

Large-Scale Simulations of Turbulent Flows using Lattice Boltzmann Methods on Heterogeneous High Performance Computers

Adrian Kummerländer, Fedor Bukreev, Yuji Shimojima, Shota Ito
and Mathias J. Krause

Abstract Current GPU-accelerated supercomputers promise to enable large-scale simulations of turbulent flows. Lattice Boltzmann Methods (LBM) are particularly well-suited to fulfilling this promise due to their intrinsic compatibility with highly parallel execution on both SIMD CPUs and GPUs. A novel LBM scheme for wall-modeled LES in complex geometries is described with a special focus on the efficient implementation in the open source LBM framework OpenLB. Detailed scalability results are provided for all HoreKa partitions, utilizing up to 128 nodes and covering problem sizes up to 18 billion cells.

1 Introduction

Multi-physics transport problems are predominant in the physical reality; capturing them in high-fidelity simulations enjoys ever-growing demand in both fundamental research and industrial applications. This present report summarizes our efforts on modeling and simulating such problems on the HoreKa supercomputer using the open source software framework OpenLB [1, 2]. A particular focus is placed on the efficient implementation of large-scale simulations of turbulent fluid flows on GPU clusters.

Reynolds-averaged Navier–Stokes (RANS) models and *Large-Eddy Simulation* (LES) are two often compared approaches to turbulent flow modeling: although RANS approaches outperform LES in terms of computational cost, they suffer from significant accuracy shortcomings in aerodynamic calculations [3]. Despite the availability of modern GPU-accelerated clusters such as HoreKa, LES is often considered as too expensive to replace RANS in practice. Addressing the high

Adrian Kummerländer

Lattice Boltzmann Research Group (LBRG), Institute of Applied and Numerical Mathematics (IANM), Karlsruhe Institute of Technology (KIT) e-mail: kummerlaender@kit.edu



(a) Fully-coupled FSI simulation of a wind park (b) Wall-modelled FSI of a propeller aircraft

Fig. 1: Volumetric rendering for large-scale multi-GPU LES in OpenLB

computational cost of accurate LES for turbulent flows in large-scale, complex geometries is an ongoing research question.

Modern approaches such as *lattice Boltzmann methods* (LBM) [4] offer a promising alternative to conventional *finite-volume methods* (FVM) for conducting LES. OpenLB provides a platform-independent and fully differentiable framework [1, 5–7] for implementing and efficiently executing LBM-based simulations on heterogeneous high-performance computers. In a fair comparison with the FVM solver OpenFOAM, it has previously demonstrated a 32× faster time-to-solution—even without leveraging its multi-GPU capabilities [8].

As covering all recent applications [9–22] of such a multi-physics simulation tool within LBRG, an interdisciplinary group of mathematicians and engineers is beyond the scope of a single report, we focus instead in detail on recent work specifically on modeling large-scale turbulent flows in urban areas [9]. Studies [23–28] show that accurate LES are fundamentally feasible even in larger built-up areas. LBMs have already been used for evaluating wind comfort in complex urban geometries [24], for simulating moist air convection or pressure loads on high-rise buildings [25], for analyzing atmospheric boundary-layer dynamics [26], or for investigating pollutant dispersion [27]. In particular, real-time numerical simulations of urban flows with grid sizes on the order of one meter are possible [27, 28].

Section 2 describes macroscopic equations used to model air flows of urban geometries, their solution by a novel LBM scheme and its efficient platform-transparent implementation in OpenLB. Detailed parallel performance results on all partitions of HoreKa as well as a summary of the experimental validation of the developed numerical wind channel setup is provided in Section 3.

2 Methodology

Modeling air flows and pollution transport in urban geometries is particularly challenging for *computational fluid dynamics* (CFD) due to the combination of large (landscape, buildings) and small length scales (building surfaces, trees). In our

digital twin study [9] we modeled this complex problem using a filtered Brinkman–Navier–Stokes equations (FBNSE), capturing both LES for subgrid-scale turbulence modeling and homogenized porous media for tree modeling in a single target equation. This equation was solved efficiently on HoreKa Green using a *homogenized lattice Boltzmann method* (HLBM) discretization in OpenLB.

2.1 Filtered Brinkmann–Navier–Stokes Equations

The macroscopic motion of fluids is commonly described using the Navier–Stokes equations (NSE). Incompressible flows in heterogeneous domains consisting of both fully fluid regions and porous media can be described using the filtered Brinkman–Navier–Stokes equations (FBNSE)

$$\begin{cases} \nabla \cdot \bar{\mathbf{u}} = 0, & \text{in } \Omega \times I, \\ \frac{\partial \bar{\mathbf{u}}}{\partial t} + \bar{\mathbf{u}} \cdot \nabla \bar{\mathbf{u}} = -\frac{\nabla \bar{p}}{\rho} + \nu_{\text{mo}} \nabla^2 \bar{\mathbf{u}} + \frac{\nu_{\text{mo}}}{K} \bar{\mathbf{u}} - \nabla \cdot \mathbf{T}_{\text{sgs}}, & \text{in } \Omega \times I, \end{cases} \quad (1)$$

for filtered pressure \bar{p} , velocity $\bar{\mathbf{u}}$ density ρ on spatial domain $\Omega \subseteq \mathbb{R}^3$ and time $I \subseteq \mathbb{R}_{>0}$. The molecular kinematic viscosity is defined as ν_{mo} and the permeability coefficient $K > 0$ of the porous medium is given by the Forchheimer equation

$$K = \frac{\mu_F Q}{A \left(\frac{\Delta P}{L} - \frac{\rho}{K_B} \frac{Q^2}{A^2} \right)}, \quad (2)$$

with dynamic viscosity μ , volume flow rate Q , characteristic length L , projected area A , pressure difference ΔP and nonlinear permeability coefficient K_B . The term $\nabla \cdot \mathbf{T}_{\text{sgs}}$ models the subgrid-scale turbulence using the Smagorinsky LES approach

$$\mathbf{T}_{\text{sgs}} = 2\nu_{\text{turb}} \bar{\mathbf{S}}, \quad (3)$$

$$\nu_{\text{turb}} = (C_S \Delta x)^2 |\bar{\mathbf{S}}|, \quad (4)$$

where $C_S > 0$ is the Smagorinsky constant, Δx is the filter width, and $\bar{\mathbf{S}}$ is the filtered strain rate tensor:

$$\bar{S}_{\alpha\beta} = \frac{1}{2} \left(\frac{\partial \bar{u}_\alpha}{\partial x_\beta} + \frac{\partial \bar{u}_\beta}{\partial x_\alpha} \right). \quad (5)$$

2.2 Homogenized Lattice Boltzmann Method

The HLB M is used to discretize the FBNSE (1) on a regular space-time grid with the *D3Q19* velocity stencil (cf. Figure 2). Specifically, we utilize a *homogenized hybrid regularized recursive Lattice Boltzmann method with Smagorinsky LES model*

(HHRRLBM-LES) that extends the classic HLBM [29] with a hybrid third-order recursive regularized collision model [30, 31].

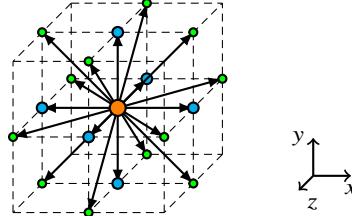


Fig. 2: Schematic illustration of the discrete velocity set $D3Q19$. Figure from [32].

The filtered and homogenized LB equation is given by

$$f_i(\mathbf{x} + \xi_i \Delta t, t + \Delta t) = f_i^{\text{eq}}(\mathbf{x}, t) + \left(1 - \frac{1}{\tau_{\text{eff}}(\mathbf{x}, t)}\right) \tilde{f}_i^{(1)}(\mathbf{x}, t), \quad \text{in } \Omega_{\Delta x} \times I_{\Delta t}, \quad (6)$$

for distribution functions f_i along q discrete velocities ξ_i on a regular lattice $\Omega_{\Delta x} \subset \Omega \subseteq \mathbb{R}^3$ with cell size Δx at discrete times $I_{\Delta t} \subset I \subseteq \mathbb{R}_{\geq 0}$ separated by step size Δt . Here, the non-equilibrium distribution $\tilde{f}_i^{(1)}$ is computed as a linear combination

$$\tilde{f}_i^{(1)}(\mathbf{x}, t) = \sigma f_i^{(1)}(\mathbf{x}, t) - (1 - \sigma) f_i^{(1,FD)} \quad \text{for } \sigma \in [0, 1]. \quad (7)$$

That is, the distribution is *hybridized* between reconstructions using the rate of strain tensor obtained either from local macroscopic moments or from a non-local finite difference (FD) approximation.

For the local part, the non-equilibrium distribution function $f_i^{(1)}$ is expanded in terms of Hermite polynomials $\mathbf{H}_i^{(n)}$ of the discrete velocity ξ_i as

$$f_i^{(1)}(\mathbf{x}, t) = \omega_i \sum_{n=0}^{N=3} \frac{1}{c_s^{2n} n!} \mathbf{H}_i^{(n)} : \mathbf{a}_1^{(n)}(\mathbf{x}, t), \quad (8)$$

where ω_i are the lattice weights. The Hermite expansion coefficients are defined as

$$\mathbf{a}_1^{(n)}(\mathbf{x}, t) = \sum_{i=0}^{q-1} \mathbf{H}_i^{(n)} f_i^{(1)}(\mathbf{x}, t). \quad (9)$$

For the non-local part, the FD non-equilibrium distribution function is defined as

$$f_i^{(1,FD)} := \frac{\rho \tau}{c_s^2} \mathbf{H}_i^{(2)} : \mathbf{S}^{\text{FD}}(\mathbf{x}, t). \quad (10)$$

The equilibrium distribution function is defined as

$$f_i^{\text{eq}}(\mathbf{x}, t) = \omega_i \left(\rho + \frac{\xi_i \cdot \rho \hat{\mathbf{u}}}{c_s^2} + \frac{\mathbf{H}_i^{(2)} : \hat{\mathbf{a}}_0^{(2)}}{2c_s^4} + \frac{\mathbf{H}_i^{(3)} : \hat{\mathbf{a}}_0^{(3)}}{2c_s^6} \right) \quad (11)$$

using Hermite coefficients $\hat{\mathbf{a}}_0^{(0)} = \rho(\mathbf{x}, t)$ and $\hat{\mathbf{a}}_0^{(n)} = \mathbf{a}_0^{(n-1)} \hat{\mathbf{u}}(\mathbf{x}, t)$.

In the general case [29], we define the homogenized velocity $\hat{\mathbf{u}}$ as a convex combination of the fluid velocity moment \mathbf{u} and the solid velocity \mathbf{u}^B , given by

$$\hat{\mathbf{u}}(\mathbf{x}, t) = (1 - d(\mathbf{x}, t))\mathbf{u}(\mathbf{x}, t) + d(\mathbf{x}, t)\mathbf{u}^B(\mathbf{x}, t), \quad (12)$$

where d is the so-called lattice porosity

$$d(\mathbf{x}, t) = 1 - \frac{\Delta x^2 \nu \tau_{\text{mo}}}{K(\mathbf{x}, t)}. \quad (13)$$

and τ_{mo} is the molecular relaxation time. While the the velocity of the solid geometry is set to zero for our urban flow cases, non-zero solid velocities are dynamically prescribed by the wall model in Section 2.2.1.

Finally, the subgrid scale turbulence is accounted for by locally computing the effective relaxation time $\tau_{\text{eff}}(\mathbf{x}, t)$ using the Smagorinsky BGK model

$$\tau_{\text{eff}}(\mathbf{x}, t) = \frac{\nu_{\text{eff}}(\mathbf{x}, t)}{c_s^2} \frac{\Delta t}{\Delta x^2} + \frac{1}{2}. \quad (14)$$

Connecting the HHRLBM (6) to the FBNSE (1) target equation, we expect a second-order approximation in space for the fluid velocity moment [14, 32].

2.2.1 Turbulent Wall Model

It is computationally very expensive and basically infeasible to fully resolve the turbulent structures at the walls in large-scale urban flows. An established approach to this problem is the use of wall modeling s.t. the velocity profiles and shear stresses in the boundary layer are modeled using *universal turbulent velocity profiles*. These functions characterize the streamwise mean velocity u through the normalized variables

$$u^+ = u \sqrt{\frac{\rho}{\tau_w}} = \frac{u}{u_{\tau_w}} \quad (15)$$

and

$$y^+ = \frac{y}{\nu} \sqrt{\frac{\tau_w}{\rho}} = y \frac{u_{\tau_w}}{\nu}, \quad (16)$$

where ν is the fluid kinematic viscosity, y the distance from the wall and τ_w the wall shear stress. For the present case, we use the Spalding wall function which is valid for the range $y^+ \leq 1000$ and defined as

$$y^+ = u^+ + \frac{1}{E} \left[e^{\kappa u^+} - 1 - \kappa u^+ - \frac{(\kappa u^+)^2}{2} - \frac{(\kappa u^+)^3}{6} \right], \quad (17)$$

with a wall roughness parameter E .

The modeled velocity and shear stresses are incorporated into the HRRLLBM as the velocity moment of the solid medium and by setting the hybridization factor σ to 0 (i.e. 100% FD rate of strain tensor) in the wall-modeled region.

2.2.2 Turbulence Generation at the Inlet

The air movements that interact with urban structures follow established *atmospheric velocity profiles* that need to be prescribed at the inlet of the simulation. One possibility of prescribing such perturbed profiles is via the *Vortex Method* (VM) [12].

Let $u_{\text{mean}}, u_{\text{prev}} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ be the mean velocity profile and the inflow velocity of the previous timestep, $A_{\text{inlet}} > 0$ the inlet area and $d_{\text{inlet}} \in \mathbb{R}^3$ the normalized inflow direction. Given the vortex size $\sigma > \delta_x$ and turbulence intensity $I > 0$ as parameters, $n_{\text{vortices}} \in \mathbb{N}$ discrete seed points are placed randomly on the *inflow plane* at positions $p_i \in \mathbb{R}^3$ and associated with signs $s_i \in \{-1, 1\}$. Based on this, per-vortex circulations Γ_i are computed from turbulent kinetic energies k_i :

$$k_i := \frac{3}{2} (|u_{\text{mean}}(p_i)|I)^2, \quad (18)$$

$$\Gamma_i := 4s_i \sqrt{\frac{\pi}{6 \ln 3 - 9 \ln 2} \frac{k_i A_{\text{inlet}}}{n_{\text{vortices}}}}. \quad (19)$$

Together, this allows for recovering the perturbed velocity field

$$u_{\text{vortex}}(x) := \frac{1}{2\pi} \sum_{i=1}^{n_{\text{vortices}}} \Gamma_i \frac{((p_i - x) \times d_{\text{inlet}}) \left(1 - \exp \left(\frac{|x-p_i|^2}{2\sigma^2} \right) \right)}{|x - p_i|^2} \quad (20)$$

which results in the final inflow velocity field with streamwise fluctuation using the Langevin equation

$$u(x) := u_{\text{mean}}(x) + u_{\text{vortex}}(x) - \frac{u_{\text{vortex}}(x) \times (\nabla u_{\text{prev}})(x)}{|(\nabla u_{\text{prev}})(x)|} d_{\text{inlet}}. \quad (21)$$

The perturbed inflow velocity field is computed for each discrete LB timestep and applied to the lattice using a regularized local velocity boundary condition.

2.3 Implementation in OpenLB

At its core, OpenLB is a framework for efficiently and in parallel applying operators to distributed field-structured data on regular lattices. Any such operator can transparently be executed both on CPU and GPU targets, transferring this ability to LB schemes implemented within this framework [7].

A special case is the application of the LBM propagation step, the implementation of which is a critical component in the resulting performance [5]. OpenLB utilizes an *implicit* pattern that enables the propagation of populations between neighboring cells without copying but changing the view of the data. As depicted in Figure 3 this is achieved by storing the data in a vectorization friendly *Structure of Arrays* (SoA) layout and rotating the array views within cyclic buffers.

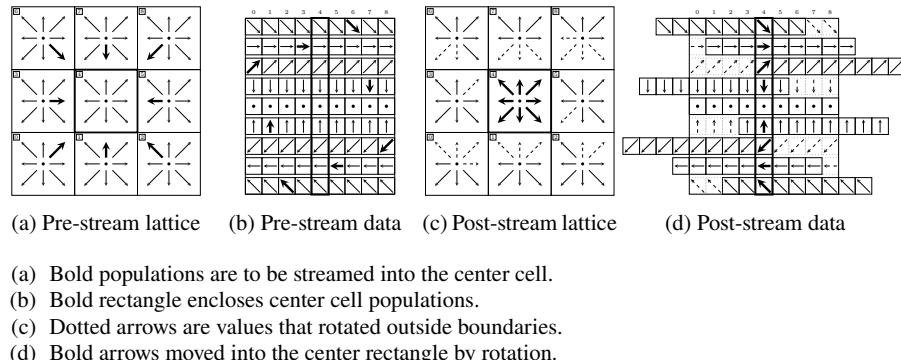


Fig. 3: Implicit propagation without data-transfer using Periodic Shift (PS) [5]

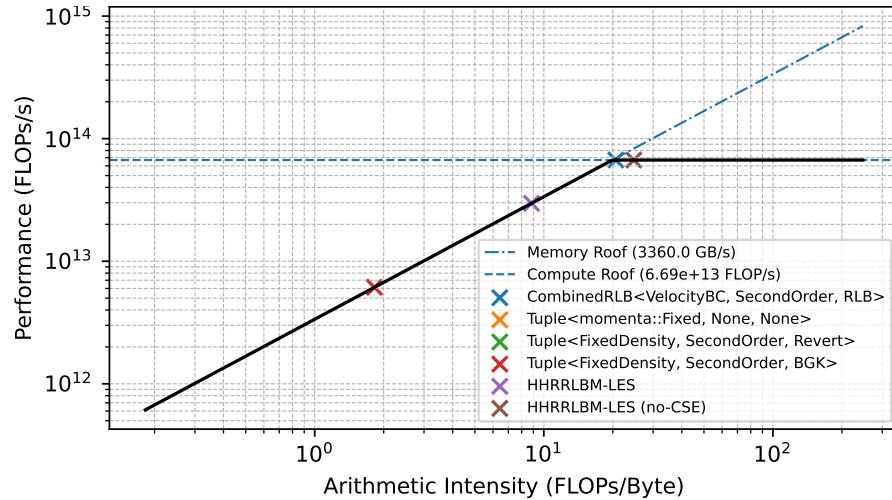
The HHRRRLBM (6) collision step detailed in Section 2.2 is implemented using OpenLB’s dynamics tuple system, a kind of *domain specific language* (DSL) for local LB cell models [7]. Listing 1 shows how the model is constructed as a tuple of moment system, equilibrium and collision operator. Due to the abstract implementation of all LB models against the *concept of a cell* they are not only amenable to platform-transparent execution but also automatic code optimization using common subexpression elimination (CSE). For the present HHRRRLBM model, OpenLB reports a per-collision memory bandwidth of 188 bytes and a floating point complexity of 1661 FLOPs (cf. 4644 FLOPs in the unoptimized case, a ~ 2.8 fold reduction). A theoretical roofline analysis for the involved dynamics utilizing OpenLB’s introspection data is shown in Figure 4.

```

1 using HHRLBM = dynamics::Tuple<
2   T, descriptors::D3Q19>,           // Value type and lattice stencil
3   typename momenta::Tuple<        // Macroscopic moment system
4     momenta::BulkDensity,
5     momenta::MovingPorousMomentumCombination<momenta::BulkMomentum>,
6     momenta::BulkStress,
7     momenta::DefineToNEq
8   >,
9   equilibria::ThirdOrder,          // Equilibrium distribution
10  collision::ParameterFromCell< // Modified collision operator
11    collision::LES::SMAGORINSKY,
12    collision::SmagorinskyEffectiveOmega<collision::ThirdOrderRLB>>
13 >;

```

Listing 1: Dynamics tuple formulation of the HHRLBM scheme



CSE optimization is critical for reducing the arithmetic intensity of HHRLBM-LES, rendering the collision step bandwidth-limited in the ideal case. Memory and compute roofs given by theoretical maximums provided in the data sheets.

Fig. 4: Roofline analysis of local cell models for NVIDIA H100-94

3 Evaluation

3.1 Parallel Efficiency

We first investigate OpenLB's weak- and strong-scaling efficiency for the established lid-driven cavity benchmark case on all partitions of the HoreKa supercomputer. OpenLB provides transparent execution for LB models across CPUs and NVIDIA GPUs, utilizing hybrid MPI+OpenMP+SIMD execution of CPU targets

and MPI+CUDA on GPUs. The base arithmetic type is set to IEEE single-precision for all samples.

Figures 5a and 5b are expanded versions of the previously published scalability study [33] on up to 128 nodes of HoreKa Blue resp. Green while Figure 5c details performance on up to 16 HoreKa Teal nodes (maximum number allocatable in accelerated-h100). Problem sizes range between 190 million and 18 billion cells and strong efficiencies are listed additionally in Table 1.

In all plots, the absolute and per-node performance is measured in *billions of cell updates per second* (GLUPs). This is the established reference quantity for LBM performance measurements and it is directly related to the *time-to-solution* by

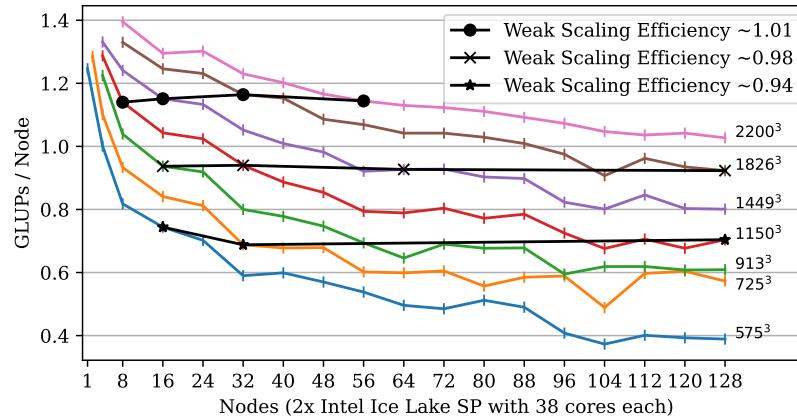
$$t_{\text{solution}}(N) = \frac{\Sigma_{\text{cells}}(N)\Sigma_{\text{timesteps}}(N)}{1e9p(N)} \quad (22)$$

for number of cells Σ_{cells} , number of timesteps $\Sigma_{\text{timesteps}}$ and performance measurement in GLUPs p for fixed resolution N .

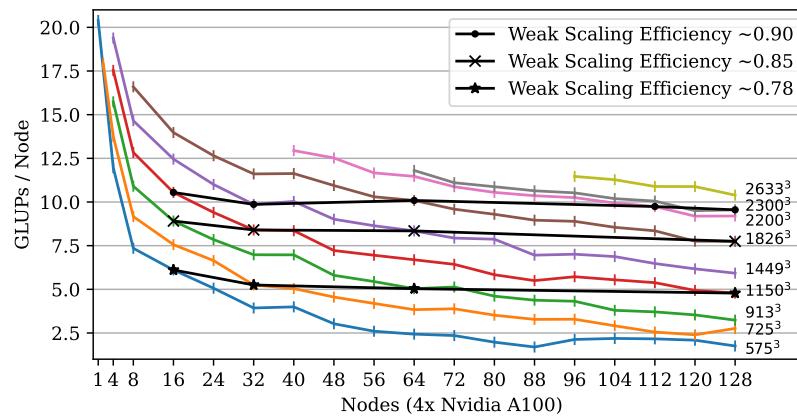
Size	HoreKa Blue		HoreKa Green		HoreKa Teal	
	[32→128]	[64→128]	[32→128]	[64→128]	[4→16]	[8→16]
575 ³	0.66	0.78	0.45	0.72	0.48	0.80
725 ³	0.83	0.96	0.53	0.72	0.54	0.83
913 ³	0.76	0.94	0.46	0.64	0.56	0.83
1150 ³	0.75	0.89	0.57	0.71	0.62	0.86
1449 ³	0.76	0.86	0.60	0.71	0.67	0.91
1826 ³	0.79	0.89	0.67	0.77	—	—
2200 ³	0.84	0.91	—	0.80	—	—
2300 ³	—	—	—	0.81	—	—

Table 1: Strong scaling efficiencies on HoreKa for different problem sizes

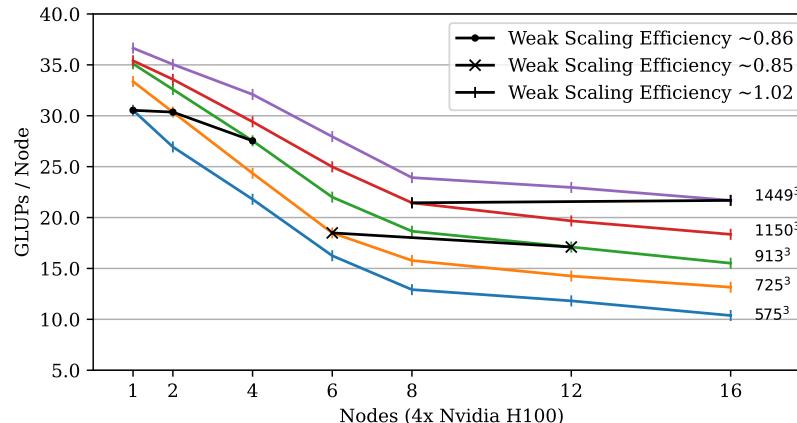
Summarizing the use of HoreKa for all applications within the project and not just the exemplary urban flow studies, we observe a continuing switch towards using a low number of accelerated nodes instead of many CPU-only nodes. Even a single HoreKa GPU node is sufficient for simulations up to $\sim 1e9$ cells which is already beyond the needs of many practical applications.



(a) HoreKa Blue using MPI+OpenMP+AVX512



(b) HoreKa Green using MPI+CUDA



(c) HoreKa Teal using MPI+CUDA

Fig. 5: Parallel efficiency of OpenLB on HoreKa

3.2 Urban Flow Reference Case

For a full description and evaluation of the urban digital twin implemented using the HHRLBM approach detailed in Section 2 we refer to [9]. Here, we summarize the validation against experimental and simulation data [34].

The geometric setup depicted in Figure 6 represents two rows of buildings forming a *street canyon*. In OpenLB, the atmospheric inlet profile [23] is modeled using the VM, the outlet using a fixed local pressure condition with fringe zone and the outer walls using full slip boundaries.

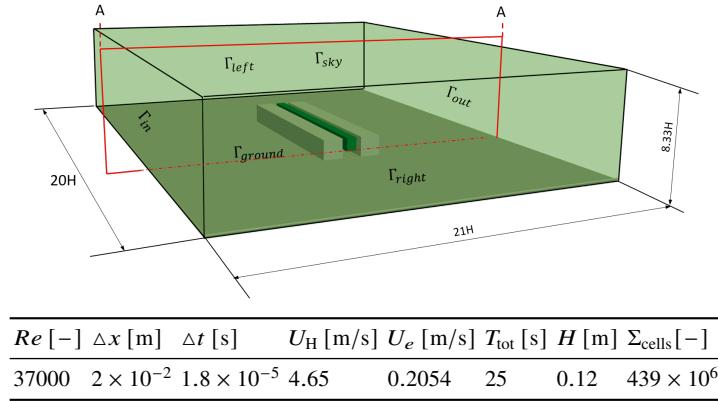
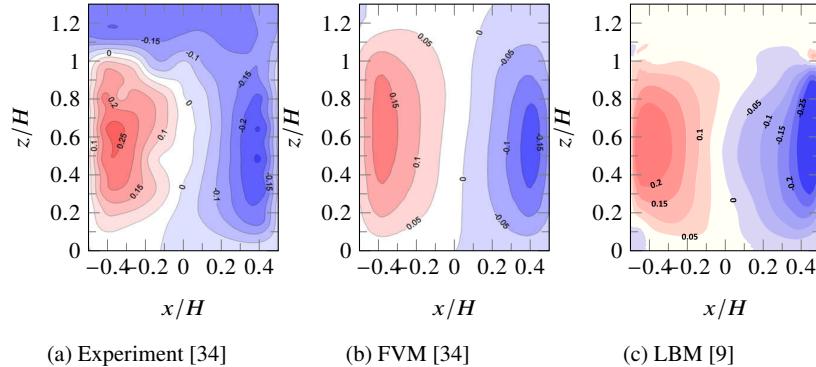
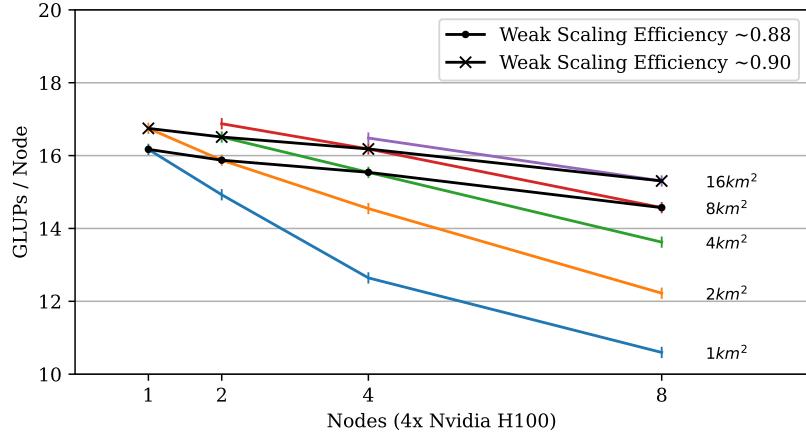


Fig. 6: Simulation setup for reproducing the experimental reference case [9, 34]



Normalized vertical velocity magnitude for the canyon reference case along plane cut A-A (cf. Figure 6). FVM results obtained using FLUENT [34] and LBM results using OpenLB [9]. Both simulations used turbulent wall modeling. LBM results obtained using $438e6$ cells on HoreKa Teal.

Fig. 7: Comparison of experimental and simulation results for the canyon case [34]



Single-precision simulations of the FBNSE-only numerical wind channel case including VM turbulent inlet condition and wall modeling. City sections between 1 and 16km² are resolved with a voxel size of 1m yielding problem sizes between 0.5×10^9 and 8×10^9 cells.

Fig. 8: Parallel efficiency of the numerical wind channel on HoreKa

Based on the validation of the basic simulation setup summarized in Figure 7, a full-scale version of the numerical wind channel was used for building a digital twin [9] incorporating real-world pollution measurements. Figure 8 show a combined weak- and strong-scaling study of the numerical wind channel in OpenLB for city sections of up to 16km². Despite the lower absolute performance explained by the significantly more complex simulation setup, parallel efficiency on HoreKa Teal compares well to the lid-driven cavity benchmark results in Figure 5c.

4 Conclusion

HHRRLBM-LES was described as a parallelization-friendly discretization for the filtered Brinkmann-Navier-Stokes equations. Its efficient implementation for platform-transparent execution on both SIMD CPUs (using MPI+OpenMP+AVX512) and GPUs (using MPI+CUDA) in OpenLB was detailed. Automatic code optimization was discussed and the performance of local cell models was studied using a roofline analysis. Extensive strong and weak scaling evaluations with efficiencies up to 1.02 for problem sizes up to 18 billion cells were provided. Transferability of parallel efficiency from ideal benchmark cases to complex wall-modeled turbulent flow cases was demonstrated.

In summary, this showcases how OpenLB can deliver on the promise of fast large-scale turbulent flow simulations on state-of-the-art supercomputers.

Resource Usage

As the budgets are not separated between the current and past CPE project in the `kit-project-usage` reporting command we can only provide total used resources:

	Project	Resource	Granted	Used	Percent
2	hk-project-cpe	cpu	81932606	72868631.5	88.93%
3	hk-project-cpe	gres/gpu	719495	848282	117.89%

References

- [1] M. J. Krause, A. Kummerländer, S. J. Avis, H. Kusumaatmaja, D. Dapelo, F. Klemens, M. Gaedtke, N. Hafen, A. Mink, R. Trunk, J. E. Marquardt, M.-L. Maier, M. Haussmann, S. Simonis, OpenLB—open source lattice boltzmann code, *Computers & Mathematics with Applications* 81 258–288. doi:10.1016/j.camwa.2020.04.033.
- [2] A. Kummerländer, T. Bingert, F. Bukreev, L. E. Czelusniak, D. Dapelo, C. Gaul, N. Hafen, S. Ito, J. Jeßberger, D. Khazaeipoul, T. Krüger, H. Kusumaatmaja, J. E. Marquardt, A. Raeli, M. Rennick, F. Prinz, M. Schecher, A. Schneider, Y. Shimojima, S. Simonis, P. Sitter, P. Spelten, A. Tacques, D. Teutscher, M. Zhong, M. J. Krause, OpenLB release 1.8.1: Open source lattice boltzmann code. doi:10.5281/zenodo.15440776.
- [3] B. Blocken, S. , Ted, C. , Jan, , J. L. Hensen, Application of computational fluid dynamics in building performance simulation for the outdoor environment: an overview, *Journal of Building Performance Simulation* 4 (2). doi:10.1080/19401493.2010.513740.
- [4] S. Chen, G. D. Doolen, LATTICE BOLTZMANN METHOD FOR FLUID FLOWS, *Annual Review of Fluid Mechanics* 30. doi:10.1146/annurev.fluid.30.1.329.
- [5] A. Kummerländer, M. Dorn, M. Frank, M. J. Krause, Implicit propagation of directly addressed grids in lattice Boltzmann methods, *Concurrency and Computation: Practice and Experience* 35 (8) (Apr. 2023). doi:10.1002/cpe.7509.
- [6] A. Kummerländer, M. J. Krause, Research Software Engineering in OpenLB: Refactoring a Legacy Code to State-Of-The-Art Performance, deRSE23 Conference For Research Software Engineering in Germany, Paderborn, 2023 (Feb. 2023). doi:10.5281/zenodo.7662082.
- [7] A. Kummerländer, S. Ito, M. Schecher, M. J. Krause, Openlb: A comprehensive approach to lattice boltzmann methods on heterogeneous high performance computers, In Preparation.
- [8] M. Haussmann, F. Ries, J. B. Jeppener-Haltenhoff, Y. Li, M. Schmidt, C. Welch, L. Illmann, B. Böhm, H. Nirschl, M. J. Krause, A. Sadiki, Evaluation of a Near-Wall-Modeled Large Eddy Lattice Boltzmann Method for the Analysis

- of Complex Flows Relevant to IC Engines, *Computation* 8 (2) (2020). doi: 10.3390/computation8020043.
- [9] D. Teutscher, F. Bukreev, A. Kummerländer, S. Simonis, P. Bächler, A. Rezaee, M. Hermansdorfer, M. J. Krause, A digital urban twin enabling interactive pollution predictions and enhanced planning, *Building and Environment* 281. doi:10.1016/j.buildenv.2025.113093.
 - [10] S. Ito, S. Großmann, F. Bukreev, J. Jeßberger, M. J. Krause, Benchmark case for the inverse determination of adsorption parameters using lattice boltzmann methods and gradient-based optimization, *Chemical Engineering Science* (2025). doi:10.1016/j.ces.2025.121467.
 - [11] J. E. Marquardt, B. Eysel, M. Sadric, C. Rauh, M. J. Krause, Potential for damage to fruits during transport through cross-section constrictions, *Journal of Food Engineering* (2025). doi:10.1016/j.jfoodeng.2025.112473.
 - [12] M. Hettel, F. Bukreev, E. Daymo, A. Kummerländer, M. J. Krause, O. Deutschmann, Calculation of single and multiple low reynolds number free jets with a lattice-boltzmann method, *AIAA Journal* (2025). doi: 10.2514/1.J064280.
 - [13] M. Zhong, T. Xiao, M. J. Krause, M. Frank, S. Simonis, A stochastic galerkin lattice boltzmann method for incompressible fluid flows with uncertainties, *Journal of Computational Physics* (2024). doi:10.1016/j.jcp.2024.113344.
 - [14] S. Simonis, N. Hafen, J. Jeßberger, D. Dapelo, G. Thäter, M. J. Krause, Homogenized lattice Boltzmann methods for fluid flow through porous media – part I: kinetic model derivation, *ESAIM M2AN* (2025). doi:10.1051/m2an/2025005.
 - [15] F. Bukreev, A. Kummerländer, J. Jeßberger, D. Teutscher, S. Simonis, D. Bothe, M. J. Krause, Benchmark simulation of laminar reactive micromixing using lattice boltzmann methods, *AIAA Journal* (2025). doi:10.2514/1.J064234.
 - [16] S. Simonis, J. Nguyen, S. J. Avis, W. Dörfler, M. J. Krause, Binary fluid flow simulations with free energy lattice boltzmann methods, *Discrete and Continuous Dynamical Systems* (2024). doi:10.3934/dcdss.2023069.
 - [17] J. E. Marquardt, N. Hafen, M. J. Krause, A novel model for direct numerical simulation of suspension dynamics with arbitrarily shaped convex particles, *Computer Physics Communications* (2024). doi:10.1016/j.cpc.2024.109321.
 - [18] J. E. Marquardt, N. Hafen, M. J. Krause, A novel particle decomposition scheme to improve parallel performance of fully resolved particulate flow simulations, *Journal of Computational Science* (2024). doi:10.1016/j.jocs.2024.102263.
 - [19] S. Ito, J. Jeßberger, S. Simonis, F. Bukreev, A. Kummerländer, A. Zimmermann, G. Thäter, G. R. Pesch, J. Thöming, M. J. Krause, Identification of reaction rate parameters from uncertain spatially distributed concentration data using gradient-based pde constrained optimization, *Computers & Mathematics with Applications* (2024). doi:10.1016/j.camwa.2024.05.026.

- [20] F. Prinz, J. Pokorný, J. Elcner, F. Lízal, O. Mišík, M. Malý, M. Bělka, N. Hafen, A. Kummerländer, M. J. Krause, J. Jedelský, M. Jícha, Comprehensive experimental and numerical validation of lattice boltzmann fluid flow and particle simulations in a child respiratory tract, *Computers in Biology and Medicine* (2024). doi:10.1016/j.combiomed.2024.107994.
- [21] N. Hafen, J. E. Marquardt, A. Dittler, M. J. Krause, Simulation of dynamic rearrangement events in wall-flow filters applying lattice boltzmann methods, *Fluids* (2023). doi:10.3390/fluids8070213.
- [22] F. Bukreev, S. Simonis, A. Kummerländer, J. Jeßberger, M. J. Krause, Consistent lattice boltzmann methods for the volume averaged navier–stokes equations, *Journal of Computational Physics* (2023). doi:10.1016/j.jcp.2023.112301.
- [23] M. Pasquier, S. Jay, J. Jacob, P. Sagaut, A Lattice-Boltzmann-based modelling chain for traffic-related atmospheric pollutant dispersion at the local urban scale, *Building and Environment* 242 (2023) 110562. doi:10.1016/j.buildenv.2023.110562.
- [24] N. H. Ahmad, A. Inagaki, M. Kanda, N. Onodera, T. Aoki, Large-eddy simulation of the gust index in an urban area using the lattice boltzmann method, *Boundary-Layer Meteorology* 163 (3). doi:10.1007/s10546-017-0233-6.
- [25] J. Jacob, L. Merlier, F. Marlow, P. Sagaut, Lattice boltzmann method-based simulations of pollutant dispersion and urban physics, *Atmosphere* 12 (7) (2021). doi:10.3390/atmos12070833.
- [26] A. Inagaki, M. Kanda, N. H. Ahmad, A. Yagi, N. Onodera, T. Aoki, A numerical study of turbulence statistics and the structure of a spatially-developing boundary layer over a realistic urban geometry, *Boundary-Layer Meteorology* 164 (2). doi:10.1007/s10546-017-0249-y.
- [27] N. Onodera, Y. Idomura, Y. Hasegawa, H. Nakayama, T. Shimokawabe, T. Aoki, Real-time tracer dispersion simulations in oklahoma city using the locally mesh-refined lattice boltzmann method, *Boundary-Layer Meteorology* 179 (2). doi:10.1007/s10546-020-00594-x.
- [28] S. Lenz, M. Schönherr, M. Geier, M. Krafczyk, A. Pasquali, A. Christen, M. Giometto, Towards real-time simulation of turbulent air flow over a resolved urban canopy using the cumulant lattice boltzmann method on a GPGPU, *Journal of Wind Engineering and Industrial Aerodynamics* 189 151–162. doi:10.1016/j.jweia.2019.03.012.
- [29] M. J. Krause, F. Klemens, T. Henn, R. Trunk, H. Nirschl, Particle flow simulations with homogenised lattice Boltzmann methods, *Particuology* 34 (2017) 1–13. doi:10.1016/j.partic.2016.11.001.
- [30] C. Coreixas, G. Wissocq, G. Puigt, J.-F. m. c. Boussuge, P. Sagaut, Recursive regularization step for high-order lattice boltzmann methods, *Phys. Rev. E* 96 (Sep 2017). doi:10.1103/PhysRevE.96.033306.
- [31] O. M. Jérôme Jacob, P. Sagaut, A new hybrid recursive regularised Bhatnagar–Gross–Krook collision model for Lattice Boltzmann method-based

- large eddy simulation, *Journal of Turbulence* 19 (11-12) (2018). doi:
10.1080/14685248.2018.1540879.
- [32] S. Simonis, Lattice Boltzmann Methods for Partial Differential Equations, Doctoral thesis, Karlsruhe Institute of Technology (KIT) (2023). doi:10.5445/IR/1000161726.
 - [33] A. Kummerländer, F. Bukreev, S. F. R. Berg, M. Dorn, M. J. Krause, Advances in computational process engineering using lattice boltzmann methods on high performance computers, in: W. E. Nagel, D. H. Kröner, M. M. Resch (Eds.), High Performance Computing in Science and Engineering '22, Springer Nature Switzerland, Cham, 2024, pp. 233–247.
 - [34] C. Gromke, R. Buccolieri, S. Di Sabatino, B. Ruck, Dispersion study in a street canyon with tree planting by means of wind tunnel and numerical investigations – Evaluation of CFD data with experimental data, *Atmospheric Environment* 42 (37) (2008). doi:10.1016/j.atmosenv.2008.08.019.