



Introduzione al corso di Calcolo Parallelo e Distribuito

a.a. 2003-2004

Almerico Murli

Università degli Studi di Napoli Federico II
&

Istituto di Calcolo e Reti ad Alte Prestazioni del CNR
Sezione di Napoli

A. Murli - Lezioni di Calcolo Parallelo e Distribuito



Introduzione al corso

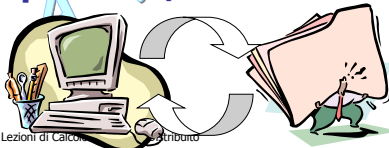
A. Murli

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Obiettivo del corso

Fornire idee, strumenti software
e metodologie alla base della
risoluzione computazionale di
un problema mediante
calcolatore
parallelo/distribuito

Attività di
laboratorio



Lezioni
in aula

A. Murli - Lezioni di Calcolo Parallelo e Distribuito



L'attività di laboratorio

Progetto, sviluppo, analisi e
implementazione
di algoritmi
in grado di sfruttare
le "prestazioni" dei
calcolatori paralleli

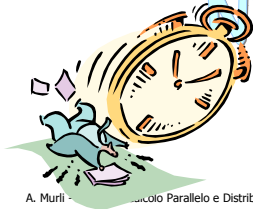
al fine di ...



A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Al fine di

Ridurre il tempo necessario alla
risoluzione computazionale di
un problema



" wall - clock" time

A. Murli - Lezioni di Calcolo Parallelo e Distribuito



A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Il termine "supercalcolatore" si riferisce ad un sistema
che fornisce le prestazioni più elevate
(in quel dato momento).

('80s)

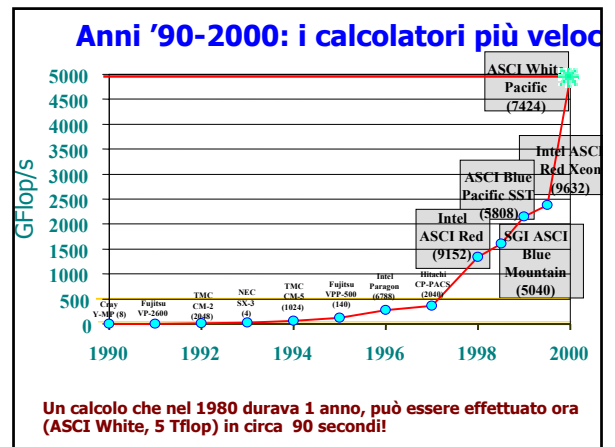
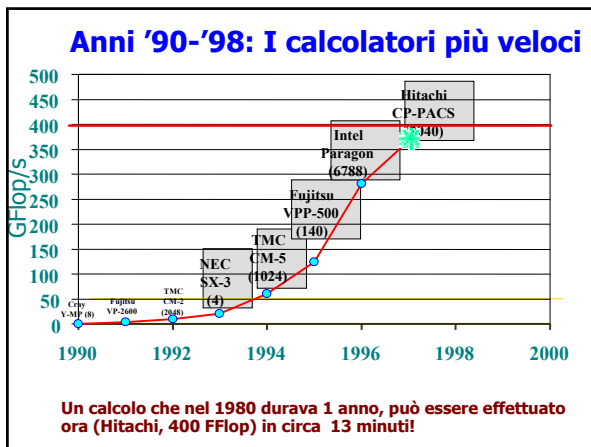
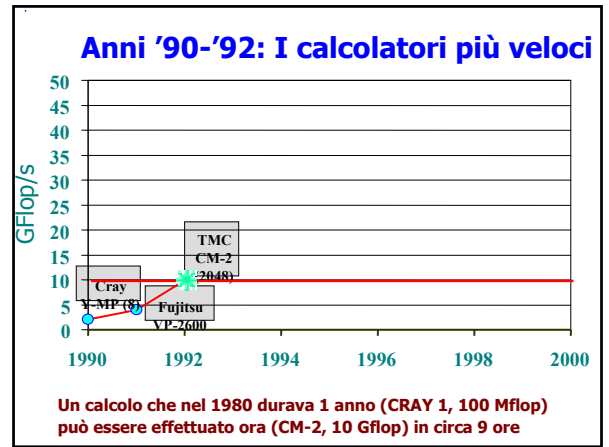
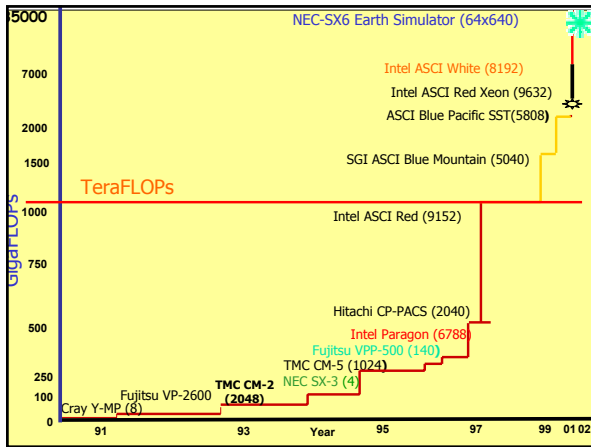
La prestazione è misurata dal tempo necessario per risolvere
una particolare applicazione
(**application-dependent**)

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Dal 1993 ...

- TOP 500 : lista dei calcolatori più veloci nel
mondo
- Rmax : performance misurata dai benchmark
di LINPACK per la risoluzione di $AX=B$

A. Murli - Lezioni di Calcolo Parallelo e Distribuito



Calcolo ad "alte prestazioni"

- Anni '70 – '80:
✓ 10^6 flops (MFLOPs)
Calcolatori sequenziali-scalari
CDC 7600
IBM 360
- Anni '80 – '90:
✓ 10^9 flops (GFLOPs)
Calcolatori vettoriali
CRAY 1
CRAY X-MP
- Anni '90-2000 :
✓ 10^{12} flops (TFLOPs)
Calcolatori a parallelismo massiccio
CRAY T3D
ASCI Red
- Anni 2010 :
✓ 10^{15} flops (PFLOPs)
Griglie computazionali

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Ma ... il calcolo ad alte prestazioni
è davvero necessario?

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Una Ferrari o una
Panda ?



Euro 200.000.



Euro 5.000

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

dipende dall'uso!



In città:
traffico,
parcheggi,...

In autostrada:
velocità, affidabilità,
sicurezza,....



A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Ricerca su INTERNET



Ogni giorno **circa un milione di persone**
interroga su Internet un motore di ricerca

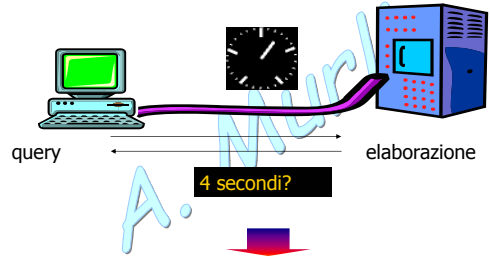
alta vista: SEARCH-

Google™

A. Murli - Lezioni di Calcolo Parallelo e Distribuito



Quanto si e' disposti ad aspettare?

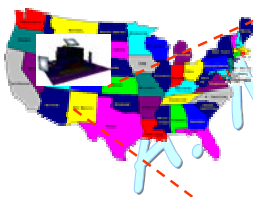


$\tau < \text{alcuni secondi}$

A. Murli - L.

Difesa del Territorio

Nella Guerra del Golfo il Calcolo ad
Alte Prestazioni ha guidato Patriot
contro gli Scud Iracheni



A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Quanto si e' disposti ad aspettare?

Il missile deve essere distrutto **prima** che arrivi
sull'obiettivo



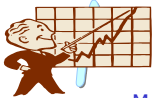
Entro 2 minuti dal lancio del missile nemico!

$\tau < 2 \text{ minuti}$

A. Murli - Lezioni di

Time to Market

I tempi di produzione
sono fortemente influenzati
da quelli di **simulazione**



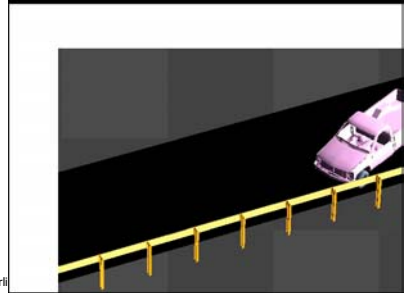
Meno aspettiamo...

...meglio e'!

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Quanto si e' disposti ad aspettare?

Previsione degli effetti di un crash
Simulazioni per l'industria automobilistica



A. Murli

Studio dell'influenza del flusso d'aria sul clima interno del veicolo



A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Car Crash test

> il dominio da simulare viene discretizzato in 20×10^6 "elementi",
per ciascun elemento bisogna determinare 6 parametri (incognite)

IN TOTALE 12×10^7 incognite

> Per simulare sono necessari 15×10^4 passi temporali
per un tempo di simulazione : 12×10^{-2} sec

> in ciascun passo sono richiesti 10^3 flop per incognita

$$12 \times 10^7 \times 15 \times 10^4 \times 10^3 \text{ flop} = \frac{2 \times 10^{16}}{1 \times 10^9 \text{ flop/s}}$$

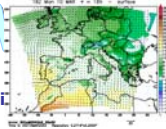
utilizzando un PC...

$$= 2 \times 10^7 \text{ sec} = 5.000 \text{ ore!}$$

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

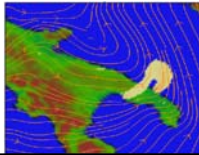
Previsioni metereologiche

- la superficie da monitorare è di **20 milioni di Km²**
- l'altezza sul livello del mare è di **20 Km**
- la risoluzione numerica prevede la discretizzazione dello spazio 3D mediante cubi di lato 100 m
 - in 1 km³ ci vogliono **$10 \times 10 \times 10 = 10^3$ cubi**



$$20 \times 10^6 \text{ km}^2 \times 20 \text{ km} = 4 \times 10^8 \text{ km}^3$$

$$4 \times 10^8 \times 10^3 = 4 \times 10^{11} \text{ cubi}$$



A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Il modello deve fornire le previsioni per i prossimi 2 giorni.

- supponiamo che l'algoritmo effettua, per ogni ora di simulazione

10^2 flop per cubo

- per l'intera superficie sono richiesti **$4 \times 10^{11} \times 10^2$ flop**

$4 \times 10^{11} \times 10^2 \times 48$ flop in 2 giorni

Circa **2×10^{15} operazioni**

$$\frac{2 \times 10^{15} \text{ flop}}{1 \times 10^9 \text{ flop/s}} = 2 \times 10^6 \text{ sec} = 23 \text{ giorni}$$

Utilizzando un PC **1×10^9 flop/s**

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Una previsione effettiva deve fornire una risposta in **al più mezz'ora**

$$30 \text{ minuti} = 1.8 \times 10^3 \text{ sec} = 2 \times 10^{15} \text{ flop}$$

$$= 1 \times 10^{12} \text{ flop/s}$$

1 TFLOPS

33 minuti !

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

In sintesi...aumentare le prestazioni consente di

- Risolvere problemi in **"tempo reale (utile)"**
- Risolvere problemi di **grandi dimensioni** (larga scala)

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Come ridurre i tempi di risposta
(*turnaround time*)
di una simulazione computazionale

?

In generale:

una rappresentazione semplificata del
tempo richiesto dalla risoluzione numerica
di un problema

è:

$$\tau = k \cdot T(n) \cdot \mu$$

$T(N)$ = complessità di tempo dell'algoritmo μ = tempo di esecuzione di 1 op. f.p.

Dipendenza dall'algoritmo

e Distribuito

Dipendenza dal calcolatore

Come ridurre τ ?

$$\tau = k \cdot T(n) \cdot \mu$$

μ = tempo di esecuzione di 1 op. f.p.

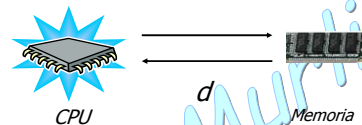
A:

riducendo μ

ovvero

migliorando la tecnologia

Come ridurre μ ?



I dati percorrono la distanza d alla
velocità della luce ($c = 3 \times 10^8$ m/s)

$$d = \mu \cdot c$$

Riprendiamo l'esempio del motore di ricerca



Solo 1 secondo di attesa ?!

$$\tau = T(N) \mu$$

$$1 \text{ sec} = 200 \times 10^9 \mu$$

$$\mu = \frac{1}{2 \cdot 10^{11}} \text{ s}$$

Consideriamo un calcolatore con una potenza di calcolo di 1 Teraflop (10^{12} flop/sec) :

$$d = \mu c$$



Poiché il segnale viaggia alla velocità della luce $c = (3 \times 10^8 \text{ m/s})$ si ha

$$d < \frac{3 \cdot 10^8 \text{ m/s}}{10^{12} \text{ flops}} = 0.3 \text{ mm}$$

1 TERAFLIPS

A: ridurre m?

2002



Notevoli problemi di packaging e raffreddamento

LIMITI TECNOLOGICI

STOP!!

Come ridurre t ?

$$\tau = k \cdot T(n) \cdot \mu$$

B: riducendo $T(n)$
ovvero

Riorganizzando l'algoritmo

B: ridurre $T(n)$

Problema:
Esempio: risoluzione di un sistema diagonale

$$\begin{pmatrix} a_1 & & \\ & \ddots & \\ & & a_n \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Algoritmo "immediato"

$$x_i = y_i / a_i \quad i=1, n$$

$T(n)=n$

Non esiste un algoritmo con complessità inferiore

B: ridurre $T(n)$

E' possibile dimostrare
(teoria della complessità degli algoritmi)
che per alcune classi di problemi
esistono algoritmi con complessità di tempo minima

(algoritmi ottimali)

Esempio:

Problema: prodotto di 2 matrici $n \times n \Rightarrow T(n) = n^3$
Algoritmo ottimale: Strassen ($T(n) = n^{\log_2 7}$)

Progettare un algoritmo 10x più veloce equivale a costruire un calcolatore 10x volte più potente

cioè

Riorganizzare l'algoritmo
per pervenire ad uno con
complessità di tempo
inferiore

Matematica
computazionale

→ tecnologia

"La legge fondamentale della computer science: all'aumentare della potenza di calcolo, aumenta l'importanza dell'efficienza degli algoritmi."

Come ridurre t ?

$$\tau = k \cdot T(n) \cdot \mu$$

μ = tempo di esecuzione di 1 op. f.p.

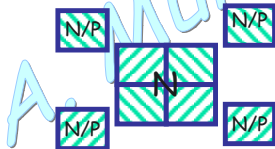
C: ?

B: riducendo $T(n)$
(riorganizzando
l'algoritmo)

A: riducendo μ
(migliorando la
tecnologia)

Soluzione: CALCOLO PARALLELO

Decomporre un problema di dimensione N in P sottoproblemi di dimensione N/P e risolverli **contemporaneamente** su più calcolatori



Sviluppo di nuovi strumenti computazionali

Hardware/ software/ algoritmi

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Esempio: costruzione di una casa



A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Primo tipo di parallelismo

Tecnica della **catena di montaggio (pipeline)**

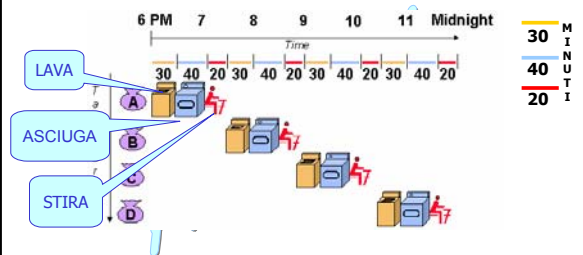


I tre operai **eseguono contemporaneamente** fasi **successive** dello stesso lavoro

PARALLELISMO TEMPORALE

A. Murli

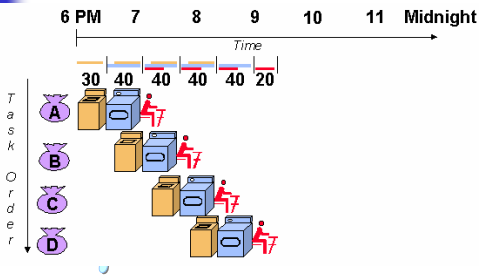
Esempio:



Per completare il lavoro sono necessarie **6 ore!**

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

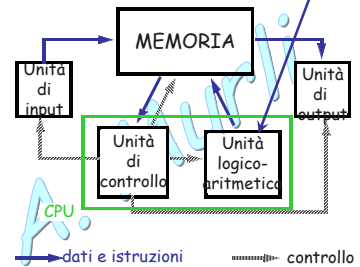
Quanto tempo si risparmia ?



Con una **pipeline** bastano **3 ore** !

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Dove è realizzato il parallelismo temporale in un calcolatore ?



Parallelismo on-chip

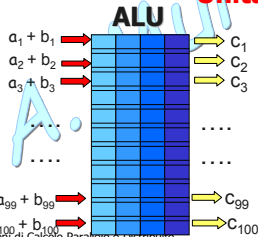
A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Esempio

somma di 100 numeri floating point

$$c_i = a_i + b_i \quad i=1,100$$

Unità tradizionale



A. Murli - Lezioni di Calcolo Parallelo e Distribuito

L'unità funzionale per l'addizione floating point è **divisa in segmenti**

Ciascun segmento è preposto alla esecuzione di una fase dell'operazione, ad esempio Per la somma:

CONFRONTO DEGLI ESPONENTI	FASE 1
SHIFT DELLA MANTISSA	FASE 2
SOMMA DELLE MANTISSE	FASE 3
NORMALIZZAZIONE	FASE 4

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

In particolare...

somma di 2 numeri floating point

0.956 -1.23

Supponiamo che il sistema floating point abbia:

base 10
precisione 3
esp. min. -2
esp. max. 2

In tale sistema i due numeri sono così rappresentati:

$0,956 \times 10^0$

$-0,123 \times 10^1$

L'operazione di addizione f.p. è composta da **4 fasi**:

1. Confrontare gli esponenti

$\text{esp}(+0,956 \times 10^0) < \text{esp}(-0,123 \times 10^1)$

C

2. Shiftare la mantissa

$+0,956 \times 10^0 = 0,095 \times 10^1$

S

3. Addizionare le mantisse

$+0,095 - 0,123 = 0,028$

A

4. Normalizzare il risultato

$-0,028 \times 10^0 = -0,280 \times 10^1$

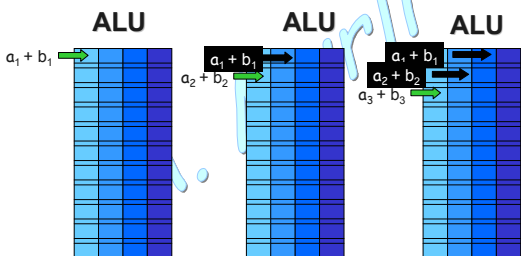
N

Esempio

somma di 100 numeri floating point

$$c_i = a_i + b_i \quad i=1,100$$

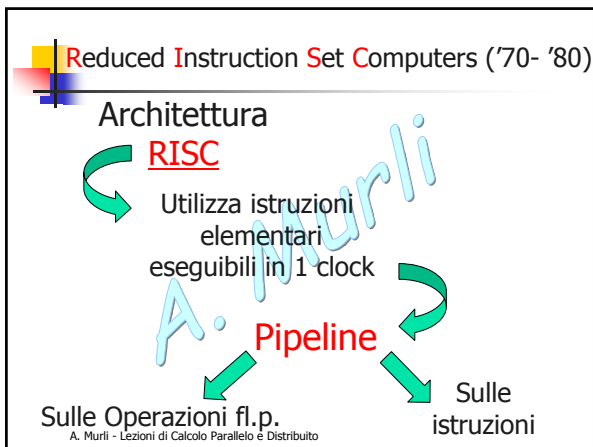
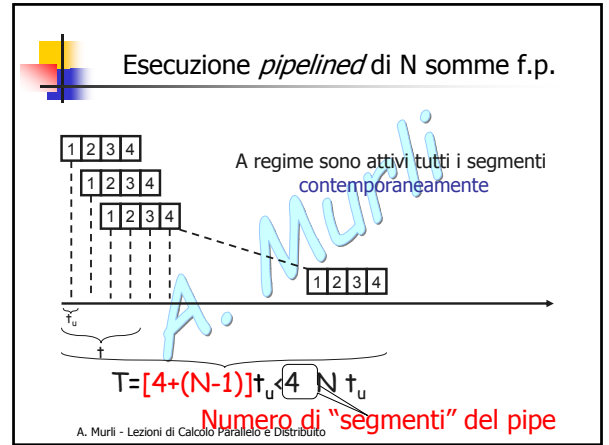
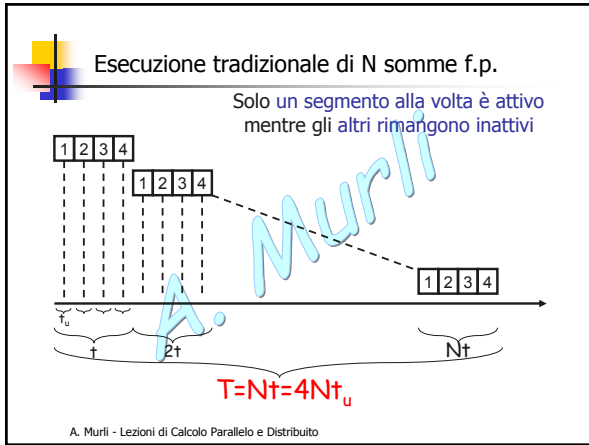
Unità pipelined

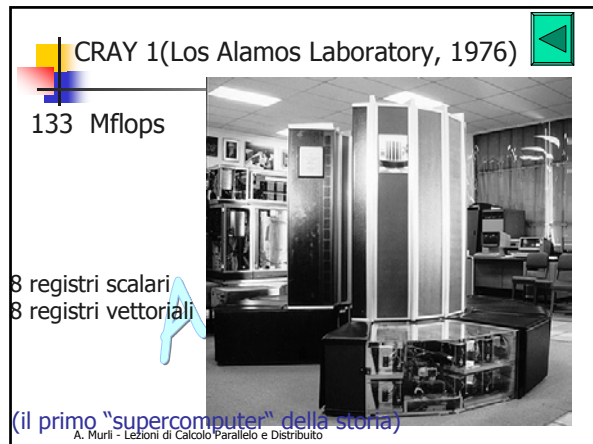
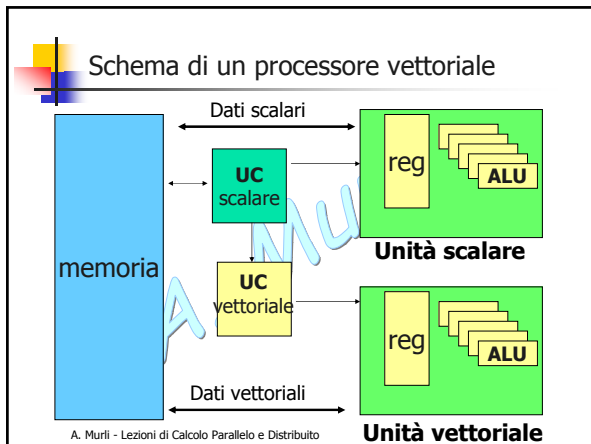


A regime sono attivi tutti i segmenti contemporaneamente

Per ogni clock, 3 segmenti su 4 rimangono **INATTIVI** !

Ciascun segmento può operare **concorrentemente** agli altri





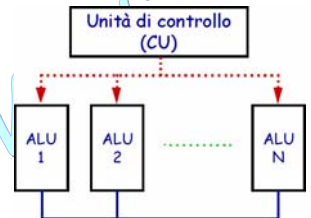
Parallelismo spaziale

Come è realizzato il parallelismo spaziale in un calcolatore ?

Parallelismo a livello di unità funzionale
(più unità aritmetico-logiche)

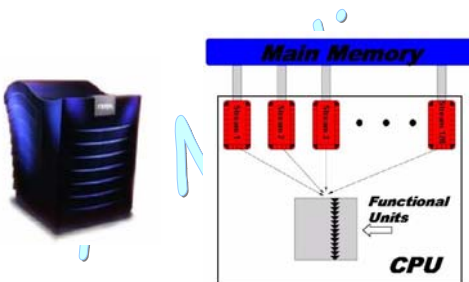
Parallelismo spaziale

Più unità aritmetico-logiche (ALU) operano sotto un comune controllo (CU) eseguendo in parallelo la *stessa istruzione* su *dati diversi*



Calcolatori SIMD
(*Single Instruction Multiple Data*)

The TERA MTA



Terzo tipo di parallelismo



I tre operai eseguono contemporaneamente azioni diverse su parti diverse

PARALLELISMO ASINCRONO

Parallelismo asincrono

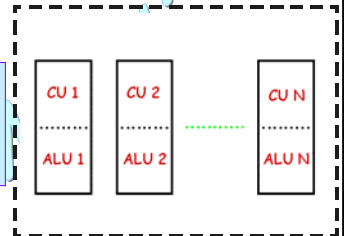
Differenti processori (CPU) cooperano eseguendo istruzioni diverse su dati diversi

Come è realizzato il parallelismo asincrono in un calcolatore ?

Parallelismo a livello di CPU
(più CPU (ALU+CU))

Parallelismo asincrono

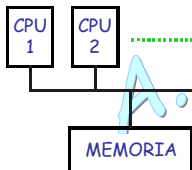
Più CPU (ALU + CU) eseguono in parallelo le *istruzioni diverse* su *dati diversi*



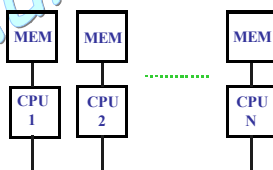
Calcolatori MIMD
(Multiple Instruction Multiple Data)

MIMD

Calcolatori MIMD a memoria *condivisa*
(shared-memory)

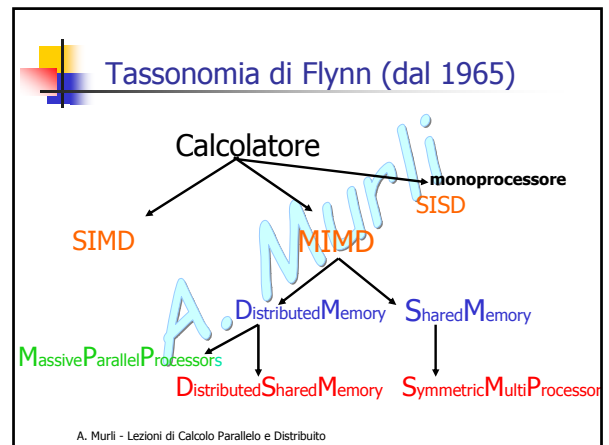
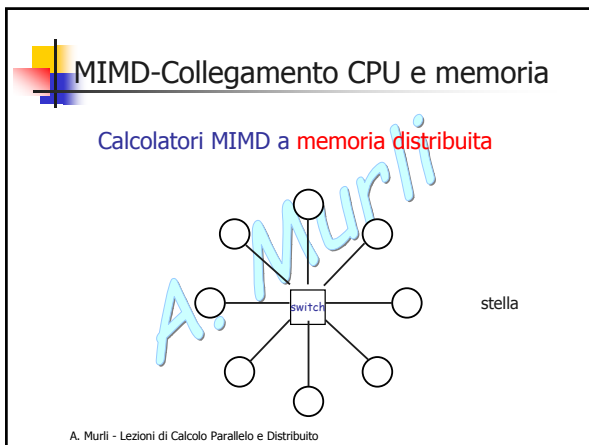
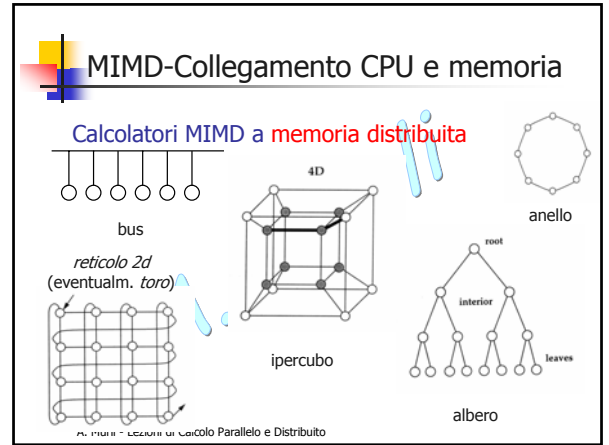
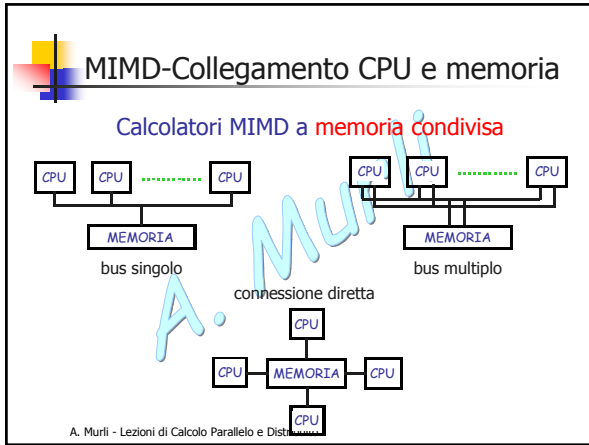


Calcolatori MIMD a memoria *distribuita*
(distributed-memory)



MIMD

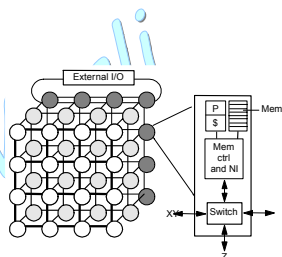
Come sono collegate CPU e memorie in un calcolatore MIMD?



Esempi di calcolatori MIMD-DSM



Cray T3E



A. Muri - Lezioni di Calcolo Parallelo e Distribuito

Esempi di calcolatori MIMD-MPP

IBM RS/6000 SP



Sistema scalabile fino a 512 nodi
Ogni nodo è composto da:
4-16 processori POWER 3 (RISC) a 64 bit
1-64 GB di memoria condivisa
Connessione tra i nodi tramite switch ad alte prestazioni (topologia a stella)

Control workstation

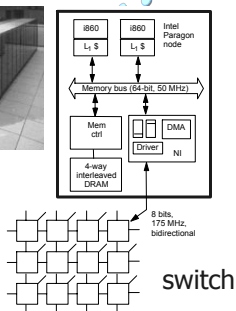


Esempi di calcolatori MIMD-MPP

Intel Paragon



Sandia's Intel Paragon XP/S-based Supercomputer



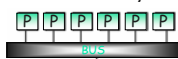
2D grid network
with processing node
attached to every switch

switch

A. Muri

Esempi di calcolatori MIMD-SMP

UMA A memoria condivisa
Uniform memory access



SMP: symmetric multi processors

Tutti i processori

- Hanno lo stesso tempo di accesso alla memoria
- Eseguono copie identiche del s.o.



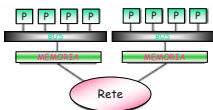
SGI POWER CHALLENGE

Esempi di calcolatori MIMD-DSM

NUMA

A memoria condivisa

Non uniform memory access



SGI ORIGIN 2000

Processori: da 2 a 512

Ibrido:DSM

Memoria Distribuita ma virtualmente condivisa (unico spazio indirizzabile globalmente)

Esempi di calcolatori MIMD-DSM

A memoria fisicamente distribuita ma virtualmente condivisa



SUN E 10000

Processori: da 2 a 64

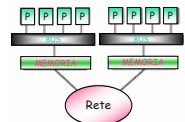
Larghezza di Banda del Sistema: 12GB/Sec

Memoria Massima: 64GB

Architettura a Memoria Condivisa Parallelo e Distribuito

NUMA

Non-uniform memory access



Esempi di calcolatori MIMD-MPP

MPP

A memoria distribuita

Massive Parallel Processor: processori strettamente integrati, che danno un' immagine singola del sistema



ASCI Red

Architettura MPP Intel TeraFLOPS
9632 processori
Topologia a griglia
Performance sostenuta: 2.3 TFlop
Picco di Performance: 3.2 TFlop

A. Muri - Lezioni di Calcolo Parallelo e Distribuito

Esempi di calcolatori MIMD-CLUSTER

A memoria distribuita

Clusters

Computer individuali connessi tramite un software

Cplant (Computational Plant)/Ross

Cluster di 1369 nodi Compaq con processore Alpha

Performance: 706 GFlop
Peak performance: 1.2 TFlop

Realizzato al Sandia Laboratories componendo i singoli elementi



TOP 500 1° posto novembre 2000

A memoria condivisa distribuita



Realizzato nell'ambito dell'Accelerated Strategic Computing Initiative per fornire uno strumento di simulazione sulle armi nucleari in seguito alla moratoria negli USA sui test reali.

Nodi: 512 da 16 processori tipo Power 3 in frame RS/6000 SP

Processori totali: 8192

Performance sostenuta: 7.2 TFlop

Picco di performance: 12.2 TFlop

Memoria totale: 8 TByte

www.top500.org

A. Muri - Lezioni di Calcolo Parallelo e Distribuito

TOP 500 1° posto giugno 2002

A memoria condivisa distribuita



Realizzato in Giappone per fornire uno strumento di simulazione dell'evoluzione sia climatica sia geologica del pianeta, per la previsione di eventi catastrofici.

638 Nodi da 8 processori vettoriali del tipo Nec SX

Processori totali: 5104

Performance sostenuta: 35.6 TFlop

Picco di performance: 40 Tflop

Memoria totale: 10 TByte

A. Muri - Lezioni di Calcolo Parallelo e Distribuito

Esempi di calcolatori MIMD-CLUSTER

A memoria distribuita


Cplant (Computational Plant)/Ross

Cluster di 1369 nodi Compaq con processore Alpha


Performance: 706 GFlop

Peak performance: 1.2 TFlop

Realizzato al Sandia Laboratories

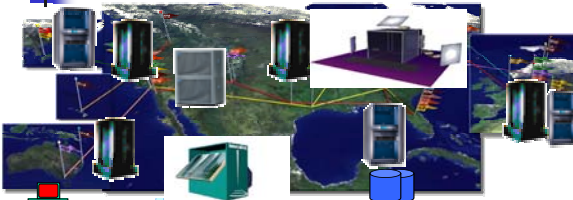


Cluster

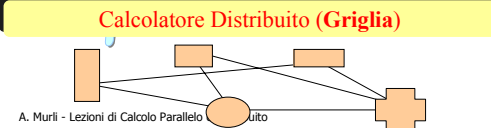


A. Muri - Lezioni di Calcolo Parallelo e Distribuito

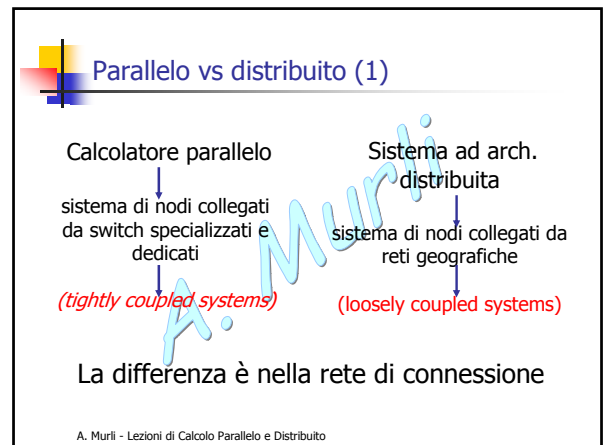
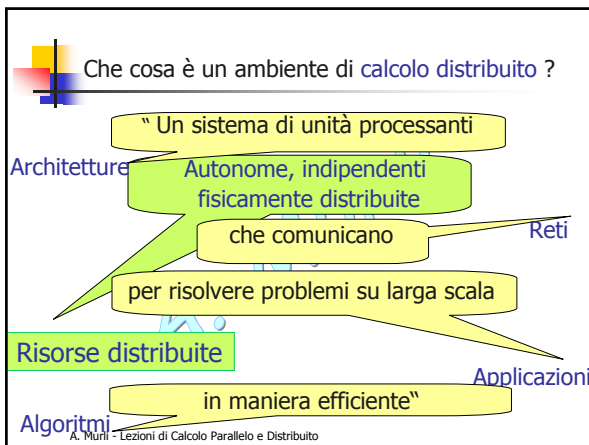
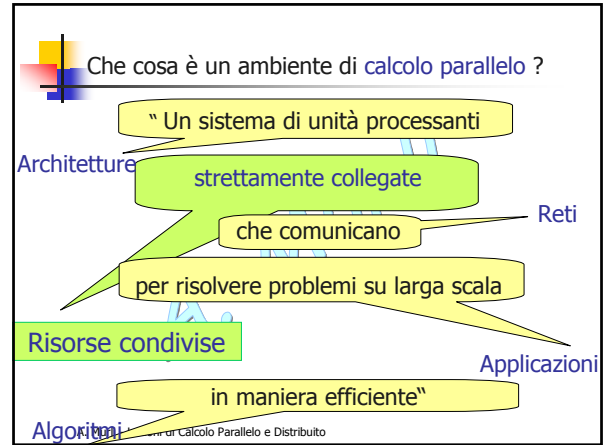
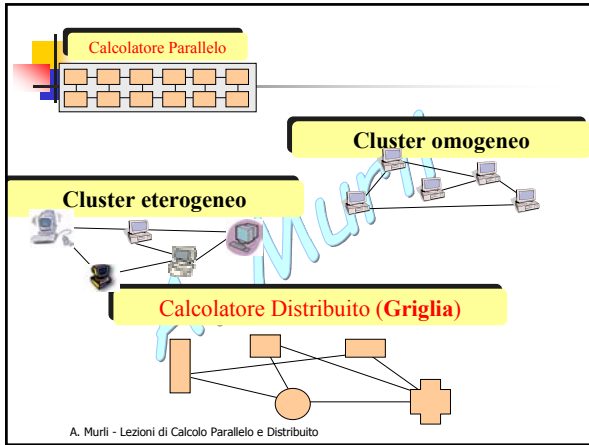
CALCOLATORE DISTRIBUITO



Calcolatore Distribuito (Griglia)



A. Muri - Lezioni di Calcolo Parallelo e Distribuito



Parallelo vs distribuito (2)

Calcolatore parallelo

Principale obiettivo:
Performance

Risorse di calcolo
omogenee

Sistema ad arch.
distribuita

Principale obiettivo:
Risparmio di risorse esistenti

Risorse di calcolo
eterogenee

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Parallelo vs Distribuito

Sistemi
Massicciamente
paralleli

Sistemi
distribuiti

ASCI

Cluster

NOW

Beowulf

Grid

5% overhead

vs

10-20% overhead

Space - shared

vs

time - shared

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Cos' è il calcolo parallelo/distribuito?

"in **parallel computing** we decompose into parts,
in **distributed computing** we assemble parts"

G.J. Fox, IEEE CISE, 2002

"nel **calcolo parallelo** decomponiamo il problema,
nel **calcolo distribuito** assembliamo le risorse"

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Calcolo parallelo... al fine di

Ridurre il tempo necessario alla risoluzione computazionale
di un problema reale



"wall - clock" time

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Calcolo distribuito,al fine di

- Riutilizzare "efficacemente" risorse hardware e software distribuite geograficamente sul territorio



"(ri)uso efficiente delle risorse"

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

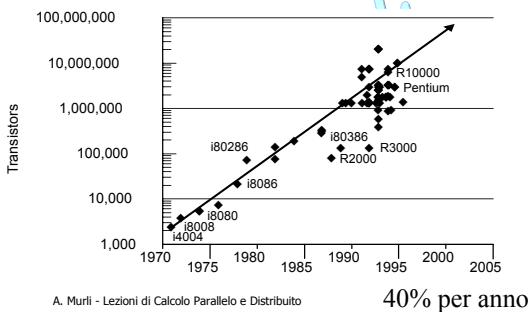
La legge di Moore (1965)

Gordon Moore (co-fondatore di Intel) predisse nel 1965 che la **densità dei transistor** sui chip di semiconduttori sarebbe raddoppiata approssimativamente ogni 18 mesi. Egli stimò che questo sarebbe accaduto almeno fino al 1975



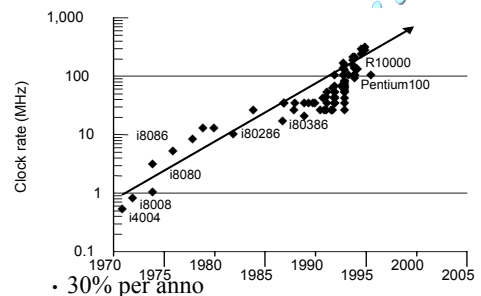
A. Murli - Lezioni di Calcolo Parallelo e Distribuito

In effetti le previsioni erano corrette....

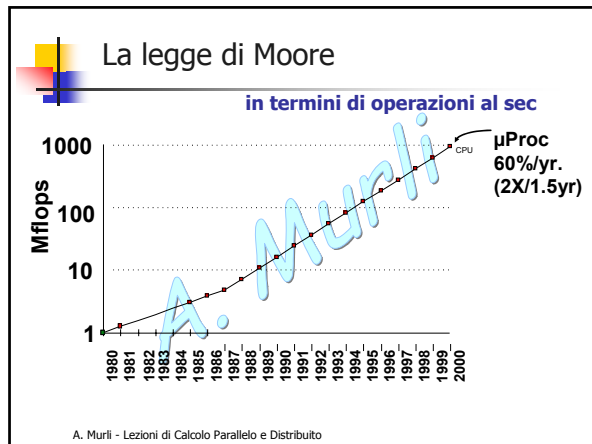
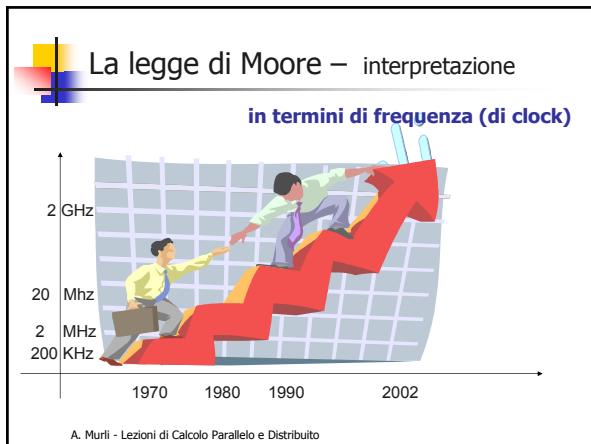


A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Equivalentemente...



A. Murli - Lezioni di Calcolo Parallelo e Distribuito



In realtà, la ri- lettura della legge di Moore in termini di aumento di prestazioni (clock-rate) è possibile solo se si tiene conto del parallelismo intrinseco a livello del processore (parallelismo on chip)

A. Murli

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Ieri

anni '70

Intel 4004, 1971

Frequenza di clock: 1 KHz
2300 transistors.
0.06 MIPS.

anni '80

375 Mhz

The IBM® Power 3®

IBM Power3-II chip

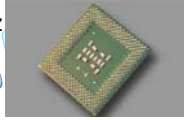
» Power3 - (2xfloatKD flops/sec) (375 Mhz) = 1500 MFLOP/s

A. Murli

A. Murli - Lezioni di Calcolo Parallelo e Distribuito

Oggi

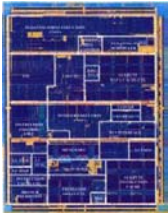
The Intel® Celeron® : 1.80 GHz



The Intel® Pentium® 4: 2.53 GHz

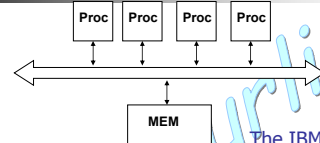


The Intel® Athlon® : 1.80 GHz



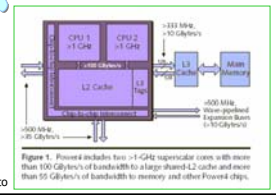
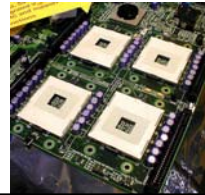
» $\text{Athlon} = (2 \text{ fpu}) * (1500 \text{ cycles}) * (600 \text{ Mhz}) = 1200 \text{ MFLOP/s}$

Oggi SMP microprocessori



The Intel® Pentium Foster®

The IBM® Power 4®



Illo e Distributo

.. E la memoria ?

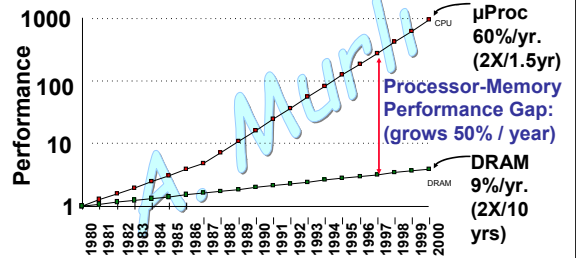
I processori possono eseguire una operazione ogni decimo di nanosecondo

Le DDR RAM possono avere un accesso ogni 3 nanosecondi (!)



Anche le memorie più veloci non riescono a mantenere il processore occupato!

A. Murli - Lezioni di Calcolo Parallelo e Distribuito



A. Murli - Lezioni di Calcolo Parallelo e Distribuito