Technische Universität München
Lehrstuhl für Informatik V
Univ.-Prof. Dr. M. Bader
Dipl.-Tech. Math. P. Neumann
Dipl.-Inf. W. Eckhardt

SS 2013
Sheet 11
15.07.2013

Scientific Computing II

Molecular Dynamics

Exercise 16: Linked Cells algorithm

Assume we simulate a molecular dynamics scenario with N molecules. If we explicitely compute the forces between all pairs of molecules, O(N^2) operations are necessary. For short-range potentials, we neglect interactions between particles that have a mutual distance bigger than a certain cut-off radius. This reduces the number of required operations to O(N).

(a) Someone tries to convince you that this is not true:

Assume that a simululation with N molecules requires C operations for the force calculations. Double the number of molecules in the domain. Then we of course have to compute forces for twice as many molecules. But, in addition, the number of molecules within the cut-off radius of a certain molecule doubles, too. Thus, we need 4C operations, which means that the number of operations behaves like O(N^2).

Why is that argumentation wrong?

(b) In 3D, apart from the cell itself, 26 additional cells have to be examined (8 cells in 2D). By reducing the size of the cells, more cells have to be used, but the volume covered by these cells is smaller. As the covered volume corresponds to the number of distance calculations, smaller cell sizes can increase the performance. Calculate the covered volume for $l = rc$, $l = \frac{rc}{2}$, $l = \frac{rc}{4}$ and $l \to 0$ in 2D and 3D.

	$l = rc$	$l = \frac{rc}{2}$	$l = \frac{rc}{4}$	...	$L \to 0$
$2D$					
(% unnecessary)					
$3D$					
(% unnecessary)					

Solution:

(a) The fault is that with doubling the number of molecules in a domain the scenario changes completely, as we now deal with matter of double the density. Thus doubling the problem size doesn't correspond to simply doubling the number of molecules, but rather to doubling the size of the domain, with constant density. In that case the algorithm really needs only twice the number of calculations.

(b) a) $l = rc$: In this case, we have to consider only the directly neighbouring cells:

	2D: 9 cells	3D: 27 cells
A_{cells}	$9rc^2$	$27rc^3$
$A_{\text{in cutoff}}$	πrc^2	$\frac{4}{3}\pi rc^3$
$A_{\text{in cutoff}}$ / A_{cells}	$\frac{\pi}{9} \approx 35\%$	$\frac{4\pi}{81} \approx 16\%$
$\Rightarrow$ unneccessary:	$\approx 65\%$	$\approx 84\%$

b) $l = \frac{rc}{2}$: In this case, we have to consider two neighbouring cells in each direction, thus five cells per dimension:

	2D: 25 cells	3D: 125 cells
A_{cells}	$25(\frac{rc}{2})^2 = \frac{25}{4}rc^2$	$\frac{125}{8}rc^3$
$A_{\text{in cutoff}}$	πrc^2	$\frac{4}{3}\pi rc^3$
$A_{\text{in cutoff}}$ / A_{cells}	$\frac{4\pi}{25} \approx 50\%$	$\frac{32\pi}{375} \approx 27\%$
$\Rightarrow$ unneccessary:	$\approx 50\%$	$\approx 73\%$

c) $l = \frac{rc}{4}$: Here we have to take 9 cells in each dimension into consideration. But now we can get rid of some cells. In 2d, e.g. the smallest distance possible between a particle in the middle cell to a particle in a corner cell is $\sqrt{2(\frac{3}{4}rc)^2} = 1.06rc$. Thus we don't have to search the corner cells for particles within the cutoff-radius.

In 3d, searching all cells along the edges is unneccessary. Moreover we can save 4 more cells per surface of the cube.

	2D: 77 cells	3D: 613 cells
A_{cells}	$77(\frac{rc}{4})^2 = \frac{77}{16}rc^2$	$\frac{613}{64}rc^3$
$A_{\text{in cutoff}}$	πrc^2	$\frac{4}{3}\pi rc^3$
$A_{\text{in cutoff}}$ / A_{cells}	$\frac{16\pi}{77} \approx 65\%$	$\frac{256\pi}{3\cdot 613} \approx 44\%$
$\Rightarrow$ unneccessary:	$\approx 35\%$	$\approx 56\%$

Here is an overview of all the results:

	$l = rc$	$l = \frac{rc}{2}$	$l = \frac{rc}{4}$	$\ldots$	$L \to 0$
$2D$	9	6.25	4.81		3.14
(% unneccessary)	65%	50%	35%		0%
$3D$	27	15.63	9.58		4.19
(% unneccessary)	84%	73%	56%		0%

Exercise 17: Parallel Linked Cells algorithm

Now consider the simulation of a nucleation process. The simulation is started with a million molecules in gas phase, so the distribution of the molecules in the domain is homogenous. Over time, small droplets will evolve and the distribution of the molecules will become increasingly inhomogenous.

(a) For the simulation we will use the parallel linked cell algorithm. Describe a scalable implementation of the algorithm.

(b) Initially, every computer is assigned an equally sized part of the domain. Then, during the run of the simulation, a load imbalance will occur. What are suitable criteria for repartitioning?

Solution:

(a)
- The subdomain of each process is surrounded by a layer of halo-cells.
- Communicate boundary particles along spatial dimensions, e.g. first in x-, then in y-, last in z-direction. In 3D, each process has then only 6 instead of 26 communication partners.
- Use Non-blocking, overlapping MPI Send/Receive to allow for maximum parallelism:
  - Start receive operation for both neighbors along spatial dimension
  - Start send operation
  - Wait until all operations finished

(b)
- load imbalance per process (i.e. if load imbalance is small, no repartitioning is needed).
- Load per cell (in order to find partitions with equal load).
- Communication cost required by a given partitioning (e.g. if you cut through a droplet, many particles need to be communicated).

Exercise 18: Complexity of the Barnes-Hut method

In the lecture the costs for the d-dimensional Barnes-Hut-Method were given as $O(\theta^{-d} N \log N)$. First derive the costs for the twodimensional Barnes-Hut-Method. Then explain descriptively (without proof), why the formula is also correct for 3d.

Solution:

The "outer loop" of the Barnes-Hut-algorithm iterates over all particles and calculates the force effective on every particle. As there are N particles in total, for each of those particles the costs have to be $O(\theta^{-d} \cdot \log N)$. For the given complexity of $O(\theta^{-d} \cdot N \cdot \log N)$ we assume that the tree is not degenerated, thus the tree has $O(\log N)$ levels. If we can show, that at each level the costs are $O(\theta^{-d})$, the formula is proven valid.

We start out from the twodimensional case. First we consider an arbitrary level and determine the number of nodes (i.e. particles of pseude-particles) we have to consider.Those are exactly the nodes for which the parent node didn't fulfill the θ-criterion. For the parent node it has to hold:

$\frac{diam}{|distance|} > \theta$. Thus it follows for the distance:

$$|distance| < \frac{diam}{\theta} \tag{1}$$

We will now try to determine an upper bound for the number of parent cells which fulfill that condition. Therefor we consider only one coordinate axis first.

Not considering the absolute value, (1) can be rewritten as $-\frac{diam}{\theta} < distance < \frac{diam}{\theta}$. Thus the "area" of the distance is $2 \cdot \frac{diam}{\theta}$. As one node covers an area with the diameter *diam* there can be only

$$\frac{2 \cdot \frac{diam}{\theta}}{diam} = \frac{2}{\theta}$$

nodes next to each other, for which the θ-criterion does not hold. In the second axis there are just as many. Thus there are

$$\left(\frac{2}{\theta}\right)^2 = 4 \cdot \theta^{-2}$$

nodes in total, which don't fulfill the θ-criterion. Those are the parent nodes we have to consider at the current level. That number still has to be multiplied by 4, what doesn't change the order.

Thus the maximum computational costs on each level are $O(\theta^{-2})$. As there area $\log N$ levels in total and N particles, the total computational costs sum up to $O(\theta^{-2} \cdot N \cdot \log N)$.

For three dimensions it is easy to see that there are not $c \cdot \theta^{-2}$ any more for which the θ-criterion is fulfilled, but only $c \cdot \theta^{-3}$, as we are dealing now with a cubic area.