Accelerating FFT towards Exascale Computing

A. Ayala, S. Tomov, S. Cayrols, J. Li, G. Bosilca, J. Dongarra
Innovative Computing Laboratory

M. Stoyanov
Oak Ridge Nat. Lab.

*A. Haidar**
NVIDIA Corporation
** Contribution done while author was at ICL-UTK*

SPONSORED BY: U.S. DEPARTMENT OF ENERGY; NNSA National Nuclear Security Administration

1. Introduction

Many large-scale application-software require Fast Fourier Transforms (FFT), e.g., within the Exascale Computing Project (ECP) of the United States.

Hybrid CPU-GPU systems are widely used and are expected for the upcoming exascale machines. FFT libraries targeting such architectures have been accelerated via tuning and asynchronous kernel evaluation on GPUs [2], obtaining up to 2x speedup compared to fully CPU libraries.

We present techniques to further accelerate FFT computation by overcoming the communication bottleneck, we provide architecture-aware selection of FFT algorithm, a novel All-to-All routine (which can considerably speedup default MPI standard routines), and a mixed-precision implementation.

Our efforts led to ***heFFTe*** *library v2.0 [1], which is open-source, portable, scalable, robust; and targets exascale applications.*

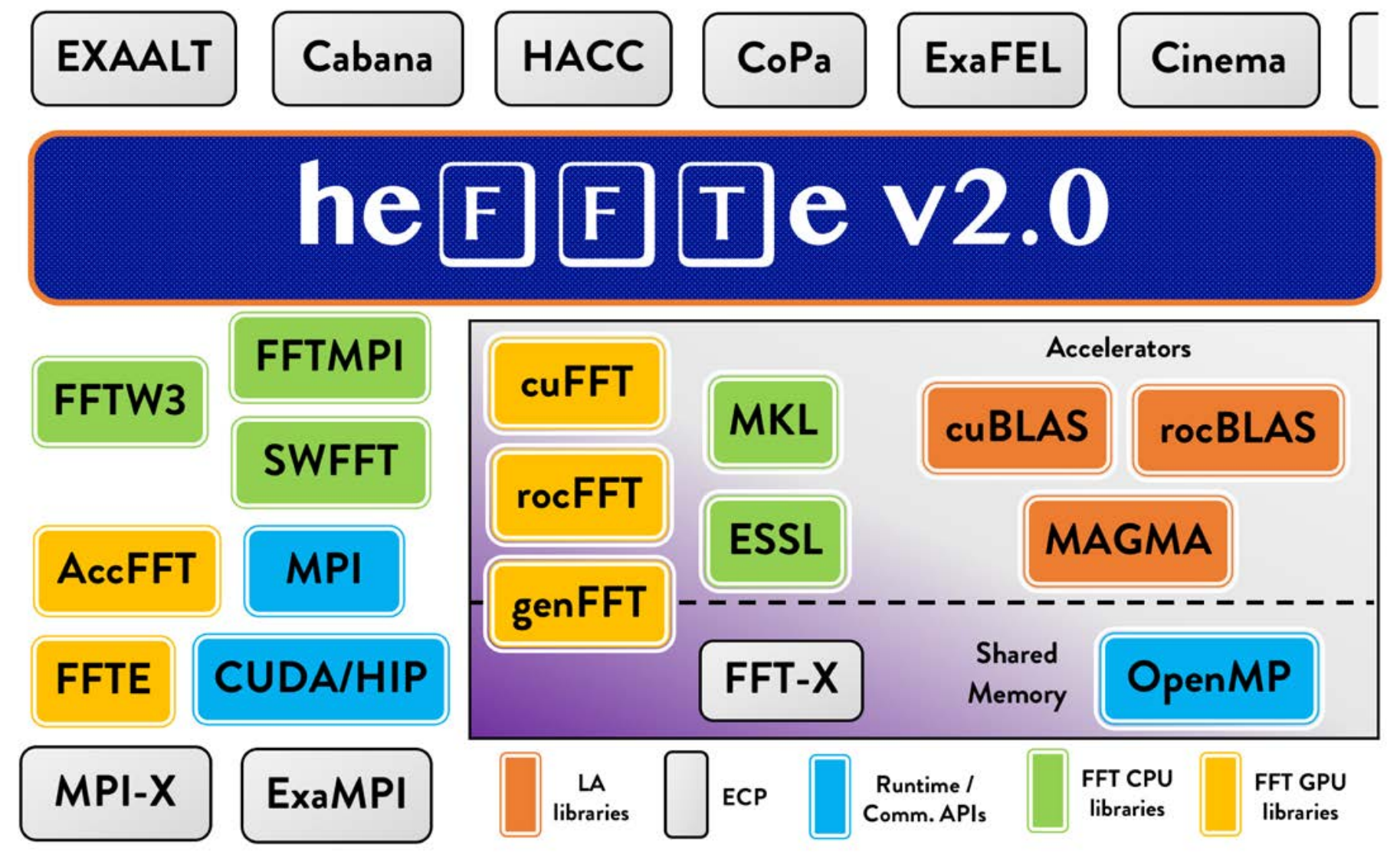


Fig 1. *heFFTe v2.0* in the ECP software stack.

2. Even faster FFTs

Multidimensional FFT are computed by a sequence of 1D or 2D FFTs, with intermediate data reshapes. The latter is the most expensive task (>90% of runtime [2]), moving data among processors (P), typically in an All-to-All fashion [2,3].

Reshape Type	# Messages
Brick $\Longleftrightarrow$ Pencil	$P^{1/3}$
Brick $\Longleftrightarrow$ Slab	$P^{2/3}$
Pencil $\Longleftrightarrow$ Pencil	$P^{2/3}$
Pencil $\Longleftrightarrow$ Slab	$P^{2/3}$
Slab $\Longleftrightarrow$ Slab	P

Fig 2.1. Cost for data reshaping

heFFTe supports any type of reshaping technique (c.f., Fig. 2.1) and provides a tool to create architecture-aware ***Phase diagrams*** *[1,2]. In Fig. 2.2., we show the case of Summit, where users can input their resources and FFT size to select the fastest reshape approach.*

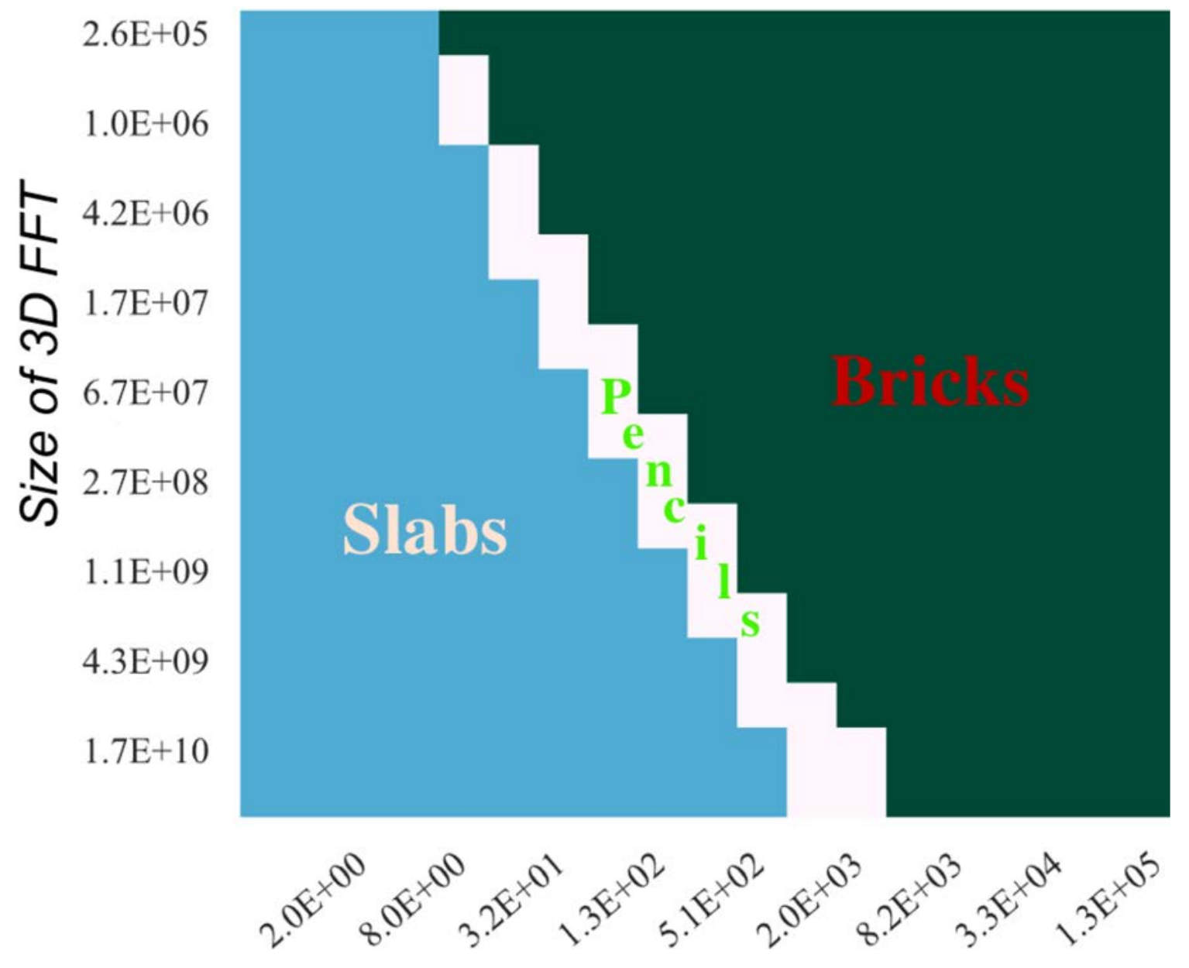


Fig 2.2. Phase diagram for algorithm tuning

Current MPI_Alltoall distributions for GPUs perform poorly compared to theoretical peaks [3].

NVDIA collective library (NCLL) does not have an All-to-All option.

Hence, we developed a novel algorithm, based on One Sided Communication (OSC_A2A). It can achieve up to 30% speedup.

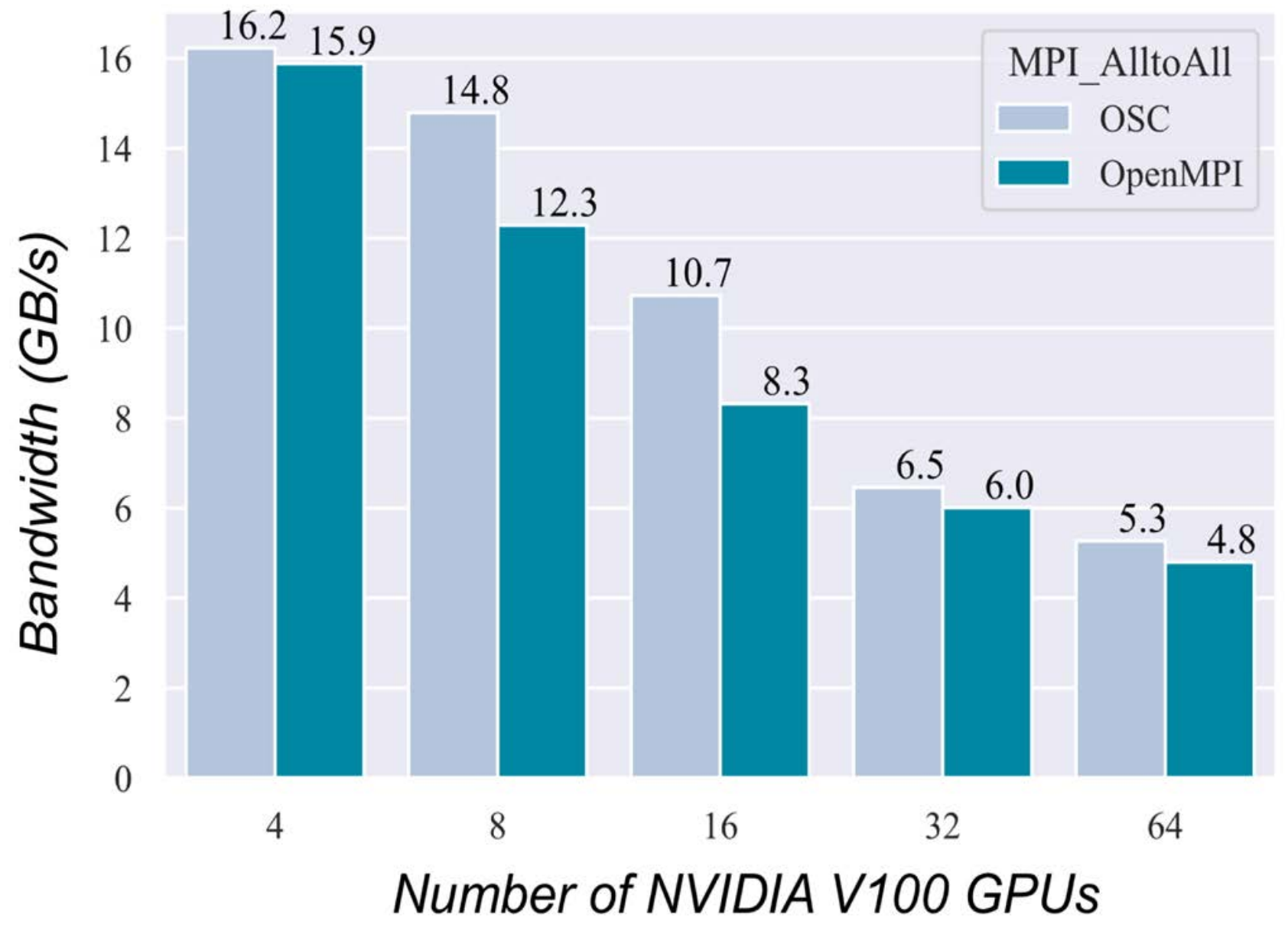


Fig 2.3. Average throughput of all-to-all exchange of 2GB of data, 1GPU per node.

3. Mixed-Precision FFT

Fig. 3, shows that *heFFTe* linearly scales. We use a 3D complex-data grid, and compute both: single (FP32) and double (FP64) precision FFTs.

We developed a mixed-precision version for *heFFTe*, which exploits GPU power to compress data (using casting/ZFP) to save in All-to-All cost (which usually takes over 90% of runtime [2,3]). We used a ring version of our OSC_A2A routine.

CR2 means a compression ratio of 2 times. CR2 is up to 2.6x faster than FP64 and up to 1.4x faster than FP32. CR4 is up to 5x faster than FP64 and up to 2.6x faster than FP32. CR2 validation error is $\mathcal{O}(10^{-7})$, while for FP32 it is $\mathcal{O}(10^{-6})$; i.e., we move the same data volume faster, while getting a better quality FFT output.

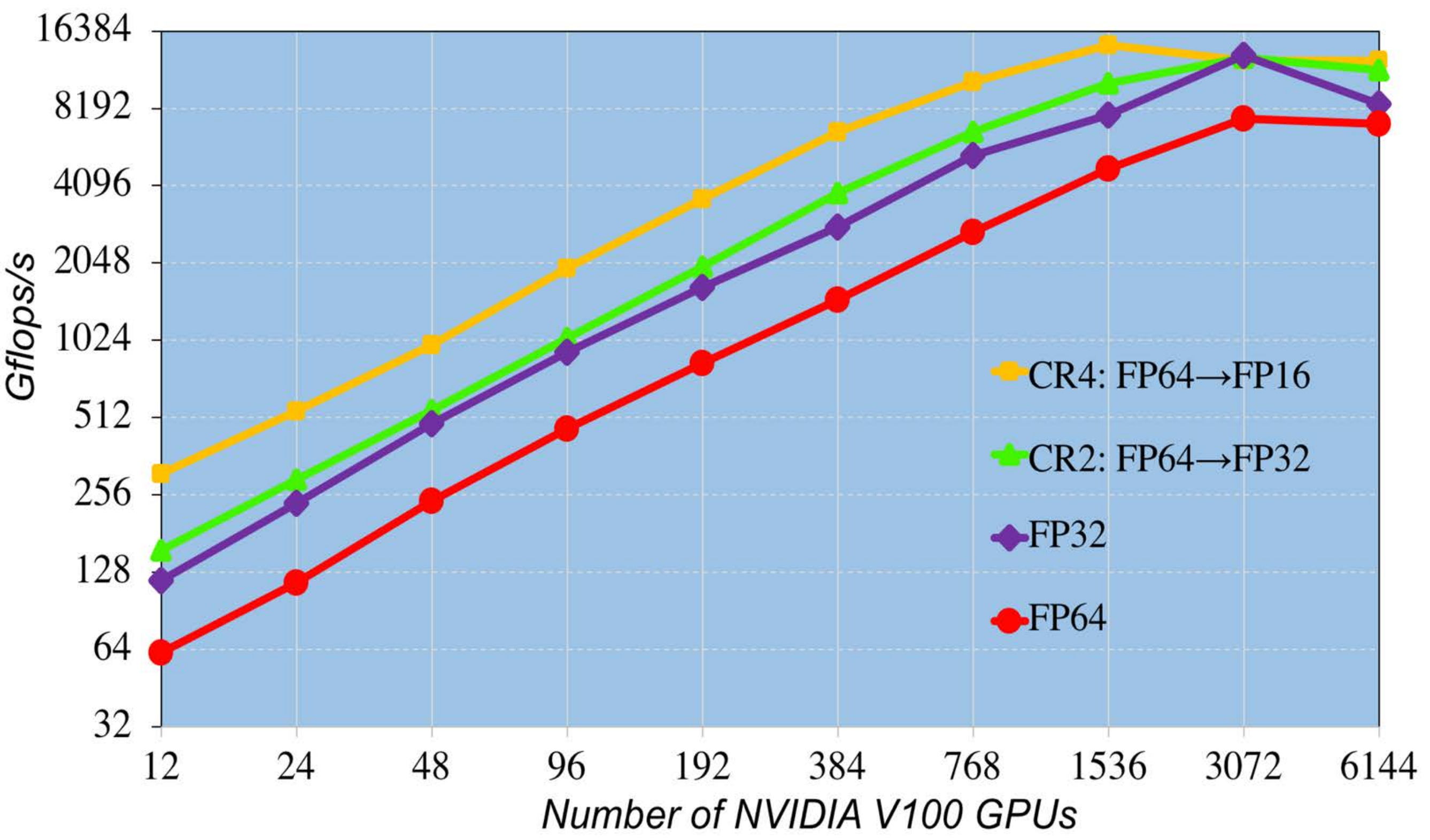


Fig3. Strong scalability of *heFFTe on a* 1024^3 FFT, using 6GPUs per Summit node.

4. Communication model

We introduce a novel communication model for hybrid-distributed FFTs which can adapt to any architecture [2], and gives a theoretical estimation of the reshaping cost.
This model assumes that fast communication is available within a node, e.g., Summit at ORNL, which has NVLINK connections.

For Summit, we get [2]:

$$T_{comm} = \frac{7.8125\, P \log(N)}{\#\ of\ reshapes},$$

where N is the FFT size.

Using this model, we derive a roofline model theoretical peak performance. Fig. 4 shows heFFTe achieving about 90% of peak value up to 48 GPUS; this percentage then decreases, indicating that too many resources are being used for the given FFT size (↑latency).

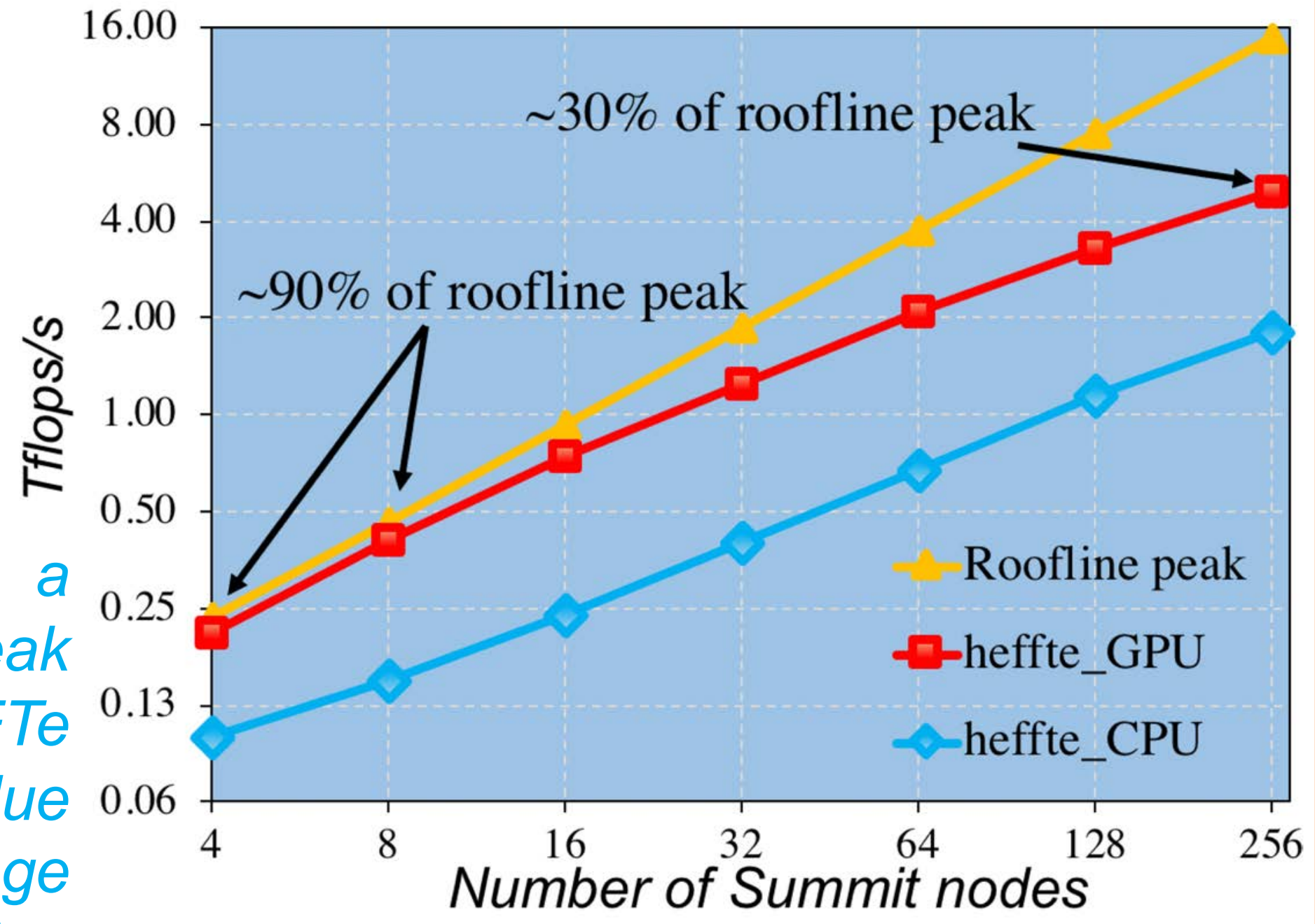
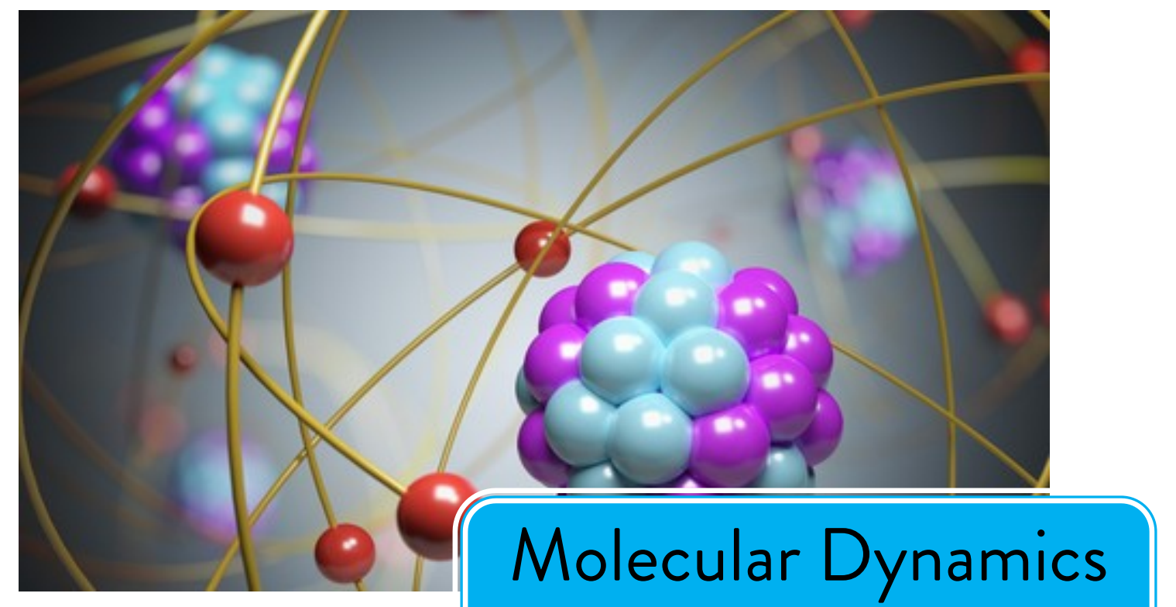


Fig 4. Roofline & scalability for *heFFTe*, *GPU version uses 6 Volta100 GPUs per node, CPU version uses 40 IBM Power9 per node.*

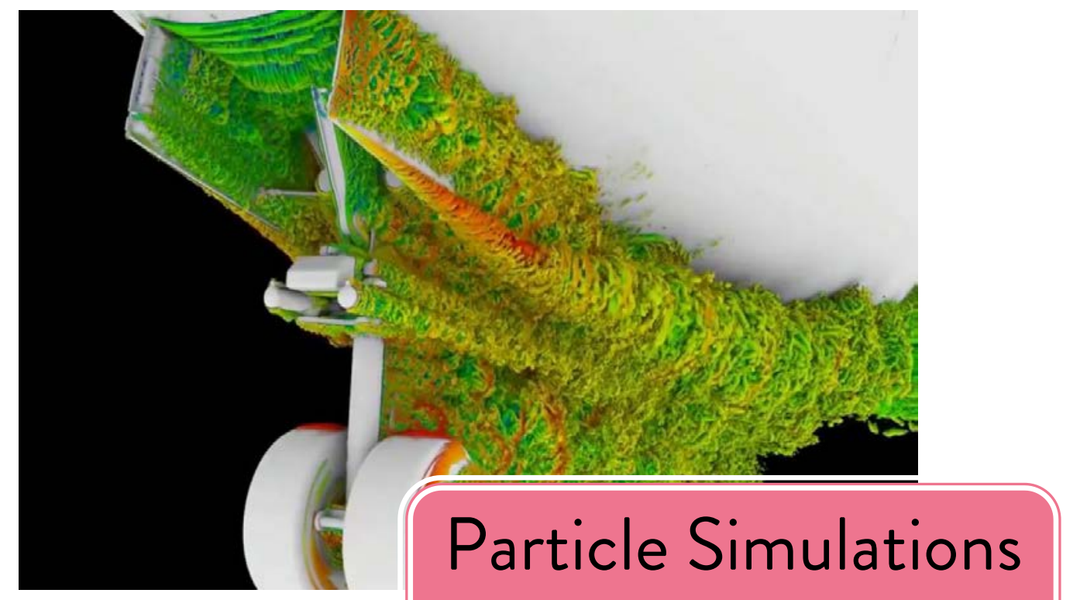
5. Applications

Following figures show some applications *heFFTe* targets, to some of which it has already been integrated, to accelerate FFT calculus while ensuring scalability.

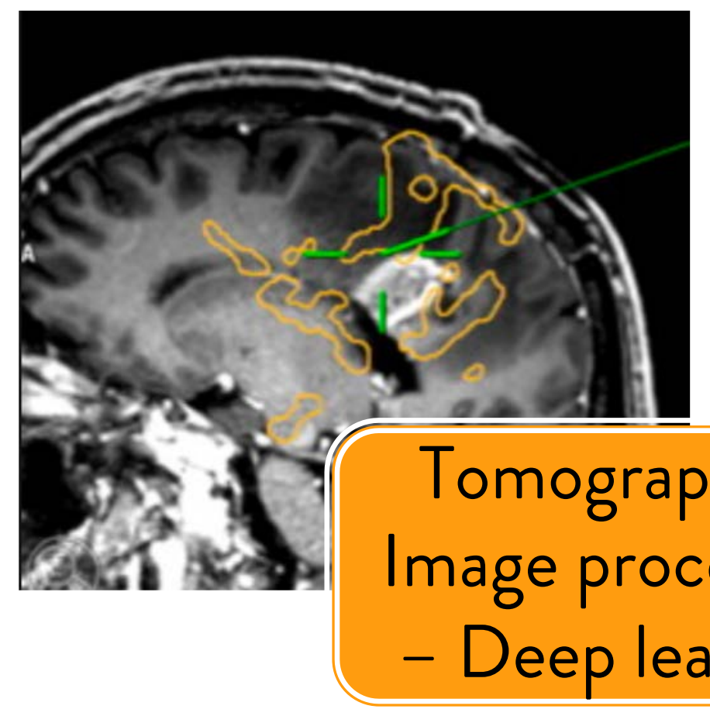
Cosmology
HACC



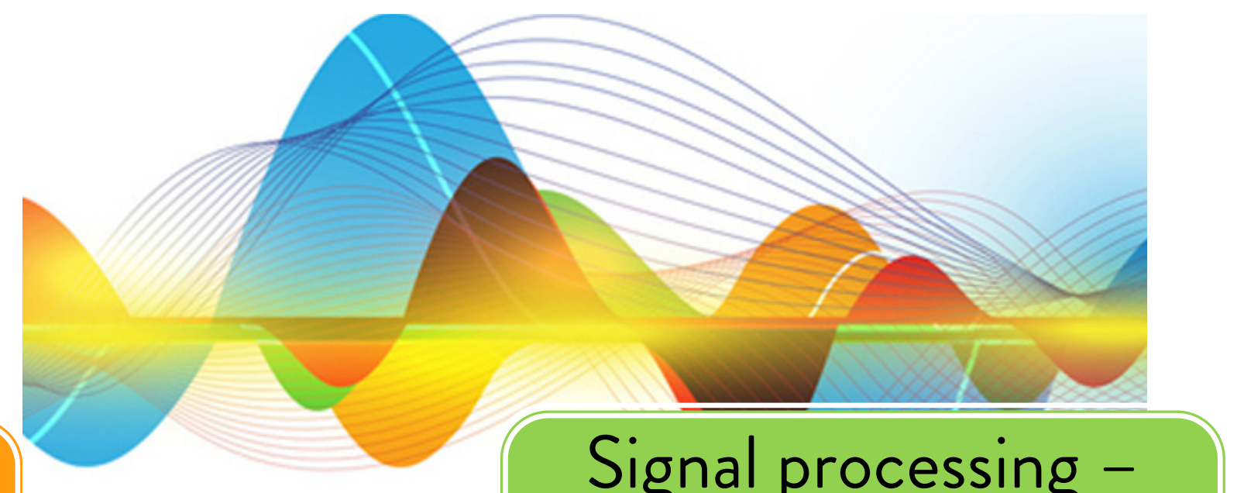
Molecular Dynamics
EXAALT



Particle Simulations
CoPa



Tomography –
Image processing
– Deep learning



Signal processing –
Randomized FFT –
Matrix Compression

6. References

[1] https://bitbucket.org/icl/heffte/

[2] A. Ayala, S. Tomov, A. Haidar, J. Dongarra: *heFFTe: Highly Efficient FFT for Exascale*, Lecture Notes on Computational Sciences – Springer, 2020.

[3] A. Ayala, X. Luo, S. Tomov, H. Shaiek, A. Haidar, G. Bosilca, J. Dongarra: *Impacts of Multi-GPU Collective Communications on Large FFT Computation*, IEEE/ACM Exascale MPI, 2019.