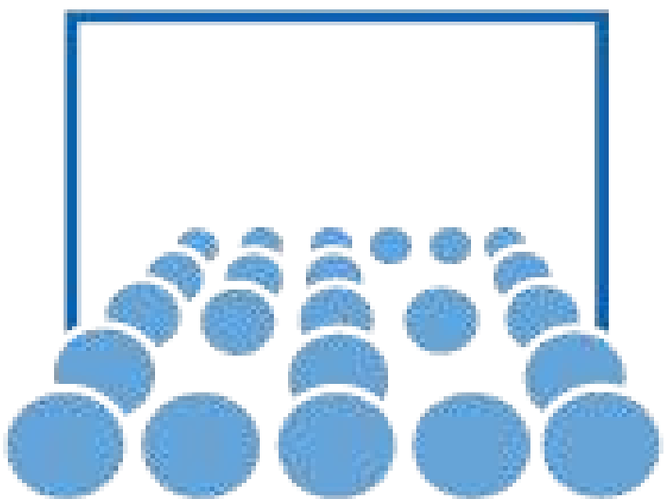
PSE Molekulardynamik

OpenMP, Multicore Architectures, PAPI

Wolfgang Eckhardt

20.12.2013



Outline

- Schedule
- Presentations: Worksheet 4
- OpenMP
- Multicore Architectures
- PAPI
- Membrane, Crystallization
- Preparation: Worksheet 5

Schedule (big meetings)

Date	Worksheet
18.10.2013	
31.10.2013	1
15.11.2013	2
06.12.2013	3
20.12.2013	4
17.01.2014	Questions? DemoDay? World-Record MD Simulation
31.01.2014	5

Presentations: Worksheet 4

Institut für Informatik — TU München WS 2012/13
Scientific **C**omputing in **C**omputer **S**cience
Prof. Dr. H.-J. Bungartz
Dipl.-Inf. W. Eckhardt
Dipl.-Math. A. Breuer

PSE Molekulardynamik
Sheet 4: Thermostats, Rayleigh-Taylor-Instability and "falling drops"

Exercise at 07th December 2012

Task 1 "Thermostats"

In order to keep the simulated material at a constant temperature, as well as to be able to simulate heating or cooling processes, it is neccessary to have a thermostat. The temperature of the phase is determined by the velocity of the particles. The dependency is given by

$$E_{\text{kin}} = \frac{\#\text{dimensions} \cdot \#\text{particles}}{2} k_B T$$

or dimensionless:

$$E_{\text{kin}} = \frac{\#\text{dimensions} \cdot \#\text{particles}}{2} T$$

where E_{kin} can be calculated as the sum of the kinetic energy of all particles:

$$E_{kin} = \sum_{i=1}^{\#\text{particles}} \frac{m_i \langle v_i, v_i \rangle}{2}$$

As seen in the tutorial, the scaling factor can be calculated as:

$$\beta = \sqrt{\frac{E_{\text{kin}}^D}{E_{\text{kin}}}}$$

- Implement such a thermostat.
- The input parameters are
  - initial temperature:
    the initial scaling value for the velocity for the Maxwell-Boltzmann distribution is calculated as:

$$v = \sqrt{\frac{2E_{\text{kin}}}{\#\text{dimensions} \cdot \#\text{particles} \cdot m}}$$

    The initialization of the phase to a given temperature may be moved to the thermostat and be called after all particles have been instantiated.
    For our applications (at least for the moment) it is ok to set $m = 1$ and hardcode it.
    However, the initialization with the Brownian Motion should be optional.
  - the number of timesteps after which the thermostat is applied
- Furthermore for simulations with a different target temperature:
  - target temperature
  - temperature difference, i.e. the step size in which the temperature should be changed
  - the number of timesteps after which the temperature has to be changed
- Adapt the XML and tests accordingly.

Task 2 "Simulation of the Rayleigh-Taylor instability"

The Rayleigh-Taylor instability occurs at the interface of two fluids when a fluid with higher density resides on a fluid with lower density and is subject to gravity.
In this task we will perform the simulation of a Rayleigh-Taylor instability. We will impose reflecting boundary conditions on the horizontal boundaries of the domain and periodic boundary conditions on the vertical boundaries.
The two different fluids are represented by two molecular cuboids.

- Implement periodic boundaries:
  - particles which left the domain at a boundary have to be inserted at the opposite boundary
  - particles from the boundary layer of the opposite boundary have to be inserted into the halo layer of the boundary.
- The fluids are exposed to gravity. Implement a force calculation routine which adds a gravitational force G (along the y-axis) according to

$$G = m \cdot g_{grav}$$

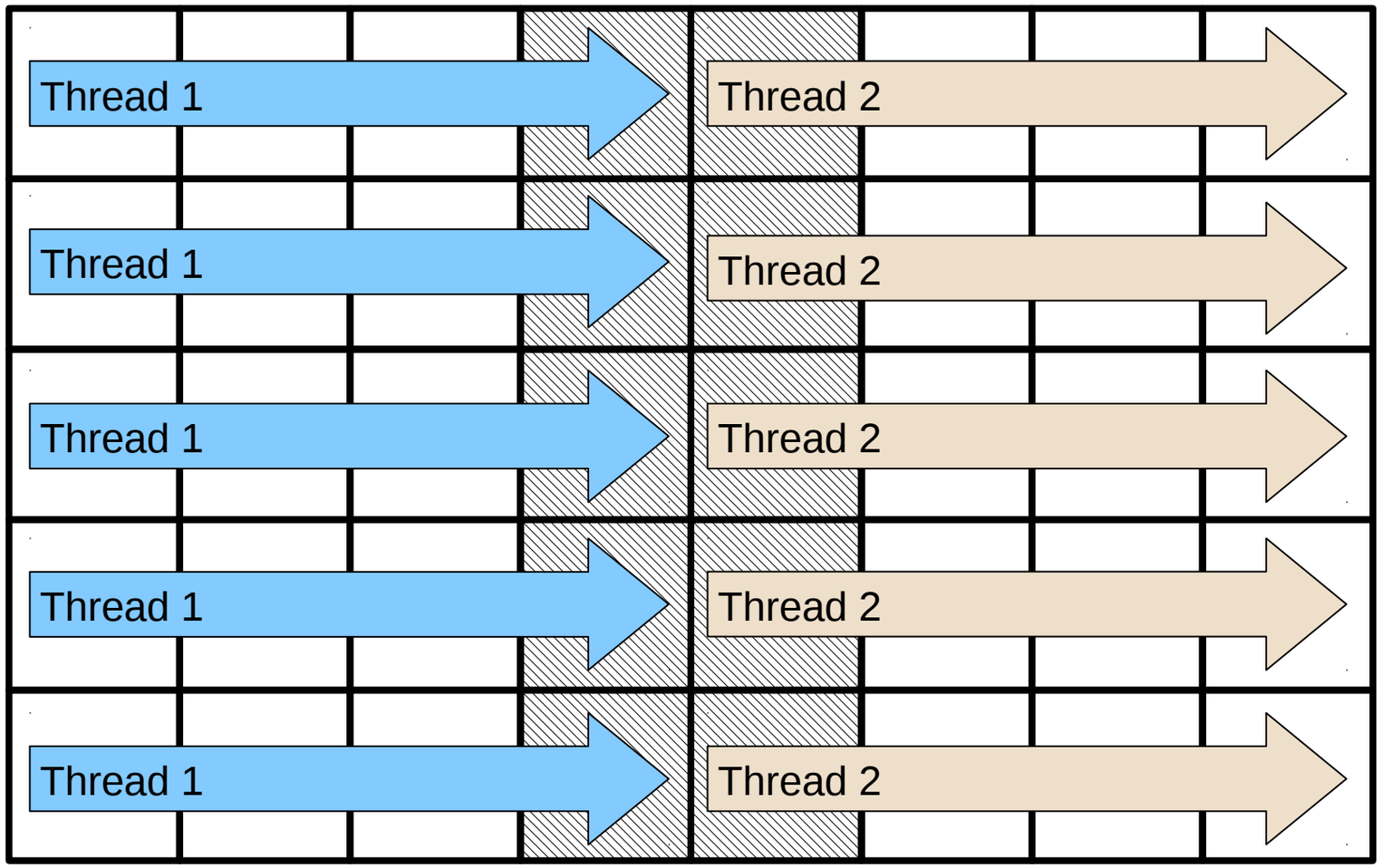
  It should be possible to specify the factor g_{grav}.
- We have to model two different fluids. Extend the molecules so they can be of different type. Especially they may have different

OpenMP

- References
  - https://computing.llnl.gov/tutorials/openMP/
  - http://openmp.org
  - Hager, Wellein: Introduction to High Performance Computing for Scientists and Engineers. CRC Press, ISBN 978-1439811924, 356 pages, July 2010
  - http://icl.cs.utk.edu/papi/

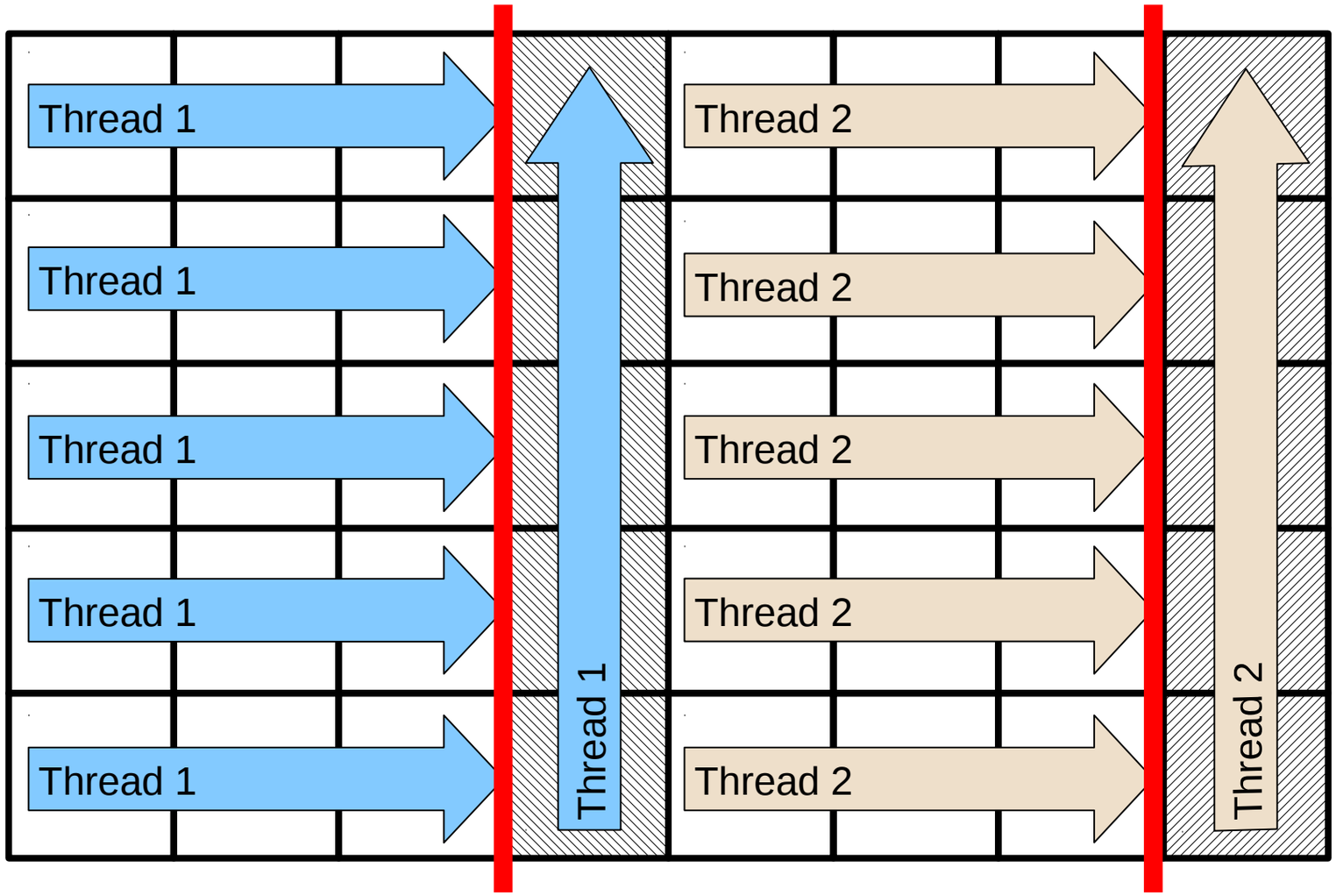
- Make yourself familiar with given hardware and shared memory parallelization
- Parallelize compute intensive parts first
  - Here: Loop for force calculation
- NUMA: Allocate memory in parallel

OpenMP: Linked Cell



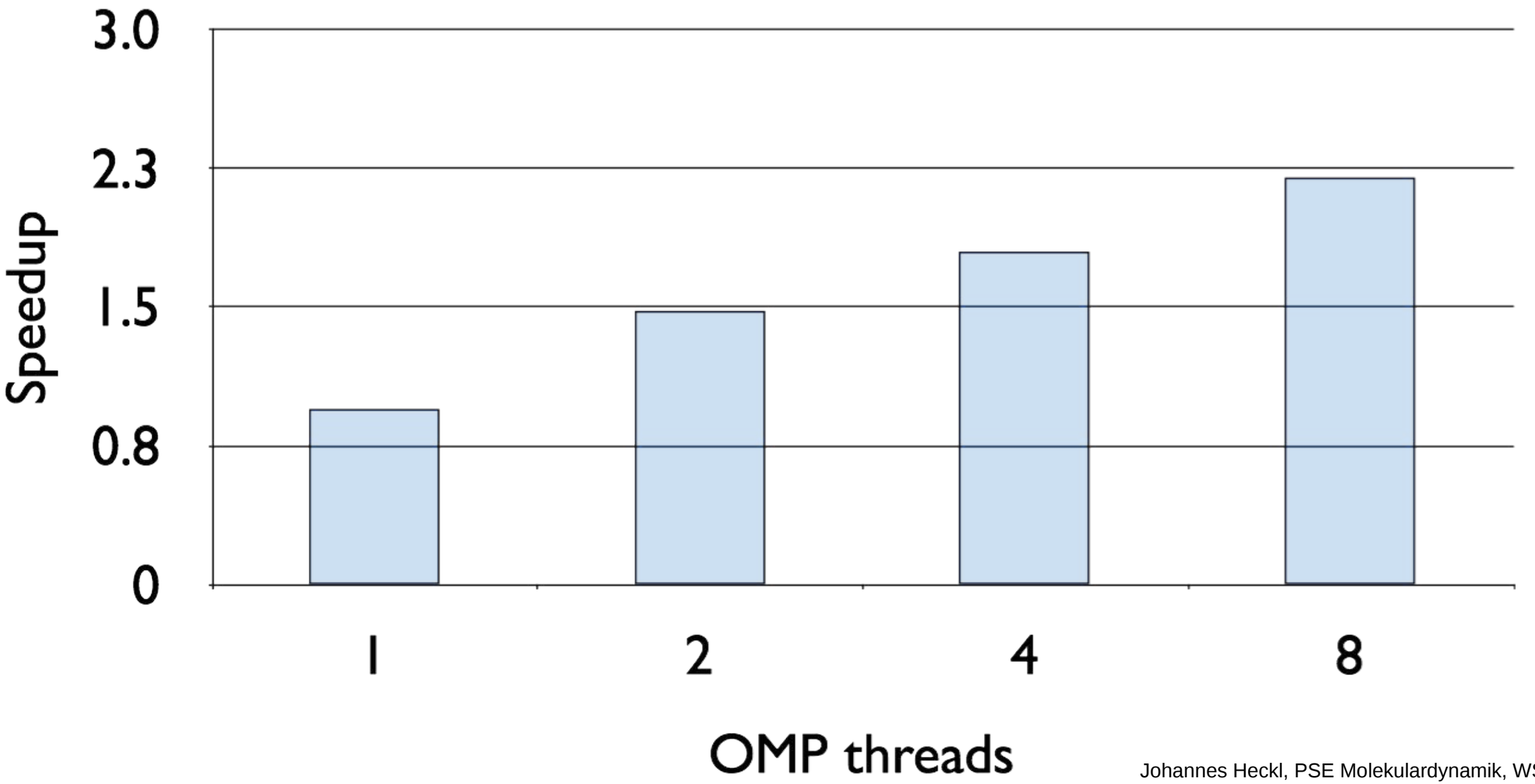
Problem: Synchronisation for boundary cells between Thread 1 and 2, due to Newton 3.

OpenMP: Linked Cell



Possible solution: - Compute inner cells
- Synchronize all threads
- Compute boundary cells

OpenMP: Molecular Dynamics - 2011/2012



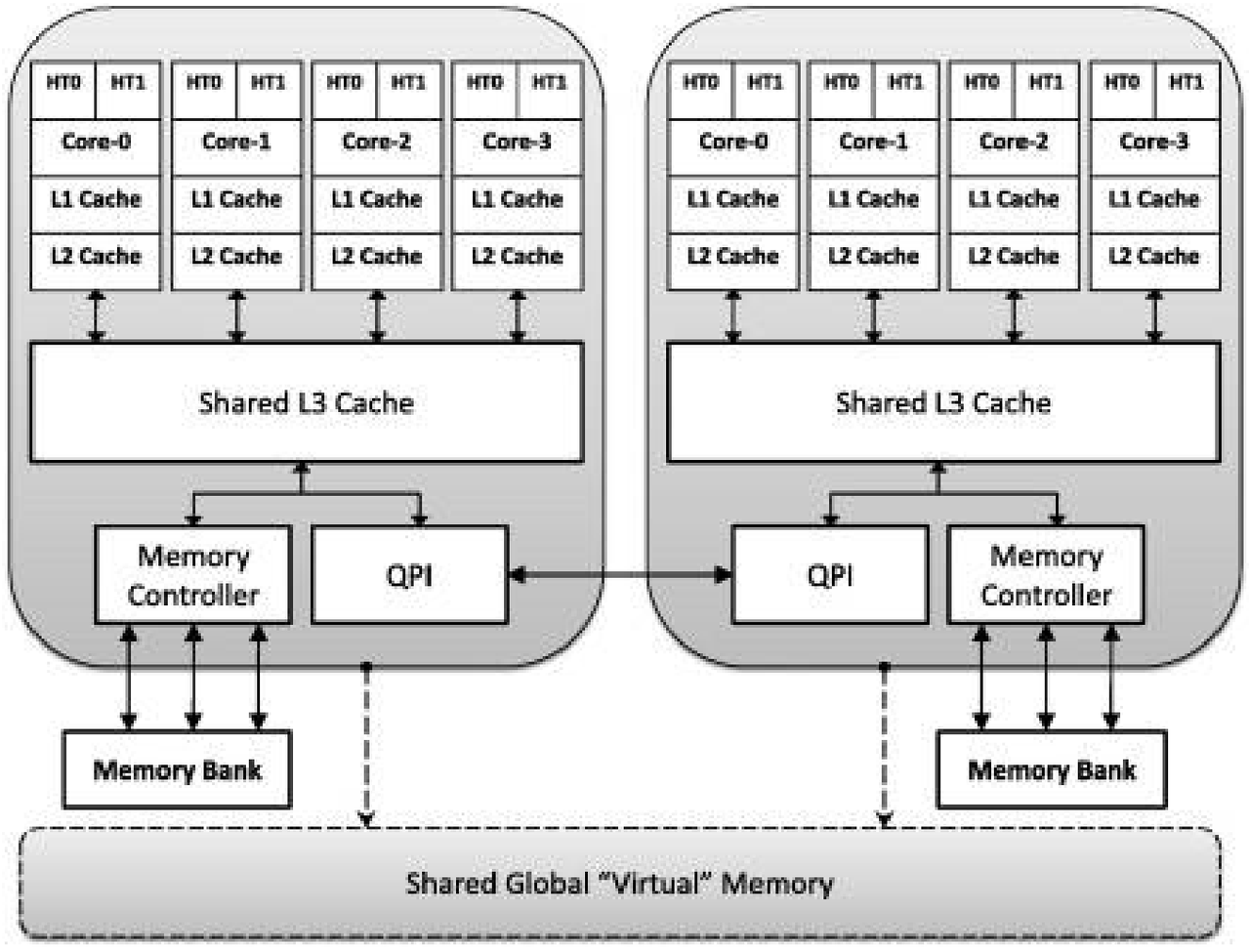
Johannes Heckl, PSE Molekulardynamik, WS11/12

Revision: Linux Cluster / MAC Cluster

- http://www.lrz.de/services/compute/linux-cluster/
- http://www.mac.tum.de/wiki/index.php/MAC_Cluster
- Read usage instructions carefully!
- Here: ICE (optional) and MAC-Cluster AMD-BDZ / Intel SNB Partition

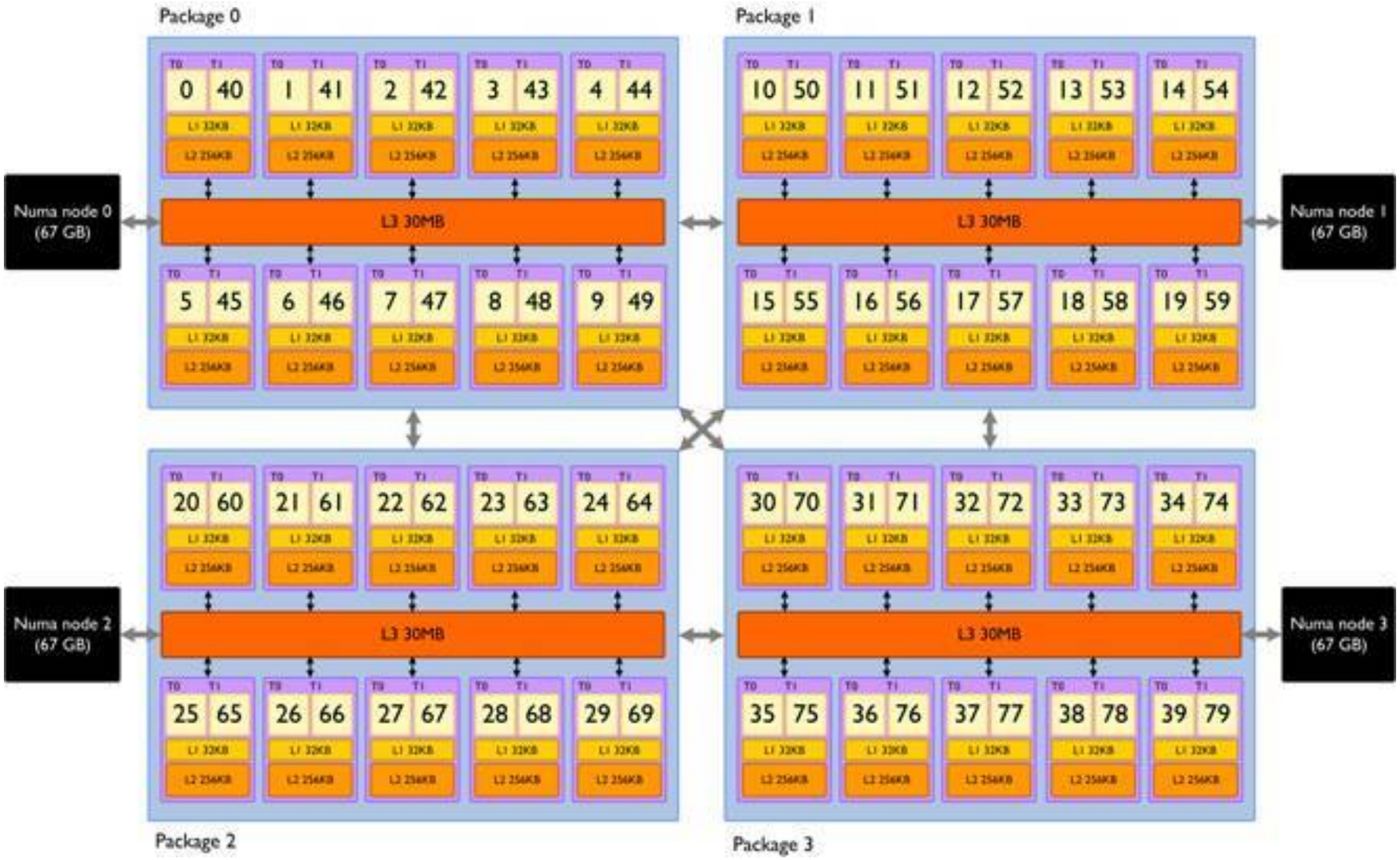
	SGI ICE	MAC-Cluster SNB Partition	MAC-Cluster BDZ Partition
Architecture	Dual socket quad-core Intel Nehalem	Intel SandyBridge-EP Xeon E5-2670 (2.6 GHz)	AMD Opteron 6274 (2.2 GHz)
#Nodes	64	28	19
#Processor cores	512	448	1216
Aggregate peak performance (Tflop/s)	5.2	9.3	10.7
Aggregate memory (TByte)	1.5	3.5	4.9

Multicore Architectures: Nehalem EP



Çatalyüreka et. al, Graph coloring algorithms for multi-core and massively multithreaded architectures
Parallel Computing
Volume 38, Issues 10–11, October–November 2012, Pages 576–59

Multicore Architectures: Westmere EX



LRZ, SuperMIG/SuperMUC Best Practice Guide v0.1
http://www.prace-ri.eu/Best-Practice-Guide-SuperMUC-HMTL

Multicore Architectures: Intel SNB

```
lu32reb2@mac-login-intel:~> salloc --partition=snb --ntasks=1 --cpus-per-task=32 --time=1:00:00
lu32reb2@mac-login-intel:~> srun papi_component_avail
Available components and hardware information.
--------------------------------------------------------------------------------
PAPI Version                  : 5.0.0.0
Vendor string and code        : GenuineIntel (1)
Model string and code         : Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz (45)
CPU Revision                  : 7.000000
CPUID Info                    : Family: 6  Model: 45  Stepping: 7
CPU Max Megahertz             : 2601
CPU Min Megahertz             : 1200
Hdw Threads per core          : 2
Cores per Socket              : 8
NUMA Nodes                    : 2
CPUs per Node                 : 16
Total CPUs                    : 32
Running in a VM               : no
Number Hardware Counters : 0
Max Multiplex Counters        : 64
--------------------------------------------------------------------------------

Compiled-in components:
Name:   perf_events           Linux perf_event CPU counters
  \-> Disabled: /proc/sys/kernel/perf_event_paranoid prohibits using counters
Name:   rapl                  Linux SandyBridge RAPL energy measurements
```

PAPI

PAPI provides the tool designer and application engineer with a consistent interface and methodology for use of the performance counter hardware found in most major microprocessors. PAPI enables software engineers to see, in near real time, the relation between software performance and processor events.

http://icl.cs.utk.edu/papi/

- High- and low-level API to read hardware counters
- Used in task 3 of sheet 5
- Available on the SGI ICE / UltraViolet / MAC-Cluster

```
echo 'loading papi 5.1'
module load papi
gcc $PAPI_INC $PAPI_SHLIB -o test test.c
```

PAPI: Example

$$C = C + AB$$

```
 /* Initialize the Matrix arrays */
for ( i=0; i<INDEX*INDEX; i++ ){
  mresult[0][i] = 0.0;
  matrixa[0][i] = matrixb[0][i] = rand()*(float)1.1;
}

/* Setup PAPI library and begin collecting data from the counters */
if((retval=PAPI_flops( &real_time, &proc_time, &flpins, &mflops))<PAPI_OK)
  test_fail(__FILE__, __LINE__, "PAPI_flops", retval);

  /* Matrix-Matrix multiply */
  for (i=0;i<INDEX;i++)
    for(j=0;j<INDEX;j++)
      for(k=0;k<INDEX;k++)
        mresult[i][j]=mresult[i][j] + matrixa[i][k]*matrixb[k][j];

  /* Collect the data into the variables passed in */
  if((retval=PAPI_flops( &real_time, &proc_time, &flpins, &mflops))<PAPI_OK)
    test_fail(__FILE__, __LINE__, "PAPI_flops", retval);

 printf("Real_time:\t%f\nProc_time:\t%f\nTotal flpins:\t%lld\nMFLOPS:\t\t%f\n",  real_time,
proc_time,   flpins, mflops);
 printf("%s\tPASSED\n", __FILE__);
 PAPI_shutdown();
```

PAPI: Example - Results

$$C = C + AB$$

SGI ICE

```
Real_time:      0.323314
Proc_time:      0.322478
Total flpins:   5401127424
MFLOPS: 16748.810547
PAPI_flops.c       PASSED
```

MAC-Cluster SNB

```
Real_time: 0.005946
Proc_time: 0.007444
Total flpins:      2040789
MFLOPS:            274.164734
test.c   PASSED
```

Alternativ: Likwid (Like I knew what I'm doing)

http://code.google.com/p/likwid/

likwid-topology -g

Preparation: Worksheet 5

Institut für Informatik — TU München WS 2012/13
Scientific **C**omputing in **C**omputer **S**cience
Prof. Dr. H.-J. Bungartz
Dipl.-Inf. W. Eckhardt
Dipl.-Math. A. Breuer

PSE Molekulardynamik
Sheet 5: Parallelization

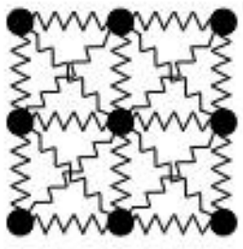
Exercise at 21st December 2012

Remark Hand in configurations, animations, performance measurements and analyses of all simulations.

Task 1 "Simulation of a membrane"

Up to now, we assumed molecules to be spherical and of a fixed internal structure. Thinking about e.g. large biomolecules, this clearly isn't true. In this task we want to simulate a membrane, where the molecules have fixed neighbour relationships, but the structure of the whole membrane is nevertheless flexible. Neigbouring particles interact through a harmonic potential.
To demonstrate the dynamical behaviour, the membrane is subject to gravity, and a force pulls a few molecules of the membrane "upwards" for the first 15.000 timesteps. Self-penetration is avoided by placing a repulsive Lennard-Jones center with each of the molecules.

- Extend the molecule class so that the neighbouring molecules can be stored. Adapt the initialisation so that the molecules are arranged on a rectangular grid, and for each molecule it's direct neighbours along the x- and y-axis and it's diagonal neighbours are set:



- Directly neighbouring particles interact via the harmonic potential, i.e. the force is calculated as

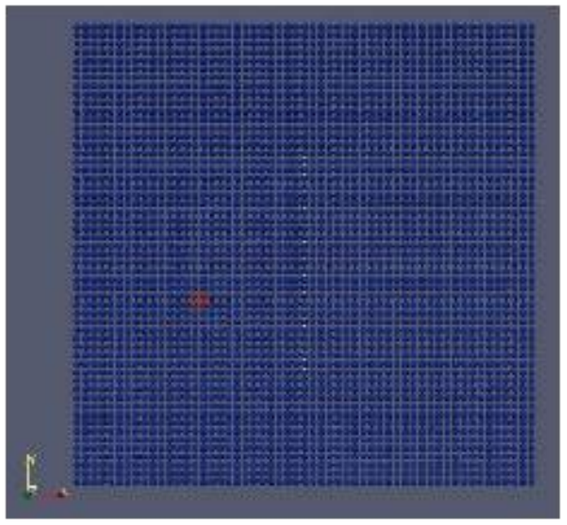
$$F(x_i, x_j) = k \cdot (||x_i - x_j||_2 - r_0)$$

, where k is the stiffness constant and r_0 the average bond length of a molecule pair.

Diagonal neighbours interact via

$$F(x_i, x_j) = k \cdot (||x_i - x_j||_2 - \sqrt{2} r_0)$$

- The calculation of the Lennard-Jones potential is truncated at $2^{\frac{1}{6}}\sigma$, so that only the repulsive part is effective.
- The particles with x/y-indices (17/24), (17/25), (18/24) and (18/25) are pulled by a constant force $F_{\text{Z-UP}}$ "upwards" along the z-axis:



That force is only effective until time $t = 150$.

- Perform the simulation with the following parameters: