Il progetto HPC
dell'Università e dell'INFN di Parma

Roberto Alfieri

Outline

Il progetto HPC:
Motivazioni, tempi, organizzazione

Il Nuovo Cluster:
caratteristiche tecniche dell'hardware
valutazioni preliminari delle prestazioni
configurazioni software

Applicazioni scientifiche:
alcuni Case Study
esperienze di vettorizzazione

LE TAPPE DEL PROGETTO

Nel 2015 l'Ateneo riorganizza i servizi informatici:

- Centralizzazione di tutto il personale tecnico informatico
- Centralizzazione (in corso) di tutte le risorse in un nuovo Data Center di Ateneo
- Convenzione INFN- Dip. Fisica → INFN-Ateneo

Nuovo servizio centralizzato per il calcolo scientifico?

4 Giugno 2015: "WorkShop sul Calcolo Scientifico in Ateneo"

20 Gennaio 2016: Presentazione del Progetto al Rettore.

7 Novembre 2016: Pubblicazione della gara per un nuovo cluster

Aprile – Maggio 2017: Installazione del cluster, **T**uning e Collaudo

Fine Maggio - Giugno 2017: Pre-produzione

Gruppi di ricerca

Attivi sul calcolo e interessati al progetto:

Dipartimento	**Gruppi di ricerca**
Scienze Chimiche, della Vita e della Sostenibilità Ambientale	8
Scienze Matematiche, Fisiche e Informatiche	9
Scienze degli Alimenti e del Farmaco	3
Medicina e Chirurgia	2
Scienze Economiche ed Aziendali	1
Ingegneria ed Architettura	18

Organizzazione del progetto

Modello organizzativo:

- **Ateneo** fornisce il **personale**, il **data center** , un **budget annuale per spese ordinarie** un **budget iniziale per l'avvio del progetto**
- **INFN** contribuisce integrando il proprio cluster di calcolo e contribuisce all'infrastruttura (storage e virtualizzazione)
- **I singoli gruppi di ricerca** possono contribuire con risorse nuove (calcolo e storage) che verranno gestite in modo integrato, con accesso prioritario.

Comitato Scientifico: 1 rappresentante per Dipartimento scientifico + 1 INFN

Obiettivi:

- Incrementare e ottimizzare l'uso delle **risorse disponibili**
- Favorire la **crescita delle competenze** attraverso sinergie tra gruppi di ricerca, didattica, sperimentazione

Il nuovo cluster

Survey tra i gruppi di ricerca

Tipologia di Nodo	**Gruppi di ricerca**
CPU	23
GPU	18
XEON PHI	10

Cluster **eterogeneo** composto da piccoli cluster ma con tecnologie innovative
Broadwell, KNL, GPU NVIDIA P100

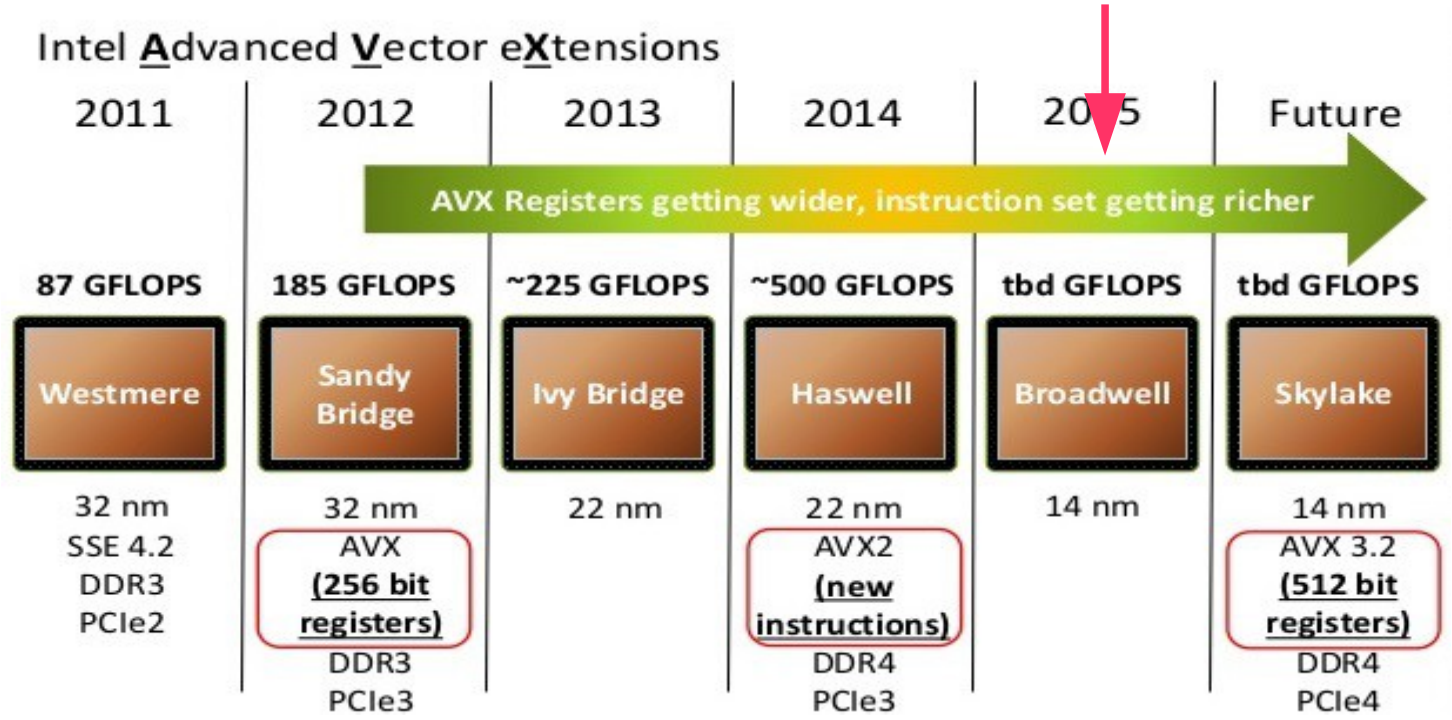
In **comune**
l'autenticazione, il sistema di code, lo storage e la rete veloce OmniPath

Ogni cluster può **scalare** integrando nuovi nodi da gruppi / progetti di ricerca.

Cluster Broadwell (BDW)

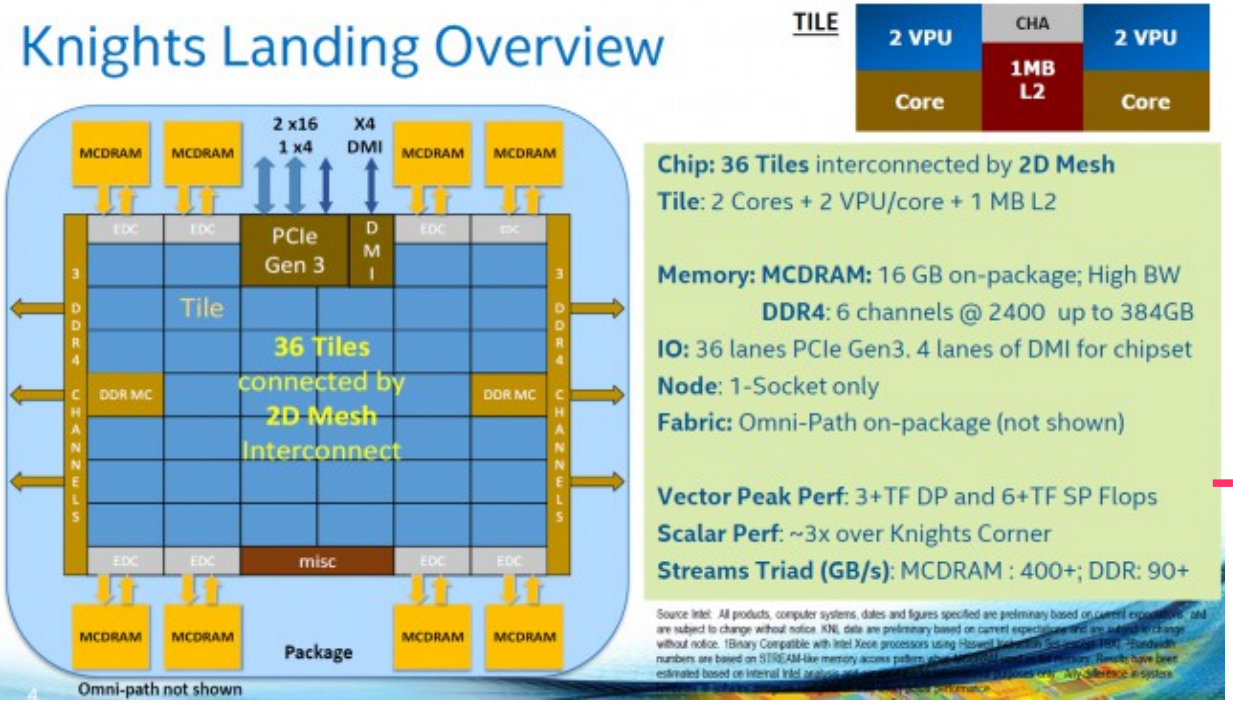
8 nodi con 2x Intel Xeon E5-2683v4 128 GB ram.
1 nodo con 2x Intel Xeon E5-2683v4 **1024 GB ram.**

Performance 1 nodo: 2 *16 (core) *2,1 (GHz) *16 (AVX2) = **1 Tflops dp peak**



Cluster Xeon PHI Knight Landing (KNL)

4 nodi con Intel Xeon PHI 7250, 192 GB ram
Performance per nodo: 68 (cores) * 1.4 (Ghz) * 32 (AVX512) = **3 TFlops peak**



Mod.	GHz	core	OPA integ
7210	1.3	64	no
7210F	1.3	64	si
7250	1.4	68	no
7250F	1.4	68	si
7290	1.5	72	no
7290F	1.5	72	si

Cluster GPU

2 nodi con 2 Intel Xeon E5-2683v4 , 128 GB ram, **5 GPU Pascal P100** 12GB (PCIe)
Performance 1 nodo: 1 (Broadwell) + 5 x 4.7 (P100) = 24,5 TFlops peak

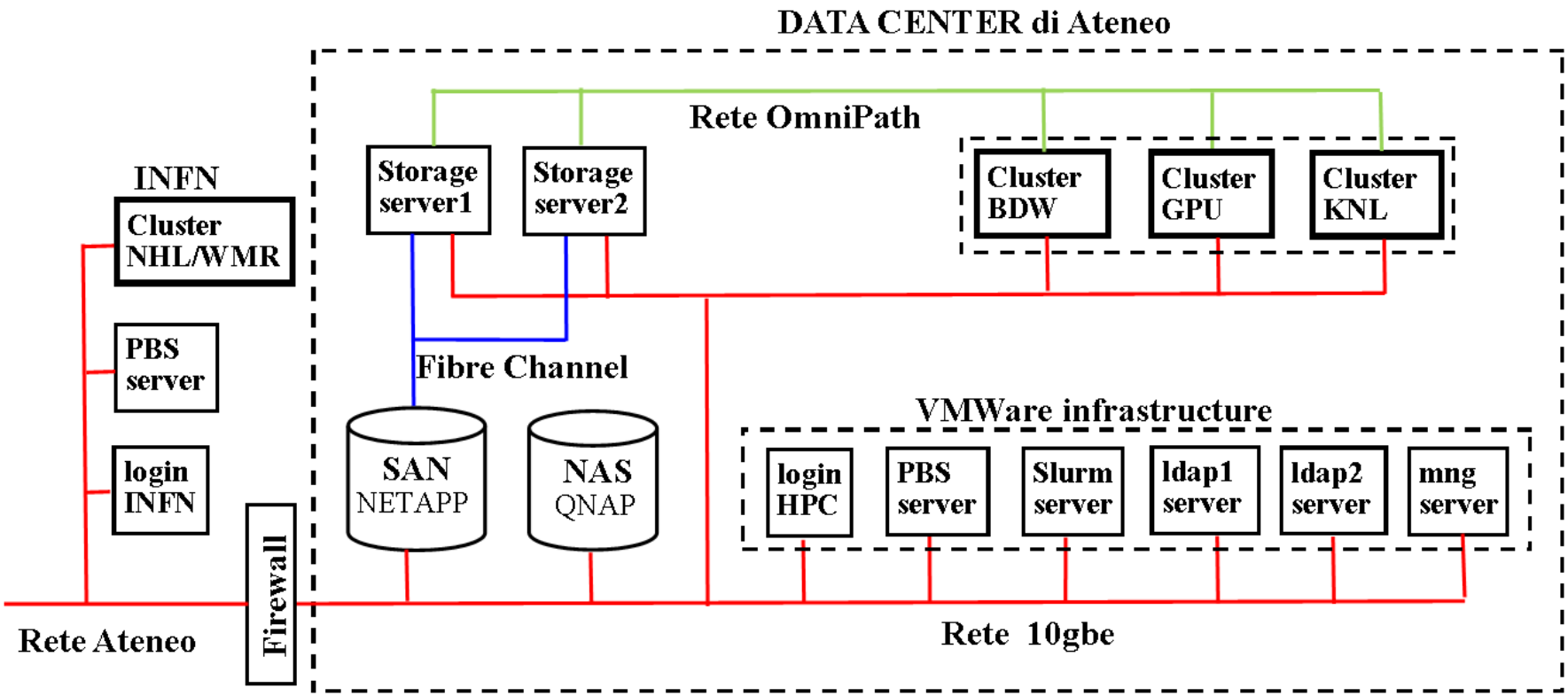
Tesla P100
for NVLink-enabled Servers

5.3 TF DP · 10.6 TF SP · 21 TF HP
720 GB/sec Memory Bandwidth, 16 GB

Tesla P100
for PCIe-Based Servers

4.7 TF DP · 9.3 TF SP · 18.7 TF HP
Config 1: 16 GB, 720 GB/sec
Config 2: 12 GB, 540 GB/sec

Schema


DATA CENTER di Ateneo
Rete OmniPath
INFN
Cluster NHL/WMR
PBS server
login INFN
Storage server1
Storage server2
Cluster BDW
Cluster GPU
Cluster KNL
Fibre Channel
SAN NETAPP
NAS QNAP
VMWare infrastructure
login HPC
PBS server
Slurm server
ldap1 server
ldap2 server
mng server
Firewall
Rete Ateneo
Rete 10gbe

Storage

NOME	USO	TECNOLOGIA	DIM.	TEMPI
HOME + progetti	Programmi e dati	NAS	240 TB	
SCRATCH	Dati run-time	SAN	40 TB	estate 2017 Lustre o GPFS?
ARCHIVE	Archivio a medio / lungo termine	NAS /cloud	~ PByte	Fine 2017?

file-system distribuiti Competenza locale GPFS (grazie a INFN).
Nei principali Data Center HPC internazionali si sta diffondendo Lustre
Da decidere entro l'estate.

Cloud UniPR fa parte del gruppo di lavoro “Cloud” del CODAU, insieme ad altri 15 Atenei con cui sta definendo un modello di servizio condiviso per la gestione dell’archiviazione a supporto dei gruppi di ricerca.

Software

Linux CentOS 7.x

Clobber e puppet per installazione e configurazione dei nodi

Compilatore Intel per programmazione ottimizzata di Intel Xoen e Xeon PHI

Compilatore PGI per programmazione GPU Nvidia con openACC
- *Versione community del 2016*

Compilatori GNU, librerie MPI e **scientifiche** e **gestione moduli da openHPC**

OpenHPC

openHPC è un consorzio parte di Linux Foundation nato nel nov. 2015 (SC15) per facilitare l'installazione e la gestione dell'ambiente Software di un cluster HPC

Principali membri
Commerciali: Altair, Arm, Cray, Dell, Fujitsu, HP, Intel, Lenovo, Redhat, SGI, SUSE
HPC Center: Berkeley, Barcelona, Cea, Cineca, Livermore, Los Alamos, Pittsburgh, Sandia, TACC

Piattaforme supportate: **x86_64** e **ARM64**

Si basa su componenti software prevalentemente open source, testati, integrati e pacchettizzati in formato rpm:

Linux: **CentOS, SUSE**
WareWulf (da Berkeley Labs) per la gestione dei nodi basata su immagini
Nagios e Ganglia per il monitoraggio del cluster
Lustre come file system distribuito (parte client)
Driver Infiniband e OmniPath
Compilatori GNU, Intel (licenza) **, librerie MPI** e **scientifiche** con moduli **lmod**
Slurm e PBSpro come resource manager

PBSpro o SLURM?

PBSpro

Nel 2016 Altair (distributore di PBSpro) diventa membro di openHPC e da Giugno 2016 PBSpro è rilasciato con licenza OpenSource. In uso al CINECA su MARCONI

Installato inizialmente sul cluster.

Problemi riscontrati nella configurazione:

- **Gestione multi GPU: Manca il GPU binding**
  E4 ha dovuto realizzare un hook custom per PBS.
- **Gestione Accounting (reporting/quota)**
- Non sono gestite la reportistica e le quote (core-hours) per utente e gruppo di utenti. Occorre realizzare un tool custom.

Slurm

Gestione MultiGPU: integrata
Accounting (reporting/quota): da realizzare

Soluzione transitoria (in collaborazione con E4):
Un Server PBSpro di pre-produzione + un Server Slurm per test

Memory Performance

NOTA: Tutti i test di performance sono stati eseguiti prevalentemente per il collaudo dell'installazione e devono essere considerati preliminari.

Benchmark IMB-MPI1 pingpong tra 2 processi sullo stesso nodo

LATENCY	**BDW** Same socket	**BDW** Different sockets	**KNL**
OpenMPI GNU	0.9 µs	1.7 µs	
OpenMPI PGI	0.8 µs	1.6 µs	
Intel MPI	0.6 µs	1 µs	1.6 µs

BANDWIDTH	**BDW** Same socket	**BDW** Different sockets	KNL
OpenMPI GNU	5.4 GB/s	5 GB/s	
OpenMPI PGI	5 GB/s	3.2 GB/s	
Intel MPI	8.9 GB/s	8.0 GB/s	4 GB/s

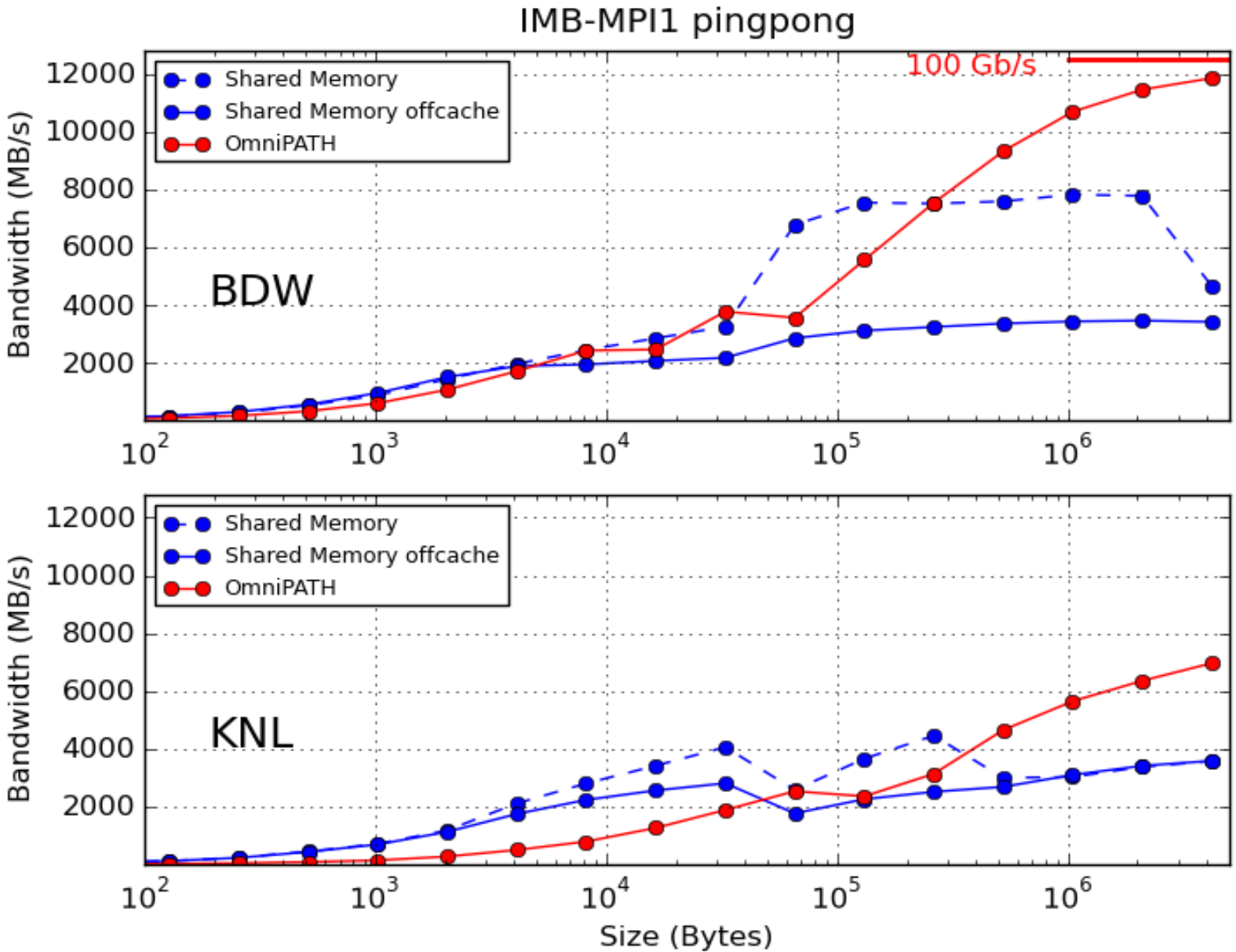
Network Performance

Benchmark IMB-MPI1 pingpong tra 2 processi su nodi diversi

LATENCY	BDW OPA	BDW TCPoOPA	KNL OPA	KNL TCPoOPA
OpenMPI GNU	2.2 µs			
OpenMPI PGI	3.7 µs	16 µs		
Intel MPI	1.5 µs	13 µs	3.8 µs	51 µs

BANDWIDTH	BDW OPA	BDW TCPoOPA	KNL OPA	KNL TCPoOPA
OpenMPI GNU	6.6 GB/s			
OpenMPI PGI	5 GB/s	0.22 GB/s		
Intel MPI	11 GB/s	3.4 GB/s	6 GB/s	0.9 GB/s

Shared Memory vs OmniPath



Case study su GPU
Simulazione di fenomeni di allagamento

La simulazione numerica di problemi di allagamento è una sfida scientifica di notevole impatto economico. I cambiamenti climatici e il notevole sviluppo delle grandi aree urbane richiedono nuovi sistemi di difesa dalle piene e l'aggiornamento delle mappe di rischio idraulico.

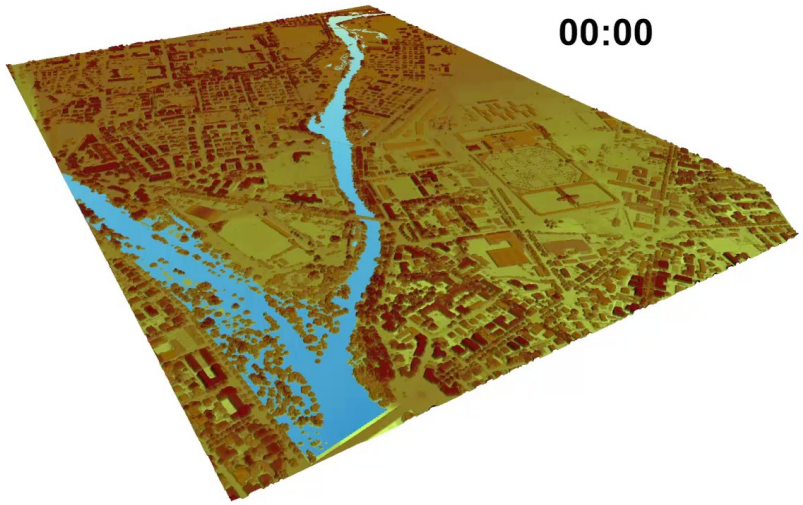
PROGETTI
SIR 2015, Agenzia Regionale di Protezione Civile (Emilia-Romagna) – 2015 , Servizio Tecnico dei Bacini degli affluenti del Po (STB) della Regione Emilia-Romagna

Vacondio, Dal Palù, et. al, Nat. Hazard, doi: 10.1007/s11069-015-1959 (2015).

CALCOLO
Codice sviluppato dagli autori in CUDA

HyLab
PARMA HYDRAULIC RESEARCH LAB

Università degli Studi di Parma
DICATeA
Dipartimento di Ingegneria Civile, dell'Ambiente, del Territorio e Architettura



Quote idriche della simulazione dell'evento alluvionale del 14 ottobre 2014 nella città di Parma

Case study su GPU
Simulazione di fenomeni di allagamento

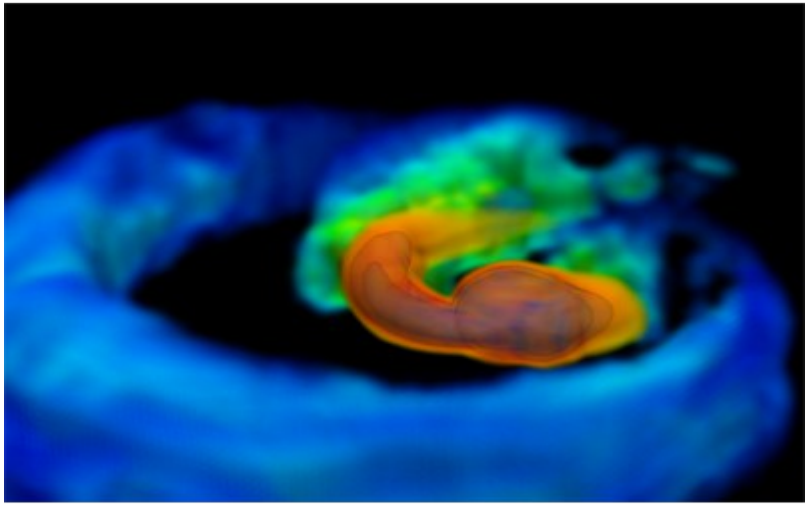
Il programma è scritto in CUDA e attualmente gira su 1 GPU NVIDIA K40.
Risultati preliminari: Speed-up K40-P100 ~4.
Riduzione del tempo di una simulazione x4.

E' in fase di sviluppo la versione multi-GPU del programma, con MPI.

GPU	Peak perf. (s.p.)	Accesso Memoria	Bacino fluviale	Tempo simulazione / Tempo reale
1 K40	4.3 TFlops	288 GB/s	100 Km	1/5 - 1/20
1 P100	9.3 TFlops	540 GB/s	100 Km	1/20 - 1/80
10 P100	93 TFlops		1000 Km	1/20 - 1/80

Case study su KNL
Simulazione di stelle di neutroni binarie

La simulazione numerica di stelle di neutroni binarie è la sorgente attesa più rilevante di onde gravitazionali osservabili dalla nuova rete di Osservatori interferometrici, LIGO e VIRGO.



Simulazione relativistica dei resti di uno scontro tra due stelle di neutroni
Dietrich, Bernuzzi, Ujevic, Brügmann Phys. Rev. D 91, 124041, 2015

Progetto BNSdisk
- In produzione su Galileo e Marconi BDW
- PRACE 2017 **30M core hours su KNL**
P.I. R. De Pietri

Calcolo
Einstein Toolkit (piattaforma software).
Codice MPI/openMP sviluppato da una comunità internazionale

Progetto GWBNS
- In produzione su Marconi BDW
- PRACE 2017 : **42M core hours su KNL**

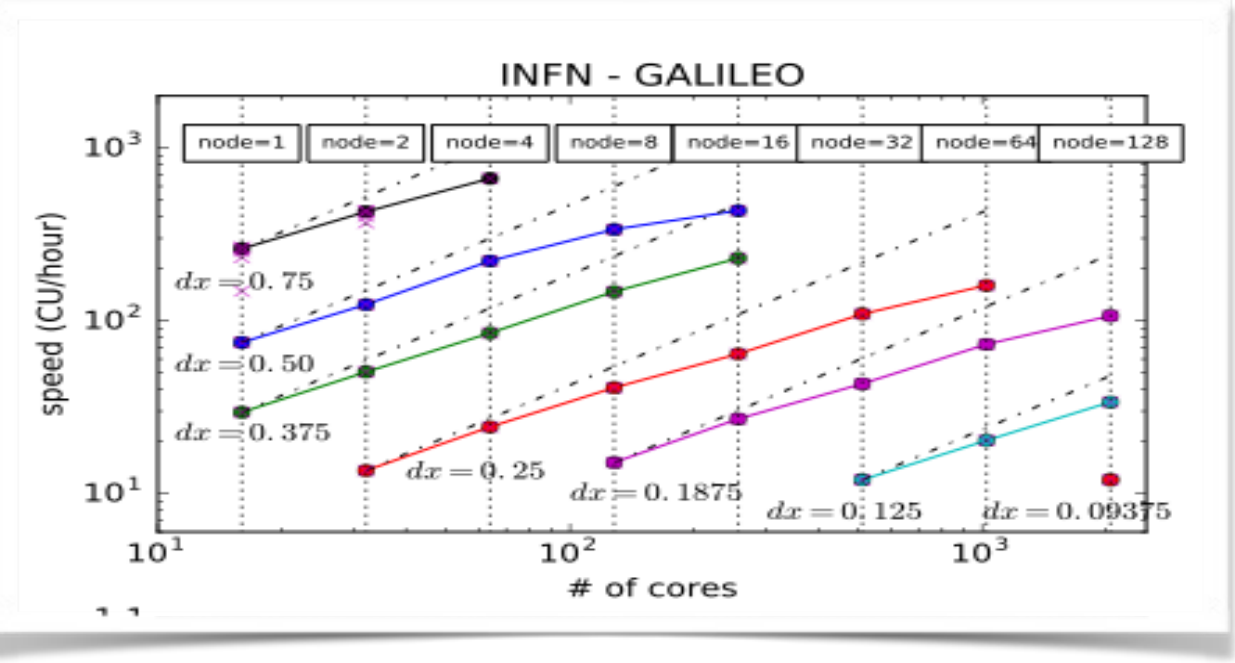
P.I: S. Bernuzzi

Calcolo
Codice sviluppato dagli autori in MPI/openMP

Case study su KNL
Einstein Toolkit

E.T: è realizzato per applicazioni di relatività computazionale
Insieme di packages sviluppato a partire dagli anni 90 da una comunità scientifica internazionale. Il codice è sviluppato da circa 100 programmatori utilizzando diversi linguaggi (F77, F90, C, C++) e parallelismo ibrido MPI/ openMP.

Utilizzato a Parma per simulare l'evoluzione idrodinamica di una stella di neutroni.
PRACE nel 2015 per l'utilizzo di Galileo (CINECA)
PRACE nel 2017. Assegnate 30 M core-hours su Marconi KNL



Sono in corso test di performance e scalabilità del codice attuale di E.T. su KNL.

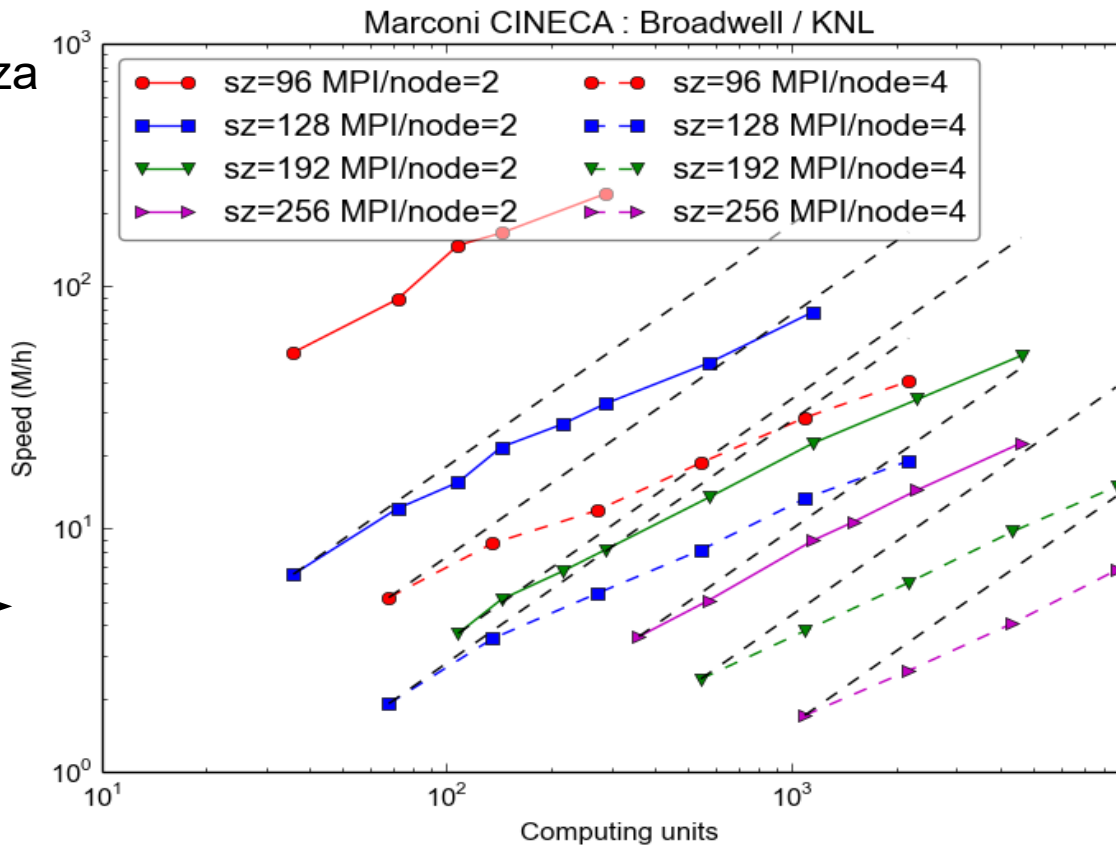
La distribuzione piu' recente di E.T. (ET_2016_11) non supporta la vettorizzazione.

Case study su KNL
BAM code : scaling e vettorizzazione

BAM e' un codice scritto in C e utilizza il parallelismo ibrido MPI/openMP

In produzione sul cluster Marconi Broadwell.

Confronto scaling su Marconi BDW (solid line) e KNL (dashed line) con la sola ricompilazione →

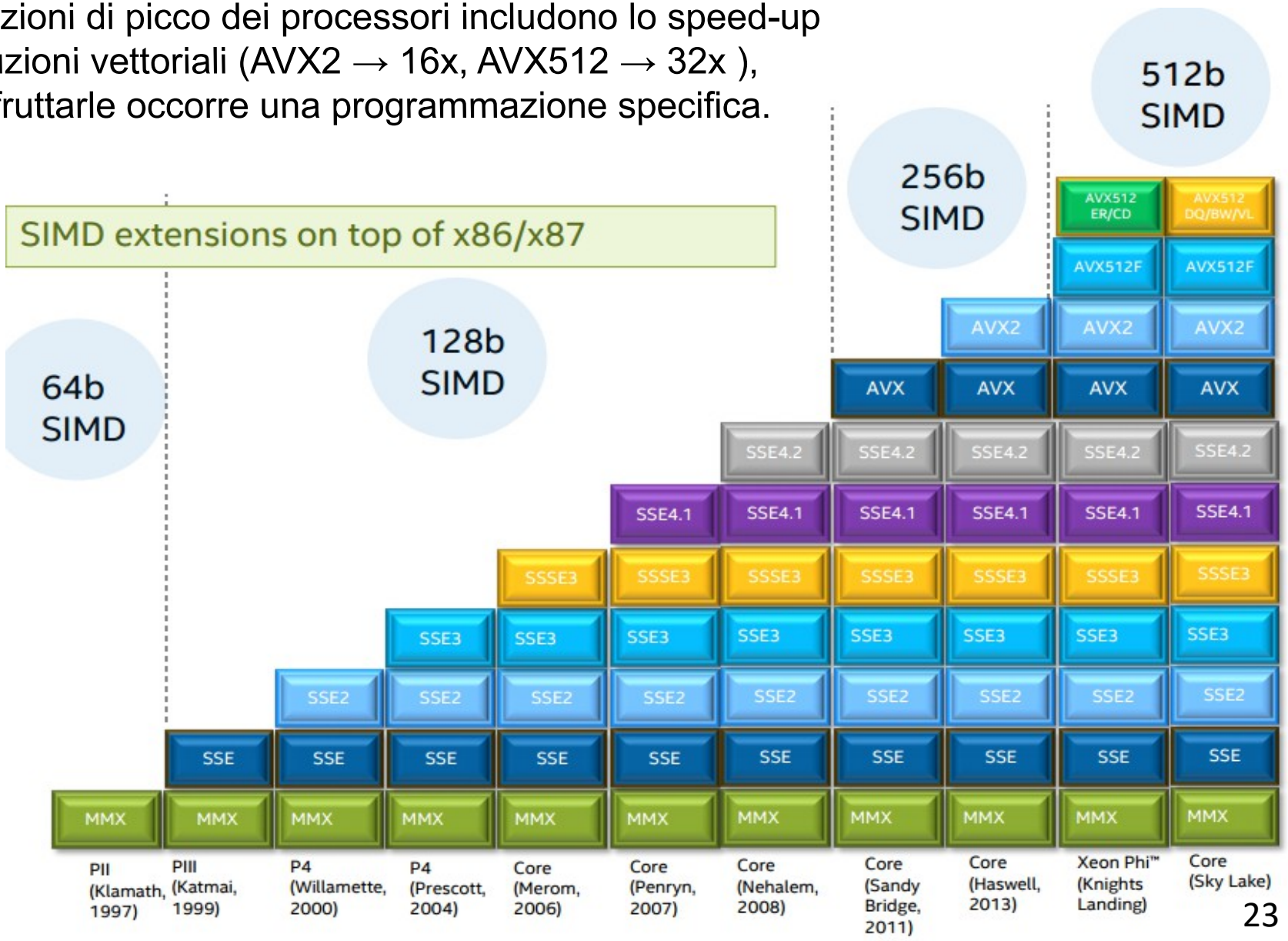


La ricompilazione su BDW e KNL disabilitando la vettorizzazione (-no-vec) non mostra differenze apprezzabili. →

	-xMIC-AVX512	-no-vec
1 nodo BDW	81.94	81.22
1 Nodo KNL	19.68	18.21

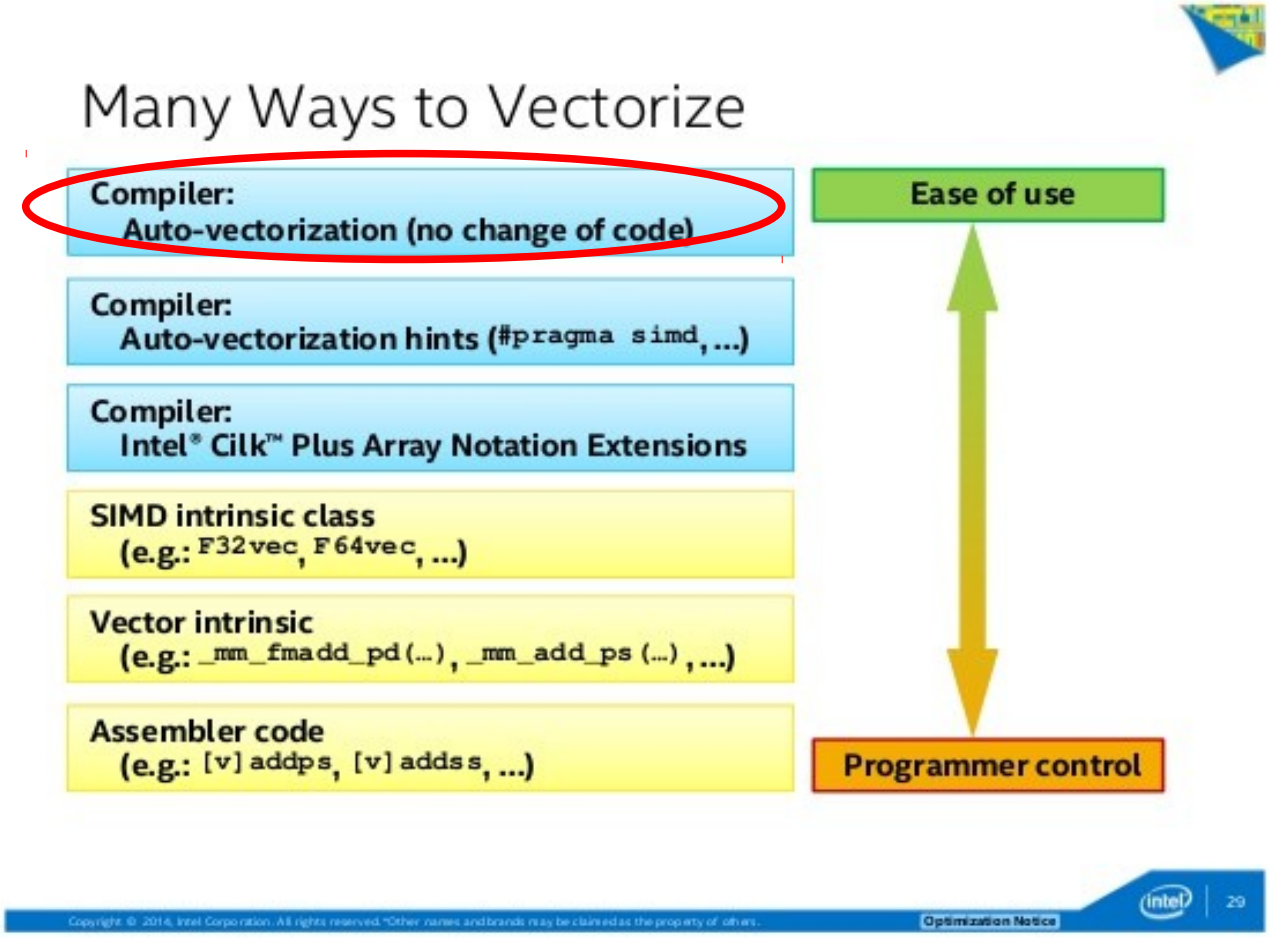
Le istruzioni vettoriali

Le prestazioni di picco dei processori includono lo speed-up delle istruzioni vettoriali (AVX2 → 16x, AVX512 → 32x), ma per sfruttarle occorre una programmazione specifica.



Many Ways to Vectorize

Per molte applicazioni (*O*(Prace)) è difficile stare al passo con le nuove tecnologie di calcolo o i nuovi paradigmi di programmazione; la riscrittura è onerosa.
E' preferibile, se possibile, demandare al compilatore la gestione delle architetture.



Auto-Vettorizzazione

Non tutti i loop possono essere auto-vettorizzati.

Alcuni esempi:

- Non possono esserci dipendenze tra le iterazioni

```
for (i=1; i<MAX; i++) {
   d[i] = e[i] – a[i-1];
   a[i] = b[i] + c[i];
}
```

- Loop complessi

```
for (int i = 0; i < N; i++)
    a[i] = foo(b[i]);
```

- Chiamate a funzione nel loop

- Loop su dati organizzati in Array of Structure

```
struct Point{ //AoS
 float r;
 float g;
float b;  }
Point Points[1000];
```

- Loop su dati non allineati in memoria

La Vettorizzazione del codice

Per valutare le prestazioni dell'auto-vettorizzazione è stato creato un modello semplificato 2D. Nelle routine con indipendenza tra le iterazioni è stata vettorizzato il loop più interno, aggiungendo l'allineamento in memoria e l'utilizzo delle funzioni matematiche vettoriali di MKL (Intel® Math Kernel Library).
Esempio:

Versione originale

```
#pragma omp parallel for
 for (j=jmin; j<=jmax; j++) {
  for (i=imin; i<=imax; i++) {
   ij = ij2i(i,j,Sx);
   r[ij] += ( pow(log(Pi+sqrt(j)+i), 3.1)*u[ij] \
         + pow(u[ij],4/3) )*kill_source;
  }
 }
```

Test preliminari:
speed-up> 4

```
#pragma omp parallel
{
double *Va = (double *)mkl_malloc( Sx*sizeof(double ),64 );
…
#pragma omp for
for (j=jmin; j<=jmax; j++) {
  …
 for (i=0; i<Sx; i++) Va[i]= ijmin + i + sqrtj + Pi;
 vdLn(Sx,Va,Vb);
 vdPowx(Sx,Vb, 3.1, Vc);
 for (i=0; i<Sx; i++) Vd[i]=Vc[i] * u[ijmin+i];
 vdPowx(Sx,&u[ijmin],f4o3,Vd );
 vdAdd(Sx, Vc, Vd, Va);
 for (i=0; i<Sx; i++) r[ijmin+i]+=(Va[i])*kill_source;
 }
mkl_free(Va);
…
}
```

Corso HPC per dottorandi

18 ore dal 30 Maggio al 15 Giugno 2017
Senza particolari prerequisiti di programmazione
Iscritti 36 dottorandi da tutti i dipartimenti scientifici

Contenuti

1. **Ambiente Linux: Nozioni pratiche necessarie per l'utilizzo dell'ambiente HPC (3 ore)**
- La shell di Linux, shell scripting, il sistema di code
2. **Ambiente software scientifico, compilatori, librerie (3 ore)**
- Laboratorio con l'ambiente software installato sul cluster
3. **Hardware (3 ore)**
- Sistemi di storage: NAS, SAN e archiviazione a lungo termine
- Architetture tradizionali e accelerate, gerarchie di memoria e comunicazione
- L'architettura del cluster di Ateneo
4. **Algoritmi paralleli (3 ore)**
- Performance, speed-up, efficienza
- Qualche esempio di parallelizzazione di algoritmi: impostazione del problema
5. **Modelli di programmazione parallela (6 ore)**
- Parallelismo a scambio di messaggi, MPI
- Parallelismo a memoria condivisa, openMP, vettorizzazione
- Parallelismo con GPU, CUDA

Grazie per l'attenzione!

Domande?