EPICURE

Unlocking European-level HPC Support

EPICURE HPC in ARM Architecture Hackathon

Assessing the Performance and Scalability of ARM Processors for Industrial Applications

Gabriel Marcos Magalhães

INESC TEC / PIEP

UNIVERSIDADE DO MINHO

INESCTEC

piep

iPC INSTITUTE FOR POLYMERS AND COMPOSITES

Agenda

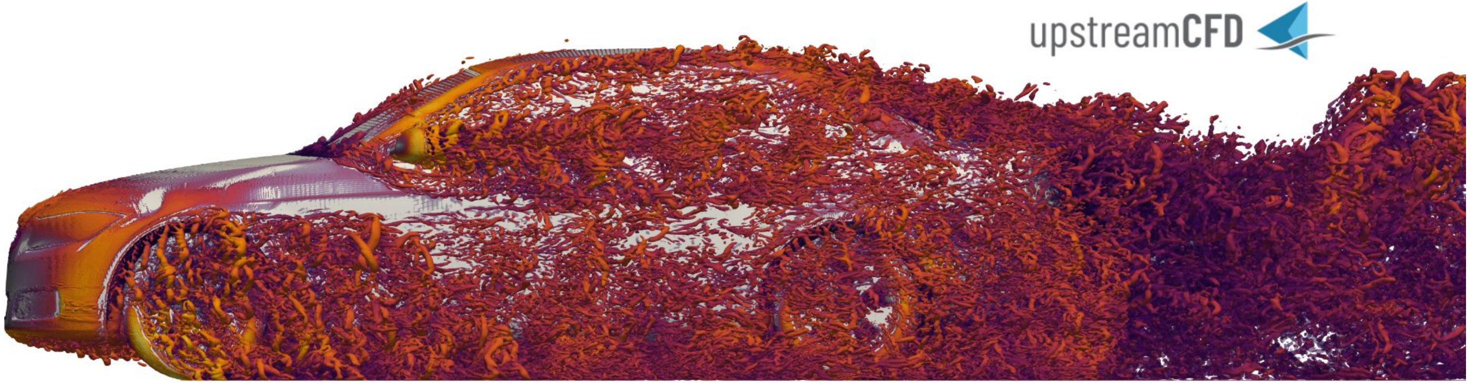
- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

Agenda

- **Introduction**
  - **What is CFD?**
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

What is CFD?

- **Computational fluid dynamics** (CFD) is the science of using computers to **predict fluid flows** based on the **governing equations**, which are most often the conservation of mass, momentum, and energy.
- CFD allows for the analysis of **various aspects of fluid flow**, such as temperature, pressure, velocity, and density, and can be applied to a **wide range of engineering problems** across industries.

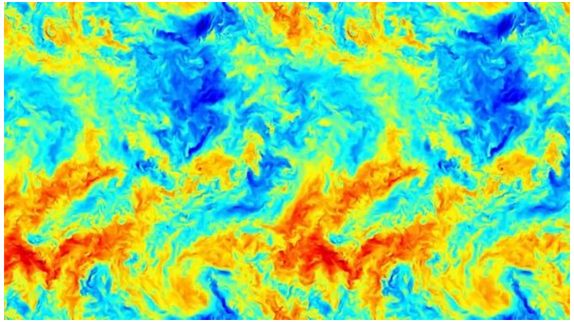


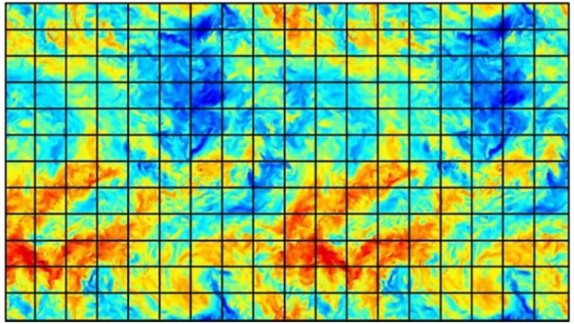
Fonte: [Ansys Blog Post: What is Computational Fluid Dynamics (CFD)?]

How CFD works

*Grid based Methods

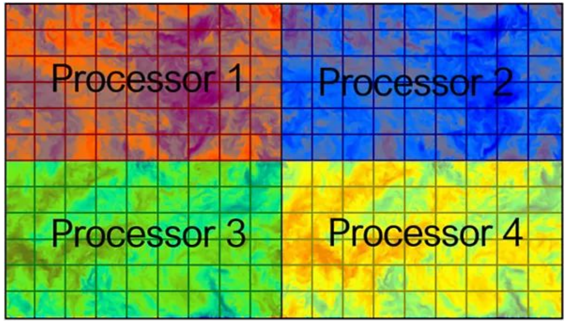
Identify the fluid flow domain to be solved





Discretize the domain into the desired mesh size or grid spacing*

Assign processors to different regions and solve the associated algebraic system of equations



Fonte: Ansys Blog Post: What is Computational Fluid Dynamics (CFD)?

How to use HPC in CFD

- Select one of the available CFD codes

Commercial

ANSYS

STAR-CCM+

Moldex3D

COMSOL

SIMULIA

Open-source

SU2 code

NEKTAR++ SPECTRAL/HP ELEMENT FRAMEWORK

OpenFOAM

Agenda

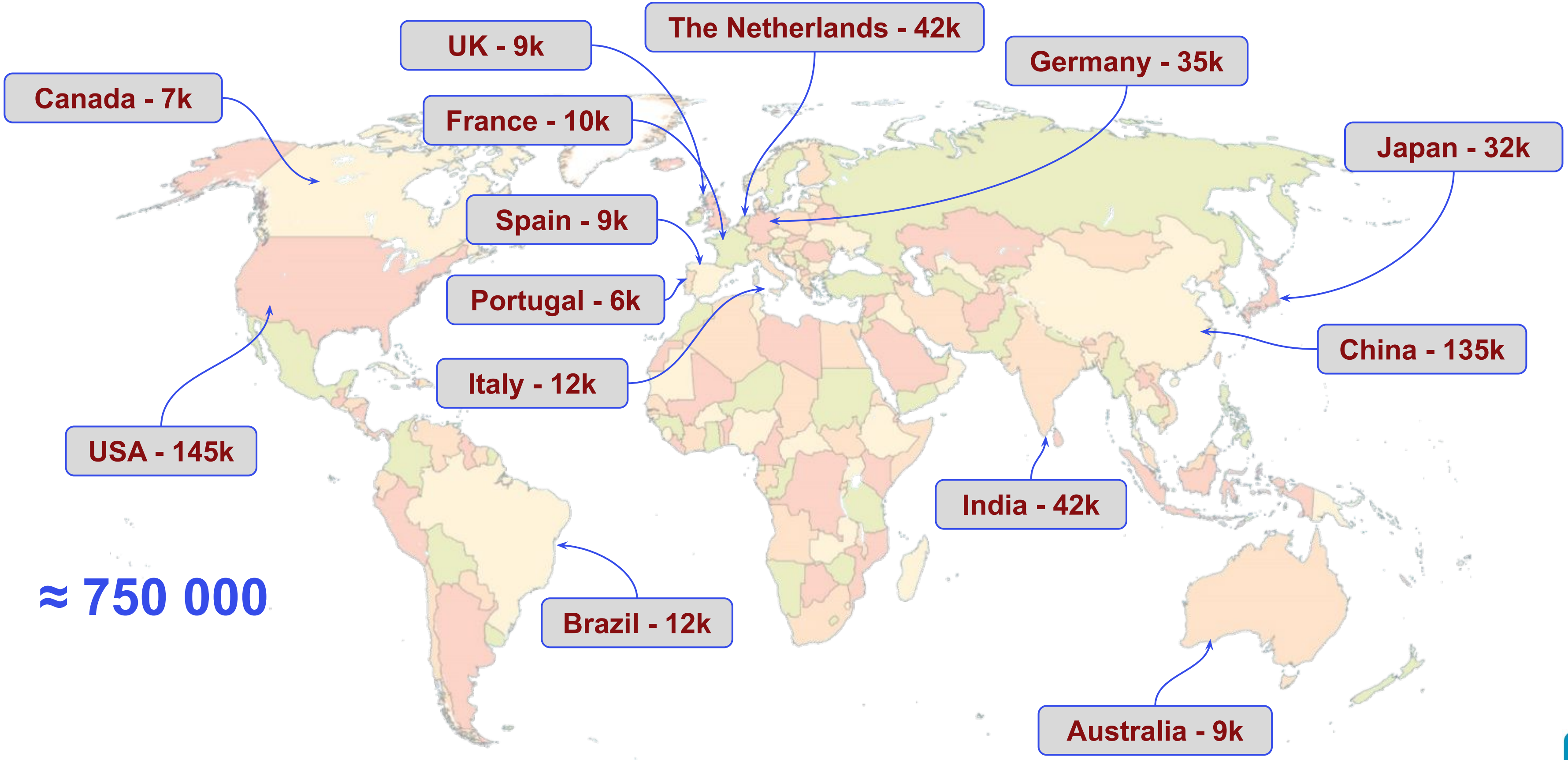
- **Introduction**
  - What is CFD?
  - **What is OpenFOAM?**
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

What is OpenFOAM?

- **Open**-Source **F**ield **O**peration **A**nd **M**anipulation
- Completely **free** (possibility to implement new features)
- Uses C++ language
- Wide range of applications
- Native tools for pre and post processing
- Large and worldwide **active community**
- Works under **GPL 3 licence**
- **Tested, validated** and **evaluated** (performance) in different fields to simulate complex problems
- No limitation in HPC (no limitation on the number of cores)

Is there a World Record?

Downloads during 2023 of the three major forks



≈ 750 000

Use in industries

Automobilistic

TOYOTA

MAN

Polymers

SOPREFA

Logoplaste

Formula 1 Teams

AMG PETRONAS FORMULA ONE TEAM

Sauber F1 Team

Other Industries

IBM

FM Global

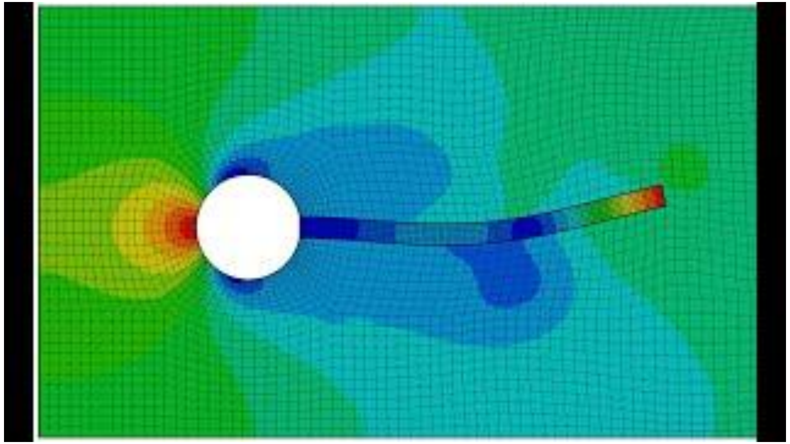
gm

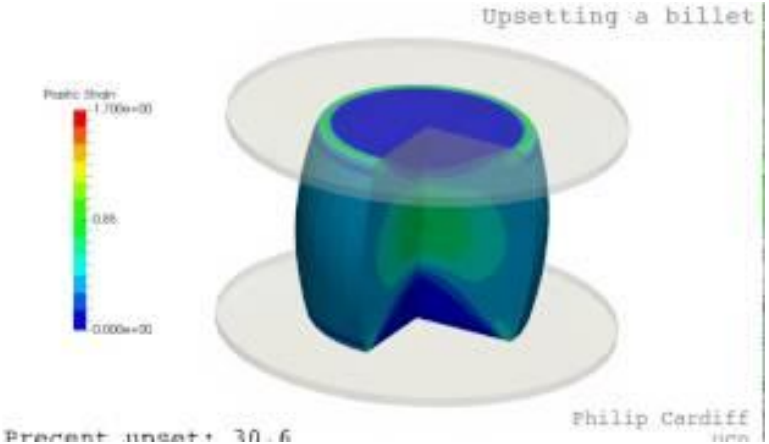
SIEMENS energy

TMG GROUP

Applications

Solid Mechanics



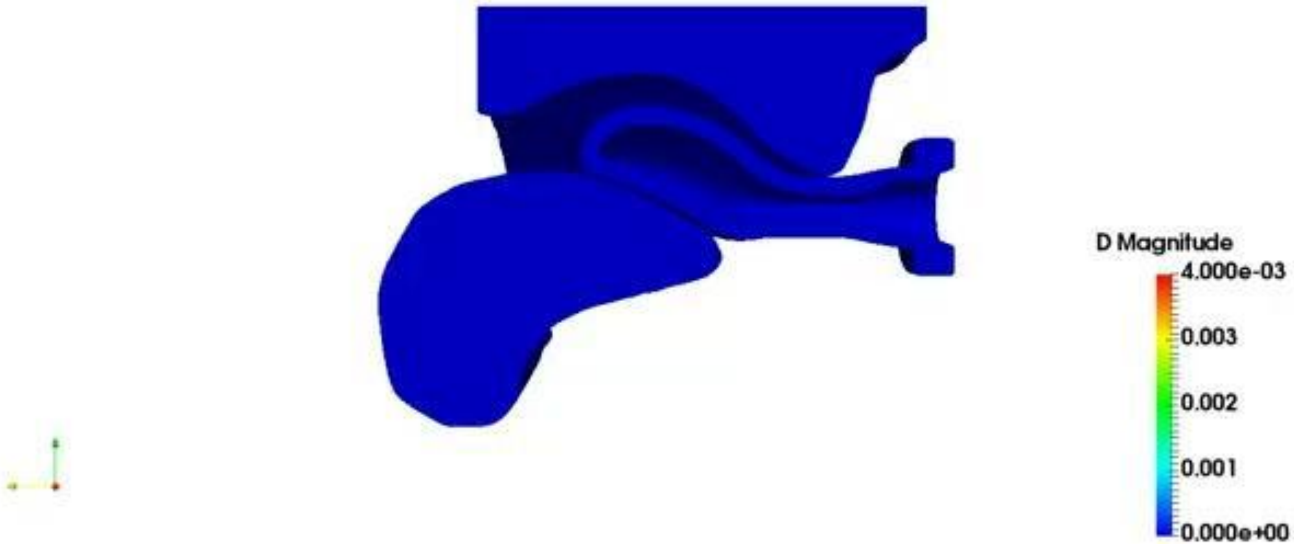


- Fluid-solid interaction between an elastic object and laminar incompressible flow
- Upsetting a billet

Fonte: [Solids4Foam YouTube channel] and [Philip Cardiff YouTube Channel]

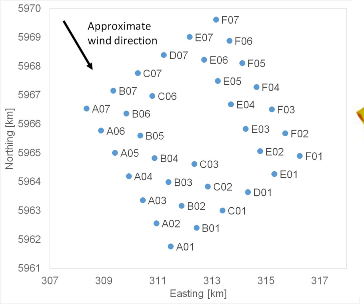
Applications

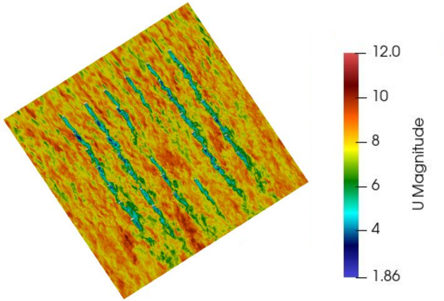
Solid Mechanics (Pacifiers Assessment)

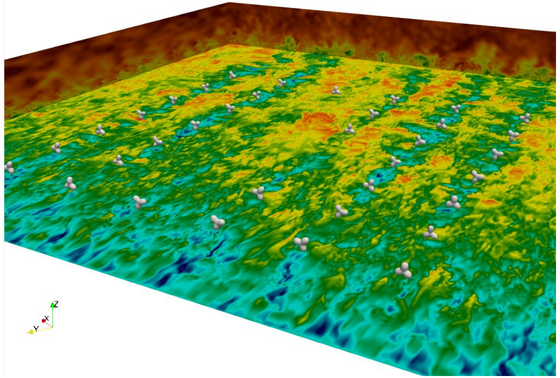


Applications

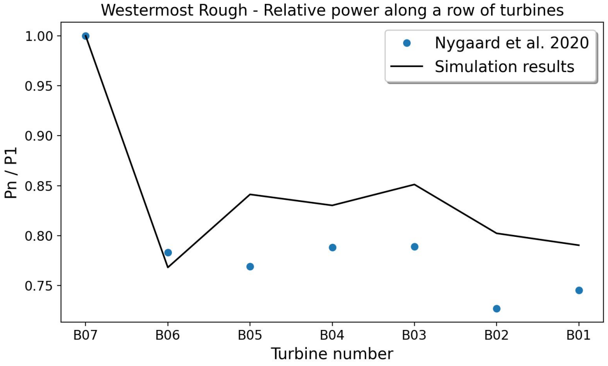
Offshore Wind Park







- 35 Siemens 6 MW turbines with 154 m rotor diameter
- 13 million cells
- LES turbulence model

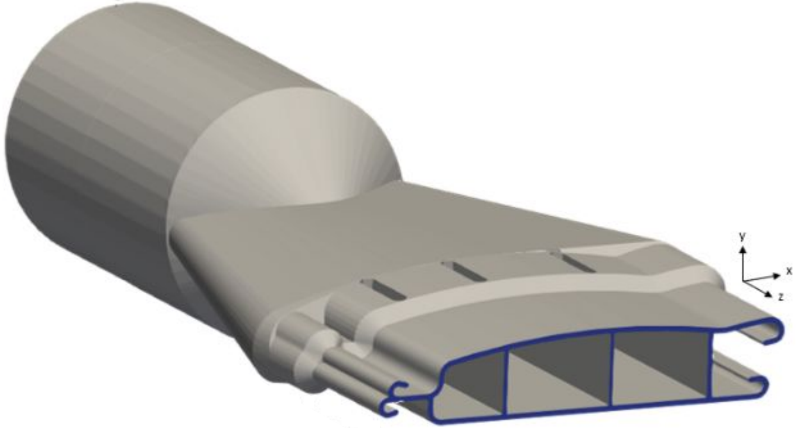


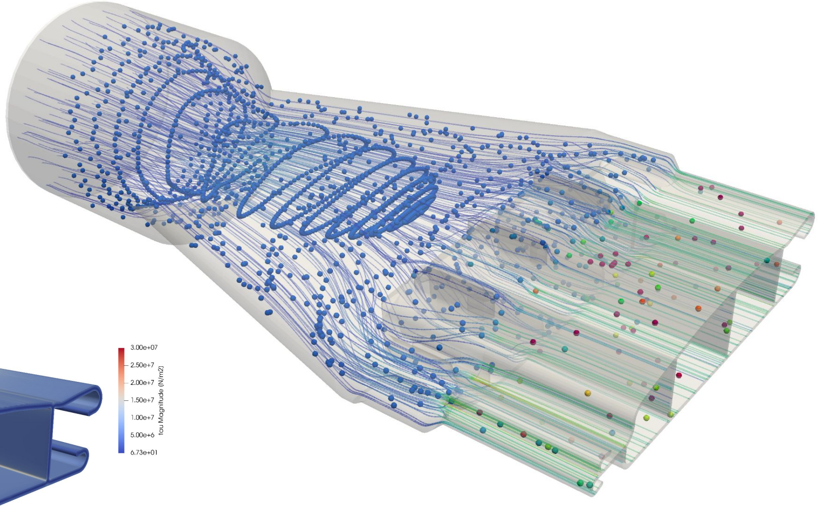
Fonte: [Case source (exaFOAM repository)]()

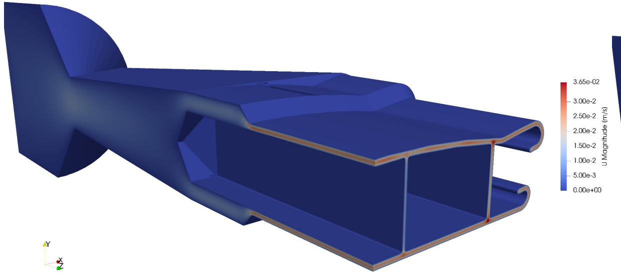
Applications

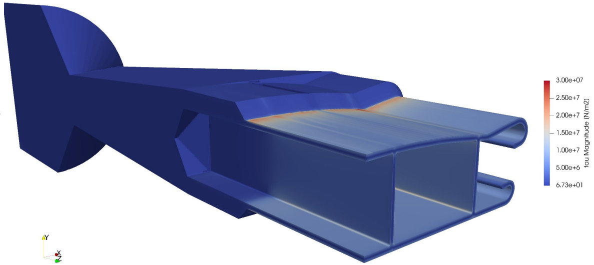
Viscoelastic Complex Profile Extrusion

- Real multi-mode viscoelastic fluid (6)
- 40 variables per cell
- 20 million cells





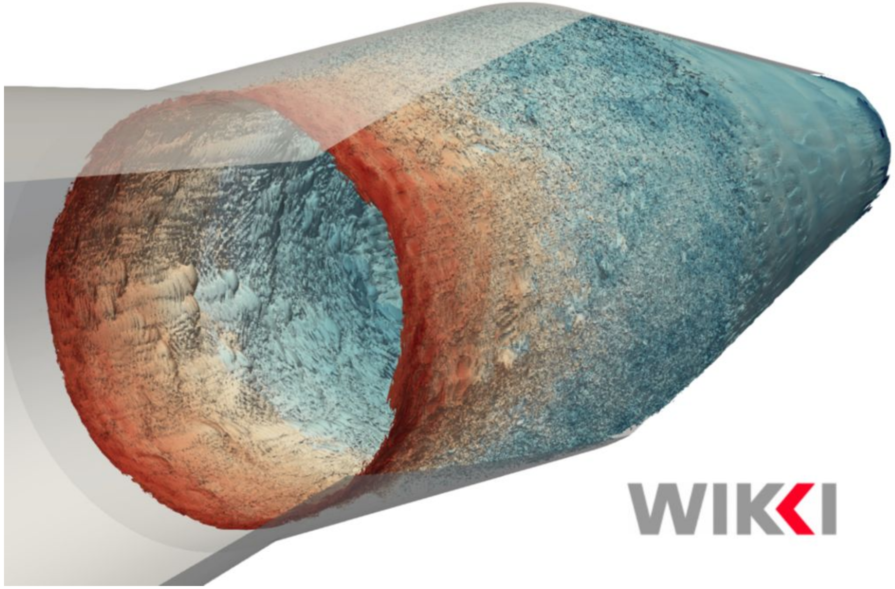




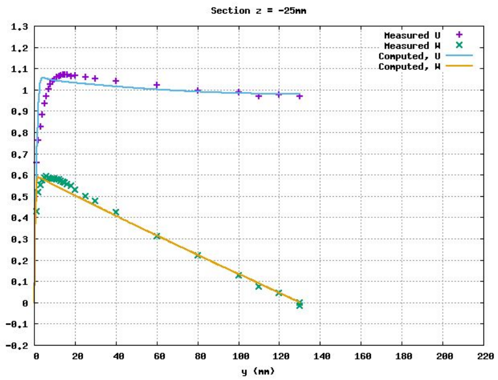
Fonte: [Case source (exaFOAM repository)]

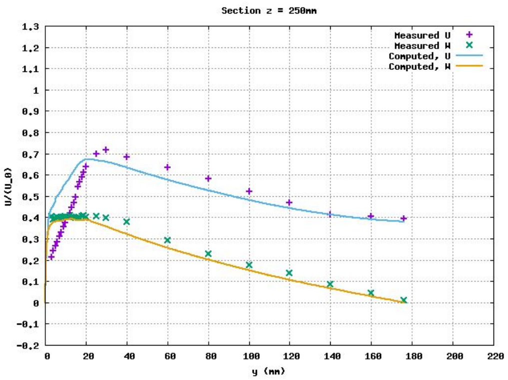
Applications

ERCOFTAC Conical Diffuser



- Turbulence (LES and RANS Model)
- Two configurations:
  - 3M cells run on 28 cores, 1 node;
  - 145M cells run on 2880 cores, 30 nodes.

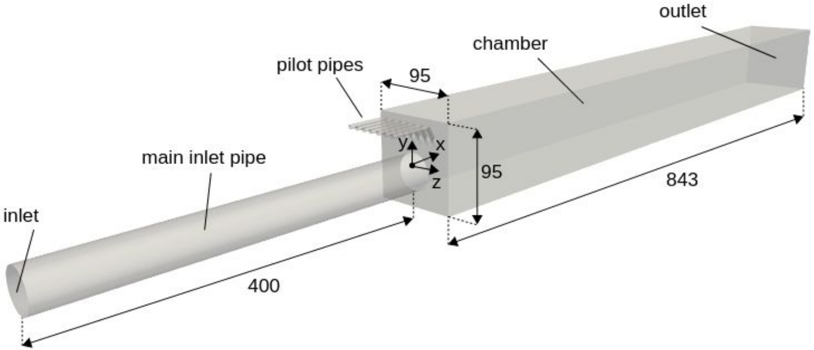




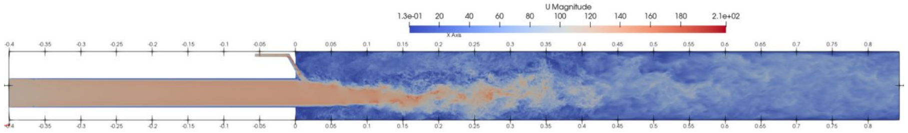
Fonte: Case source (exaFOAM repository)

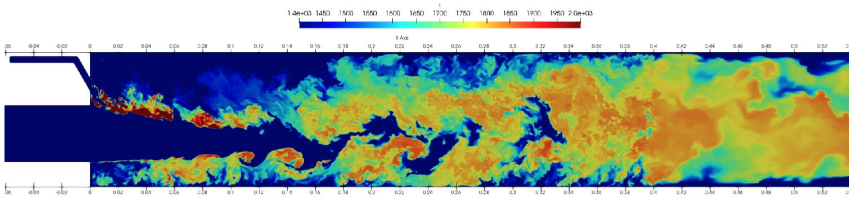
Applications

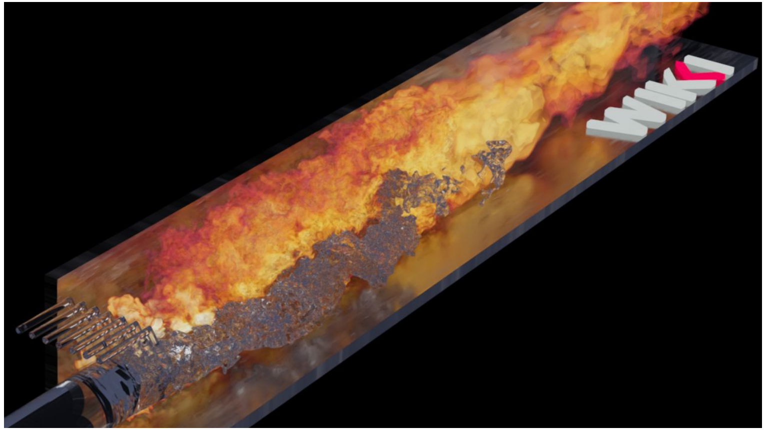
DLR Confined Jet High Pressure Combustor



- Application to the gas turbines
- Combustion: fuel mixture is inhomogeneous
- Turbulence model: LES + van Driest
- 489M cells
- 4096 cores, 32 nodes







Fonte: [Case source (exaFOAM repository)]

Agenda

- **Introduction**
  - What is CFD?
  - What is OpenFOAM?
  - **History of our studies in Deucalion**
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

History of our studies in Deucalion

- Move to Portugal in 2023 beginning to work with Miguel Nóbrega in University of Minho

Miguel Nóbrega (UMinho)

- One of the most recognized names both in Portugal and worldwide
- One of the founders and editors of OpenFOAM Journal
- Big enthusiast of OpenFOAM and HPC
- Coordinator of one Work package in the exaFOAM project

History of our studies in Deucalion

The exaFOAM project

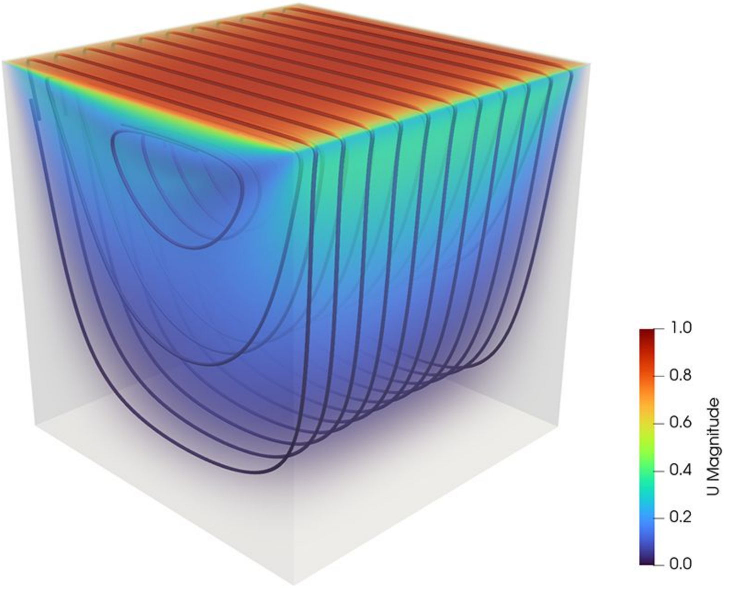
- Aimed at overcoming the current limitations of CFD technology, especially in what concerns the exploitation of massively parallel HPC architectures.
- 13 industrial stakeholders and 5 industrial supporters.
- Development and validation of a range of algorithmic improvements, across the entire CFD process chain.
- Elaboration of an open repository containing all the validation and benchmark cases

exaFOAM

Font: https://exafoam.eu/our-mission/

History of our studies in Deucalion

New record on OpenFOAM scalability - exaFOAM



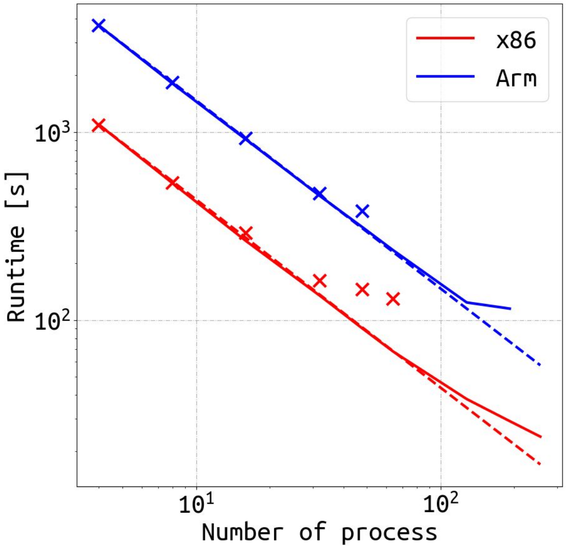
- Won the 19th HPC Innovation Excellence Awards
- 4.096 nodes (524.288 CPU cores)
- HLRS's Hawk supercomputer **(26 PFlops)**
- Exceeded the previous scaling record for a simulation using OpenFOAM by a factor of four

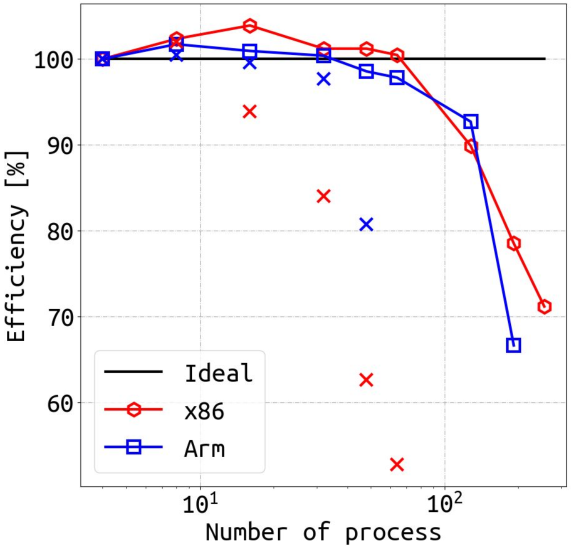
Font: https://www.hlrs.de/news/detail/coherent-file-format-accelerates-time-to-solution-with-openfoam

History of our studies in Deucalion

EVALUATING OPENFOAM PERFORMANCE ON DEUCALION, THE LARGEST PORTUGUESE SUPERCOMPUTER

- GABRIEL M. MAGALHÃES, BERNARDO F. MALACA, ANTÓNIO L. SOUSA, JOÃO M. NÓBREGA
- Published in FOAM@Iberia 2024 - Ferrol, Spain





History of our studies in Deucalion

Bruno Santos (wyldckat)

- One of the main contributors of OpenFOAM community in forums, wikis, …
- Main developer of blueCFD-Core
- More than 10,000 posts in CFD-Online forum, the most famous CFD discussion forum (all related to OpenFOAM)
- More than 50 open source repositories
- Advanced knowledge both in OpenFOAM, compilers and processor architectures.

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- **Challenges of OpenFOAM in ARM architecture**
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

Challenges of OpenFOAM in ARM architecture

- Most of the OpenFOAM distributions are ready to compile in ARM architecture
- The foam-extend distribution needs some adjustments to allow using the Fujitsu compiler
  - Git path developed during exaFOAM by UMinho team
- Deucalion has different OpenFOAM versions and distributions available as modules (load and use)

Which are the main difficulties for OpenFOAM in ARM processors?

Spoiler: are the same as for x86 architecture

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- **Challenges of OpenFOAM in ARM architecture**
  - **The OpenFOAM matrix format**
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

The OpenFOAM matrix format

- The matrix A has a symmetric sparsity pattern with a full diagonal
- Matrix A is decomposed as the sum of a diagonal. a lower and an upper triangular matrix

$$A = L + D + U$$

D is stored as a **dense vector**

L has the **same sparsity pattern** of **U**. In case of symmetric matrix, even the same values.

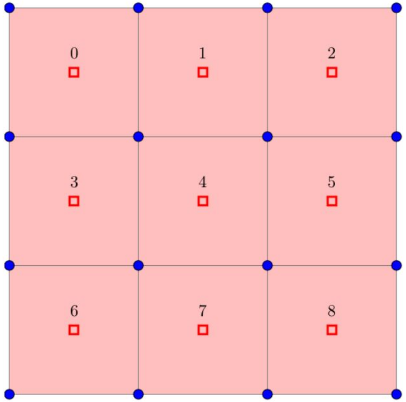
- The idea is to store a single sparsity pattern for U and L. For D the sparsity pattern is implicit.
- We can define a family of LDU storage formats, according to the storage format of U (and implicitly for L)

Font: https://exafoam.eu/wp-content/uploads/2023/07/exaFOAM_Workshop-Impact_of_matrix_data_structure.pdf

The OpenFOAM matrix format

LDU (ordered COO variant) storage formats

- The LDU variant implemented in OpenFOAM is L(ordered COO)+D+U(ordered COO), where COO stands for Coordinate Format.
- Owner and Neighbors are the two int vectors that store the sparsity pattern in COO format.
- In order to use the same vectors for the sparsity pattern of U and L. U values are stored per rows. L values per columns



i \ j	0	1	2	3	4	5	6	7	8
0	4	-1		-1					
1	-1	4	-1		-1				
2		-1	4			-1			
3	-1			4	-1		-1		
4		-1		-1	4	-1		-1	
5			-1		-1	4			-1
6				-1			4	-1	
7					-1		-1	4	-1
8						-1		-1	4

upper = [-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]

diag = [4 4 4 4 4 4 4 4 4]

lower = [-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]

upperAddr = [1 3 2 4 5 4 6 5 7 8 7 8]

lowerAddr = [0 0 1 1 2 3 3 4 4 5 6 7]

Problem: this format doesn't allow for vectorization

Font: https://exafoam.eu/wp-content/uploads/2023/07/exaFOAM_Workshop-Impact_of_matrix_data_structure.pdf

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- **Challenges of OpenFOAM in ARM architecture**
  - The OpenFOAM matrix format
  - **Is it possible to use vectorization in OpenFOAM?**
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

Vectorization and optimization in OpenFOAM

- It is possible to use several flags in the compilation of OpenFOAM but the question is:

Is it safe to use these flags?

- We analyzed different scenarios in Deucalion using different benchmark cases

Let’s check the results…

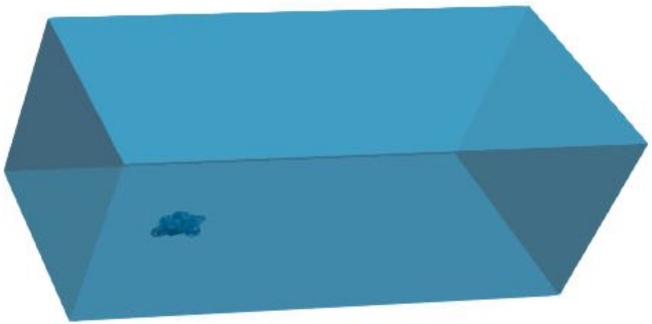
Flags in the compilation: the OpenFOAM versions

- OpenFOAM v23.12: **Arm** partition **GCC**
  - O2
  - O2 + Vectorization
  - OFast
  - OFast + Vectorization

- OpenFOAM v23.12: **Arm** partition **Fujitsu**
  - O2
  - O2 + Vectorization
  - KFast
  - KFast + Vectorization

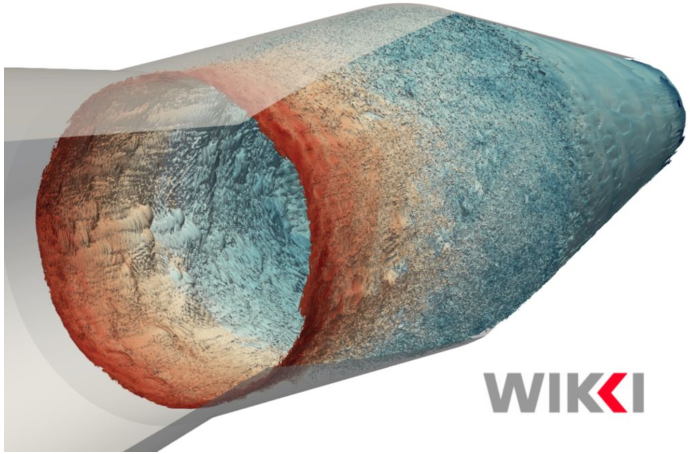
Flags in the compilation: the cases

Small motor bike [Link]



- 8.6M cells
- From begin
- 500 iterations

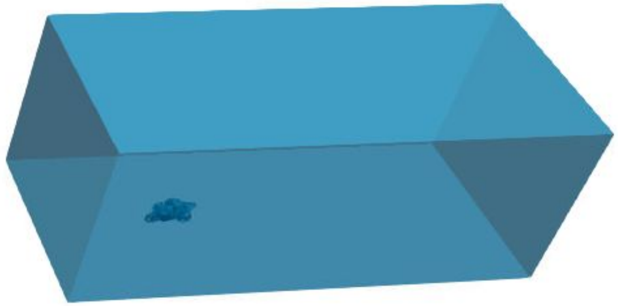
(MB10) ERCOFTAC Conical Diffuser LES [Link]



- 3M cells
- From restart [Link]
- 500 iterations

Flags in the compilation: results

Small motor bike [Link]



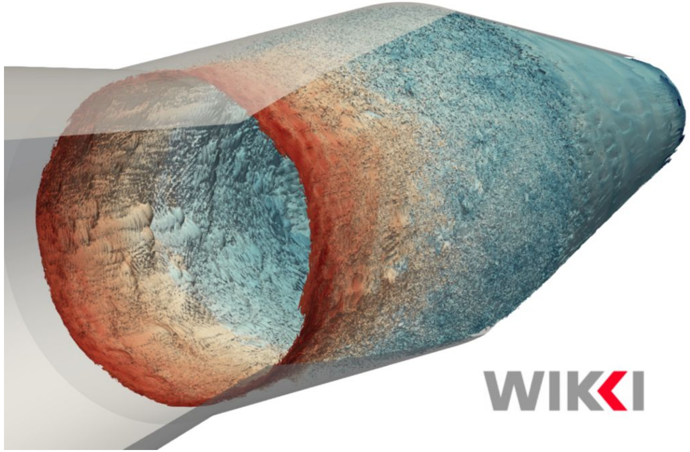
- 8.6M cells
- From begin
- 500 iterations

Flag	Vec	Time [s]
O2	No	4393
OFast	No	4155
O2	Yes	4068
OFast	Yes	3770
O2	No	4263
KFast	No	3533
O2	Yes	4063
KFast	Yes	3133

↘ 29%

Flags in the compilation: the cases

(MB10) ERCOFTAC Conical Diffuser LES [Link]



- 3 million cells
- From restart [Link]
- 500 iterations

Flag	Vec	Time [s]
O2	No	1649
OFast	No	1547
O2	Yes	1804
OFast	Yes	1714
O2	No	1695
KFast	No	1284
O2	Yes	1608
KFast	Yes	1402

↘ 22%

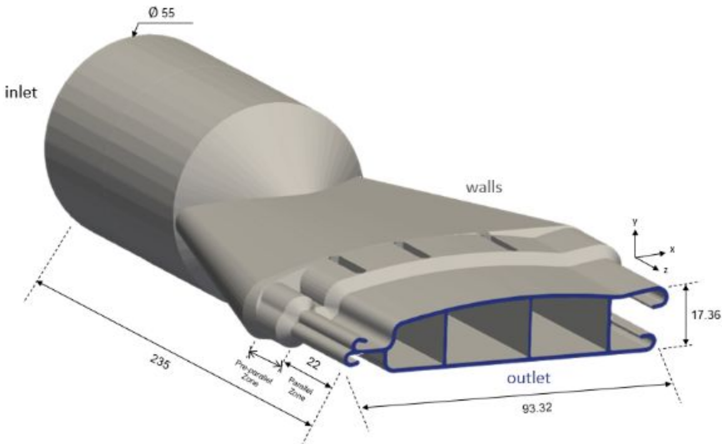
Pay attention on the accuracy!
The result changes

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- **Challenges of OpenFOAM in ARM architecture**
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - **Memory requirement**
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

Memory requirements

- The ARM nodes in Deucalion have 32GB of RAM versus 256GB in the AMD nodes
- Some operations in CFD workflow requires a lot of memory (e.g.. mesh generation and other mesh operations)
- Huge industrial cases can require a lot of memory to run the simulation



exaFOAM benchmark B4

- 40 million cells
- Real industrial case (geometry and fluid)
- Coupled solver

845.6 GB (~27 nodes)

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- **Evaluation of OpenFOAM performance in ARM processors**
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

Evaluation of OpenFOAM performance in ARM processors

A64FX

FUJITSU Supercomputer PRIMEHPC FX700
CPU: A64FX (2.0GHz, 48core/chip, 1 chip/node)
Memory: 32GiB (HBM2: 8GiB x4)
DISK: 1x M.2 SSD 512GB NVMe
Interconnect: 1x HDR100HCA
RAM total speed: 1024 GiB/s

AMD EPYC

Bull Sequana X440 A5
CPU: 2x AMD EPYC 7742 (2.25-3.4GHz,64 Cores)
Memory: 256GB DDR4
DISK: 1x 480GB SSD
Interconnect: 1x HDR100HCA
RAM total speed per socket: 190.7 GiB/s

NOTES

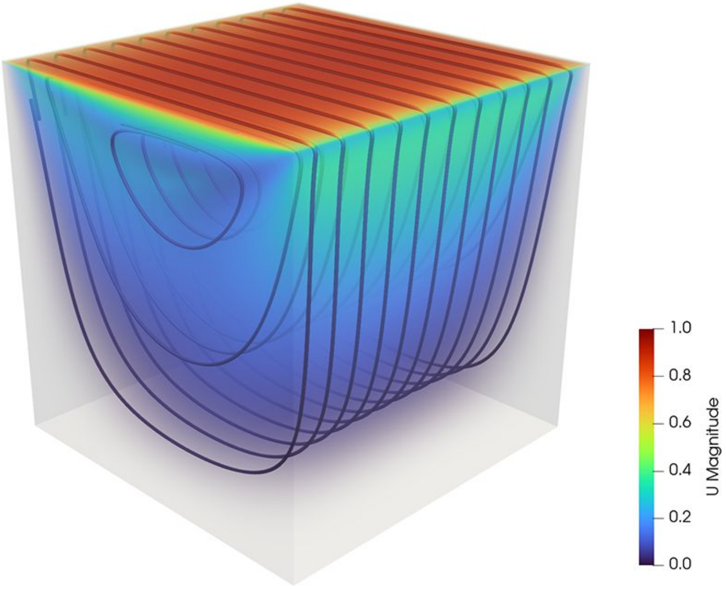
- As the hyperthread is not activated the clock of AMD is 3.4GHz
- The clock ratio is 1.7 (3.4 GHz/2 GHz)
- Each node has 2 AMD EPYC with 8 memory channels each
- AMD L3 cache is 256 MB (shared)
- AMD L3 cache is roughly 2x RAM speed

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- **Evaluation of OpenFOAM performance in ARM processors**
  - **Tests descriptions and motivations**
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

Tests descriptions and motivations

Our main test case



- Micro-benchmark 1 of exaFOAM (MB1)
- OpenFOAM v23.12
- icoFoam solver (just solves pressure)
- 3D case
- Meshes from 262k to 64M cells
- Regular hexahedral mesh
- t = 0.05 s (100 iterations)

Tests descriptions and motivations

Evaluated meshes

262 k	512 k	8 M	42 M	64 M

Evaluated scenarios

- OpenMPI flags (ordered)
- Method for domain decomposition
- Bounded or no by cache memory

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- **Evaluation of OpenFOAM performance in ARM processors**
  - Tests descriptions and motivations
  - **Speed-up and efficiency results**
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

Memory binding policy results

Compared to "spread", this policy should maximise the amount of L3 cache available to each thread.

close: The idea here is to keep the threads as close as possible, i.e. minimising core-core

threads	16	core							
policy	close	0	1	2	3	4	5	6	7
NUMA node	0	0	1	2	3	4	5	6	7
	1	8	9	10	11	12	13	14	15
	2								
	3								
	4								
	5								
	6								
	7								

`--map-by core`

spread: With the idea to make use of all available memory bandwidth

threads	16	core							
policy	spread	0	1	2	3	4	5	6	7
NUMA node	0	0	1						
	1	2	3						
	2	4	5						
	3	6	7						
	4	8	9						
	5	10	11						
	6	12	13						
	7	14	15						

`--map-by numa`

scatter: The idea behind this is that each NUMA node is divided again into 2 groups of 4 cores, where each group has its own L3 cache.

threads	16	core							
policy	scatter	0	1	2	3	4	5	6	7
NUMA node	0	0				1			
	1	2				3			
	2	4				5			
	3	6				7			
	4	8				9			
	5	10				11			
	6	12				13			
	7	14				15			

`--map-by l3cache`

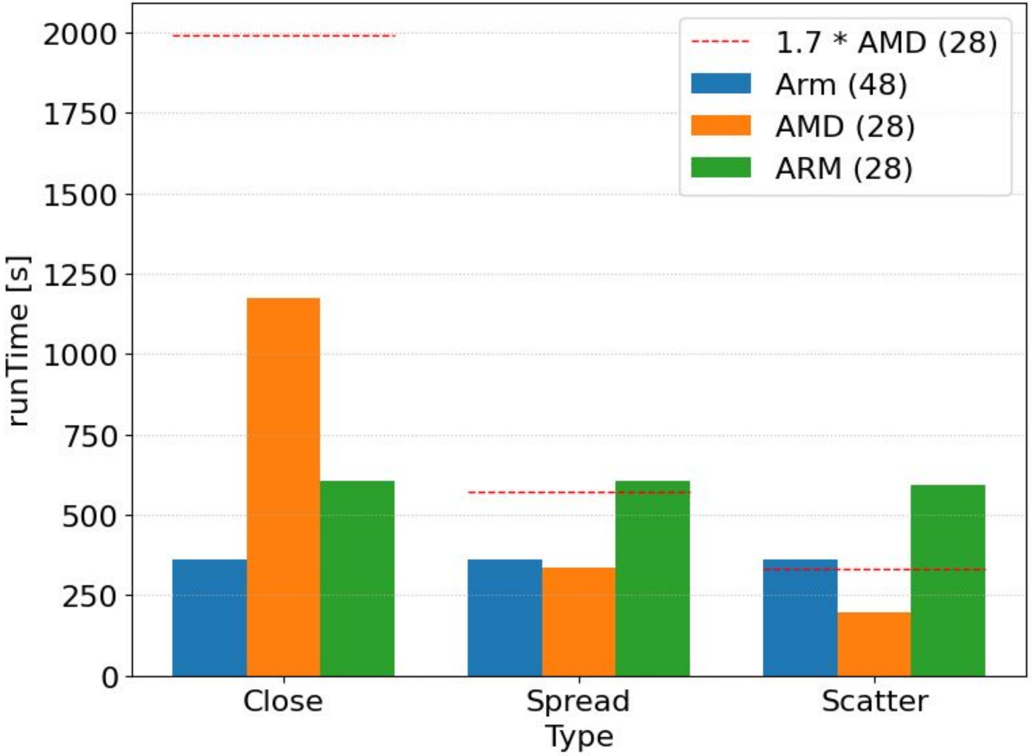
Font: https://stackoverflow.com/questions/58133746/open-mpi-4-0-core-and-memory-binding

Memory binding policy results - runTime [s]

Type	Arm (48)	AMD (28)	ARM (28)
Close	362	1172	605
Spread	361	337	604
Scatter	362	196	591

Cavity 8M cells

1 node



Important points to perform a speed-up test

- The speedup σ for a simulation with n cores beginning with i cores is computed as follows:

$$\sigma_n = \frac{T_i}{T_n}$$

T is the run time

- The parallel efficiency is defined as:

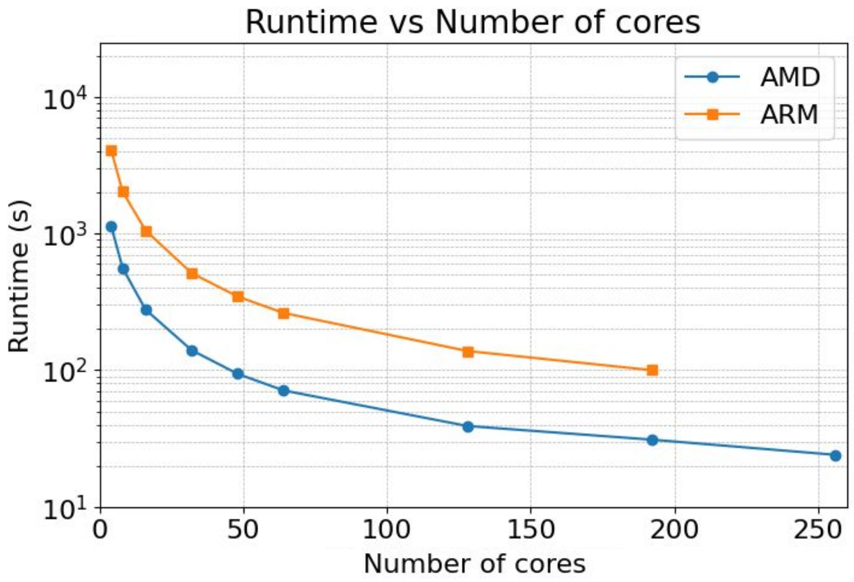
$$\epsilon_n = 100\left(\frac{i}{n}\sigma_n\right)$$

- The speedup must be close to n and the efficiency must be close to 100%

Speed-up and efficiency results

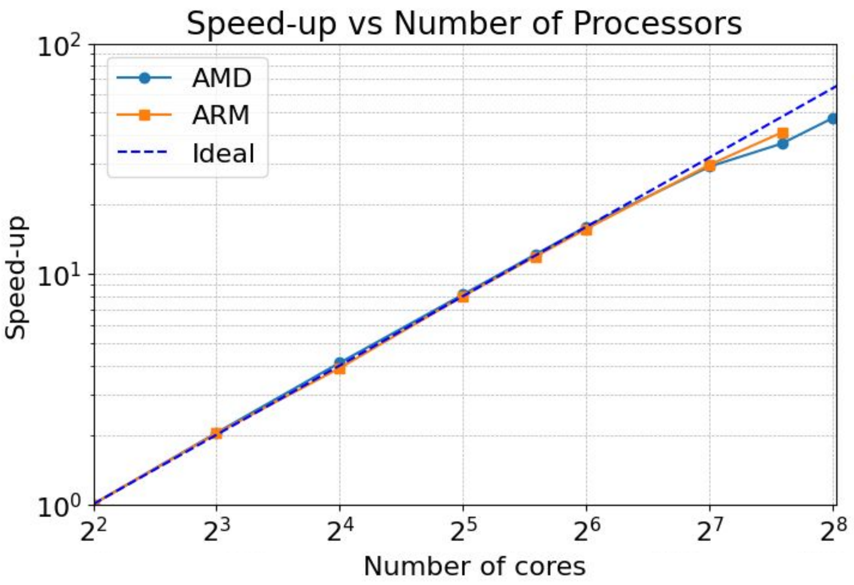
8M cells

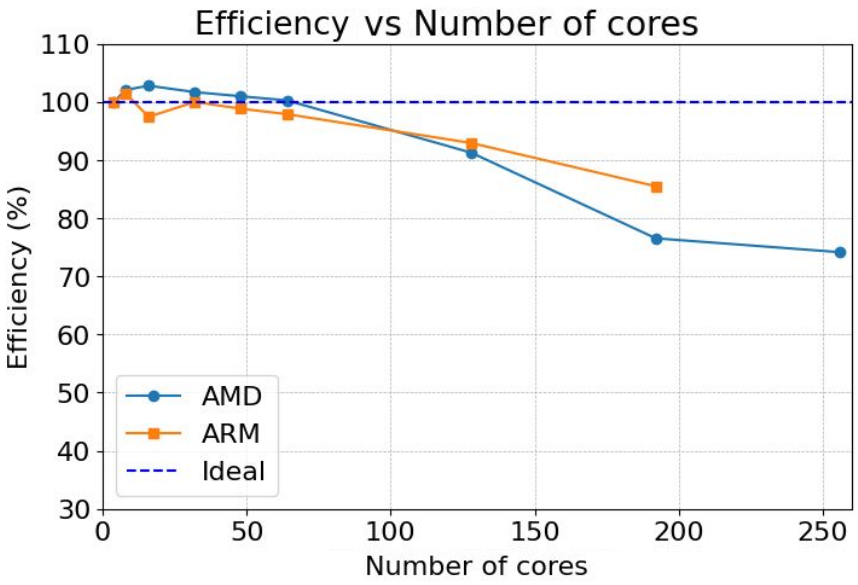
		Runtime [s]		Speed-up		Efficiency [%]	
nProcs	Cells per proc	AMD	ARM	AMD	ARM	AMD	ARM
4	2,000,000	1,139	4,104	1.00	1.00	100.0	100.0
8	1,000,000	558	2,022	2.04	2.03	102.1	101.5
16	500,000	277	1,053	4.11	3.90	102.8	97.4
32	250,000	140	513	8.14	8.00	101.7	100.0
48	166,667	94	346	12.12	11.86	101.0	98.8
64	125,000	71	262	16.04	15.66	100.3	97.9
128	62,500	39	138	29.21	29.74	91.3	92.9
192	41,667	31	100	36.74	41.04	76.5	85.5
256	31,250	24		47.46		74.2	



Speed-up and efficiency results

8M cells

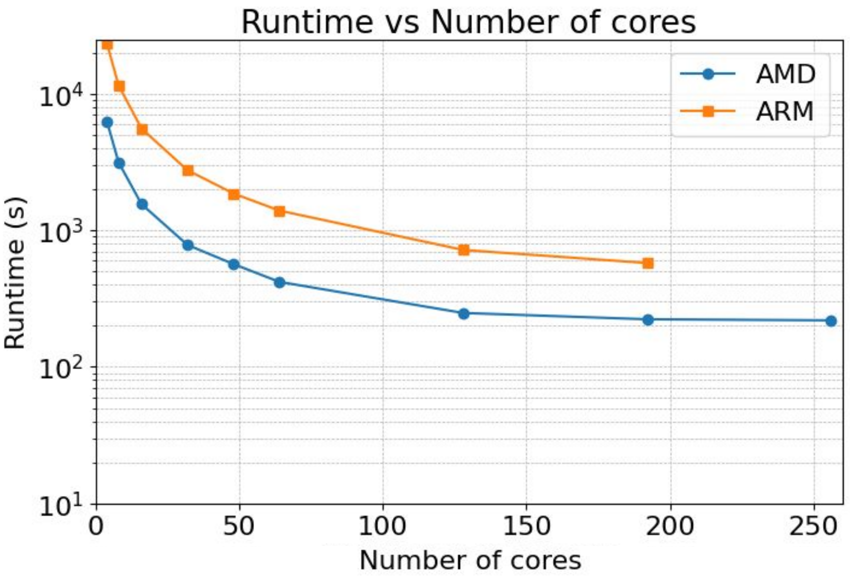




Speed-up and efficiency results

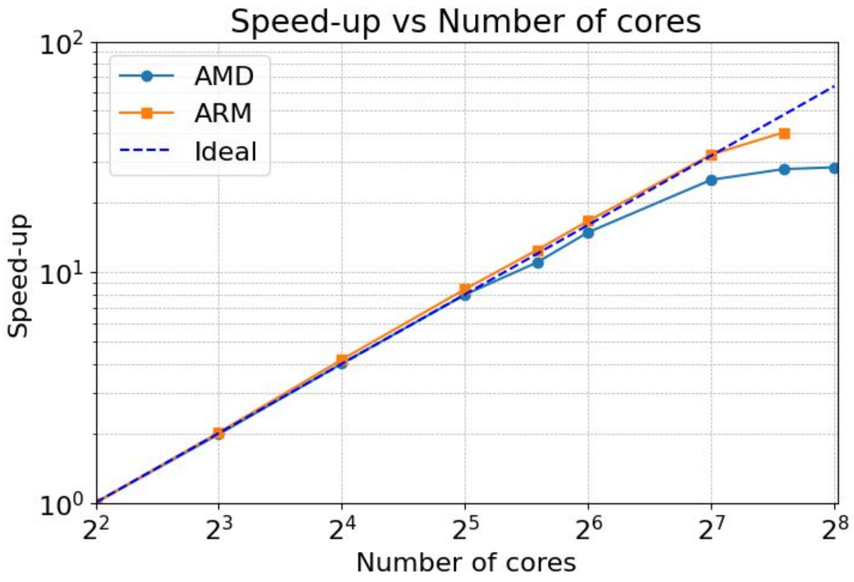
42M cells

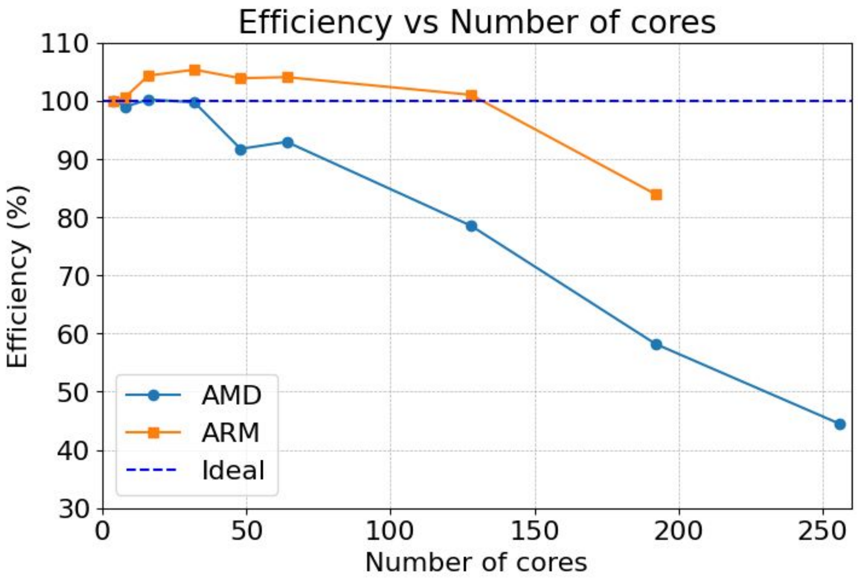
		Runtime [s]		Speed-up		Efficiency [%]	
nProcs	Cells per proc	AMD	ARM	AMD	ARM	AMD	ARM
4	2,000,000	6231	23210	1.00	1.00	100	100.0
8	1,000,000	3148	11532	1.98	2.01	99.0	100.6
16	500,000	1554	5562	4.01	4.17	100.2	104.3
32	250,000	781	2754	7.98	8.43	99.7	105.3
48	166,667	566	1862	11.01	12.47	91.7	103.9
64	125,000	419	1394	14.87	16.65	92.9	104.1
128	62,500	248	718	25.13	32.33	78.5	101.0
192	41,667	223	576	27.94	40.30	58.2	83.9
256	31,250	219		28.45		44.5	



Speed-up and efficiency results

42M cells

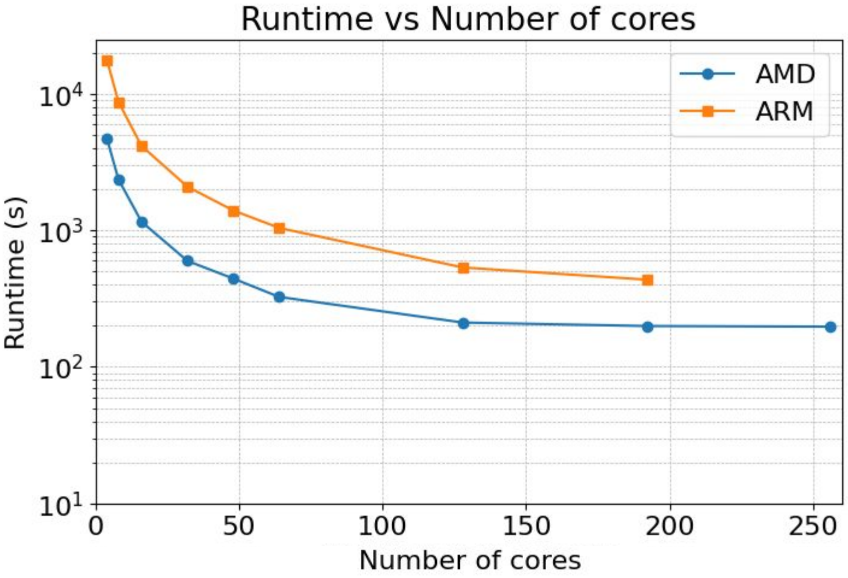




Speed-up and efficiency results

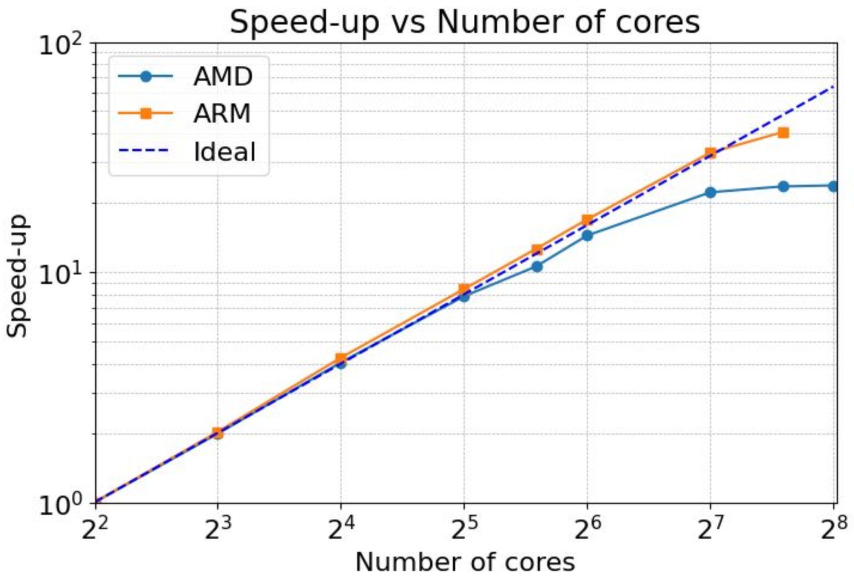
64M cells

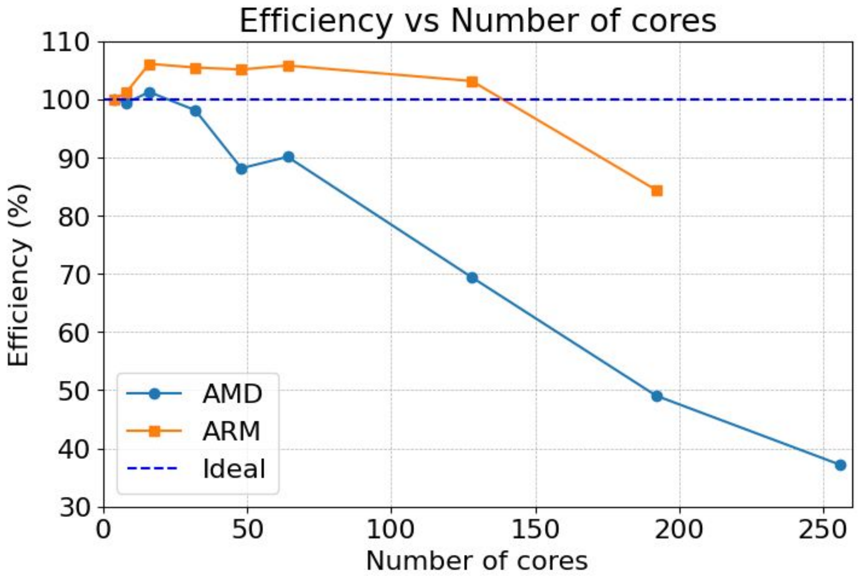
		Runtime [s]		Speed-up		Efficiency [%]	
nProcs	Cells per proc	AMD	ARM	AMD	ARM	AMD	ARM
4	2,000,000	4686	17624	1.00	1.00	100	100
8	1,000,000	2358	8710	1.99	2.02	99.4	101.2
16	500,000	1157	4153	4.05	4.24	101.3	106.1
32	250,000	597	2089	7.85	8.44	98.1	105.5
48	166,667	443	1397	10.58	12.62	88.1	105.1
64	12,5000	325	1041	14.42	16.93	90.1	105.8
128	62,500	211	534	22.21	33.00	69.4	103.1
192	41,667	199	435	23.55	40.51	49.1	84.4
256	31,250	197		23.79		37.2	



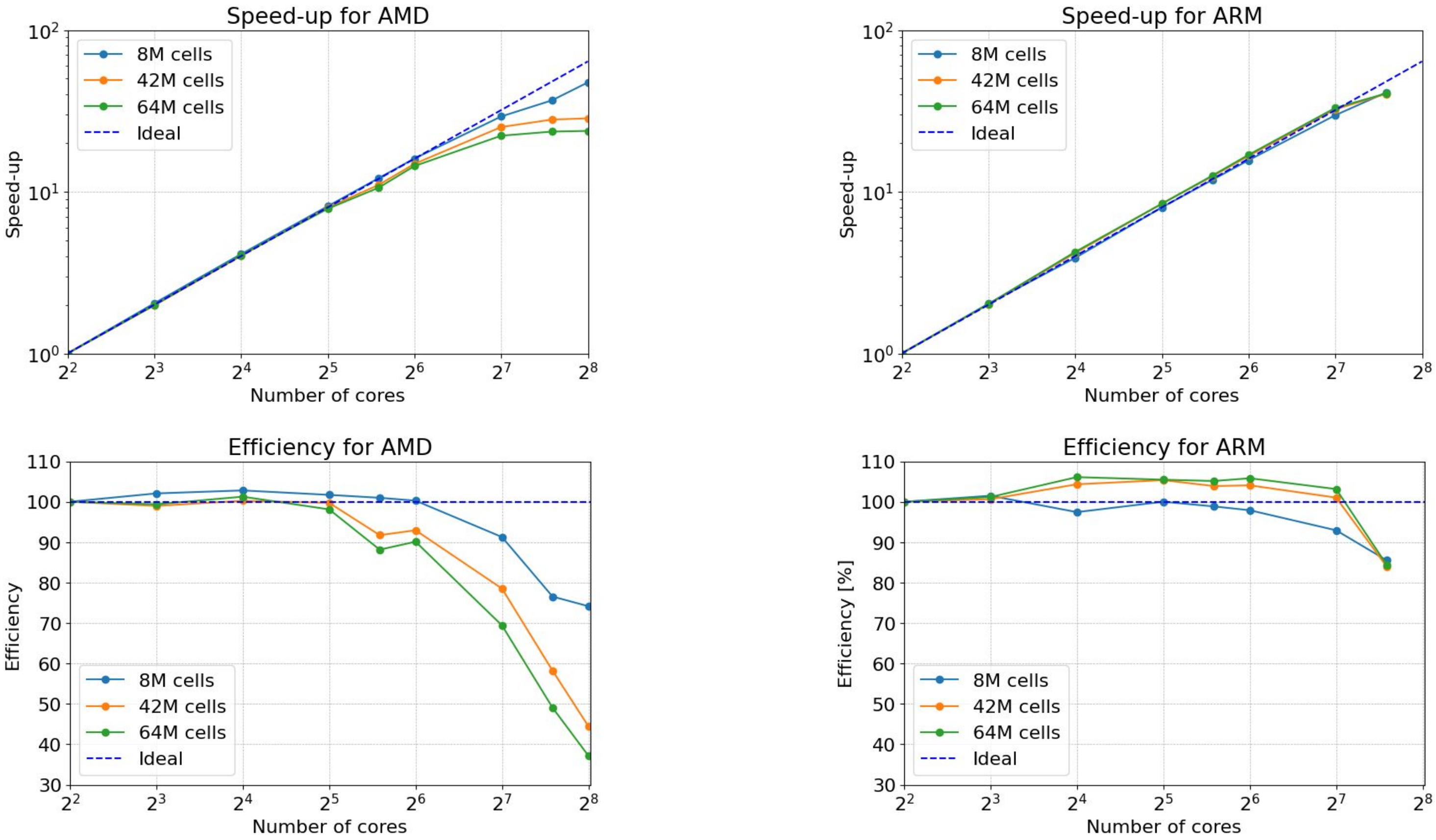
Speed-up and efficiency results

64M cells





Speed-up and efficiency results

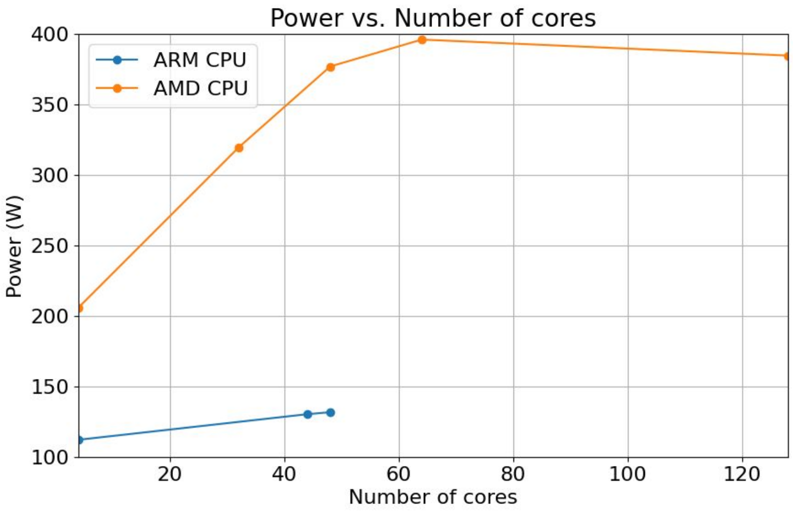


Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- **Evaluation of OpenFOAM performance in ARM processors**
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - **Energy efficiency**
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

Energy efficiency

Processor	# cores	# cells per proc	runTime [s]	Elapsed time [s]	Energy CPU [kWh]	Power CPU [W]
ARM	4	2000000	4065	4175	0.13017	112.24
ARM	44	181818	386	484	0.01753	130.39
ARM	**48**	**166667**	**361**	**464**	**0.01699**	**131.82**
AMD	4	2000000	1142	1182	0.06763	205.98
AMD	32	250000	164	218	0.01935	319.54
AMD	**48**	**166667**	**133**	**183**	**0.01916**	**376.92**
AMD	64	125000	128	176	0.01936	396.00
AMD	128	62500	122	187	0.01998	384.64



Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- **Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures**
- Take home messages

How to improve OpenFOAM performance in ARM architecture?

1 - Build matrices in a vectorizable format

- By modifying how OpenFOAM constructs the equation matrices (LDU-orderedCOO) directly into a vectorizable algorithm (e.g. CSR format).
- It would allow direct plugging into PETSc, which has several optimized algorithms for this format.

NOTE: Some implementations exist in this way, but all of them are based on conversions from the OpenFOAM format to CSR, which penalizes the efficiency (e.g. FOAM2CSR)

How to improve OpenFOAM performance in ARM architecture?

2 - Use a vectorizable format

- Once the CSR matrix is available, the first option would be to simply use a compatible implementation present in PETSc to ARM's SVE.
- Not sure which exact configuration it is, but it should be similar to Intel/AMD's AVX512 (advanced vector extensions 512bits - i.e. 64 double floating point values).

NOTE: Some implementations exist in this way, but all of them are based on conversions from the OpenFOAM format to CSR, which penalizes the efficiency (e.g. FOAM2CSR)

Font: https://www.hlrs.de/news/detail/coherent-file-format-accelerates-time-to-solution-with-openfoam

How to improve OpenFOAM performance in ARM architecture?

3 - Use an OpenMP implementation

- As a second (cumulative) option, leverage OpenMP implementation in PETSc to feed matrix chunks from RAM onto L2 cache and crunch it with all cores. E.g.: 4 processes, 12 threads per process; each process would populate the L2 cache and then all 12 cores would crunch the buffered payload.
- Use --*with-openmp* flag to allow PETSc to be used within an OpenMP application. If your application calls PETSc from within OpenMP threads then also use --*with-threadsafety* flag.
- Use --*with-openmp-kernels* flag to have some PETSc numerical routines use OpenMP to speed up their computations. This requires --*with-openmp* flag.
- Note that using OpenMP within MPI code must be done carefully to prevent too many OpenMP threads that might overload the calculation cores.

Font: https://petsc.org/main/install/install/#openmp

How to improve OpenFOAM performance in ARM architecture?

Important notes

- The possibilities for future works were thought aiming to extract the maximum performance of ARM architectures for OpenFOAM runs
- Use the CSR structure directly on the matrix building could open additional possibilities in other fronts
- When implementing the suggested approaches, the x86 architectures could also reach a better performance
- Some of the developments in the future works could contribute for better results of OpenFOAM in GPUs
- The HMM project at ROCm apparently is implementing something in this way (apparently they have something related to OpenMP also)

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- **Take home messages**

Take home messages

The ARM partition of Deucalion is fully prepared to run OpenFOAM simulations

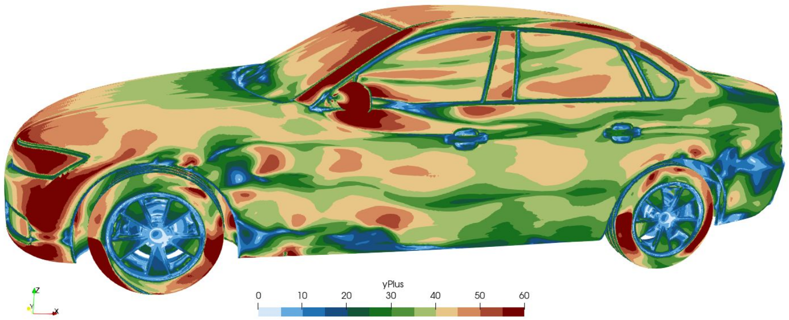
Compare ARM and x86 architectures when using OpenFOAM is not trivial

The energy efficiency of ARM processors is better than AMD (green500)

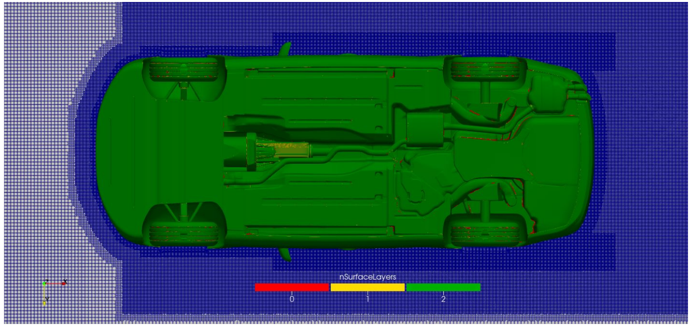
Efficiency of ARM architecture in the speed-up tests is better than AMD

Industrial case

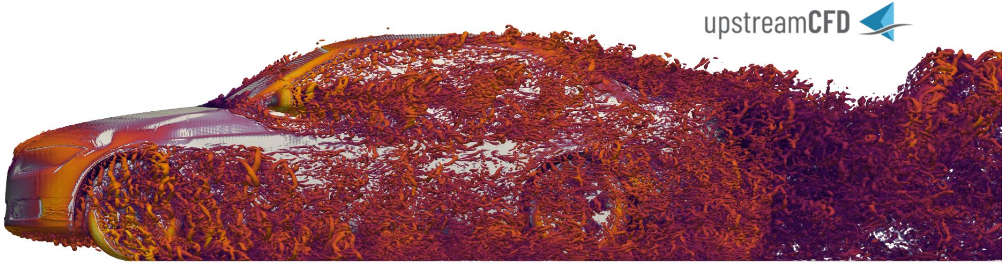
Open-closed cooling DrivAer variant with rotating mesh



- Full car model with closed coolings and a complex underbody
- Turbulence model: k-ω SST DDES
- 238M cells
- 384 cores (24 nodes)



*We are doing our post processing



Fonte: [Case source (exaFOAM repository)]

Agenda

- Introduction
  - What is CFD?
  - What is OpenFOAM?
  - History of our studies in Deucalion
- Challenges of OpenFOAM in ARM architecture
  - The OpenFOAM matrix format
  - Is it possible to use vectorization in OpenFOAM?
  - Memory requirement
- Evaluation of OpenFOAM performance in ARM processors
  - Tests descriptions and motivations
  - Speed-up and efficiency results
  - Energy efficiency
- Future work - Ideas of how to improve the OpenFOAM performance in ARM architectures
- Take home messages

EPICURE
Unlocking European-level HPC Support

EPICURE HPC in ARM Architecture Hackathon

Assessing the Performance and Scalability of ARM Processors for Industrial Applications

Gabriel Marcos Magalhães

INESC TEC / PIEP

UNIVERSIDADE DO MINHO INESCTEC piep iPC INSTITUTE FOR POLYMERS AND COMPOSITES