Lattice Quantum ChromoDynamics Revisited on Multicore Vector Machines

$$D\psi(x) = A\psi(x) - \frac{1}{2}\sum_{\mu=0}^{4}\{[(I_4 - \gamma_\mu) \otimes U_{x,\mu}]\psi(x+\hat{\mu}) + [(I_4 + \gamma_\mu) \otimes U^{\dagger}_{x-\hat{\mu},\mu}]\psi(x-\hat{\mu})\}.$$

Claude Tadonki

MINES ParisTech – PSL Research University
Centre de **R**echerche **I**nformatique
claude.tadonki@mines-paristech.fr

- **Q**uantum **C**hromo**D**ynamics (**QCD**) is the theory of strong interactions, whose ambition is to explain nuclei cohesion as well as neutron and proton structure, i.e. most of the visible matter in the Universe.

  ➡ Creation of the universe (matter synthesis right after the BIG BANG) !!!

- The main computational model for Quantum chromodynamics (QCD) investigations is the so-called **L**attice **Q**uantum **C**hromo**D**ynamics (**LQCD**).

  ➡ Based on the LQCD model, (large scale) simulations are implemented !!!

- In the LQCD model, the space-time is represented by a finite discrete system with **cartesian coordinates**, while the interaction between subparticles is governed by strong force theory.

  ➡ LQCD provides an analytical formalism, the reference for numerical studies.

- With a **fixed lattice spacing**, **fine-grained** simulations are considered through **larger lattice** volume, thus increasing **memory** and **computation** complexities.

  ➡ Solving the Wilson-Dirac system is the most critical computation kernel.

The **Wilson-Dirac operator D** on a site x (a spinor) of a quark field ψ can be defined as follows

$$D\psi(x) = A\psi(x) - \frac{1}{2}\sum_{\mu=0}^{4}\{[(I_4 - \gamma_\mu) \otimes U_{x,\mu}]\psi(x+\hat{\mu}) + [(I_4 + \gamma_\mu) \otimes U^{\dagger}_{x-\hat{\mu},\mu}]\psi(x-\hat{\mu})\}$$

where

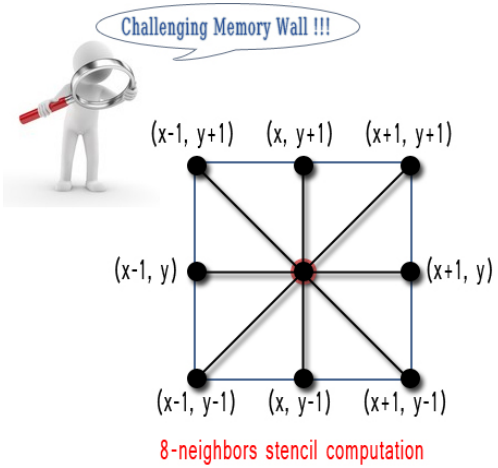
- A is a 12×12 complex matrix of the form $\alpha I_{12} + \beta(\nu \otimes \gamma_5)$, where α, β are complex coefficients and ν a 3×3 complex matrix
- x is a given point of the lattice, which is a finite subset of $\mathbb{N}^4$
- ψ (called *quark field* or *Wilson vector*) is a vector of 12-components complex vectors over the whole lattice
- $U_{x,\mu}$ is a 3×3 complex matrix (called *gluon field matrix* or *gauge matrix*) at (x, μ).

$$\gamma_0 = \begin{pmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix} \quad \gamma_1 = \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & -i & 0 \\ 0 & i & 0 & 0 \\ i & 0 & 0 & 0 \end{pmatrix}$$

$$\gamma_2 = \begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix} \quad \gamma_3 = \begin{pmatrix} 0 & 0 & -i & 0 \\ 0 & 0 & 0 & i \\ i & 0 & 0 & 0 \\ 0 & i & 0 & 0 \end{pmatrix}$$

$$\gamma_5 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

4D stencil (x, y, z, t):
(x + 1, y , z, t), (x - 1, y, z, t), (x, y + 1, z, t), (x, y - 1, z, t)
(x, y , z + 1, t), (x, y, z - 1, t), (x, y, z, t + 1), (x, y, z, t – 1)



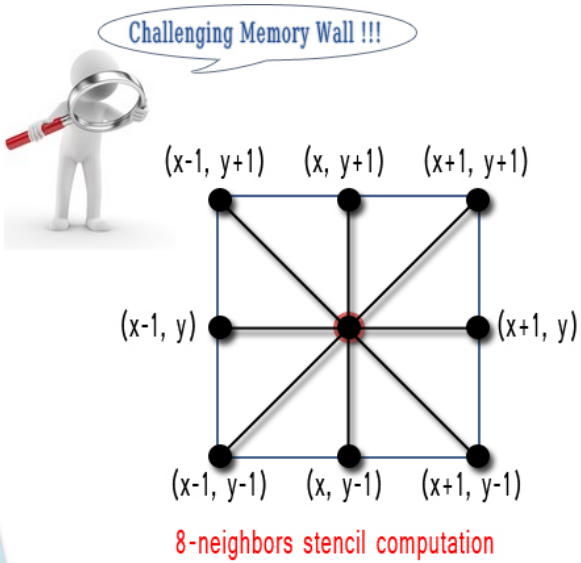
The application of the Wilson-Dirac operator over all *spinors* of a *quark field* can be seen as a matrix-vector product, thus the name ***Wilson-Dirac matrix***.

Remark 1. Given u a 4-components complex vector, $s \in \{-1, 1\}$, and $\mu \in \{0, 1, 2, 3\}$, the 4-components complex vector v defined by

$$v = (I_4 - s\gamma_\mu)u,$$

can be more efficiently calculated using the following relations

	$\mu = 0$	$\mu = 1$	$\mu = 2$	$\mu = 3$
$s = 1$	$v_1 = u_1 + u_3$ $v_2 = u_2 + u_4$ $v_3 = v_1$ $v_4 = v_2$	$v_1 = u_1 + iu_4$ $v_2 = u_2 + iu_3$ $v_3 = -iv_2$ $v_4 = -iv_1$	$v_1 = u_1 + u_4$ $v_2 = u_2 - u_3$ $v_3 = -v_2$ $v_4 = v_1$	$v_1 = u_1 + iu_3$ $v_2 = u_2 - iu_4$ $v_3 = -iv_1$ $v_4 = iv_2$
$s = -1$	$v_1 = u_1 - u_3$ $v_2 = u_2 - u_4$ $v_3 = -v_1$ $v_4 = -v_2$	$v_1 = u_1 - iu_4$ $v_2 = u_2 - iu_3$ $v_3 = iv_2$ $v_4 = iv_1$	$v_1 = u_1 - u_4$ $v_2 = u_2 + u_3$ $v_3 = v_2$ $v_4 = -v_1$	$v_1 = u_1 - iu_3$ $v_2 = u_2 + iu_4$ $v_3 = iv_1$ $v_4 = -iv_2$



Remark 2. Using the *normal factors decomposition* and its commutativity, we have

$$\begin{aligned}(I_4 - s\gamma_\mu) \otimes U &= ((I_4 - s\gamma_\mu) \otimes I_3)(I_4 \otimes U) \\ &= (I_4 \otimes U)((I_4 - s\gamma_\mu) \otimes I_3)\end{aligned}$$

Thus, the computation of $[(I_4 - s\gamma_\mu) \otimes U]\psi$ can be done as follows

$$[(I_4 - s\gamma_\mu) \otimes U]\psi = (I_4 \otimes U)\{((I_4 - s\gamma_\mu) \otimes I_3)\psi\}$$

- Using the above factorizations & simplifications leads to an optimal evaluation of Wilson-Dirac.
- Data locality is the main issue that will impact on memory and data communication costs.

8×N spinors	$8 \times (24 \times 8) \times N$	1536×N bytes
8×N U matrices	$8 \times (18 \times 8) \times N$	1152×N bytes
8×N integer indexes	$8 \times (18) \times N$	64×N bytes

Fig. Data inventory for a single Dirac operator

N = LxLyLzLt
Double precision complex numbers

- In order reduce the cost of irregular memory accesses related to SU(3) matrices, the so-called GAUGE-COPY strategy is considered: for each site of the whole lattice, the 8 required SU(3) matrices are stored contigously → better memory performance at the expense of redundancy!

- Examples of GAUGE configuration sizes ($T \times L^3$)
  sizeof(U) = 3x3x2x8 = 144 bytes in double precision

L	T	Sizeof(Us)
16	32	0.28 GB
32	64	4.5 GB
64	128	72 GB
128	256	2 304 GB

- Parallelism is considered also because of the global memory requirement (forget disk!), even if work inefficient!!!

- Efficient Wilson-Dirac is vital for LQCD simulations as it is evaluated so many times.

Our goal is to solve the following linear system

$$D\psi(x) = \phi$$

- This system is an important step the synthesis of a statistical gauge configuration sample.
- The solution of the system, called propagator, appears in the expression of the so-called ***fermionic force***, used to update the momenta associated with the gauge fields along a trajectory in the **H**ybrid **M**onte **C**arlo (**HMC**) algorithm.
- Many propagators are computed on a trajectory, they should fullfil the **reversibility criteria**.
- Large-scale simulations involve huge Dirac matrices with bad condition numbers for small pion masses.
- Because of the reversibility condition and the quality expected for the propagators in order to be useful for physic, a high numerical precision is required to solve the Dirac system.
- Because of the implicit form of the Dirac matrix and the required precision, **iterative methods** are considered. We should prepare for a huge number of iteration or divergence.
- Large-scale inversions need a combination of state-of-the-art HPC and matrix computation !!!
- LQCD research is active through collaborations (projects) and dedicated supercomputers.

- Large volume of data (disk / memory / network)
- Significant number of solvers iterations due to numerical sensitivity
- Redundant memory accesses from interleaving data dependencies
- Use of double precision because of the need for accuracy (hardware penalty)
- Misaligned data (inherent to specific data structures)
  - Exacerbates cache misses (depending on cache size)
  - Becomes a serious problem when considering accelarators
  - Leads to « false sharing » with Shared-Memory paradigm (Posix, OpenMP)
  - Padding is one solution but would dramatically increase memory requirement
- Memory/Computation compromise in data organization (e.g. gauge replication)
- Important interprocessor communication with distributed memory parallel machines

- **High-precision L**attice **Q**uantum **C**hromo**D**ynamics simulations.
- The ANR project **PetaQCD** was targeting **256×128³** lattices.
- One evaluation of the *Dirac operator* on a **256×128³** lattice involves

$256 \times 128^3 \times 1500 \approx 10^{12}$ (stencil) **floating-point operations**

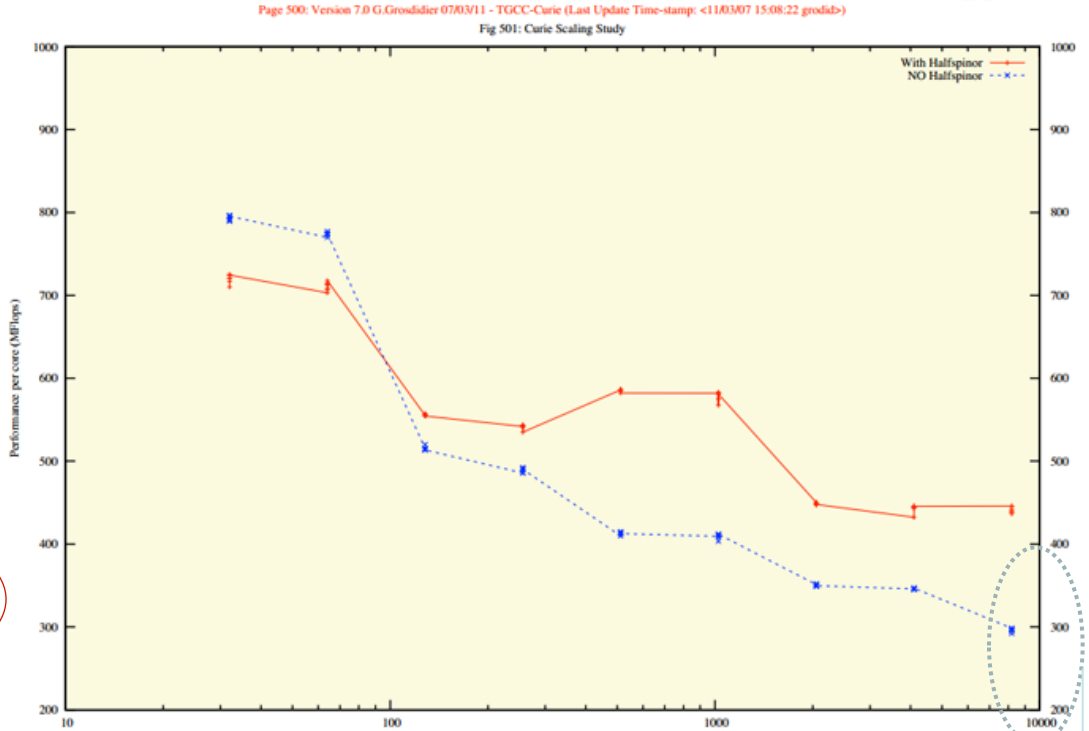
With our 10,000 cores, we can roughly perform $500 \times 10^3 \times 10^6 = 5 \times 10^9$ fps

Our 256×128^3 lattice would then require 200 seconds ≈ **3 minutes** for each evaluation of the *Dirac operator*.

10 days !!!

Now, imagine that we have to do it 5000 times to solve one Dirac linear system !!!

Curie Fat performance (weak scaling)



G.Grosdidier, « *Scaling stories* », PetaQCD Final Review Meeting, Orsay, Sept. 27th – 28th 2012

500 Mflops/core

- The (peak/sustained) performance of supercomputers is increasing significantly (≈ $x10^{14}$ since 1993).

  The following characteristics are becoming a standard
  - Several cores (2012 list: 84% ≥ 6 cores and 46% ≥ 8 cores)
  - Vector units (with larger vector registers)
  - Accelerated (mainly GPU: 62 systems of the 2012 and #1)

- The **gap between peak and sustained** performances on real-life applications is clearly questionable.

  ☞ HPC investigations should focus on this.

- When it comes to accelerators, **data transfer** is critical.

  ☞ Compromise and overlap (scheduling)

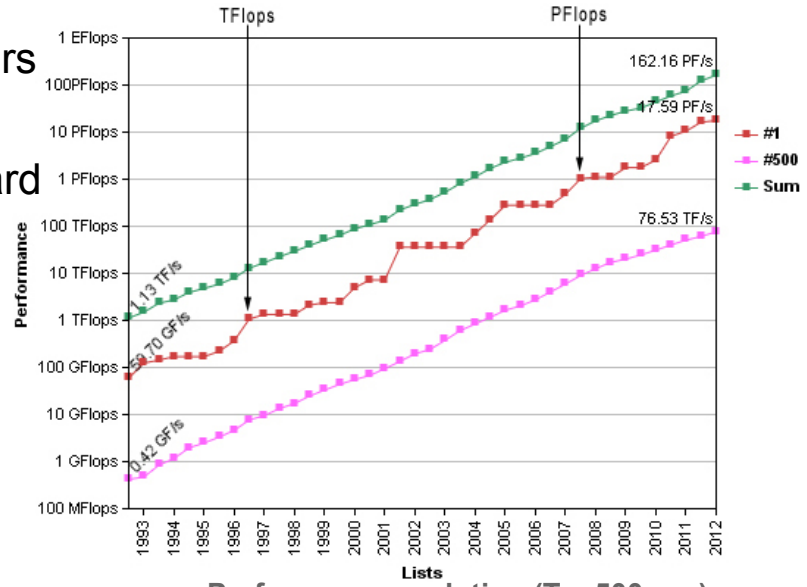
Chris Gregg and Kim Hazelwood, *Where is the Data? Why You Cannot Debate CPU vs. GPU Performance Without the Answer*, International Symposium on Performance Analysis of Systems and Software (ISPASS), Austin, TX. April 2011.

- Taking advantage of all aspects of a computing nodes requires a complex **hybrid/heterogeneous programming** .

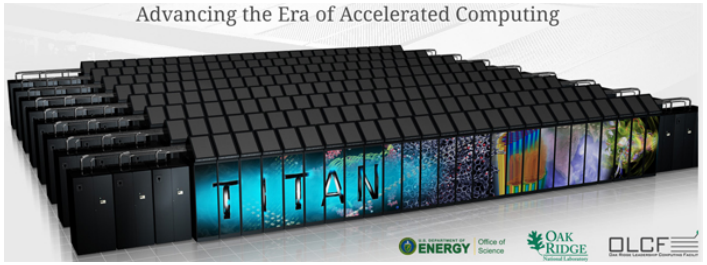
  ☞ Code generation and transformation

- **Energy** is a major concern.

  ☞ Power aware programming and scheduling.



Performance evolution (Top500.org)



TITAN CRAY-XK7 the (2012) world fastest supercomputer
- 299 008 CPU cores (16-cores AMD Opteron 6274)
- 18 688 NVIDIA Tesla K20 GPUs
- Peak: **27.11** PFlop/s.
- Sustained: **17.59** PFlop/s (Linpack)

IBM System Technology Group

Blue Gene/Q

1. Chip: 16+2 μP cores

2. Single Chip Module

3. Compute card: One chip module, 16 GB DDR3 Memory, Heat Spreader for H_2O Cooling

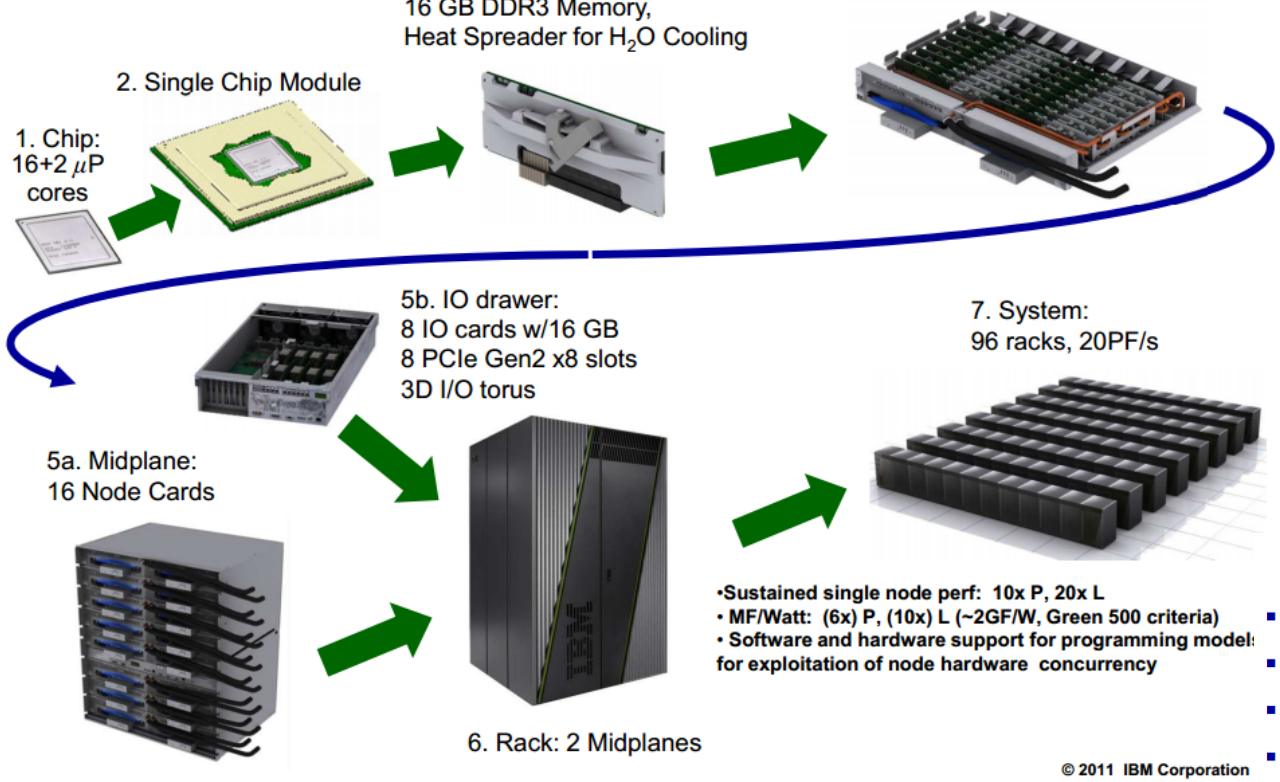
4. Node Card: 32 Compute Cards, Optical Modules, Link Chips; 5D Torus

5a. Midplane: 16 Node Cards

5b. IO drawer: 8 IO cards w/16 GB 8 PCIe Gen2 x8 slots 3D I/O torus

6. Rack: 2 Midplanes

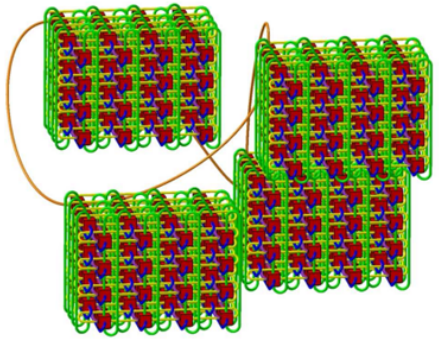
7. System: 96 racks, 20PF/s



- Sustained single node perf: 10x P, 20x L
- MF/Watt: (6x) P, (10x) L (~2GF/W, Green 500 criteria)
- Software and hardware support for programming models for exploitation of node hardware concurrency

NETWORK CHARACTERISTICS

- Integrated 5D torus at 1.76 GB/s (bidirectional)
- 2 GB/s raw bandwidth on all 10 links
- Hardware latency 80ns to 3us
- Additional 11th link for communication to IO nodes



Network Performance

- All-to-all: 97% of peak
- Bisection: > 93% of peak
- Nearest-neighbor: 98% of peak
- Collective: FP reductions at 94.6% of peak

- Nov 2011 TOP500 Entry

# of cores:	65,536
# of nodes:	4096 (4 racks)
R_{max}:	677 TF
R_{peak}:	838.9 TF
N_{max}:	2719743 (very important for intensive simulations)
Power:	85 kW (network excluded)
Sustained perf:	80.72% (Good, but this is linpack LQCD is another story !!!)
GF/W:	1.99

RANK	SITE	SYSTEM	CORES
1	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000

Getting Tianhe-2 RPEAK:

- CPU-core frequency: 2.2 Ghz = 2.2 GFlops
- Considering the vector capability (256-bit wide - 4 DP): 4 x 2.2 = 8.8 GFlops
- Given the CPU can do ADD and MUL in one cycle (FMA): 2 x 8.8 = 17.6 GFlops
- Finally the total number of cpu-cores: 3,120,000 x 17.6 Ghz = 54.912 Pflops

Clearly, we should exploit all levels of parallelism, if we need to harvest an acceptable fraction of the peak performance for large-scale LQCD.

We consider a hyperthreaded quad-core machine based on

- 2.4 GhZ Intel Core i7,
- 8 GB of DDR3 main memory
- private L2 cache of 256 KB per core and
- shared L3 cache of 6 MB
- 256-bit-wide vector instructions with AVX intrinsics.

Thus

- $4 \times 4 =$ 9.6 GFlops per core in double precision
- 38.4 GFlops for the whole processor.

#threads	t(s)	GFlops	%Peak	speedup
1	0.0506	4.17	45 %	1
2	0.0257	8.20	45 %	1.97
4	0.0213	9.91	27 %	2.38
8	0.0154	13.70	37 %	3.29

TABLE I
BASELINE PERFORMANCES FROM THE FIRST STAGE CODE

☞ Good scaling with hyperthreading, but weak absolute performance.

#threads	t(s)	GFlops	%Peak	speedup
1	0.0476	4.42	48 %	1
2	0.0251	8.38	46 %	1.89
4	0.0166	12.72	34 %	2.86
8	0.0132	15.87	44 %	3.60

TABLE II
PERFORMANCE OF THE POOL OF TASKS SCHEDULING

	4	**8**	**16**	**32**	**64**	**128**
4 threads	9.78	9.47	11.24	12.37	12.51	12.72
8 threads	10.02	12.52	13.85	15.52	13.10	15.87
	256	**512**	**1024**	**2048**	**4096**	-
4 threads	12.20	12.29	11.76	12.45	13.26	-
8 threads	16.99	13.52	16.25	15.80	8.33	-

TABLE III
GFLOPS PERFORMANCE WITH VARIOUS POOL CARDINALITIES

☞ Using a pool of tasks seems to improve scalability and global performance

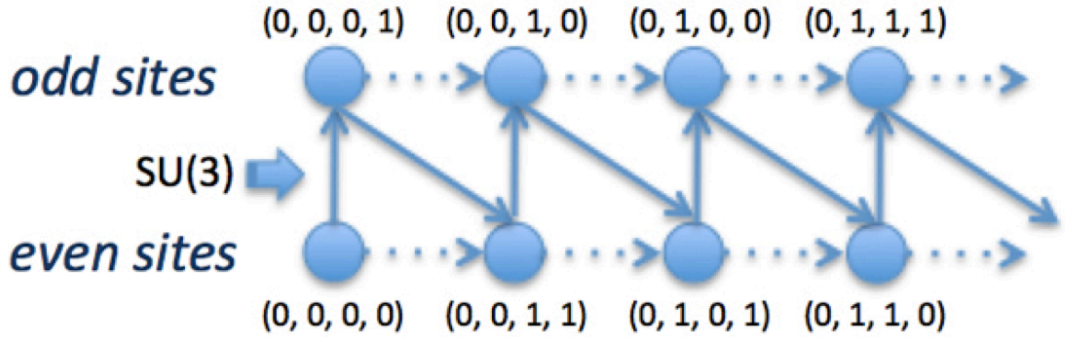


Fig. 1. Alternating Even-Odd Scheduling

#threads	t(s)	GFlops	%Peak	speedup
1	0.0499	4.23	44 %	1
2	0.0257	8.21	43 %	1.98
4	0.0177	11.88	32 %	2.81
8	0.0126	16.71	40 %	3.96

TABLE IV
IMPACT OF THE EVEN-ODD ALTERNATING SCHEDULING

☞ Additional improvement of the scalability and global performance

$s^{(j)} = [s_1^{(j)}, s_2^{(j)}, \cdots]$, $j = 1, 2, 3, 4$, the corresponding vector structure would be $s = [s_1^{(1)} s_1^{(2)} s_1^{(3)} s_1^{(4)}, s_2^{(1)} s_2^{(2)} s_2^{(3)} s_2^{(4)}, \cdots]$.

#threads	t(s)	GFlops	%Peak	speedup
1	0.0261	8.07	84 %	1
2	0.0144	14.68	76 %	1.82
4	0.0128	16.45	42 %	2.04
8	0.0140	15.03	39 %	1.86

TABLE V
PERFORMANCE WITH AOS-TO-SOA

☞ Good performance up to 4 threads, **then memory wall**.

- The third row of a SU3 matrix can be reconstructed from its first two rows by taking The complex conjugate of their cross product.

$$u_3 = \overline{u_1 \wedge u_2}$$

#threads	t(s)	GFlops	%Peak	speedup
1	0.0233	9.06	94 %	1
2	0.0139	15.12	79 %	1.67
4	0.0099	21.27	55 %	2.35
8	0.0094	22.38	58 %	2.47

TABLE VI
EFFECT OF THE TWO-ROWS RECONSTRUCT

☞ Memory bandwith saving at the expense of the SU3 reconstruct yields a significant improvement

Intel® Xeon® Processor E5-2699 v4
Released in April 2016

- 22x2 = 44 cores
- 2.2 Ghz/core
- 3.6 GHz Boost
- Hyperthreading
- 256-bit vectors
- 256 Gb RAM
- 76.8 Gb/s
- 500 Gb disk
- 1.54 Tflops SP
- 0.78 Tflops DP

Hardware

CPU Name:	Intel Xeon E5-2699 v4
CPU Characteristics:	Intel Turbo Boost Technology up to 3.60 GHz
CPU MHz:	2200
FPU:	Integrated
CPU(s) enabled:	44 cores, 2 chips, 22 cores/chip, 2 threads/core
CPU(s) orderable:	1,2 chip
Primary Cache:	32 KB I + 32 KB D on chip per core
Secondary Cache:	256 KB I+D on chip per core
L3 Cache:	55 MB I+D on chip per chip
Other Cache:	None
Memory:	256 GB (16 x 16 GB 2Rx4 PC4-2400T)
Disk Subsystem:	1 x SATA, 500 GB, 7200 RPM
Other Hardware:	None

```
node   0   1   2   3
  0:  10  11  21  21
  1:  11  10  21  21
  2:  21  21  10  11
  3:  21  21  11  10
```

Fig. 3. Numa Specifications of our Hardware

#threads	t(s)	GFlops	Peak	%Peak	speedup
1	0.0300	7.02	8.8	80 %	1
2	0.0159	13.28	17.6	75 %	1.89
3	0.0115	18.31	26.4	69 %	2.61
4	0.0087	24.19	35.2	69 %	3.45
5	0.0072	29.41	44.0	67 %	4.19
6	0.0062	33.93	52.8	64 %	4.83
7	0.0056	37.58	61.6	61 %	5.35
8	0.0053	39.91	70.4	57 %	5.69
9	0.0050	42.06	79.2	53 %	6.00
10	0.0048	44.32	88.0	50 %	6.25
11	0.0044	48.02	96.8	50 %	6.81

TABLE VII
PERFORMANCE ON ONE INTEL-BROADWELL NODE

☞ Good performances on one NUMA node of the Broadwell-based system.

- Study and implement efficient NUMA awareness
- Investigate other SU3 compressions
- Evaluate the benefit of our implementation on a large-scale DM supercomputer
- Design a virtual communication topology close enough to the physical one

END & QUESTIONS