CEEC

Centre of Excellence in Exascale CFD

h-type adaptive mesh refinement for spectral element solvers. Application to Nek5000 and Neko

Adam Peplinski (KTH)

CEEC

Centre of Excellence in Exascale CFD

Outline

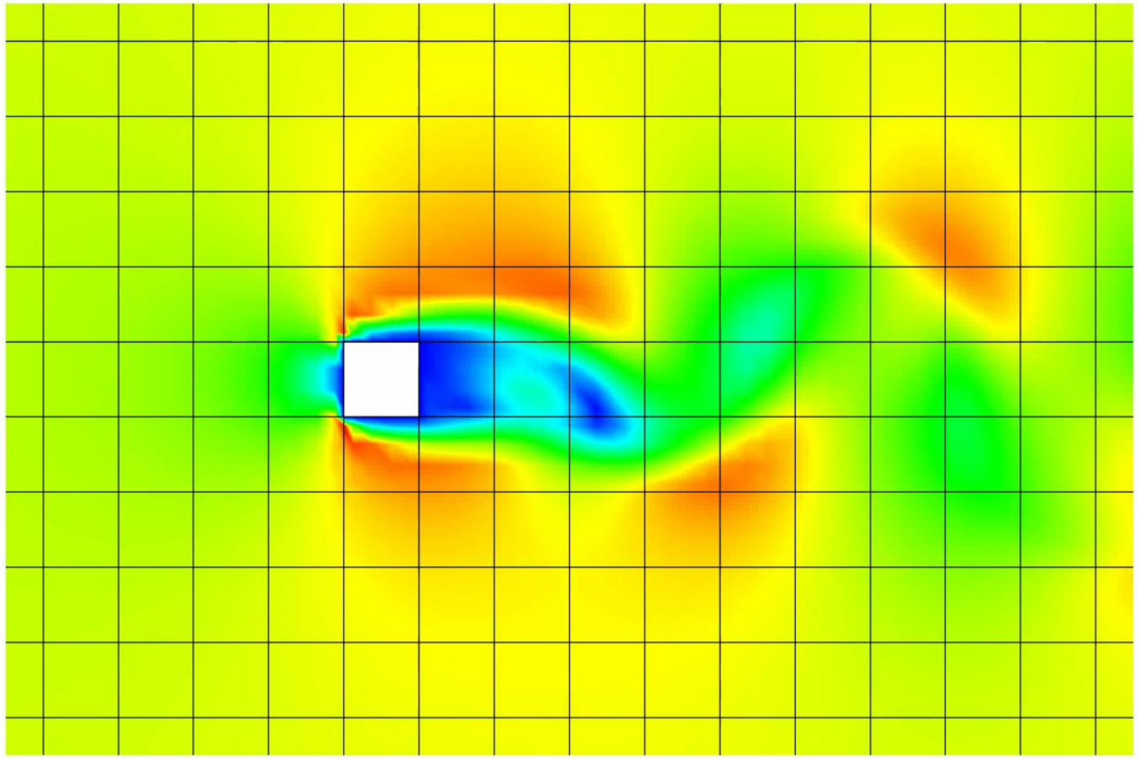
- Introduction
- Spectral element method
- Adaptive mesh refinement
- Nek5000 implementation
- Current work on Neko

Introduction

- Presented material is a result of work supported by multiple EU projects : CRESTA, ExaFLOW and EXCELLERAT
- There are two current EU project focusing on Neko and adaptive mesh refinement: CEEC and EXCELLERAT P2
  - Relatively big project, so collaboration needed

Centre of Excellence in Exascale CFD

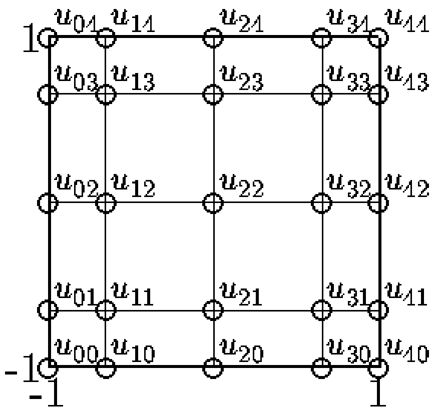
Introduction



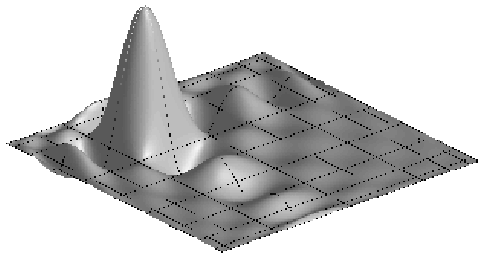
- Exascale simulations:
  - Pose problem for meshing
  - Unknown physics
  - Reliability of simulation data
- Specific goals
  - Resolve sensitive regions
  - Quality assessment/assurance of data
  - Reduced computational cost
- Required tools
  - Grid management library
  - Load balancing/grid repartitioning
  - Refinement indicator/error estimators
  - Solver for non-conformal meshes

Spectral element method

- Variational method, similar to FEM, using GL quadrature
- Domain partitioned into $\boldsymbol{E}$ high-order **hexahedral elements**
- Trial and test functions represented as $\boldsymbol{N}$th-order **tensor-product polynomials** within each element.
- Converges **exponentially fast** with $\boldsymbol{N}$ for smooth solutions.



N=4



N=10

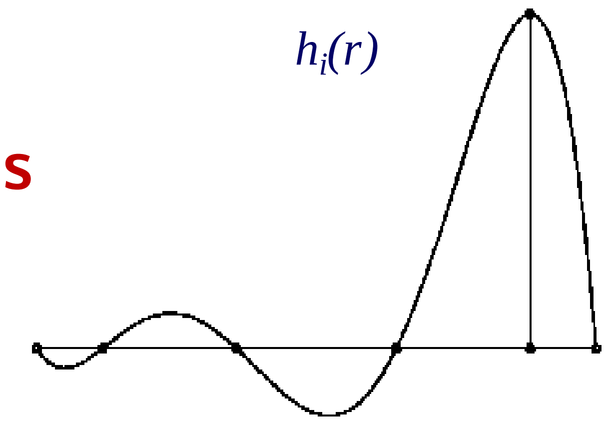
Deville et al 2002

$$u(x,y) = \sum_{i=0}^{N} \sum_{j=0}^{N} u_{ij} h_i(x) h_j(y), \;\; h_i(\xi_p) = \delta_{ip}, \;\; h_i \in \mathbb{P}_N$$

Spectral element method

Local tensor-product form (2D),

$$u(r,s)=\sum_{i=0}^{N}\sum_{j=0}^{N}u_{ij}h_i(r)h_j(s), \quad h_i(\xi_p)=\delta_{ip},\ h_i\in\mathbb{P}_N$$



Deville et al 2002

allows derivatives to be evaluated as **matrix-matrix products**

$$\left.\frac{\partial u}{\partial r}\right|_{\xi_i,\xi_j}=\sum_{p=0}^{N}u_{pj}\left.\frac{dh_p}{dr}\right|_{\xi_i}=\sum_p \underbrace{\hat{D}_{ip}u_{pj}}_{mxm}=:D_r\underline{u}$$

Matrix-free operator evaluation

diagonal mass matrix

$$D_r=(I\otimes I\otimes\hat{D}) \qquad G_{rs}=J\circ B\circ\left(\frac{\partial r}{\partial x}\frac{\partial s}{\partial x}+\frac{\partial r}{\partial y}\frac{\partial s}{\partial y}+\frac{\partial r}{\partial z}\frac{\partial s}{\partial z}\right)$$

Spectral element method

- Spectral element coefficients stored on element basis ($\underline{u}_L$ not $\underline{u}$)

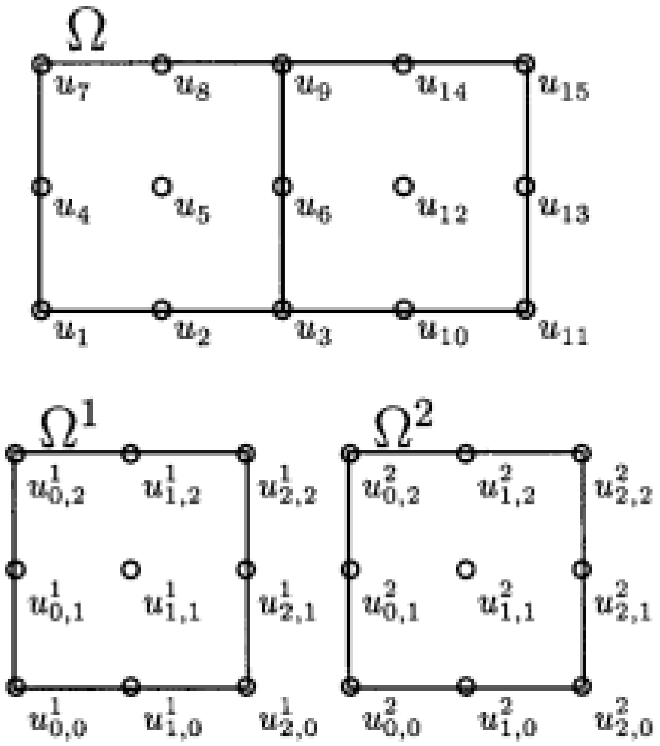
$$\underline{w} = A\underline{u} = Q^T A_L Q\underline{u}, \qquad \underline{w}_L := Q\underline{w}, \qquad \underline{u}_L := Q\underline{u}$$

$$\underline{w}_L = QQ^T A_L \underline{u}_L$$

local work (matrix-matrix products)

nearest-neighbor (gather-scatter) exchange

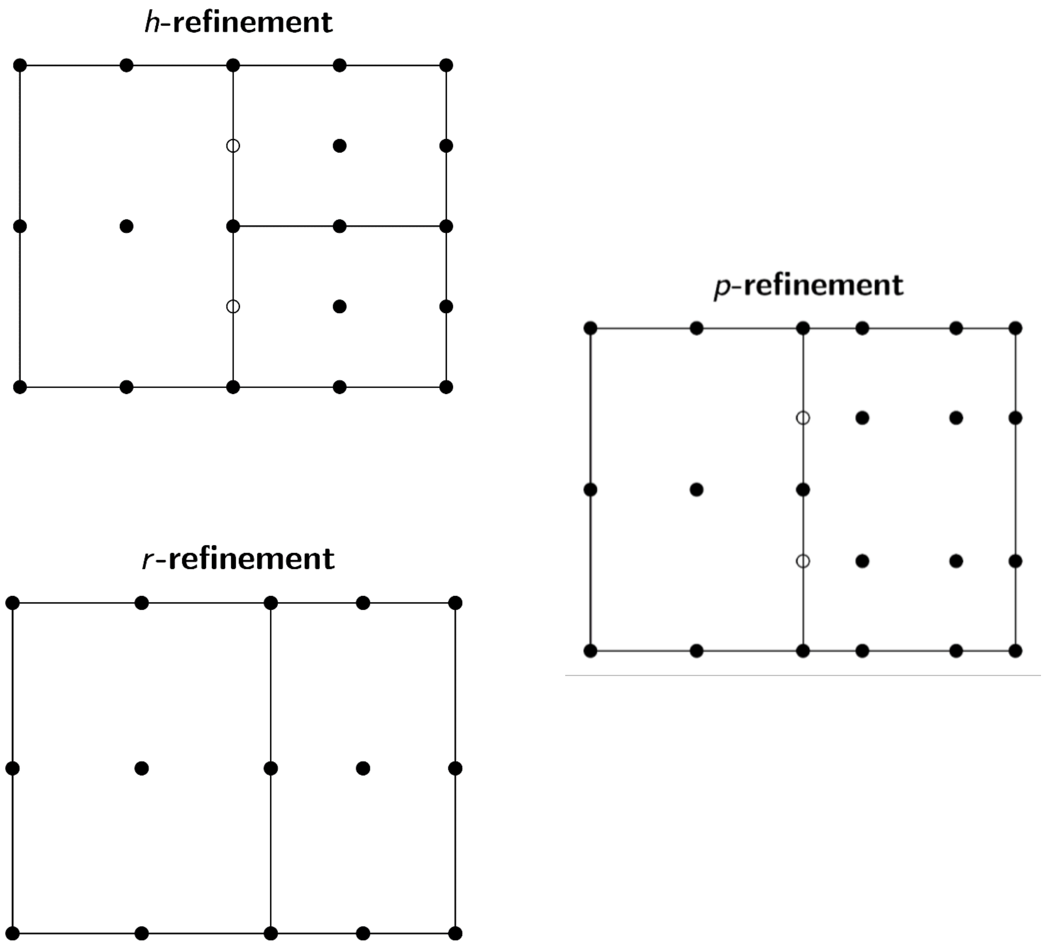
$$A_L := \begin{bmatrix} A^1 & & & \\ & A^2 & & \\ & & \ddots & \\ & & & A^E \end{bmatrix}$$



- Decouples complex physics (A_L) from communication (QQ^T)

Fischer at al 2002

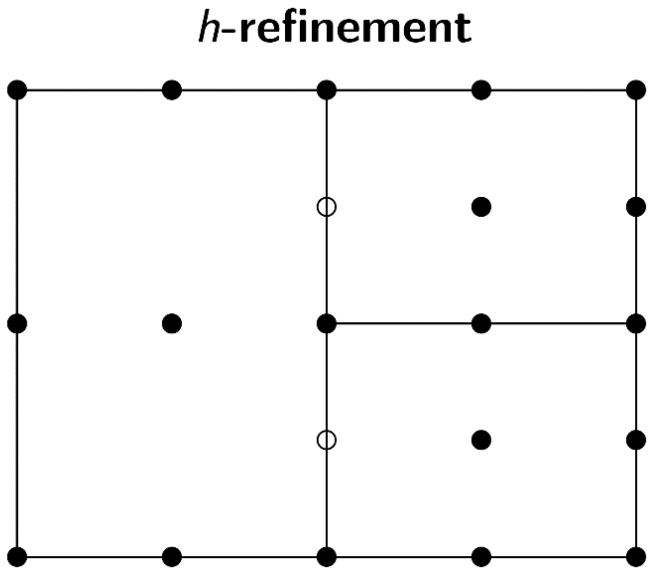
Adaptive mesh refinement



AMR approach:

- **h-refinement** - number of degrees of freedom modified by splitting/merging elements
- **p-refinement** - number of degrees of freedom modified by local variation of the polynomial order
- **r-refinement** – constant number of degrees of freedom; resolution modified by redistribution of grid points

Adaptive mesh refinement

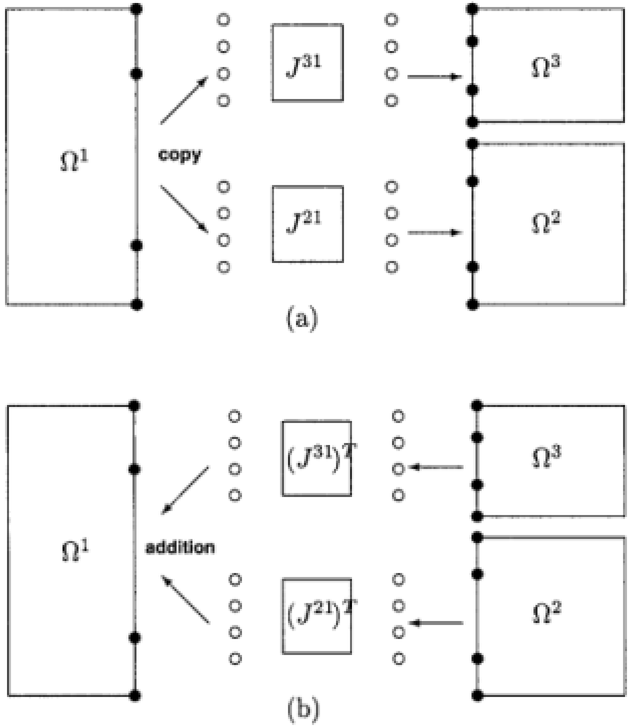


AMR approach:

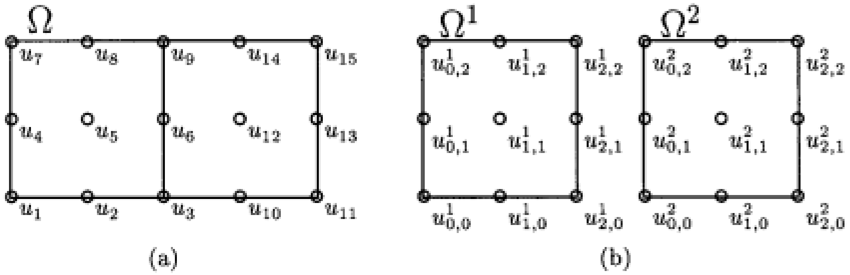
- **h-refinement** - number of degrees of freedom modified by splitting/merging elements

P. F. Fisher, G. W. Kruse, F. Loth, **Spectral element methods for transitional flows in complex geometries**, J. of Sci. Comput. 17 1, 81-98, 2002

Adaptive mesh refinement

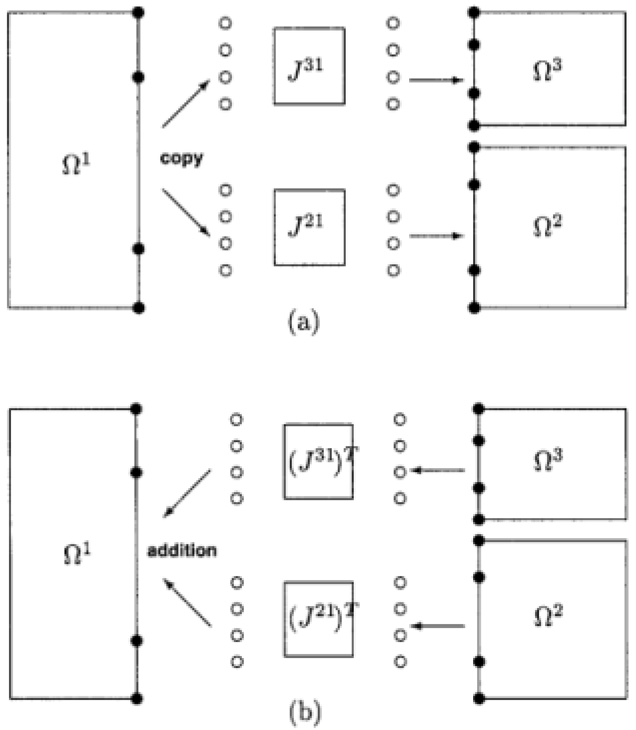


- **Geometrically nonconforming**
- **Parent-child communication** between elements requires additional interpolation operator J
- **Hanging nodes**: information required

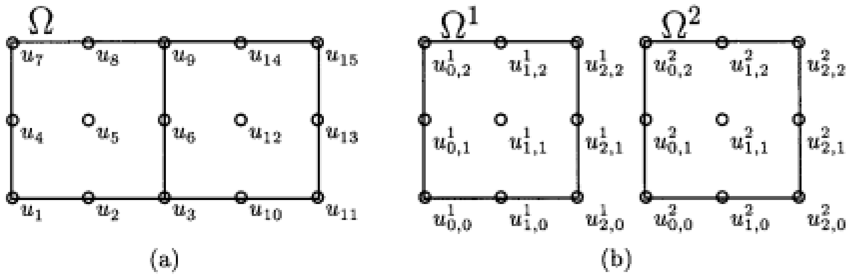


Fischer at al 2002

Adaptive mesh refinement



- **Functionally conforming**
- J – spectral interpolation
- Conforming vs. nonconforming matching condition on interface
- **Conforming-space/ nonconforming-mesh approach**



Fischer at al 2002

Adaptive mesh refinement

- Gather-scatter operator
  - Helmholtz equation
    $$J_L Q Q^T J_L^T$$
  - Pressure preconditioner
    $$J_L Q Q^T J_L^{-1}$$
- Mass matrix diagonalization
  $$\tilde{b} \coloneqq Q^T J_L^T B_L e_L$$
- Coarse grid operator
  $$\hat{A}_{ij} \coloneqq a_i^T A_L a_j$$

Interpolation operator

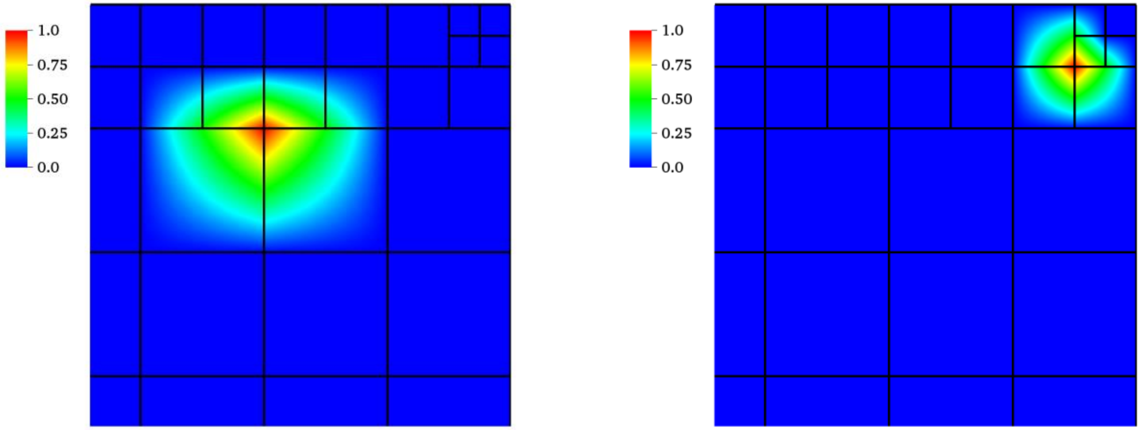
$$J_{ij} = h_j(\xi_i^{cp})$$

Inverse interpolation operator

$$(J^{-1})_{ij} = \\ = \begin{cases} h_j(\xi_i^{pc}) & , if\ \xi_j^p \in \partial\Omega^p \cap \partial\Omega^c \\ 0 & , \quad otherwise \end{cases}$$

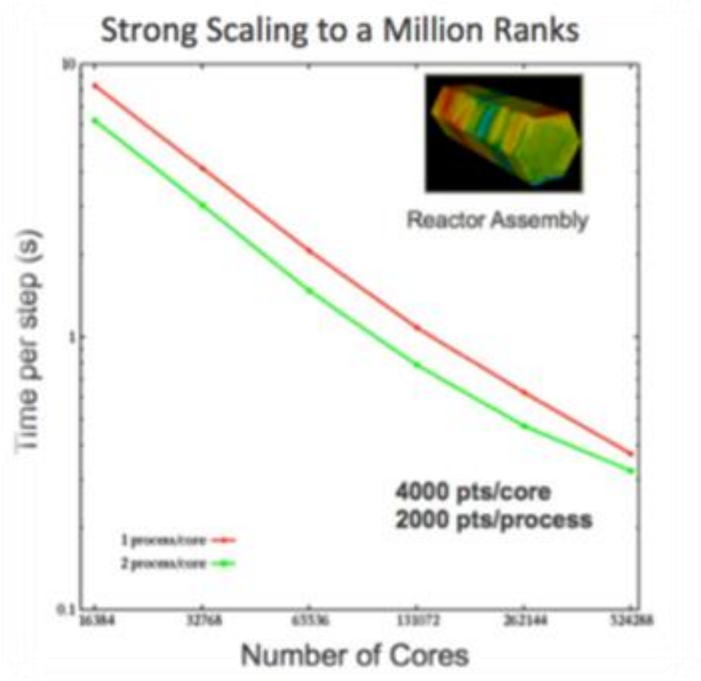
Adaptive mesh refinement

- Gather-scatter operator
  - Helmholtz equation
    $$J_L Q Q^T J_L^T$$
  - Pressure preconditioner
    $$J_L Q Q^T J_L^{-1}$$
- Mass matrix diagonalization
  $$\tilde{b} := Q^T J_L^T B_L e_L$$
- Coarse grid operator
  $$\hat{A}_{ij} := a_i^T A_L a_j$$



Examples of the base functions for coarse grid solver interpolation

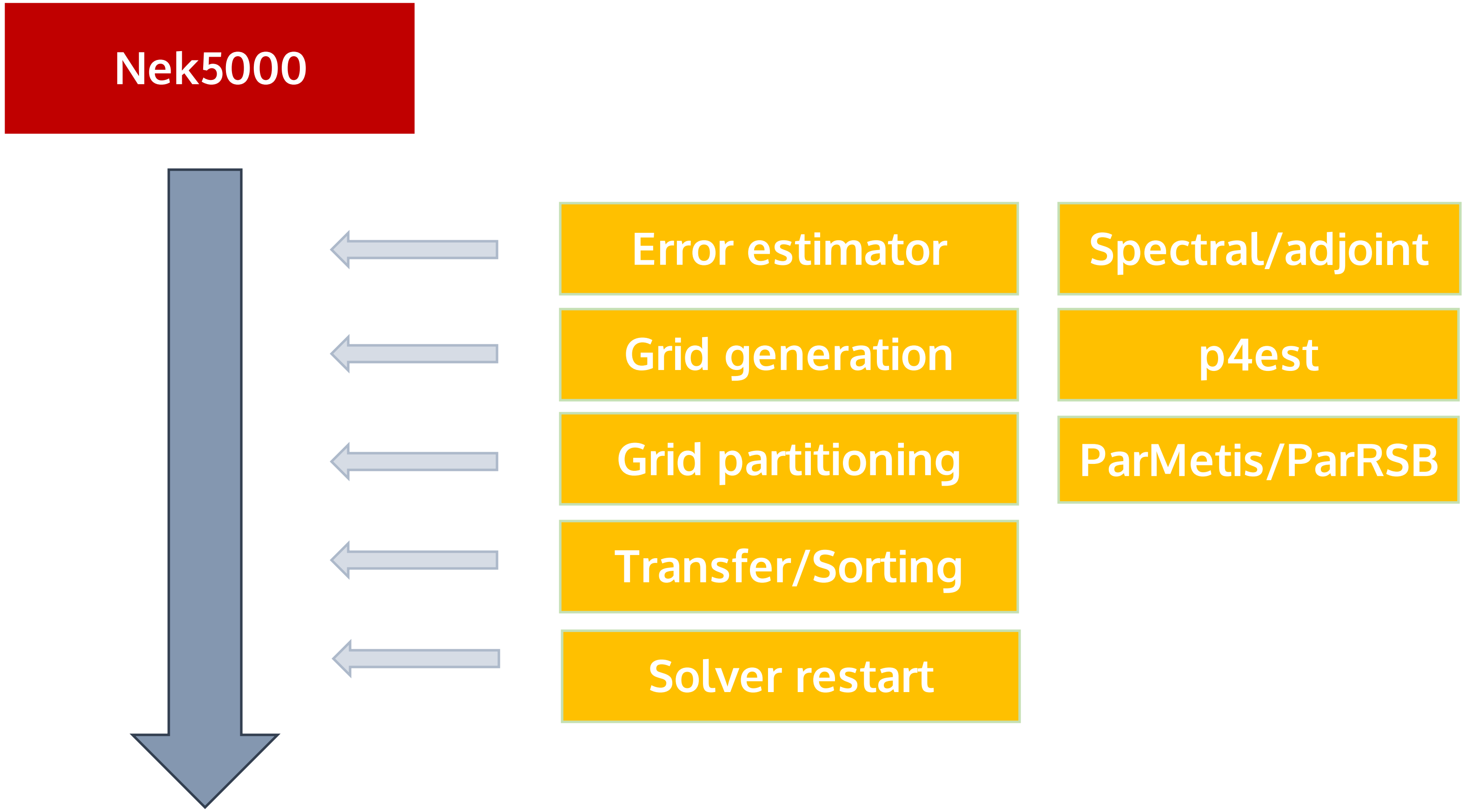
Nek5000 implementation



- bla>500k cores on ALCF BG/Q Mira
- Full scale parallel efficiency 0.6 with two processes per core

- General CFD solver:
  - incompressible and low Mach approximation
  - conjugate heat transfer
  - …
- Long development history
  - First commercially-available code for distributed memory computers
  - Gordon Bell Prize 1999
- Based on Spectral Element Method
  - High order discretisation
  - Low numerical diffusivity
  - Good scaling properties

Nek5000 implementation



Nek5000 implementation

Spectral error estimator

- Sum of the truncation error and the quadrature error
- Local calculation performed within single spectral element
- Based on variable expansion in Legendre base

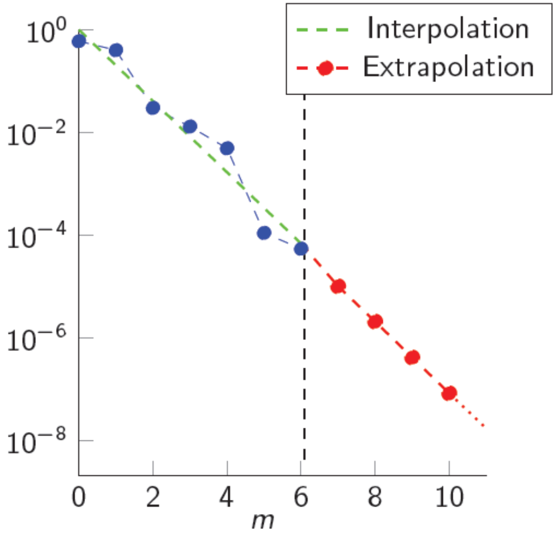
$$u_N^e(r) = \sum_{n=0}^{N} a_n^e L_n(r),$$

- Legendre coefficient approximation

$$a(n) = C\, e^{-\sigma n}$$

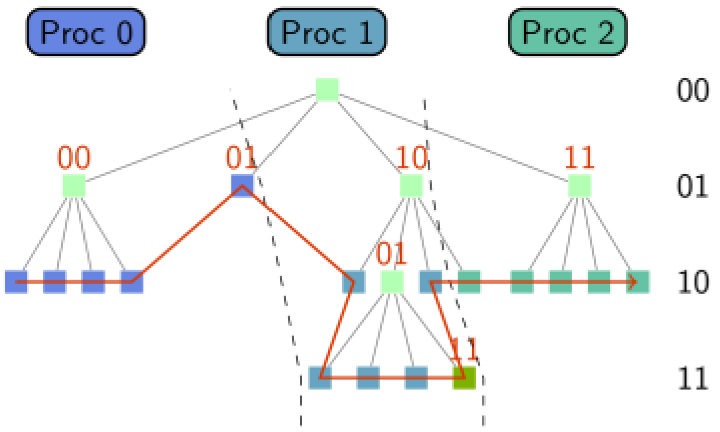
- Error estimate

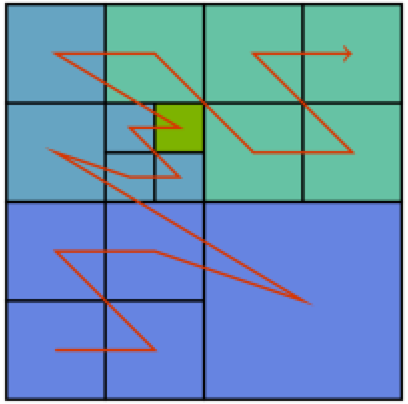
$$\epsilon_{est} = \left(\frac{2a_N^2}{2N+1} + \int_{N+1}^{\infty} \frac{2[a(n)]^2}{2n+1} dn \right)^{1/2}$$



Legendre coefficient

Nek5000 implementation



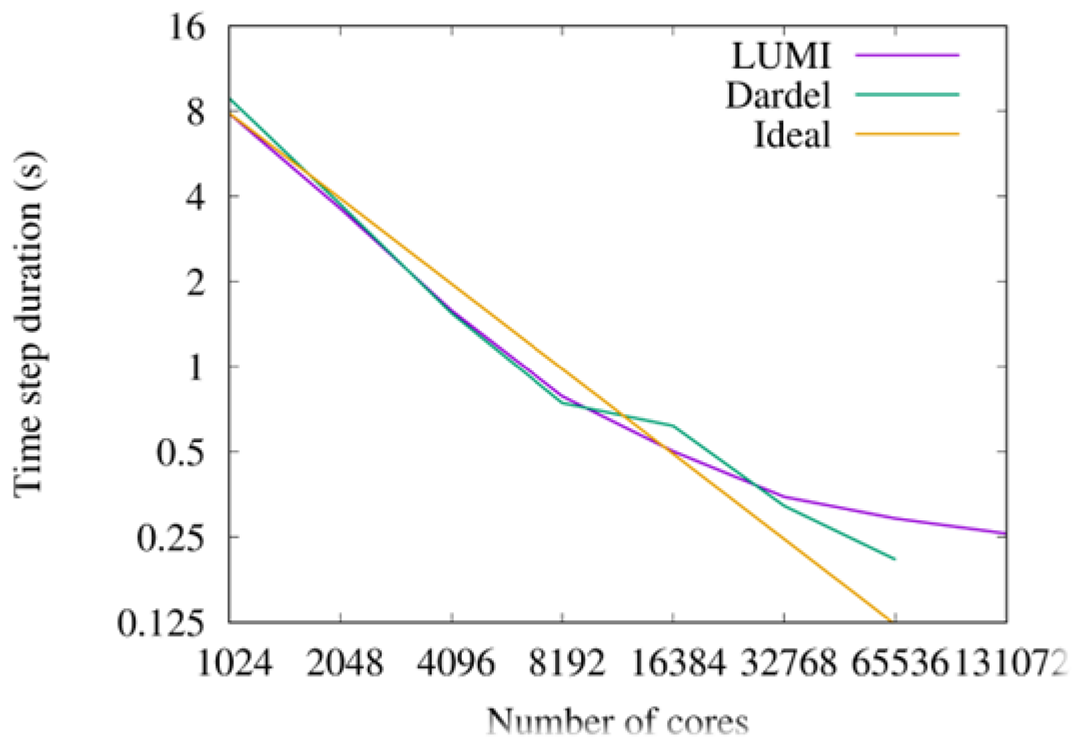


Burstedde, Wilcox, Ghattas, SISC, 2011

- p4est - AMR on forests of octrees
- Adaptive, **multi-block tree-code** to manage a quad- or octree of non-overlapping fixed sized grids
- Parallel, highly scalable on realistic applications; up to 220320 cores with parallel efficiency 70%
- **Refinement/coarsening performed locally**
- Element based with no geometrical restriction
- www.p4est.org
- C. Burstedde, L. Wilcox, and O. Ghattas. **p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees.** SIAM J. Sci. Comput., 33(3):1103–1133, 2011

Nek5000 implementation

Strong scaling on Dardel and LUMI

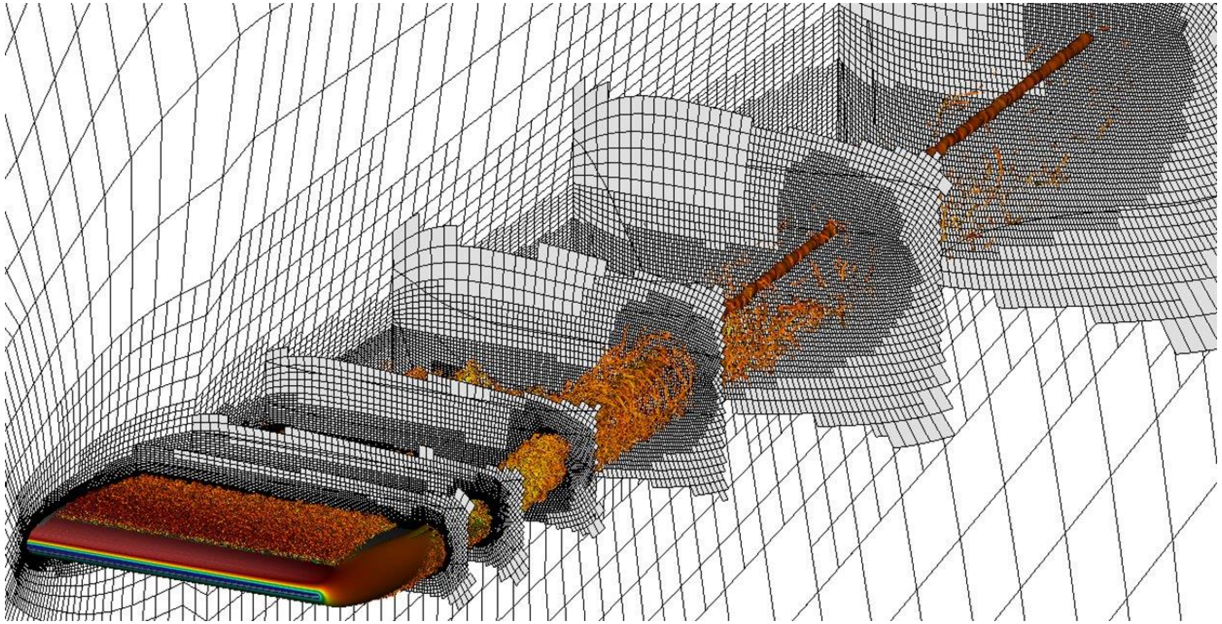


Performance of AMR run:

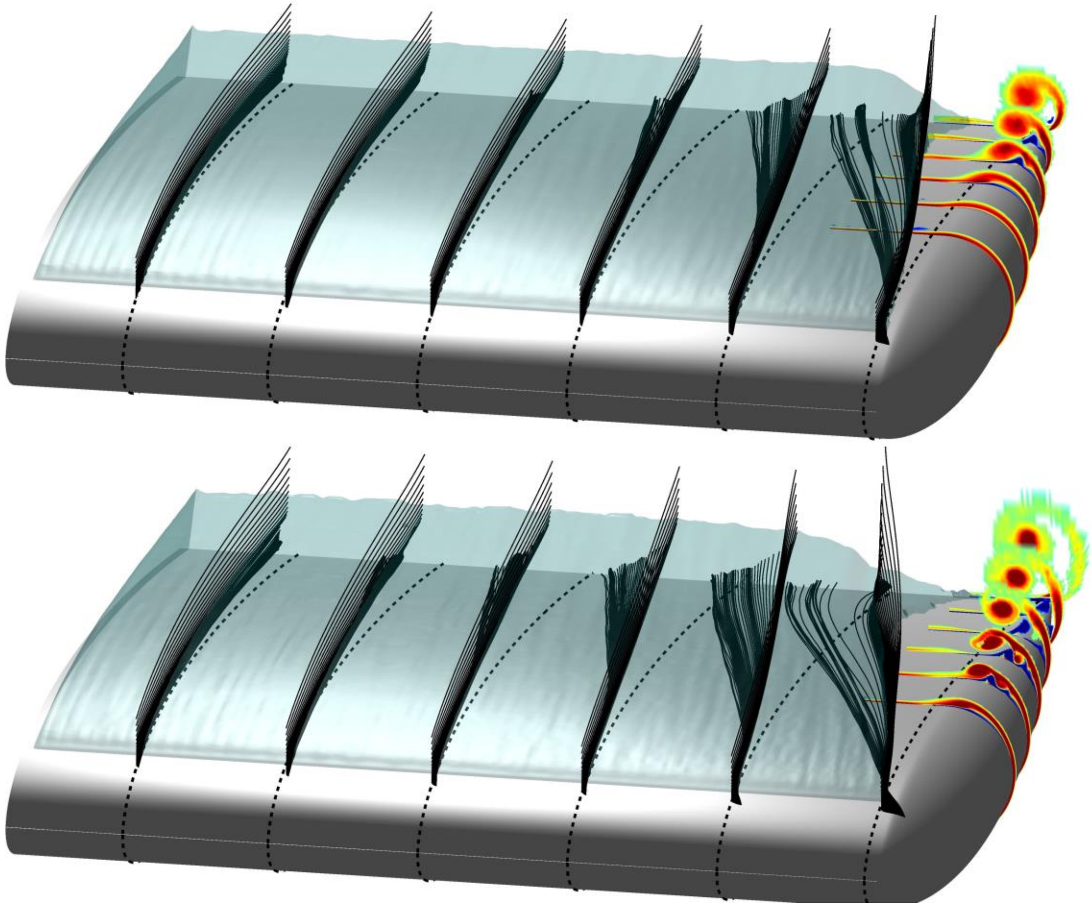
- 1024 cores
- Global element number (21 ref.)
  - Initial 113952
  - Final 131781
- Total simulation time 85242 sec
- **Total AMR time 1269 sec (1.5%)**
- Percentage of different operations (AMR – 100%):
  - I/O 5.1%
  - Error estimate 13.7%
  - Mesh regeneration 0.3%
  - Data transfer 0.5%
  - Solver restart 77.0%

NACA0012 airfoil with rounded tip

Mesh structure and vortical flow features

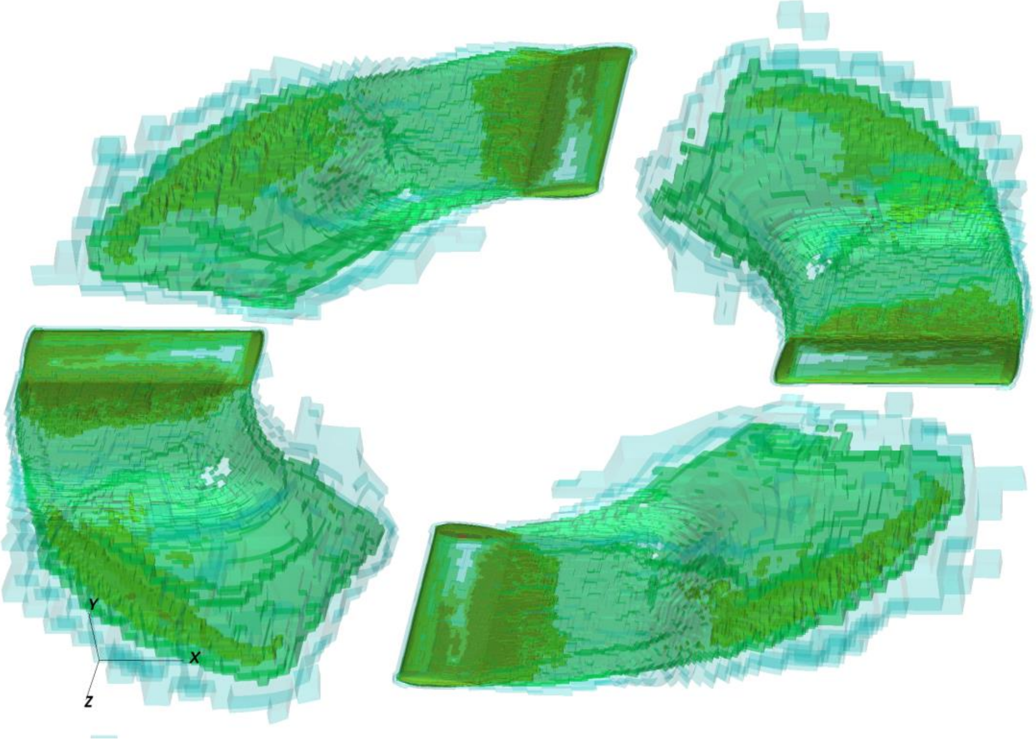


Flow lines of an averaged velocity field and position of a tip vortex

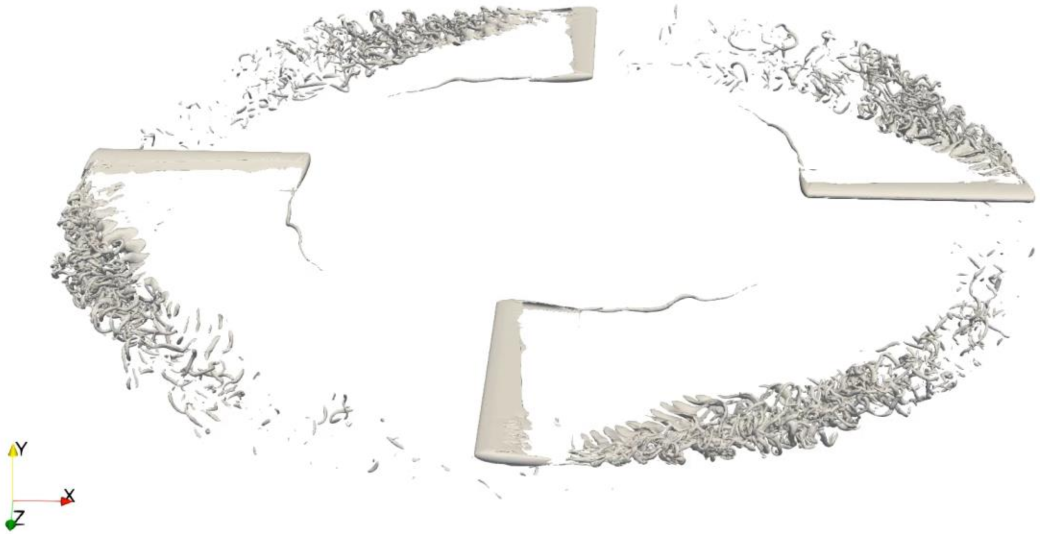


A simplified rotor with four blades

Domain regions covered by different refinement levels



Vortical structures

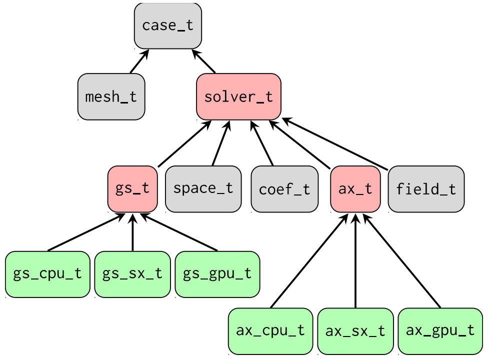


Current work on Neko

- High-order spectral element flow solver
  - Incompressible Navier-Stokes equations
  - Matrix-free formulation, **small tensor products**
  - **Gather-scatter** operations between elements
- Modern **object-oriented** approach (Fortran 2008)

```fortran
! Base type for a matrix-vector product providing Ax
type, abstract :: ax_t
  contains
    procedure(ax_compute), nopass, deferred :: compute
end type ax_t

! Abstract interface for computing Ax
abstract interface
   subroutine ax_compute(w, u, coef, msh, Xh)
     implicit none
     type(space_t), intent(inout) :: Xh
     type(mesh_t),  intent(inout) :: msh
     type(coef_t),  intent(inout) :: coef
     real(kind=dp), intent(inout) :: w(:,:,:,:)
     real(kind=dp), intent(inout) :: u(:,:,:,:)
   end subroutine ax_compute
end interface
```

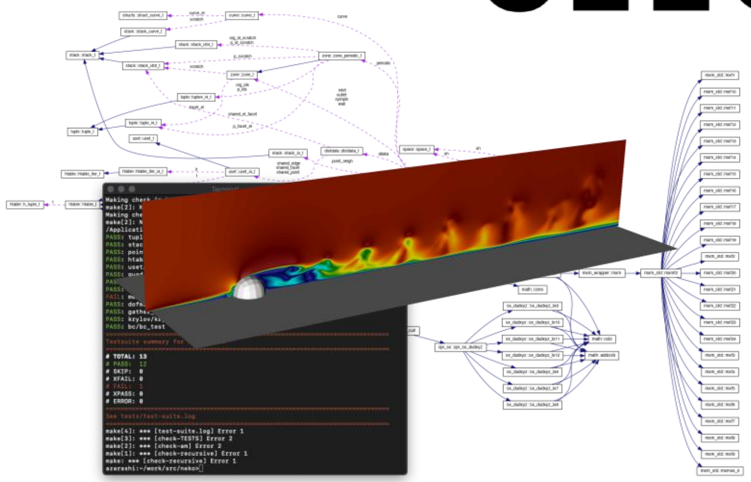


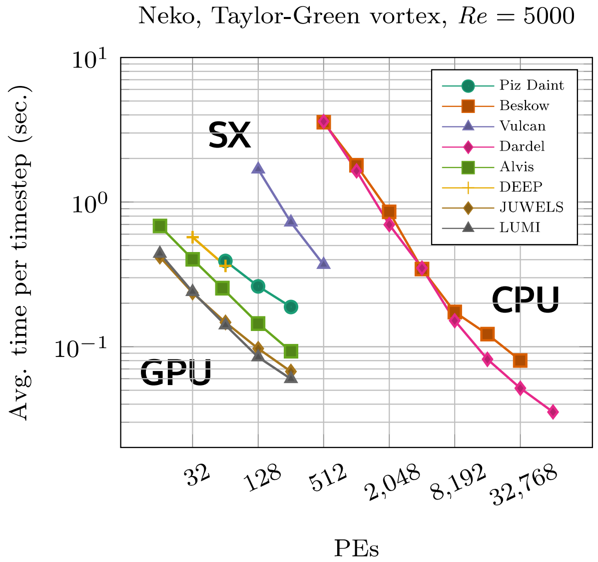
- Various hardware-backends
  - CPUs, GPUs down to exotic vector processors and FPGAs
    - **Device abstraction layer** for accelerators (CUDA/HIP/OpenCL)
- Modern Software Engineering (pFUnit, ReFrame, **Spack**)

> spack install neko+cuda

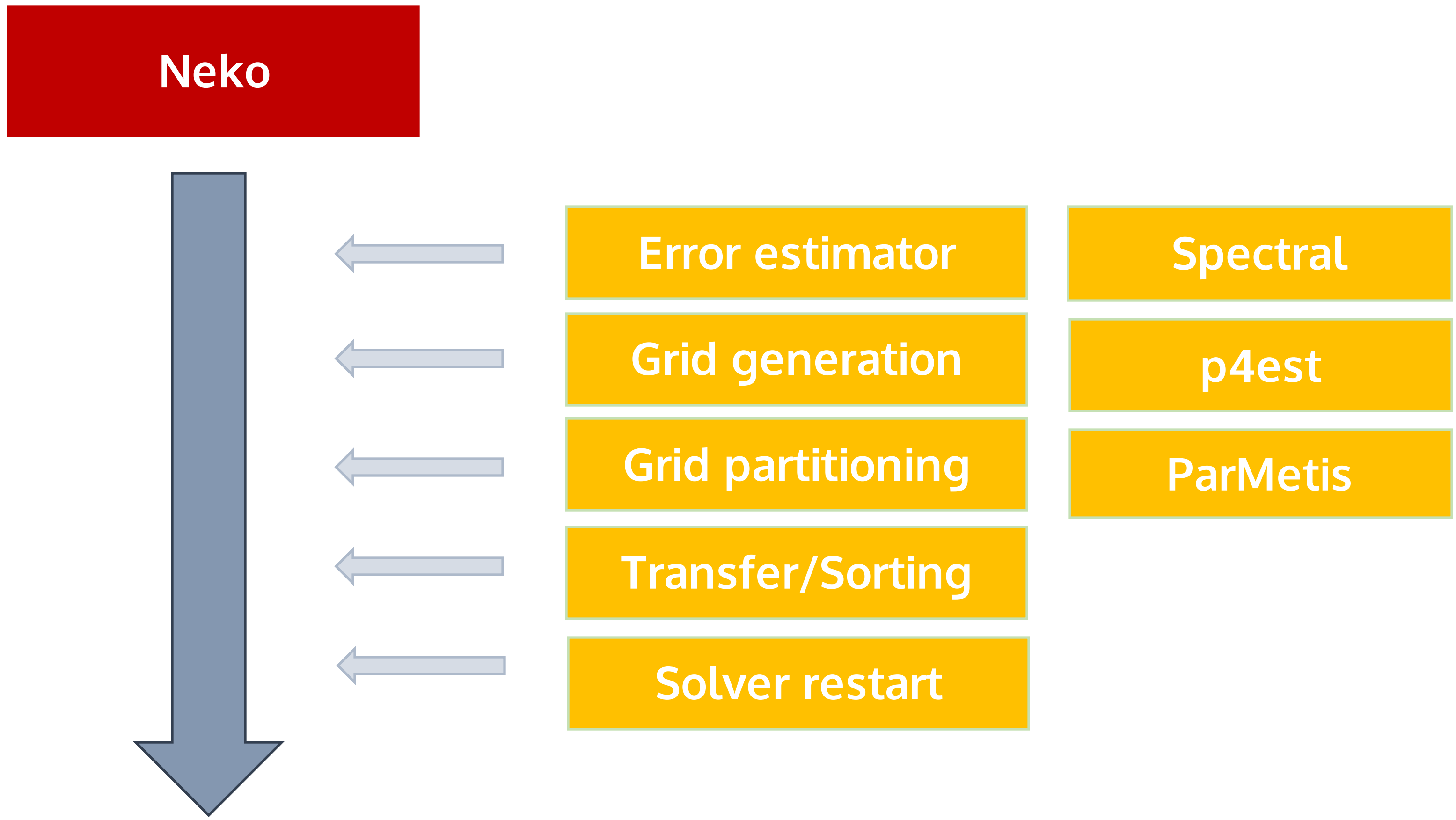
> spack install neko+cuda

ExtremeFLOW/neko

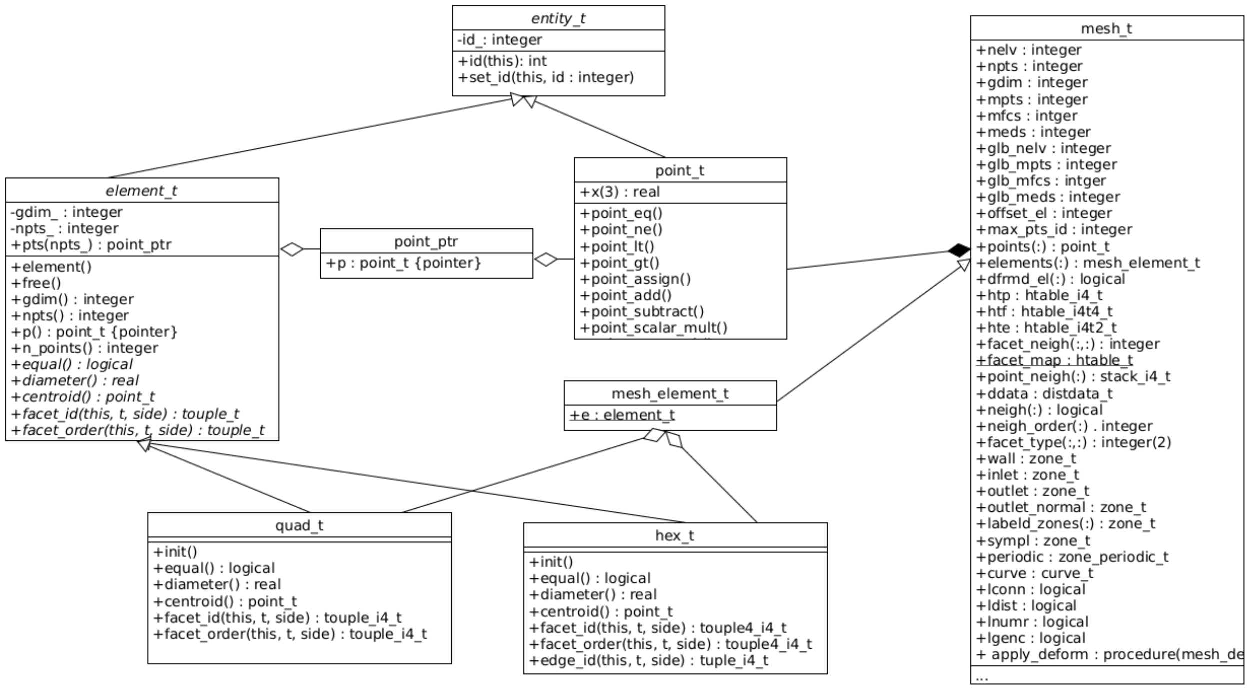




Current work on Neko



Current work on Neko

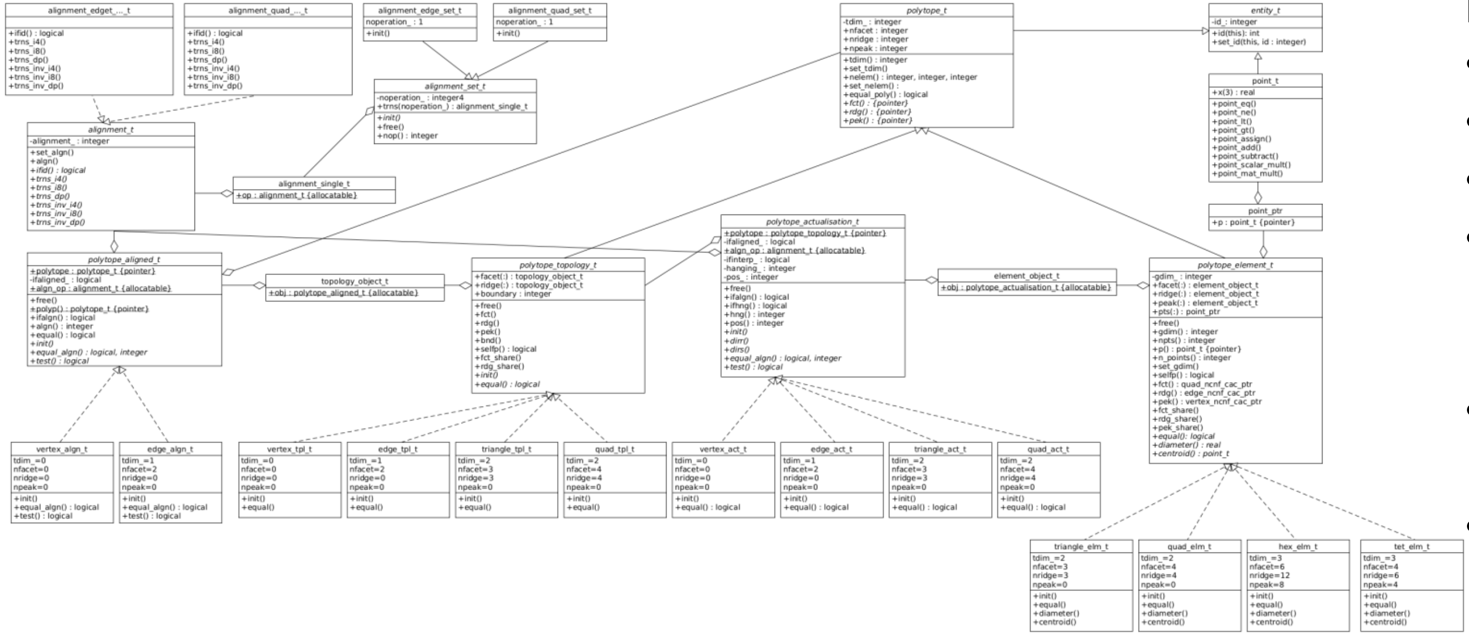


Class diagram for mesh type

Mesh description:

- Compact and flexible
- Well suits conforming meshes
- Combines topology and geometry
- Connectivity defined by global id of a physical point
  - No easy way to describe hanging nodes (element local perspective needed)
- Straightforward to import conforming p4est data, but problem with nonconforming meshes

Current work on Neko



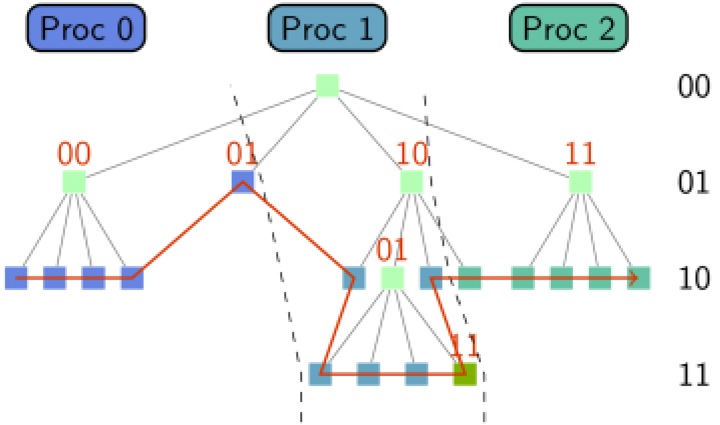
Class diagram for mesh type

New mesh description:

- Flexible, but complex
- Supports nonconforming meshes
- Splits topology and geometry
- Distinguishes abstract object and its realisation (element local perspective)
- Straightforward to import nonconforming p4est data
- Under construction

Current work on Neko

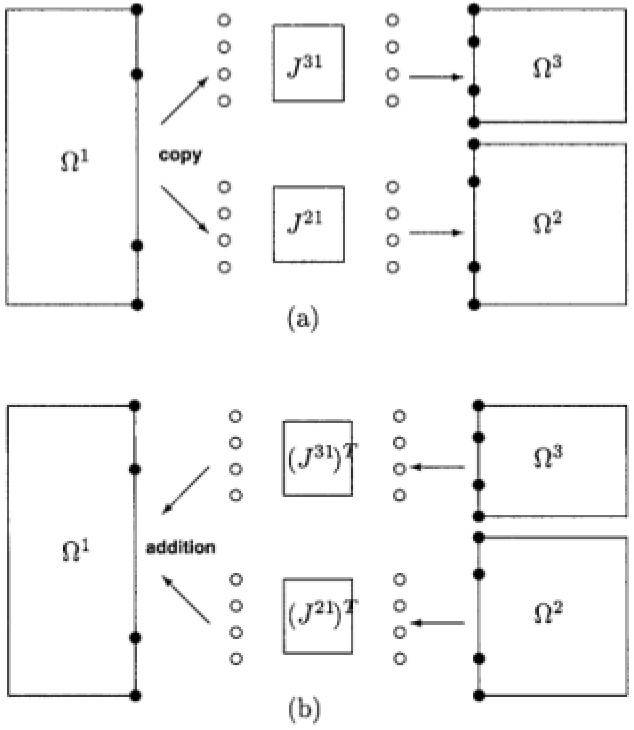
AMR workflow:
- Collect error indicator on GPU
- Perform refinement and mesh reconstruction on CPU
- Reconstruct simulation variables on GPU
- Restart the solver



Schematic representation of element distribution among MPI ranks.
Burstedde, Wilcox, Ghattas, SISC, 2011

Main challenges:
- Interpolation operator for direct stiffness summation
  - Frequent, performed on GPU
  - Interpolation operator acting on a small amount of element faces and edges
  - Four different 2D operators for faces and two 1D operators for edges
- Reconstruction of the simulation variables
  - Rare, performed on GPU
  - 3D interpolation operator acting on a small amount of elements
  - Requires communication
  - Three steps: refinement, global redistribution, coarsening
- Solver restart

Current work on Neko

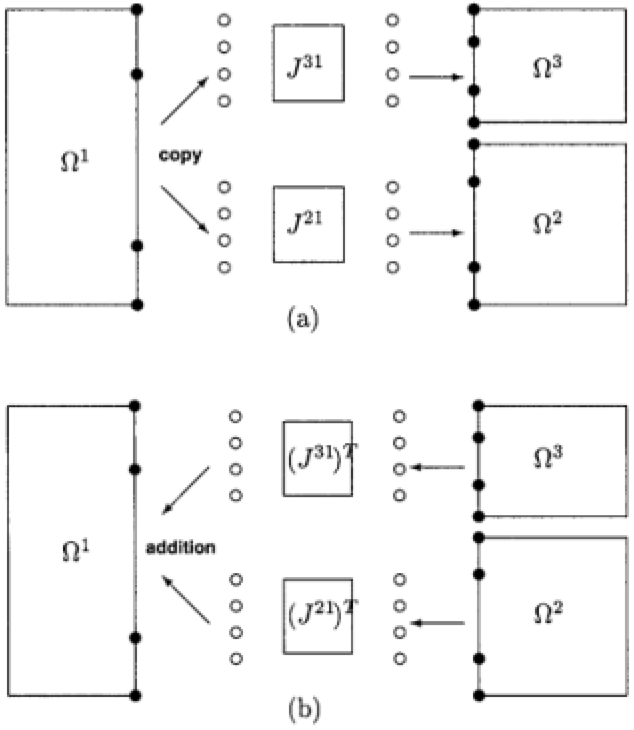


Data exchange at nonconforming interface
Fischer at al 2002

Communication kernels refactoring:

- Direct stiffness summation based on a single point information
- Nonconforming solver introduces interpolation operator acting on objects (element faces and edges)
- Challenging on GPUs as interpolation operator acts on a small data sets randomly distributed in a big array
- Work imbalance in the interpolation operator

Current work on Neko



Data exchange at nonconforming interface
Fischer at al 2002

- Data for interpolation operation not contiguous and randomly ordered
- Performance will be dominated by memory latency
  - Necessary to have as many as possible memory operations overlapping to get maximum memory bandwidth
- Possible ways for data exchange:
  - Keep high point multiplicity and treat nonconforming interfaces separately
  - Lower face multiplicity by separately numbering each of the children faces

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and Sweden, Germany, Spain, Greece, and Denmark under grant agreement No 101093393.

Co-funded by
the European Union

EuroHPC
Joint Undertaking

Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European High Performance Computing Joint Undertaking (JU) and Sweden, Germany, Spain, Greece, and Denmark. Neither the European Union nor the granting authority can be held responsible for them.

CEEC

Centre of Excellence in Exascale CFD

Thank you
for your attention!

CEEC

Centre of Excellence in Exascale CFD