OpenACC

A QUALITATIVE STUDY

Introduction

Directives-based programming model for **parallel computing**

Add Simple Compiler Directive

```
main()
 {
  <serial code>
  #pragma acc kernels
  {
    <parallel code>
  }
}
```

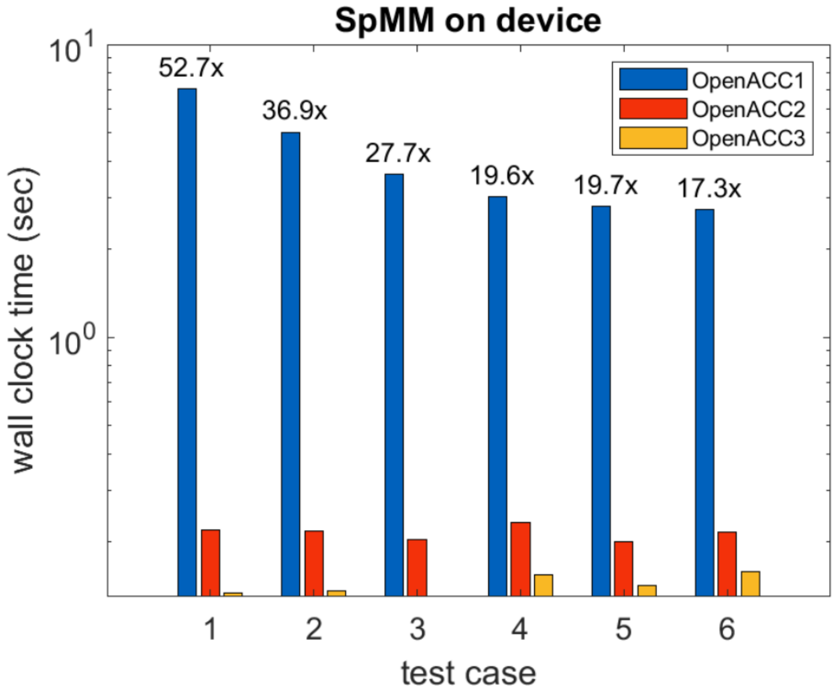
SIMPLE

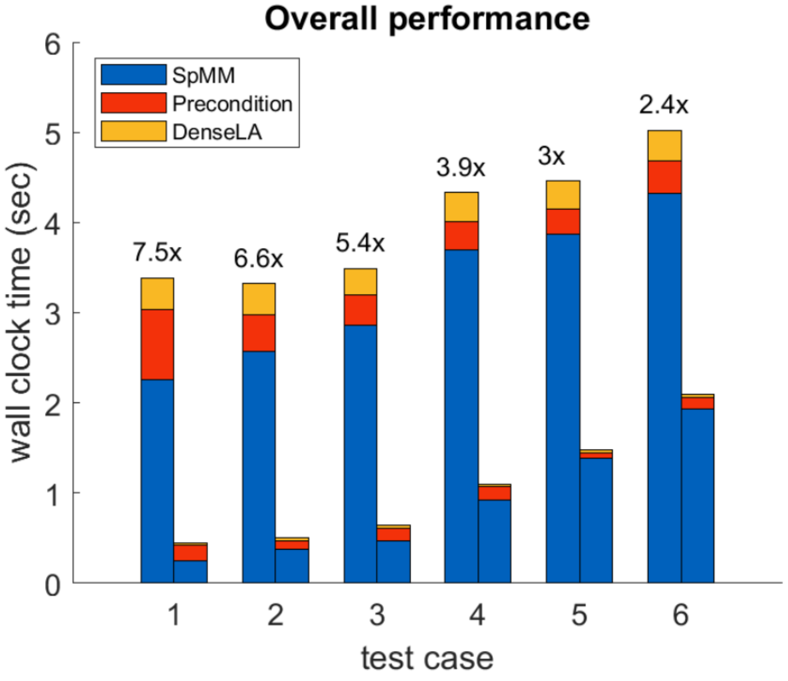
Designed for **performance** and **portability** on CPUs and GPUs

POWERFUL & PORTABLE

Many-Fermion Dynamics-nuclear

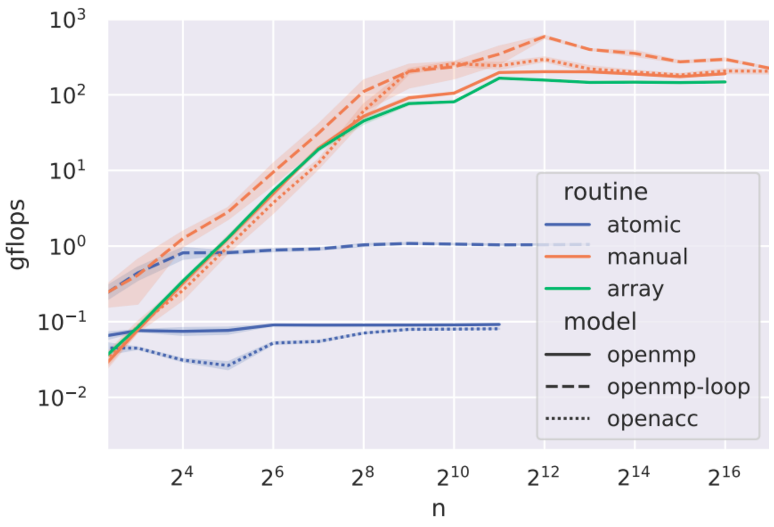
1. Fortran 90
2. Eigenpairs calculation using LOBPCG method
3. LOBPCG uses Sparse Matrix Matrix multiplication
4. Implementation already existed for OpenMP



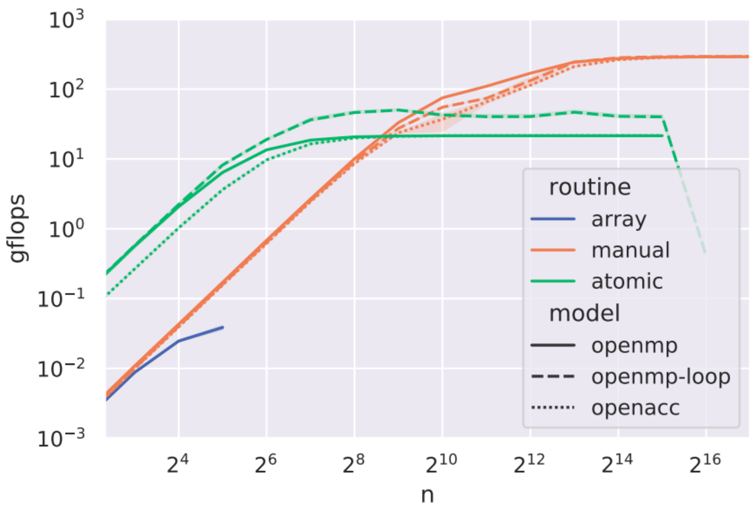


Many-Fermion Dynamics-nuclear

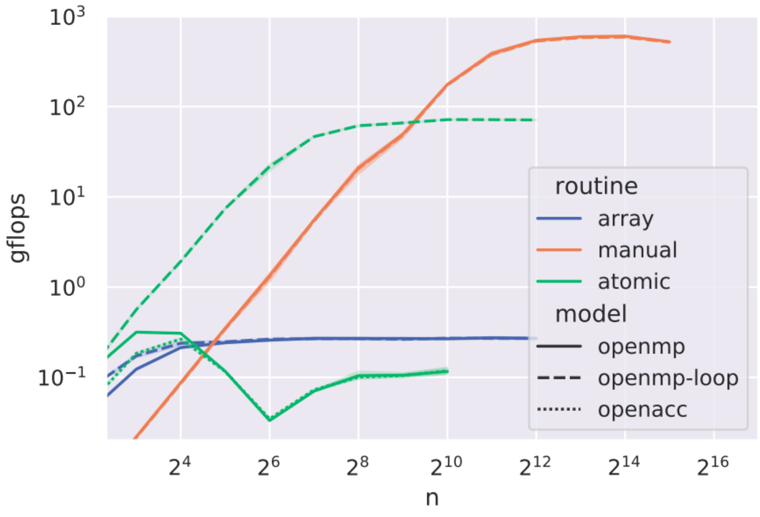
Array Reductions



Skylake CPU



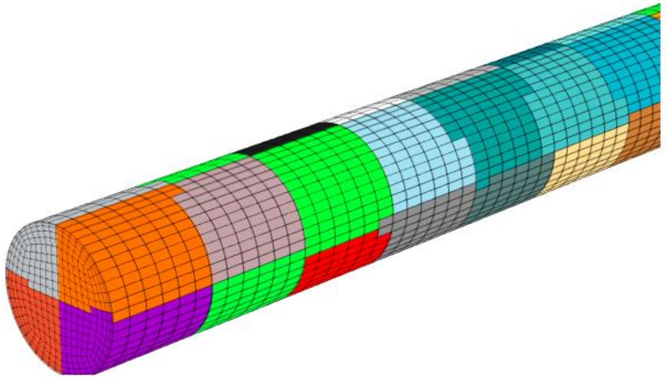
NVIDIA A100 (GPU)



NVIDIA MI100 (GPU)

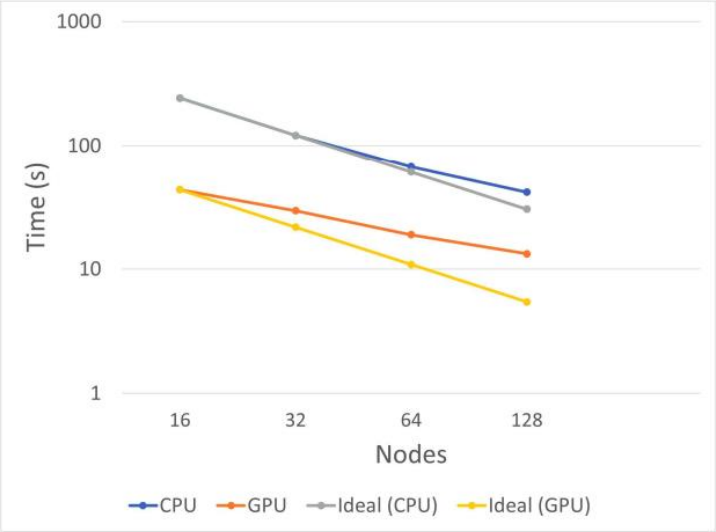
Nek5000

- High fidelity Computational Fluid Dynamics (CFD)
- Simulation of air around Airplanes
- Simulation of ocean currents
- Already built in the 1980s using MPI
- Fortran 77
- Spectral element method ➔ partitioning the Volume into smaller ones
- Volumes are representented by Langrange polynom

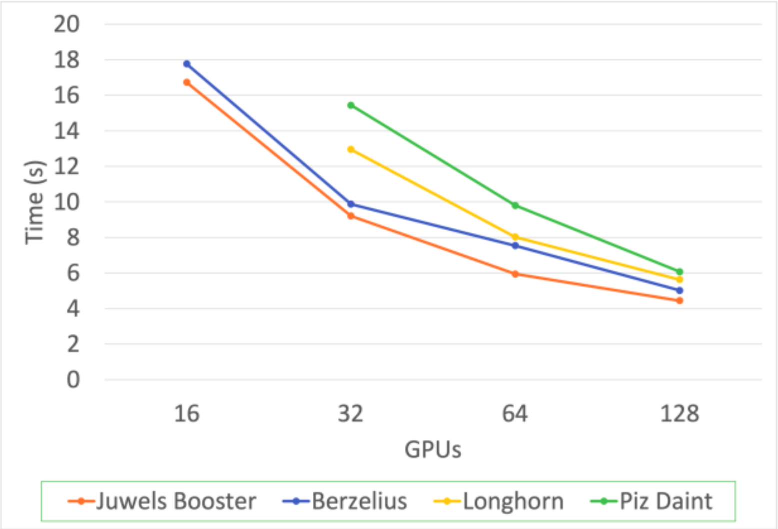


Partitioning of a pipe simulation into 64 elements

Nek5000



CPU versus GPU performance on Juwels Booster



Time for Reynolds number Re=360 and maximal polynomial order of N=7

FluTAS

- Navier-Stokes-equation solver
- Navier-Stokes-equations model fluid-fluid Interactions, fluid-gas interactions etc.
- Written in Fortran 90 with MPI for CPU parallelization and OpenACC for GPU parallelization

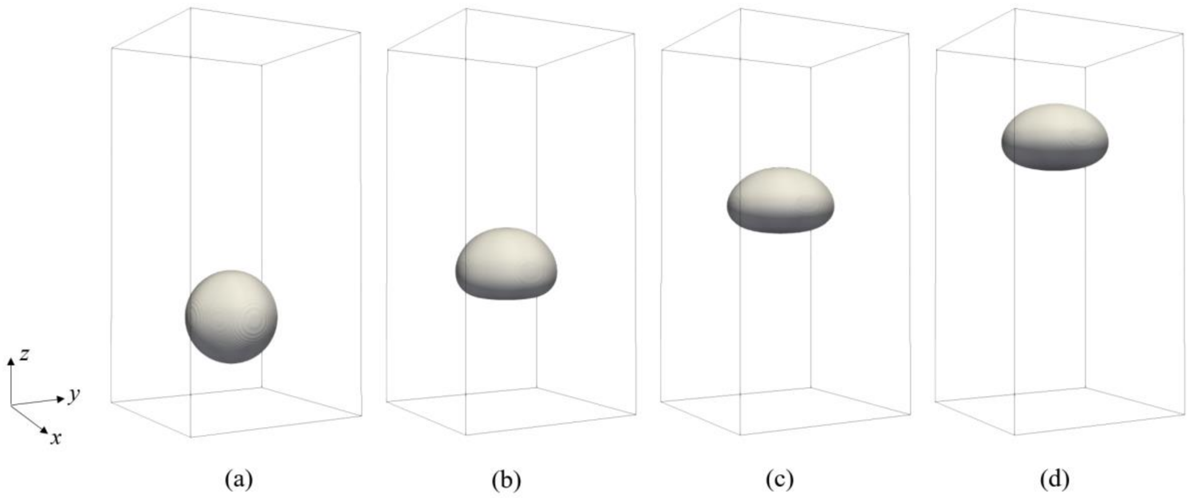
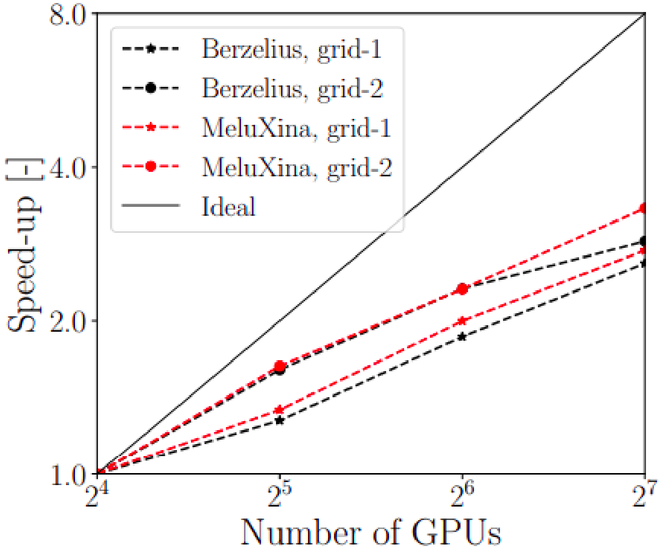
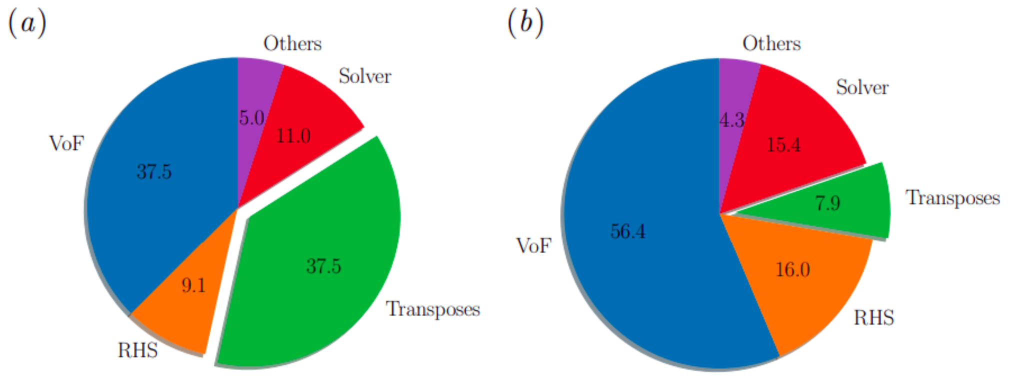


Figure 3: Isosurfaces of $\phi = 0.5$ at dimensionless times $t\sqrt{|\mathbf{g}|/d_0}$ (a) 0, (b) 1.4, (c) 2.8 and (d) 4.2.

FluTAS

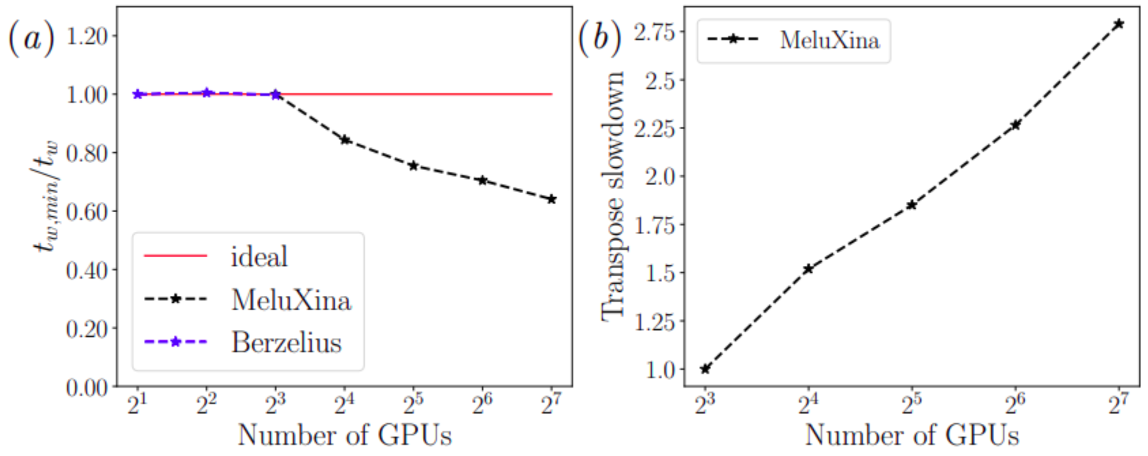


Strong Scaling test



Time spent in each subarea

a) GPU
b) CPU



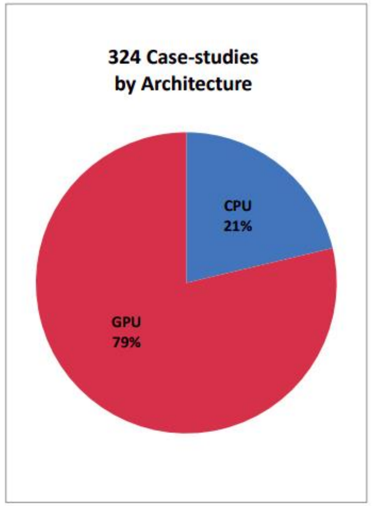
a) Weak scaling
b) Transpose slowdown

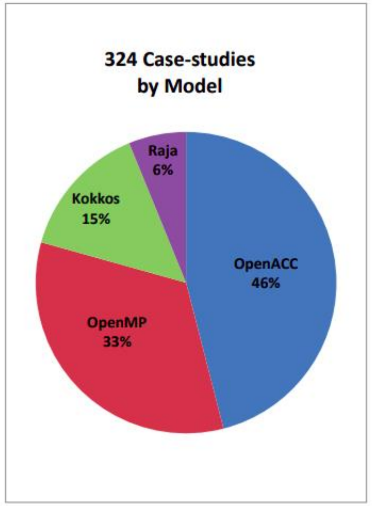
Performance Portability

$$\bar{\Phi}_{M_M} = \frac{\sum_{i \in T} e_i(a,b,c)}{|T|}$$

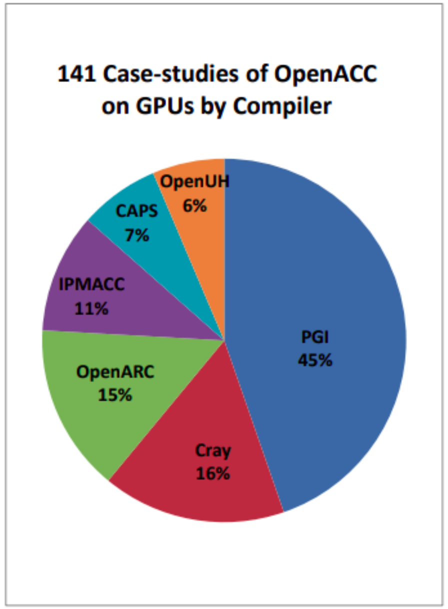
$$e_i(a,b,c) = \frac{OpenACC\ Performance}{CUDA\ Performance}$$

- M = Model
- T = case study
- a = application
- b = problem
- C = platform





Performance Portability



Performance Portability

	Exc. outliers					Inc. outliers					# of outliers		
Model	Case Studies	$\bar{\Phi}_M$	std. dev.	max	min	Case Studies	$\bar{\Phi}_M$	std. dev.	max	min	< 50%	50% - 100%	> 100%
CPU													
OpenACC	3	**71%**	10%	85%	60%	8	**105%**	29%	148%	60%	0	3	5
OpenMP	21	**88%**	13%	100%	60%	25	**97%**	39%	226%	42%	1	21	3
Kokkos	15	**83%**	15%	98%	51%	27	**92%**	47%	230%	13%	5	15	7
RAJA	5	**82%**	9%	100%	71%	9	**109%**	66%	216%	11%	1	5	3
GPU													
OpenACC	109	**81%**	13%	100%	51%	141	**78%**	26%	200%	3%	23	109	9
OpenMP	62	**81%**	14%	100%	52%	83	**77%**	25%	145%	7%	15	63	6
Kokkos	14	**86%**	12%	100%	58%	20	**85%**	23%	138%	28%	2	14	4
RAJA	6	**85%**	13%	100%	63%	11	**80%**	25%	103%	28%	2	7	3

Summary

- OpenACC is directive based programming model
- Promises ease of use
- Porting from OpenMP is easy
- Has good Performance Portability of around 80%

(similar to other Projects)

- Speedups range between factor 2 to 10
- No magic tool ➔ still requires to optimize manually
- Same drawbacks as native ports, e.g. CUDA

➔ Limit memory bottlenecks and/or communication overhead

OpenACC

More Science, Less Programming

Sources

Slide 2:

https://www.openacc.org/blog/evaluating-performance-openacc-gcc, accessed on 1. Jul 2022

Slide 3-4:

Brandon Cook, Patrick J. Fasano, Pieter Maris, Chao Yang, and Dossay Oryspayev. Accelerating quantum many-body configuration interaction with directives. CoRR, abs/2110.10765, 2021.

Pieter Maris, Chao Yang, Dossay Oryspayev, and Brandon Cook. Accelerating an iterative eigensolver for nuclear structure configuration interaction calculations on gpus using openacc. CoRR, abs/2109.00485, 2021.

Slide 5-6:

Jonathan Vincent, Jing Gong, Martin Karp, Adam Peplinski, Niclas Jansson, Artur Podobas, Andreas Jocksch, Jie Yao, Fazle Hussain, Stefano Markidis, Matts Karlsson, Dirk Pleiter, Erwin Laure, and Philipp Schlatter. Strong scaling of openacc enabled nek5000 on several GPU based HPC systems. CoRR, abs/2109.03592, 2021.

Slide 7-8:

Marco Crialesi-Esposito, Nicolo Scapin, Andreas D. Demou, Marco Edoardo Rosti, Pedro Costa, Filippo Spiga, and Luca Brandt. Flutas: A gpu-accelerated finite difference code for multiphase flows, 2022.

Slide 9-10:

Ami Marowka. On the performance portability of openacc, openmp, kokkos and raja. In International Conference on High Performance Computing in Asia-Pacific Region, HPCAsia2022, page 103–114, New York, NY, USA, 2022. Association for Computing Machinery.

Slide 11:

https://www.codee.com/training/courses/openacc-programming-course/, accessed on 1. Jul 2022