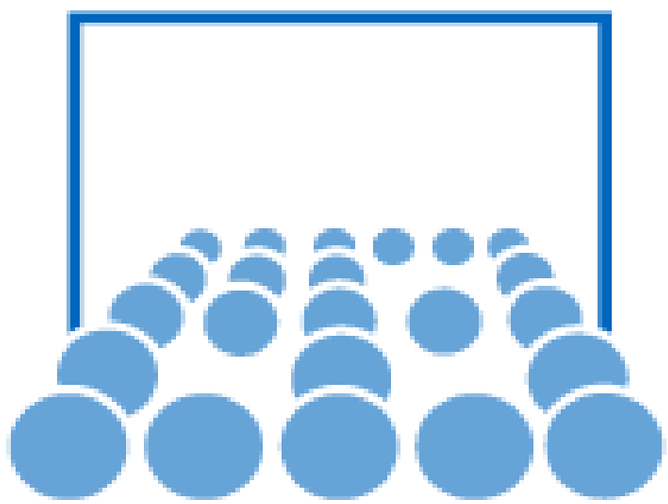
PSE Molekulardynamik

Short range interaction, Linked Cell algorithm

Philipp Neumann, Wolfgang Eckhardt

15.11.2013



Outline

- Schedule
- Presentations: Worksheet 2
- Short range interaction
- Linked-cell algorithm
- Preparation: Worksheet 3

Schedule (big meetings)

Date	Worksheet
18.10.2013	
31.10.2013	1
15.11.2013	2
06.12.2013	3
20.12.2013	4
31.01.2014	5

Presentations: Worksheet 2

Institut für Informatik — TU München WS 2012/13
Scientific Computing in Computer Science
Prof. Dr. H.-J. Bungartz
Dipl.-Inf. W. Eckhardt
Dipl.-Math. A. Breuer

PSE Molekulardynamik
Sheet 2: Collision of two bodies

Exercise at 2nd November 2012

Task 1 „Unit Tests“

Basis for the further development of the simulator will be that it is possible to test the functionality of components before they are completely integrated with the existing code.

- Inform yourself about CppUnit and assertions in C++. Download CppUnit (http://apps.sourceforge.net/mediawiki/cppunit/).
- Build the CppUnit-library according to the instructions (file INSTALL).
- Write a simple UnitTest for the particle container you created in sheet 1, as described in the CppUnit CookBook. Also implement a runner. The unit tests should be executed when your programme is called with the option “–test”.
- Extend the code so that it is possible to specify the name of a single test case which is then being executed.
- In the following parts of the PSE always add assertions and new tests to the components you implement!

Task 2 „Logging “

For debugging it can be very useful to make use of a logging framework. We will use Log4CXX, a c++-port of the java logging framework log4j.

- Make yourself familiar with Log4CXX (http://logging.apache.org/log4cxx/). Download the code of the library from the course website and build it with the autotools. *Remark:*
  - Additionally, you need the libraries apr1-dev and apr-util1-dev installed on your system.
- Think about a reasonable stucture for the loggers and use log-instructions instead of std::cout in your code

  For the configuration use a file in the property format.

Task 3 „Collision of two bodies “

- Implement a particle generator with which we can initialize the simulation. The generator should create a cuboid, where the molecules are arranged in a 3d rectangular grid.

  The cuboid is being described by the following parameters:
  - The coordinate of the lower left front-side corner.
  - Number of particles per dimension: $N_1 \times N_2 \times N_3$.
  - Distance h of the particles (mesh width of the grid)
  - Mass m of the particles
  - Initial velocity v of the particles (3 Components).
  - The mean-value of the velocity of the Brownian Motion. For now it is sufficient if you hard-code it in the code.

  The movement of the molecules should be superposed by Brownian Motion. Use the code from the website to add the thermal friction!

  Specify the parameters via the command line, or read them from a file. It has to be possible to add several cuboids! Moreover, it should be still possible to use a file for input.
- The interaction between two molecules i and j can be described with the help of the Lennard-Jones potential:

$$U(x_i, x_j) = 4 \cdot \epsilon \left(\left(\frac{\sigma}{||x_i - x_j||_2} \right)^{12} - \left(\frac{\sigma}{||x_i - x_j||_2} \right)^{6} \right)$$

.

The resulting force can be calculated as

$$F_{ij} = \frac{24 \cdot \epsilon}{(||x_i - x_j||_2)^2} \cdot \left(\left(\frac{\sigma}{||x_i - x_j||_2} \right)^{6} - 2 \cdot \left(\frac{\sigma}{||x_i - x_j||_2} \right)^{12} \right) \cdot (x_j - x_i)$$

.

Implement a new method for force calculation, which computes the Lennard-Jones force between two molecules! For the moment it’s ok if you hard-code the Lennard-Jones parameters ϵ und σ.

Revision: Lennard-Jones potential

- Interaction between molecules or atoms

$$U(x_i, x_j) = 4\epsilon \left(\left(\frac{\sigma}{\|x_i - x_j\|_2} \right)^{12} - \left(\frac{\sigma}{\|x_i - x_j\|_2} \right)^{6} \right)$$

- Resulting force calculation

$$F_i = \sum_{\substack{j=1 \\ j \neq i}}^{\#particles} F_{ij}, \quad F_{ij} = \frac{24\epsilon}{(\|x_i - x_j\|_2)^2} \left(\left(\frac{\sigma}{\|x_i - x_j\|_2} \right)^{6} - 2 \left(\frac{\sigma}{\|x_i - x_j\|_2} \right)^{12} \right) (x_j - x_i)$$

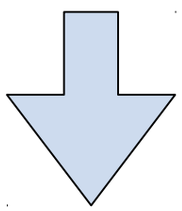
Revision: Lennard-Jones potential

- Dense force matrix

$$\begin{pmatrix} 0 & F_{1,2} & F_{1,3} & \cdots & F_{1,\#particles} \\ F_{2,1} & 0 & F_{2,3} & \cdots & F_{2,\#particles} \\ F_{3,1} & F_{3,2} & 0 & \cdots & F_{2,\#particles} \\ \vdots & \vdots & \vdots & & \vdots \\ F_{\#particles,1} & F_{\#particles,2} & F_{\#particles,3} & \cdots & 0 \end{pmatrix}$$

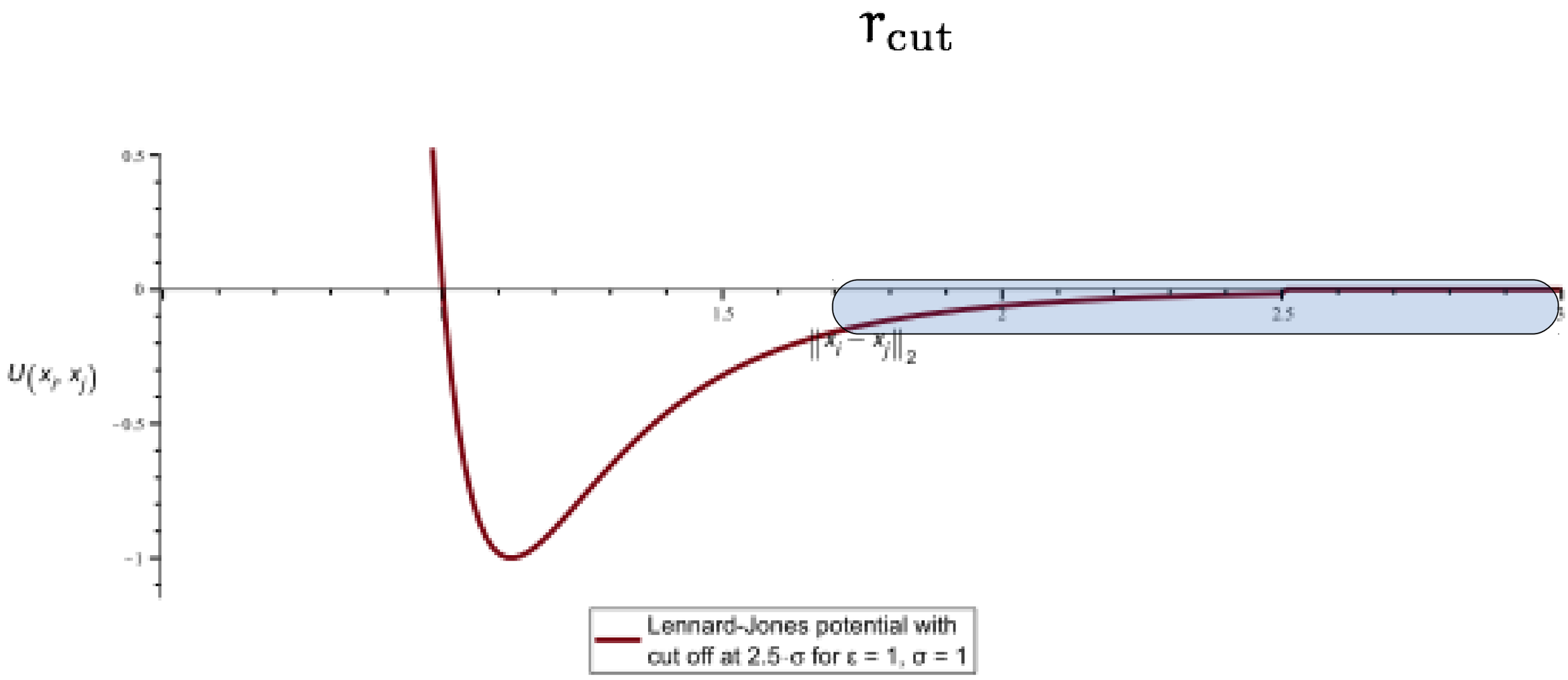
⇒ complexity:

$$\mathcal{O}(\#particles^2)$$

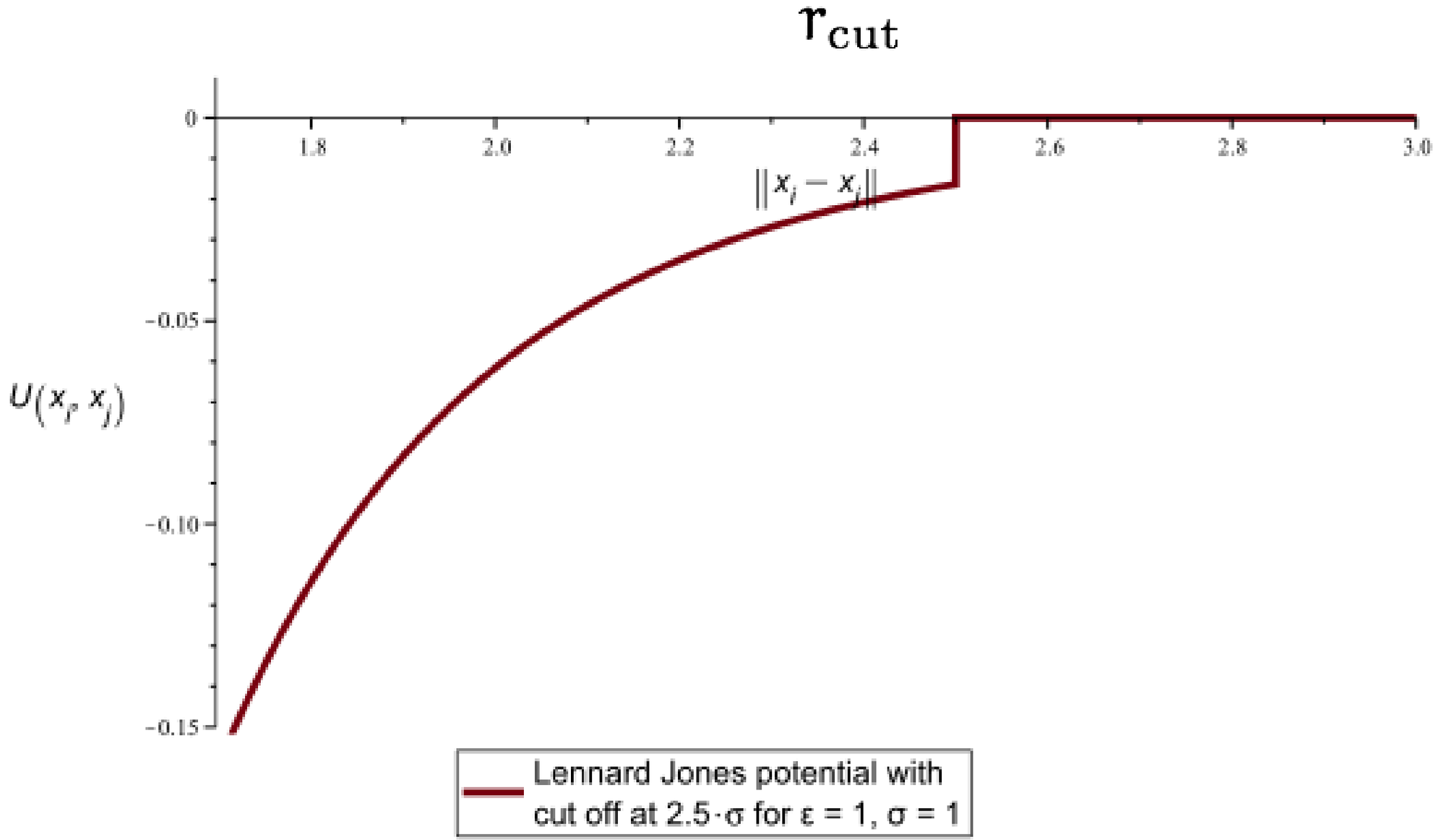


Idea: Neglect near zero forces

Short range interaction

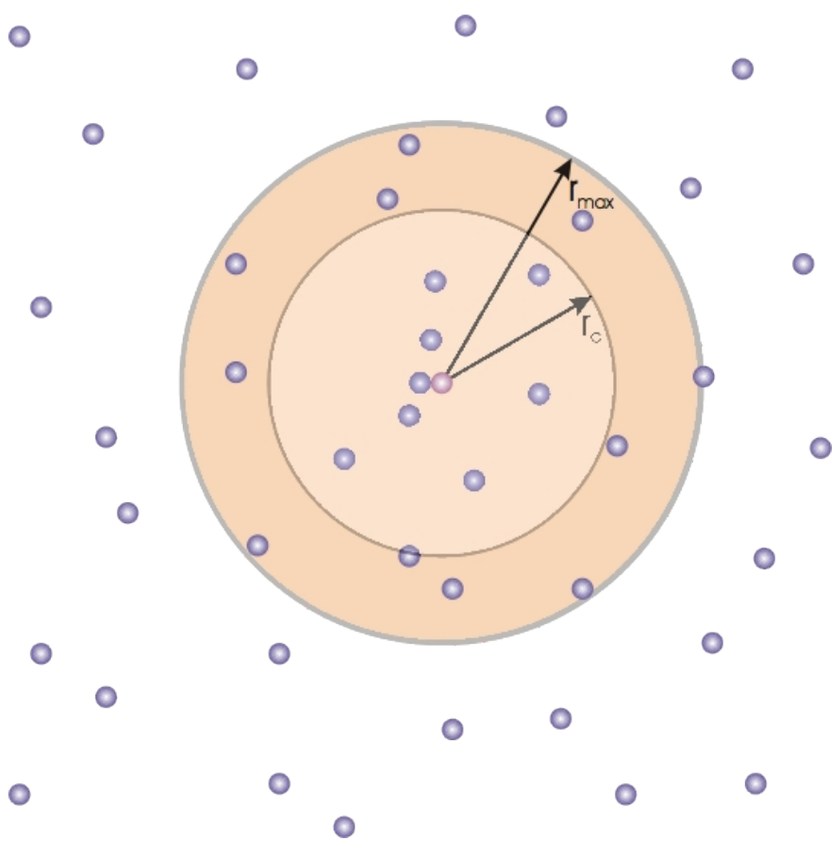


Short range interaction



Short range interaction

$$\mathcal{O}(\#\text{particles})$$

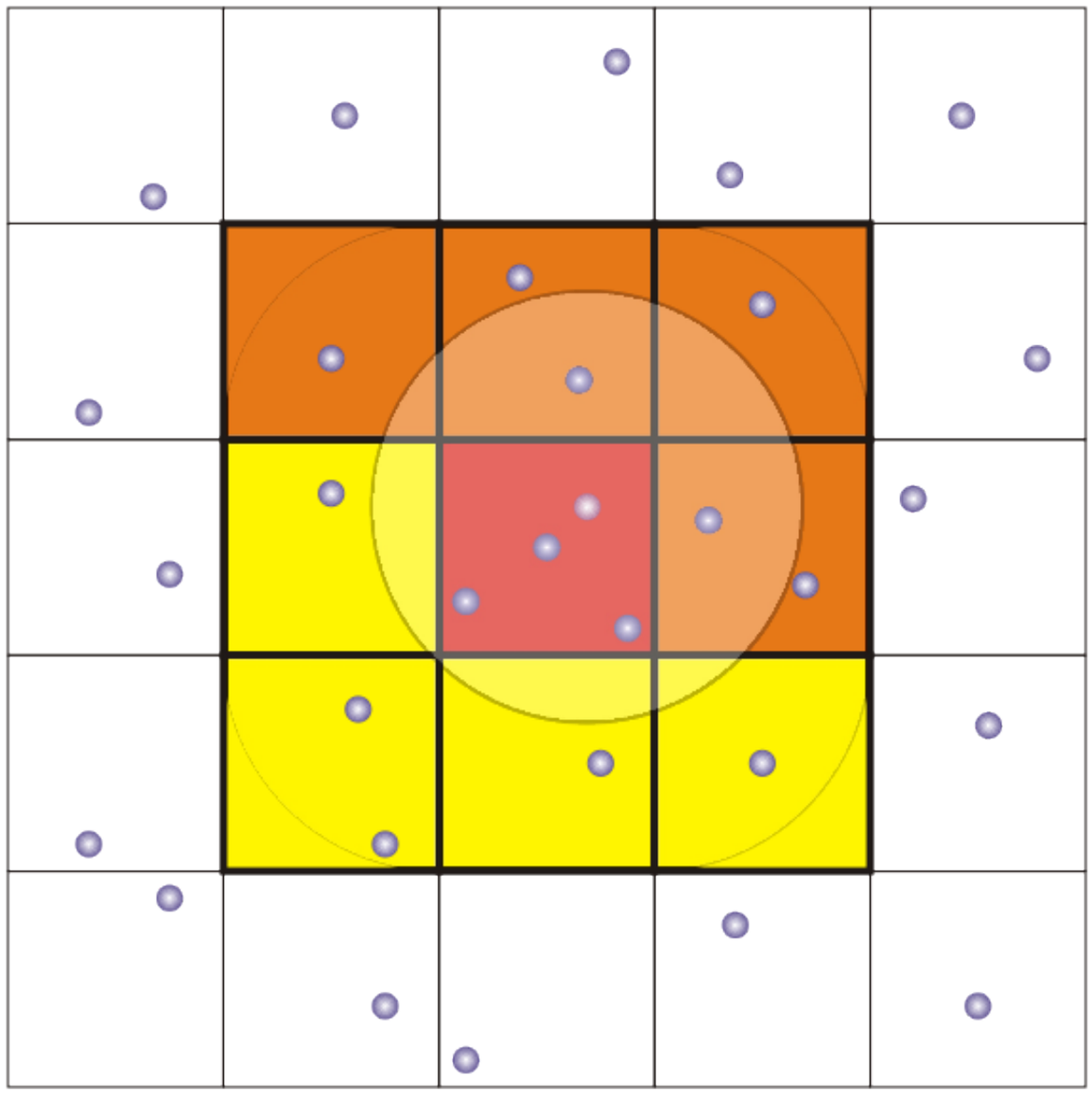


- Each molecule → list of neighboring molecules depending on r_{max}
- List update every $n_{update}(r_{max})$time steps

⇒ buffer with:

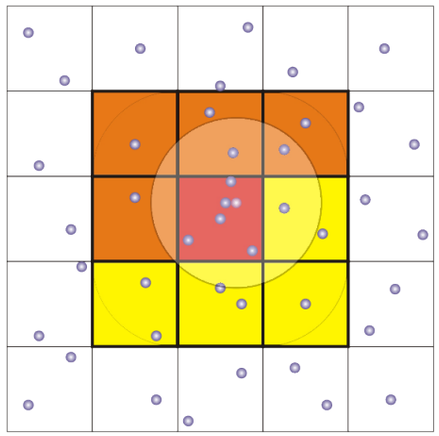
$$r_{max} - r_c > n_{update}(r_{max}) \cdot \Delta t \cdot v_k$$

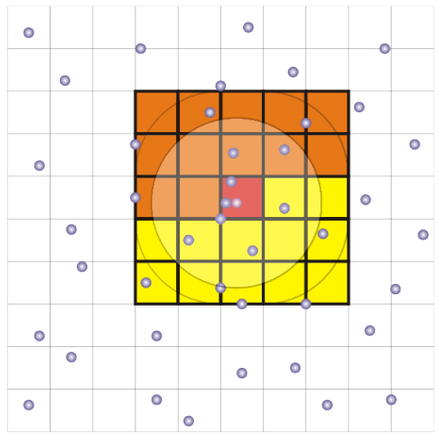
Linked-cell algorithm (classic)

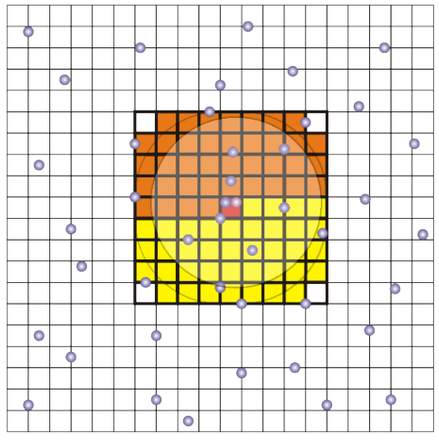


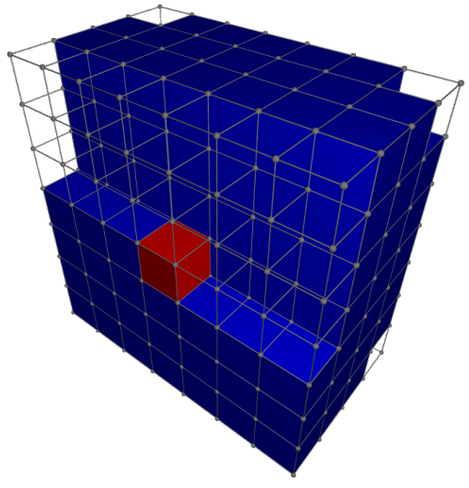
- Introduction of a cubic spatial mesh
- Side length given by cut-off radius
- Newton’s third law: Iterate only over neighbor cells
- natural application of existing particle containers (sheet 3)
- Domain decomposition
  ⇒ parallelization? (sheet 5)

Linked-cell algorithm (outlook)







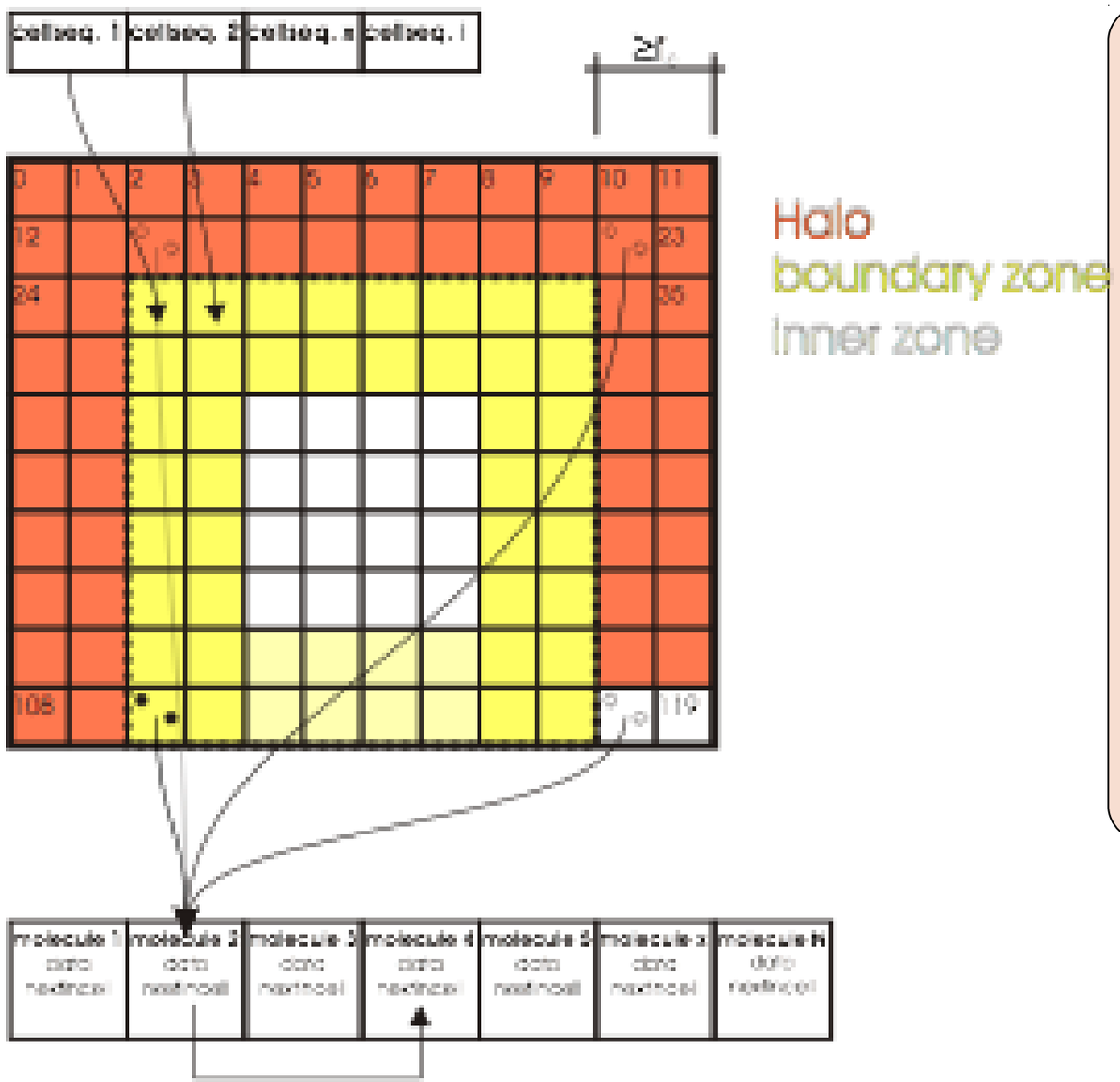


- cubic mesh with side length:

$$\frac{r_c}{\lambda}, \quad \lambda \in \mathbb{R}^+ \vee \lambda \in \mathbb{N}^+$$

- Converges against cut-off sphere for $\lambda \to \infty$

Linked-cell algorithm: Data structure



- linearized array (2D/3D → 1D)
- mesh: 1D vector of cells
- cells: list of molecules

- Halo region: boundary conditions/ghost layer
  - outflow: delete molecules after time step (sheet 3)
  - reflecting boundaries
  - periodic? (sheet 4)
  - distributed memory parallelization?

Preparation: Worksheet 3

Institut für Informatik — TU München WS 2012/13
Scientific Computing in Computer Science
Prof. Dr. H.-J. Bungartz
Dipl.-Inf. W. Eckhardt
Dipl.-Math. A. Breuer

PSE Molekulardynamik
Sheet 3: XML, Linked-Cell-Algorithm and "falling drops"

Exercise at 16th November 2012

Task 1 "XML-Input"

As the configuration of a simulation is getting more complex, we will use xml input files.

- Download the tool Codesynthesis XSD (`http://www.codesynthesis.com/products/xsd/`).
- Make yourself familiar with the tool!
  (Have a look at the example "Hello" as it is contained in the examples-folder, and as described in the Getting Started Guide.)
- Develop an input file reader using the the C++/Tree-Mapping. It should be possible to specify at least the following things:
  - Base name of the output files
  - Write frequency of the output files
  - delta_t, t_{end}, etc.
  - Input files and specification of the cuboids
- Encapsulate the use of the XML-component in a suitable way! Also take care of a testable design.

Note:

- You can find more information about XML and XML Schema at
  - XML: `http://www.xml.com/pub/a/98/10/guide0.html`
  - XSD: `http://www.w3schools.com/Schema/schema_intro.asp`
- Note that the XML DOM object may be constructed by a method taking std::istream as a parameter.
- Note that the parser tries to validate against an .xsd-file by default. Thus take care to specify the .xsd document correctly, or switch validation off by passing the flag xml_schema::flags::dont_validate to the parser.

Task 2 "Linked-Cell Algorithm"

As you experienced on the last sheet, the computational effort for a naiv molecular-dynamics simulation is pretty high ($\mathcal{O}(n^2)$). Thus we simplify the model and make use of the linked-cell algorithmus, as discussed in the lecture.

- Implement a new particle container, realizing the linked-cell algorithm. However, it should still be possible to use the existing implementation.
- Adapt the XML format. Especially, it is now necessary to specify the size of the domain (that is the size of the inner area including the boundary of the domain, but not the halo particles), and the cutoff-radius.
- The container has to offer access to the halo particles as well as to the boundary particles. It has to support the deletion of the particles in the halo.
- Perform the simulation "Collision of two bodies" once again, now with the following parameters:

$x_1 = \{20, 20, 0\}$	$x_2 = \{70, 60, 0\}$
$v_1 = \{0, 0, 0\}$	$v_2 = \{0, -10, 0\}$
$N_1 = \{100, 20, 1\}$	$N_2 = \{20, 20, 1\}$
	$h = 2^{1/6} \cdot \sigma \approx 1.1225$
$\epsilon = 5$	$\sigma = 1$
$m = 1$	
$\delta t = 0.0005$	$t_{end} = 20$
Size of the domain: L = {180, 90}	$r_{cutoff} = 3.0$

- How does the runtime per iteration behave for the two different implementations? Compare the times for a simulation of a 2d-square with e.g. 1000, 2000, 4000 and 8000 molecules

  Embedd a graph comparing the two implementations in the doxygen documentation.