
ANNO ACCADEMICO 2024/2025

Sistemi di Calcolo Paralleli e Distribuiti

Teoria

Altair's Notes



DIPARTIMENTO DI INFORMATICA

CAPITOLO 1	INTRODUZIONE	PAGINA 5
1.1	Il Corso in Breve...	5
1.2	Introduzione	5
	Perché si fa Parallel Computing? — 5 • Usi Recenti del Parallel Computing — 6	

CAPITOLO 2	TEST2	PAGINA 9
-------------------	--------------	-----------------

Premessa

Licenza

Questi appunti sono rilasciati sotto licenza Creative Commons Attribuzione 4.0 Internazionale (per maggiori informazioni consultare il link: <https://creativecommons.org/version4/>).



Formato utilizzato

Box di "Concetto sbagliato":

Concetto sbagliato 0.1: Testo del concetto sbagliato

Testo contenente il concetto giusto.

Box di "Corollario":

Corollario 0.0.1 Nome del corollario

Testo del corollario. Per corollario si intende una definizione minore, legata a un'altra definizione.

Box di "Definizione":

Definizione 0.0.1: Nome delle definizioni

Testo della definizione.

Box di "Domanda":

Domanda 0.1

Testo della domanda. Le domande sono spesso utilizzate per far riflettere sulle definizioni o sui concetti.

Box di "Esempio":

Esempio 0.0.1 (Nome dell'esempio)

Testo dell'esempio. Gli esempi sono tratti dalle slides del corso.

Box di "Note":

Note:-

Testo della nota. Le note sono spesso utilizzate per chiarire concetti o per dare informazioni aggiuntive.

Box di "Osservazioni":

Osservazioni 0.0.1

Testo delle osservazioni. Le osservazioni sono spesso utilizzate per chiarire concetti o per dare informazioni aggiuntive. A differenza delle note le osservazioni sono più specifiche.

1

Introduzione

1.1 Il Corso in Breve...

Obiettivi:

- Capire i concetti fondamentali del calcolo parallelo:
 - Sistemi paralleli.
 - Metriche, concetti, leggi fondamentali.
 - Approccio strutturato alla programmazione parallela.
 - Programmazione parallela di alto livello.
- Scrivere codice parallelo efficiente su una macchina multicore (shared memory).
- Scrivere codice parallelo efficiente su un cluster di multicore (distributed memory).
- Scrivere codice parallelo efficiente su GPUs.

Percorso:

- Perché le performance sono importanti.
- I principi della programmazione parallela.
- Approccio strutturato alla programmazione parallela.
- Strumenti standard e di ricerca per la programmazione parallela.
- Multicore e cluster.

1.2 Introduzione

1.2.1 Perché si fa Parallel Computing?

Domanda 1.1

Che cos'è il Parallel Computing?

Definizione 1.2.1: Parallel Computing

Il *parallel computing* è la pratica di usare multipli processori per risolvere problemi più velocemente di quanto farebbe un singolo processore. Esso implica la capacità di:

- Identificare ed esporre il parallelismo negli algoritmi e nei sistemi software.
- Comprendere i costi, benefici e limiti di una determinata applicazione parallela. Alcuni codici potrebbero essere più efficienti con un'implementazione sequenziale.

Esempi di macchine parallele:

- *Cluster di Computer*: multipli computer indipendenti connessi con una o più reti ad alta velocità.
- *SMP* (Symmetric Multi-Processor): multipli chip connessi a una gerarchia di shared memory.
- *CMP* (Chip Multi-Processor, Multicore): multiple unità di computazione contenute su un singolo chip.
- *GPUs* (acceleratori): come visto nel corso "Architetture degli Elaboratori II".

Le motivazioni del calcolo parallelo per *aumentare le prestazioni*:

- Molto usate nella scienza e nelle simulazioni ingegneristiche.
- Soprattutto per problemi troppo grandi per essere risolti su un singolo computer.

Note:-

Per lungo tempo, il calcolo parallelo, è stato considerato una nicchia della computer science.

Domanda 1.2

Ma perché si vuole fare Parallel Computing oggi?

- Perché l'intera industria, attorno al 2004-2005, ha iniziato a spostarsi verso la computazione con multipli processori (multicore) in tutti i sistemi (laptops, desktops, workstations, servers, cellulari, dispositivi embedded).
- Attualmente è una parte fondamentale del background di ogni computer scientist.
- Lo spostamento verso i CMP ha certificato la fine della cosiddetta "*Free Lunch Era*".

1.2.2 Usi Recenti del Parallel Computing

- *Big Data Analytics (BDA)*:
 - Dataset molto grandi (Terabyte, Petabyte, Exabyte).
 - Analisi in tempo reale.
- *HPC and/or AI*:
 - Utilizzo massivo di GPUs potenti.
 - HPC con framework ottimizzati per AI (TensorFlow, PyThorch,...).
- Nel 1965, Gordon Moore (co-fondatore di Intel) predisse che la densità dei transistor sarebbe raddoppiata ogni 18 mesi (*legge di Moore*).
- Questo implica che sia possibile integrare sempre più transistor su un singolo chip.
- Nel corso degli anni i chips dei microprocessori sono diventati sempre più piccoli, densi e potenti.

Implicazioni della legge di Moore:

- Il costo dei chips dei processori e delle memorie si è ridotto.
- I processori sono diventati più veloci:
 - Transistor più piccoli e veloci.
 - Migliori microarchitetture, maggiori istruzioni per ciclo di clock.
 - Minori ritardi sulle porte logiche, più frequenza di clock.
 - Cache sempre più grandi per ridurre il collo di bottiglia di von Neumann.
 - Istruzioni SIMD.
- Per cui, per molto tempo, si è pensato che la programmazione parallela fosse uno spreco in quanto sarebbe bastato aspettare un paio di anni.

Limiti del singolo chip:

- Il limite principale dell'aumento di prestazioni è dovuto a:
 - Velocità della luce (insormontabile).
 - Densità di potere (che aumenta ogni anno).
- *Legge scalare di Dennard*: nel 1974, Dennard osserva che l'aumento della densità di parole è costante con il fatto che i transistor diventano sempre più piccoli:
 - Questo fu vero per quasi 30 anni.
 - Tra 2004-2005 si incontrò un muro all'aumento delle prestazioni.

2

Test2

