Un algoritmo è una risposta finita per un infinito numero di domande
Esempi: somma di due numeri (decimale o binario), determinazione del massimo comune divisore (algoritmo di Euclide)
Programmazione: serve per eseguire algoritmi, ci sono di diverse tipologie (imperativo, logico, funzionale, object-oriented…
Uno degli utilizzi nelle biotecnologie è quello nella genomica e nella medicina moderna

Storia dell'informatica

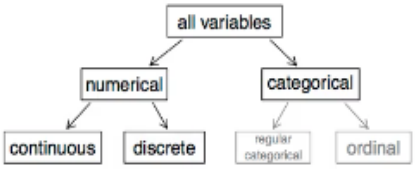
Pascal (prime forme di computazione – primo semplice calcolatore Pascalina)
Ada Byron su idee gia usate da Jaquard
Hollerith - per il censimento negli stati uniti (poi ha fondato IBM)
Von Neumann -
Turing – lavoro per il governo britannico, creò una macchina per decifrare i messaggi segreti tedeschi (macchina di turing – modello elementare di computazione)

Cosa sono i dati?

I dati sono valori di qualitative o quantitative variabili, che appartengono a una collezione di oggetti (o elementi) detta popolazione. Quando parliamo di analisi dei dati abbiamo delle entità che vogliamo studiare.

Variabile: è una caratteristica interessante degli elementi di una popolazione (attributi delle unita statistiche a cui siamo interessati
Esempi di variabile: risposta di un paziente a un trattamento medico, durata di una visita a un sito web, vita media di un automobile…

Le varabili possono essere:
→ qualitative (categoriale): ha valori che non sono numerici (la cittadinanza di un individui, il sesso di un individuo, il trattamento somministrato a un paziente…)
  - sconnessa: non hanno un ordinamento intrinseco (sesso di una persona – non ordinabile)
  - ordinale: l'ordinamento può essere stabilito (freddo o caldo – caldo -> medio -> freddo)
→ quantitative: variabili che posso misurare e rappresentare mendiate unita nuemrihe (altezza di una persona, gli esami superati…)
  - discreto (nuemro di esami superati nel corso universitario)
  - continuo (la temperatura, la pressione sanguigna)



I dati devono essere messi in una forma processabile

Le variabili in questo caso sono:
→ colonne: country
→ cr_req: numero di richieste di rimozione di contenuti (quantitativa discreta)
→ cr_comply: percentuale di richieste soddisfatte (quantitativa continua)
→ ud_req: numero di richieste quantitativa discreta)
→ ud_comply: percentuale di richieste soddisfatte (quantitativa continua)
→ hemisphere: emisfero (categoriale sconnessa)
→ hdi: Human Development Index (categoriale ordinale)

country	cr_req	cr_comply	ud_req	ud_comply	…	hemisphere	hdi
Argentina	21	100	134	32	…	southern	very high
Australia	10	40	361	73	…	southern	very high
Belgium	<10	100	90	67	…	northern	very high
Brazil	224	67	703	82	…	southern	high
…	…	…	…	…	…	…	…
United States	92	63	5950	93	…	northern	very high

esempio
AlmaLaurea: effettua un'indagine sui neolaureati nelle università italiane. Una delle variabili oggetto dello studio è il tempo necessario a trovare un lavoro dopo aver conseguito la laurea. Per ciascun intervistato, AlmaLaurea registra un 1 se l'intervistato ha trovato lavoro entro sei mesi dalla laurea, un 2 se ha trovato lavoro entro un anno e un 3 se ha trovato lavoro dopo un anno.
Il tempo necessario a trovare lavoro è in tal caso una variabile categoriale.

Dati grezzi e dati processati

Dati grezzi

→ Nella stessa forma della fonte da cui provengono, senza alcuna modificazione
→ Molto spesso non usabili direttamente per le analisi
→ Processamento da forma grezza a forma analizzabile spesso non banale (“80%of the effort of analysis is in data cleaning”
→ Ogni modificazione dei dati dev'essere documentata

Dati processati

→ Forma che può essere direttamente usata per l'analisi
→ Conversioni, fusioni, eliminazioni, trasformazioni di dati
→ Standard di processamento dei dati (**pipeline**)
→ Ogni fase dell'analisi dev'essere documentata e riproducibile

Pipeline

Serve per sequenziare materiale genetico (tra i primi)
Fa una sequenza di operazioni che permettono di elaborare il materiale biologico (frammenti DNA – che vengono posti su una piastra), tramite la lettura del DNA vengono estrapolate delle immagini che rappresentano i vari nucleotidi, a sua volta questi dati vengono ulteriormente elaborate con degli algoritmi viene determinando iascuna posizione del nucleotide (output).

Esempio di dati processati (matrice dei dati)

→ Ogni colonna rappresenta una variabile
→ Ogni riga rappresenta un’osservazione
→ Ogni tabella contiene dati su una sola tipologia di osservazioni (di tipo omogeneo – no dati che non servono o non centrano tra loro)

I dati non bastano: i dati possono non contenere la risposta alla domanda che ci interessa. ??non tutte le tipologie di dati si prestano a tutti i tipi di analisi, ci vogliono anche i modelli, poiché la scienza deve tenere presente sia dati sia i metodi per trattali (ipotesi e modelli).

Tipologie di analisi dei dati

Processo d'indagine

1. Identificare una domanda per cui si vuole trovare risposta o un problema che si vuole risolvere
2. Raccogliere un'adeguata quantità di dati utili allo studio
3. Analizzare i dati
4. Formulare conclusioni (interpretazione dati)

Tipi di analisi

Approssimativamente, in ordine crescente di difficoltà:

1. Descrittiva
2. Esplorativa
3. Inferenziale
4. Predittiva
5. Causale
6. Meccanicistica

Ciascun tipo di analisi risponde a domande diverse.

→ Bisogna innanzitutto chiarire qual è l'obiettivo dell'analisi.

→ Non è detto che i dati a disposizione consentano di effettuare un certo tipo di analisi (non è detto che i dati possano essere analizzati con tutte queste tipologie di analisi)

Analisi descrittiva

Obiettivo: descrivere un insieme di dati

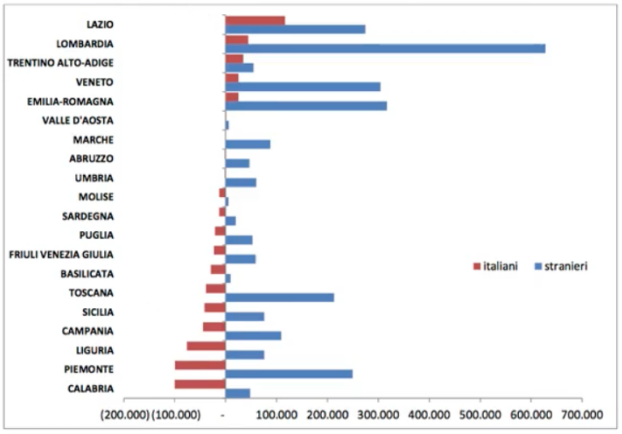
→ Il primo tipo di analisi da effettuare

→ Esempio tipico: censimento di una popolazione, efficacia di un farmaco

→ Descrizione e interpretazione dei dati sono due fasi distinte

→ La generalizzazione di un'analisi descrittiva (ad esempio, ad altri insiemi di dati) di solito non è possibile, non si può passare da conclusioni che riguardano la popolazione e generalizzare questi risultati al di fuori dei dati che ho considerato

Figura 1 - Variazioni assolute 2001-2011 della popolazione residente per cittadinanza e regione



Analisi esplorativa

Obiettivo: cercare nuove potenziali relazioni tra i dati

→ Su un'intera popolazione, ma più frequentemente su un campione

→ Utile come guida per studi più approfonditi

→ Tipicamente, insufficiente per generalizzazioni, inferenze o predizioni

→ La correlazione non implica la causalità (se io stabilisco una correlazione tra due variabili il cambiamento di una delle due variabili non implica che il cambiamo dell'altra)

Esempio: aree del cervello che si attivano durante un freestyle, correlazione tra la cioccolata e i premi nobel

Analisi inferenziale

Obiettivo: usare un campione per dedurre qualche proprietà dell'intera popolazione

→ Obiettivo tipico dei modelli statistici

→ Stima di una quantità d'interesse *e* misura dell'incertezza della stima

→ L'analisi inferenziale dipende fortemente sia dalle caratteristiche della popolazione sia da quelle del campione

Esempio: inferire da un campione la relazione tra inquinamento atmosferico e speranza di vita, voto di maturita in una classe

Analisi predittiva

Obiettivo: usare i dati su qualche oggetto per predire valori di altri oggetti

→ L'accuratezza di una predizione dipende fortemente dalle variabili considerate
→ Modelli predittivi, anche se semplici, possono essere efficaci, ma è essenziale comprenderne le ipotesi e le limitazioni
→ Fare predizioni è in generale estremamente difficile
→ Anche se *X* predice *Y* non è detto che *X sia la causa* di *Y*
→ "The best way to predict the future is to invent it"...non si applica all'analisi dei dati

Analisi causale

Obiettivo: determinare che cosa accade a una variabile (*risposta*) quando un'altra variable (*esplicativa*) cambia (causa effetto)

→ Di solito, richiede uno *studio randomizzato*
1) Un insieme di individui è diviso in due gruppi in modo casuale
2) Solo al primo gruppo è somministrato il trattamento
3) Si misurano le differenze tra i due gruppi
→ Una relazione di causalità è tipicamente identificata come effetto medio, ma non è detto che si applichi al singolo individuo

Un trattamento dimostratosi efficace, in seguito a opportuni indagini, per una particolare popolazione, può avere un effetto avverso su un particolare paziente

→ Analisi estremamente difficile da compiere, richiede una comprensione profonda dell'oggetto di studio

Esempio: trattamento su infusioni di feci da un donatore, mediante studio randomizzato

Analisi meccanicistica

Obiettivo: comprendere e descrivere precisamente la dinamica di un processo

→ Identificare i meccanismi con cui alcune variabili influenzano le altre
→ Modelli basati su equazioni differenziali (e.g., fisica classica)
→ I modelli sono spesso parametrici e la determinazione dei parametri può essere molto difficile
→ I modelli possono essere computazionalmente complessi
→ Esiste ancora una componente di casualità nei dati dovuta agli errori di misurazione

Esempio: non vengono fatte in medicina, ma vengono fatte in biologie dei sistemi

Raccolta dei dati

Principali modalità

→ Censimento (analisi descrittiva)
→ Studio osservazionale (analisi inferenziale, *non* causale)
→ Campioni convenienti (possono essere sbilanciati), se ho un nuemro di elementi limitato da analizzare (non è detto che l'analisi può essere generalizzata)
→ Studi sperimentali randomizzati (*randomized trial*) (analisi causale)

Altre modalità

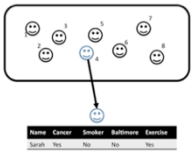
→ Studi di predizione (analisi predittiva)
→ Studi prospettivi (o di coorte)
- trasversali(*cross sectional*)(analisi inferenziale)
- longitudinali (*longitudinal*) (analisi inferenziali e predittive)

→ Studi retrospettivi o caso-controllo (analisi inferenziale, *non* causale)

Una popolazione

Immaginiamo una popolazione di 8 individui, di cui vogliamo studiare qualche predeterminato carattere

Campionamento

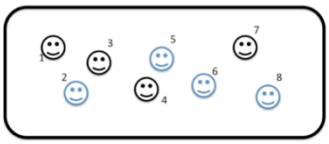


- Si può procedere scegliendo un individuo e misurando le proprietà d'interesse
- Tale procedimento può essere eseguito in vari modi (campionamento semplice, stratificato, clusterizzato)

Censimento

- Tutti gli individui della popolazione sono selezionati e misurati
- Nessun problema inferenziale

Studio osservazionale



- Campionamento casuale
- Inferenza dal campione alla popolazione

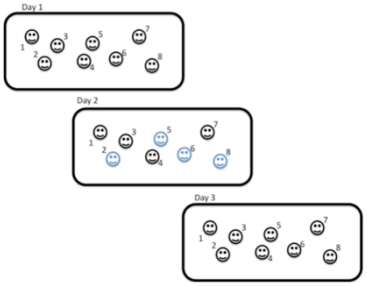
Campionamento di convenienza

Studio di predizione: test set

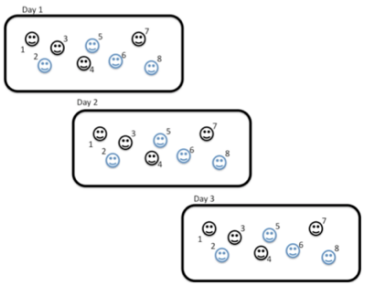
- Il test set è usato per valutare il modello di predizione

Studio randomizzato

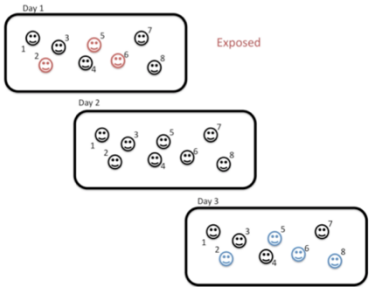
Studio trasversale (cross-sectional)



Studio longitudinale



Studio retrospettivo



Memorizzazione dati

La nozione fondamentale nel contesto dei dispostivi digitali è il bit (binary digit – cifra binaria)
Possiamo rappresentare i bit in vari modi: numeri (0, 1), cifre in base 2, simboli, acceso/spento, si / no – vero / falso

Con sequenze di bit possiamo rappresentare qualunque tipo di informazione, i calcolatori manipolano qualunque sequenza di bit.

I bit possono essere visti anche come la quantità di informazione contenuta in un carattere (entropia informazionale): S insieme qualsiasi.

Quante domande devo fare per trovare x $\in$ S? H (S) = log I S I

In un insieme di due elementi (0, 1), l'entropia vale 1 (log 2 =1), l'entropia di un insieme fatto di due elementi è un bit, un bit di informazione ci consente di codificare due valori rappresentabili.
Se considero un insieme di 4 elementi, questo insieme avrà un entropia informazionale pari a 2 (log 4 =2), quindi 2 bit di entropia informazionale mi consentono di rappresentare 4 valori distintiti (elementi).

Se per rappresentare un' informazione un certo numero di bit (n), con quel numero di bit posso rappresentare 2^2 valori diversi.

Se voglio memorizzare una sequenza di DNA nel computer (A, T, C, G) mi bastano due bit. es. 00 adenina, 01 citosina, 10 guanina, 11 timina

Quanta informazione contante un elemento in S ?

Operazioni Booleane

Le operazioni con i bit vengono fatte mediante due valori logici (Vero 1, Falso 0)

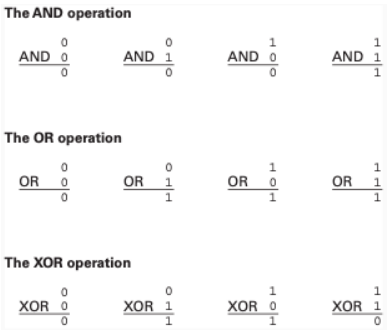
Congiunzione – and (binaria 2 argomenti)

Disgiunzione – or (binaria)

Negazione – not (unaria)

Disgiunzione esclusiva – xor

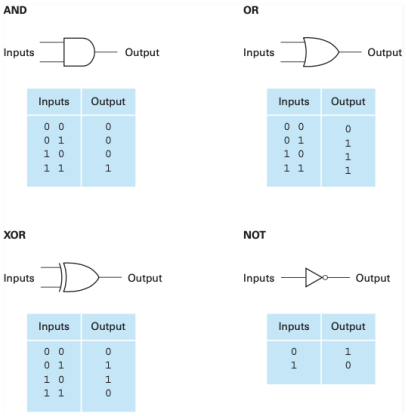
Implicazione – if-then



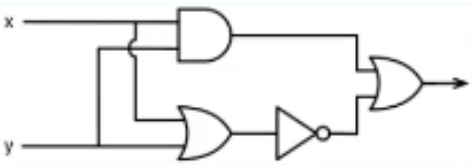
Porte (logiche) – gates

Elementi fondamentali costituitivi di un qualunque dispositivo digitale.
Realizzazioni in hardware (fisica) degli operatori del calcolo proposizionale.

A sx (imput) ci sono gli ingressi della porta logica, ossia i segnali (1 ,0) quando la porta logica riceve due segnali in entrata ne produce uno in uscita (output) secondo la regola del operazione logica.
Le porte logiche possono essere collegate tra loro creando dei circuiti più complessi cioè permettendo di effettuare dei calcoli più complessi.



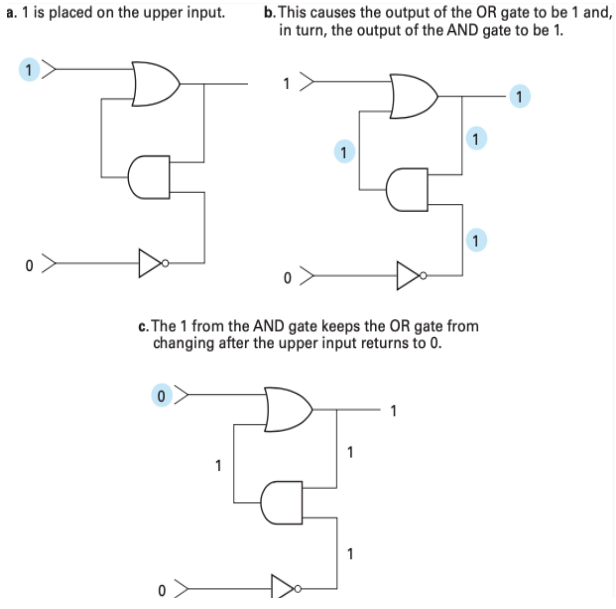
Esempio
Se l'imput Y vale 1 e l'output del circuito è 0 quanto vale x?



Y entra sia nel gate or sia nel gate and, l'imput y (1) nel gate and è 1, se x è 1 l'output sarà 1 se x è 0 l'imput del gate sara (0 e 1) quindi l'output sara 0. Quindi è necessariamente 0, poiche se x fosse 1 l'output del circuito sarebbe 1.

Flip – flop

È un circuito fondamentale che viene utilizzato nelle memorie RAM (volatili) dei calcolatori.
Ha la caratteristica di mantenere costante l'outpt del circuito, fintanto passa corrente, questo output può essere modificato con un impulso (cioe passaggio per un breve periodo da 0 a 1) in uno degli input del circuito. (L'output di un gate influenza l'imput del gate stesso)



Partiamo che entrambi gli input sono 0 e anche l'output è 0.
Se l' input iniziale diventa 1 per un breve periodo, il gate (or) produce in output 1, ma l'output del gate or diventa l'input del gate and .
Se dopo un po di tempo l'input inziale ritorna 0, comunque l'output del circuito continua ad essere 1 perche dal canale imput il gate or riceve 1.

Quindi: se i due input sono 0 e l'input sopra passa a 1 per un breve periodo e poi torna a 0, lo stato (l'output) del flip flop cambia da 0 a 1.

→ Combinando porte ottengo circuiti che hanno comportamenti complessi (memoria).
→ Una volta costruito il circuito non mi interesso più ai dettagli: astrazione.
→ Tanti circuiti tanta potenza (in poco spazio: Very Large Scale Integration—VLSI—, computers on a chip)
Il flip flop è un circuito in grado di mantenere uno stato, di mantenere un valore persistente finche nel circuito passa corrente.

Per rappresentare una grande quantità di informazioni possiamo farlo codificadno utilizzando delle stringhe ossia delle sequenze di bit.
Se io volessi fare delle operazioni aritmetiche utilzzando un circuito digitale dovrei trovare un modo per codificare i numeri.

Bit pattern	Hexadecimal representation
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Un esempio di codifica dei numeri è quella esadecimale (base 16) dove qualsiasi cifra viene codificata da una sequenza di bit.
esempio
5 + 8 = 13 0101 + 1000 = 1101

7 + 13 = 20 0111 + 1101= 10100

Ram (random access memory)

Hanno la caratteristica di essere volatili ossia l'informazione in queste memorie non è persistente in quanto le informazioni permangono finche il computer è alimentato da corrente.

main memory: hw per memorizzare tutti i bit che mi servono durante il calcolo (tanti flip-flop)
→ cella di memoria: collezioni di bit (normalmente 8) che vengono manipolati insieme, sequenze di byte (8 bit)

contenuto di una cella di memoria (bit piu significativo a sx, bit meno significativo a dx)

→ indirizzo (di una cella): posizione della cella nella lista delle celle. Posso accede ad una cella qualunque specificandone l'indirizzo

Queste memorie sono accessibili in maniera rapida (memorie veloci), il tempo necessario per scrivere o leggere una cella non dipende dalla posizione della cella.

Esercizio
Supponiamo di voler scambiare il valore memorizzato in due celle di memoria (cella 2 e 3). Come faccio?
Devo usare una terza cella di memoria per poter trasferire momentaneamente una delle due celle, poi prendo il contenuto della cella 2 e lo sposto nella cella 3 (temporaneamente spostata nella 4) e successivamente il contenuto della 4 lo posto nella 2).
!! non posso muovere contemporamente due celle e non posso mettere due celle assieme altrimenti verrebbero sovrascritte.

i date (le celle sono ordinati):
→ posso parlare non solo della cella che si trova ad un dato indirizzo, ma anche della cella che viene dopo/ prima
→ posso memorizzare stringhe lunghe usando celle consecutive

1 bit: 2 informazioni
2 bit: 4 informazioni
3 bit: 8 informazioni
...
8 bit: 256 informazioni ⇒ un byte B.

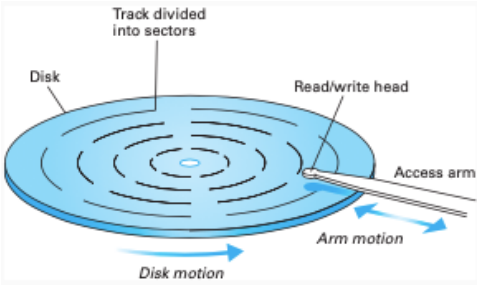
1 bit: unità fondamentale
8 bit: 1 **byte**
10^3 byte: 1 **kilobyte** (**kB**)
10^6 byte: 1 **megabyte** (**MB**)
10^9 byte: 1 **gigabyte** (**GB**)
10^{12} byte: 1 **terabyte** (**TB**)
10^{15} byte: 1 **petabyte** (**PB**)

Kilo byte: $2^{10} = 1024$ B
Mega byte: $2^{10} = 1024$ K
Giga byte: $2^{10} = 1024$ M
Tera byte: $2^{10} = 1024$ G
Peta byte: $2^{10} = 1024$ T

Memoria secondaria

→ memorie di massa: memoria non volatile, l'informazione persisite anche se il dispositivo non è alimentato
→ capacita tipiche ordine dei Tb o decine di Tb (costo inferiore rispetto alla memoria princiaple
→ memorie lente. Spesso richiedono movimenti di parti meccaniche
→ esempi: nasti magnetici, CD, DVD, hard-disc, SSD, memorie flash

funzionamento del disco magnetico: la testina scorre sopra il piatto in continua rotazione leggendo l'informazione scritta sulla superficie del piatto. Sono organizzati in uno o più piatti in rapida rotazione, ciascun piatto è diviso in tracce suddivise ulteriormente in settori dove viene memorizzare una quantità fissa di bit (più o meno 4 kb)
L'unita di trasferimento dati da e verso un dispositivo di memoria è il blocco. A differenza della memoria principale, non è possibile indirizzare unita piu piccole del blocco, in particolare, non è possbile indirizzare un singolo byte.



L'accesso comporta dei ritardi, per accedere in lettura o in scrittura a un settore:
1. ricerca: la testina deve posizionarsi lungo una traccia (seek)
2. latenza: si attende che il settore passi sotto la testina (latency)
3. tasso di trasferimento: i dati sono letti/scritti a blocchi a un dato tasso (transfer rate)

tempo di lettura: ricerca + latenza + tasso di trasferimento

Esempio di dati del costruttore:
capacità di un settore B=512 byte
numero medio di settori/traccia S=792
velocita di rotazione r = 7200 rpm
seek time medio s= 5 ms

capacità di una traccia T = S x B = 405504 byte
tasso di trasferimento dati t = T x r = 405504 x 7200/60 = 46 MB/s

Il sistema decimale

Nella notazione10 posizionale decimale, il valore di una cifra numerica è un multiplo di una potenza di 10 che dipendere dalla posizione della cifra all'interno del numero. Ad esempio

$167_{10} = 10^2 \times 1 + 10^1 \times 6 + 10^0 \times 7$

N_b denota il numero *n* espresso in base *b*

In generale, dato un numero composto dalle cifre $d_{n-1}\, d_{n-2}\ldots d_0$ in base 10 (dove d_{n-1} è la cifra più significativa e d_0 quella meno significativa), il valore corrispondete è dato da:

$(d_{n-1}\, d_{n-2}\ldots d_0)_{10} = 10^{n-1} \times d_{n-1} + 10^{n-2} \times d_{n-2} + \ldots + 10^0 \times d_0$

Da base 10 a base 2:

1. dividere il numero in input per 2 e tenere traccia del quoziente e del resto (che sarà 0 o 1)
2. dividere il quoziente precedentemente ottenuto per 2 e tenere traccia di quoziente e resto
3. ripetere il passo 2 finche il quoziente non diventa 0
4. concatenare i resti ottenuti dell'ultimo al primo

11:2=5 (resto 1) 5:2= 2 (resto 1) 2.2 =1 (resto 0) 1:2 = 0 (resto 1) $11_{10} = 1011_2$

$101_2 = 2^2 \times 1 + 2^1 \times 0 + 2^0 \times 1 = 5_{10}$

Da base 2 a base 10:

Calcolo successive potenze di 2 con esponenti diversi
La cifra meno significativa è quella a dx corrisponde a una potenza di 2 elevata alla 0 e poi si provece verso sx incrementando l'elevazione

$(b_{n-1}b_{n-2}\ldots b_0)_2 = 2^{n-1} \times b_{n-1} + 2^{n-2} \times b_{n-2} + \ldots + 2^0 \times b_0$

$10110_2 = 22_{10}$ $2^0 \times 0 + 2^1 \times 1 + 2^2 \times 1 + 2^3 \times 0 + 2^4 \times 1 = 2 + 4 + 16 = 22$
$22_{10} = 10110_2$ 22:2=11 (resto 0) 11:2=5 (resto 1) 5:2=2 (resto 1) 2:2=1(resto 0) 1:2=1 (resto 1)

Da decimale a esadecimale

0=0 … 9=9 10=A 11=B 12=C 13=D 14=E 15=F
Viene usata per rappresentare dei numeri binari 16 combinazioni possiamo rappresentare con 4 bit.
Nella conversione da esadecimale a binario ogni cifra decimale viene espansa con una sequenza di quattro bit e viceversa da binario a esadecimale si raggruppano le cifre binarie a gruppi di 4 partendo dalla cifra meno significativa (ossia quella a dx) e si sostituiuscono con la corrispondete esadecimale.
$0_{10} = 0_{16} = 0000_2$ $1_{10} = 1_{16} = 0001_2$ $9_{10} = 9_{16} = 0101_2$ $10_{10} = A_{16} = 1010_2$ $15_{10} = F_{16} = 1111_2$

$10110_1 = 0001\ 0110_2 = 16_{16}$
$F3_{16} = 1111\ 0011_2 = 243_{10}(16^1 \times 15 + 16^0 \times 3)$

Cos'`e un file?

Formalmente si definisce come una: unità concettuale di memoria di massa
Definizioni (usiamo l'inglese)

- physical record blocco minimo trasferibile dalla tecnologia
- utilizzata per la memorizzazione dei dati
- logical record blocco naturalmente definibile in funzione del tipo di dati memorizzati
- field sotto-unit`a dei record logici
- chiave campo identificativo (ci ritorneremo parlando di basi di
- dati)
- buffer (in questo caso) regione della RAM usata per contenere adeguate quantit`a di record fisici e per indirizzare record logici

Rappresentazione dei dati: testo

Caratteri: codice ASCII (American Standard Code for Information Interchange): in questa codifica è possbile rappresnetare 128 caratteri diversi con 7/8 bit, codifico lettere maiuscole, minuscole e digit da 0 a 9 ....
Non `e l'unico codice ma l'idea `e sempre la stessa.
International Organization for Standardization (ISO)

Un'altra codifica del testo ampiamente usata è UTF‑8, che usa un numero variabile di byte (da uno a quattro) per ciascun simbolo

Esercizio
Che differenza c'è tra la rappresentazione di 124924596 come testo o come numero?
$77232B4_{16}$
Ciscuna cifra numerica viene trattata come un carattere che viene codificata

Text files

- Text editors – utili per apportare modifiche ai ddati che vogliamo manipolare
- Word processors

Rappresentazione dati numerici

- Numeri naturali: (semplice) rappresentazione binaria.
- Numeri interi: uso il primo bit per il segno. C'e sempre un limite massimo di bit utilizzabili)
  Numeri reali (non possono essere rappresentati in un calcolatore)
    - virgola fissa: un intero (parte intera) ed un razionale (parte decimale);
    - virgola mobile: mantissa ed esponente.

Ogni numero reale puo essere rappresentato (mantissa moltiplicata per 10 ad un esponente)
Segno (+ o -) x mantissa (numero reale compreso tra 0 e 10) x 10^x (x intero)

Rappresentazione di immagini e suoni

Pixel (Picture element)

bit map: un pixel un(o o piu`) bit(s)

- ok per printer e schermi,
- un pixel pu`o essere 0/1 o un valore numerico (scala di grigi,
- RGB, ...),
- problematico se dobbiamo riscalare l'immagine;

rappresentazione vettoriale: l'immagine `e rappresentata come collezione di oggetti geometrici ed il dispositivo "decide" come rappresentarla (deve essere in grado di farlo ⇒ CPU)

- TrueType (Microsoft and Apple per i caratteri)

PostScript (Adobe Systems per i caratteri e non solo)

Suono

ampiezze d'onda (MP3) il suono è memorizzato attravero la serie di campionamneti successivi fatti a frequenze molto elevate
MIDI (Musical Instrument Digital Interface) viene codificata la nota per la durata

Esercizio
Vogliamo digitalizzare un suono a 44100 hz/16 bit della durata di 1s. quanti KB sono necessari
16* 44100 / 8 /10^3=88,2 KB
Per convertire devo dividere per 8 perche ogni byte e comporto da 8 bit e poi devo ulteriormente per 10 ^3 perchèper ogni KB corrisponde a 10 ^3 bit.
Per un minuto:
88,2* 60 = 5929 KB= 5 MB

Tutte le informazioni sono codificate come bit, le immagini possiamo codificare ciasun pixel con una sequenza di bit per esempio se abbiamo delle immagini codificate come matrici di pixel ma ci sono altri modi per codificare le

immagini in cui l’immagine viene costruita specificando con un opportuna codifica la forma di determinati oggetti (SVG)

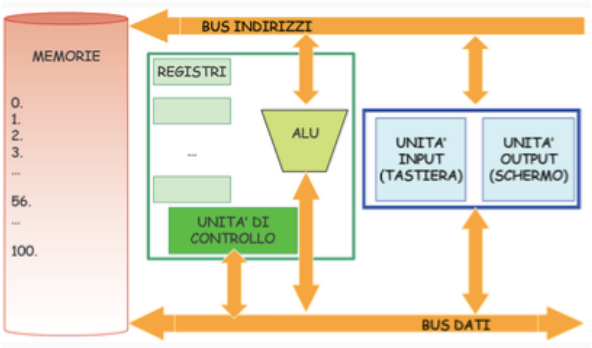
Anche i programmi, nel calcoaltore, sono codificati come sequenze di bit, queste sequenze devono rappresentare le istruzioni (codice operativo) ossia cosa il computer cosa deve eseguire e i dati (operando) sul quale viene operato.

Architettura del calcolatore e manipolazione dei dati

Il modello di von Neumann

I componenti fondamentali dei calcolatori moderni non è sostanzialmente cambiata rispetto ai primi calcolatori, ossi sono composti da:

- unita di elaborazione: CPU (unita centrale di processamento dei dati) coordina le varie parti
- memoria centrale: RAM (circuiti che permettono di memorizzare dati e programmi in maniera volatile)
- periferiche: memorie di massa, (memorizzano i dati in maniera permanente ossia anche in assenza di corrente e per lungo tempo), schermo, stampante, tastiere…
- bus di sistema fili attraverso cui passano bit di informazione e collegano la CUP alla memoria e viceversa



CPU (central processing unit) chi manipola i dati

`e l'unità centrale di elaborazione, permette di effettuare un qualsiasi tipo di computazione. Alloggiata sul microprocessore (su wafer di silicio), dirige e controlla ogni attività del computer e coordina le attività di memoria e delle unità periferiche oltre ad eseguire tutte le operazioni aritmetiche e logiche relative ad esempio ad un programma che si sta eseguendo. Tutte le operazioni sono eseguite in bit (0 e 1), a livello fisico della CPU abbiamo dei segnali elettrici a livelli di tensione distinti

Caratteristiche:

- è elettronica
- interpreta le istruzioni del linguaggio macchina (il limitato insieme di istruzioni che la CPU è in grado di comprendere ed è specifico per ogni CPU- un programma su window non pou essere letto allo stesso modo da IOS)
  CUP diverse hanno codificano, a livello del hardware, linguaggi macchina (set di informazioni) disitinti
- accede alle locazioni di memoria della RAM: un dato deve essere trasferito dalla memoria ad una propria memoria interna e viceversa
- è costituita da molte componenti, delle quali la piu importante è la Arithmetic Logic Unit (ALU) insieme di circuiti che consentono di interpretare ed eseguire le istruzioni del linguaggio macchina (istruzioni aritmetiche e logiche, di controllo di un programma, di trasferimento dati)
- è il processore

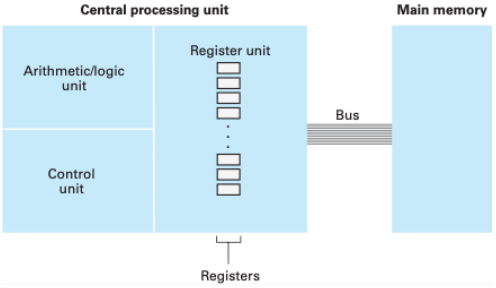
Caratteristiche tecniche

- Piccole (chiamate anche microprocessori)
- Montate su una scheda madre

Oggigiorno le CPU possono essere costituite da diversi "core", ognuno dei quali costituisce una CPU indipendente: di fatto si viene a creare un sistema "multiprocessore" utilizzando un solo integrato; Questo rende possibile l'esecuzione contemporanea di numerosi processi (programmi) senza rallentamento del sistema. La velocita di una CPU si misura in Hz ossia il nuemro di istruzioni al secondo che la CPU puo eseguire.

Tre componenti

- Arithmetic Logic Unit (ALU)
- Control unit
- Register unit (special/ general purpose register) è la memoria interna della CPU composta insieme di celle chiamati registri



Per eseguire una somma elementare il sistema digitale deve fare un certo lavoro. Cosa significa fare una somma?

- Non solo l'algoritmo di somma
- Si muovono dati
- Si usano registri e indirizzi di memoria

Stored-program (nella memoria principale) vengono trattati in maniera uniforme dati e programmi
Cache memory (e gerarchie di memorie) tra la CPU e la RAM sono presenti altri livelli di memorie piu piccoli e veloci (accesso è veloce se la cella è vicina)

Il linguaggio macchina

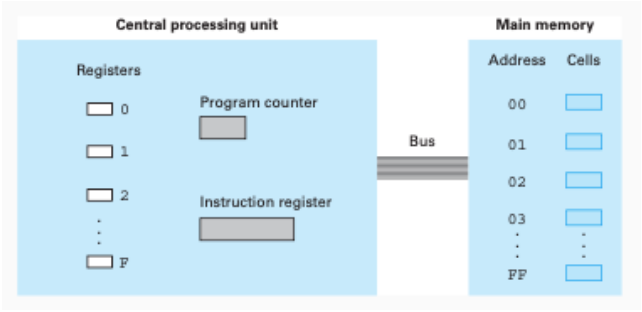
Collezione delle istruzioni direttamente eseguibili dalla CPU, due diverse filosofie di approccio alla progettazione di una CPU:

- reduced instruction set computer (RISC: power PC – fixed length) ja una circuiteria piu complessa: offre un linguaggio di istruzioni elementari piu amplio, implementa nell'hardware delle operazioni relativamente soffisticate
  sono meno potenti dal punto di visto computazionale
- complex instruction set computer (CISC: intel – variable length) si costruisce un architettura piu semplice con un linguaggio macchina ridotto
  maggiore potenza computazionale ma un maggiore consumo energetico

(poche istruzioni)

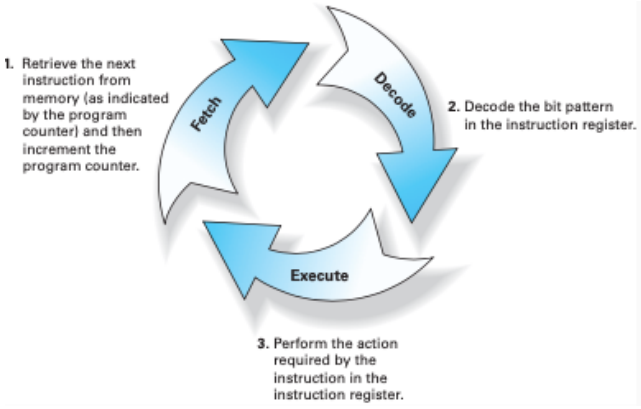
Le istruzioni del linguaggio macchina si dividono in tre tipologie:
data transfer (LOAD/STORE – I/O)
arithmetic-logic (AND, OR, XOR, SHIFT, ROTATE)
control (JUMP, STOP…) istruzioni di controllo di flusso di esecuzione dell'istruzione

esempio
consideriamo una architettura da 16 registri e RAM da 256 celle (in esadecimale).
Usiamo indirizzi scritti in esadecimale
Il program counter è un registro il cui scopo è conntenre l'indirizzo di memoria della prossima istruzione da eseguire (ogni CPU esegue un istruzione alla volta), deve avere una dimensione sufficiente a indirizzare ogni cella della RAM (256 celle -> 8 bit)



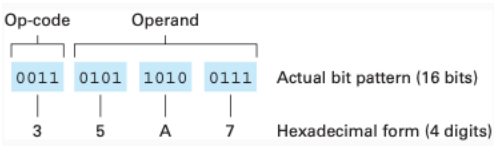
Istuction register ha il compito di memorizzare l'istruzione che deve essere eseguita cioe la sequenza di bit che rappresenta l'operazione che la CPU che deveeseguire nel successivo ciclo di clock. Puo avere una dimensione diversa dal program counter poiché la dimensione dipende da quante istruzioni la CPU supporta e dalla dimensione degli operandi che operano su queste istruzioni
Quindi l'esecuzione di una qualsiasi istruzione deve avventire in questo modo (esecuzione di un programma):

1. la CPU deve vedere nel program counter qual è l'indirizzo di memoria dove andare a reperire la prossima istruzione (es. 03)
2. successivamente chiede di accedere all'indirizzo di memoria e trasferisce il contenuto di quel indirizzo (ed eventualmente anche degli indirizzi successivi) nel instruction register
3. l'istruzione viene trasportata nel istruction register, a questo punto la CPU decodifcia quel'informazione attraverso una serie di circuiti interni ed esegue l'operazione corrispondete (es. operazione di somma)
4. determinato che si tratta di un operazione di somma dovra andare a recperare gli operandi che si trovano nella RAM nelle celle successive
5. gli operandi vegnono memorizzati nei registri
6. CPU è pronta a eseguire l'operazione e ha i dati su cui operare
7. Trasferimento dati



Codifica delle istruzioni macchina (appendix C)
La codifica dell'istruzione dipende dall'architettura della CPU

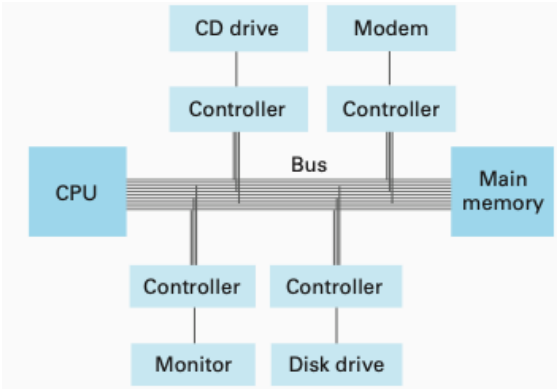
- Op-code
- Operand



Usiamo il primo carattere dell'operando per indicare il registro (ne abbiamo 16) e gli altri due per indicare la cella nella RAM (ne abbiamo 256) potenze di due

Le istruzioni devono venire:

1. caricate (dalla memoria principale) nella CPU
2. interpretate
3. eseguite



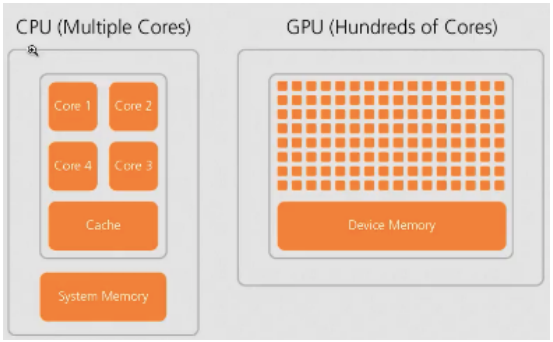
il ruolo dei registri, due importanti registri special purpose

- instruction register
- program counter

una importante nozione

- ciclo macchina

nei dispositivi moderni le CPU sono dispoitivi di computazione paralleli
il ciclo di esecuzione nei dispositivi moderni viene eseguita all'interno dei core (solitamente sono presenti piu core indipendenti tra loro).
La CPU ha un basso grado di parallelismo ma è piu sequenziale
La GPU esegue dei task che tipicamente operano eseguendo la stessa istruzione in parallelo su grandi quantità di dati.

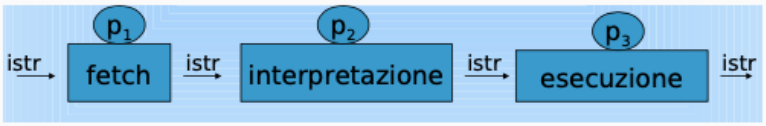


Sistemi operativi ed estensione dell'archiettura

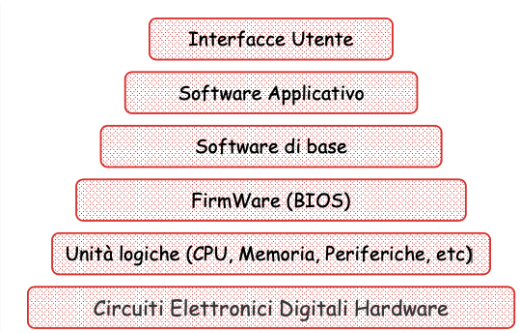
I limiti dell'architetttura di von Neumann:
non implementa alcun tipo di parallelismo
la CPU trasferisce i dati sui bus in modo sequenziale
…sono presenti dei limiti facilmente superabili senza stravolgere il sistema.

Estensioni e miglioramenti

Processori dedicati e co-processori (matematici)
Pipelining: le varie fasi vengono affidate a diverse componenti che operano contemporaneamente



- Gerarchie di memorie e caches
- Processori dedicati al I/O
- Architetture multi-processore



- ➔ Ci permettono di interagire con queste macchine
- ➔ Software di tipo applicativo
- ➔ Sistema operativo
- ➔ Istruzioni del linguaggio macchina codificate in un dispositivo fisico (funzione capire quali sono le periferiche collegate
- ➔ Architettura fondamentale dei dispositivi
- ➔ Struttura del hardware

Il sistema operativo è il software che opera direttamente sul hardware (coordina tutte le attivita del dispositivo)
Definisce una gerarchia di macchine virtuali
È composto da vari strati che sono definiti in termini di funzioni fornite all'utente (buccia di cipolla), c'è un nucleo (karnel) che offre funzionalità elementari a tutti gli strati superiori, al di sopra del nucleo è possibile costruire strati che ci permettono di costruire file, di gestire la memoria e i processi in atto

Definizione sistema operativo: è il software che consente agli utenti di utilizzare i calcolatori. Dal punto di vista dell'utente, il sistema oerativo consente l'utilizzo efficace delle risorse fisiche della macchina definendo le modalita d'interazione utente calcolatore

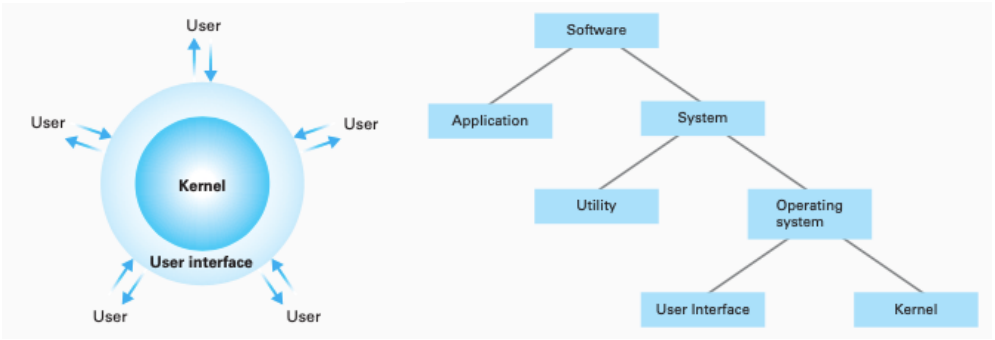
Compiti del sisitema operativo:

- semplificare agli utenti l'accesso e l'utilizzo delle risorse
- presentare ai processi le risorse in modo astratto
- arbitrare
- proteggere

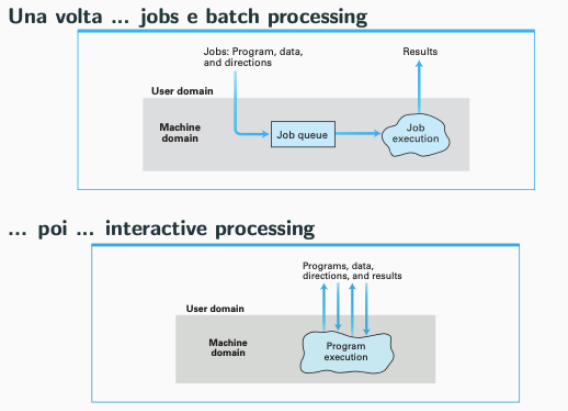
le funzioni più importi di un sistema operativo includo:

- la gestione dei processi
- la gestione della memoria
- la gestione del file system
- la gestione delle periferiche
- la gestione della rete
- la gestione della sicurezza e della privatezza dei dati.

Esempi di sistemi operativi: windows, unix, macos, linux



- Interfaccia: è un entità che funge da intermediario nella comunicazione tra il computer e l'utente umano.
- kernel: fornisce le funzionalità di base e poi l'interfaccia utente permette l'interazione con il sistema operativo stesso.
- Utility: terminale, editor di testo
- Applicazioni



Nei primi sistemi operativi tutto era svolto in maniera sequenziale e non era interattivo per cui era necessario creare un programma inserirlo nel computer e venivano messi in "coda"(processo sequenziale)

Il processo interattivo è quello utilizzato oggi dai calcolatori eseguiamo più applicazioni ottenendo dei risultati e normalmente eseguendo più tack contemporaneamente, il processamento è di tipo real time (non era possibile con i vecchi calcolatori).

Ogni CPU puo eseguire un istruzione alla volta pero con i dispositivi attuali son in grado di fare piu cose contemporaneamente, questo è permesso dal parallelismo inoltre viene suddiviso il tempo in intervalli molto brevi (chiamati quanti) e a allocare a ciascun tratto di tempo un particolare tipo di processo (time-sharing: permette di eseguire più task contemporaneamente – cambio di contesto). Il cambio di contesto avviene:

1. copiando il contentuo dei registri della CPU nel PCB del processo attuale in esecuzione
2. Copiare il contesto del processo successivo dal PCB ai registri della CPU

Il PCB è una tabella residente in memoria,che contiene varie informazioni, le più importanti delle quali sono il process ID (ogni processo è identificato da un numero) e il contesto del processo (contento dei registri della CPU a un dato istante)
Multitasking: l'utente può fare più operazioni simultaneamente

Nucleo – kernel

E' lo strato del S.O. che gestisce i processi

- Un processo `e la coppia costituita da un programma e dall'insieme dei valori contenuti in memoria (RAM e registri) per la sua esecuzione
- i processi sono in esecuzione /attesa (in attesa di un dato ottenuto dalla periferica) /ready (pronti ad essere eseguiti)

Un processo è un programma in esecuzione (un programma in esecuzione viene detto processo)
il programma è un identità statica ossia una sequenza di istruzioni scritta in un linguaggio (linguaggio macchina)

Memory manager -Gestore della memoria

E' lo strato del S.O. che

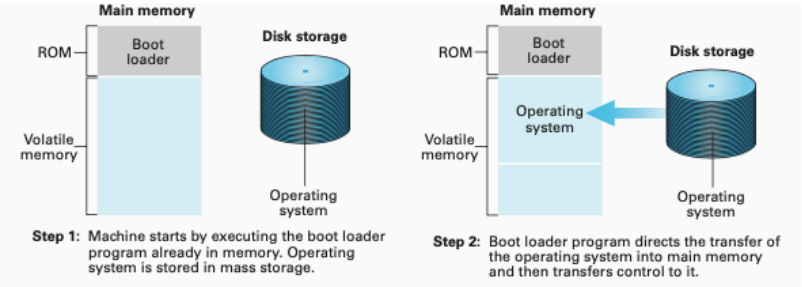
- ripartisce la RAM fra i programmi in esecuzione
- alloca la memoria (ripartizione, assegnazione e conversione degli indirizzi di memoria
- gestisce la segmentazione (blocchi di lunghezza variabile) e la paginazione (blocchi di lunghezza fissa)

Memoria virtuale: spazio logico ossia che non esiste fisicamente ma è cio che il sistema operativo mostra ai processi, spazio di indirizzi logici che permettono di asseganre ai processi delle locazioni di memoria in questo spazio virtuale

Gestione delle periferiche (gestore)

E' lo strato del S.O. che

- si occupa dell'interfaccia con le periferiche (gestione delle periferiche)
- "nasconde" all'utente le caratteristiche dell'hw per le operazioni di I/O
- gestisce le code per l'accesso alle periferiche
- periferica di imput: tastiera, mouse, microfono
- periferica di output: schermo, stampante



Gli strati piu` importanti (per noi)

- File System (permette di utilizzare la memoria di massa)
- Interprete dei comandi (permette di sfruttare a pieno le potenzialità della macchina)

File System e sua gestione

E' lo strato del S.O. che si occupa di leggere, scrivere, ... gestire i file nella memoria di massa

- Creazione
- rinomina, movimento, privilegi, ...
- resa trasparente dell'hw per l'utente

Memoria virtuale.

Example

L'interfaccia ed i comandi per accedere al file system di questo computer

- i files devono essere manipolabili in modo trasparente all'utente rispetto all'hw
- directories (cartelle): servono a dare una struttura ad albero al file system (sono i nodi interni all'albero)
- il file system contiene directories in cui `e memorizzato il sistema operativo e tutti i moduli ad esso necessari

Le finestre sono i nodi dell'albero

"entrare" in una finestra equivale a scendere nell'albero • i files vengono rappresentati mediante icone e:

- hanno un suffisso
- il suffisso ne identifica il tipo
- l'icona ne rappresenta graficamente il tipo

la nozione di alias diventa particolarmente importante ed utile

Reti

Gestione di rete

È fornita dal sistema opertivo poter collegare un dispositivo a una rete di calcolatori.
Il collegamento puo essere un cavo una fibra ottica…. La comunicazione avviene scambiando informazioni codificate in un formato digitale.
Due o piu dispositivi collegati attraverso un canale di comunicazioni formano una rete di calcolatori o network.
Ciasuno dei dispositivi collegati a una rete è chiamato host oppure nodo della rete.
La comunicazione si basa su una codifica digitale dell'informazione in termini di seqeunze di bit. la velocita di trasmissione dei dati attraverso una rete è solitemnte misurata in multipli di bit al secondo. Un insieme di regole che stabilisce come possa avvenire la comunicazione tra dispositivi interconnessi, a livello hardware o software prende il nome di protocollo di rete.
Le reti possono essere classificare in:

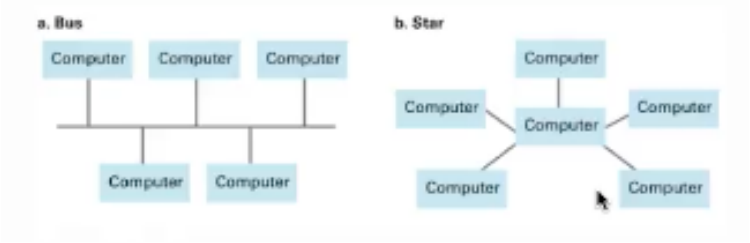
- scala (dimensioni geografiche)
    - PAN reti con estensione limitata a pochi metri, solitamente wireless (bluetooth)
    - LAN (rete locale) rete confinata a un edificio o parte di esso (rete casalinga). È molto veloce ed efficiente (wifi) , è interamente sotto il controllo di una singola organizzazione.
    - MAN sono reti che coprono aree chilometriche (TV via cavo)
    - WAN sono reti che collegano computer a distanza molto grande, i nodi e l'infrastrutttura di rete sono gestiti separatemente. (3G, 4G)

La rete internet puo essere definita come una rete di reti, essendo costituiuta da una moltitudine di reti a volte interconnesse tra loro. Poiche network diversi possono adottare protocolli di rete incompatibili, per permettere a dispositivi che si trovano in reti separate di capirsi sono necessari dispositivi in grado di effettuare le necessarie traduzioni da un protocollo all'altro, tali dispositivi si chiamano gateway

- Tipi di protocolli
    - Reti aperte sono reti il cui funzionamento interno è basato su protocolli di pubblico dominio.(TCP/IP)
    - Reti chiuse sono reti basate su protocolli proprietari, controllari da una particolare organizzazione spesso vincolati da licenze

Internet è un sistema aperto

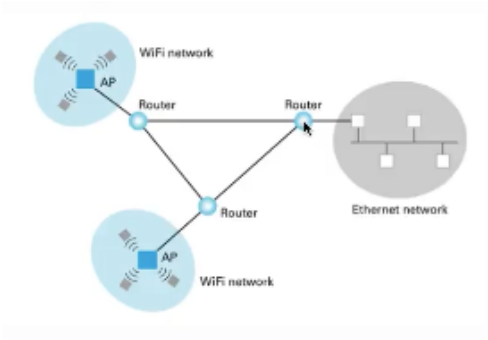
- Topologia si riferisce alla forma con cui i dispositivi sono collegati tra loro
    - Topologia a bus i nodi della rete sono collegati a una linea di comunicazione. Sono un dispositivo alla volta puo comunicare con il bus (altriemnti le informazioni si mescolano) sono quindi necessari meccanismi di arbitraggio per risolvere i conflitti quando due o piu nodi tentano di trasmettere simultaneamente.
    - Topologia a stella: esiste un computer che ha un ruolo centrale e funge da intermediario tra tutti gli altri dispositivi collegati alla rete. I nodi della rete sono collegati a un unico nodo speciale e la comunicazione passa semrpe attraverso tale nodo. Le reti wireless hanno tale topologia: la comunicazione in tal caso è coordinata e passa attraverso un dispositivo centrale detto access point (AP).



- tecnologia di trasmissione
    - reti broadcast: sono reti in cui i nodi comunicano attraverso un canale di trasmissione condiviso (reti ethernet)un host che voglia comunicare con un altro host invia messaggi detti pacchetti a tutti i nodi della rete. (LAN)
    - Reti punto a punto: sono reti le cui connessioni sono connessioni dirette tra coppie di nodi. Per passare da un nodo all'altro un messaggio deve passare diversi altri nodi intermedi. La comunicazione punto a punto tra un nodo mittente e un nodo ricevente è talvolta chiamata unicasting (WAN, MAN)

Un dispositivo può inviare messaggi in modo diretto esclusivamente ad altri dispositivi nella stessa rete. In una moderna rete basata su Ethernet, i nodi di una LAN sono connessi tra loro attraverso special dispositivi chiamati switch, che hanno lo scopo di inoltrare il traffico proveniente da un nodo verso un altro.

La connessione internet è possibile grazie a dei cavi che passano attraverso gli oceani che ci permettono di comunicare con un computer che sta dall'altro capo del mondo



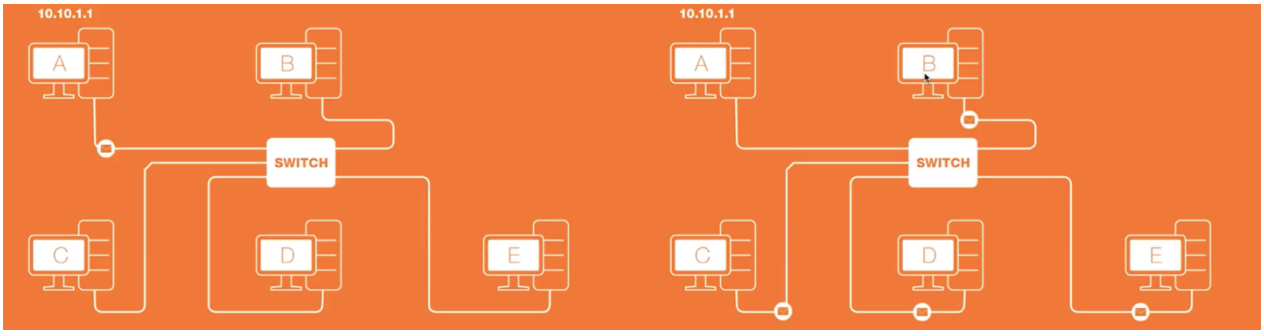
due LAN WiFi e una LAN Ethernet sono collegate insieme attraverso tre router, a formare un "internetwork". Se un host in una dette LAN vuole invieare un messaggio a un host che si trova in una LAN diversa deve innanzitutto trasmettere il messaggio al proprio AP (access point). L'AP lo inoltra al router associato, il quale a sua volta lo inoltra al router dell'altra rete, che insine lo trasmette all'host di destinazione. La comunicazione tra i due host è dunque di tipo punto a punto.

La destinazione di un messaggio è specificata mediante uno schema di indirizzamento. Qualunque dispositivo connesso a una rete possiede una perifericahardware, detta scheda di rete (NIC) cui è assegnato un indirizzo fisico (un indirizzo MAC) che è univoco a livello globale.
Gli indirizzi MAC sono sequenze di 48 bit solitamente rappresentate mediante 12 cifre esadecimali.
A qualunque dispositivo connesso a una rete viene inoltre associato un indirizzo logico o indirizzo IP. A differenza dell'indirizzo MAC, che è incolato all'hardware, l'indirizzo IP è associato a un dispositivo via software dal sistema oeprativo ed è dunque dinamico, nel senso che lo stesso dispositivo puo avere piu indirizzi IP in istanti diversi.
Oltre all'identificatore di rete, esiste un altro indirizzo speciale, detto indirizzo di broadcast, che viene usato quando un host vuole inviarti un messaggio non a host spcecifico, bensi a tutti gli altri host della rete (broadcasting)
l'indirizzo si ottiene a partire da un indirizzo IP e una maschera di sottorete mettedno a 1 tutti i bit dell'host id.

Esercizio si assuma che una rete abbia maschera di sottorere 255.255.255.192
Qual è il netword id di un host con indirizzo IP 10.10.1.48?
Convento in una sequenza di bit la maschera di rete e l'indirizzo IP faccio l'and logico e poi riconverto in decimale
Qual è il suo host id?
È la parte finale del netword id i cui bit sono a 0
L'indirizzo di broadcast?
Lo ottengo mettendo a 1 tutti i bit che nel network ID sono a 0
Quanti host al massimo puo avere la rete?
si tratta di capire qual è il numero di bit dedicati all'identificatore di host, in base a quel numero so quanti host diversi posso allocare

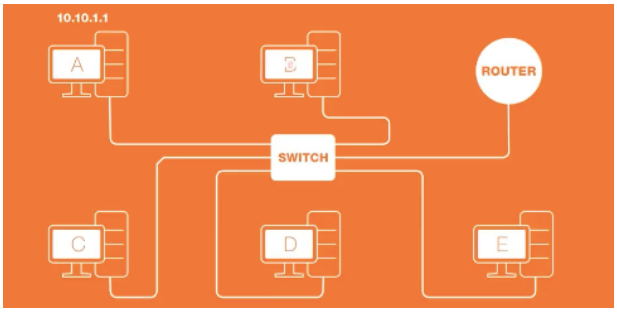
Comunicazione in una switched LAN

Vediamo più in dettaglio come avviene uno scambio di messaggi tra due host che appartengono alla stessa rete locale. Dopo aver determinato che B si trova nella stessa rete di A, A deve determinare l'indirizzo fisico (MAC) di B, perché tale indirizzo è necessario per poter inoltrare correttamente un messaggio. Supponiamo che, inizialmente, A non conosca l'indirizzo MAC di B: per poterlo scoprire, può inviare una richiesta in broadcast. Il contenuto di tale messaggio, informalmente, è: "Hey, sono l'host 10.10.1.1 e ho l'indirizzo MAC 01:02:03:04:05:05:06 se sei 10.10.1.2, per favore inviami il tuo indirizzo MAC". Tale messaggio è ricevuto da tutti gli host della rete (anche C,D ed E nel video), ma è da questi ultimi ignorato. L'host B tuttavia, risponde al messaggio comunicando ad A il proprio MAC (B non ha bisogno di fare un broadcast, perché l'indirizzo MAC di 4 gli è noto, essendo contenuto nel primo messaggio). A questo punto A può iniziare la comunicazione vera e proprio indirizzando i propri messaggi direttamente a B.



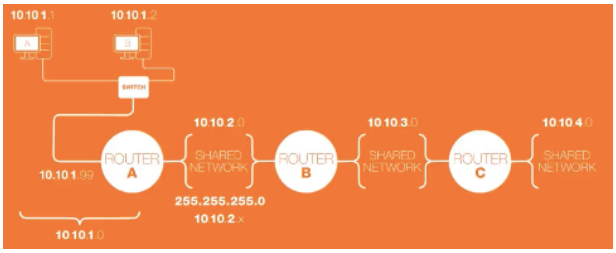
cosa succede se un nodo vuole comunicare con un nodo che non è all'interno della rete.

L'host A raggiunge la Switch e questo messaggio (indirizzato ad un indirizzo IP che è al di fuori della rete) la switch fa un broadcast del messaggio a tutti gli host della rete locale. Tutti gli host lo rifiuteranno tranne il router che riconsce che il messsaggio è indirizzato ad un computer al di fuori della rete e sulla base dell'indirizzo IP del messaggio decide a quale altro dispositivo al di fuori della rete inviarlo.



Il router (gateway) ha almeno due interfaccia di rete

- Una che comunica all'interno della rete (rete locale)
- Una che comunica con l'esterno

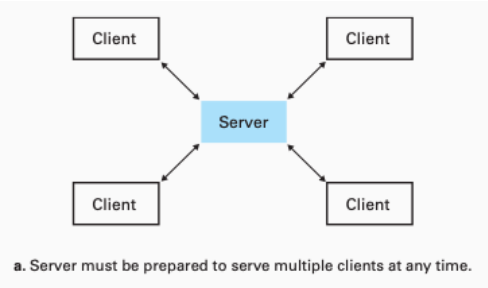


10.10.1.1(host A) , 10.10.1.2 (host B) e 10.10.1.99 (gateway) questi tre nodi sono nella stessa rete locale il cui network è 10.10.1.0 quindi la sua maschera di rete è 255.255.255.0. Grazie alla switch i due dispositivi possono comunicare tra loro. Host a e host B hanno un interfaccia di rete mentre il gateway ha due interfaccia di rete (una connessa alla rete locale e una comunicante con l'esterno.
Il router traduce gli indirizzi IP dei messaggi che provengono dall'interno della rete verso indirizzi IP che sono visibili dall'esterno (dal punto di vista del router B tutti i messaggi che provengono dalla rete locale con indirizzo 10.10.1.0 sembrano tutti provenire del router A), quindi dal puto di vista dei router esterni il traffico di rete sembra provenire dal router A, quindi quando un messaggio esce dalla rete locale il suo indirizzo IP viene tradotto nel indirizzo IP del router A, quando questo messaggio ottiene una risposta il router A deve dare la traduzione opposta scoprendo a quel dei dispositivi all'interno della rete è destinato della rete locale.

All'interno del dispositivo devo individuare il processo con la quale voglio comunicare, per poter definire una comunicazione è necessario conoscere l'indirizzo IP del dispositivo con cui voglio comunicare e un qualche identificatore del processo all'interno del dispositivo con cui voglio comunicare (porta – numero intero)
Esempio
Se voglio accedere al sito dell'università di Udine scrivo https://www.uniud.it/it ma in realtà sto stabilendo una connessione tra questo browser e qualche processo di qualche computer da qualche parte, per poter stabilire questa connessione io ho specificato l'identificatore di questo processo (che è la porta -443 nuemro convenzionale).

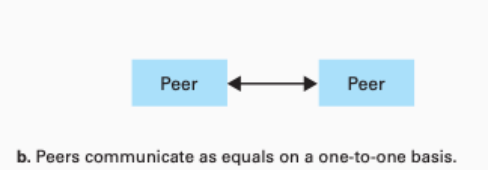
Metodi di comunicazione tra processi

Client/server: modalita di comunicazione per cui esiste un computer (server) nel quale è in esecuzione un processo che rimane "sempre in esecuzione " e rimane in attasa di ricevere richieste di comunicazione. La comunicazione è sempre iniziata da un altro dispositivo chiamato client, il client deve conscere l'indirizzo IP del server e la porta del processo con cui vuole comunicare in quel server (indiando una richiesta di comunciazione con il server) modaita del WEB, print server, file server). Puo accettare richieste di molteplici dispositivi, processarle e dare a ciascuno dei dispositivi una risposta.



a. Server must be prepared to serve multiple clients at any time.

Peer to peer (P2P) nella quale I dispositivi giocano tutti lo stesso ruolo quindi possono essere sia che inviano richieste (client) che ricettori delle richieste (server), client e server sono uniti (torret)
Anche in questo caso è necessario che i dispostivi conoscano gli indirizzi IP e le porte dei processi.

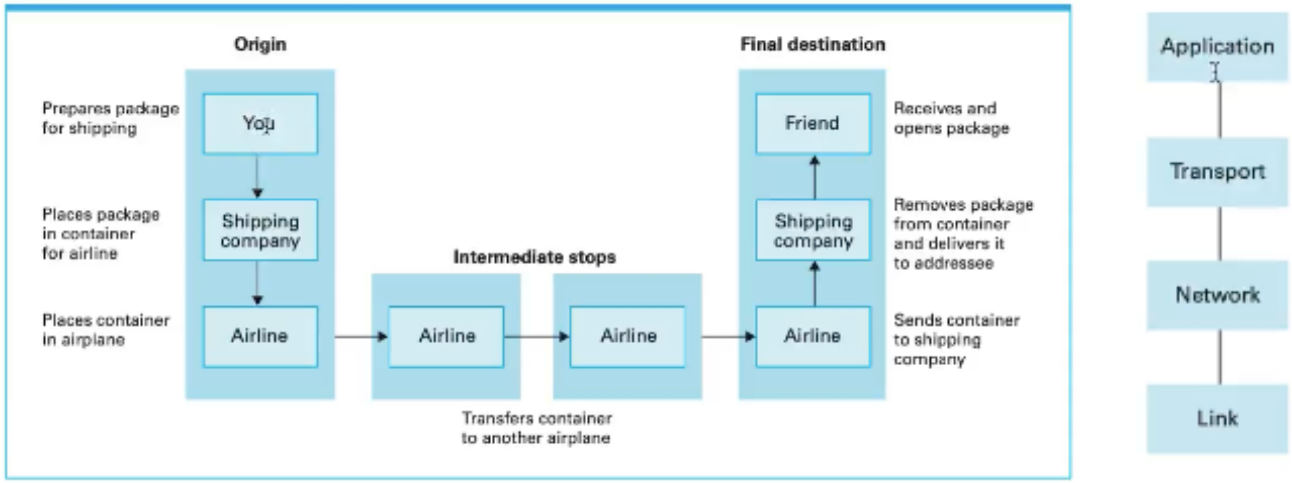


b. Peers communicate as equals on a one-to-one basis.

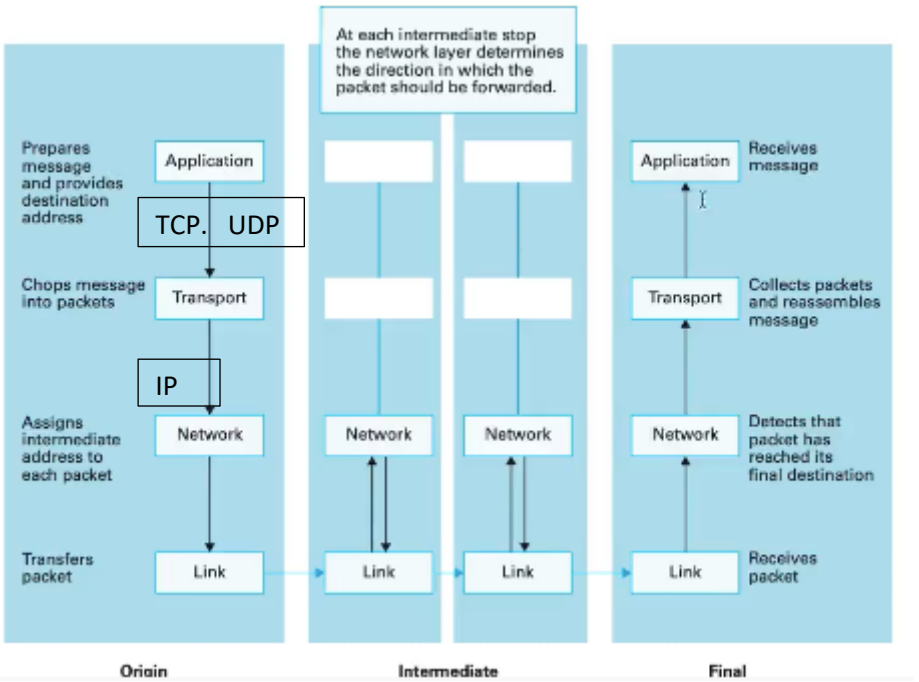
L’indirizzo IP cambia quindi è molto piu difficile per un dispositivo ricnoscere l’indirizzo IP dell’altro dispositivo, questo viene mediato da dispositivi client server che funzono da una sorta di catalogo di interizzi e mantengno traccia di dove si trovano i dispositivi.

Qualunque sia il metodo di comunicazione se il dispositivo non conosce l’indirizzo IP e la porta del dispositivo non può comunicare

I servizi di rete sono forniti dal sistema operativo e l’approccio che normamlemnte viene utilizzato nella realizzazione di questi protocolli di rete è detto stratificato (pila di protocolli)



Le reti funzionano nello stesso modo: ll’livello piu alto sono le applicazioni (browser) che interagiscono con altri livelli che si trovano nel sistema operativo e servono per il trasporto dei messaggi (trasporto, network, rete) e successivamente al livello fisico (cavi, fibra ottica che ci permettono di trasportare fisicamente il messaggio.



esempio: supponiamo di voler comunicare con un altro dispositivo, per esempio voglio collegarmi con il sito dell’universita di udine:

1. l’applicazione a sx è browser – applicazione a dx è il programma in esecuzione in un server che fornisce la funzionalità delle applicazioni web. Il browser invia la richiesta di comunicare con il sito uniud,
2. il messaggio viene passato al livello di trasporto nella pila di protocolli di rete il quale si occupa di suddividere il messaggio in una serie di pacchetti e di numerare i pacchetti in modo tale da poterne ricostruire l’ordine
3. passa poi questi pacchetti al livello di rete, questo ha la funzionalità di determinare dove inviare i pacchetti (instradamento dei pacchetti), ciascun pacchetto viene inoltrato in maniera indipendente dall’altro, questo è possibile perché questi pacchetti sono numerati e poi riordinati.

4. Infine questi pacchi vengono passati al livello di collegamento (link) dove sono definiti i dettagli del canale di comunicazione che viene utilizzato per la connessione. I dati vengono trasmessi ai canali fisici.
5. I pacchetti vengono poi trasmessi di nodo in nodo, in ciascuno di questi nodi intervengono il livello di rete e di collegamento perché in ciascuno di questi nodi funge da router e quindi deve determinare dove devono essere inoltrati i pacchetti
6. Quando il pacchetto giunge al nodo di destinazione, il pacchetto viene passato al livello di trasporto, i pacchetti vengono riordinati e passano all'applicazione

Internet:

rete aperta
logicamente una sola rete – fisicamente milioni di reti
unico protocollo per i livelli intermedi di comunicazione (TDP/IP)
internet service providers (ISPs): collegano una rete ad internet

gli indirizzi su internet:

IP address: Internet Protocol address

- 32 bit: quattro numeri compresi tra 0 e 255 scritti in dotted decimal notation
- Vengono distribuiti dalla Internet Corporation for Assigned Names and Numbers (ICANN)
- IP address: versione human readable, Top Level Domains (TLD), domini (registrati), sottodomini

Name servers: domanin name servers (DNS) DNS lookup

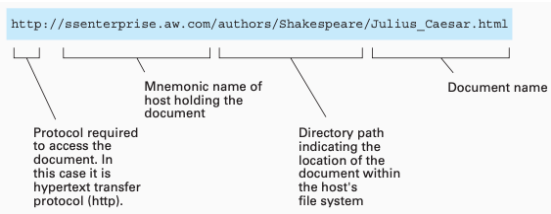
Protocolli presenti nella comunicazione

TCP: protocollo di trasporto, stabilisce le regole per suddividere un messaggio in pacchetti (affidabile – garantisce che un messaggio inviato sarà ricevuto al destinatario) richiede di stabilire una connessione con la destinazione nel senso che i primi messaggi che vengono inviati tra due nodi rappresentano una garanzia (nei siti WEB, posta elettronica).
La trasmissione dei messaggi è coordinata tra i due nodi mediante dei. Meccanismi che consentono di modulare il tasso con cui vengono trasmessi i pacchetti, per cui se la rete è congestionata il server si trova in uno stato di sovraccarico il client rallenta la trasmissione dei pacchetti
UDP: protocollo di trasporto (non affidabile perché una volta che il messaggio è stato suddiviso in pacchetti ciascun pacchetto viene inoltrato al protocollo di rete e viene "dimenticato" nel senso che non viene più controllato – può andar perso). Non ha la possibilità di modulare il tasso di invio dei messaggi. Richiede meno computazioni è adatto ad applicazioni che non necessitano di una sicura ricezione dei pacchetti.
IP: protocollo a livello di rete che si occupa dell'inoltro e dell'istradamento dei pacchetti

Quando stabilisco una connessione scrivo una stringa di caratteri e non l'indirizzo IP, quindi per conoscere l'indirizzo IP corrispondente al server interessato, nei servizi di rete, esiste una funzionalità di traduzione di nomi in indirizzi IP; questa funzionalità è data dal Domain Name Service, basato su un protocollo client server che si appoggia al protocollo UDP) , che è presente nei sistemi operativi e ha lo scopo di associare a dei nomi i corrispondenti indirizzi IP.
www.uniud.it

- it (TLD -top level domain) suffisso della stinga, determina che il sito di trova sotto un dominio in italia
- uniud: "la parte caratteristica"
- www: risorsa del world wild web

http è un protocollo client server che appoggia al protoccolo TCP, viene utilizzato per le connessioni che fanno i nostri browser per i vari siti web, è un protocollo di tipo testuale



la rete essendo utilizzata da tutti devono esserci delle protezioni (per evitare che tutto ciò che una persona comunica e che riceve venga registrato da più parti) ci sono più tipologie di protezione:

crittografia: tutto ciò che non viene trasmesso in maniera crittografata è visibile da tutti i nodi che i pacchetti attraversano, non tutto può essere crittografato poiché molti dati devono essere trasmessi in chiaro (es. indirizzo IP) quindi i nodi intermedi possono non essere in grado di sapere quali informazioni fluiscono tra un nodo e un altro ma sono in grado di determinare che due nodi vogliono comunicare tra loro (e questa è già un informazione) un esempio e la cittografia end to end il messaggio viene cifrato nel mio dispositivo prima di essere trasmesso, poi passa attraverso i nodi intermedi, arriva al destinatario e il messaggio viene decifrato nel dispositivo del destinatario (gli unici che possono leggere il messaggio sono il mittente e il destinatario).

Algoritmi (e complessità)

Definizione Kleene: Un algoritmo è una risposta finita ad un numero di domande, cioè la possibilità di codificare in qualche modo una procedura o delle operazioni che ci permettono di rispondere ad un numero arbitrariamente grande di domande
Un algoritmo è un insieme ordinato di passi elementari non ambigui ed eseguibili. Questa sequenza di passi definiscono un processo che ha termine. È una procedura descritta in maniera rigorosa ma che puo essere definita in un tempo finito.

Prima fase: capire il problema
Seconda fase: elabora un'idea si come una funzione algoritmica potrebbe risolvere il problema
Terza fase: formula l'algoritmo e rappresentalo come un programma
Quarta fase: valuta il programma per la precisione e per il suo potenziale come strumento per risolvere altri problemi

!! Un problema (computazionale) specifica una relazione input-output
Esempi: un problema di ordinamento
Input: una sequenza di numeri naturali
Output: la sequenza ordinata
Un algoritmo che risolve il problema specifica una procedura effettiva (descritta da passi elementari) per ottenere la relazione desiderata

Modelli di computazione

Un modello di computazione `e un'astrazione matematica di ciò che riteniamo sia una "macchina calcolatrice"
Definisce la tecnologia con cui sono realizzati gli algoritmi
Specifica:

- le operazioni primitive e il loro costo
- le risorse disponibili (spazio, memoria e tempo di esecuzione) e il loro costo

Ogni problema può avere diverse soluzioni algoritmiche, o non averne alcuna. . .
Qual `e l'algoritmo "migliore"?

- Occorre un modo per confrontare l'efficienza degli algoritmi
- Bisogna definire che cosa si intende per "efficienza"
- L'efficienza si può misurare sulla base delle risorse richieste dall'algoritmo si misura in base alle risorse che l'algoritmo richiede (tempo di esecuzione e spazio di memoria)

Analisi degli algoritmi

Analizzare un algoritmo significa prevedere le risorse che l'algoritmo richiede. Quali risorse? Principalmente

- tempo di esecuzione
- spazio di memoria

Soprattutto il tempo `e un fattore importante, lo spazio si può riutilizzare, il tempo no
Queste risorse dipendono, in generale, dai dati in ingresso (input)

Tempo di esecuzione di un algoritmo su un particolare input: `e il numero di operazioni primitive, o passi, eseguiti ad esempio, assumiamo che ciascuna linea del codice precedente costituisca un passo e che ogni passo richieda un tempo fissato T0 allora, il tempo di esecuzione della ricerca varia da 3T0 (caso migliore) a (3 + 2|L|)T0 (caso peggiore).

l tempo di esecuzione dipende dall'input
dalla lunghezza della lista (numero di elementi della lista)
dalla posizione del valore nella lista
In generale, il tempo di esecuzione cresce con la dimensione dell'input

La stima del tempo di esecuzione nel caso peggiore `e particolarmente importante

- è un limite superiore al tempo di esecuzione
- "non può andare peggio di cos`ı"

Altre stime interessanti:

- tempo medio, spesso difficile da calcolare , richiede assunzioni sulla distribuzione dell'input
- tempo del caso ottimo limite inferiore alle prestazioni

L'analisi del tempo di esecuzione richiede:

- un formalismo per la specifica dell'algoritmo
- l'attribuzione di un costo ad ogni operazione primitiva

Ulteriore semplificazione:

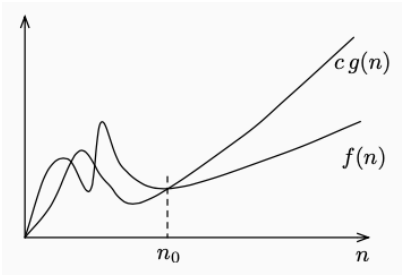
- ciò che interessa `e l'andamento, o ordine di grandezza, del tempo di esecuzione del input
- l'algoritmo di ricerca, nel caso pessimo, richiede un tempo proporzionale a |L|,
- non interessa conoscere le costanti

Gli algoritmi sono delle teconologie. Esempio
Un processore Ghepardo a 1Ghz esegue un algoritmo di ordinamento che richiede $2n^2$ operazioni per ordinare n numeri
Un processore Bradipo a 1Mhz (anni '80) esegue un algoritmo di ordinamento che richiede $50n \log_2 n$ operazioni su n numeri
Tempo per ordinare 10^7 numeri:
Ghepardo: $\frac{2\cdot(10^7)^2}{10^9} = 2\cdot 10^5 \approx 56$ ore
Bradipo: $\frac{50\cdot(10^7)\log_2 10^7}{10^6} \approx 3.2$ ore

limite superioire asintotico per una funzione g(n):

$O(g(n)) = \{f(n) | \exists c>0 \exists n_0>0 \ \forall n \geq n_0.\ 0 \leq f(n) \leq c\, g(n)\}$

f(n) tali che esiste una costate c positiva e una costante $n_0>0$ tali che per ogni valore di n maggiore di n0 la f(n) è superiormente limitata da c per g(n)

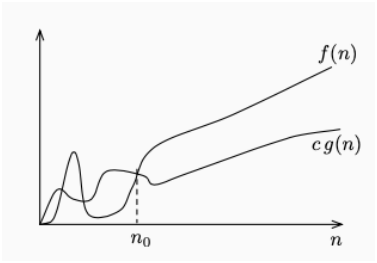


La funzione è definita su n, sono monotone crescenti
La notazione O da un limite superiore a meno di fattori costanti. Esempio: l'algoritmo di ricerca lineare tempo O(|L|) nel caso peggiore

Limite inferioire asintotico per g(n):

$\Omega(g(n)) = \{f(n) | \exists c>0 \exists n_0>0 \forall n \geq n_0.\ 0 \leq c\, g(n) \leq f(n)\ \}$

Diciamo che omega di g è un insieme di funzioni che sono limitate inferiormente in senso asintotico



L'analisi di complessità degli algoritmi permette di classificare i problemi in base alla complessita computazionale degli algoritmi che ci consentono di risolvere il porblema
Dati due problemi A e B, supponiamo che:esista un algoritmo per risolvere A in tempo O(n) nel caso pessimo, il miglior algoritmo per B sia $O(2^n)$ nel caso pessimo. Allora, possiamo concludere che B `e un problema "piu` difficile" di A, B non ammette soluzioni altrettanto efficienti quanto A

Problemi O(log n) (complessit`a logaritmica) →ricerca binaria su liste ordinate
Problemi O(n) (complessit`a lineare) →ricerca lineare o pattern matching esatto o approssimato di stringhe (ricerca di una stringa)

Problemi O(n log n) → ordinamento (non si puo ordinare un insieme di elementi senza ipotesi aggiuntive sulle caratteristiche di questi elementi in un tempo inferiore a una funzione nlogn
Problemi O(n2) (complessit`a quadratica) →allineamento di due sequenze

La classificazione basata sulla notazione asintotica `e una classificazione “fine” In generale, `e utile classificare i problemi sulla base di una classificazione piu` grossolana:

- un problema con complessità polinomiale nel caso pessimo (O(nd) per qualche d) `e considerato “trattabile”
- un problema con complessità almeno esponenziale nel caso pessimo (O(cn) per qualche c) `e considerato “intrattabile”

C’`e una classe di problemi, chiamata NP, per ciascuno dei quali esiste un algoritmo esponenziale.ma non si sa se esistano algoritmi polinomiali
Sia P la classe dei problemi risolvibili in tempo polinomiale Chiaramente, $P \subseteq NP$
Ma, $NP \subseteq P$?(Problema aperto)

Che fare se un problema di interesse biologico `e NP? (Ottenere una soluzione esatta di solito non è possibile)
Si ricorre ad algoritmi che approssimano la soluzione
E` opportuno che l’errore sia quantificabile
Si ricorre ad algoritmi probabilistici la soluzione `e esatta con un certo livello di confidenza

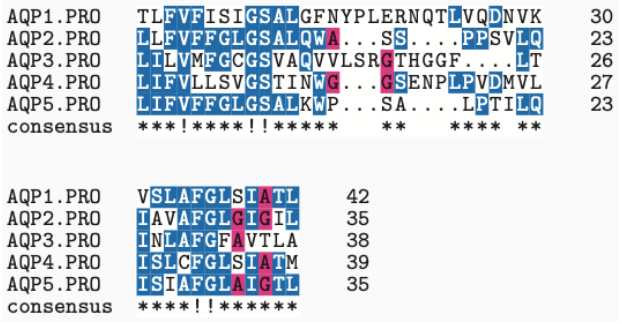
Sequence assembly

- Sequenziamento (metodo shotgun): la sequenza di partenza `e clonata , il DNA `e tagliato in posizioni casuali, i frammenti (700–800 basi) sono sequenziati
- L’ordinamento dei frammenti `e perso
- I frammenti sono sovrapponibili

Come ricostruire l’ordine corretto dei vari pezzi?
Dato un insieme S di stringhe, determinare una stringa s di lunghezza minima tale che ogni $s' \in S$ sia sottostringa di s

```
CATGCACTCAT
   CACTCATCTGCATTTTAATGA
      CTGCAT
          TTAATGATAGC
             ATAGCCAACTACGC
                AACTACGC
CATGCACTCATCTGCATTTTAATGATAGCCAACTACGC
```

l problema precedente `e noto come shortest superstring problem, ed è un problema algoritmico! Senza un algoritmo non sarebbe possibile ricostruire la sequenza del DNA
Il problema `e NP-completo, sono problemi che fanno parte della classe NP e all’interno di questa classe sono i più difficili
Esistono algoritmi polinomiali che forniscono una soluzione approssimata, garantiscono che la stringa risultante abbia lunghezza inferiore a tre volte la lunghezza della soluzione esatta
L’allineamento multiplo può essere risolto con un algoritmo con complessità quadratica
Le righe sono delle proteine rappresentate da una sequenza di amminoacidi e ogni amminoacido crrisponde a una lettera.
L’imput di questo problema è un insieme di stringhe su un alfabeto di 20 lettere e l’ouput è la rappresentazione ossia un allineamento di queste lettere in cui le sequenze vengono disposte una sopra l’altra per confrontarle tra loro
Date k sequenze, determinare l’allineamento ottimo
Il criterio di ottimalit`a si basa su una funzione che associa un punteggio a ciascuna k-upla di amminoacidi
Il punteggio di un allineamento `e la somma dei punteggi associati a ciascuna posizione (colonna)
Gli algoritmi usati in pratica approssimano il problema esatto



Ripiegamento di proteine

La funzione di una proteina `e determinata dalla sua struttura tridimensionale

Problema: data una sequenza di amminoacidi, determinare la risultante conformazione spaziale (protein folding problem).

Problema ancora aperto, si risolve solo con tecniche di laboratorio (molto costose e molto lente)
Problema computazionalmente costoso e inoltre tranne per protine di piccole dimensioni risulta impossbile la conformazione tridimensionale della proteina partendo solo dalla sequenza amminoacidica

Linguaggio di programmazione

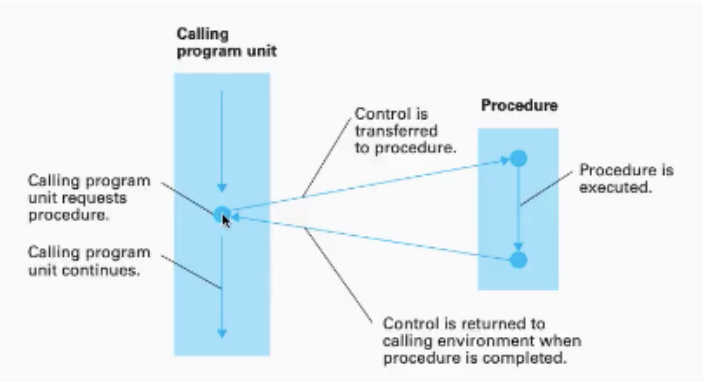
Un calcolatore sa eseguire un numero limitato di operazioni semplici.

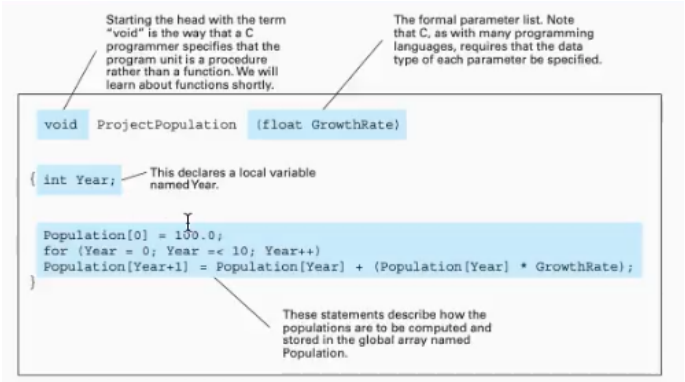
Prima generazione: linguaggio macchina
Seconda generazione: assembly language
Terza generazione: primitive di alto livello e istruzioni machine indepenendet

I linguaggi che vengono utilizzati oggi sono di alto livello perché astraggono dall'architettura del calcolatore e ci permettono di programmare i calcolatori in maniera più intuitiva
La CPU non è in grado di comprendere un linguaggio di alto livello quidni questi linguaggi devono essere tradotti in ultima istanza in istruizioni di linguaggio macchina.
Ci sono due modalita con cui è possibile effetturare questa traduzione:

- compliazione prevede che il codice del programma che si scrive in un linguaggio di alto livello sia tradotto in una sequenza di istruzioni del linguaggio macchina da un softeware chiamato compilatore. Il codice del programma scritto nel linguaggio di alto livello si chiama codice sorgente – il codice sorgente viene tradotto nel linguaggio macchina. Ha il vantaggio che viene tradotto una volta sola da luogo a u nprogramma scritto in linguaggio macchina e questo programma viene eseguito in maniera efficiente, lo svantaggio è che il programma compilato essendo una sequenza di istruzioni in linguaggio macchina è un programma specifico per una data architettura di processore. (puo essere eseguito solo in una specifica architettura di processore)
- la traduzione viene fatta in maniera dinamica mentre il codice sorgente del programma viene letto in software che attua l'interpretazione si chiama "interpreti ". (R linguaggio interpretato). Può essere eseguito su piu tipologie di architetture (il software R fa da interprete ai comandi – ci sono interpreti R diversi per ogni achitettura) uno svantaggio è che visto che la traduzione viene fatta durante l'esecuzione del programma questi linguaggi tendono a essere più lenti

un linguaggio di programmazione deve fornire dei costrutti che direttamente o indirettamente rispecchiano gli algoritmi (istruzioni di assegnamento e istruzioni di controllo – di tipo condizionale e interattivo).
Cio che cambia da linguaggio a linguaggio è la sintassi delle istruzioni. Queste istruzioni, che compongono i blocchi elementari dei linguaggi di programmazione possono essere organizzati in costrutti di piu alto livello che permettono di organizzare un programma e di gestire la complessità di un programma, in particolare è possibile definire le procedure e le funzioni (che sono degli algoritmi) tramite il nome che gli viene assegnato





Esempio di costrutti di un linguaggio di programmazione (R): un algoritmo è utile avere un linguaggio semi formale (psudo-codice per poter specificare l'algoritmo.

```
def cerca(L, v) =
    l <- |L|
    i <- 1

    while (i ≤ l and L[i] ≠ v)
        i <- i + 1
    end while

    if (i ≤ l)
        return true
    else
        return false
    end if
end def
```

Data una specifica come quella a sx, si puo realizzare un programma che implementa l'algoritmo, si passa da una destrizione semi formale (o psuedo codice) a un linguaggio di programmazione bisogna utilizzare la sintassi del linguaggio di programmazione e molto spesso bisogna specificare molte piu cose rispetto a quelle specificate nel algoritmo (in questo caso è gia molto dettagliata)

```
 x<-2 #la variabile è 2 e vado a verficare se x è minore di 3, se è vero decido di moltiplicare x per 2
if (x<3) { #risulta vera in quanto 2<3 quindi possono moltiplicare x pee due
  x <- x*2
 }
quale sarà il valore di x
 x
##[1] 4

 x<-25 #la variabile è 2 e vado a verficare se x è minore di 3, se è vero decido di moltiplicare x per 2
if (x<3) { #risulta fasa in quanto 25>3 quindi non posso moltiplicare x per due
   x <- x*2
 }
 quale sarà il valore di x
 x
##[1] 25

x<-25 #la variabile è 2 e vado a verficare se x è minore di 3, se è vero decido di moltiplicare x per 2
if (x<3) { #risulta falsa in quanto 25>3 quindi non posso moltiplicare x per due
   x <- x*2
 }else {
   x<-0
 }

quale sarà il valore di x
 x
##[1] 0
se if è vera si si puo eseguiure la parte tra {} altrimenti ( quidni se if è falsa e quindi non posso eseguire la
moltiplicaizione ) esegui la parte tra{} che viene dopo la parola else
x
##[1] 10

il ciclo prosegue finche il valore di x viene incrementato fino a 10 dopo di che la condizione x < 10 diventerebbe falsa
è  l'interazione ha termine]

esercizio ricerca lineare
cerca <- function(L, v){
  l <- length(L)
   i <-1
   while(i< 1 && L[i] !=v) {
     i <- i + 1
   }
 if (i < l){
   return(i)
 }
   else{
     return (-1)
   }
 }
##[1] 10
```