

# Molecular Dynamics - Assignment 4

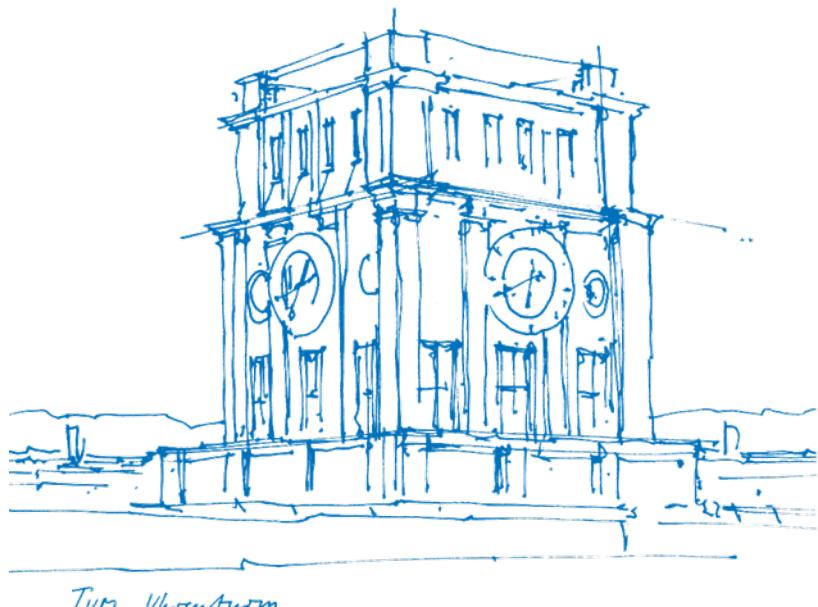
Alex Hocks Jan Hampe Johannes Riemenschneider

Technische Universität München

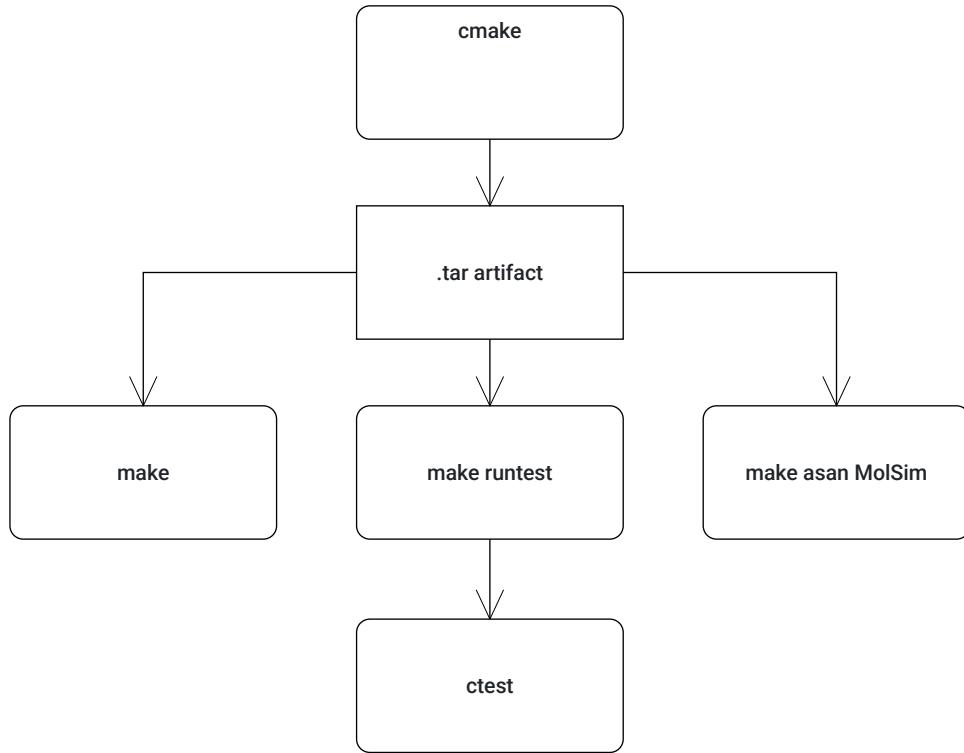
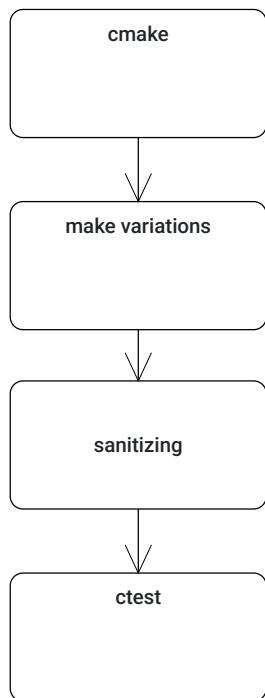
TUM CIT

Lehrstuhl für wissenschaftliches Rechnen

23. Dezember 2022

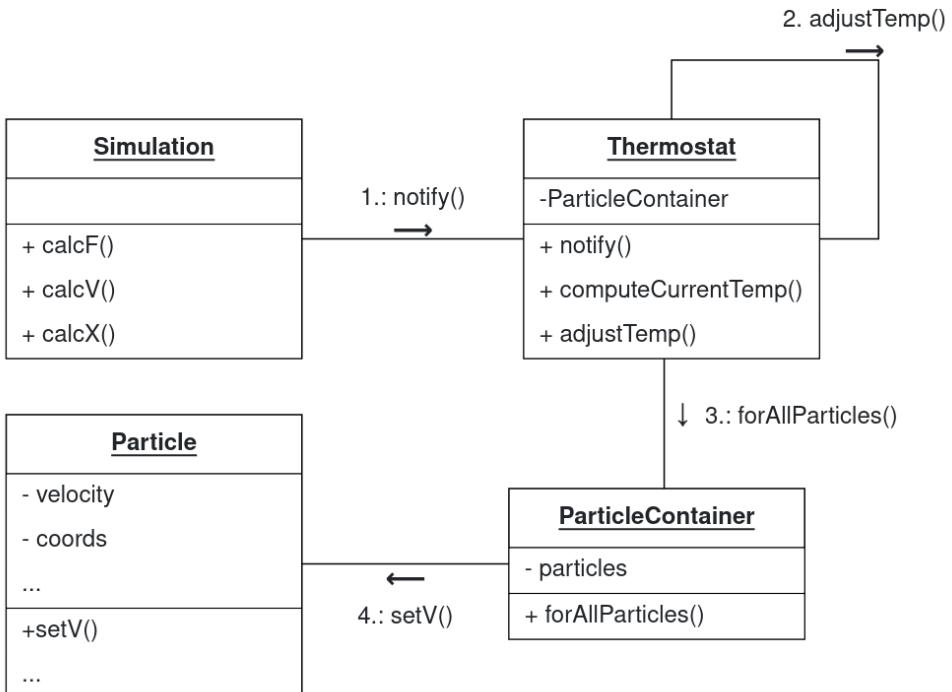


# CI/CD improvements



Reduced CI/CD time from 5m30s to 3m

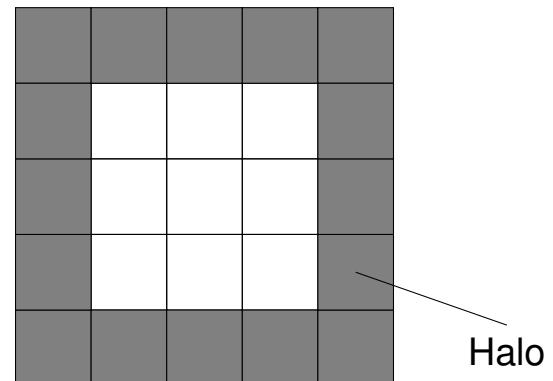
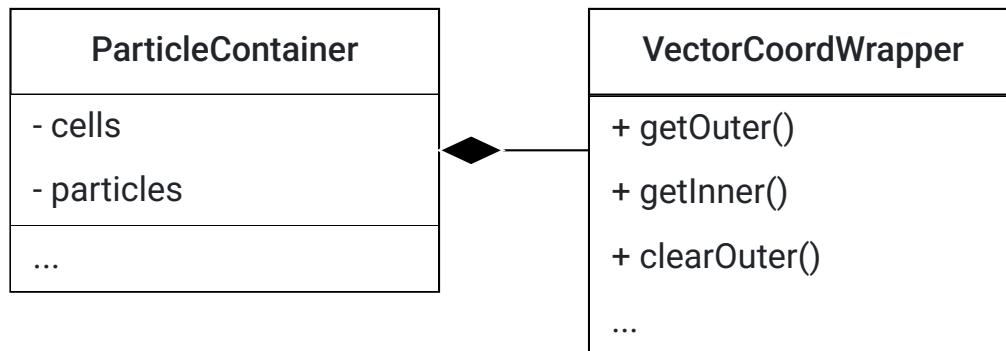
# Thermostat



# Adapting ParticleContainer for periodic bounds

Idea:

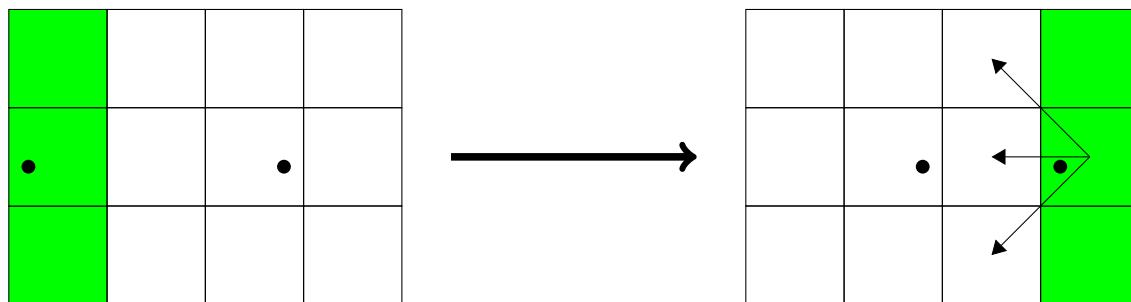
- Provide virtual cells around the actual domain for anyone who needs it
- Existence of additional cells is invisible with old interface



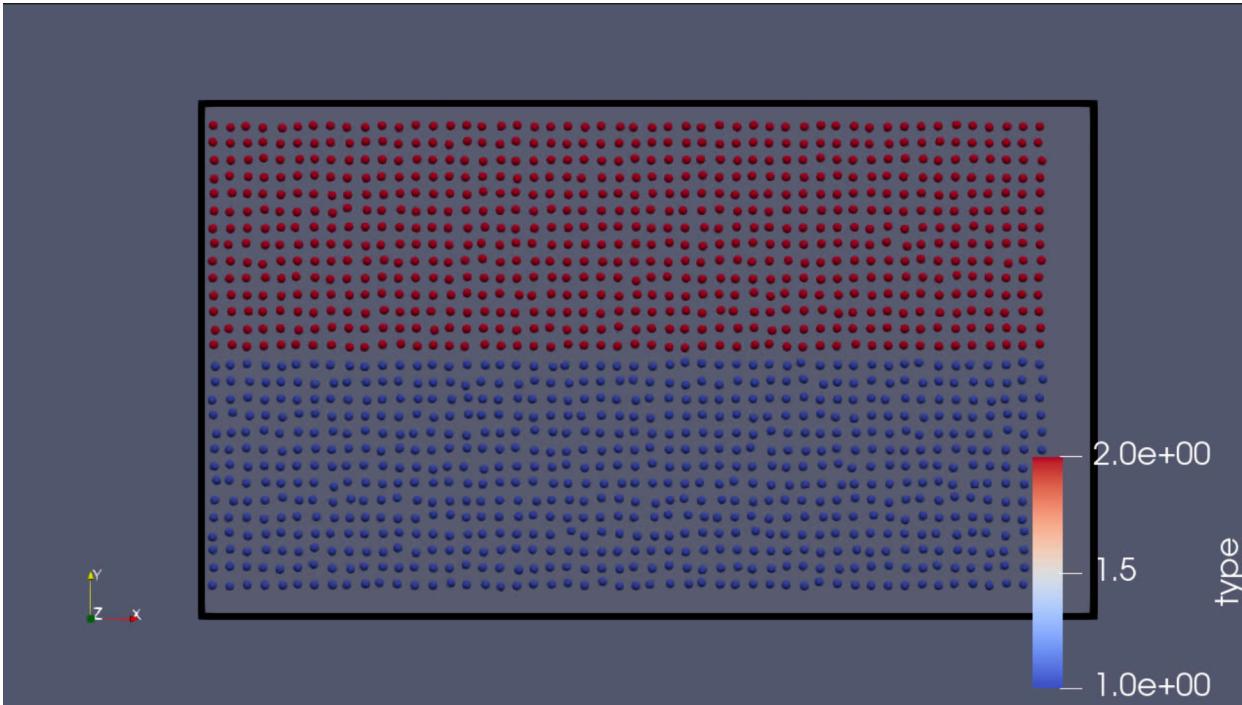
# Boundary conditions

Idea:

1. Temporarily move all particles next to Boundary of the other side
2. Let Neighbouring cells interact

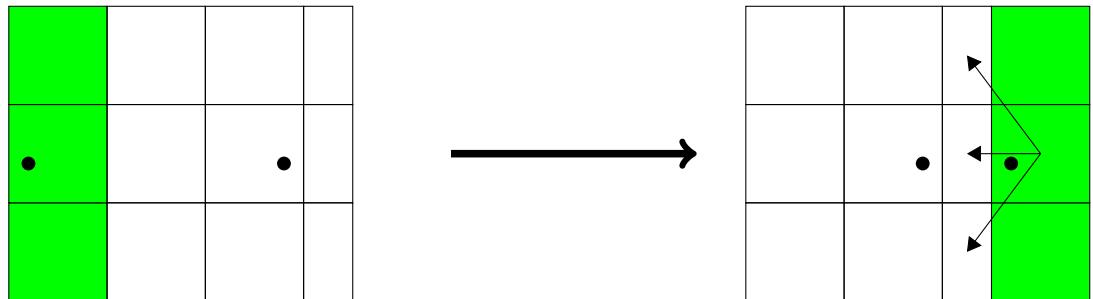


# The result



# The problem

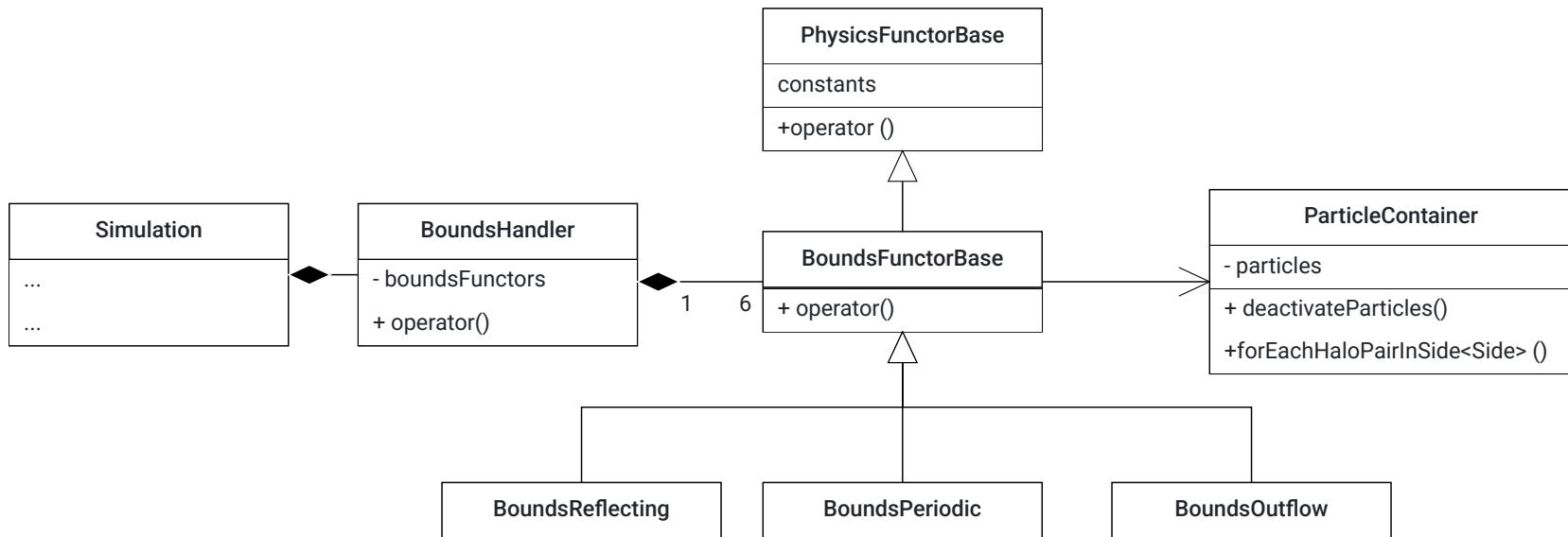
1. Outer boxes may not have the expected sidelengths
2. Interacting with neighbouring cells  $\Rightarrow$  Catching everything in  $r_{cutoff}$



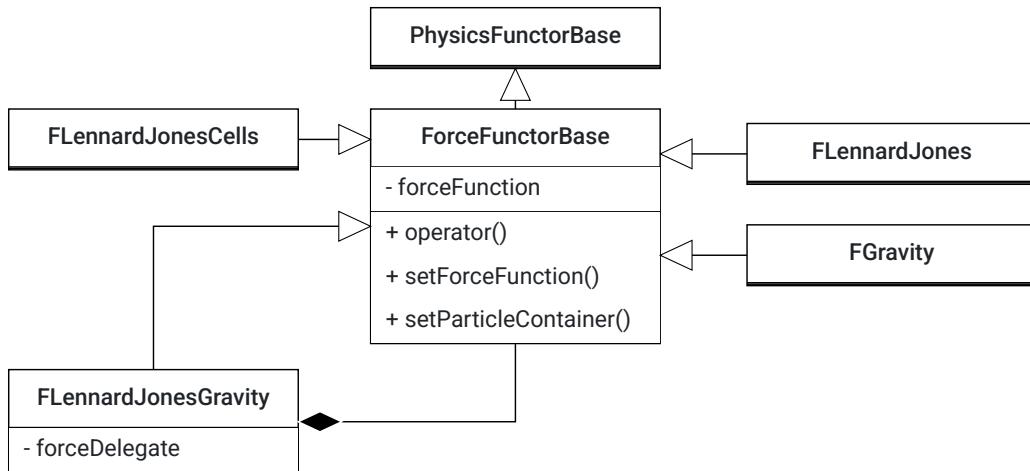
$\Rightarrow$  Interact with one more „Cellblock“ in that direction

# Adding Periodic bounds

This slide should look very familiar to Assignment 3



# Adding Gravitational Force



```
FLennardJonesGravity :: operator ()(){
    forceDelegate->operator ();
    particleContainer.forAllParticles ([]( auto& p){
        p.force[1] += p.m * gGrav;
    });
}
```

# Optimizations 1

As mentioned in Assignment 3 our ParticleContainer does not contain Particle-structs anymore. Keeping the old interface lead to the following method:

```
void ParticleContainer::forAllParticles(void(*function)(Particle &)) {  
    for (unsigned long index: activeParticles) {  
        Particle p;  
        loadParticle(p, index);  
        function(p);  
        storeParticle(p, index);  
    }  
}
```

⇒ rewriting old code where this method got used was a major improvement

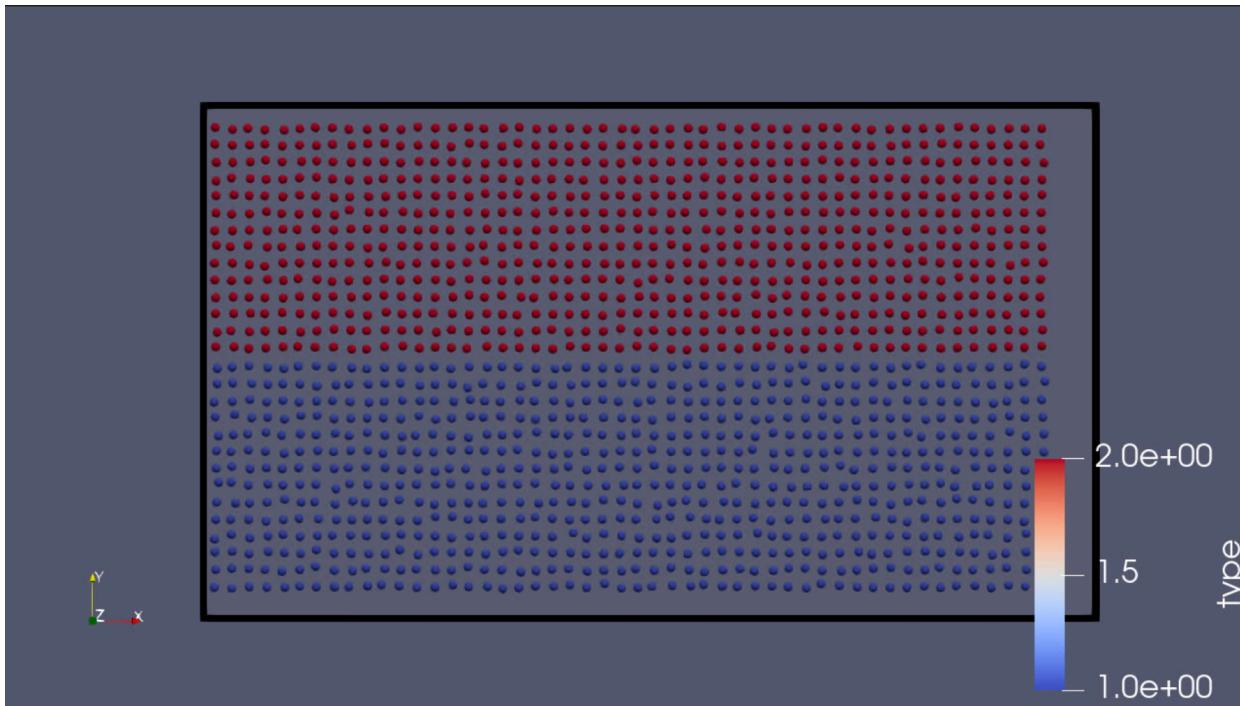
# Optimizations 2

sim::Simulation::runBenchmark	100.0%	0s	MolSim	sim::Simulati...	0x63282
sim::physics::force::FLennardJonesCells::operator()	88.2%	0s	MolSim	sim::physics::...	0x143b94
ParticleContainer::forAllDistinctCellNeighbours<sim::physics::force::FL	81.6%	0.024s	MolSim	ParticleCont...	0x143dd6
sim::physics::force::FLennardJonesCells::operator()(void)::(lambda(s	80.9%	0.640s	MolSim	sim::physics::...	0x143a00
ParticleContainer::VectorCoordWrapper::operator[]	0.4%	0.012s	MolSim	ParticleCont...	0x76be6

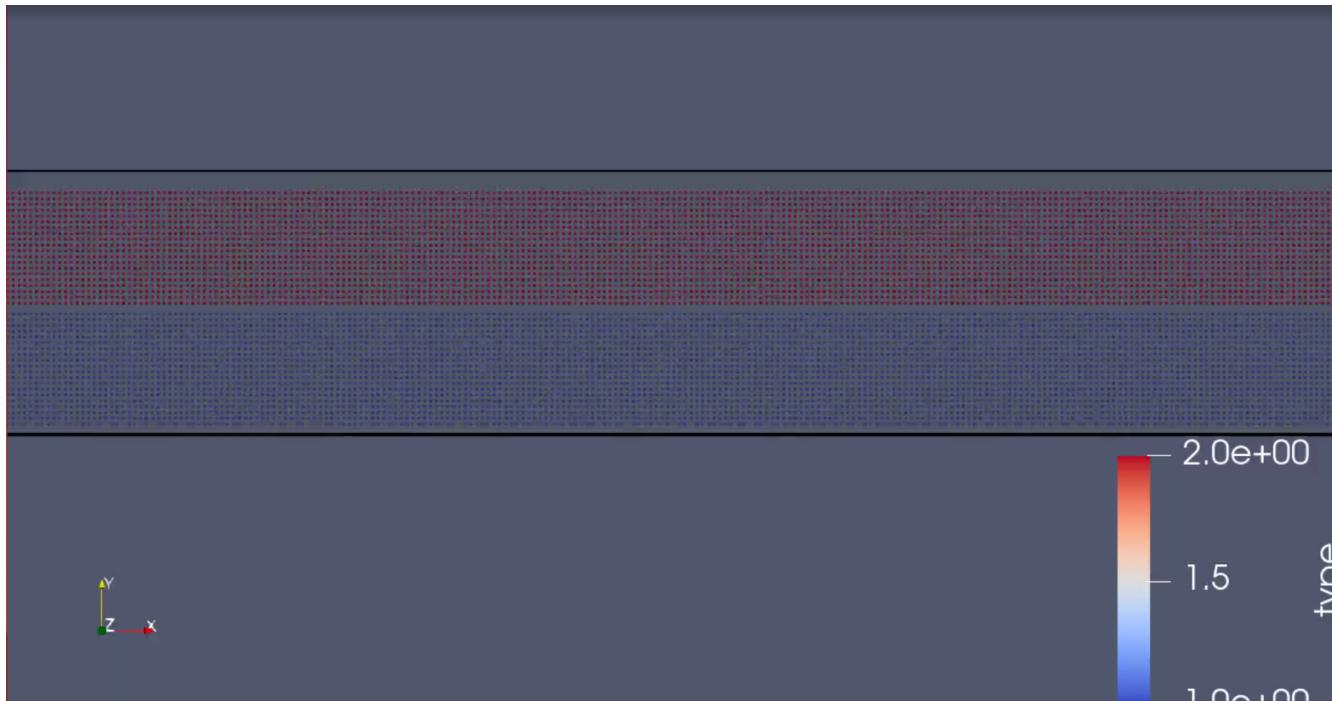
- Force calculation takes a significant portion of CPU time
- Force between two particles in Force-Functors got represented as lambda expression

⇒ Represent force as static function instead

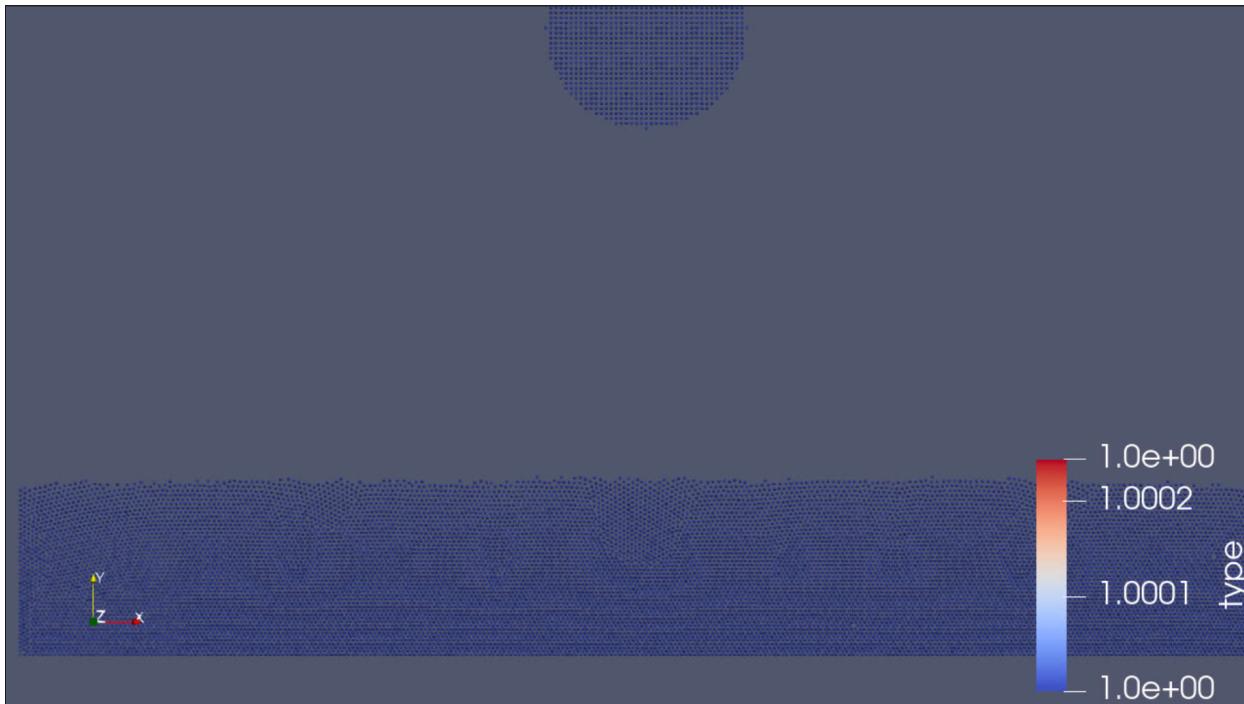
# Small Rayleigh-Taylor instability



# Rayleigh-Taylor instability



# Falling drop



# Serial Benchmarks

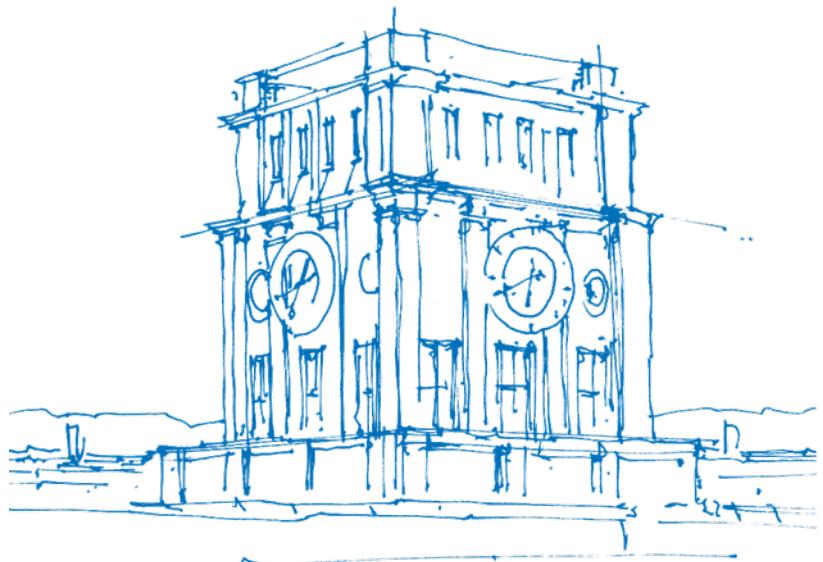
- As already mentioned building with the Intel compiler didn't work even with major time investments
- The option „-slow“ lets the code run in the pre-optimized state

Options	MMU/s Cluster	MMU/s Local
-slow -O2	0.0087	$\geq 1/6 \cdot 0.036 \wedge \leq 1/2 \cdot 0.036$
-O2	0.0087	0.036
-O0	0.0087	0.036

Since these measurements smaller by orders of magnitude than our „private test measurements“ and so close together we assume that something went wrong on the cluster.

# Roadblocks

- Compiling and running jobs on the cluster turned out to be a nightmare
- Intel compiler broke us trying to unbreak him
- Searching for bugs that may or may not be there (bouncy particles in Rayleigh-Taylor)
- Searching for bugs that definitely are there (see Boundary conditions)
- Large time investments in order to get tools to run



TUM Uhrenturm

# Recreating Profiling

1. `mkdir build`
2. `cmake ..`
3. `make ProfileMolSim` or `make CXX_FLAGS+=Dslow -std=c++20"ProfileMolSim`
4. `./ProfileMolSim ../input/[file_you_want_to_profile]`
5. `gprof ProfileMolSim gmon.out > profile-data.txt`