel filo di input c'è 1. Il primo tipo di transistor è chiamato transistor di tipo NPN (o anche di tipo N) mentre il secondo tipo di transistor è chiamato transistor di tipo PNP (o anche di tipo P). I due tipi di transistors si comportano in maniera complementare e sono rappresentati graficamente come in Figura 1.

2.1.1.2 I gate booleani

In questa sezione vediamo come è possibile connettere tra loro i transistors in modo da manipolare le lettere binarie 0 e 1 e, per esempio, realizzare dei circuiti, detti *gate booleani*, che computino i connettivi logici fondamentali: NOT, AND e OR. Il connettivo NOT rappresenta la negazione ed è una funzione di una variabile binaria che a 0 associa NOT(0) = 1 e ad 1 associa NOT(1) = 0. Tale connettivo è rappresentato dalla seguente *tavola di verità*:

x	NOT(x)
0	1
1	0

Il connettivo AND rappresenta la congiunzione ed è una funzione di due variabili binarie definita dalla seguente *tavola di verità*:

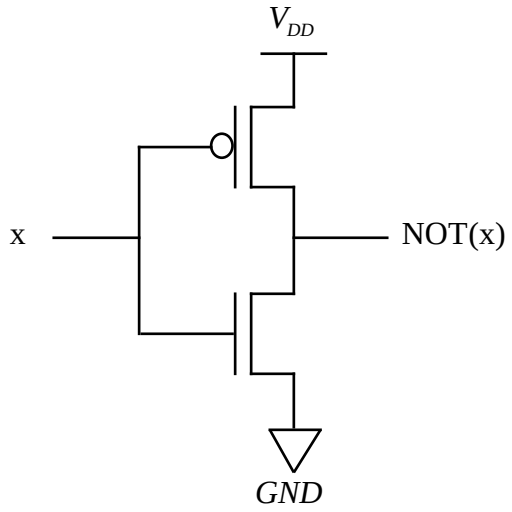
x	y	AND(x,y)
0	0	0
0	1	0
1	0	0
1	1	1

Il connettivo OR rappresenta la congiunzione ed è una funzione di due variabili binarie definita dalla seguente *tavola di verità*:

x	y	OR(x,y)
0	0	0
0	1	1
1	0	1
1	1	1

I connettivi NOT, AND e OR sono chiamati fondamentali perché componendo fra loro tali funzioni in modo opportuno è possibile ottenere tutte le funzioni di variabili binarie a valori binari.
Un modo per realizzare un gate booleano, detto anche *inverter*, che computi il connettivo logico NOT è quello definito dalla cosiddetta tecnologia CMOS (Complementary Metal Oxide Semiconductor) e raffigurato in Figura 2.
Come si può vedere dalla Figura 2, un inverter è costituito da un transistor di tipo PNP (sopra) e da un transistor di tipo NPN (sotto). Gli input dei due transistors sono connessi tra loro in un filo che trasporta il valore di input x. L'output di sopra del transistor di tipo PNP è ha V_{DD} volts (che corrisponde al valore logico binario 1). L'output di sotto del transistor di tipo NPN è ha *GND* volts (che corrisponde al valore logico binario 0). L'output di sotto del transistor di tipo PNP e l'output di sopra del transistor di tipo NPN sono connessi tra loro in un filo che trasporta il valore di output NOT(x). Vediamo come funziona il circuito. Se l'input x è 0, il transistor PNP ha i suoi fili di output connessi e il transistor NPN ha i suoi fili di output sconnessi. Quindi l'output dell'inverter è connesso con l'output di sopra del transistor PNP e quindi è al suo stesso potenziale elettrico, V_{DD} che rappresenta 1. Quindi se il filo di input all'inverter trasporta 0 il filo di output trasporterà 1. Viceversa, Se l'input x è 1, il transistor PNP ha i suoi fili di output sconnessi e il transistor NPN ha i suoi fili di output connessi. Quindi l'output dell'inverter è connesso con l'output di sotto del transistor NPN e quindi è al suo stesso potenziale elettrico, *GND* che rappresenta 0. Quindi se il filo di input all'inverter trasporta 1 il filo di output trasporterà 0.

Figura 2: NOT gate o inverter.



Analoghi circuiti si possono progettare per i connettivi logici AND e OR e per celle di memoria (ovvero circuiti che memorizzino una lettere dell'alfabeto binario ovvero i bit di dati).

2.1.2 l'architettura di von Neumann

Tutti i più moderni computer sono strutturati secondo il modello di macchina di von Neumann e sono realizzati con una tecnologia che prende il nome di VLSI (Very Large Scale Integration). Tale tecnologia permette di miniaturizzare i circuiti elettrici.

In VLSI, tutti i circuiti elettrici costituenti la logica di un computer (o un sistema in genere) sono suddivisi in vari pezzettini rettangolari di silicio detti *chips* connessi tra loro da fili elettrici e situati su un circuito stampato o scheda (circuit board). Oggi, ogni chip può contenere all'incirca fino a 34 milioni di transistors ed ha dimensioni fino a 1,5 cm^2. Questo perché la più piccola forma che può essere delineata su di un chip (the minimum feature patternable) ha una grandezza di $\lambda = 0{,}18\ \mu m$ (1 μm = 1 micron = un milionesimo di metro). È importante notare che tale parametro λ tende a diminuire con lo sviluppo della tecnologia VLSI e presto saranno sul mercato processori prodotti con un processo a $\lambda = 0{,}13\ \mu m$ e fra non molto sarà possibile realizzare il processo di fabbricazione con $\lambda = 0{,}11\ \mu m$.
Esponiamo che cosa è la macchina di von Neumann; chiarendo in tal modo il concetto di automatico senza ambiguità. Diamo la seguente Definizione.
Definizione di macchina di von Neumann (modello di calcolo pratico): una macchina di von Neumann è una terna

$$\mathbf{(N,}\ IS\mathbf{,}\ P\mathbf{)}$$

Dove

N = {0,1,2,…} è l'insieme dei numeri naturali e rappresenta l'alfabeto della macchina. Si noti che questa componente è fissa ovvero uguale per ogni macchina.

IS = {ZERO, INC, SOM, SOT, MOL, DIV, UGUALE, MINORE; SALCOND, ALT} (Instruction Set) è l'insieme di istruzioni generiche della macchina. Si noti che anche questa componente è fissa ovvero uguale per ogni macchina. Tale insieme di istruzioni generiche è definito come segue. Le prime otto istruzioni implementano le funzioni a valori in **N** così definite:

$$\text{ZERO: } \mathbf{N} \to \mathbf{N}$$
$$n \to \text{ZERO}(n) = 0,$$

$$\text{INC: } \mathbf{N} \to \mathbf{N}$$
$$n \to \text{INC}(n) = n+1,$$

$$\text{SOM: } \mathbf{N} \times \mathbf{N} \to \mathbf{N}$$
$$(n,m) \to \text{SOM}(n,m) = n + m,$$

$$\text{SOT: } \mathbf{N} \times \mathbf{N} \to \mathbf{N}$$
$$(n,m) \to \text{SOT}(n,m) = \begin{cases} n - m & \text{se } n \geq m, \\ 0 & \text{se } n < m \end{cases},$$

$$\text{MOL: } \mathbf{N} \times \mathbf{N} \to \mathbf{N}$$
$$(n,m) \to \text{MOL}(n,m) = n \times m,$$

$$\text{DIV: } \mathbf{N} \times \mathbf{N} \to \mathbf{N}$$
$$(n,m) \to \text{DIV}(n,m) = \lfloor n / m \rfloor,$$

dove $\lfloor x \rfloor$ indica la parte intera del numero reale *x*, ovvero il numero intero più grande minore od uguale ad *x*.

UGUALE: $\mathbf{N} \times \mathbf{N} \rightarrow \mathbf{N}$

$$(n,m) \rightarrow \text{UGUALE}\ (n,m) = \begin{cases} 1 & \text{se } n = m, \\ 0 & \text{se } n \neq m \end{cases},$$

MINORE: $\mathbf{N} \times \mathbf{N} \rightarrow \mathbf{N}$

$$(n,m) \rightarrow \text{MINORE}\ (n,m) = \begin{cases} 1 & \text{se } n < m, \\ 0 & \text{se } n \geq m \end{cases},$$

La nona istruzione, SALCOND, è un istruzione speciale, detta istruzione di salto condizionato. Essa è una funzione che ad un numero naturale ed un indice, detto indirizzo di *P*, associa l'azione di "saltare all'istruzione di *P"* specificata dall'indice. Questo, condizionatamente al fatto che il numero naturale sia diverso da zero o meno. Formalmente,

SALCOND: $\mathbf{N} \times J \rightarrow$ *Azione*

$$(n, j) \rightarrow \text{SALCOND}\ (n, j) = \begin{cases} \text{salta a } j & \text{se } n \neq 0, \\ \text{non salta a } j & \text{se } n = 0 \end{cases},$$

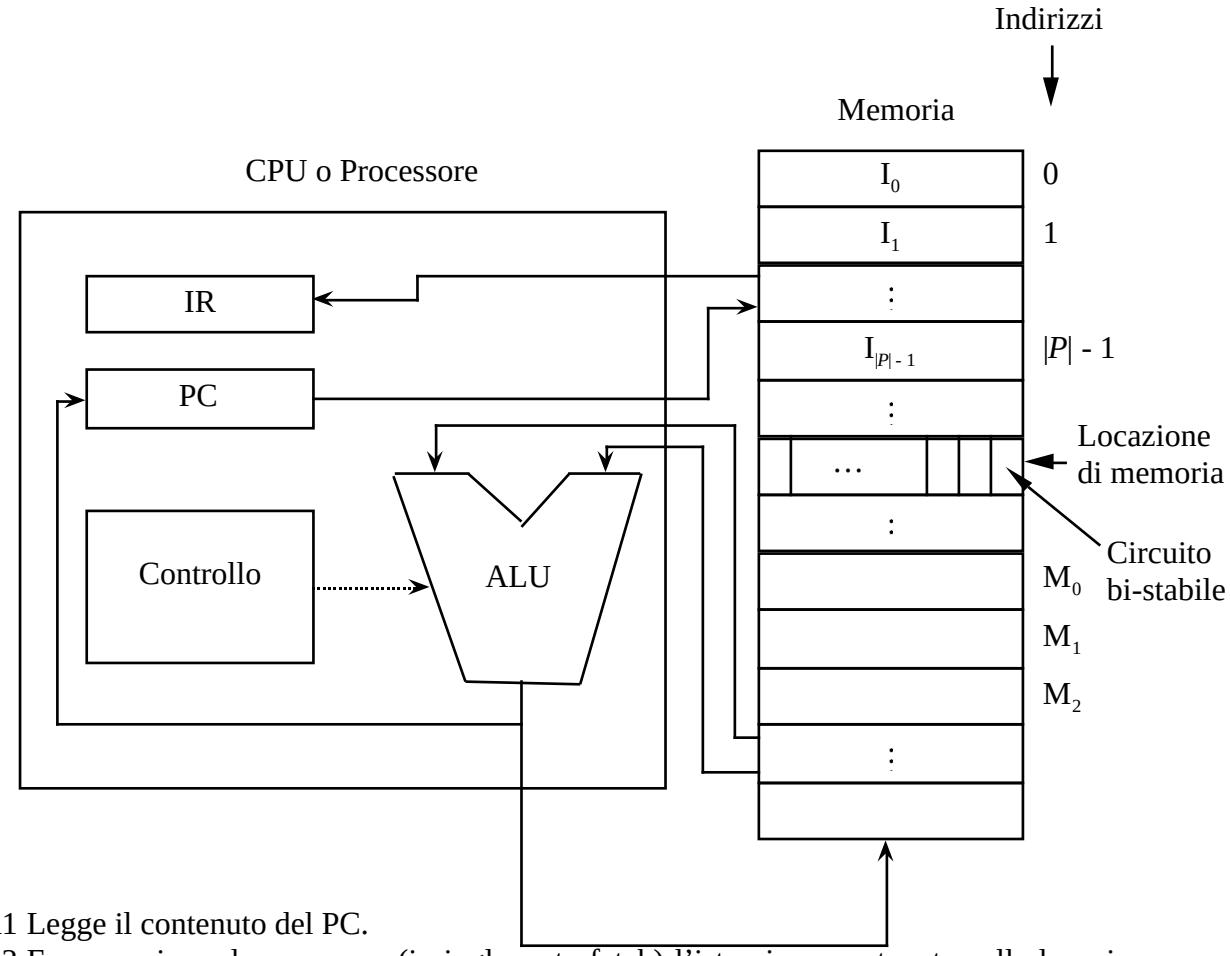
essendo $J = \{j \in \mathbf{N}: 0 \leq j < |P| - 1\}$ un insieme finito di indici.
La decima istruzione, ALT, è una costante uguale all'azione di fermare, terminare la computazione della macchina.

P $= \{I_0, I_1, I_2, \ldots, I_{|P|-1}\}$ una sequenza finita non vuota di istruzioni specifiche, detta *programma* della macchina. Per istruzione specifica intendiamo ogni istruzione (ovvero, elemento di *IS*) in cui si siano specificati particolari valori alle variabili indipendenti. L'insieme di tali valori, al variare delle istruzioni specifiche nel programma, è detto insieme dei dati del programma.

Si assume che la computazione della macchina avvenga eseguendo le istruzioni del programma, nell'ordine definito dal programma (prima si esegue la prima istruzione, poi la seconda, ecc), a meno di salti condizionati. Tutto ciò nel seguente modo. Si immagina che l'hardware della macchina sia costituito da una *memoria* e da un *processore*, come in Figura 3. La memoria è un sistema costituito da una sequenza sufficientemente grande di *locazioni di memoria*, ognuna contenente un numero naturale. Oggi che ogni computer ha un'architettura a 32 bits, ogni locazione di memoria è costituita da 32 circuiti bi-stabili (ovvero, lampadine) e può quindi contenere 32 bits = 4 bytes di dati. Ciò fa si che, in realtà, ogni locazione di memoria possa contenere solo tutti i numeri da 0 a 2^{32}-1. Ad ogni locazione di memoria è associato l'indice della locazione nella sequenza detto *indirizzo* della locazione. Nella memoria sono registrati il programma e i dati del programma. Il programma (o, per meglio dire, una codifica naturale del programma) è registrato nelle locazioni di memoria i cui indirizzi vanno da 0 a $|P| - 1$ in modo che l'*i*-esima istruzione del programma è registrata nella locazione di memoria il cui indirizzo è *i*. In un'altra parte della memoria sono registrati i dati necessari per l'esecuzione del programma, ovvero, i dati del programma. Il processore è un sistema costituito da quattro parti fondamentali: il *contatore di programma* (in inglese Program Counter o PC), il registro delle istruzioni (in inglese, Instruction Register o IR), l'*unità aritmetica e logica* (in inglese Arithmetic and Logic Unit o ALU) ed il *controllo*. Il contatore di programma è una locazione di memoria contenente l'indirizzo dell'istruzione da eseguire. Il registro delle istruzioni è una locazione di memoria contenente l'istruzione da eseguire. L'unità aritmetica e logica è un sistema che esegue l'istruzione indicata dal PC e registrata in IR. Il controllo è un sistema che, attraverso una sequenza di cambiamenti di stato,

"eccita e/o inibisce" opportunamente determinate parti del processore (tra cui l'ALU) facendo avvenire **l'esecuzione di un'istruzione** tramite l'esecuzione delle seguenti azioni.

Figura 3: hardware costituente la macchina di von Neumann.



A1 Legge il contenuto del PC.

A2 Fa pervenire nel processore (in inglese, to fetch) l'istruzione contenuta nella locazione di memoria programma il cui indirizzo e contenuto nel PC.

A3 Decodifica tale istruzione, ovvero

A3.1 "capisce" che istruzione è tra le dieci possibili, ovvero invia dei segnali all'ALU (ed eventuali altre parti del processore) in modo che questa esegua la funzione caratteristica della data istruzione, e

A3.2 acquisisce dalla memoria i dati specificati dall'istruzione necessari per eseguire l'istruzione. In un'istruzione (specifica) del programma, i dati sono specificati in maniera indiretta tramite gli indirizzi delle locazioni di memoria contenenti i dati dell'istruzione. Ad esempio, un istruzione del programma può essere SOM(M_2,M_0) in cui M_2 e M_0 sono i due indirizzi delle locazioni di memoria contenenti i due numeri (dati) da sommare. A ciò fa eccezione l'istruzione SALCOND per quanto riguarda il secondo argomento (operando) che viene dato in maniera diretta. Ad esempio un operazione specifica SALCOND può essere SALCOND(M_1,12302).

A4 Aspetta che l'ALU calcoli il risultato.

A5 Registra il risultato nella locazione di memoria specificata dall'operando più a sinistra dell'istruzione. Se, ad esempio, l'istruzione in esecuzione è SOM(M_2,M_0), il controllo registra il risultato nella locazione di memoria il cui indirizzo è M_2. Nel caso in cui, l'istruzione in esecuzione è SALCOND(*n*,*j*) ed *n* è diverso da zero, il controllo registra il numero *j* (che rappresenta l'indirizzo della prossima istruzione da eseguire) nel PC.

A6 Incrementa il PC a meno che l'istruzione in esecuzione non sia un SALCOND(*n*,*j*) con *n* diverso da zero o un ALT.

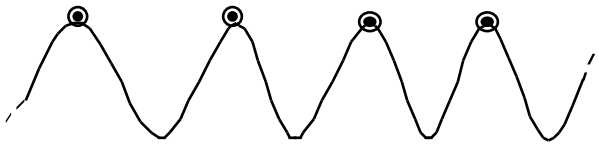
Il ciclo di esecuzione di un'istruzione si ripete fin tanto che non si incontra l'istruzione ALT, per cui la macchina si ferma. Nei computer moderni un chip contiene la CPU (ed eventualmente anche parte della memoria), mentre un altro chip (o più chips) contiene la memoria. Le comunicazioni tra la CPU e la memoria (rappresentate dalle frecce in Figura 3) avviene attraverso un canale di comunicazione composto da più fili paralleli detto *bus.*

Diamo ora un esempio di programma per la macchina di von Neumann che calcola se un numero è divisibile per 4 o meno. Si noti che un numero n è divisibile per 4 se, e solo se, il resto, r, della divisione tra n e 4 è 0. Si noti che tale resto è proprio dato da $r = n - 4 \cdot \lfloor n / 4 \rfloor$. Assumiamo che il numero n sia registrato nella locazione di memoria M_0. Il seguente programma scrive il risultato nella locazione di memoria M_1. In particolare, scrive 0 se il numero non è divisibile per 4 e scrive 1 se il numero è divisibile per 4.

0	ZERO(M_1)	
1	INC(M_1)	
2	INC(M_1)	scrive 4 in M_1.
3	INC(M_1)	
4	INC(M_1)	
5	ZERO(M_2)	copia il contenuto
6	SOM(M_2,M_0)	di M_0 (n) in M_2.
7	DIV(M_2,M_1)	
8	MOL(M_2,M_1)	calcola $r = n - 4 \cdot \lfloor n /4 \rfloor$ e
9	SOT(M_0,M_2)	mette il risultato in M_0.
10	SALCOND(M_0,14)	se r≠ 0 salta a 14.
11	ZERO(M_1)	
12	INC(M_1)	scrive 1 in M_1 se $r = 0$ e si ferma.
13	ALT	
14	ZERO(M_1)	scrive 0 in M_1 se $r \neq 0$ e si ferma.
15	ALT	

2.1.3 Il concetto di stato di un sistema e i MegaHz

L'esecuzione di un'istruzione avviene attraverso una sequenza di cambiamenti di stato del processore scandita da un segnale periodico (*clock*), diciamo, della seguente forma:



Tale segnale è trasportato da un filo per tutto il chip contenente il microprocessore. Ad ogni occorrenza del ⊙, ovvero ad ogni apice del segnale, il microprocessore cambia stato. Per stato del microprocessore si può intendere l'insieme degli stati fisici di ogni singolo filo elettrico costituente il sistema. È quindi chiaro che più è alta la frequenza del clock più il processore cambia stato velocemente e più il numero di istruzioni eseguite in un unità di tempo è grande. In altre parole, più è alta la frequenza del clock più il processore è veloce. **Un processore a 500 MHz è un sistema che cambia stato 500 milioni di volte in un secondo**.

Oggi i processori più comuni per Personal Computers (computers per tutti) sono dovuti a tre case produttrici: l'INTEL, l'AMD e l'IBM. I processori dell'INTEL sono il Pentium IV (che si può trovare a 1.4 GHz), i Pentium III e Pentium III Xeon (che si possono trovare fino a 1 GHz) ed il Celeron (che si può trovare fino a 700 MHz). I processori dell'AMD sono l'Athlon (che si può trovare fino a 1.1 GHz) ed il K6 (che si può trovare fino a 700 MHz). Tutti questi processori hanno lo stesso Instruction Set (ovvero, parlano la stessa lingua chiamata x86) costituito da un numero di

istruzioni del'ordine delle centinaia di istruzioni. Di una famiglia totalmente differente (in quanto hano un Instruction Set differente) sono invece i processori dei computer Macintosh, i quali sono costruiti dall'IBM. Essi sono il PowerPC G4 (si possono trovare fino a 500 MHz) ed il PowerPC G3 (si possono trovare fino a 500 MHz). Tutti i processori su menzionati hanno un architettura a 32 bits. È in proggetto per INTEL di produrre un processore con un'architettura a 64 bits chiamato Itanium.
In ogni modo, quando si compra un computer bisogna tener presente che l'informatica è un campo tecnologico con un evoluzione rapidissima e ciò che è all'avanguardia oggi sarà obsoleto fra un anno. Questa evoluzione rapida è espressa da una legge detta legge di Moore (uno dei quattro fondatori di INTEL) che può essere espressa così**: ogni 18 - 24 mesi le performances di un computer raddoppiano**. Fino ad ora questa legge non è stata smentita.

2.1.4 Le memorie e la loro gerarchia

Nei computer odierni ci sono vari tipi (livelli) di memorie; alcune più costose, più veloci, più piccole ed altre meno costose, più lente, più grandi. I tipi fondamentali di memoria oggi esistenti sono riportate nella Tabella 1 con le relative caratteristiche e performances. I tipi di memoria con performances migliori si dice che sono ad un livello più alto di quelle con performances peggiori.
Nella tabella, le memorie cache e RAM (Random Access Memory) sono fatte di transistors. L'hard disk è di solito costituito da un disco magnetico diviso in celle ognuna delle quali contiene 1 bit di informazione. In una cella è scritto 0 o 1 a seconda che il campo magnetico della cella si di un verso o di quello opposto. Della stessa tecnologia sono fatti essenzialmente i floppy disk ed i nastri magnetici. Mentre l'hard disk è fisso nel computer (infatti, prende anche il nome di *disco fisso*), il floppy disk ed il nastro magnetico sono removibili e possono essere trasportati a piacimento. A parte ciò, l'unica differenza tra il nastro magnetico e l'hard o floppy disk e che il nastro magnetico è un nastro e gli altri sono dischi e quindi l'accesso ai dati del primo deve essere fatto in maniera sequenziale (in media molto più lenta). La tecnologia dei Compact Disk (CD) e dei Digital Versatile Disk (DVD) è essenzialmente diversa: essi sono dischi ottici. Un disco ottico è un disco di plastica su cui è stato steso del materiale riflettente. Anche i dischi ottici sono divisi in celle ognuna delle quali contiene 1 bit di dati. In una cella si scrive 0 lasciando la cella intatta e si scrive 1 effettuando una scalfittura nella cella con un laser. In fase di lettura, un laser a bassa potenza illumina le varie celle. Quelle intatte riflettono la luce indicando che contengono 0, quelle scalfite non riflettono luce indicando che contengono 1. Data la loro natura, i dischi ottici sono Write Once Memory (WOM), ovvero memorie su cui si può scrivere una volta sola. La differenza sostanziale tra i CD ed i DVD sta nel fatto che in questi ultimi le dimensioni delle celle sono notevolmente più piccole (circa 1/7 di quelle dei CD) ed è possibile scrivere su due strati per faccia (ovvero sono memorie olografiche).
Nella Tabella 1, sono riportate le dimensioni, le velocità ed i costi di ogni tipo di memoria. La velocità è misurata in termini del tempo di accesso alla memoria. Per *tempo di accesso di una memoria* si intende la quantità di tempo che intercorre tra quando la CPU richiede un dato in una qualsiasi locazione della memoria a quando gli perviene il dato registrato nella locazione. Nella tabella sono riportati gli ordini di grandezza dei tempi di accesso delle varie memorie.
Per memoria *volatile* si intende un tipo di memoria (fatta di transistors) che svanisce quando si spegne il computer. La cache e la RAM sono memorie volatili. Le rimanenti memorie nella tabella sono non volatili o di massa per cui anche spegnendo il computer mantengono i dati in esse contenuti. Ciò fa si che la cache e la RAM siano usate per memorizzare i programmi ed i dati in esecuzione e le altre memorie siano usate per memorizzare i programmi ed i dati in modo permanente.
Nei computer moderni la memoria viene gestita con un meccanismo che prende il nome di *memoria virtuale.* Tale meccanismo serve a far sembrare all'utente (ed ai suoi programmi applicativi) di avere a disposizione più memoria ad un certo livello usando memoria di livello inferiore. La memoria virtuale funziona al seguente modo. Supponiamo ad esempio che la CPU abbia a

disposizione una RAM di 8 Mbytes, una cache di 512 Kbytes e debba eseguire (perché ordinato dall'utente) un programma applicativo delle dimensioni di 4 Mbytes già residente (memorizzato) in RAM. Per l'esecuzione, il programma viene diviso in 8 blocchi (dette pagine) di 512 Kbytes (le dimensioni della cache) per blocco, il primo blocco (quello contenete la prima istruzione del programma) viene copiato dalla RAM alla cache e l'esecuzione inizia con la CPU che "feccia" istruzioni direttamente dalla cache (e non dalla RAM che è più lenta) come descritto nella Sezione 2.1.2. Eventualmente, ad un certo punto, la CPU farà riferimento ad un'istruzione ad un'altra pagina (che non risiede nella cache ma nella RAM). Allora la CPU copierà questa nuova pagina nella cache al posto di quella vecchia (page swap) e riprenderà la computazione sulla nuova pagina. E così via. Anche se il programma non entra nella cache (perché 8 volte più grande) è come se risiedesse nella cache. Si noti che l'efficienza di questo metodo è basata su un **principio di località** empiricamente vero sostenente che **se I_j è la k-esima istruzione eseguita, è molto probabile che I_{j+1} sia la (k+1)-esima a dover essere eseguita**. Se questo principio non fosse vero, la CPU occuperebbe quasi tutto il suo tempo effettuando page swaps.

Tabella 1: gerarchia delle memorie.

Tipi di memorie		Dimensioni	Tempo di accesso	Costo
volatili	Cache	0 - 512/1024 Kbytes	1 - 10 nanosec.	alto
	RAM	4 - 256/512 Mbytes	100 - 1000 nanosec.	
di massa, non volatili	Hard Disk	2 - 32 Gbytes	1 - 100 millisec.	medio
	Digital Versatile Disk	4.7 – 18 Gbytes		
	Compact Disk	650 Mbytes		
	Floppy Disk	1,3 Mbytes		
	Nastro Magnetico	ordine di Tbytes	1 millisec. - 10 min.	basso

2.1.5 Le periferiche

In questa sezione descriveremo tutti gli apparati, oggidì più comuni, che permettono ad un computer di interagire con il mondo esterno e con l'utente. Questi apparati sono comunemente chiamati *periferiche* del computer e servono appunto per realizzare l'input e l'output (I/O) di un computer.

Lo schermo. Serve per visualizzare i responsi del computer in real time. È essenzialmente costituito da una griglia di lampadine (pixels) che possono assumere diversi colori. Gli schermi più comuni sono quelli da 15 pollici (con una griglia da 1024 × 768 pixels) e quelli da 17 pollici (con una griglia da 1280 × 1024 pixels). La loro *risoluzione* è di 72 × 72 dpi (dot per inch = pixel per inch). Ci sono gli schermi a tubo catodico (usualmente per computer fissi) e quelli a cristalli liquidi a matrice attiva o passiva (per i computer portatili). Gli schermi a matrice attiva sono meglio di quelli a matrice passiva.

La tastiera. Serve per digitare i caratteri alfanumerici e/o comporre i file di testo. È null'altro che un dispositivo che converte la digitazione di un certo carattere in una parola del codice ASCII (American Standardt Code for Information Interchange).

Il mouse. Serve per puntare e selezionare gli oggetti sullo schermo di un computer con un sistema operativo che abbia una GUI (Graphical User Interface). È stato introdotto dalla Macintosh nel 1982.

L'hard disk drive. Hardware che serve per scrivere e leggere i dati sull'hard disk.

Lettore CD/DVD. Hardware che serve per leggere i dati su un CD/DVD. Comunemente la sua velocità è espressa in *n*×, con *n* numero naturale. Più grande è *n* più è veloce il lettore. Oggi ci sono lettori CD la cui velocità di lettura è di 48×.
Il masterizzatore. Hardware che serve per scrivere i dati su un CD. Oggi ci sono masterizzatori la cui velocità di scrittura è di 6×.
Il floppy disk drive. Hardware che serve per scrivere e leggere i dati su di un floppy disk.
Il modem (modulator/demodulator). Serve a trasmettere dati (sequenze di bits) sulla linea telefonica. È un dispositivo indispensabile per connettersi ad internet da casa. Oggi ci sono modem la cui velocità di trasmissione va fino a 56 K bps (Kilo bits per secondo).
Lo scanner. Serve ad acquisire immagini (tipicamente su fogli di carta) per il computer. L'immagine viene "scannata", digitalizzata e messa nella RAM del computer pronta per essere manipolata ed eventualmente salvata.

2.1.6 I tipi di computer contemporanei.

Oggidì i tipi di computer che si trovano in commercio sono dal più piccolo, meno potente e più maneggevole, al più grande, più potente e meno maneggevole, i seguenti.
Palm-top computers. Sono computers "palmari" che possono stare sul palmo di una mano. Il loro costo è minore di 5 milioni di lire.
Lap-top computers. Sono computers che possono stare sulle ginocchia di una persona. Sono anche noti con il nome di computer portatili. Il loro costo va fino a 12 milioni di lire.
Personal computers. Sono i comuni PC che possono stare sulla scrivania di ognuno il cui costo va fino a 10 milioni di lire.
Workstation. Sono computer di elevata capacità computazionale il cui costo va dai 10 milioni di lire in su, i quali possono stare su una scrivania. Di solito usano il sistema operativo UNIX (vedi Sezione 2.2.1.5).
Mainframe. Sono computer di elevata capacità computazionale (possono avere anche più di un microprocessore) il cui costo va dai 15 milioni di lire in su. Questi occupano lo spazio di un armadio di 2 metri cubi e sono progettati per essere usati da più di un utente tramite più terminali sparsi tipicamente nel raggio di 1km. Di solito usano il sistema operativo UNIX.
Supercomputer. Sono computer di notevole capacità computazionale (sono per lo più macchine parallele con fino a 1024 microprocessori) il cui costo si agira intorno ai miliardi di lire. Questi occupano lo spazio di un armadio di 6 metri cubi e sono progettati per risolvere problemi nei grandi progetti quali la previsione del tempo, sequenziazione del genoma umano, mandare sonde su Marte, ecc. . Ci saranno un centinaio di queste macchine su tutta la terra.

2.2 Le componenti metafisiche di un computer: Il software

In questa sezione ci occuperemo di come è strutturato e come funziona il software di un computer, ovvero l'insieme di tutti i programmi (o dati in genere) contenuti in un computer che ne caratterizzano il comportamento con il mondo esterno rappresentato dall'*utente* (persone, macchine, altri computers). Uno scopo fondamentale dell'Informatica è appunto quello di studiare l'interazione tra l'utente ed il computer in modo da rendere tale interazione semplice ed efficiente per l'utente. Il software è composto da tre componenti: il *sistema operativo*, i *programmi applicativi* e i *dati* creati con programmi applicativi. Il sistema operativo è quel programma (o insieme di programmi) che si occupa dell'interazione tra l'utente ed il computer e verrà discusso nella Sezione 2.2.1. I programmi applicativi sono gli utensili che l'utente ha per risolvere i suoi vari problemi specifici (ovvero, per lavorare) e verranno discussi nella Sezione 2.2.2. Anche se fisicamente il software è composto dalle tre componenti su menzionate, da un punto di vista funzionale (cioè, della funzione che svolge nel binomio utente-computer) la terza componente si sottintende e si congloba con la seconda. Questo perché la funzione dei dati creati con i programmi applicativi non è altro che quella di rappresentare le domande (input) dell'utente e le risposte (output) ai suoi problemi sottoposti ai vari programmi applicativi. In un certo senso, i dati sono

l'informazione trasportata dai fili di un circuito elettrico e i programmi applicativi sono i circuiti elettrici (composti da fili e transistors) che la manipolano.

2.2.1 I sistemi operativi

Uno scopo fondamentale dell'Informatica è quello di studiare l'interazione tra l'utente ed il computer in modo da renderla semplice ed efficiente per l'utente. Il sistema operativo è quell'insieme di programmi che si occupa di rendere all'utente il computer semplice ed efficiente. Il computer (macchina che serve per risolvere problemi) è costituito da tre risorse: **l'hardware** (costituito dalla CPU, memorie, periferiche di I/O), **il sistema operativo** ed **i programmi applicativi** (programmi che servono per effettuare lavori specifici. Esempi di programmi applicativi sono: gli editori di testo come MS Word, fogli elettronici come MS Excel, video games, programmi scientifici, compilatori, programmi realizzati dall'utente stesso per risolvere alcuni suoi problemi molto specifici, ecc.). D'altro canto, **l'utente è un'entità che cerca di risolvere vari problemi** (ovvero, effettuare vari lavori) **usando i vari programmi applicativi esistenti o creati da lui stesso**.

2.2.1.1 Che cosa è e a che serve un sistema operativo

Nel binomio utente-computer, **il sistema operativo è un programma** (o insieme di programmi**) che agisce da intermediario tra l'utente (ed i suoi programmi applicativi) e l'hardware in modo da governare** (o almeno aiutare l'utente a governare) **le tre risorse costituenti il computer**. Lo scopo di un sistema operativo è quello di fornire un ambiente in cui l'utente possa eseguire i suoi programmi (ovvero, lavorare) in maniera semplice, **conveniente** (in inglese, user friendly) ed **efficiente** (ovvero, veloce). Tutto ciò, controllando e coordinando l'uso dell'hardware tra i vari programmi applicativi dell'utente (o degli utenti). La funzione di un sistema operativo e delle altre componenti del binomio utente-computer è schematizzata in Figura **4**. Quindi, un sistema operativo deve essere conveniente da usare per cui ad esempio:

a) deve mostrare in maniera chiara all'utente i dati contenuti nella memoria di massa;
b) deve curarsi delle cose marginali che non interessano all'utente quali: I/O con le varie periferiche (tramite i device drivers). Non deve essere l'utente ad istruire il computer su come scrivere un dato di output. Deve caricare i programmi dalla memoria di massa alla memoria volatile, ecc.;
c) deve curarsi che il passaggio di dati tra un programma e l'altro sia semplice;
d) deve curarsi che passare da un programma in esecuzione all'altro sia semplice;
e) deve controllare l'esecuzione di un programma e se questo abortisce deve dire perché;
f) deve prevenire l'uso improprio di una qualsiasi componente del sistema;

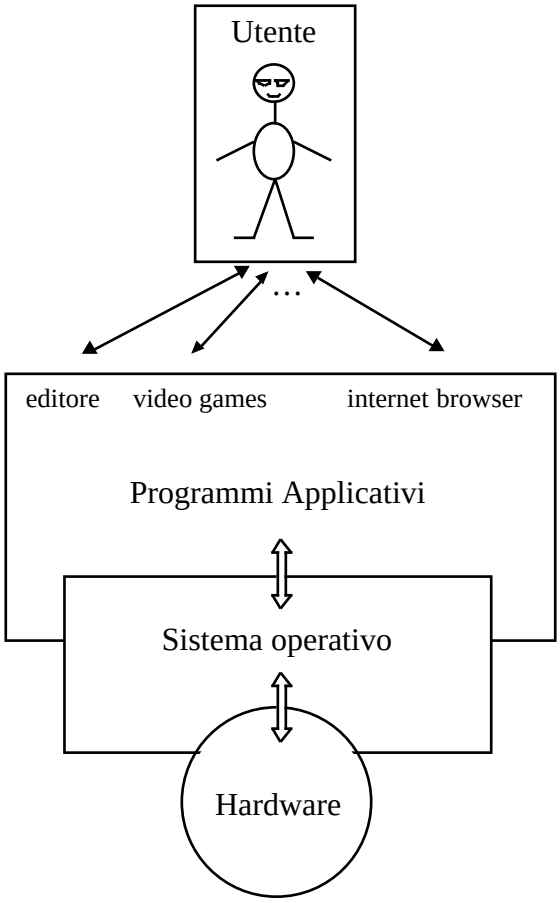
un sistema operativo deve essere anche efficiente per cui ad esempio:

a) deve allocare le varie risorse hardware (CPU, memoria, stampanti ecc.) in maniera efficiente ed equa tra i vari programmi applicativi. Deve fare in modo che le risorse siano sempre occupate a fare lavoro utile e che tutti i programmi siano serviti delle risorse in maniera equa;
b) deve badare alla sua sopravvivenza;

2.2.1.2 Come funziona un sistema operativo

Ora che abbiamo visto cosa sono e a cosa servono, vediamo come funzionano i sistemi operativi. Essi sono degli *interrupt based systems*. In parole povere, sono dei programmi sempre in esecuzione (e quindi sempre residenti in RAM) che ascoltano, e quando interrotti dall'utente o dai programmi dell'utente fanno quello per cui sono stati interrotti. Ciò fa si che l'ambiente che creano sia *interattivo*.

Figura 4: funzionalità delle varie componenti del binomio utente-computer.



2.2.1.3 Come sono organizzati i dati nei sistemi operativi: volumi, direttori, applicazioni e documenti

In un computer (macchina per manipolare dati tramite programmi) i dati sono contenuti in *files* e registrati su una memoria di massa. **Un file è una sequenza di 0 e 1 a cui è stato dato un nome**. Ci sono due tipi di files: i files contenenti i programmi applicativi che prendono il nome di *applicazioni* ed i files contenenti dati creati da applicazioni (o più propriamente da programmi applicativi) che prendono il nome di *documenti*. Logicamente, la disposizione dei files è organizzata come segue.

- I files sono contenuti in *cartelle* (o *directories*) oppure in *volumi*.
- Le cartelle sono contenute in altre cartelle oppure in volumi.
- I volumi non sono contenuti in nessuna struttura logica e di solito rappresentano una unità di memoria di massa fisica (hard disk, floppy disk, CD, DVD, ecc.).

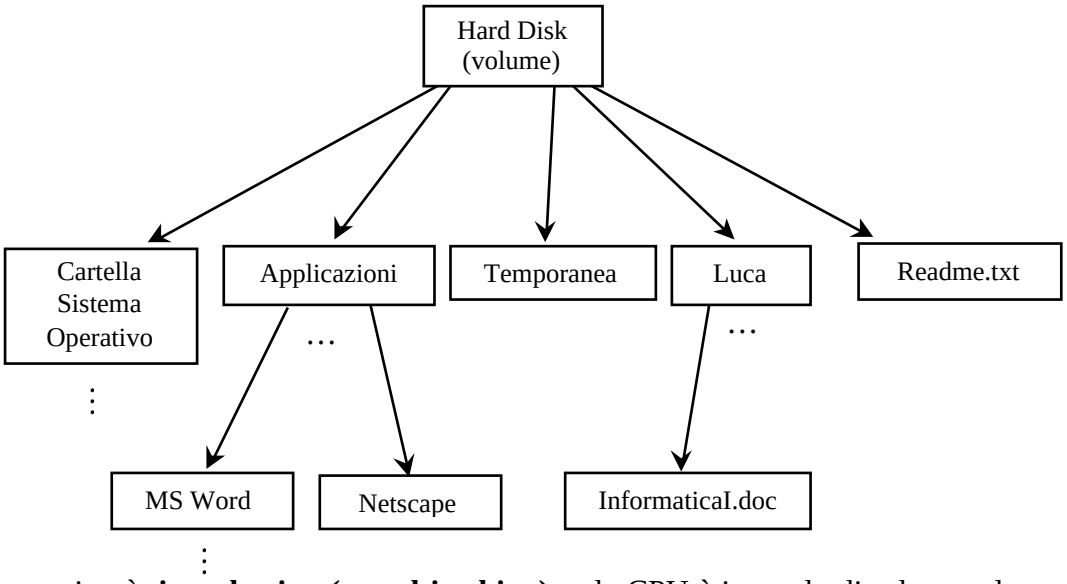
Tutto in modo da essere rappresentato da una struttura ad albero quale quella rappresentata in Figura 5. Nella figura, Hard Disk è un volume ed è quindi la radice dell'albero. Cartella Sistema Operativo, Applicazioni, Luca, MS Word, sono cartelle. Temporanea è una cartella vuota. Netscape è un'applicazione. Readme.txt e InformaticaI.doc sono documenti.

2.2.1.4 Le caratteristiche principali di un sistema operativo

In questa sezione vogliamo enucleare alcune caratteristiche fondamentali dei sistemi operativi.
Un sistema operativo è **multiprogramming** se è in grado di avere più programmi nel processo di essere eseguiti ad ogni istante. In un sistema multiprogramming la CPU salta da un lavoro ad un altro a convenienza (per esempio, nei computer antichi, il sistema saltava automaticamente da un

lavoro ad un altro se il primo doveva aspettare che la stampante avesse finito il suo lavoro per andare avanti. Oggi si può saltare da un lavoro all'altro con un click del mouse deciso dall'utente. Ciò è permesso dal fatto che i sistemi sono interattivi).

Figura 5: esempio della struttura logica di un volume.



Un sistema operativo è **time sharing (o multitasking)** se la CPU è in grado di saltare velocemente da un programma in esecuzione ad un altro e poi ad un altro ancora in modo che tutti i programmi in esecuzione siano effettivamente eseguiti e portati a termine. In un sistema time sharing questo saltare da un programma ad un altro (ovvero cambiare programma eseguito dalla CPU) è fatto in maniera così veloce da creare l'illusione che vari lavori si stiano svolgendo simultaneamente, in parallelo. Notiamo che è il time sharing la caratteristica fondamentale che permette l'interattività degli odierni sistemi operativi.
Un sistema operativo è **multi-user** se è in grado di far sembrare il computer (uno solo) come tanti personal computers: uno per ogni utente (più di uno). Gli utenti comunicano con il computer tramite più terminali. Un mainframe è un tipico computer progettato per avere un sistema operativo multi-user.

2.2.1.5 I principali sistemi operativi

In questa sezione passeremo in breve rassegna i principali sistemi operativi oggi esistenti.
Unix. Il più vecchio. Sviluppato nel mondo accademico americano per essere usato da mainframes e super computers. È un sistema completamente multiprogramming, time-sharing e multi-user. Non è un sistema GUI.
Linux. È essenzialmente il sistema Unix per PC. Sviluppato nell'ultimo decennio stà diventando molto popolare ultimamente che il gap di potenzialità computazionale tra PC e mainframes si è quasi annullato.
MS-DOS. Il primo sistema operativo per PC. È simile al sistema Unix senza essere multiprogramming, time-sharing e multi-user.
Macintosh. Il primo sistema operativo con incorporato una Graphical User Interface (GUI) per PC. È multiprogramming e parzialmente time-sharing. La Macintosh ha inventato le finestre e l'uso del mouse, ed il concetto di GUI.
Windows. GUI della Microsoft per MS-DOS. È molto simile al sistema Macintosh.
X11. GUI per Unix.
Solaris. Nuovo GUI della SUN per Unix.
KDE. GUI per Linux.

2.2.2 Le principali applicazioni

Le principali applicazioni di un computer sono:
Gli editori di testo. Servono per creare documenti di testo che di solito vengono stampati.
Gli editori di grafica. Servono per creare disegni. Ci sono due tipi di editori di grafica: gli editori di grafica bitmap e vettoriale.
I fogli elettronici. Servono per organizzare, archiviare e manipolare dati numerici sperimentali. Sono essenzialmente delle griglie fatte di celle ed ogni cella può contenere un dato o una formula contenete come variabili altre celle.
I compilatori e gli interpreti. Servono per tradurre in linguaggio macchina programmi scritti in un altro linguaggio di programmazione.

3 La rete “Internet”

Vedi guida *Internet & Computing Italian FAQ* di Vittorio Bertola. La guida è disponibile su Internet presso il sito http://bertola.eu.org/icfaq/.