Development and optimization of a coupled multi-GPU LBM-MHFEM solver for vapor transport in air

Jakub Klinkovský[a], Radek Fučík[a], Andrew C. Trautz[b], Tissa H. Illangasekare[c]

[a] Department of Mathematics, Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University in Prague
[b] Research Civil Engineer – Geotechnical Engineering and Geosciences Branch, US Army Engineer Research and Development Center
[c] Center for Experimental Study of Subsurface Environmental Processes (CESEP), Department of Civil and Architectural Engineering, Colorado School of Mines

HPCSE 2022
May 18, 2022

Outline

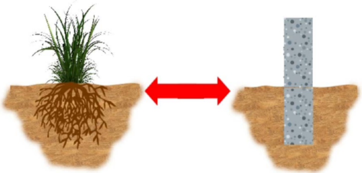
Wind tunnel experiments

Joint work by Andrew C. Trautz[b] and Tissa Illangasekare[c]

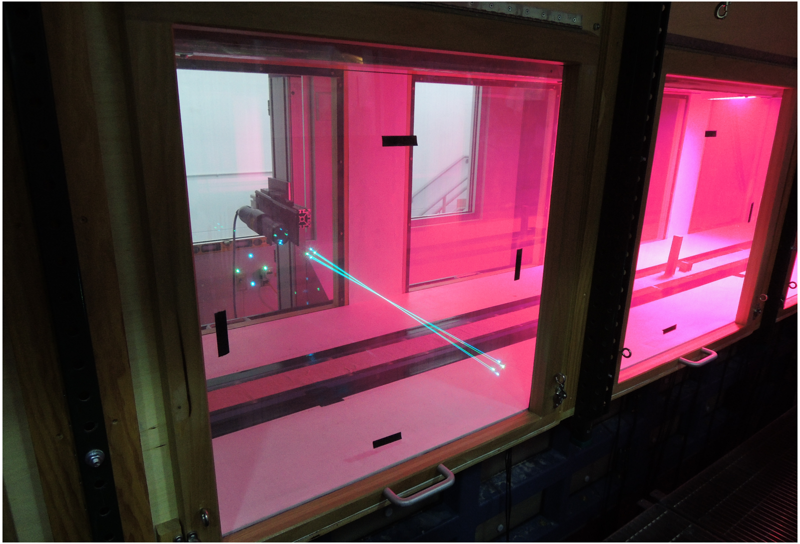
[b]US Army Engineer Research and Development Center
[c]Center for Experimental Study of Subsurface Environmental Processes, Colorado School of Mines

Experiments related to this talk:

- Climate-controlled, low-speed wind tunnel interfaced with a soil tank (CESEP, Colorado School of Mines, USA)
- Designed to study processes with mass flux across the land-atmospheric interface (e.g. water evaporation)
- Live vegetation approximated with limestone blocks

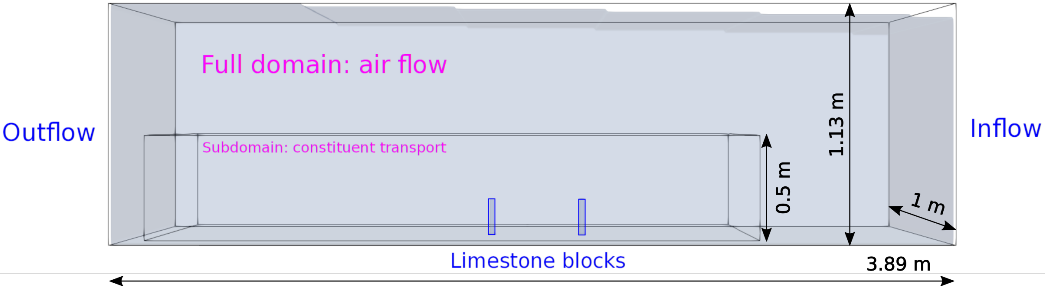


Wind tunnel experiments



Computational domain

Only part of the wind tunnel above soil surface; 2 identical blocks; different spacings.



Governing equations: air flow and vapor transport

NSE (air flow in $\Omega_1 \times (0, t_{\max})$):

$$\nabla \cdot \vec{v} = 0, \qquad (1a)$$

$$\frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} = -\frac{1}{\rho} \nabla p + \nu \Delta \vec{v}, \qquad (1b)$$

ADE (vapor transport in $\Omega_2 \subset \Omega_1$):

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \vec{v} - D \nabla \phi) = 0, \qquad (2a)$$

Or in non-conservative form:

$$\frac{\partial \phi}{\partial t} + \vec{v} \cdot \nabla \phi - \nabla \cdot (D \nabla \phi) = 0. \qquad (2b)$$

- $\vec{v}$ **fluid velocity,**
- ρ fluid density,
- p fluid pressure,
- ν kinematic viscosity of the fluid,

- $\vec{v}$ **fluid velocity,**
- ϕ relative humidity,
- D diffusion coefficient.

Coupled LBM-MHFEM approach

- Equation (1) – lattice Boltzmann method (LBM)
  - D3Q27, Cumulant collision operator *(M. Geier et al., 2015)*
  - in-house code implementation *(R. Straka, R. Fučík, P. Eichler, J. Klinkovský et al.)*
  - implementation details later in this talk
- Equation (2) – mixed-hybrid finite element method (MHFEM)
  - *NumDwarf*: numerical scheme for a system of PDEs in a general-coefficient form
  - details in *R. Fučík, J. Klinkovský, J. Solovský, T. Oberhuber, J. Mikyška, Computer Physics Communications 238 (2019)*
- One-way coupling via the velocity field $\vec{v}$
  - Interpolation from the equidistant lattice to the MHFEM mesh

LBM-MHFEM: coupling details

Interpolation of the velocity $\vec{v}$:

- Trilinear or tricubic interpolation
- Evaluation at **cell side centers** (not cell centers) – to satisfy balancing requirements imposed by the MHFEM discretization

Transport equation:

- $\nabla \cdot \vec{v} = 0$ is not satisfied exactly by the LBM solver (weak compressibility)
- The interpolated velocity field is not locally conservative
- Numerical schemes for the **conservative and non-conservative variants are not equivalent**
- Solving the non-conservative rather than conservative transport equation gives more stable results

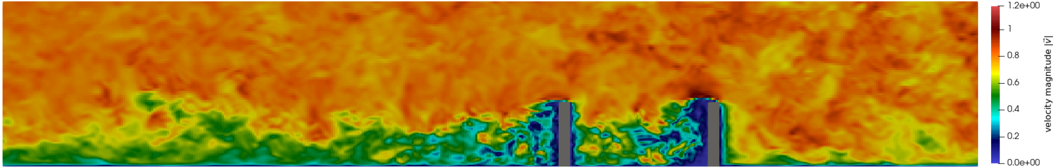
Time stepping:

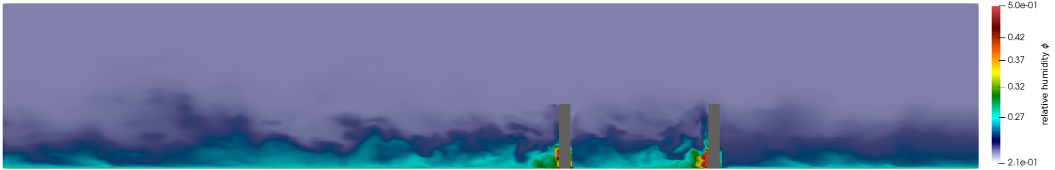
- MHFEM allows to use larger time steps than LBM
- **Adaptive time-stepping** strategy for MHFEM based on a CFL-like condition

Time-stepping algorithm

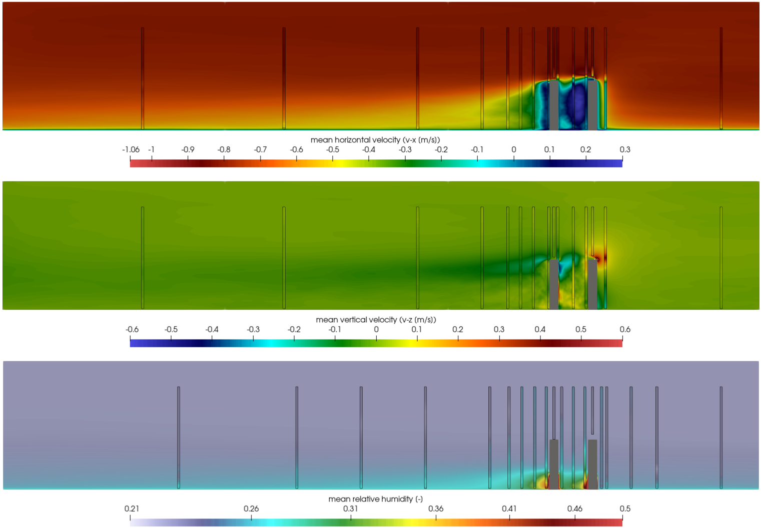
1. Set $t_L := 0$, $t_M := 0$, Δt and $t_{\max} = N_t \Delta t$
2. While $t_L < N_t \Delta t$, repeat these steps:
   1. Perform steps for one iteration of LBM (**details later**)
   2. Set $t_L := t_L + \Delta t$
   3. If $t_M < t_L$, perform these steps:
      1. Interpolate velocity from the lattice to the mesh.
      2. Compute $C = \max_E \{|\vec{v}_E| \Delta t / |E|\}$, where $E \in \mathcal{E}_h$ goes over all faces of the unstructured mesh
      3. Set the time step for MHFEM: $\Delta t_M := \Delta t \lfloor C_{\max}/C \rfloor$ if $C \le C_{\max}$, else $\Delta t_M := \Delta t / \lceil C/C_{\max} \rceil$
      4. Set the number of MHFEM iterations: $n_M := 1$ if $C \le C_{\max}$, else $n_M := \lceil C/C_{\max} \rceil$
      5. Perform n_M iterations of MHFEM with the time step Δt_M
      6. Set $t_M := t_M + n_M \Delta t_M$

Simulations – velocity and relative humidity profiles

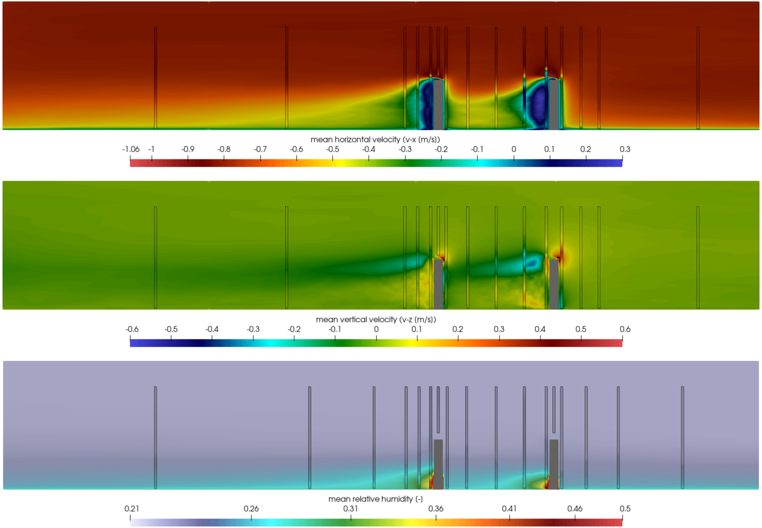




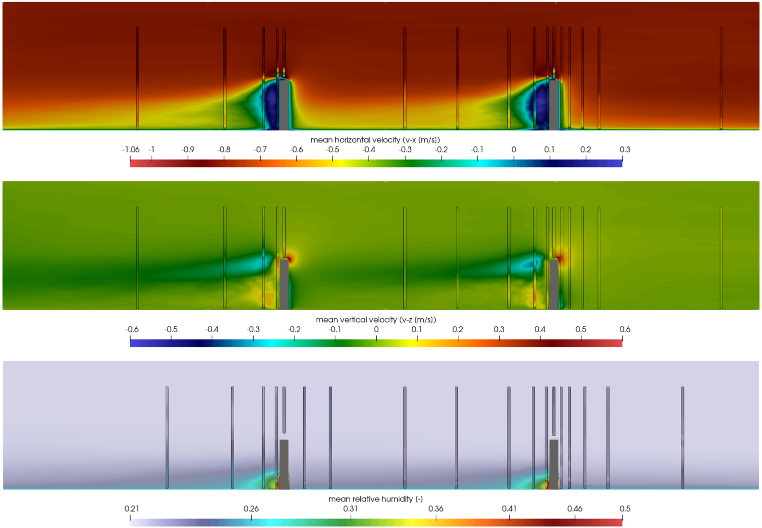
Qualitative comparison with experiment (EX-1: 15 cm)



Qualitative comparison with experiment (EX-2: 45 cm)

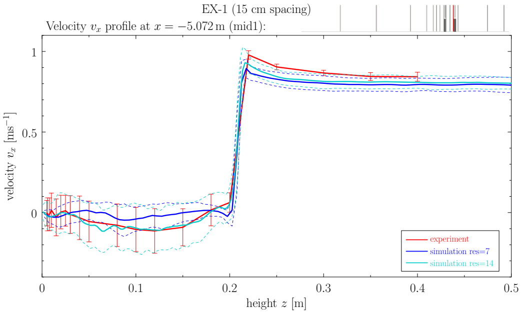


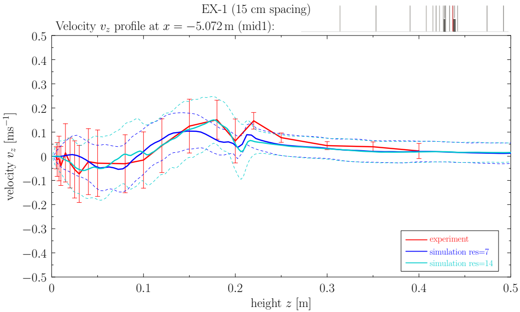
Qualitative comparison with experiment (EX-3: 105 cm)



Quantitative comparison with experiment (EX-1: 15 cm)

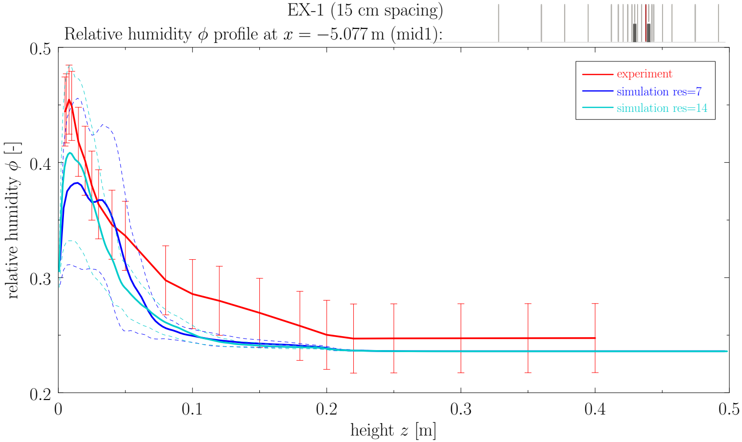
Horizontal and vertical velocity components:





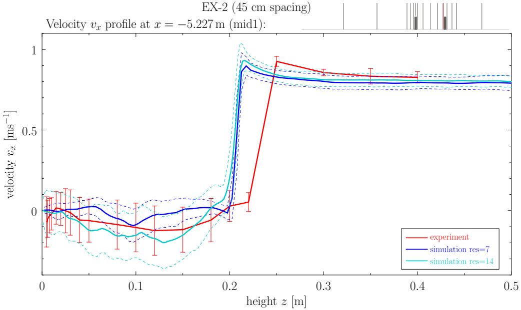
Quantitative comparison with experiment (EX-1: 15 cm)

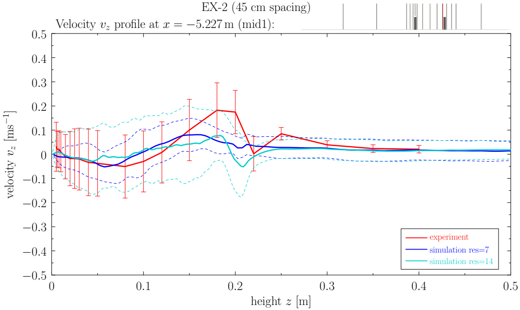
Relative humidity:



Quantitative comparison with experiment (EX-2: 45 cm)

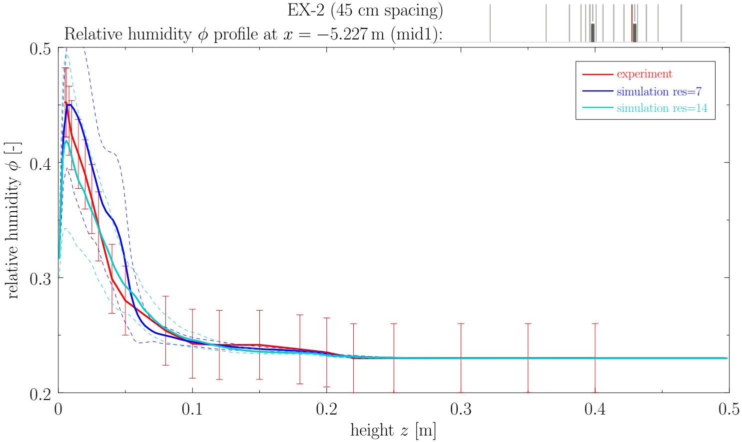
Horizontal and vertical velocity components:





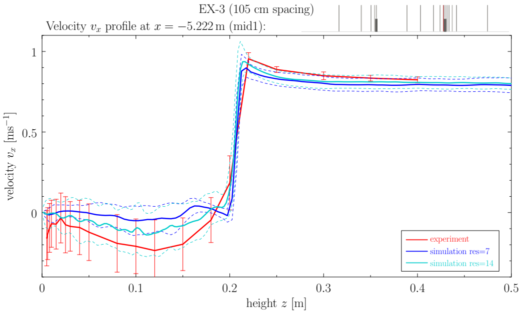
Quantitative comparison with experiment (EX-2: 45 cm)

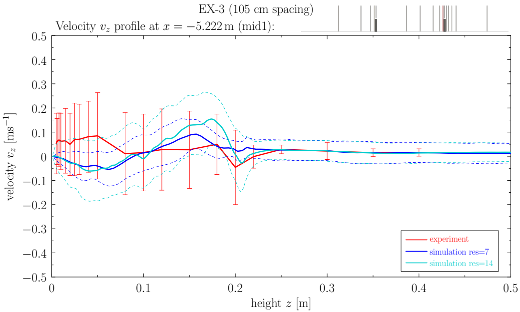
Relative humidity:



Quantitative comparison with experiment (EX-3: 105 cm)

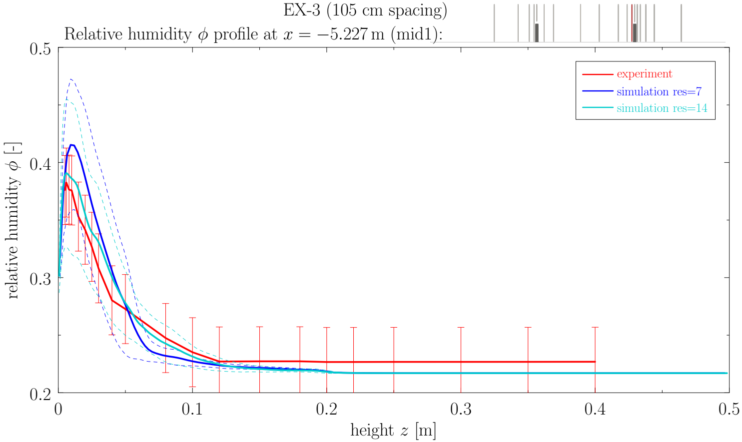
Horizontal and vertical velocity components:





Quantitative comparison with experiment (EX-3: 105 cm)

Relative humidity:



Implementation overview

Computation:

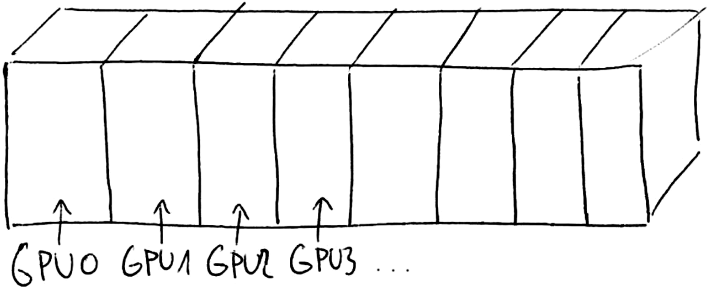
- All parts of the algorithm are computed on a GPU
- Multi-GPU implementation based on MPI

Custom code in C++ developed using:

- **Template Numerical Library**: `https://tnl-project.org/`
- **CUDA**: `https://docs.nvidia.com/cuda/`
- **Message Passing Interface**: `https://www.mpi-forum.org/`

Domain decomposition for LBM

- Computational domain = several independent subdomains + communication
- Computation: subdomains are processed on different GPUs
- Each MPI rank (process) manages its own GPU and subdomain
- Communication: 9 of 27 distribution functions need to be copied between adjacent subdomains
- For simplicity: only 1D distribution (our current implementation)

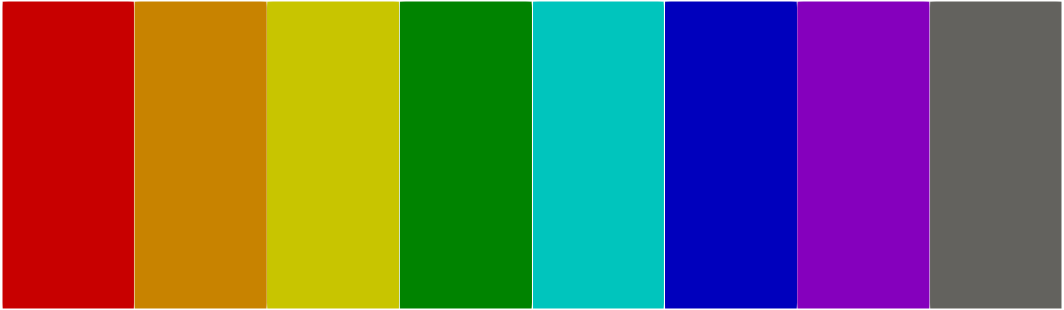


Implemented optimizations

- Domain decomposition with overlapped computation and communication (implementation based on CUDA streams)
- Avoiding buffers in communication (specific ordering of data in multidimensional arrays is necessary)
- Direct GPU-GPU copies via "CUDA-aware" MPI
- Streaming with the A-A pattern – reduced memory requirements
- **Balancing decomposition of the lattice and mesh**

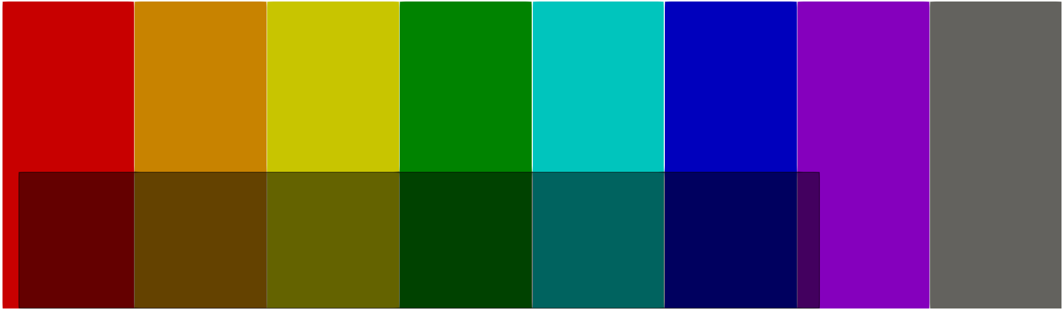
Balancing decomposition of the lattice and mesh

Uniform lattice decomposition: 1/8 of nodes in each subdomain



Balancing decomposition of the lattice and mesh

Uniform lattice decomposition: 1/8 of nodes in each subdomain

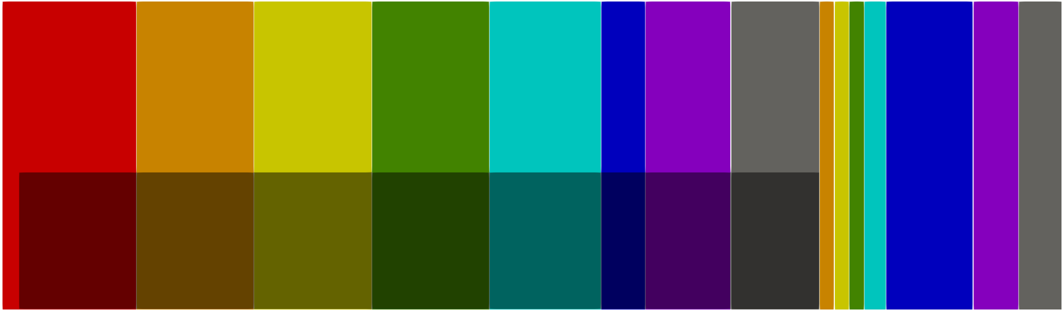


Unstructured mesh decomposition: non-uniform counts of mesh cells

12% 14% 14% 14% 24% 19% 3% 0%

Balancing decomposition of the lattice and mesh

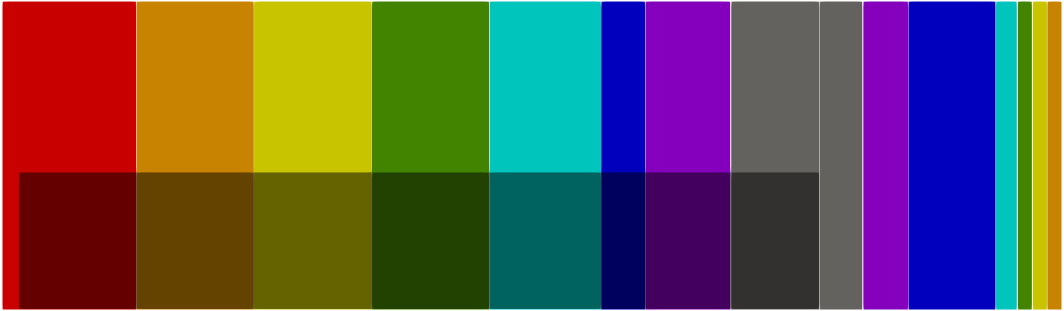
Balanced lattice and mesh decomposition:



Approx. 1/8 of mesh cells **and** approx. 1/8 of lattice nodes per MPI rank.

Balancing decomposition of the lattice and mesh

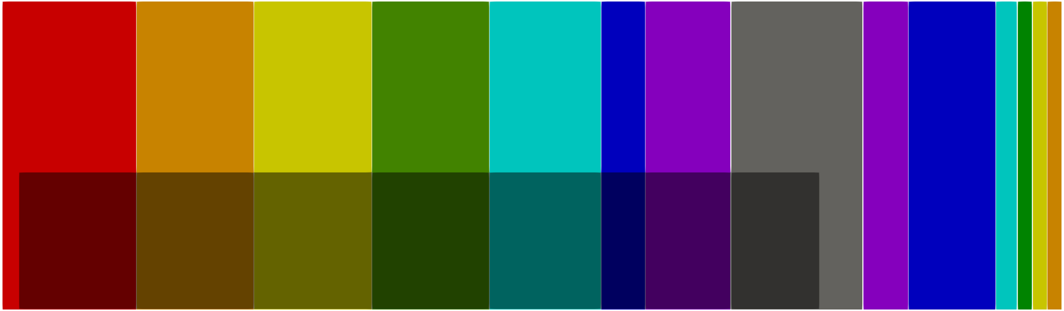
Balanced lattice and mesh decomposition:



Approx. 1/8 of mesh cells **and** approx. 1/8 of lattice nodes per MPI rank.

Balancing decomposition of the lattice and mesh

Balanced lattice and mesh decomposition:



Approx. 1/8 of mesh cells **and** approx. 1/8 of lattice nodes per MPI rank.

Sketch of the decomposition algorithm

Approx. 500 lines of `C++` code:

1. Find the range $[x_1, x_2]$ where the lattice and mesh overlap
2. Define function $F(x) =$ [no. of mesh cells whose centroid is $\leq x$]
   (evaluated on lattice coordinates x_i and interpolated for $x \in \mathbb{R}$)
3. Define **objective function** $f : \mathbb{R}^{N_\text{ranks}} \mapsto \mathbb{R}$, where $N_\text{ranks} =$ [no. of MPI ranks]:
   - input variable $\vec{w}$, where $w_i \in \mathbb{R}$ stands for the width of i-th subdomain
   - imbalance on subinterval $[a, b]$ in the partitioning $= N_\text{ranks} \frac{F(b) - F(a)}{F(x_2) - F(x_1)} - 1$
   - $f(\vec{w}) = [\ell^2$ norm of mesh imbalances for partitioning $\vec{w}]$
4. Minimize f using the **gradient descent** method and the uniform partitioning as initial condition
5. Round the solution from $\mathbb{R}$ to the lattice coordinates (from `double` to `int`)
6. Try to increment/decrement each component of the solution and check if it improves the partitioning (iterative post-optimization in integer precision)
7. Decompose the remaining parts of the lattice which do not overlap with the mesh

Decomposition notes

- Amount of work optimized at the cost of increased communication
- Only 1D decomposition is currently implemented – not scalable
- Tested with up to 16 GPUs (Nvidia A-100) on 2 nodes (RCI cluster on FEE CTU):
  - 16×40 GiB = 640 GiB memory on the GPUs
  - Up to $3115 \times 800 \times 905 \approx 2.25 \times 10^9$ lattice nodes + approx. 48×10^6 mesh cells
  - Computational time: 52 hours (simulation of 100 s physical time)
- Not tested on more GPUs/nodes due to cluster limitations:
  - global allocation limit: only 20 GPUs per user job
  - only 2 nodes have usable inter-node GPU-GPU MPI communication

Performance results (RCI cluster on FEE CTU)

LBM-only performance (i.e., not coupled with MHFEM) – **weak scaling**

NVIDIA Tesla V100:

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	2.5	1.00
1	4	10.4	1.05
2	8	19.3	0.97
4	16	39.4	0.99
8	32	50.1	0.63

NVIDIA Tesla A100:

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	4.8	1.00
1	2	9.8	1.02
1	4	?.?	
1	8	?.?	

LBM-MHFEM performance – **strong scaling**

Weak scaling study is not possible due to different time steps in the MHFEM part.

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	1.1	1.00
1	2	2.1	0.96
1	4	4.1	0.93
1	8	7.5	0.86

Performance results (RCI cluster on FEE CTU)

LBM-only performance (i.e., not coupled with MHFEM) – **weak scaling**

NVIDIA Tesla V100:

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	2.5	1.00
1	4	10.4	1.05
2	8	19.3	0.97
4	16	39.4	0.99
8	32	50.1	0.63

NVIDIA Tesla A100:

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	4.8	1.00
1	2	9.8	1.02
1	4	?.?	
1	8	?.?	

LBM-MHFEM performance – **strong scaling**

Weak scaling study is not possible due to different time steps in the MHFEM part.

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	1.1	1.00
1	2	2.1	0.96
1	4	4.1	0.93
1	8	7.5	0.86

Performance results (RCI cluster on FEE CTU)

LBM-only performance (i.e., not coupled with MHFEM) – **weak scaling**

NVIDIA Tesla V100:

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	2.5	1.00
1	4	10.4	1.05
2	8	19.3	0.97
4	16	39.4	0.99
8	32	50.1	0.63

NVIDIA Tesla A100:

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	4.8	1.00
1	2	9.8	1.02
1	4	?.?	
1	8	?.?	

LBM-MHFEM performance – **strong scaling**

Weak scaling study is not possible due to different time steps in the MHFEM part.

N_{nodes}	N_{GPUs}	GLUPS	Eff
1	1	1.1	1.00
1	2	2.1	0.96
1	4	4.1	0.93
1	8	7.5	0.86

Conclusion

- Validated model for vapor transport in air based on LBM and MHFEM
- Fully multi-GPU solver with good scalability on small number of GPUs

Future work:

- Development of the model (thermodynamics, coupling with porous media, etc.)
- Optimizations for scalability on more GPUs (e.g. multidimensional decomposition)

Thank you for your attention!

Acknowledgements:

- Czech Science Foundation (project 21-09093S)
- Ministry of Education, Youth, and Sports of the Czech Republic (Inter-Excellence grant LTAUSA19021, OP RDE grant CZ.02.1.01/0.0/0.0/16_019/0000765)
- Grant Agency of the Czech Technical University in Prague (project SGS20/184/OHK4/3T/14)

Related papers:

- J. Klinkovský, A. C. Trautz, R. Fučík, T. H. Illangasekare: Lattice Boltzmann Method–Based Efficient GPU Simulator for Vapor Transport in the Boundary Layer Over a Moist Soil: Development and Experimental Validation, in preparation
- R. Fučík, J. Klinkovský, J. Solovský, T. Oberhuber, J. Mikyška: Multidimensional mixed-hybrid finite element method for compositional two-phase flow in heterogeneous porous media and its parallel implementation on GPU, Computer Physics Communications, 2019