UNIVERSITÀ DI PARMA
DIPARTIMENTO DI SCIENZE MATEMATICHE, FISICHE ED INFORMATICHE
Corso di Laurea in Informatica

Performance dell'Hardware

Programmazione parallela e HPC - a.a. 2023/2024

Roberto Alfieri

Programmazione Parallela e HPC: sommario

Performance

La performance complessiva di un sistema di calcolo per raggiungere un determinato risultato è data **dal tempo impiegato**, dalla **quantità di risorse informatiche** utilizzate e di **energia consumata**.
Sulle prestazioni incidono le caratteristiche tecnologiche dell'**hardware** utilizzato e la qualità del **software.**
Tutte le componenti hardware incidono sulle performance (unità di processamento CPU e GPU, memoria, storage e network), con pesi diversi in base alle necessità dell'applicazione.
I fattori che incidono sulle prestazioni del **Software** sono gli algoritmi e l'organizzazione dei dati dell'applicazione, l'hardware exploitation ovvero la capacità di sfruttare le risorse hardware disponibili, le caratteristiche del software di base utilizzato (sistema operativo, librerie e compilatori).

Definizioni:

- **Theoretical Peak Performance** è una stima della performance di un componente Hardware (unità di calcolo, memoria, rete, storage) in base alle caratteristiche tecnologiche.
- **Sustained Performance** (Throughput): Prestazioni effettive misurate, di un componente hardware o di un sistema di calcolo tramite l'esecuzione di specifici programmi detti **Benchmark**

Nota: Nelle lista TOP500 *https://www.top500.org/lists/top500/2023/11/*

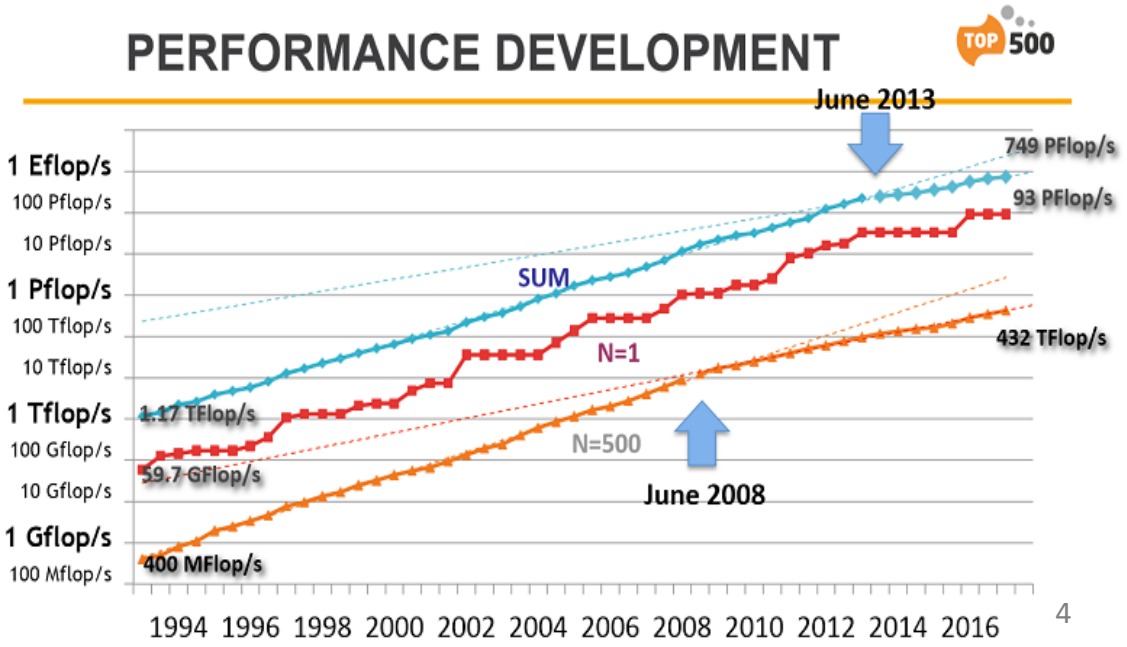
- *Rpeak -> Theoretical peak Performance*
- *Rmax -> Sustained Performance (con il benchmark HPL)*

CPU

La performance di una CPU è data principalmente dalla quantità di operazioni svolte nell'unità di tempo. In passato si consideravano tutte le istruzioni del processore e si misurava in MIPS (milioni di istruzioni eseguire per secondo) ora vengono valutate solo le operazioni in virgola mobile e si misura in MFLOPS o GFLOPS (operazioni in virgola mobile per secondo)

Le prestazioni massime (Theoretical Peak Performance) di un core di calcolo sono determinate dal numero di cicli di clock per secondo (Clock) e dal numero di operazioni f.p. per ciclo di clock (FLOPs/cycle) che il core può eseguire:
FLOPS=Clock x FLOPs/cycle

Il numero di operazioni per ciclo dipende anche dalla **precisione** del dato in virgola mobile che può essere:
- Singola (SP, 32bit)
- Doppia (DP, 64bit)
- Mezza (HP, 16bit)

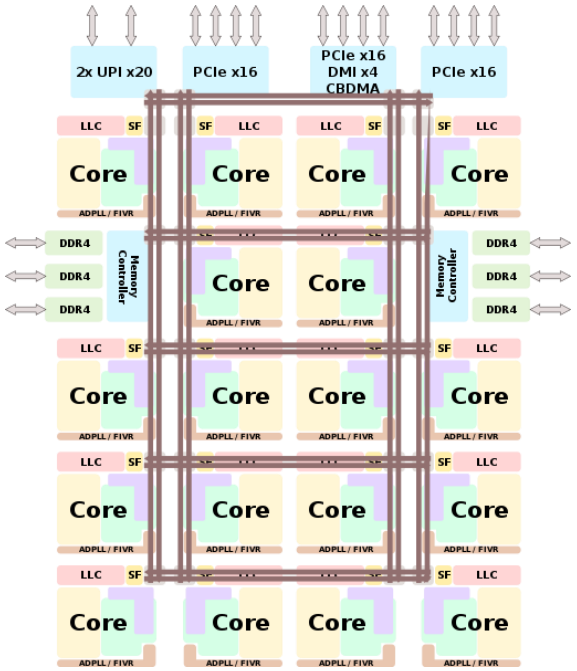


Esempio CPU: Intel Xeon E5-6140

Questo processore è utilizzato all'interno di diversi nodi di calcolo del cluster HPC di Ateneo

[Scheda tecnica Xeon E5-6140]

- Numero core: 18
- Clock: 2.3 GHz (3.7 GHz turbo mode)



Le operazioni Floating Point richiedono diversi cicli di clock ma possono generare un risultato per ciclo di clock grazie all'architettura pipeline
https://it.wikipedia.org/wiki/Pipeline_dati

FLOPs/cycle = 1

Theoretical peak performance:

Clock x FLOPs/cycle = 2.3 -> 3.7 GFLOPs

T.P. Performance per processore:

Da 41.4 (2.3 x 18) GFLOPs a 66.6 (3.7 x 18) GFLOPs

Istruzioni Vettoriali (SIMD) e FMA

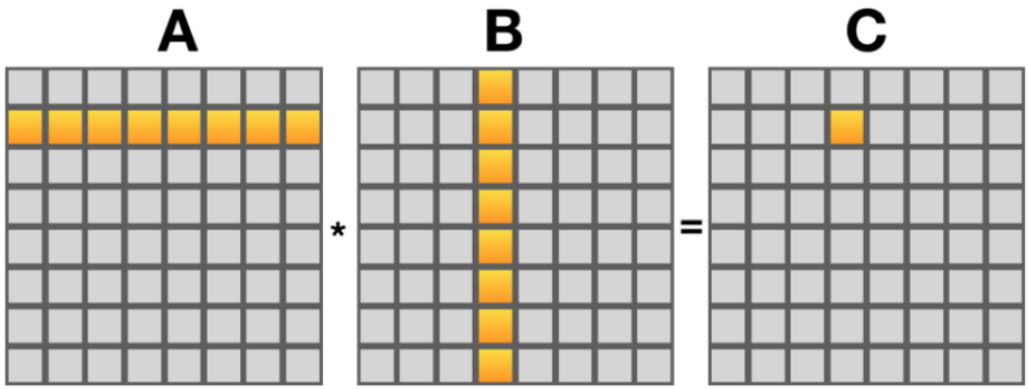
i processori Intel e AMD hanno inserito nei processori un set di istruzioni aggiuntive **SIMD** che si appoggiano su registri dedicati e possono eseguire una istruzione floating point su N dati contemporaneamente, particolarmente adatte quindi per operazioni vettoriali.

Nei processori INTEL la dimensione di questi registri era inizialmente di 128 bit (SSE) per poi aumentare fino agli attuali 512 bit (AVX-512). Un core con AVX-512 può eseguire in un ciclo di clock 16 Flops in singola precisione o 8 in doppia.

Visto che le operazioni ad alto impatto computazionale sono frequentemente delle moltiplicazioni matriciali, è stata aggiunta una ulteriore istruzione dedicata, **FMA (Fused Multiply-Add)** che in un solo ciclo di clock esegue 2 operazioni, somma e moltiplicazione.

*Nell'esempio di figura il calcolo del float C(i,j) richiede **2KFlops -> 2K cicli di clock**.*
*Se utilizziamo istruzioni AVX512 con FMA il calcolo può essere eseguito in **K/16 cicli di clock***

```
for (int i = 0; i < m; i++) {
  for (int j = 0; j < n; j++) {
    for (int p = 0; p < k; p++) {
      C(i, j) += A(i, p) * B(p, j);
    }
  }
}
```



https://gist.github.com/nadavrot/5b35d44e8ba3dd718e595e40184d03f0?permalink_comment_id=3969764

https://hpc-tutorials.llnl.gov/mpi/examples/ser_mm.c

Intel Xeon E5-6140 con SIMD e FMA

Tenendo conto di queste estensioni le prestazioni di picco di un core Intel Xeon E5 sono:

FMA (Fused Multiply-Add) : 2 flops
SIMD AVX-512: 16 flops s.p. - 8 flops d.p.

Peak performance in doppia precisione per core
2.3*2*8 GFlops = 36 Gflops (base)
3.7*2*8 GFlops = 59 Gflops (max)

Peak Perf per processore (18 cores):
1 TFlops in doppia precisione
2 TFlops in singola precisione

Esercizio: il comando lscpu di Linux

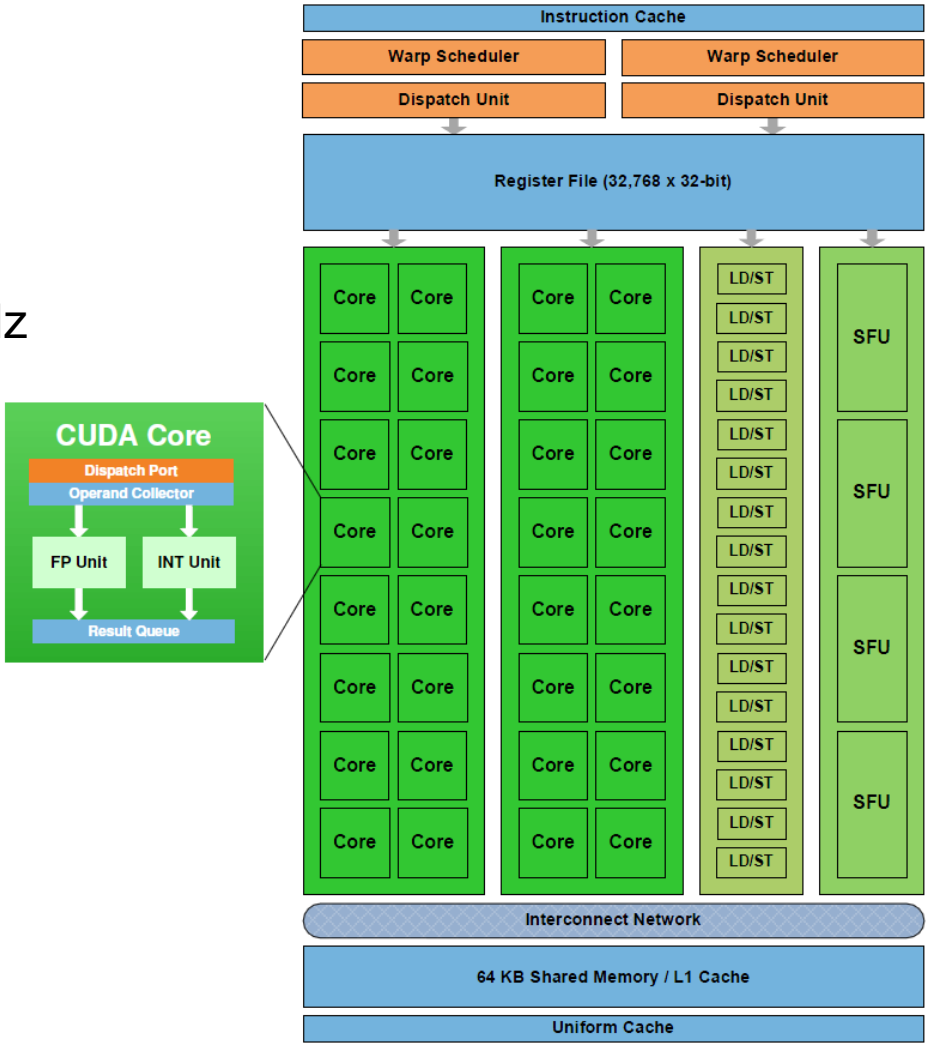
llscpu fornisce le caratteristiche del processore. Esempio:

```
$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                4
On-line CPU(s) list:   0-3
Thread(s) per core:    1
Core(s) per socket:    2
Socket(s):             2
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 79
Model name:            Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz
Stepping:              1
CPU MHz:               2099.998
BogoMIPS:              4199.99
Hypervisor vendor:     VMware
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              20480K
NUMA node0 CPU(s):     0-3
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse
sse2 ss ht syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon nopl xtopology tsc_reliable nonstop_tsc
eagerfpu pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave
avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid_single ssbd ibrs ibpb stibp fsgsbase tsc_adjust
bmi1 avx2 smep bmi2 invpcid rdseed adx smap xsaveopt arat md_clear spec_ctrl intel_stibp flush_l1d
arch_capabilities
```

Esempio GPU: NVIDIA A100

[Scheda tecnica A100]
108 Streaming Multiprocessor (SM)
64 FP32 CUDA core per SM
Totale 6192 CUDA Cores (s.p.) per GPU
Frequenza Base: 765 MHz - Boost: 1410 MHz

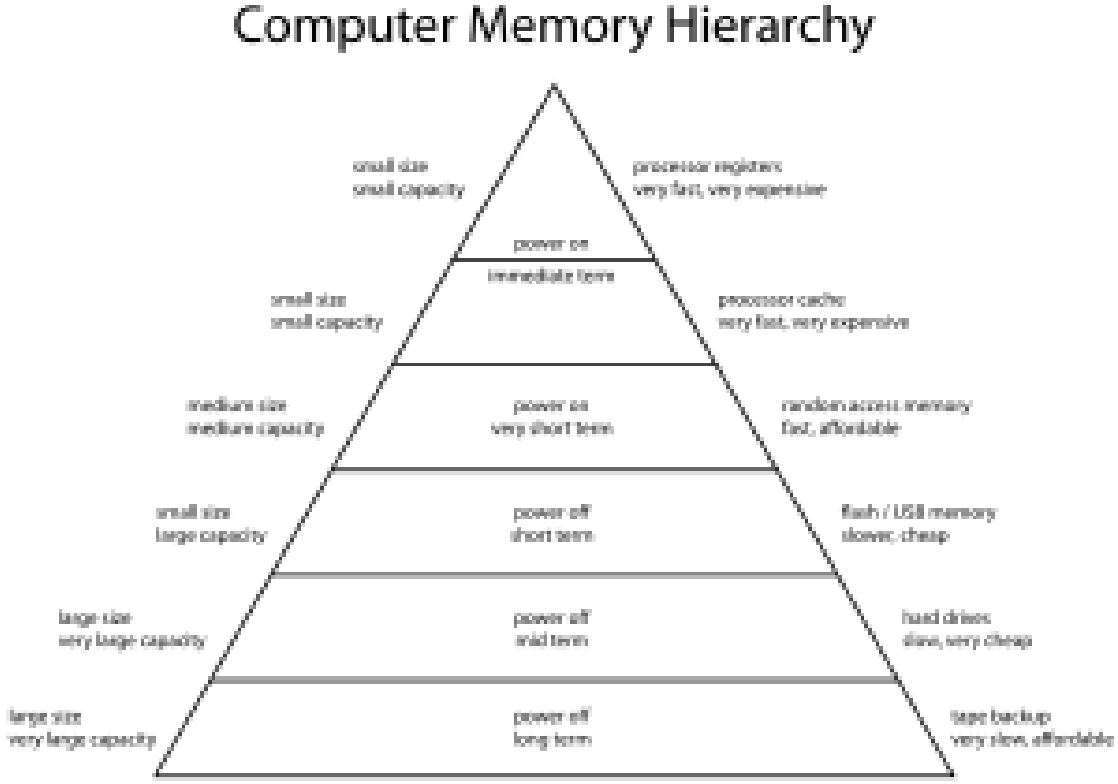
Prestazioni di picco (s.p.)
6192 core * 2 ops (FMA) * 1410 MHz =
19.5 TFlops



Confronto P100 V100 A100:
https://developer.nvidia.com/blog/nvidia-ampere-architecture-in-depth/

Memoria

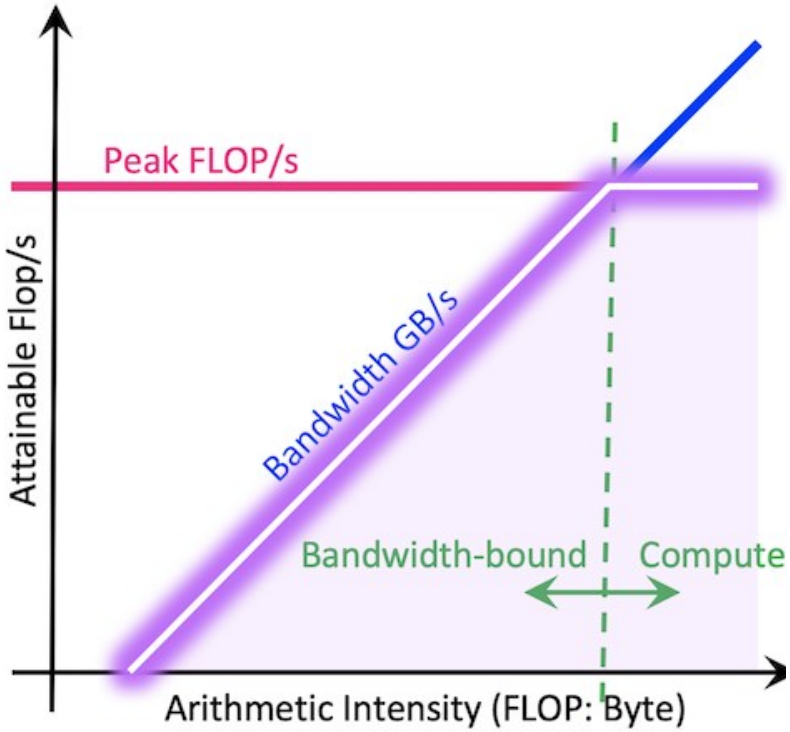
La gerarchia della memoria determina tempi di accesso differenti a seconda della localizzazione del dato.

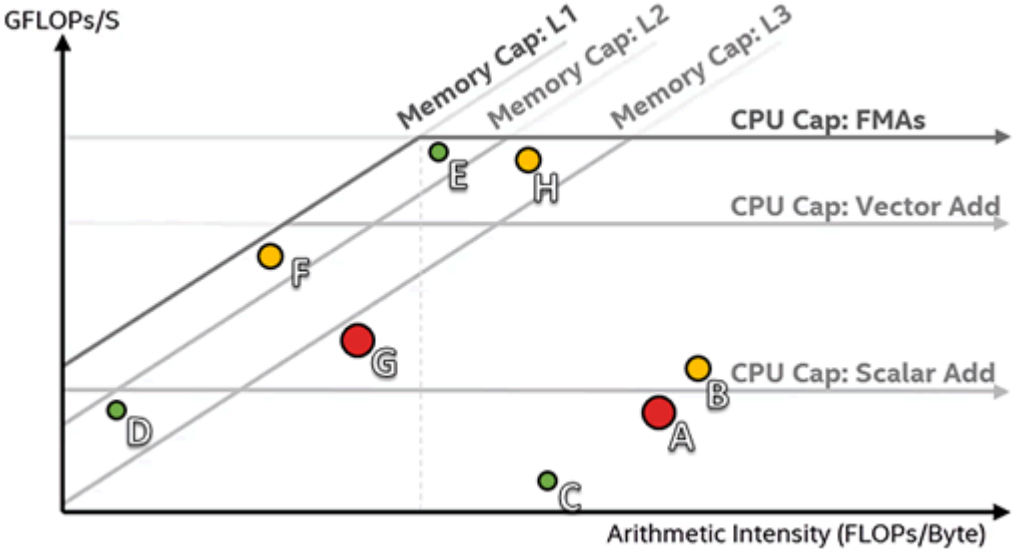


La memoria può diventare un collo di bottiglia nelle prestazioni se non è in grado di fornire dati con il ritmo richiesto dal processore.

RoofLine Analysis

Il modello Roofline consente in maniera intuitiva di stimare le performance di un kernel computazionale mostrando graficamente le limitazioni inerenti CPU/GPU e memoria.
FLOP/s = min(peak FLOP/s, Peak Memory bandwidth x Aritmetic Intensity)
Dove Arithmetic Intensity = FLOPS/ Bytes





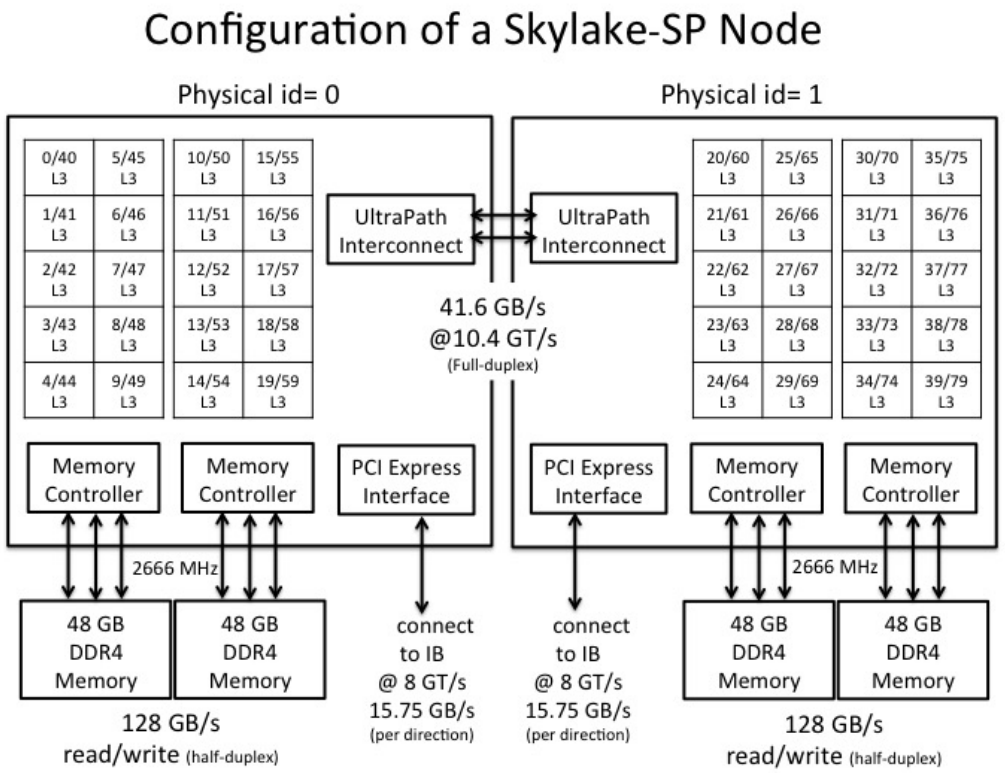
https://www.intel.com/content/www/us/en/docs/advisor/tutorial-roofline/2021-1/run-a-roofline-analysis.html

https://docs.nersc.gov/tools/performance/roofline/

Esempio memoria nella CPU Xeon E5-6140

[Skylake Processors]

Cache L1	32+32KB (per core)
Cache L2	1 MB (per core)
Cache L3	24,75 MB (per socket)
RAM DDR	384 GB (per node)

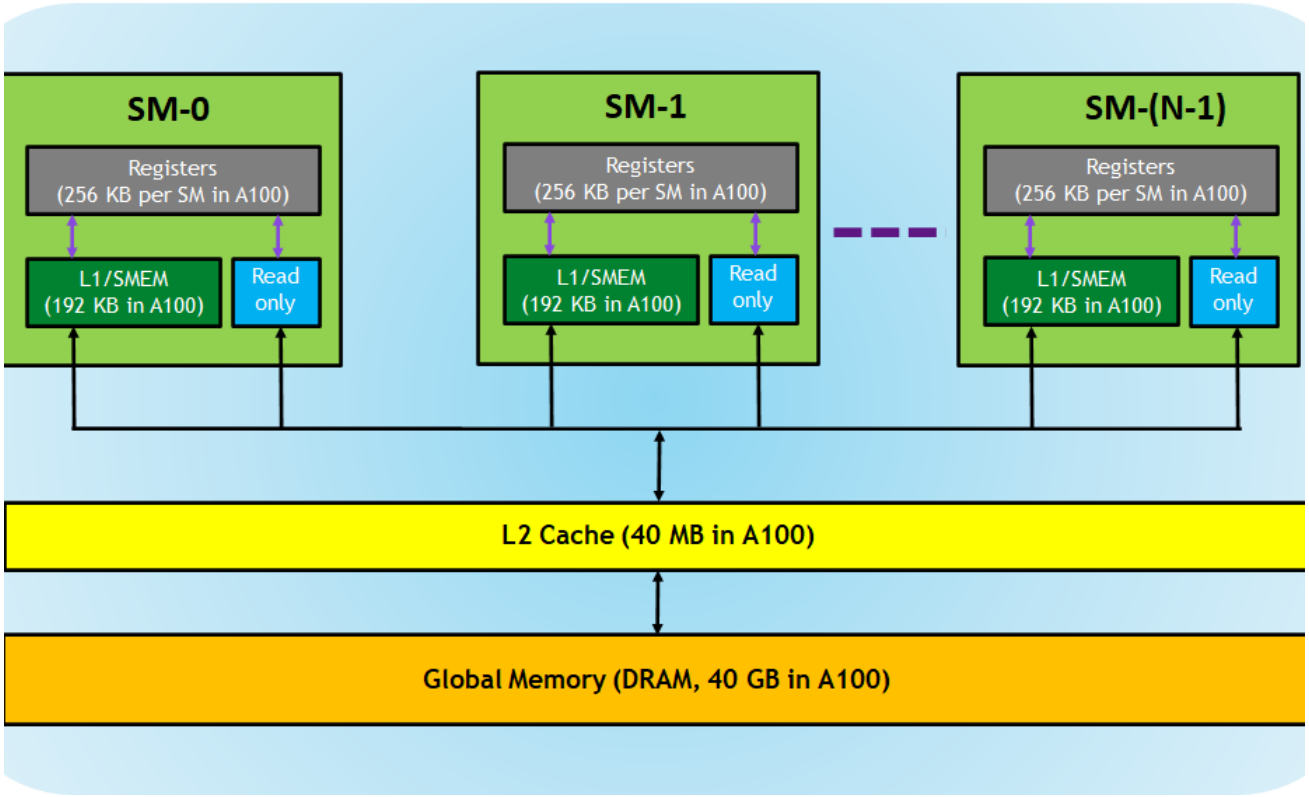


Esempio memoria nella GPU NVIDIA A100

La GPU possiede una gerarchia interna di memoria, a cui si aggiunge la comunicazione con l'host.

Esempio NVIDIA A100

Registers	256 KB/SM	visibile solo al sigolo thread
L1/ShareMem	192 KB/SM	può essere usata come L1 o come shared mem. condivisa per blocco.
ReadOnly		cache delle istruzioni, constant memory, texture memory
L2 Cache	40 MB	condivisa tra tutti i SM
DRAM	40 GB	



Storage

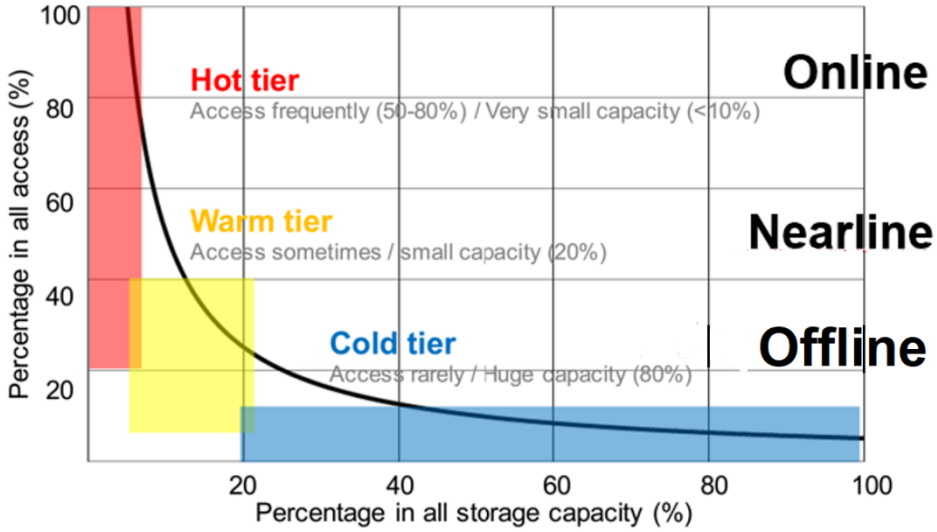
Nei sistemi HPC esistono diverse tipologie di esigenze per lo storage dei dati e tecnologie per la loro memorizzazione

Hot tier: dati usati frequentemente, richiedono alte prestazioni e capacità limitata

Warm tier: dati usati saltuariamente

Cold tier: dati di archiviazione, usati raramente. Richiedono alta capacità

Esigenze



Tecnologie

Tecnologie storage Esempi	Capacità TB	Prestazioni MB/s	Costo
SSD	7.6	600/800	
HDD	20	280	
TAPE Cartridge	30	3.6TB/hour	

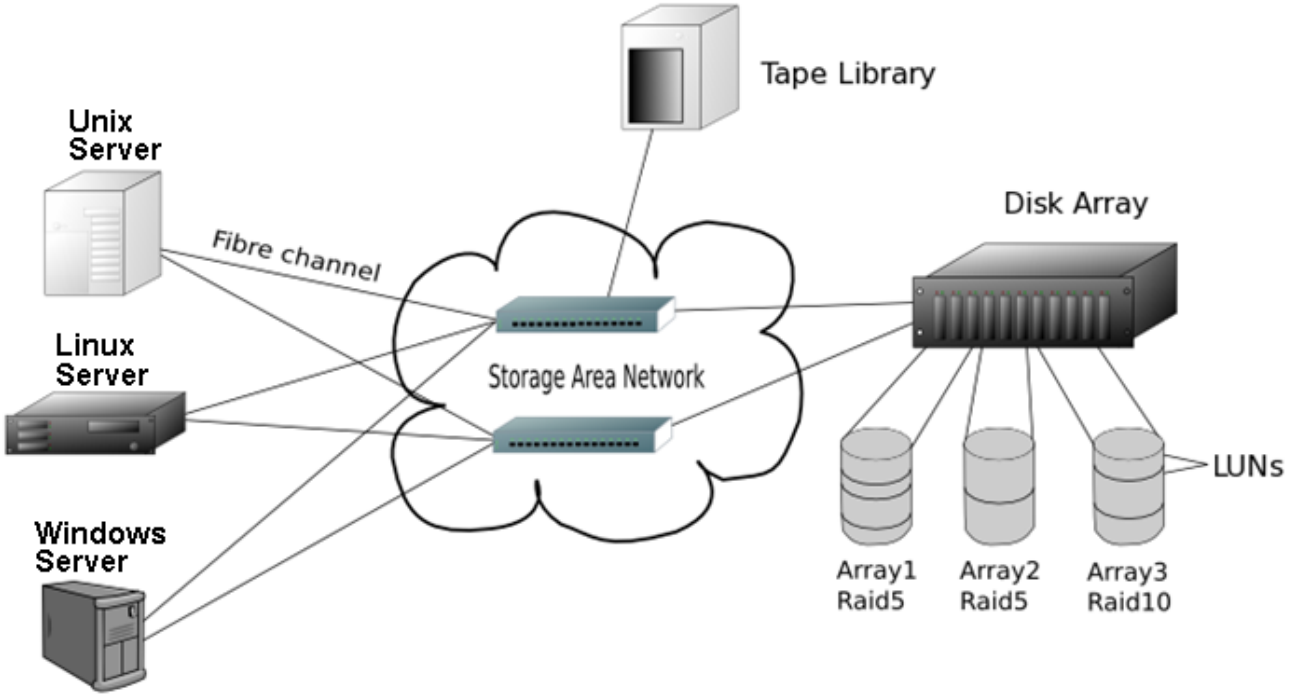
valori tipici

SAN (Storage Area Network)

Una SAN è una rete ad alta velocità di trasmissione costituita esclusivamente da dispositivi di memorizzazione di massa, in alcuni casi anche di tipi e tecnologie differenti. Il suo scopo è quello di rendere tali risorse di immagazzinamento (storage) disponibili per qualsiasi computer connesso ad essa. (wikipedia)

Vantaggi
- prestazioni
- Scalabilità
- Ridondanza

Svantaggi
- gestione complessa
- costo



File System per cluster HPC

Un file system per un cluster HPC deve avere caratteristiche avanzate quali:

Shared filesystem: Separazione dati e metadati.
Gestione centralizzata dei metadati.
Diversi nodi hanno accesso diretto ai dischi condivisi.

Clustered filesystem: tutti i dischi vengono utilizzati contemporaneamente da tutti i nodi

Parallel filesystem: il singolo file viene suddiviso in blocchi che vengono distribuiti su tutti i dischi del file system (striping)

Byte range locking: accesso concomitante di più utenti allo stesso file

Tiering: permette di definire gerarchie di storage con diverse prestazioni

I file system più diffusi per cluster HPC sono

- GPFS (IBM Spectrum Scale) https://en.wikipedia.org/wiki/GPFS
- Lustre https://en.wikipedia.org/wiki/Lustre_(file_system)

Vedi IBM-Spectrum-Scale-Concepts-and-features.pdf nel materiale didattico.

High Speed Networks

	Bandwidth (GB/s)	Latency (microsec.)	Costo scheda (K€)
Intel OmniPath Infiniband EDR	12	1	1
10GbEthernet	0.9	13	0.1
GbEthernet	0.1	47	0.03

valori tipici, costi indicativi

Referenze:

- https://www.hpcadvisorycouncil.com/pdf/IB_and_10GigE_in_HPC.pdf

Case study: Leonardo (CINECA)

Riferimenti: https://leonardo-supercomputer.cineca.eu/it/leonardo-hpc-system/

3456 booster nodes (BullSequana X2135)
Ogni nodo:
- 1 x CPU Intel Xeon 8358 32 core, 2,6 GHz
- 512 (8 x 64) GB RAM DDR4 3200 MHz
- 4x Nvidia custom Ampere GPU 64GB HBM2

89,4 TFLOPs picco per nodo, 240 PFLOPs totali

1536 data-centric nodes (BullSequana X2140)
Ogni nodo:
- 2x Intel Sapphire Rapids, 56 core, 4.8 GHz
- 512 (16 x 32) GB RAM DDR5 4800 MHz

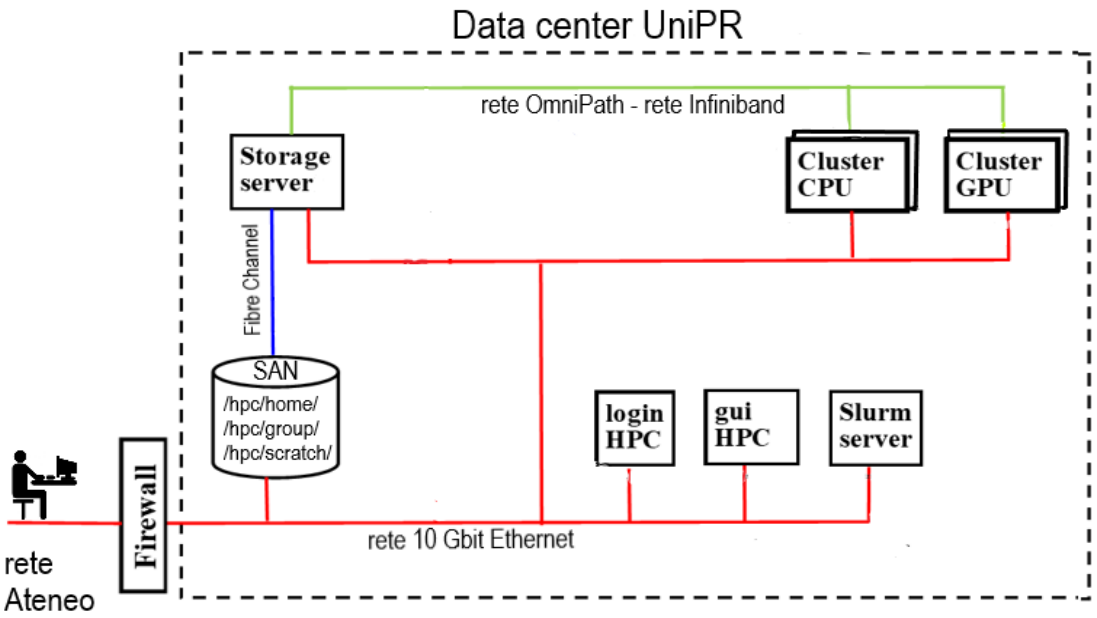
9 PFLOPs totali

Network:
- rete **Nvidia Mellanox, con Dragon Fl+**, capace di una **larghezza di banda massima di 200Gbit/s** tra ogni coppia di nodi.

Storage:
- Fast tier: 5.6 PetaByte Tecnologia: Full Flash Speed: 1400 GB/s r/w
- Capacity Tier: 106 PetaByte Tecnologia: NVMe and HDD Speed: 744 GB/s r – 620 GB/s w

Case study: cluster HPC.unipr.it

Riferimenti: [HPC.unipr.it User Guide]



Nodi Cluster CPU

22 DualSocket Broadwell:	11 Tflops
18 DualSocket AMD:	9 Tflops
1 QuadSocket Broadwell	1 Tflops
8 QuadSocket Slylake:	22 Tflops
4 Intel Knight Landing:	6 Tflops

51 TFlops d.p.

Nodi con GPU

12 NVIDIA P100:	56 Tflops
2 NVIDIA V100:	14 Tflops
14 NVIDIA A100:	135 Tflops

205 Tflops d.p.

NETWORK

OmniPath: Band: 100Gb/s Lat: 100ns

STORAGE GPFS

/hpc/home	20 TB	warm tier	dischi HDD	home directory personale
/hpc/group	50 TB	warm tier	dischi HDD	home directory di gruppo
/hpc/scatch	46 TB	hot tier	dischi HDD+SSD	programmi I/O bound in esecuzione
/hpc/archive	176 TB	cold tier	dischi nearline	archivio dati usati saltuariamente

Benchmarks

Con il termine benchmark si intende un insieme di test software volti a fornire una misura delle prestazioni reali (sustained performance) di un computer per quanto riguarda diverse operazioni.

SPEC (Standard Performance Evaluation Corporation) è una organizzazione no-profit che produce e mantiene performance benchmark per computers (Wikipedia)
Il Benchmark più recente per le CPU è SPEC CPU2017
https://www.spec.org/cpu2017/

I **Benchmark LINPACK** sono utilizzati per misurare le prestazioni dei computer nelle operazioni in virgola mobile. LINPACK è una libreria software sviluppata per eseguire operazioni di algebra lineare. Vedi Wikipedia
HPL - **High Performance Linpack**, è una versione portabile del Benchmark LINPACK che viene utilizzata per stilare la classifica TOP500.

Tempi di calcolo

Quando sviluppiamo un nostro programma abbiamo bisogno di strumenti che ci aiutano a valutarne le performance.

In un programma C la routine **clock_gettime()** consente di determinare i tempi di esecuzione all'interno di un programma.
E' possibile determinare il wall clock time (CLOCK_REALTIME) oppure il tempo di utilizzo della CPU (CLOCK_PROCESS_CPUTIME_ID)

riferimenti: https://people.cs.rutgers.edu/~pxk/416/notes/c-tutorials/gettime.html

Routine per determinare il tempo di esecuzione all'interno di un programma sono fornite anche dalle librerie parallele:

- openMP: omp_get_wtime() ritorna il wall clock time in secondi.
- MPI: MPI_Wtime() ritorna il wall clock time in secondi.
- CUDA: NVIDIA fornisce diverse routine per la misura delle performance sulla GPU
  vedi: https://developer.nvidia.com/blog/how-implement-performance-metrics-cuda-cc/

Post-processing

Per il post-processing dei dati di performance conviene utilizzare un linguaggio di scripting come Python.
Python fornisce diverse librerie ad alto livello di astrazione per come pandas e matplotilib.

pandas https://pandas.pydata.org/ è una libreria software scritta per il linguaggio di programmazione Python per la manipolazione e l'analisi dei dati.

matplotlib https://matplotlib.org/ è una libreria per la creazione di grafici per il linguaggio di programmazione Python progettata per assomigliare a quella di MATLAB.

Riferimenti: https://ourcodingclub.github.io/tutorials/pandas-python-intro/

Profiler

La profilazione è una forma di analisi dettagliata del programma che ne misura la complessità spaziale o temporale senza la necessità di modifica del codice sorgente.

gprof è il profiler del progetto GNU.
Per utilizzarlo occorre compilare con l'opzione -pg
Al momento dell'esecuzione viene generato il file gmon.out che potrà poi essere analizzato con il comando gprof
Riferimenti: https://users.cs.duke.edu/~ola/courses/programming/gprof.html

Esistono tools specifici per la profilazione di programmi openMP e MPI:
https://events.prace-ri.eu/event/1049/sessions/3349/attachments/1332/2384/OpenMP-MPI_profiling_new.pdf

NVIDIA fornisce il profiler **nvprof** per l'analisi dei programmi CUDA
https://events.prace-ri.eu/event/978/sessions/3044/attachments/1136/1863/profiling.pdf