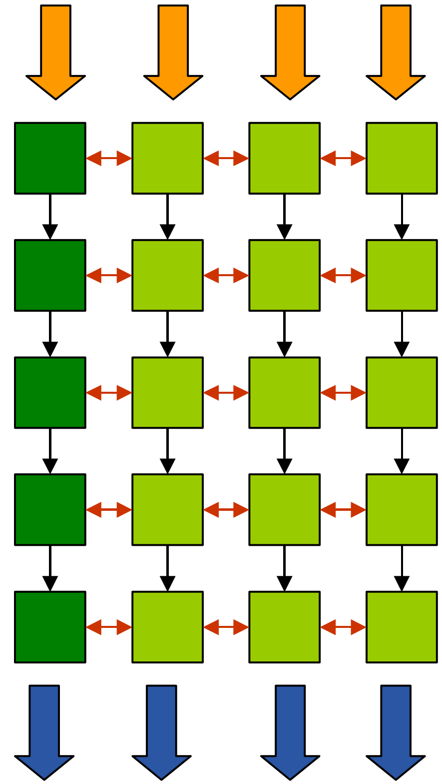
Further developments in UnTRIM: parallel implementation and its verification



Jacek A. Jankowski

BAW

Department of Inland Waterways Engineering

Fifth International Symposium on Environmental Hydraulics

Tempe, Arizona, USA, 4-7 December 2007

UnTRIM philosophy

Developed by
Vincenzo Casulli
University of Trento

A possibly short, precise definition

UnTRIM is a practical scheme for solving three dimensional equations describing free-surface flows with a semi-implicit, fractional time step integration, a finite volume/difference spatial discretisation and a semi-Lagrangian treatment of advection using an unstructured, orthogonal grid.

UnTRIM algorithm

...practical scheme...

a compromise between *stability – accuracy - efficiency*

...semi-implicit formulation...

treat terms controlling stability *implicitly*
and the remaining ones (e.g. advection) *explicitly*

...fractional step...

split pressure into *hydrostatic and dynamic* components
use *wave equation* in the hydrostatic part

UnTRIM algorithm

...finite difference discretisation...

finite difference methods for unstructured meshes

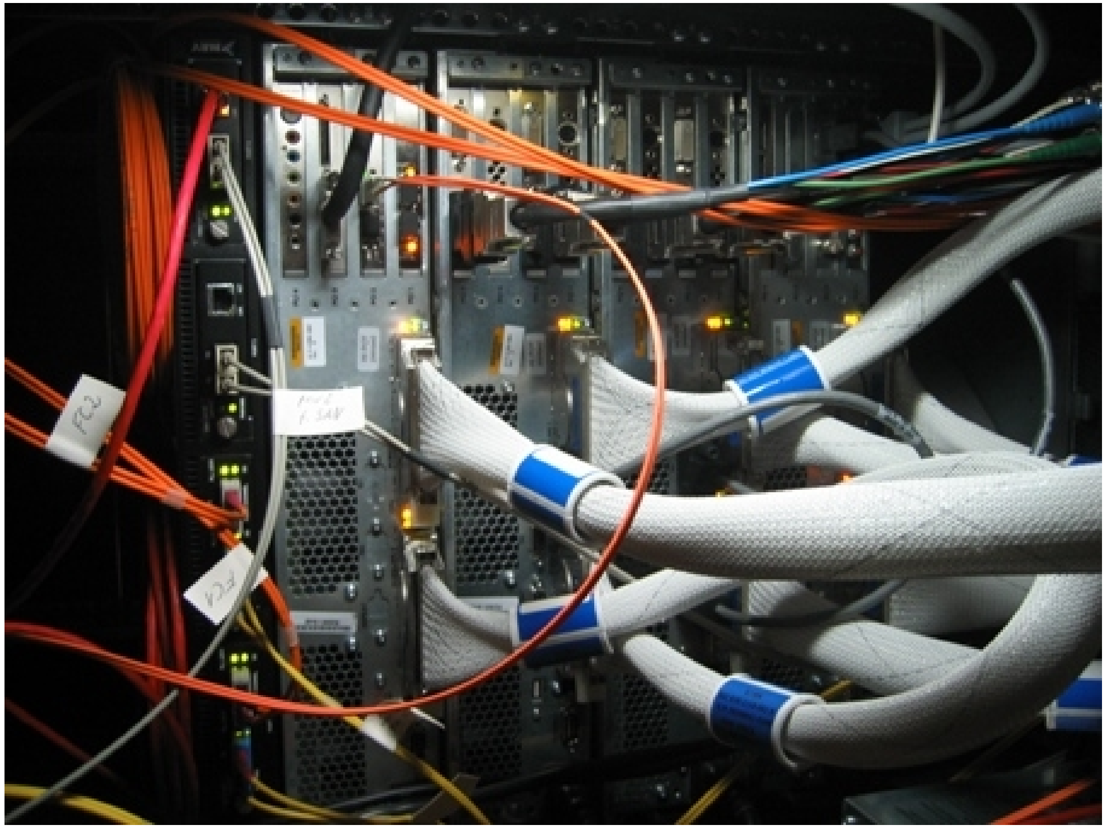
...finite volumes...

for the continuity equation – local and global mass conservation guaranteed

...semi-Lagrangian advection...

streamline tracking backward in time, unconditionally stable

Parallel computing



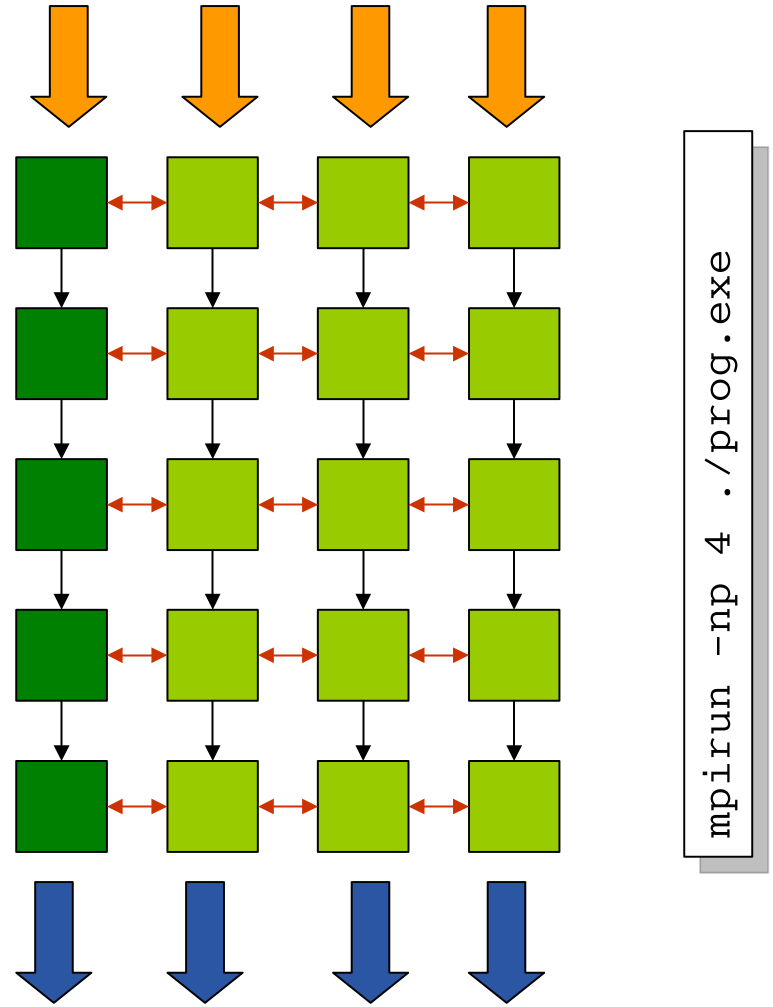
UnTRIM developed as a serial code

BAW

Message-passing parallelism

- Each processor executes a program copy with its own data

- Communication limits the scalability of the code
  - preparing data for sending
  - communication *itself*
  - integrating the received data

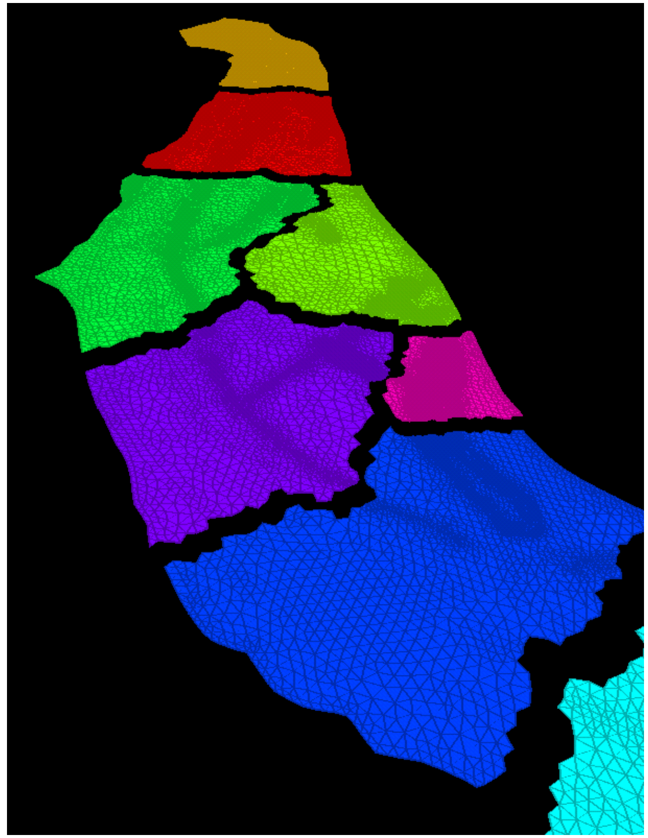


Domain decomposition method

- Parallel implementation with domain decomposition and overlapping mesh partitions

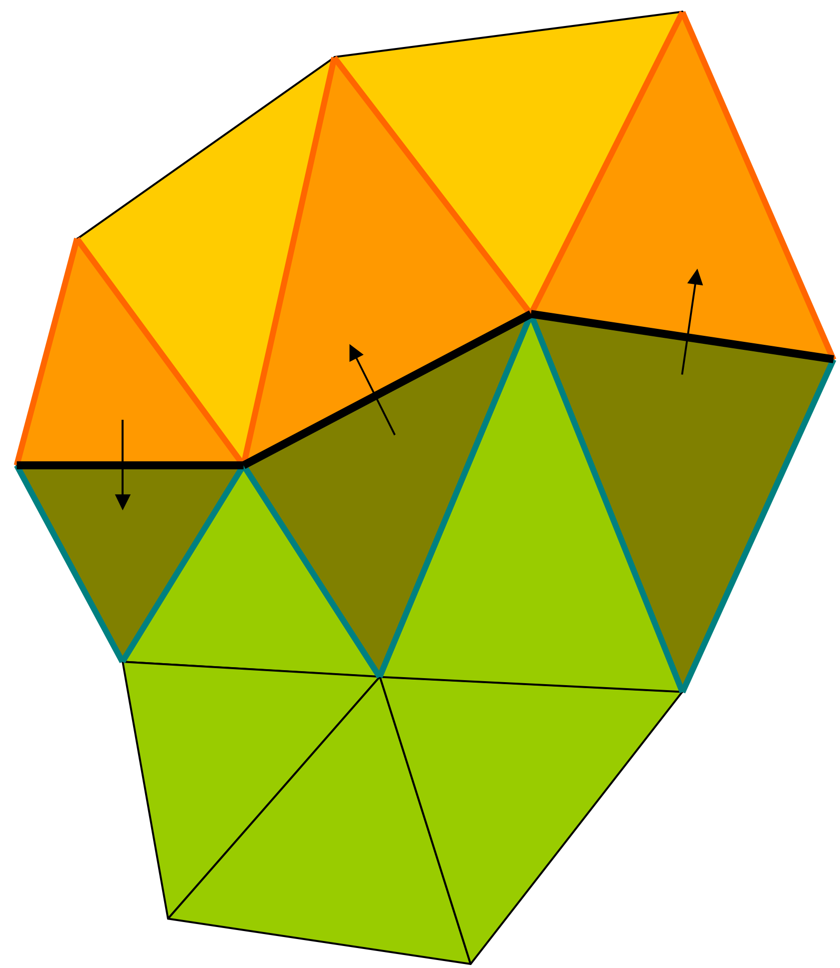
- This leads to point-to-point communication between neighbouring partitions

- Semi-Lagrangian advection methods do not fit well to this scheme: global communication

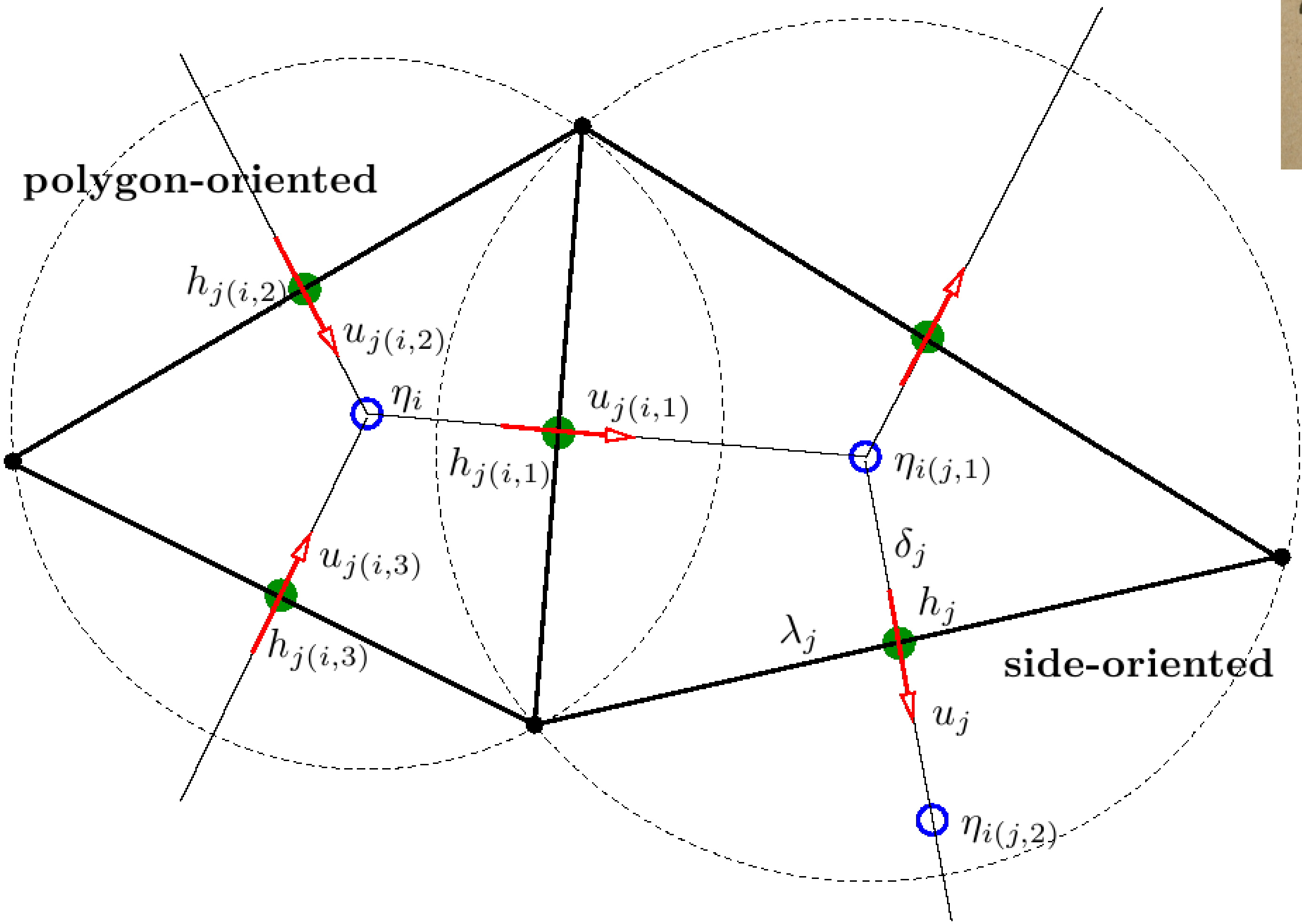


Point-to-point communication

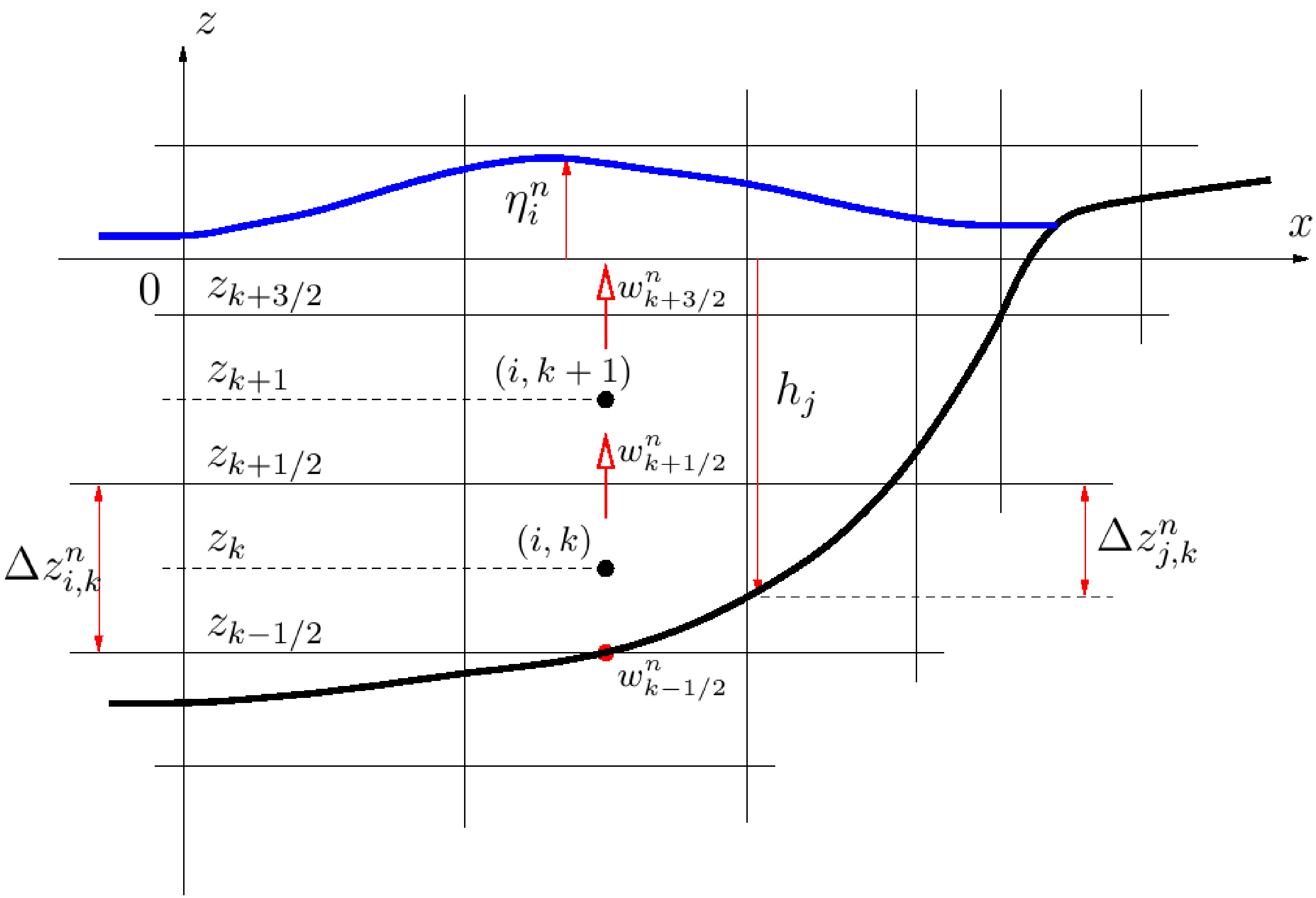
Dealing with the Finite Volumes/Differences Methods



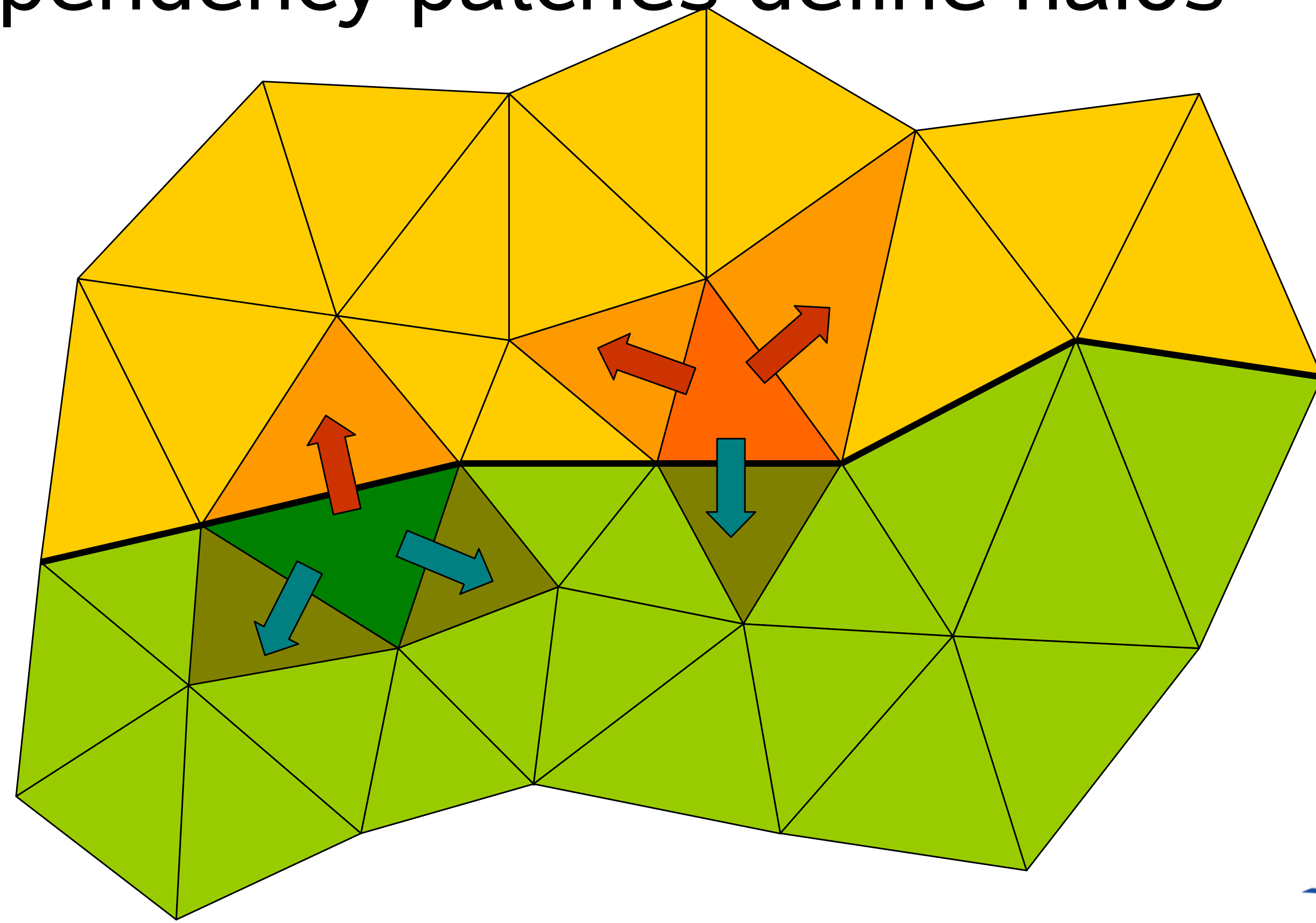
Orthogonal, staggered grid



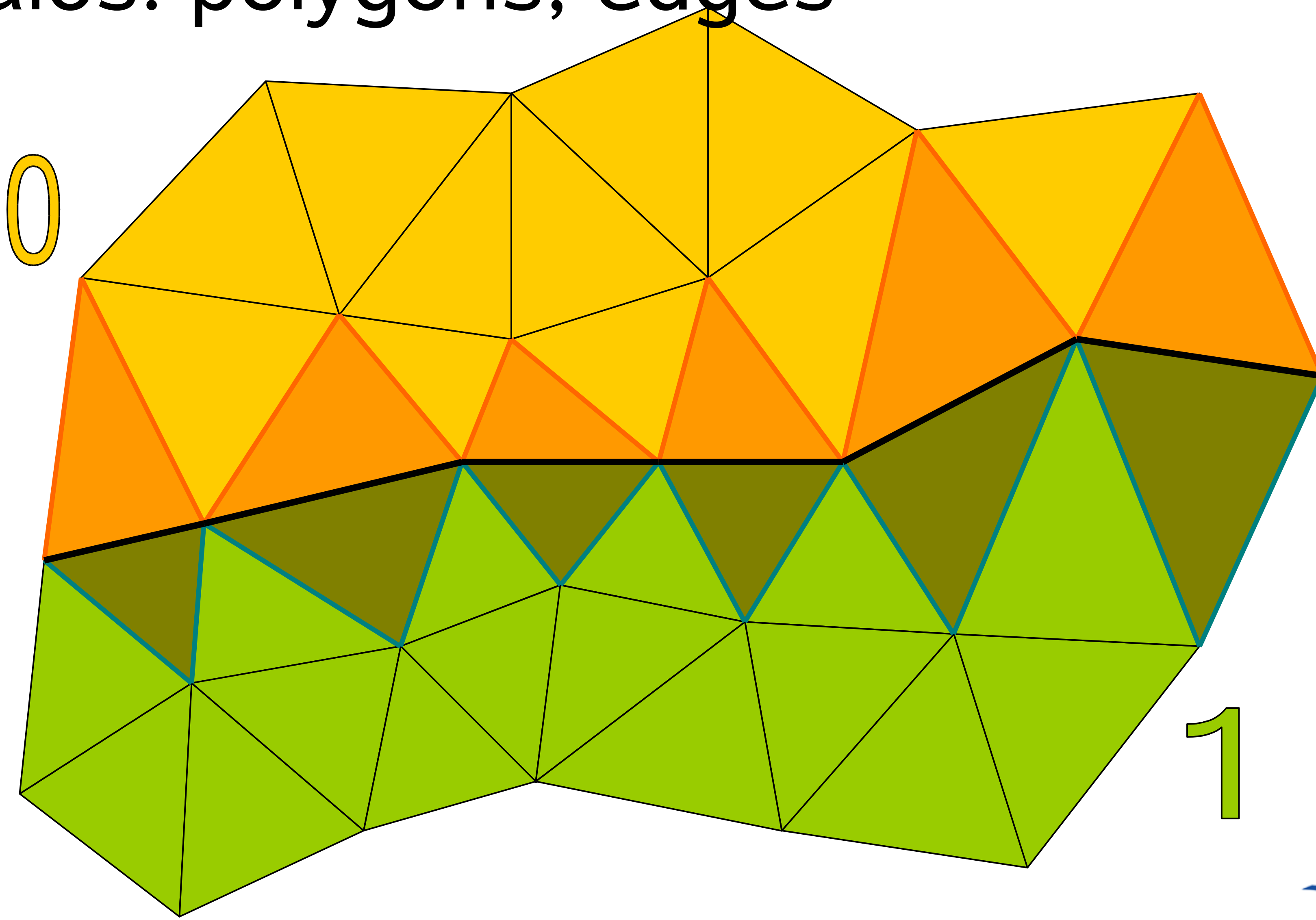
Horizontal layers of prisms



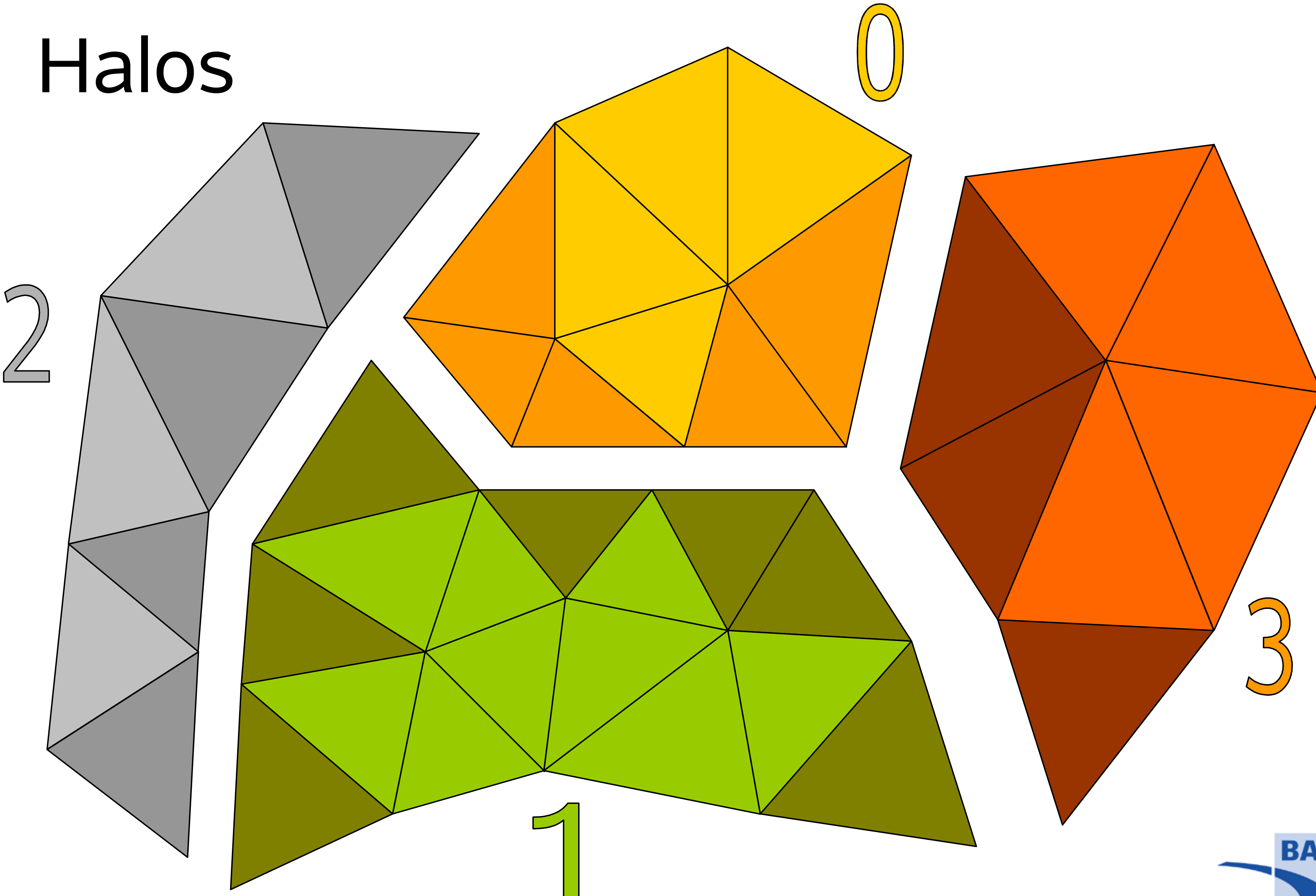
Dependency patches define halos



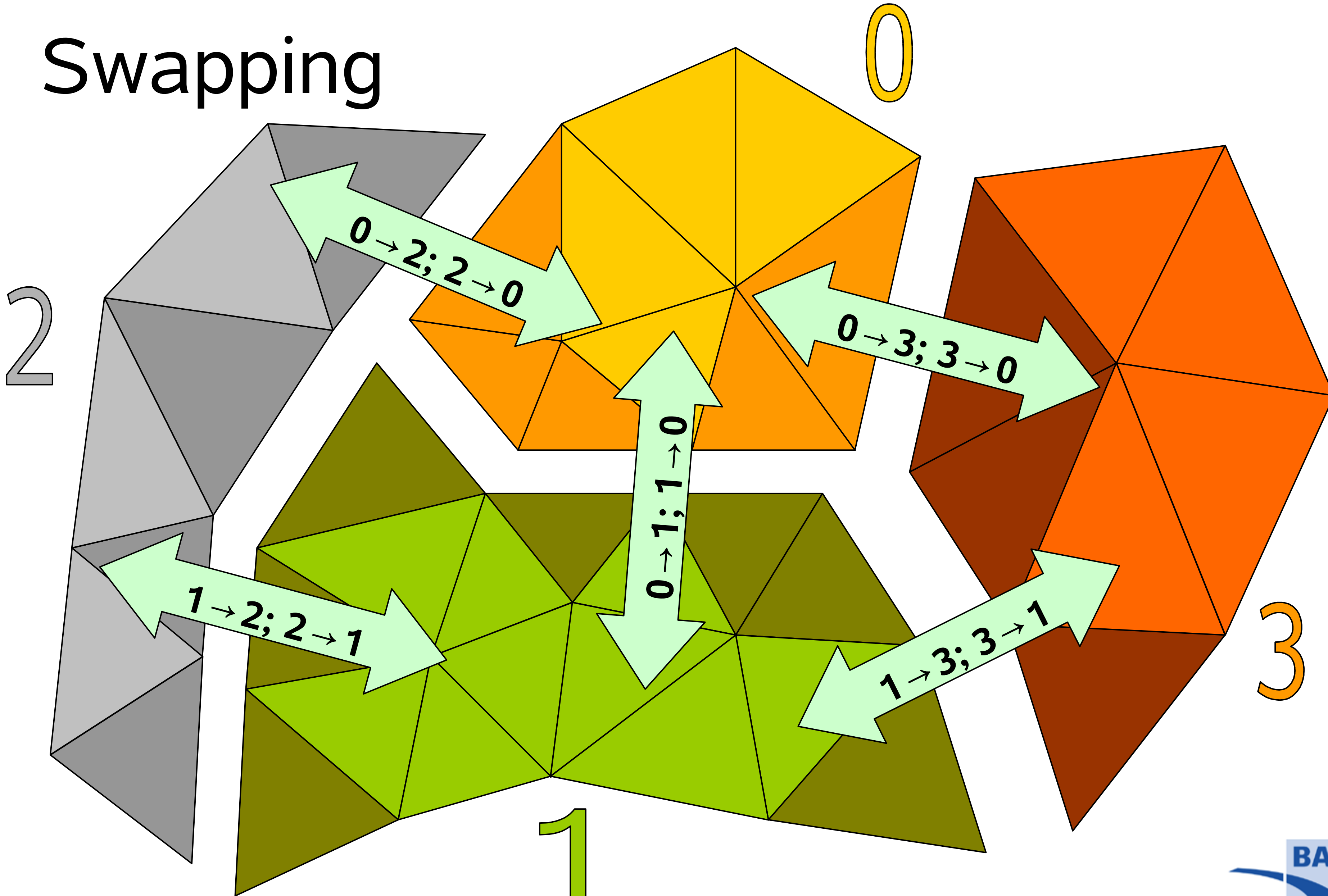
Halos: polygons, edges



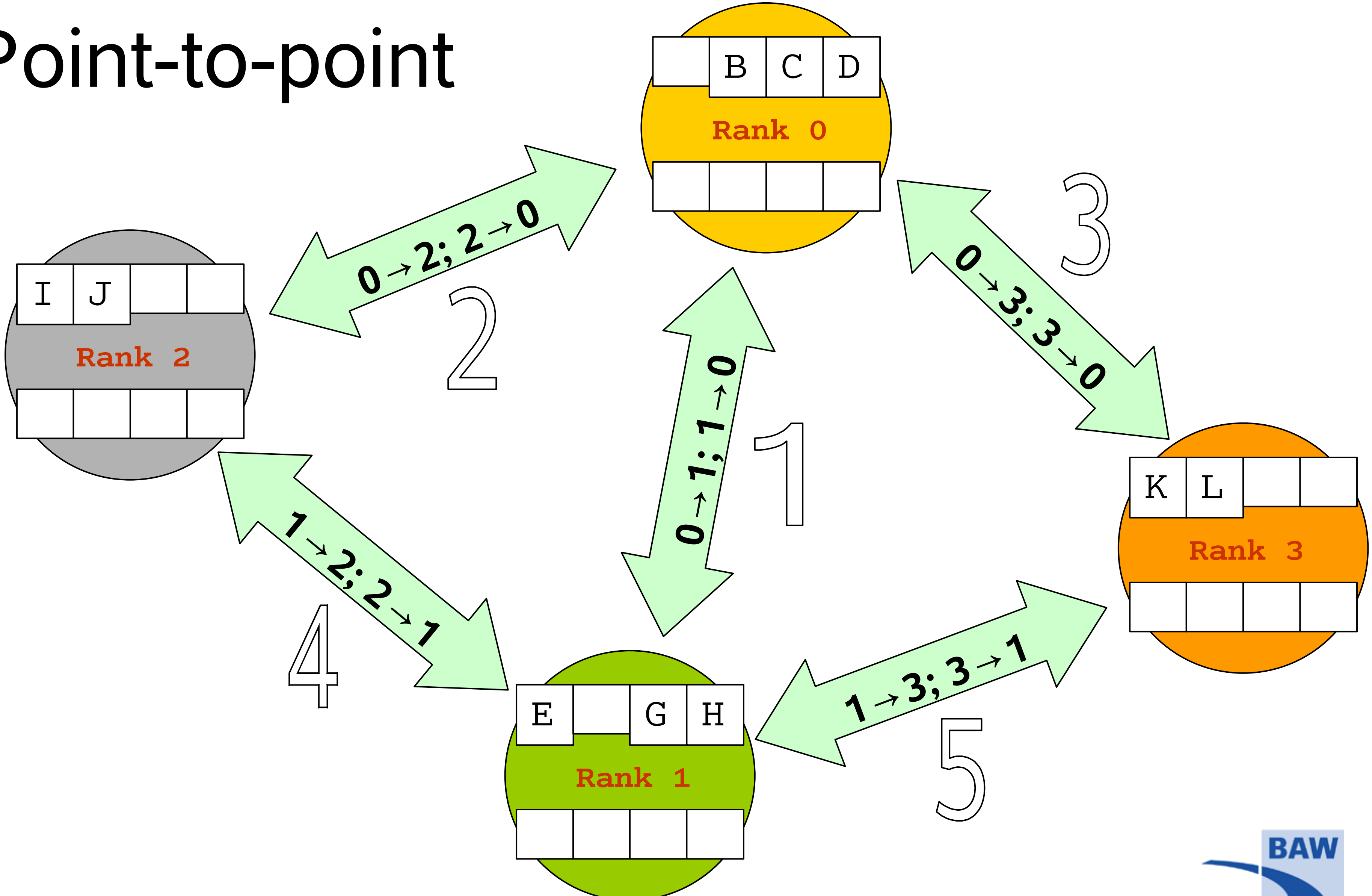
Halos



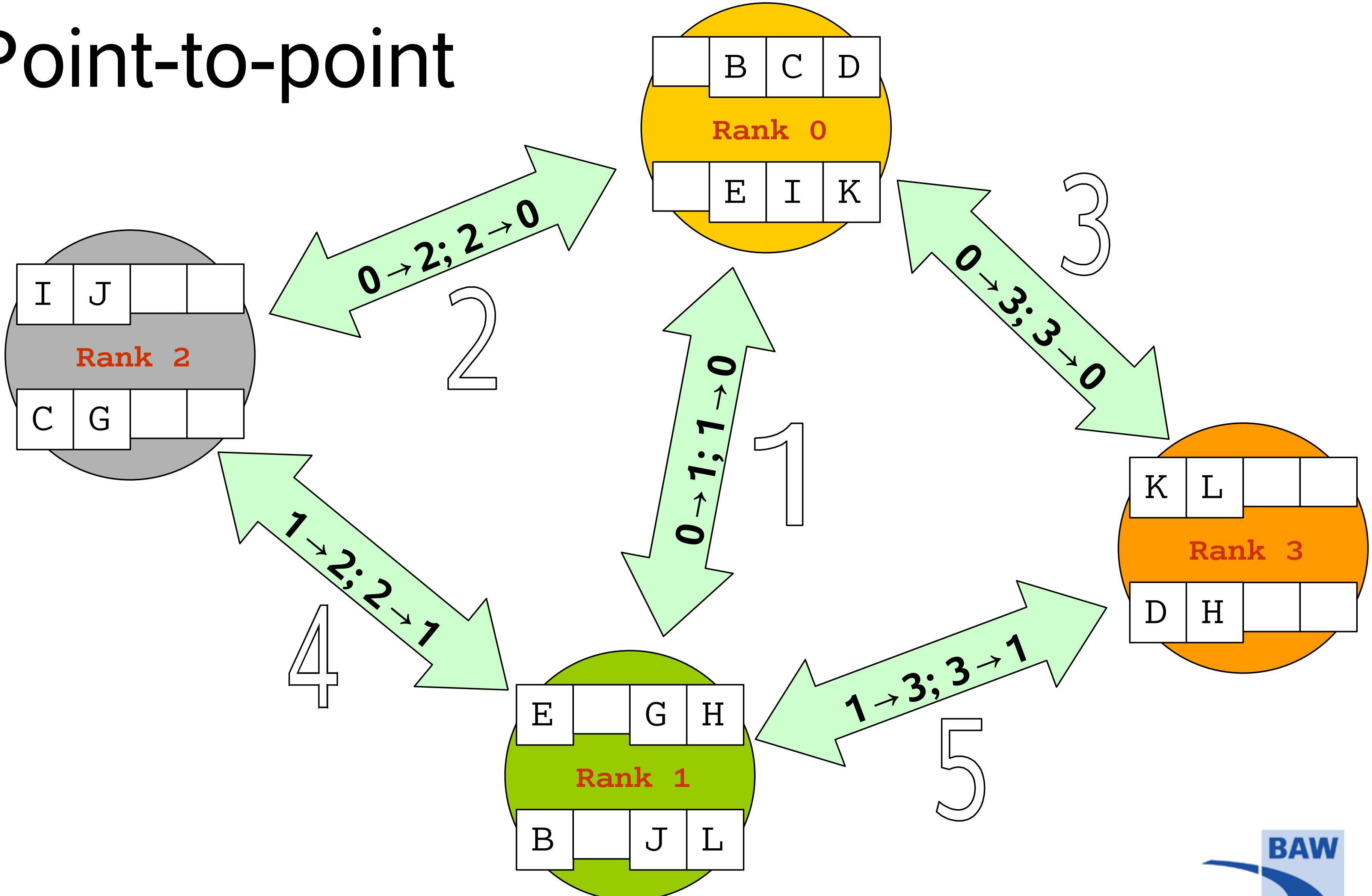
Swapping



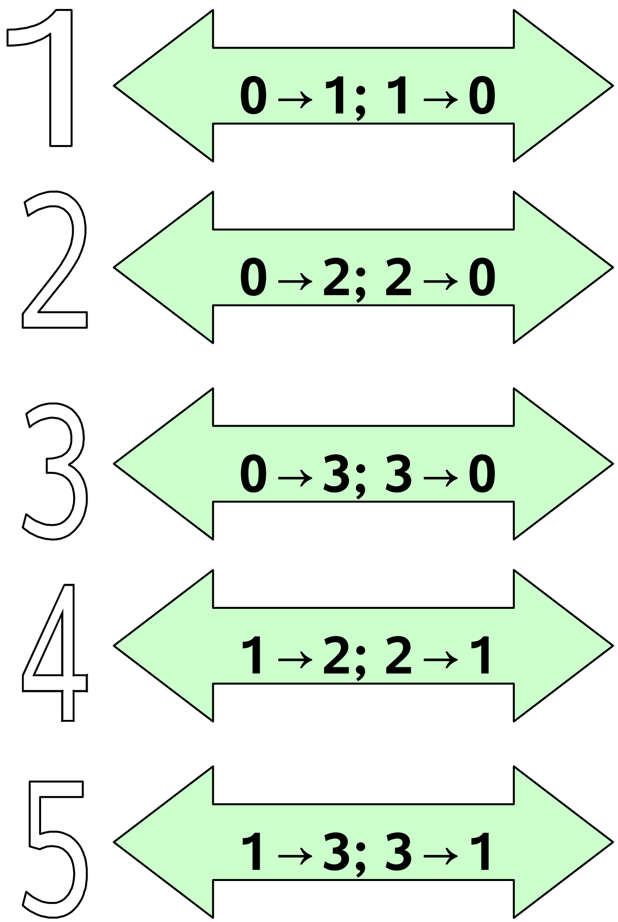
Point-to-point



Point-to-point

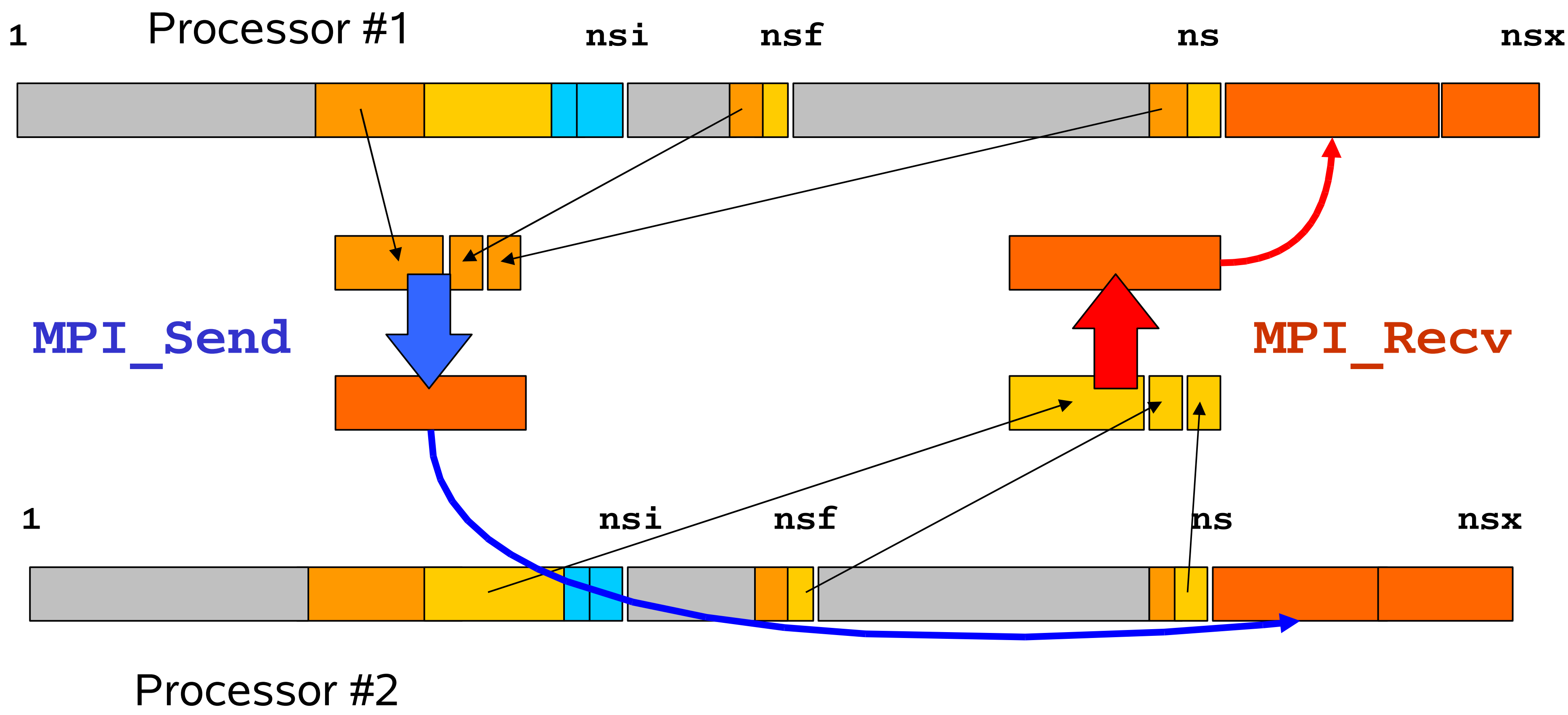


MPI_SendRecv



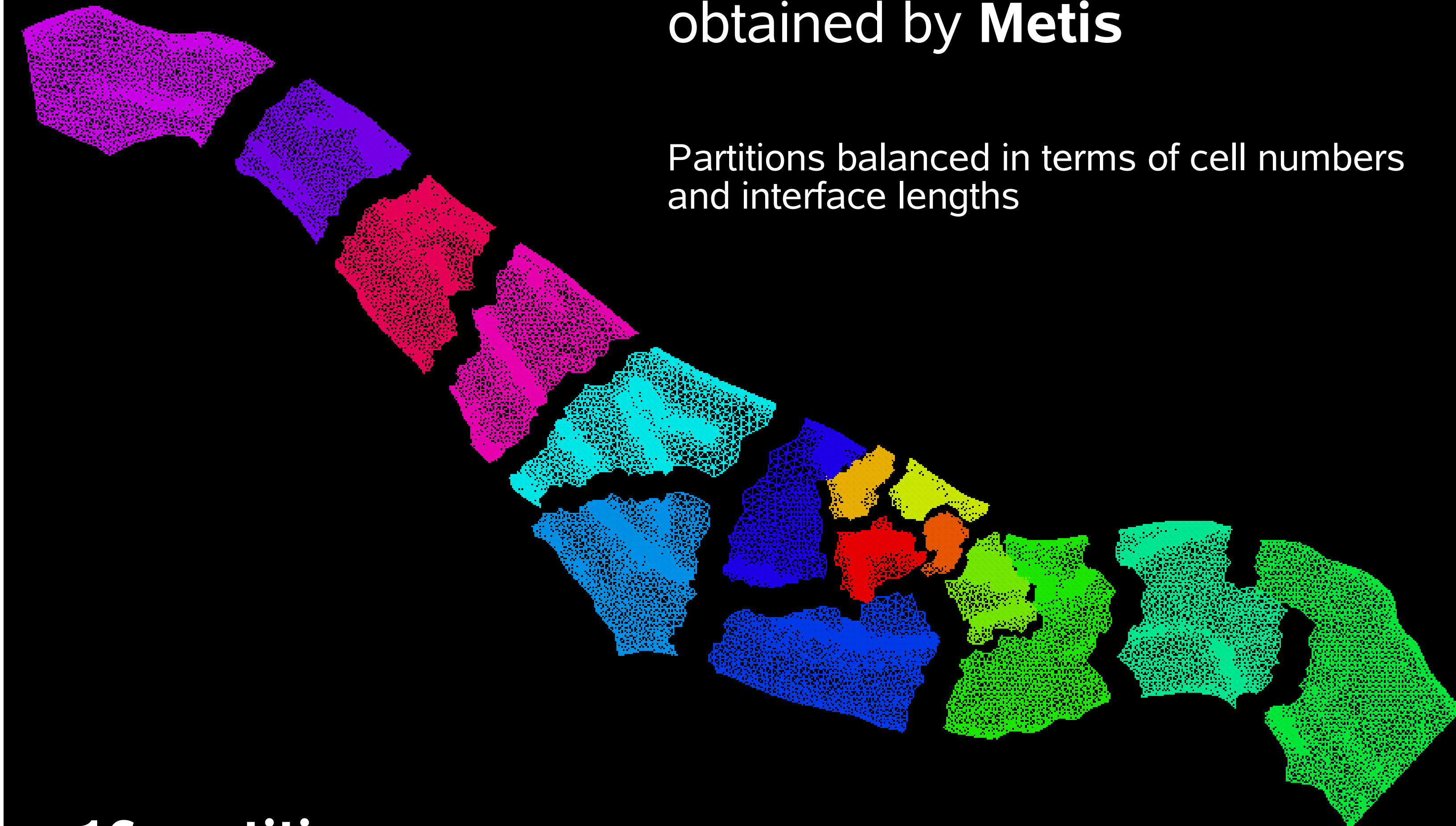
- Point-to-point communication
- Halo swapping in the order of direct neighbour pairs
- Objects are polygons, edges, cells, faces…

MPI_SendRecv with Buffers



Domain decomposition obtained by **Metis**

Partitions balanced in terms of cell numbers and interface lengths



16 partitions

Global communication

Dealing with the Lagrangian advection

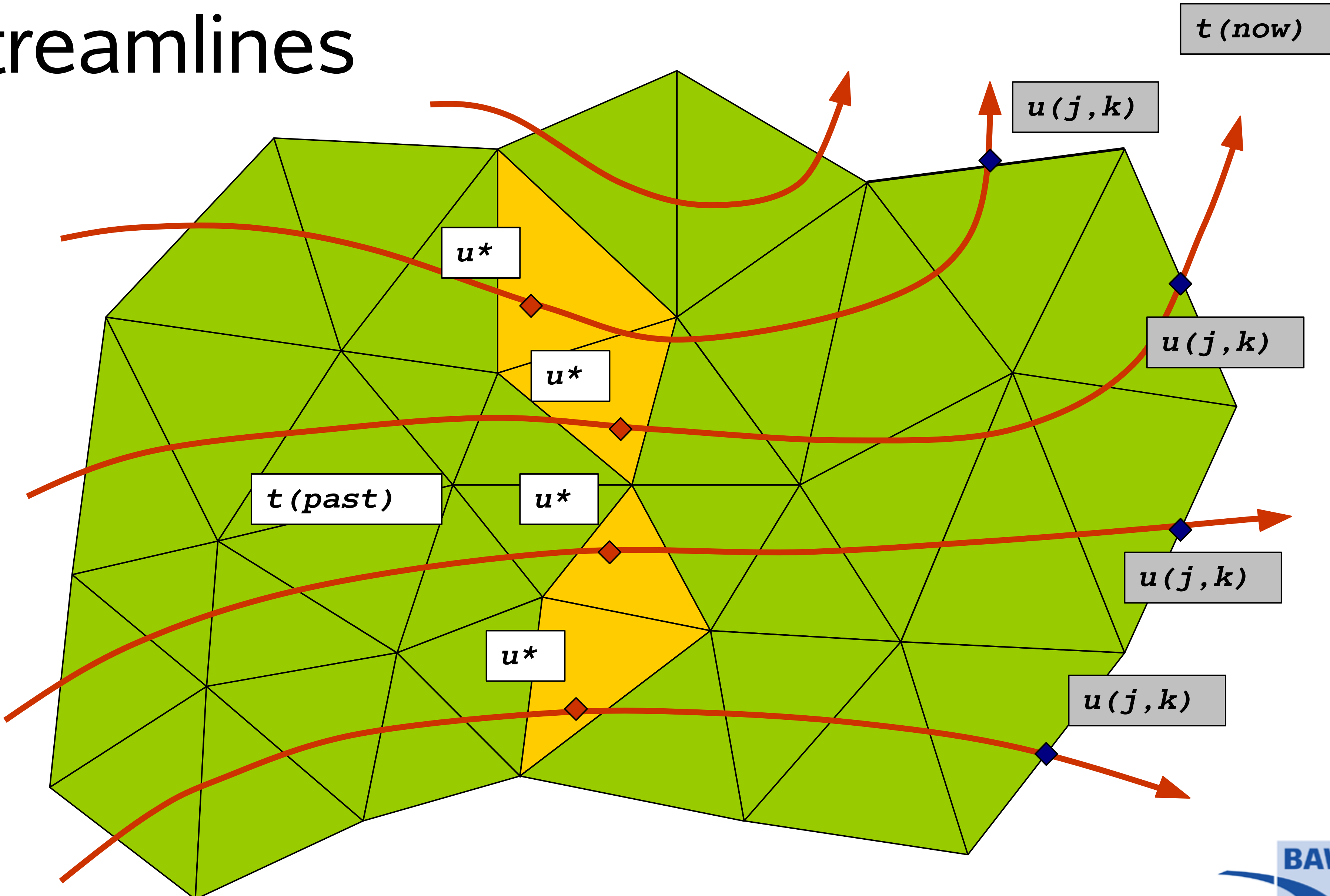
A semi-Lagrangian advection treatment

The pure advection – variable values remain constant along a streamline

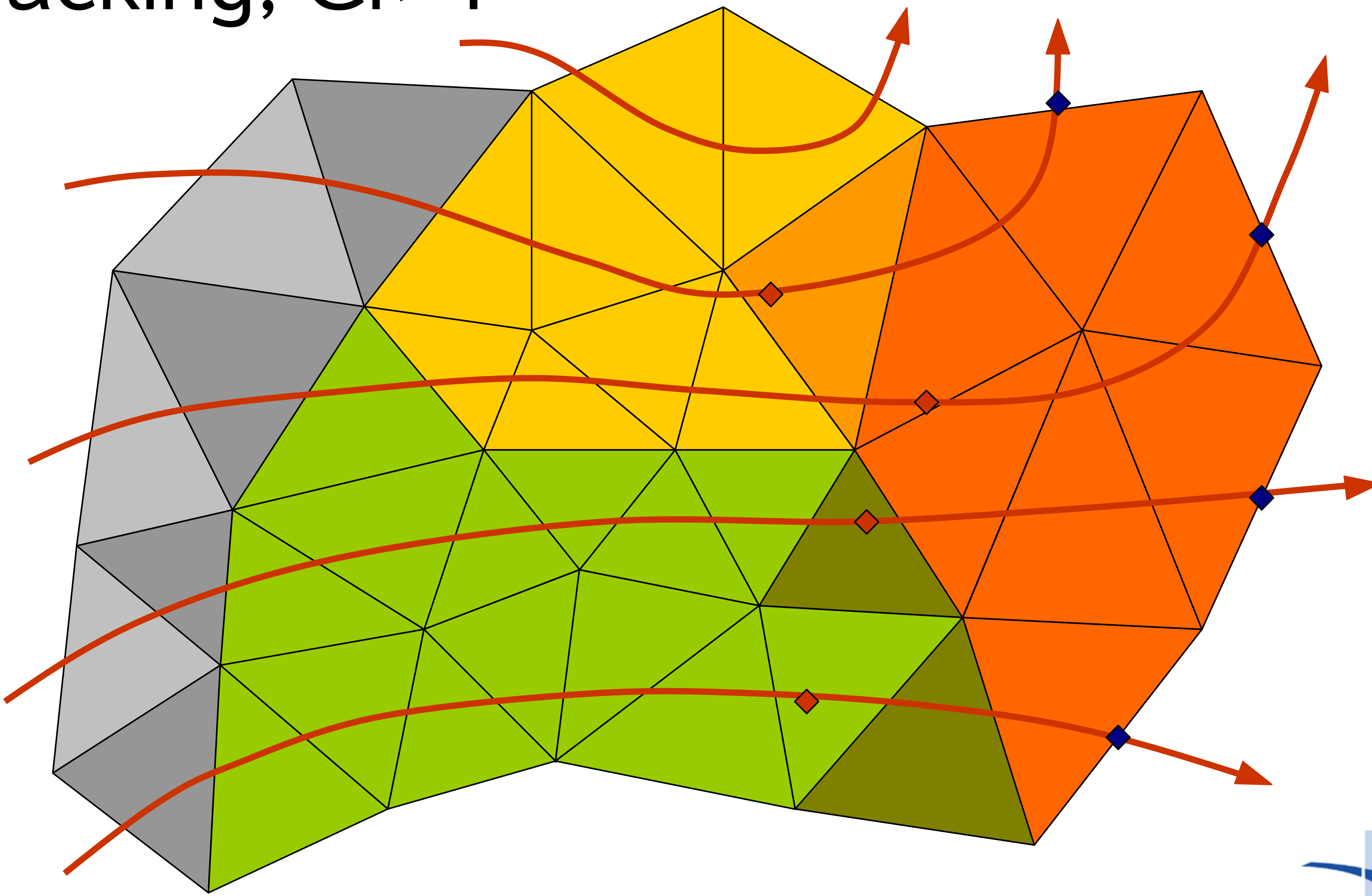
streamline tracking over the mesh – backward in time
interpolating the value at a located point in the mesh
applying the found value further on

semi – actually, the *discretised* values are applied (Eulerian mesh)

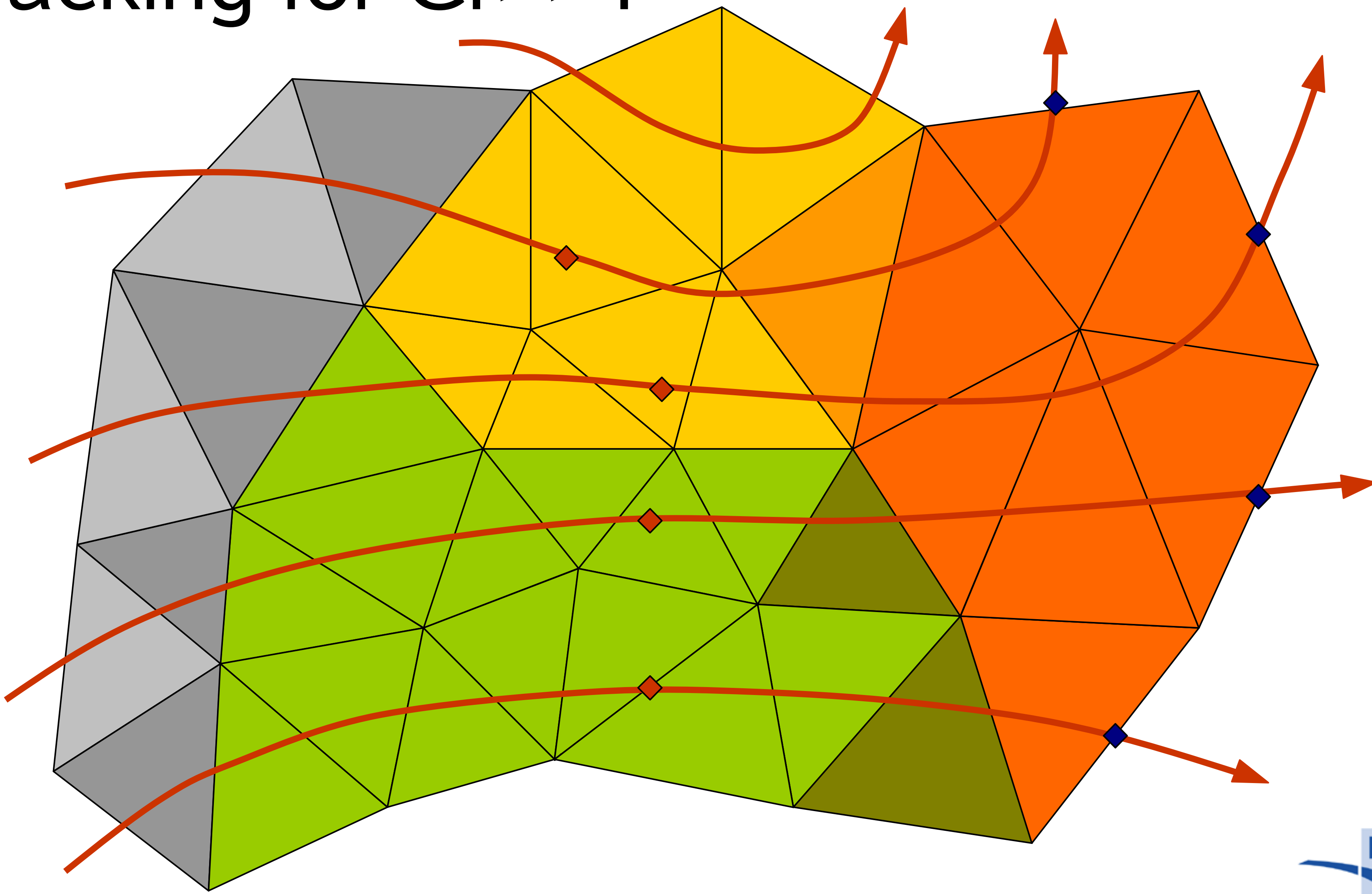
Streamlines



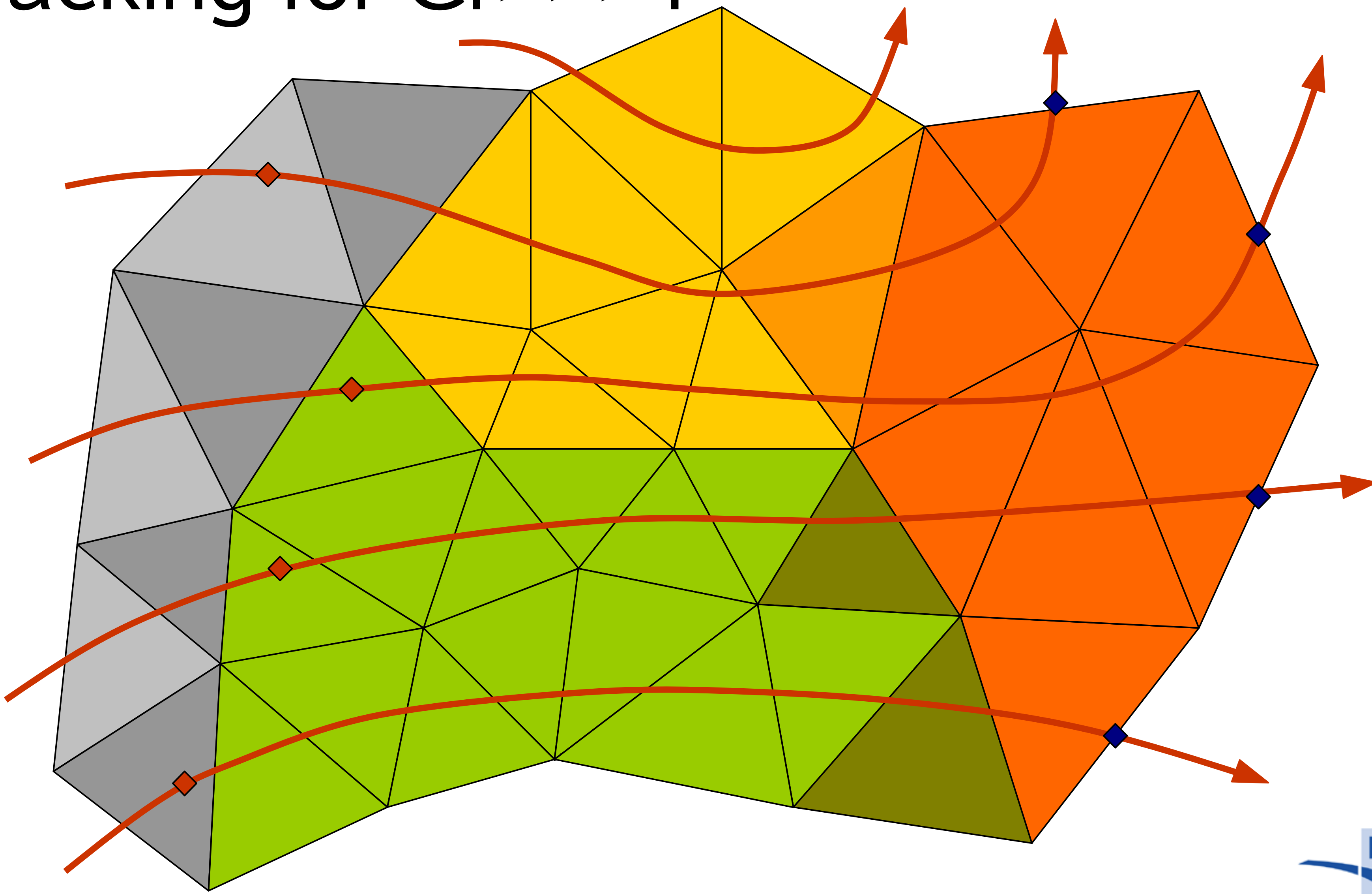
Tracking, Cr>1



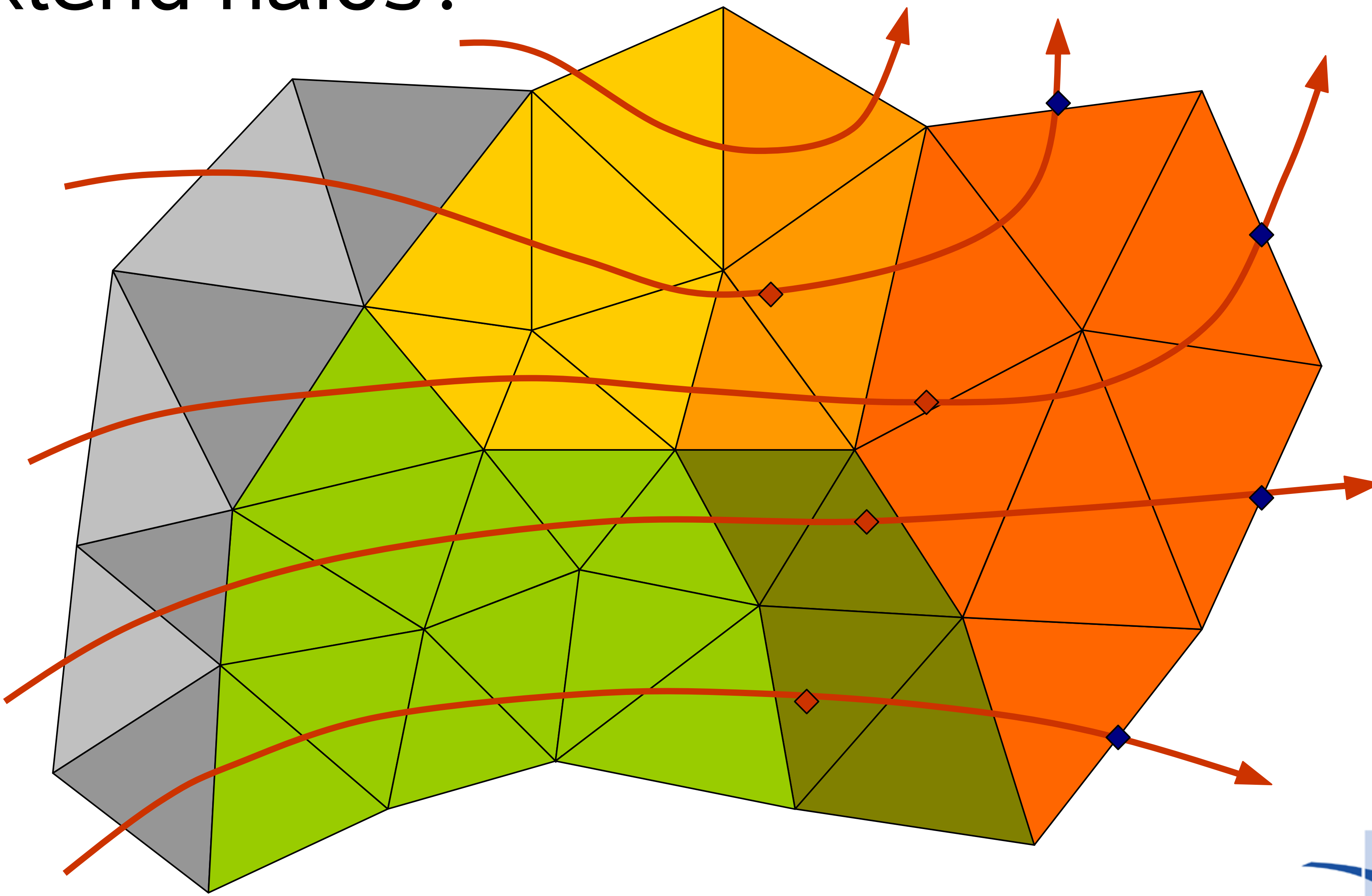
Tracking for Cr>>1



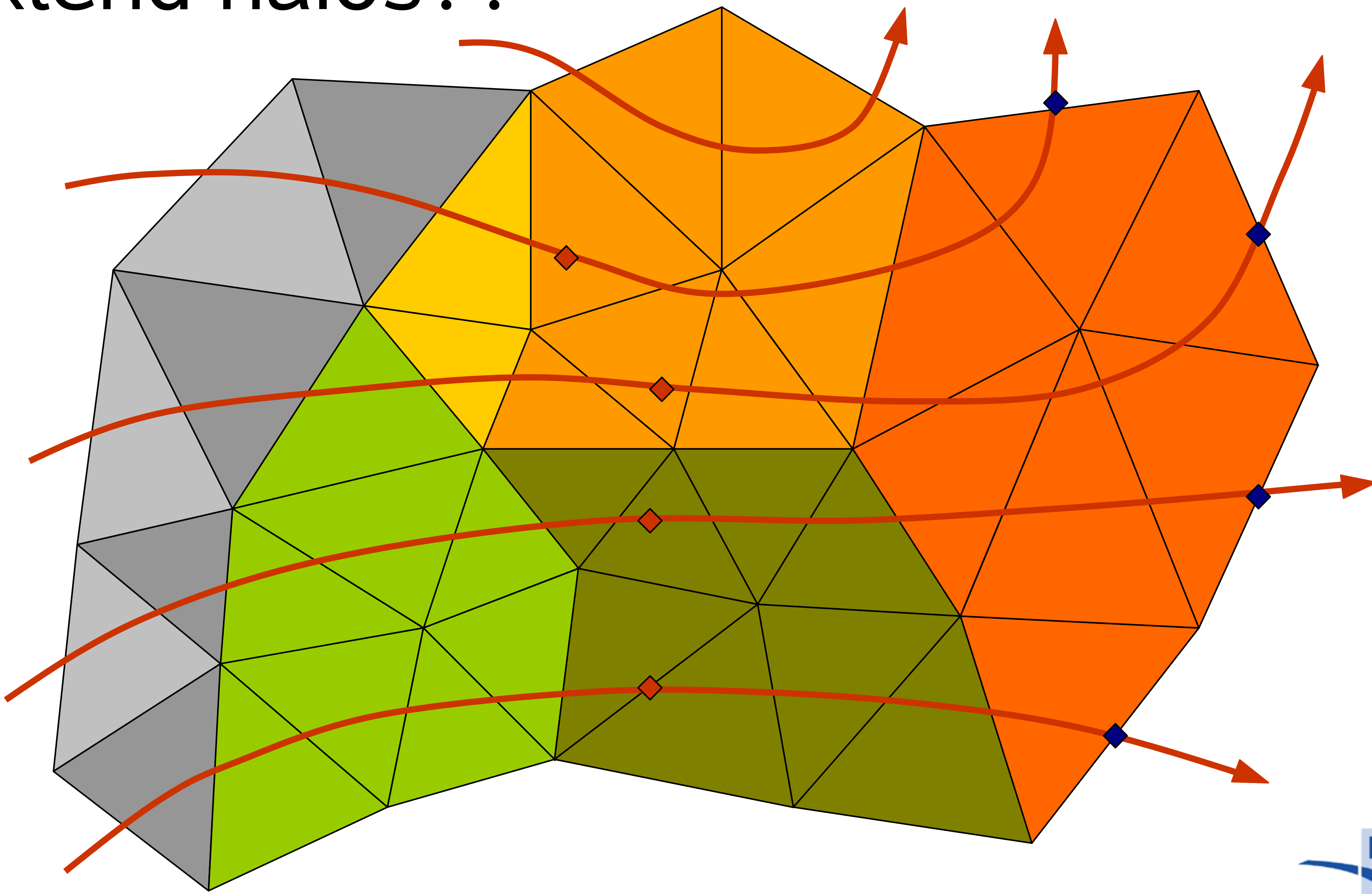
Tracking for Cr>>>1



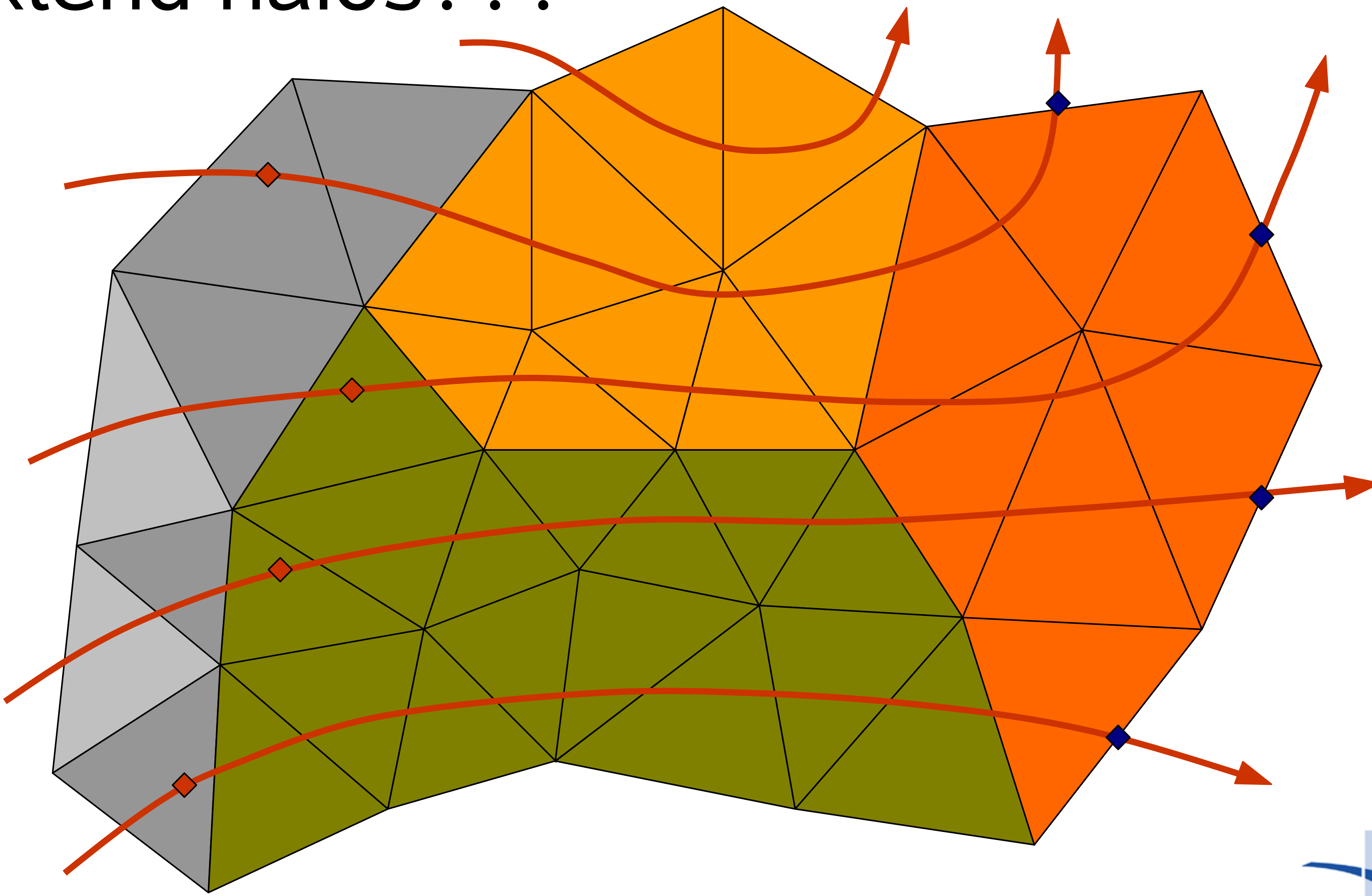
Extend halos?



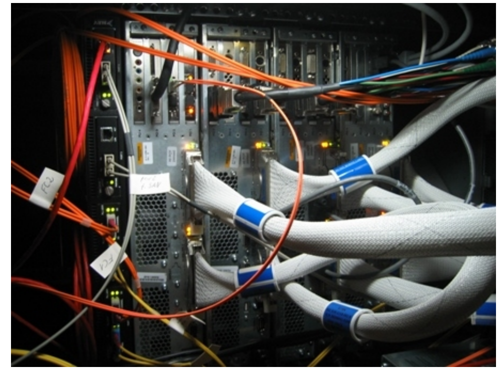
Extend halos??



Extend halos???



Tracking over partitions



- Streamline tracking is awkward in the point-to-point communication pattern between direct neighbours
- Inefficient for larger Courant numbers (large halos, further neighbours to communicate with…)
- Solution: Tracebacks leaving partitions treated as **separate objects** in an *autonomous* algorithm

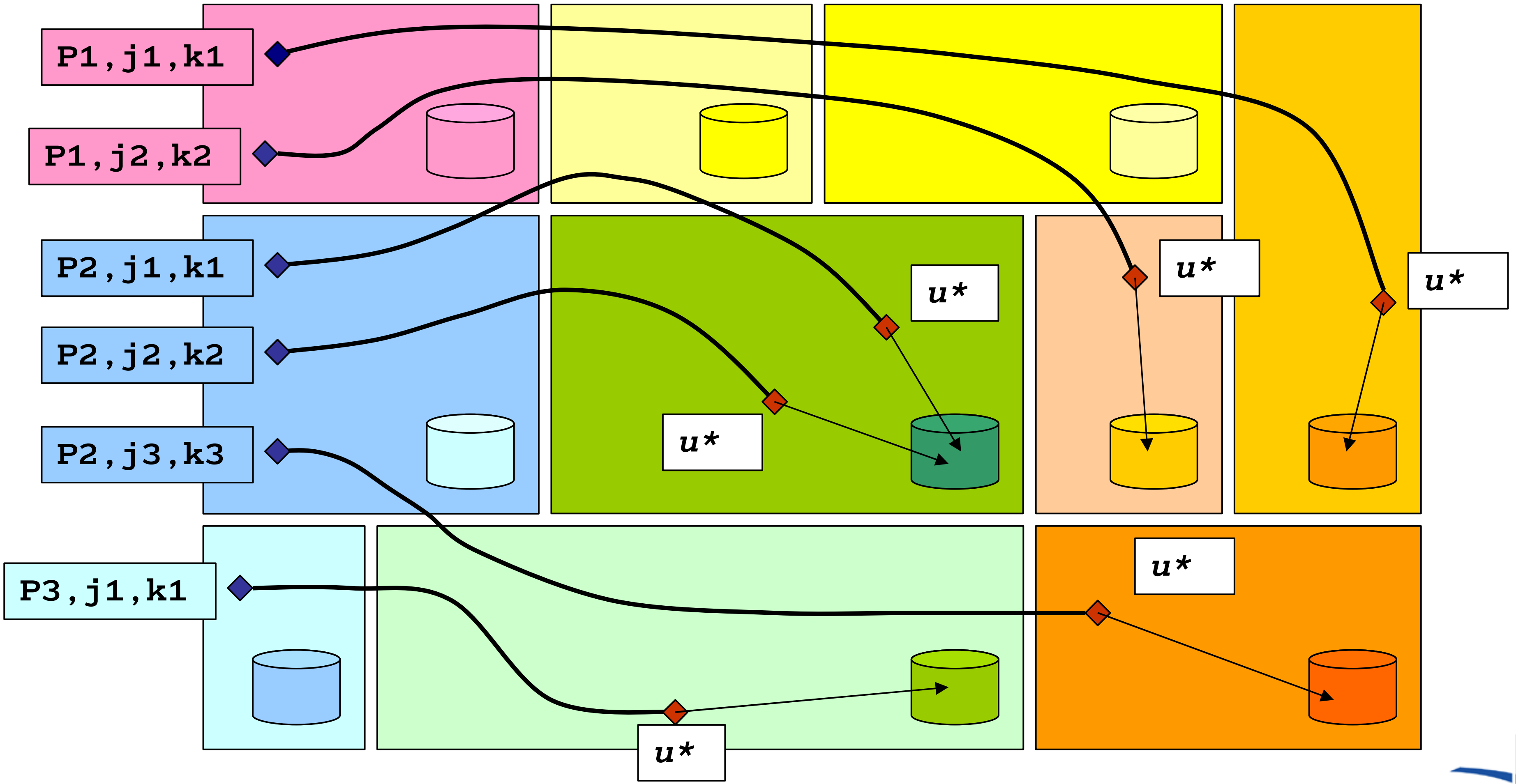
Source: dogtagshop.de

Dog tags for lost tracebacks

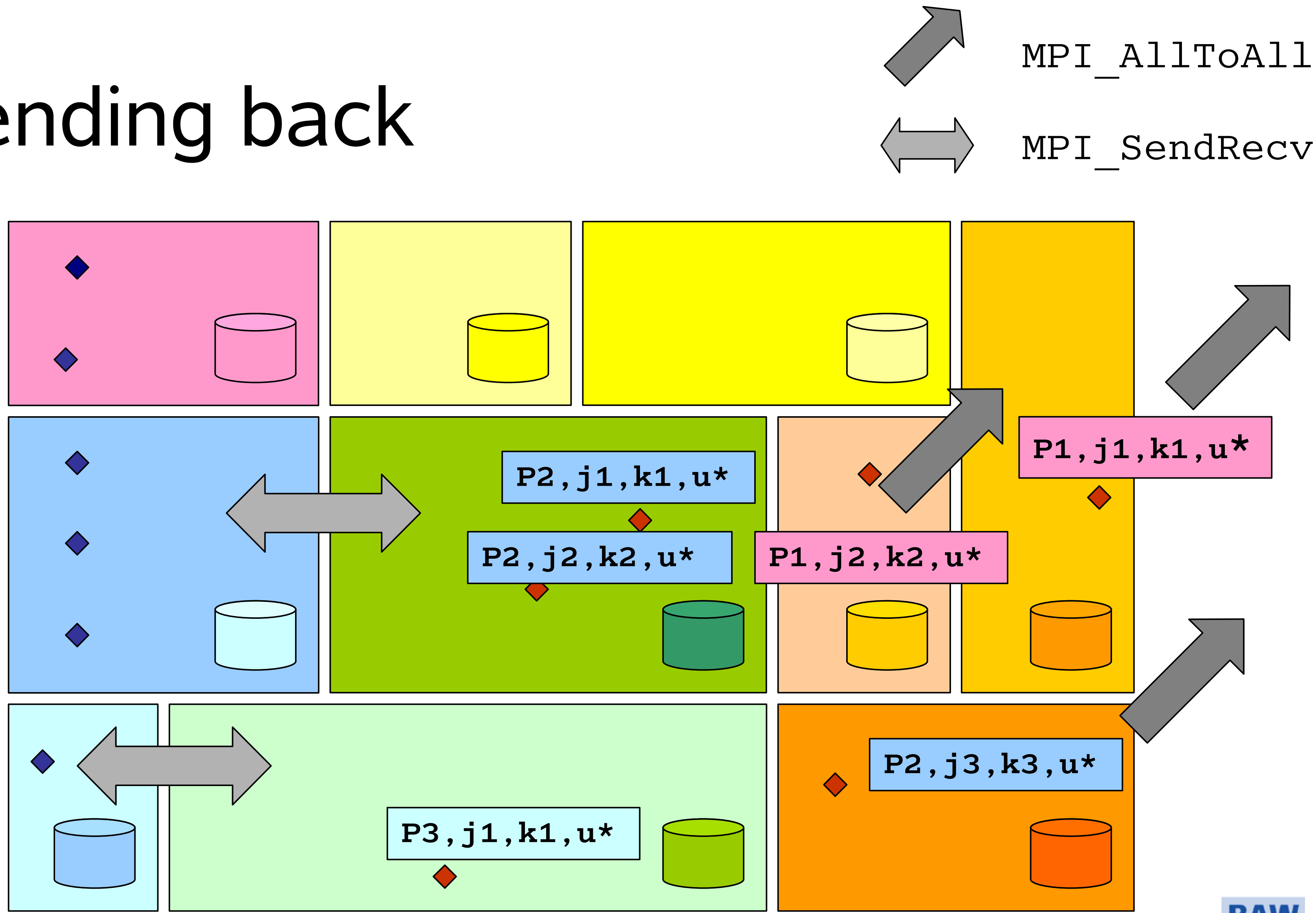
An object describing a 'lost' traceback:

```
TYPE charac_type
   INTEGER   :: mypid,ior,jor,kor
   INTEGER   :: nepid,i,k
   REAL(dp)  :: tres,xs(2),zs
   REAL(dp)  :: us(2),ws
   INTEGER   :: isat,mem
END TYPE
```

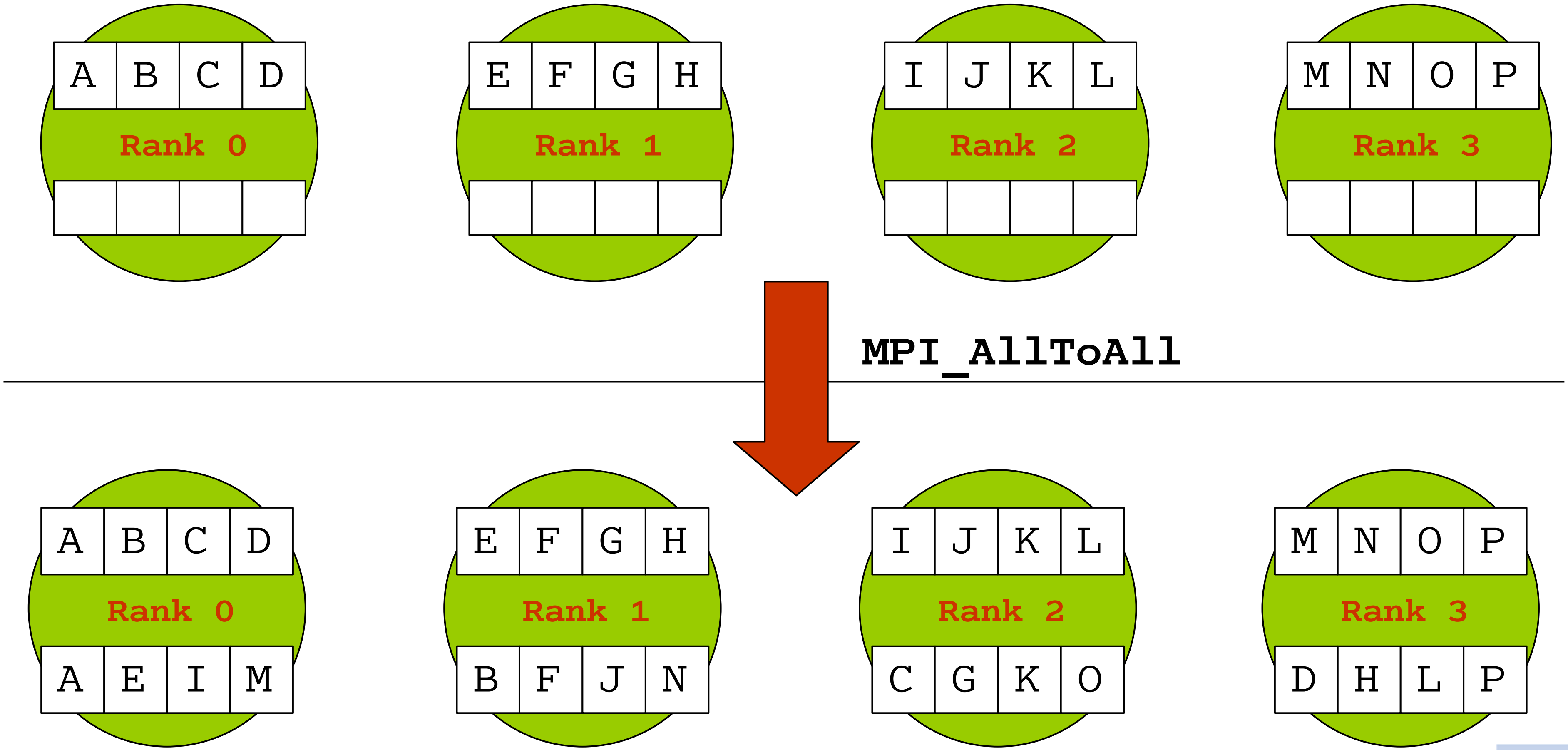
Autonomous tracking



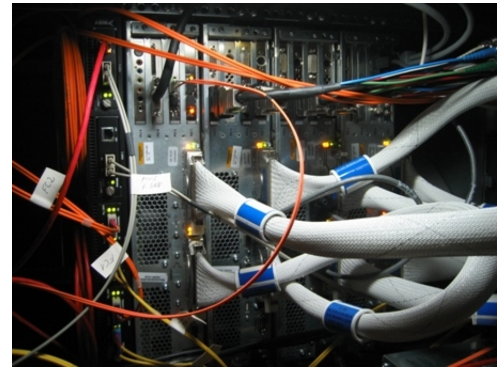
Sending back



MPI_AllToAll



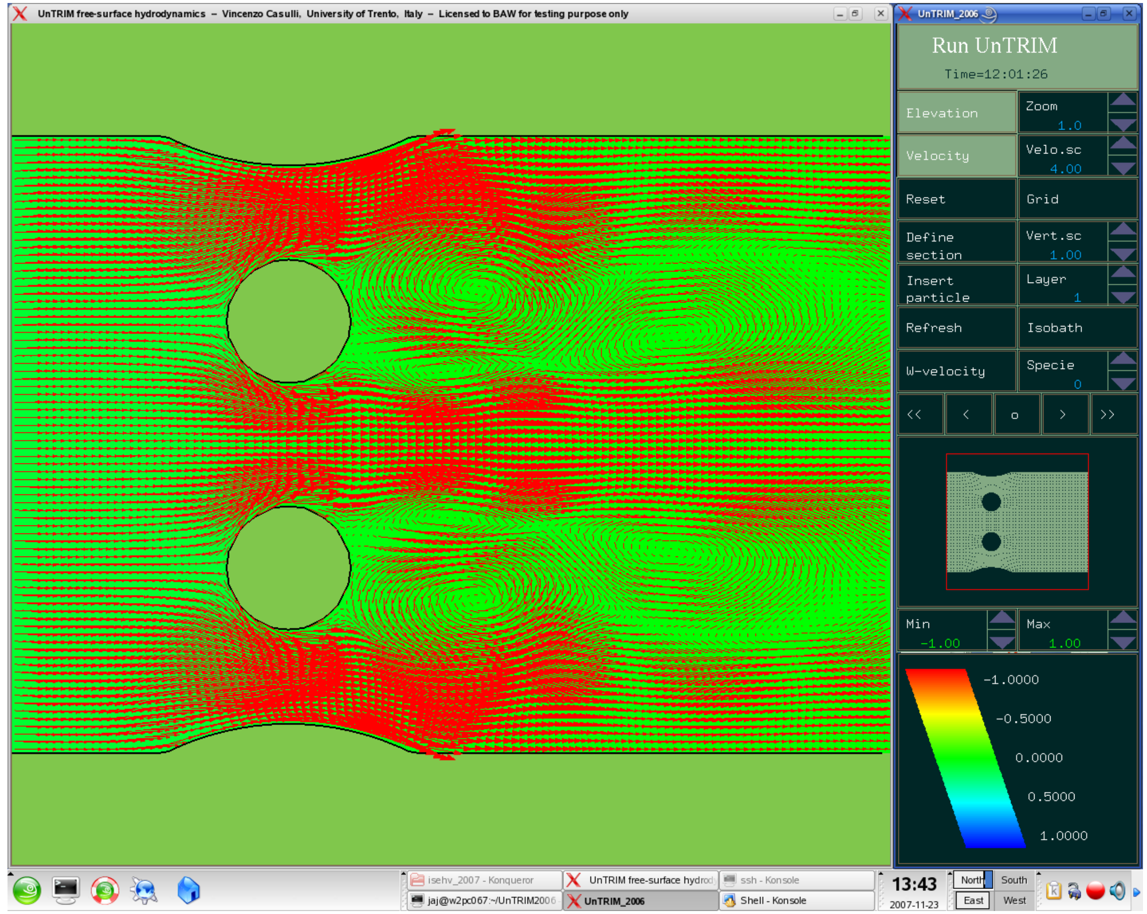
Summary: Communication



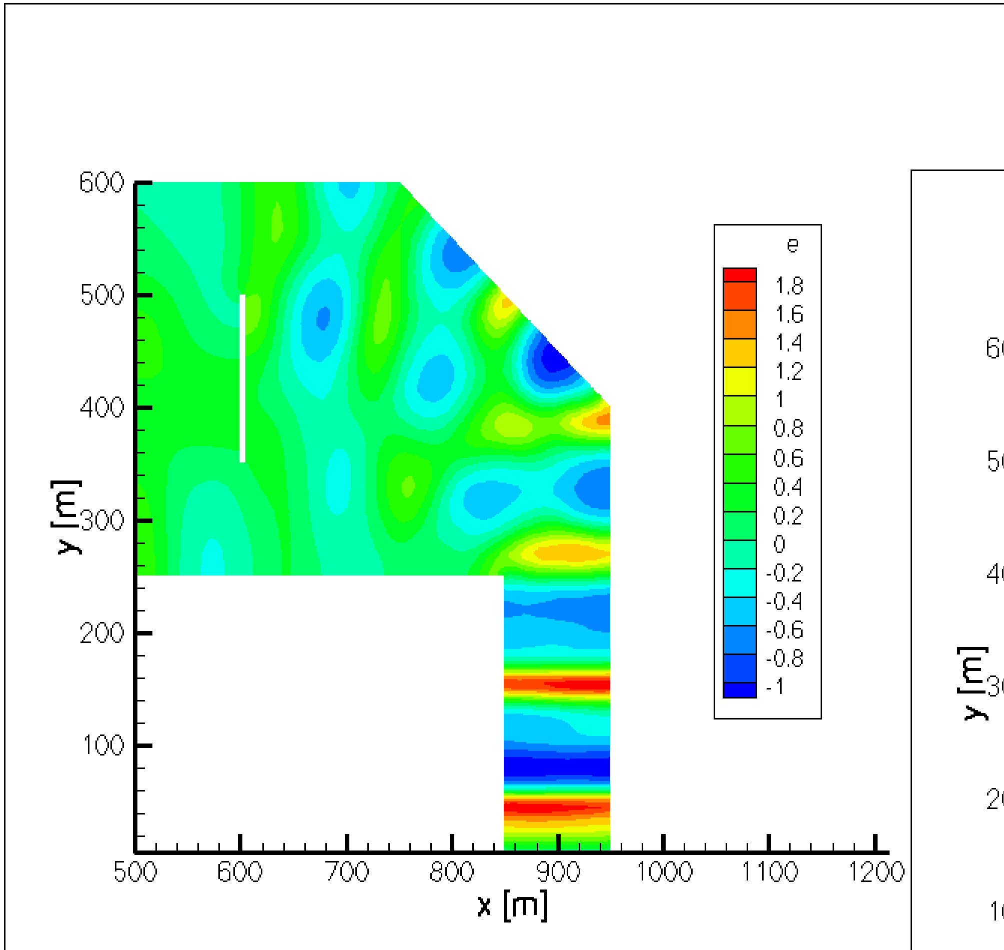
- FD/FV (Eulerian):
  - swapping halo values: point-to-point communication
- Advection (Lagrangian):
  - streamline tracking treating tracebacks as autonomous objects: global communication

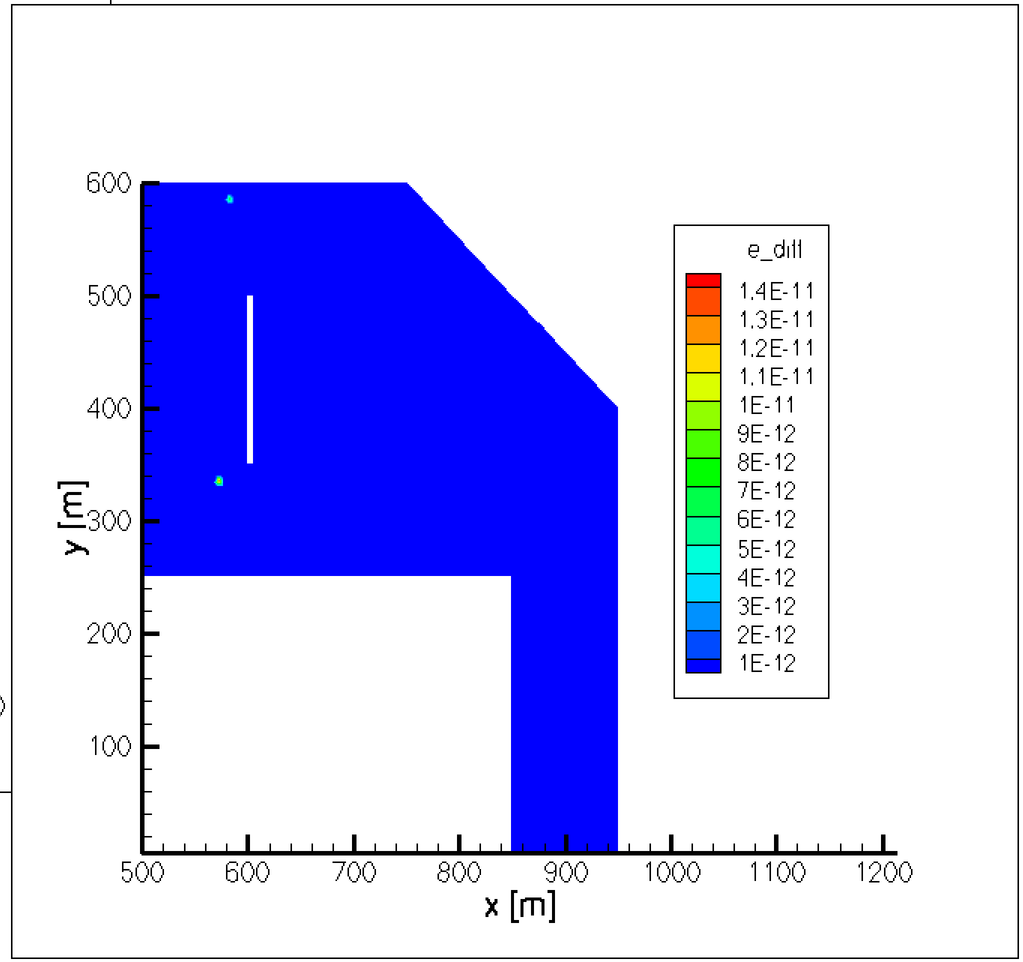
Verification

No differences between serial and parallel results?



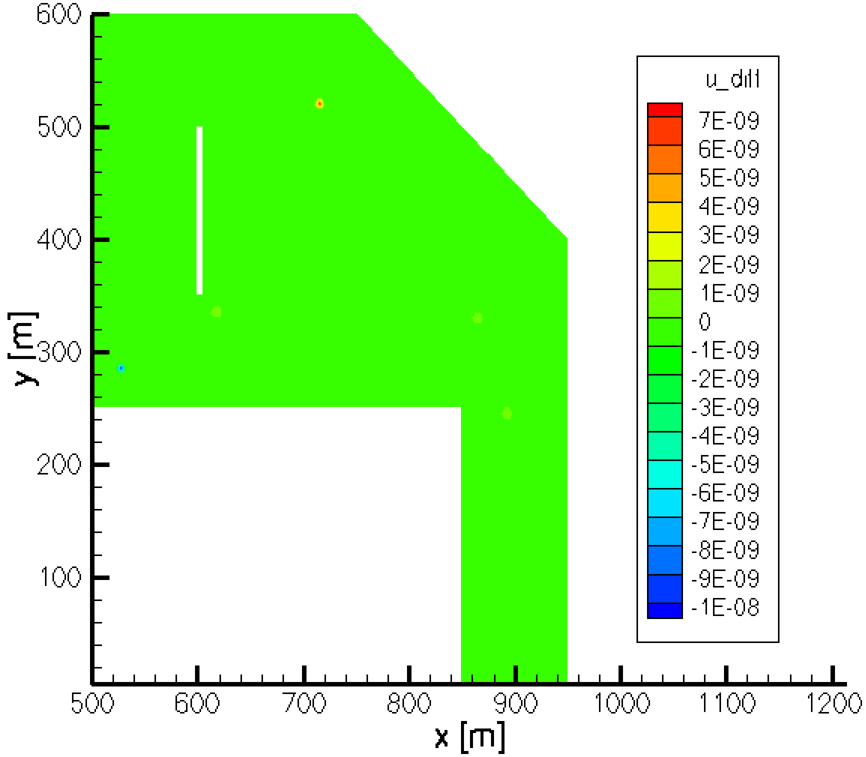
Harbour

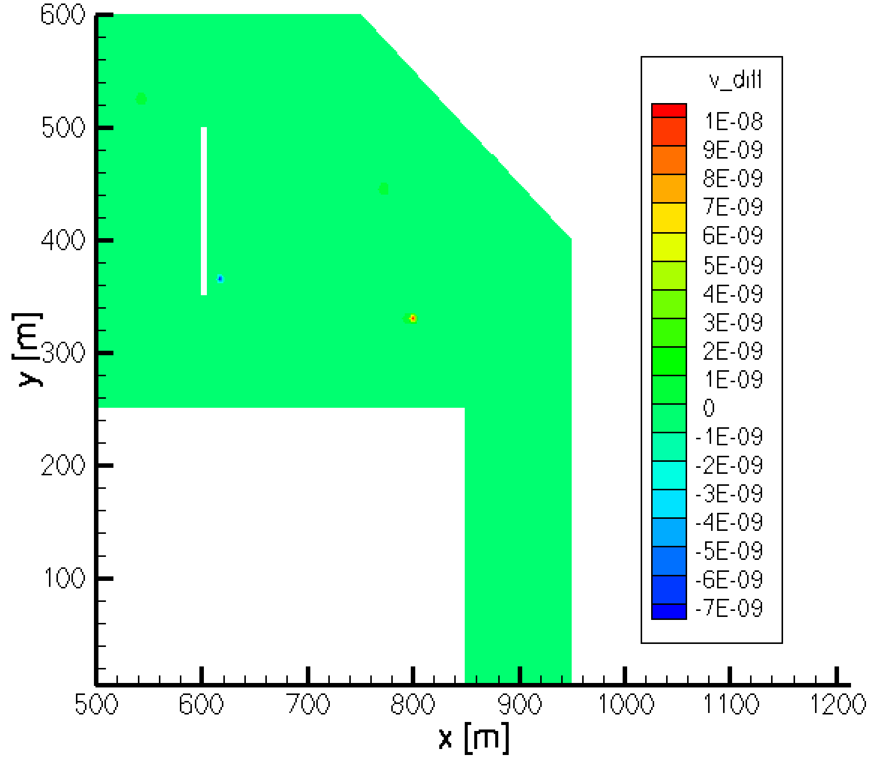




A regular mesh of triangles, flat bottom, short waves

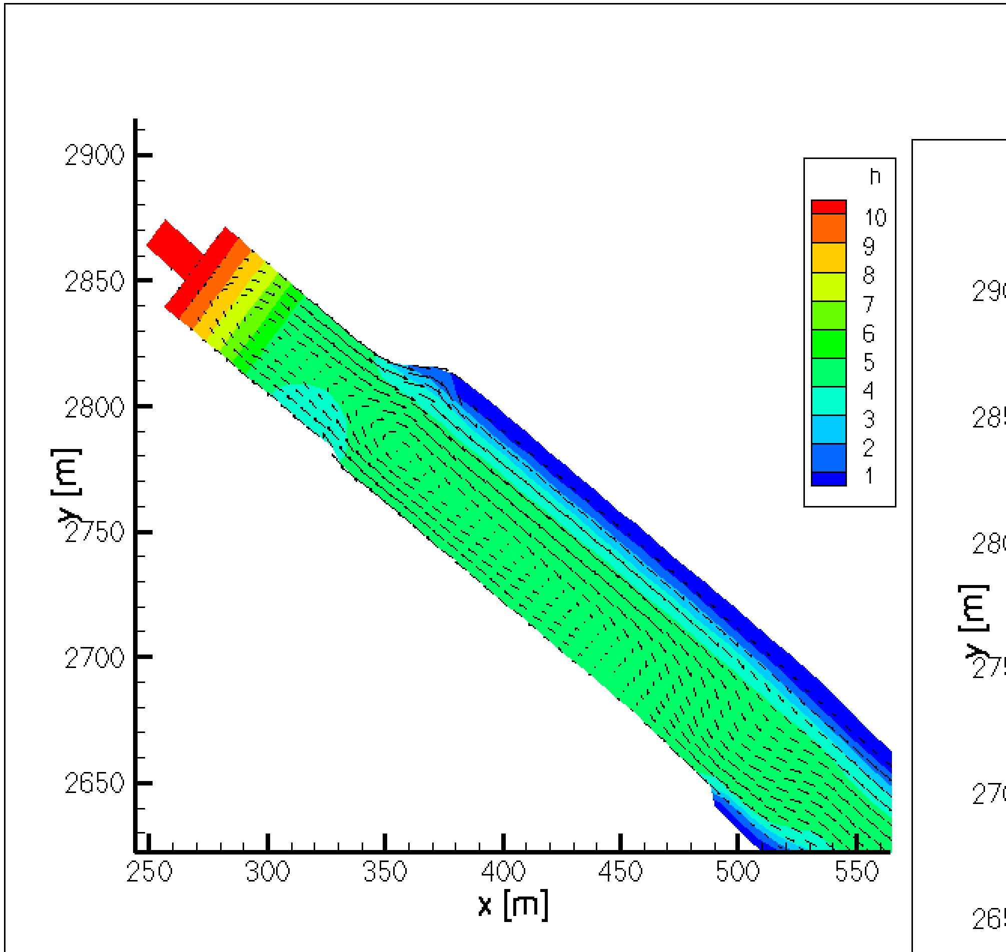
Harbour

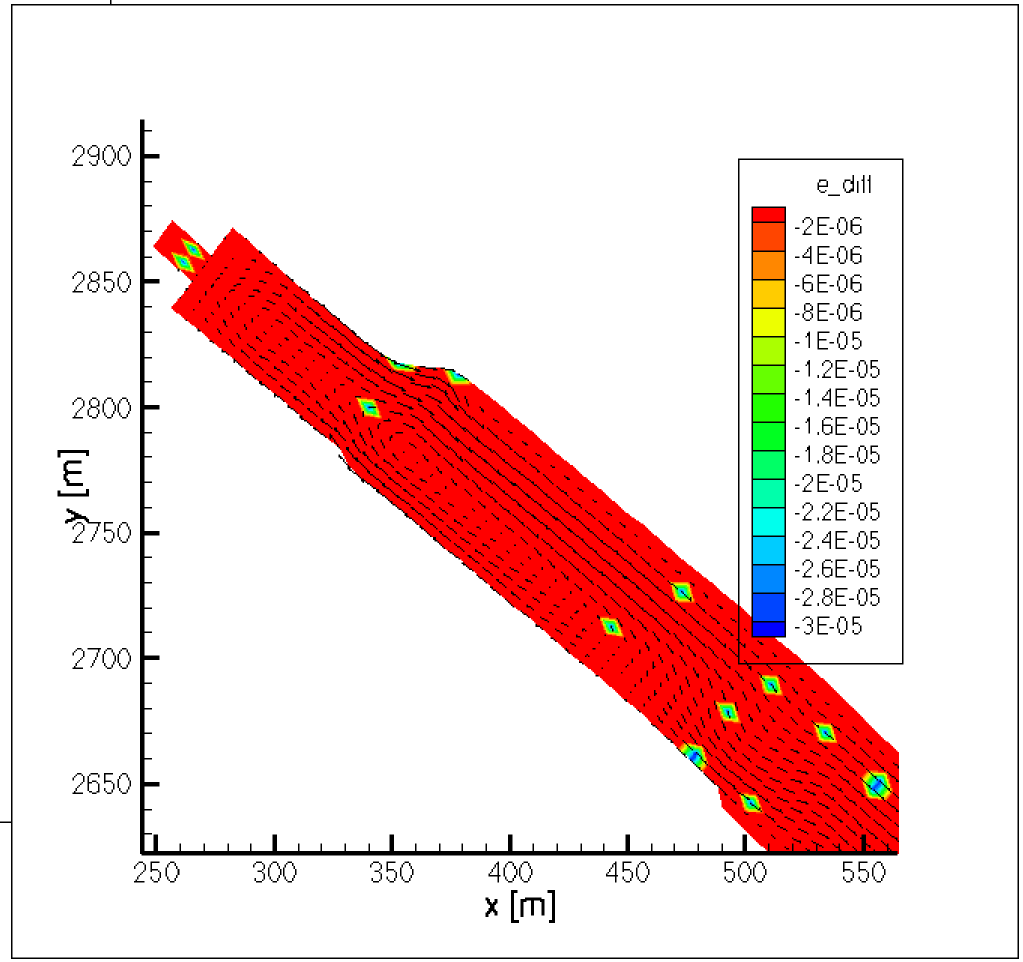




A completely non-hydrostatic test case

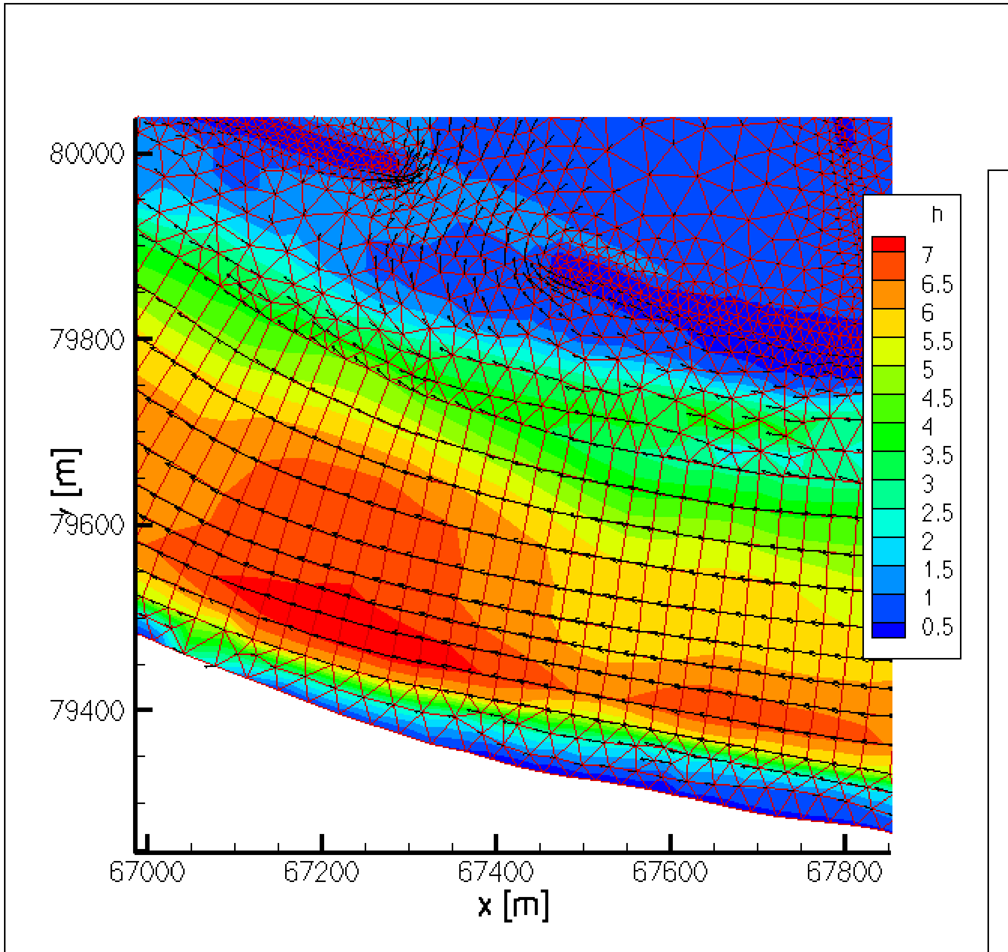
Lock waves

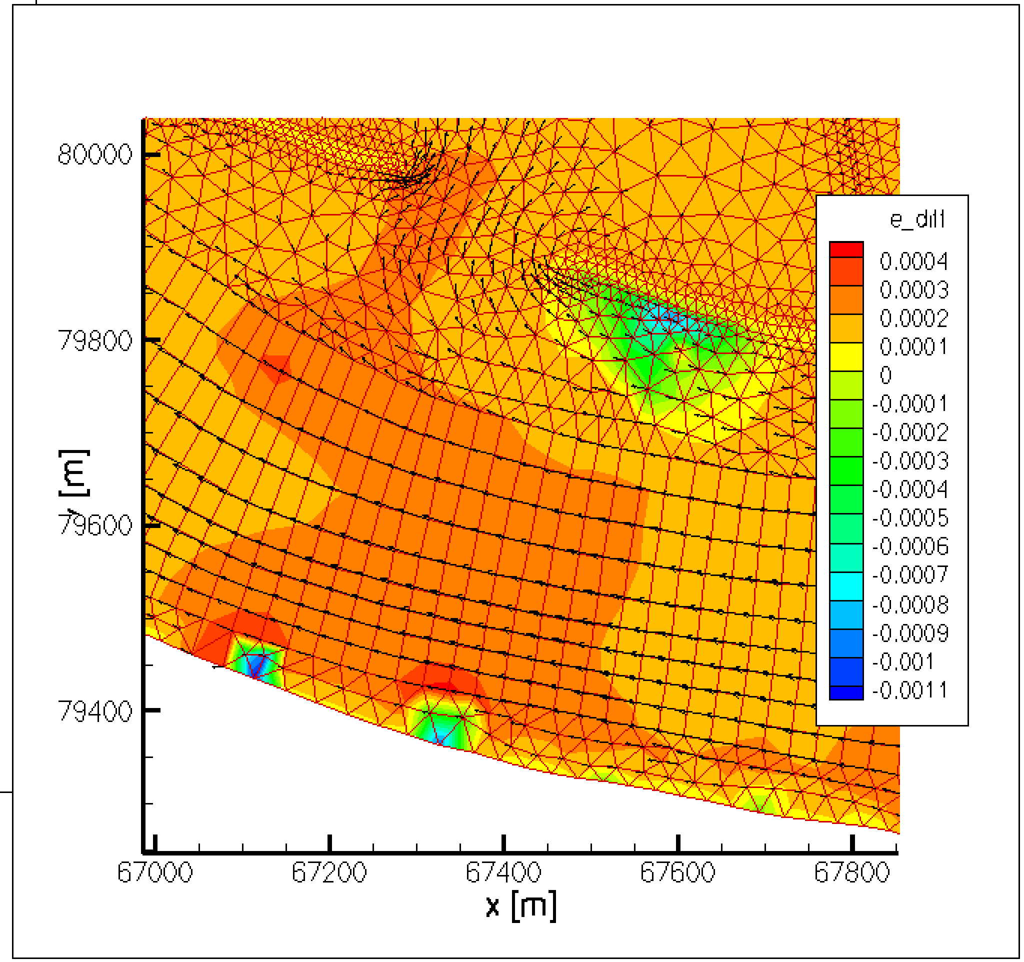




A very regular mesh, quadrangles

Lenzen

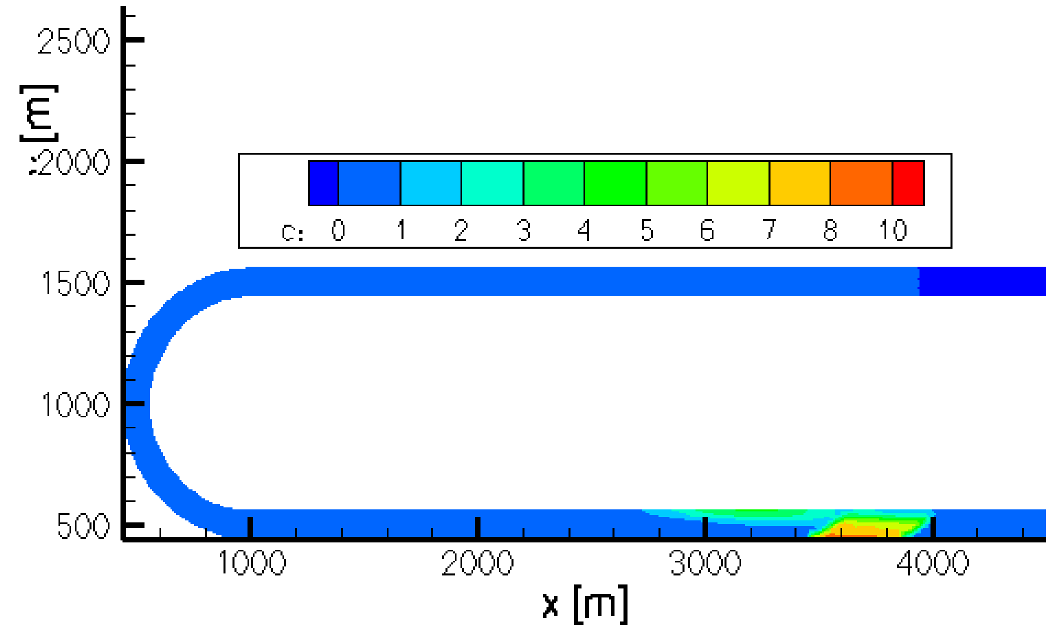


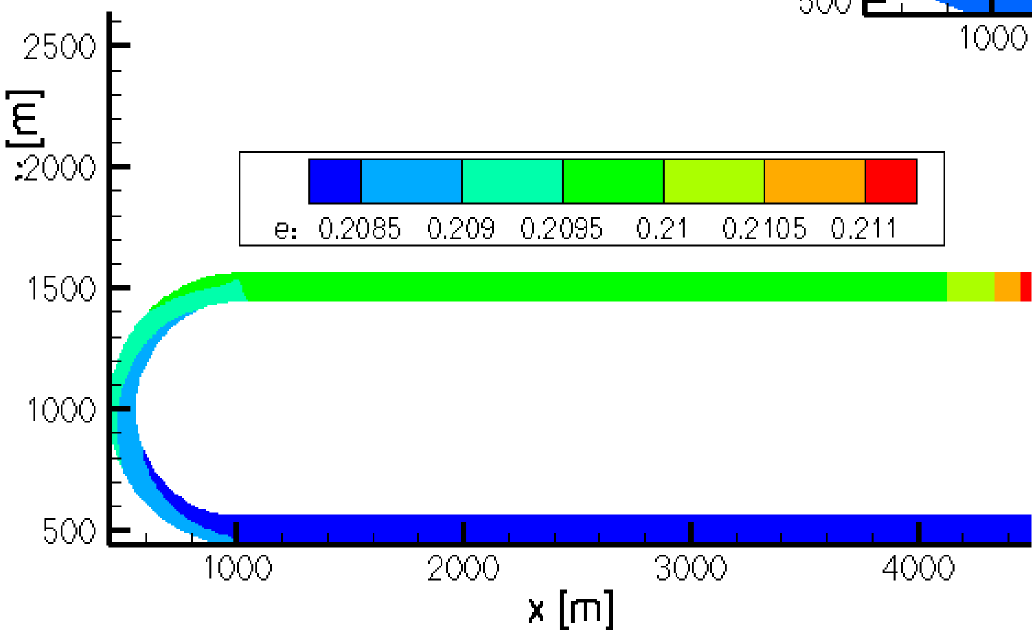


2D-case, dry/wet boundaries, steep bottom gradients

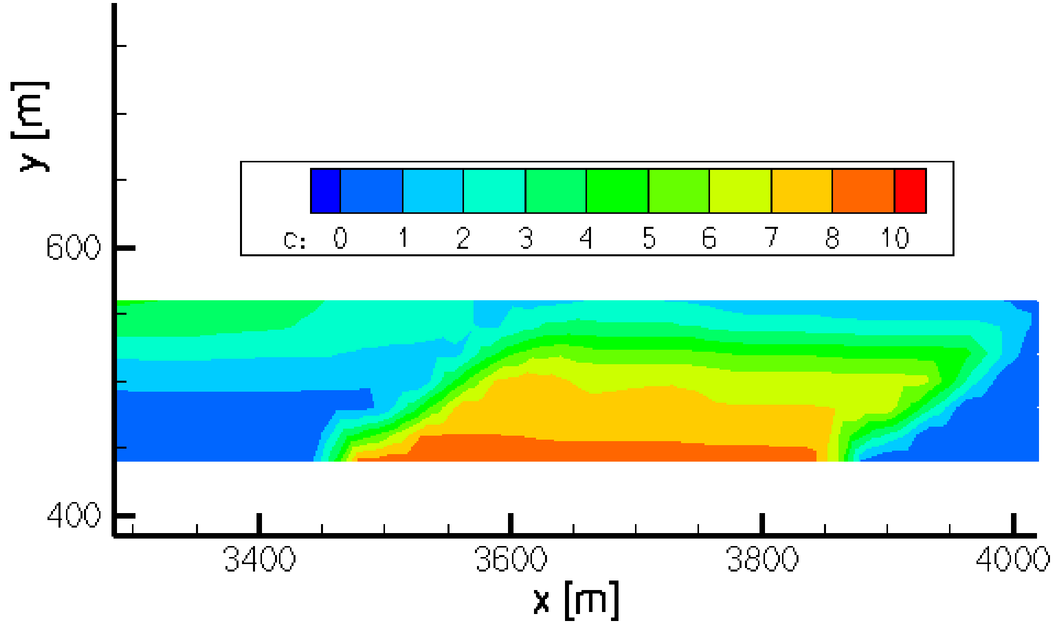
U-bend, 3D

A concentration signal of C=10 moves in the 3D U-bend channel current

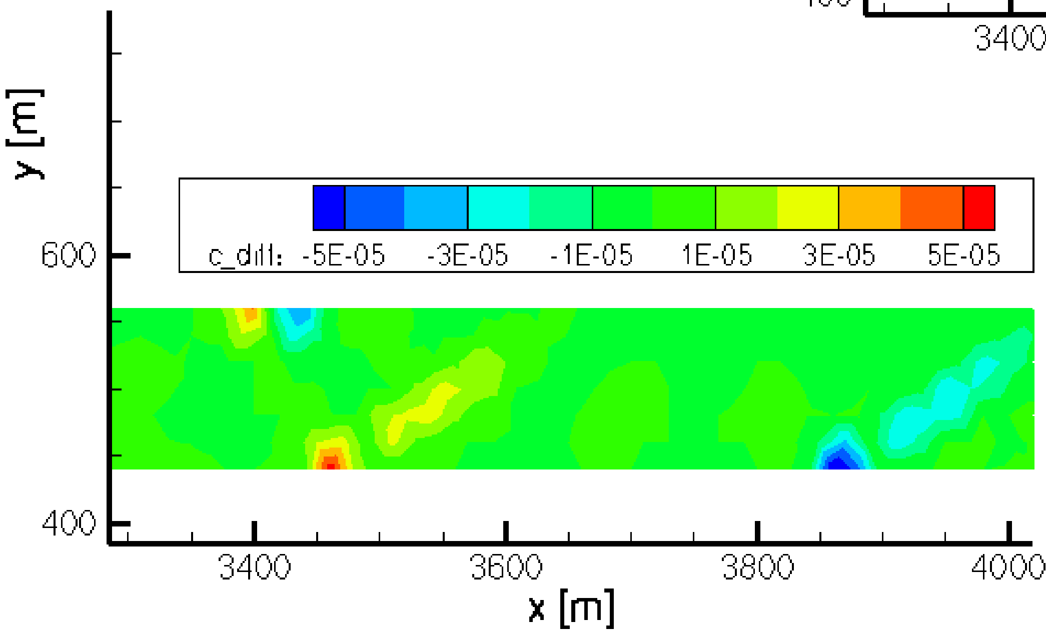




C_diff, 3D



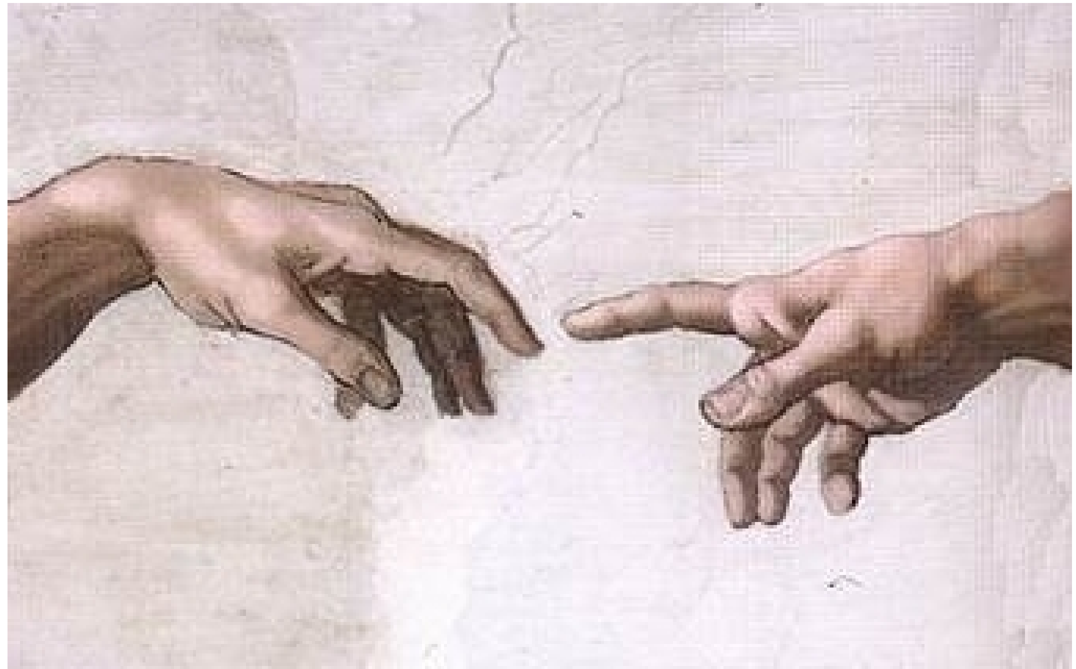
Note: this is the vertically integrated concentration profile



The concentration signal moves slightly too slowly in the parallel case (?)

Scalability

Speedups obtained with the MPI-parallel UnTRIM

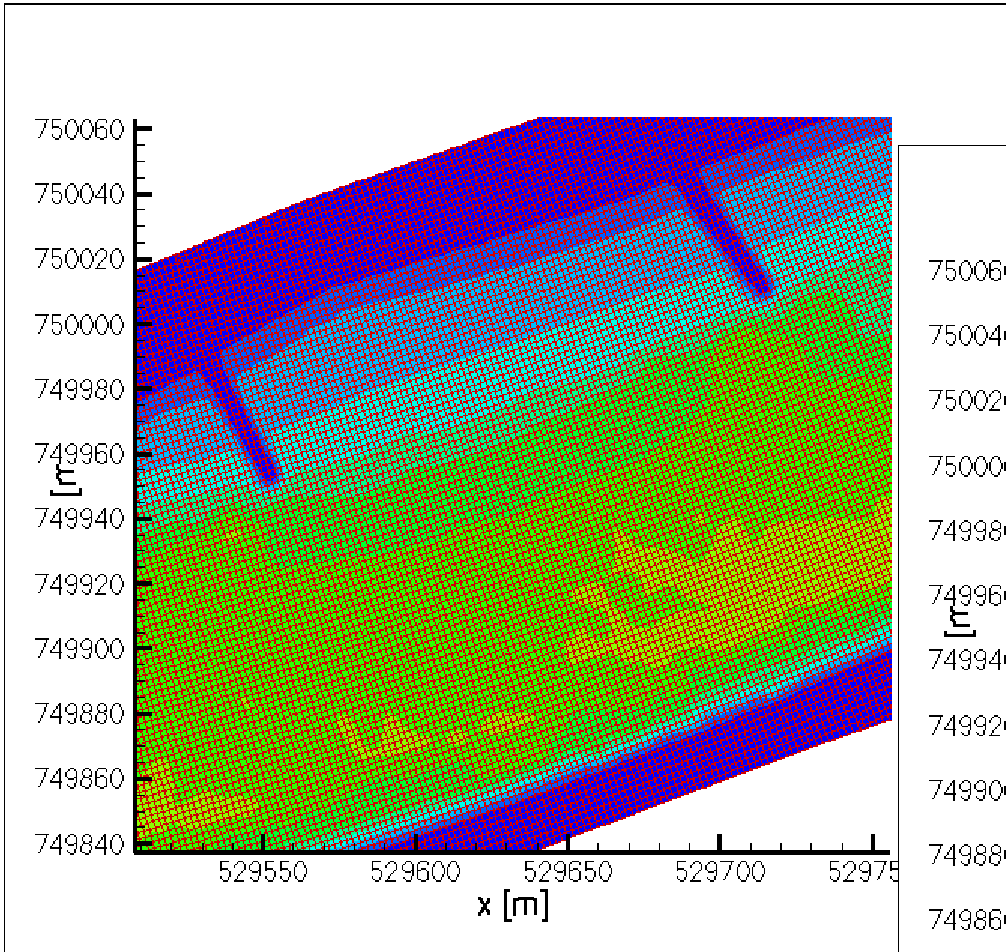


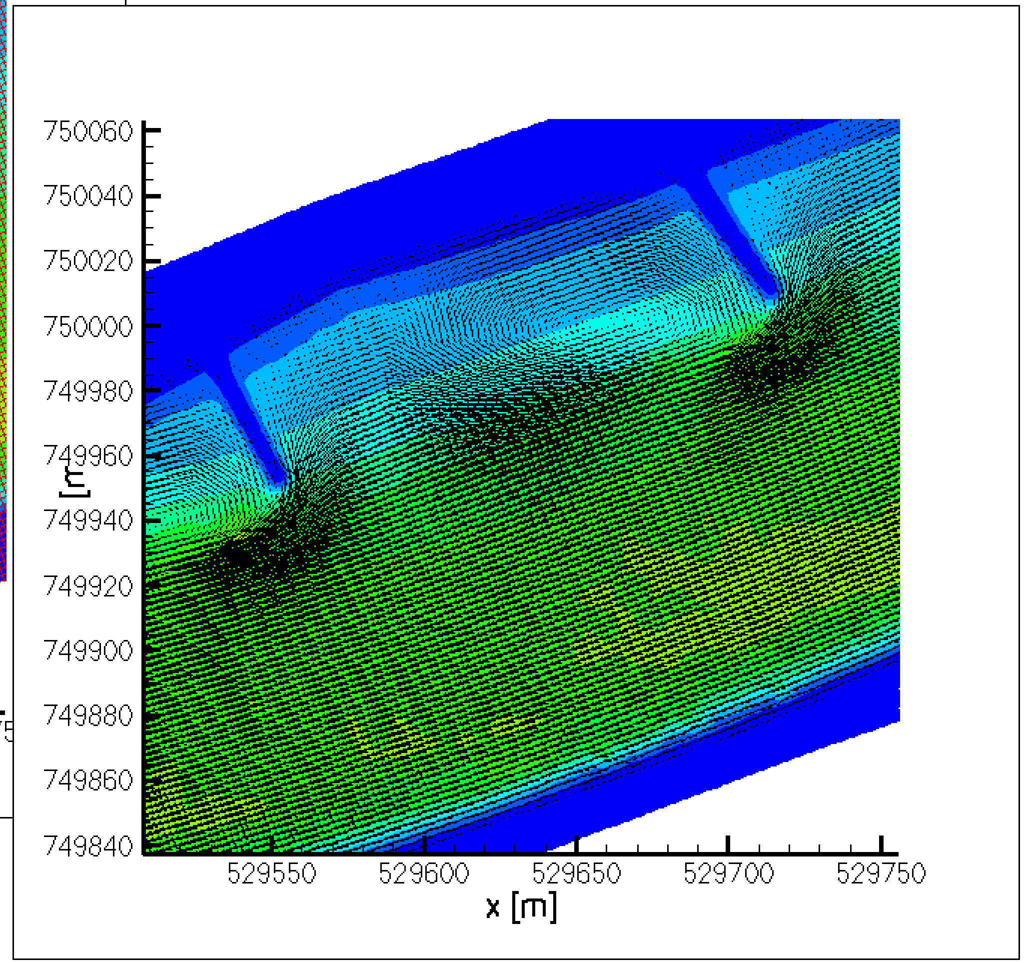
obelix.karlsruhe.baw.de

- **obelix+idefix** (4+1 cabinets)
- SGI Altix 3600
- **256+48** 1600 MHz Itanium-2 processors, 6MB cache
- 256+48GB **shared** memory
- 64-bit SuSE Linux 10.0
- CPU-Sets with PBS-Pro
- Intel and Gnu compilers
- OpenMP and MPI

- **A *state-of-the-art* parallel computer**

Coswig





14km river stretch - a mesh of very regular quadrangles with dx=2m, dz=0.5m, ne=738485, nk=22, n3e = 8006838

Computational effort

$\tilde{\eta}_i$ — Np linear equations, preconditioned conjugate gradient, **EPSI**

$\tilde{u}_{j,k}$ — Ns linear, tridiagonal systems of Nz equations, direct

$\tilde{w}_{i,k+1/2}$ — Np linear, tridiagonal systems of Nz equations, direct

or continuity, when *q=0*

3Dh

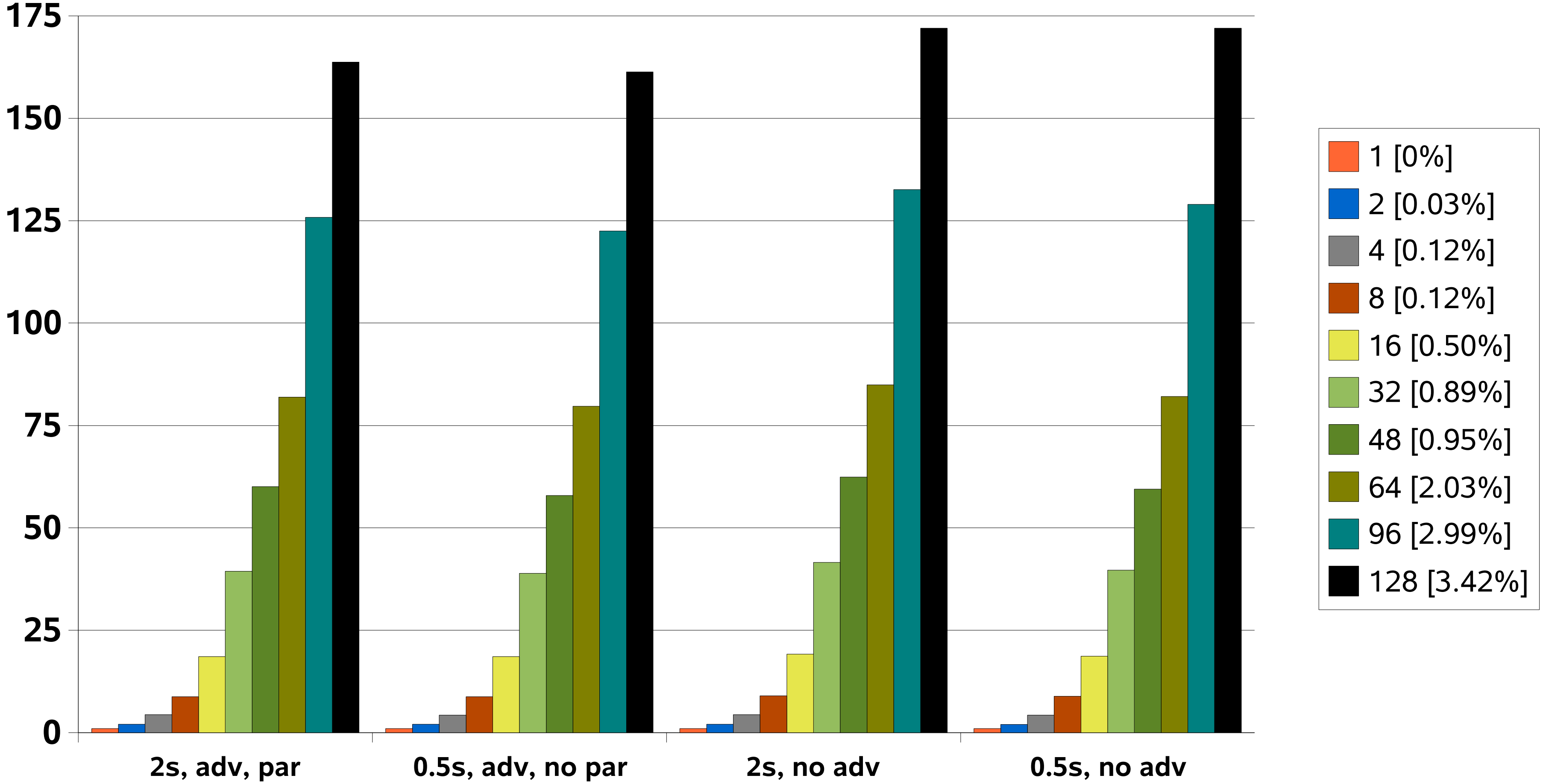
$q_{i,k}$ — Np*Nz linear equations, preconditioned conjugate gradient, **QEPSI**

η_i $u_{j,k}$ — Projection

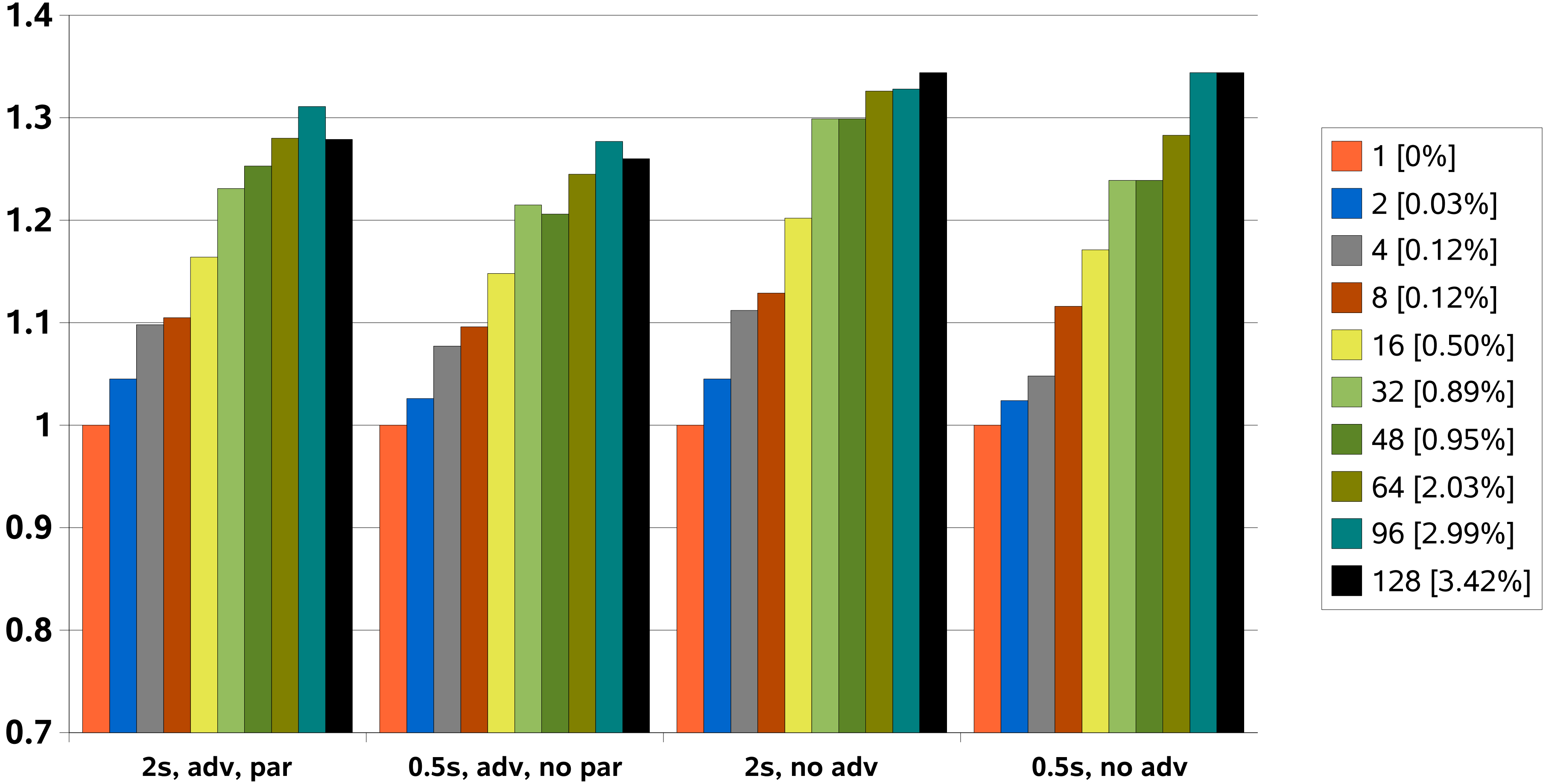
$w_{i,k+1/2}$ — Continuity

3Dnh

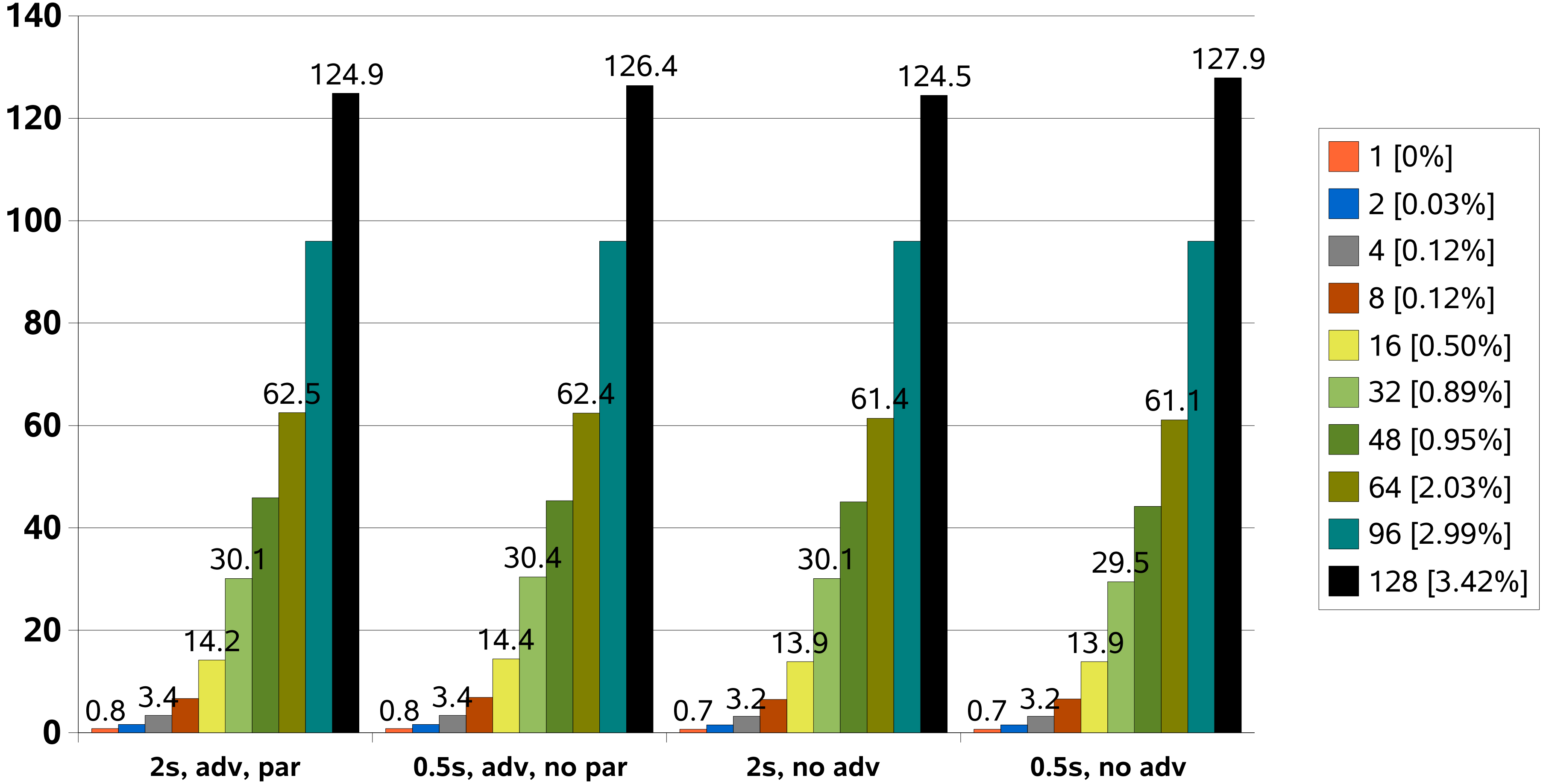
Coswig - speedup relative to 1p. (middle water, 2D, ne=738485)



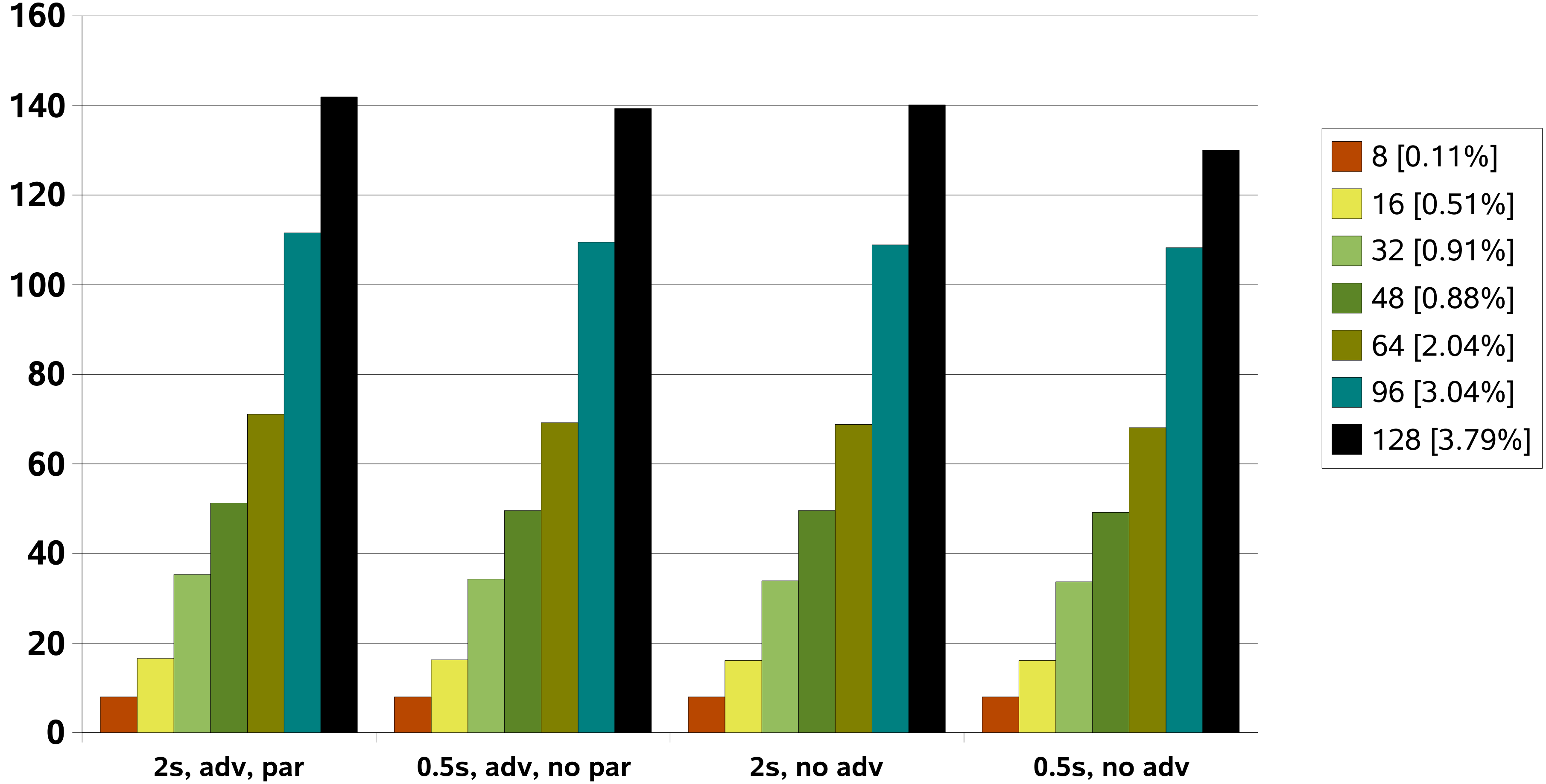
Coswig - efficiency relative to 1p. (middle water, 2D, ne=738485)



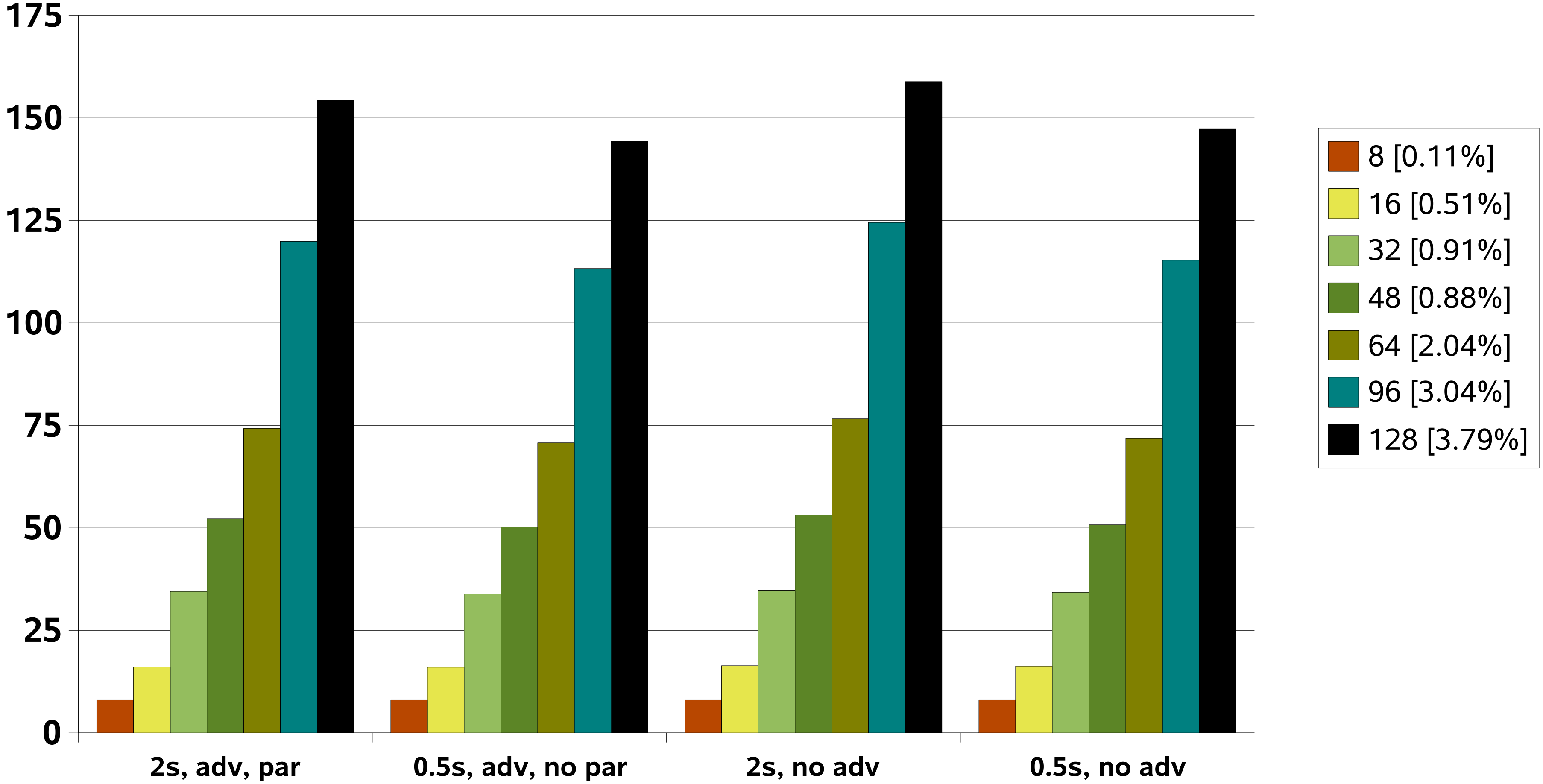
Coswig - speedup relative to 96p.
(middle water, 2D, ne=738485)



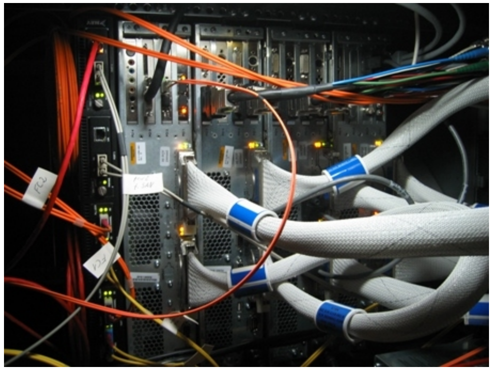
Coswig - speedup relative to 8p.
(middle water, 3D hyd, n3e=8006838)



Coswig - speedup relative to 8p.
(middle water, 3D, non-hyd, n3e=8006838)



Reached



- A parallel UnTRIM implementation without compromising the properties of the serial code

- A good scalability due to:
  - communication adequately designed for the significant parts of the algorithm
  - minimal amount of data exchanged between processors

Further developments (in the original code)

UnTRIM2007 vel **UnTRIM²**

"artificial porosity" iterative wetting/drying

higher-order interpolation in the advection scheme

improved data locality (avoiding cache misses)

OpenMP-parallelisation

Outlook

We have an efficient, robust and accurate scheme with:

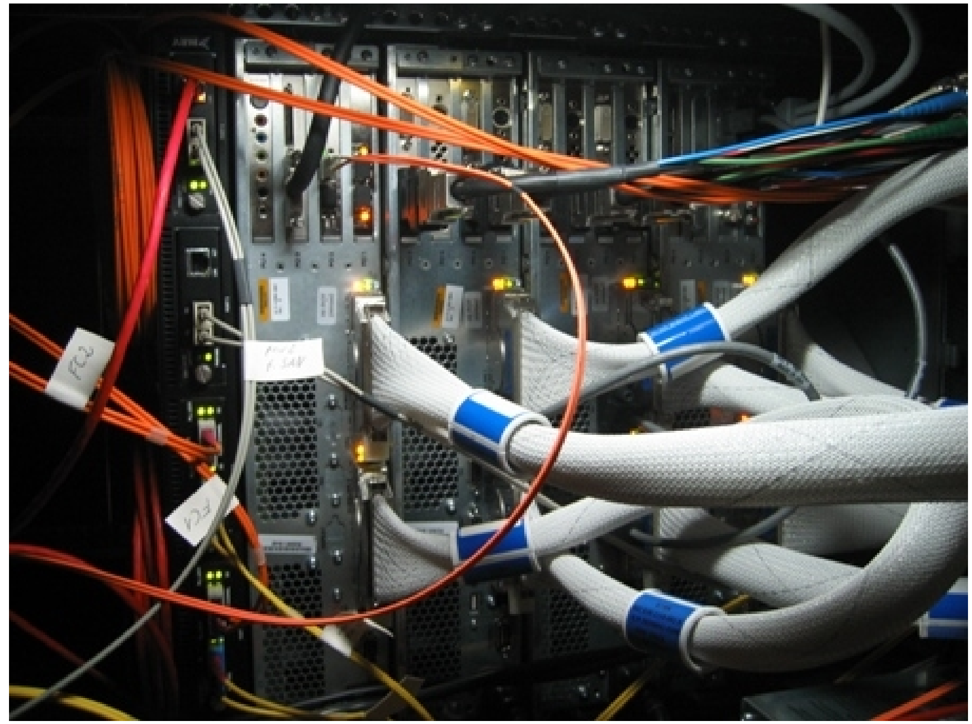
- second order discretisation error in space and time
- unconditionally stable
- very good scalability

Next step: high resolution modelling.

I listen to all questions!

Additional transparencies

Discussions?



Principles

A simple, general purpose code with known properties

Robust, accurate and efficient numerical methods

Clearly defined application domain

Flexibility in processing stages

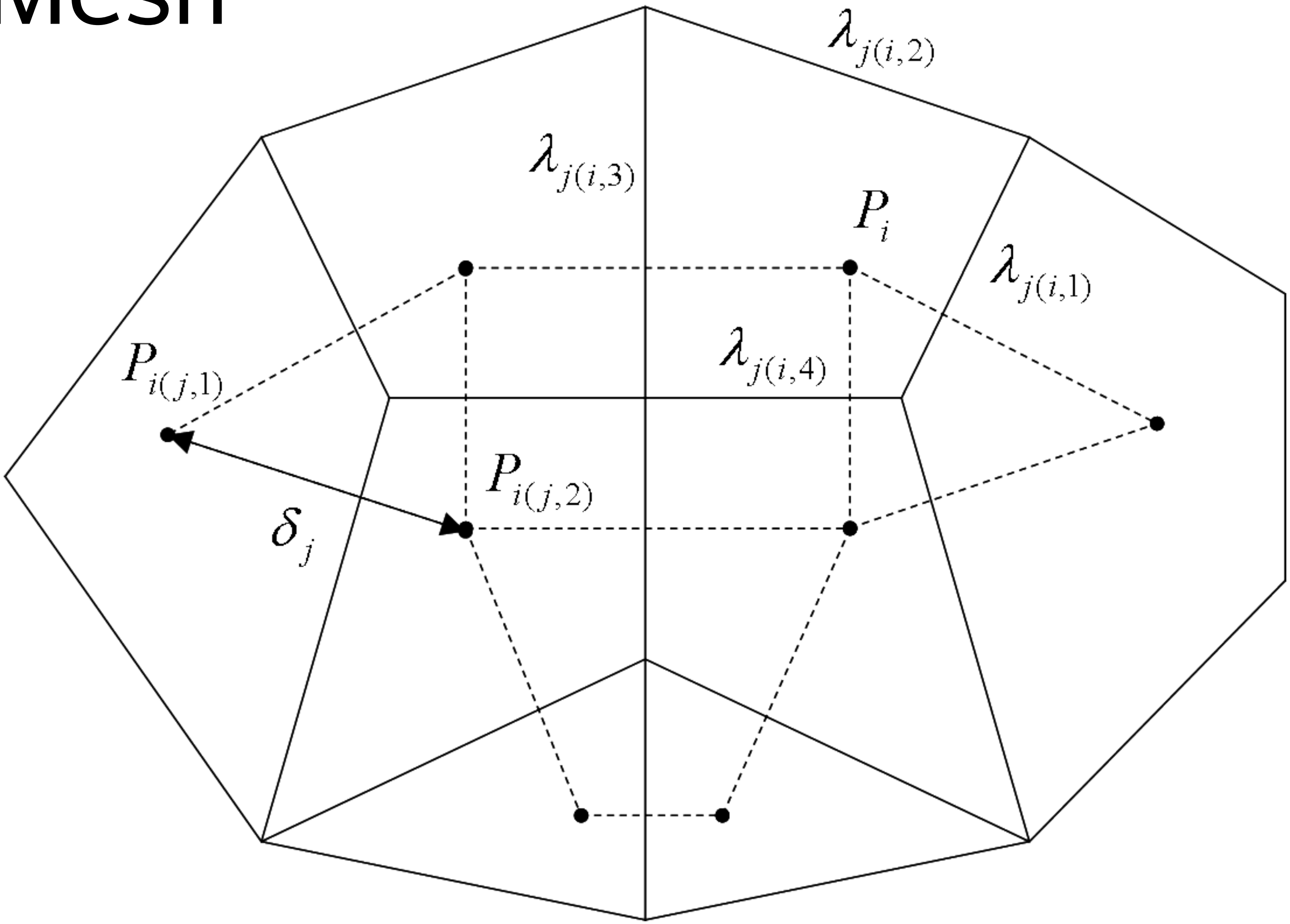
Responsibility, communication, co-operation

Pursue the code evolution

An unstructured, orthogonal mesh

A grid is said to be an *unstructured orthogonal grid* if within each polygon a point (hereafter called *center*) can be identified in such a way that the segment joining the centers of two adjacent polygons and the side shared by the two polygons have a non-empty intersection and are orthogonal to each other.

Mesh



$\lambda_{j(i,2)}$
$\lambda_{j(i,3)}$
P_i
$\lambda_{j(i,1)}$
$\lambda_{j(i,4)}$
$P_{i(j,1)}$
$P_{i(j,2)}$
δ_j

Application domain

...three dimensional equations describing free surface flows...

The application domain are:
three-dimensional,
non-hydrostatic,
environmental
free surface flows
including species transport

[...small seas, lakes, estuaries, rivers, creeks...]

Wave equation

Introduce the semi-implicitly discretised momentum equations into the continuity equation

Use the algebraic form – drying and wetting included

Solve the resulting (2D) wave equation

Obtain the fractional result for the velocity

[...this is this "Casulli formulation"...]

Properties summary

- Discretisation error in space:
  - 2nd order for regular meshes
  - diminishing downto 1st order for irregular ones
- Discretisation error in time:
  - 2nd order – semi-implicit
  - implicit: 1st order
- Mild stability condition due to horizontal viscosity
  - treated iteratively
- Unconditionally stable with respect to:
  - gravity waves speed,
  - bottom and free surface friction,
  - vertical viscosity

Numerical code features

- Fortran95
  - intensive use of matrix features
  - dynamic memory allocation – but only once
  - modular (core, get/set library, user interface/software)

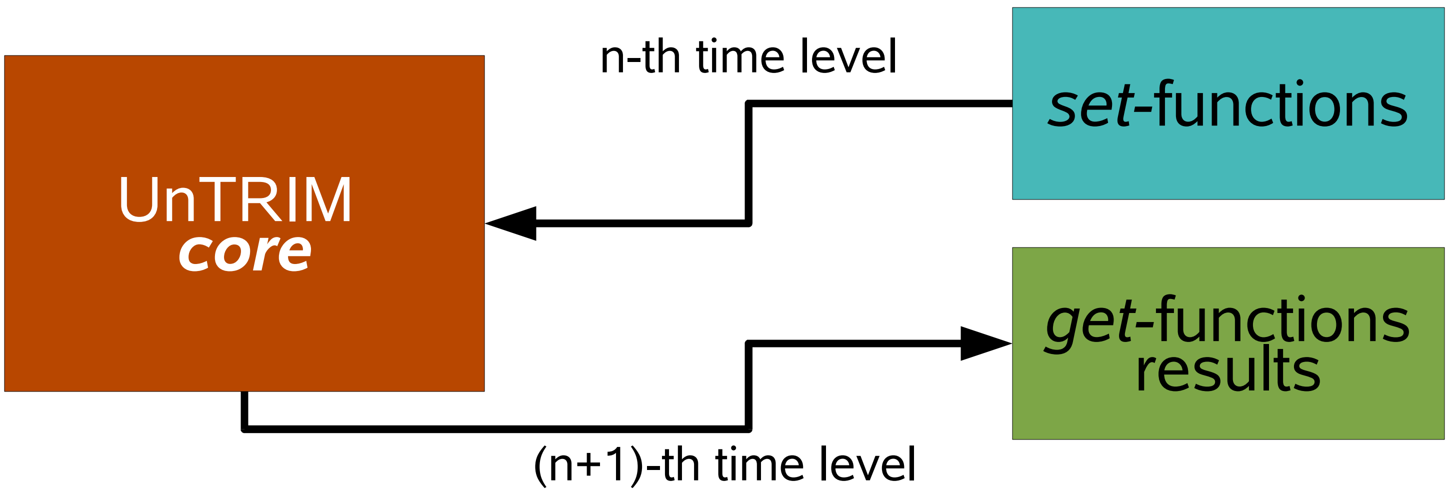
- Model core *("engine")* + User Interface
  - A library of *get*- and *set*-functions

- User supplies
  - all physical sub-models, like turbulence closure
  - all forcing functions
  - all initial and boundary conditions
  - sources/sinks, etc.

Numerical code features

Maximum Efficiency

Maximum Flexibility



Developed, maintained and quality-controlled by V.Casulli

Allows the user a creative approach to numerical modelling

Similar code examples

ELCIRC

[2004, *Zhang, Baptista, Myers,* Oregon Univ.]

DELFIN

[2005, *Ham, Pietrzak, Stelling,* TU Delft]

FINEL

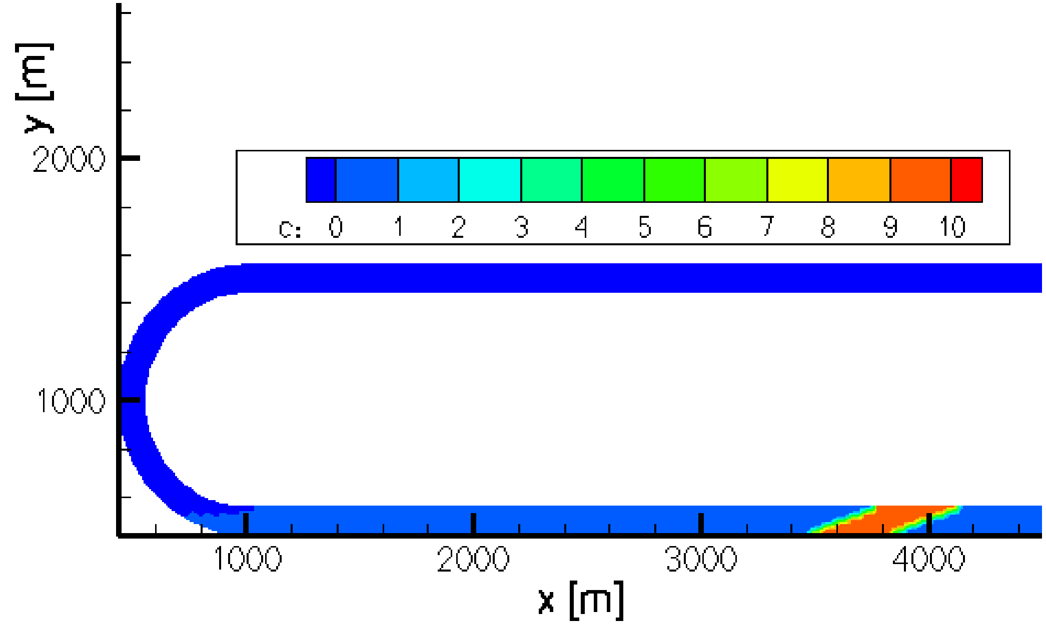
[2005, *Pietrzak, Labeur,* TU Delft]

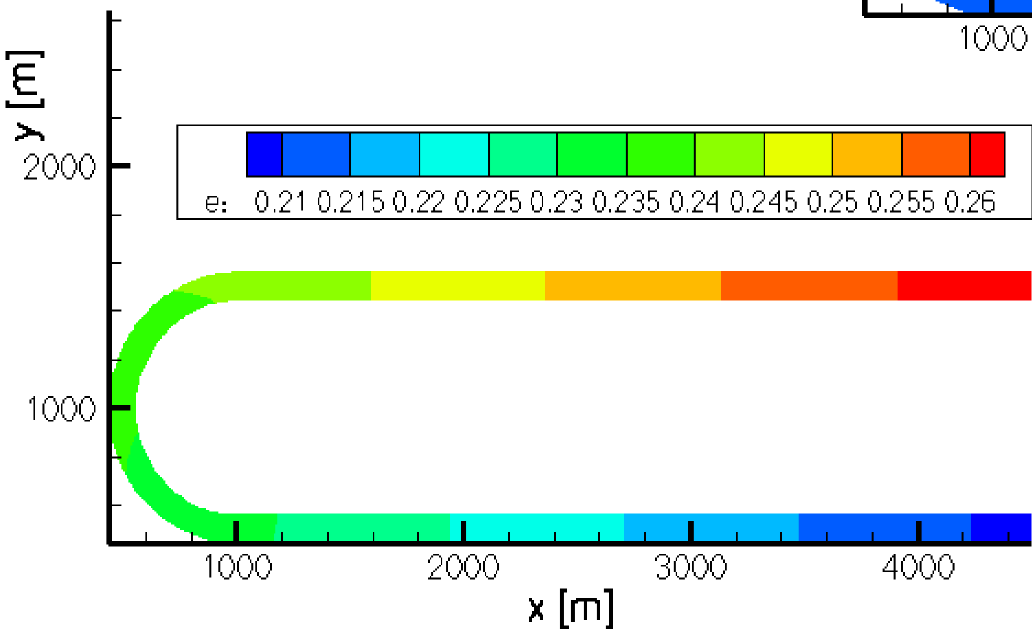
SUNTANS

[2006, *Fringer, Gerritsen, Street,* Stanford Univ.]

U-bend, 2D

A concentration signal of C=10 moves in the 2D U-bend channel current

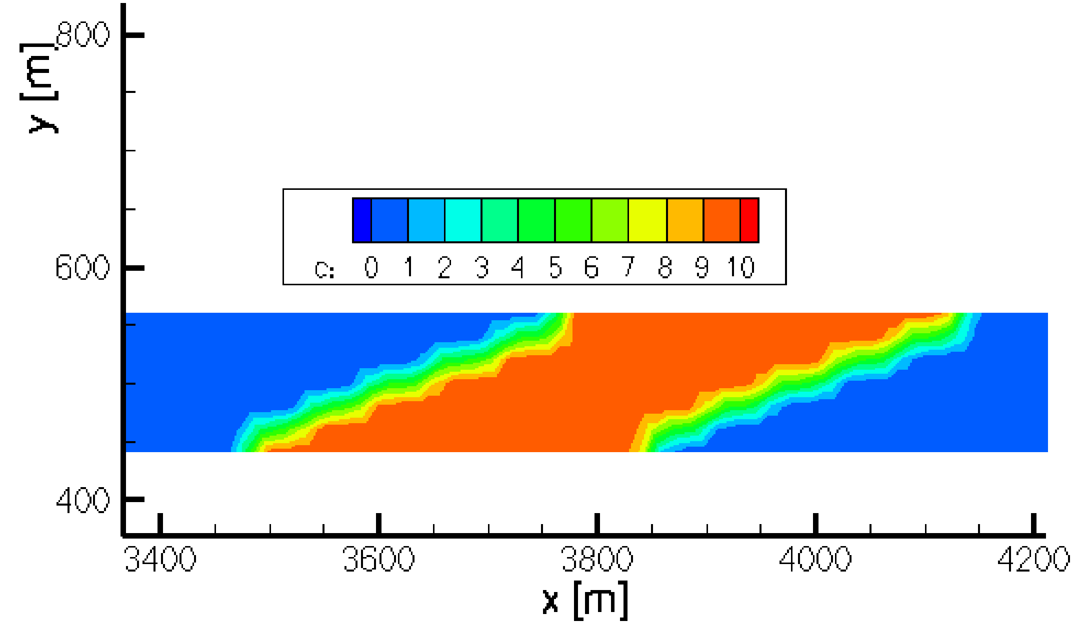


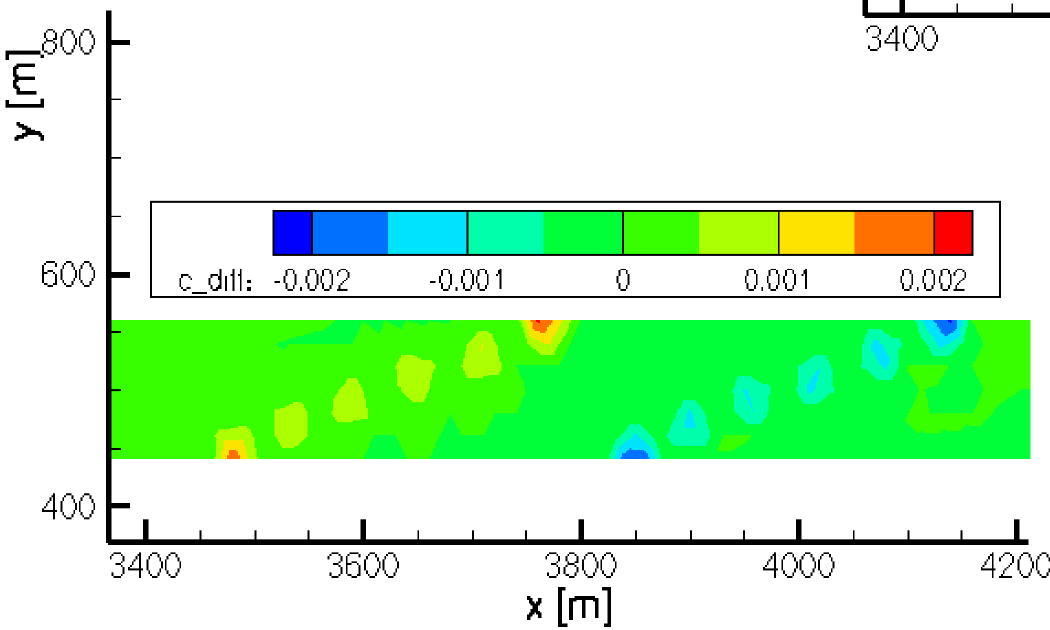


Note the sharp gradients of the concentration distribution

C_diff, 2D

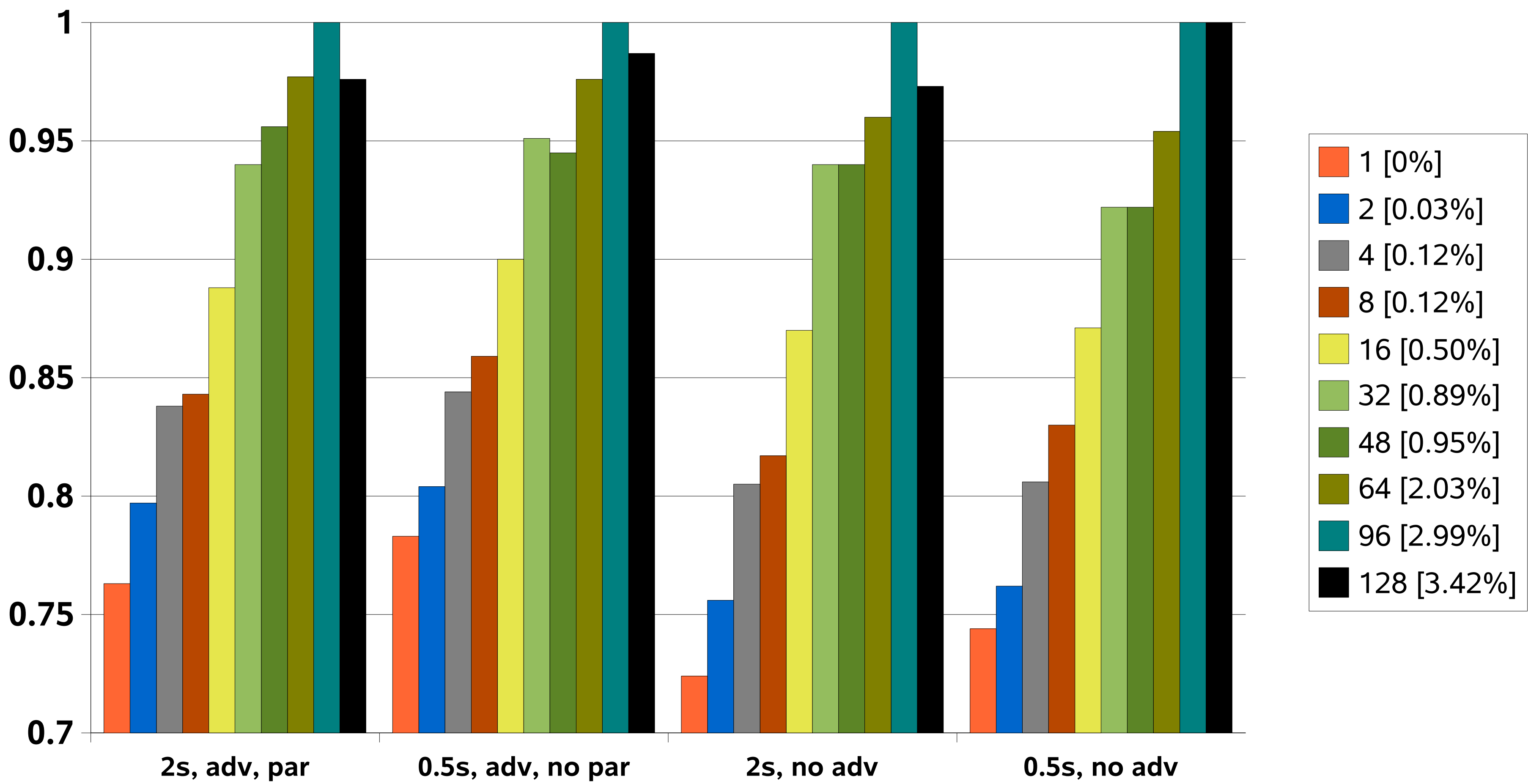
Note the sharp concentration differences





The concentration signal moves slightly too slowly in the parallel case (?)

Coswig - efficiency relative to 96p. (middle water, 2D, ne=738485)



```
export OMP_NUM_THREADS=4
./prog.exe
```

```
mpirun -np 4 ./prog.exe
```