GPU, Multi/Many Core Computing I: Introduction to HPC

Exercise 2: cell lists

FIXES

- Fixed initial conditions for LJ and:
  - timings in Table 1 changed
  - σ for each I.C. changed
- Changes in exercise sheet
  - cellsize $\geq$ cutoff can be assumed
  - supporting structure $\{(s_i,e_i)\}$ should handle empty cells
  - Q3: Eq. (5) is $1/\varepsilon$^2 and not $1/\varepsilon$^d
  - Q3: Grid spacing $\delta x \geq 1/5$ needed
  - Q3c: compare with given exact solution at u(...,t=0.1)
  - Q4e: missing def. of uniform $x_j = j*\delta x$, $\delta x = 2\pi/N$

Cell list algo

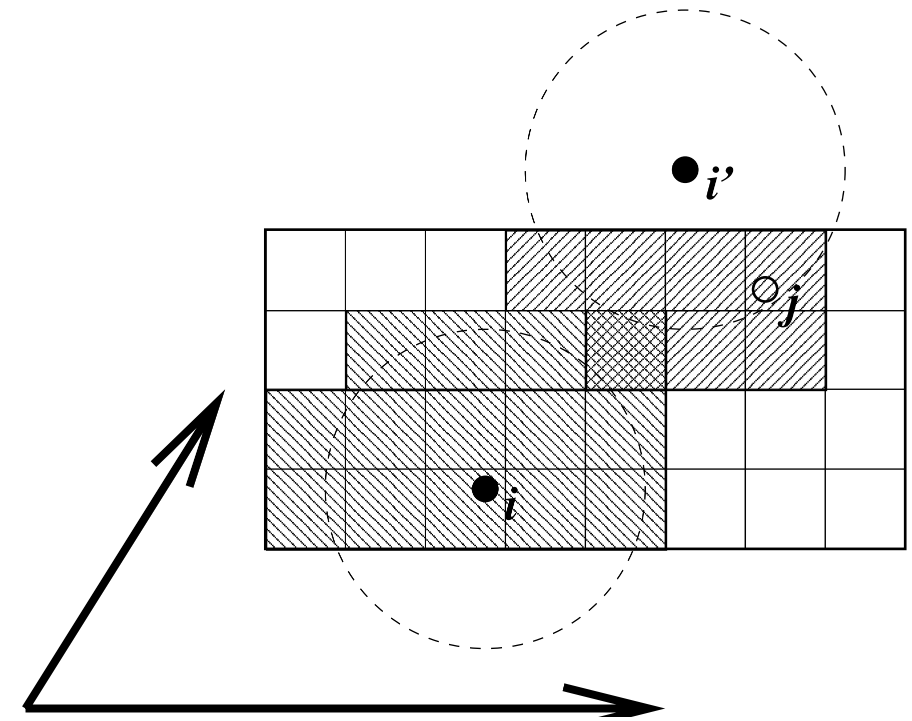
- CPU and GPU implementation needed (for Q2 and Q3)
- Steps:
  - Construct cell data => keys
  - Sort keys => see example sort options in code skeleton
  - Reorder data and create supporting structure
  - Traverse and compute pairwise forces
    - careful with periodic boundaries!
- Reminder: **Test your code**

Cell List (the first three steps)

Sort

- Efficient implementation with radix sort => O(cN)
- Sample code comparing sort algorithms:
  - N Random <key,value> pairs (key in [0,maxkey], value = 0..N-1)
  - sort(): IN: keys, values => OUT: Sorted pairs s_keys, s_values
  - TestSorting.h: simple, inefficient implementation with STL
  - TestSortingIPP.h: using Intel Performance Primitives
  - CUDA/...: using thrust library for CUDA
  - OpenCL/...: adapted from libCL (speed depends on maxkey)

GROMACS



src: GROMACS manual

- "Cell list" stage:
    - Generate cell list
    - Update neighbor lists (= Verlet lists)
    - Not needed every time step
    - Given timings assume 1 neighbor list creation and 1 force evaluation at each iteration for comparison
- Comparison should consider that
    - machine is different
    - algorithm is different => focus on "Total Time"

PSE (theory)

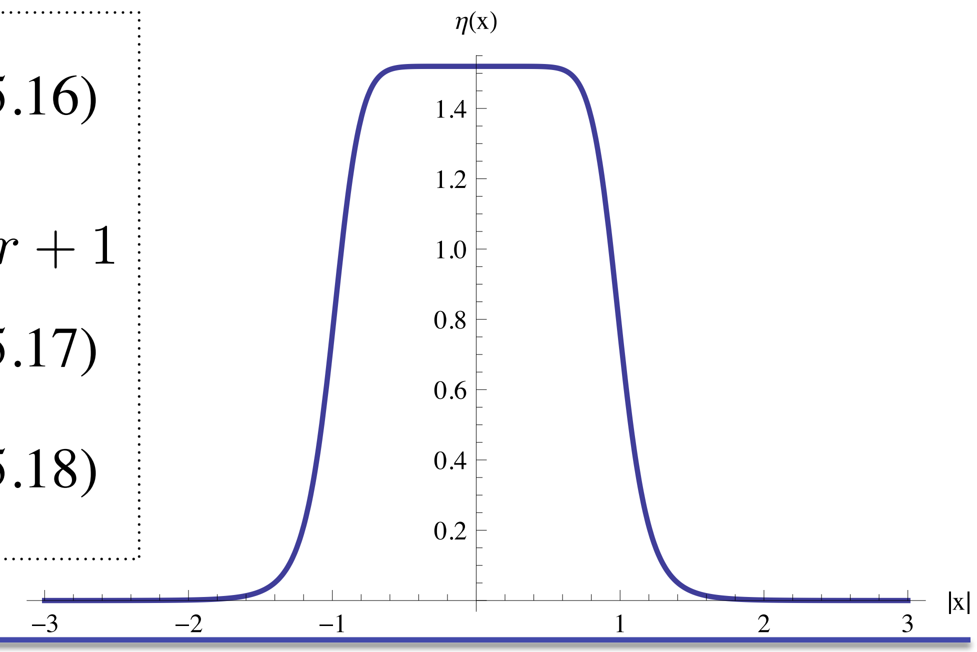
- Given η(x) is meant to work in 3D only and has r = 2
- Solve $\epsilon^{-2}\int_{\mathbb{R}^d}(u(\boldsymbol{y})-u(\boldsymbol{x}))\,\eta_\epsilon(\boldsymbol{y}-\boldsymbol{x})\,d\boldsymbol{y} = \nabla^2 u(\boldsymbol{x}) + \mathcal{O}(\epsilon^r)$ **with** $\eta_\varepsilon(\mathbf{x}) = \frac{1}{\epsilon^d}\eta\left(\frac{\mathbf{x}}{\epsilon}\right)$
  - use Taylor expansion of u(**y**) around u(**x**)
  - Integration solved with quadrature (rectangle rule)
- Requirements on η:

$$\int x_i x_j \eta(\boldsymbol{x})\,d\boldsymbol{x} = 2\delta_{ij} \qquad \text{for } i,j = 1,2,3 \qquad (5.16)$$

$$\int x_1^{i_1} x_2^{i_2} x_3^{i_3} \eta(\boldsymbol{x})\,d\boldsymbol{x} = 0 \qquad \text{if } i_1+i_2+i_3 = 1 \text{ or } 3 \leqslant i_1+i_2+i_3 \leqslant r+1 \qquad (5.17)$$

$$\int \|\boldsymbol{x}\|_2^{r+2}|\eta(\boldsymbol{x})|\,d\boldsymbol{x} < \infty \qquad (5.18)$$

src: PhD thesis Ivo Sbalzarini



$$\eta(\mathbf{x}) = \frac{15}{\pi^2}\frac{1}{|\mathbf{x}|^{10}+1}$$

Convergence

- Hints:
  - no need to recompute cell lists (no moving particles)
  - compute error of $q_{pI} = \delta t\ \nu\ \Delta u$ at $t = 0$ for debug
    - should converge to $-3\delta t \nu \pi^2 \cos(x\pi)\cos(y\pi)\cos(z\pi)$
- Example convergence plot (fully unrelated to PSE)