On the performance of parallelized Gromacs simulations of spatial inhomogeneous system (Slab geometry): domain decomposition and PME nodes

Kathleen Kirchner
Physics and Life Sciences, Nanoscience Division, Department of Physics, Strathclyde University, G4 0NG Glasgow, U.K.

April 27, 2012

Abstract

We performed molecular dynamics (MD) simulations with Gromacs 4.5.5 [1] to investigate the structural orientation of imidazole based room temperature ionic liquids (RTILs) at charged electrodes. The resulting configurations consist of 45000 to 50000 atoms per simulation box. To obtain correct electrostatics we used Particle Mesh Ewald (PME) with 3DC correction for slab geometry introduced by Yeh and Berkowitz.[2] The amount of particles requires high parallelization, which requires some additional considerations due to the large unoccupied volume resulting in high spatial inhomogeneity. Some considerations for domain decomposition and node partitioning are given in this document.

Result: When assigning number of PME nodes and domain decomposition by hand it is possible to increase the performace by 5-10 %. More over it is possible to increase the total number of nodes used for a simulation and thus increase the total performance by 50-100 % compared to the maximum number of nodes Gromacs is automatically able to use.

Contents

1 Problem description

To perform a parallelized simulation run, the whole simulation box needs to be split into smaller cells to allow the calculations be performed on more than one node. When using PME electrostatics, also a certain amount of nodes needs to be set as PME nodes. This is usually automatically done by Gromacs, but in case of high or low spatial inhomogeneity of the system (e.g. in a simulation with a vacuum slab), there might be the need for self adjustment to obtain a better performance. Also despite the fact that a theoretical domain decomposition might be possible, Gromacs might not be able to find it on its own. Therefore numerous options are provided for `mdrun` to allow the self adjustment of node decomposition.

2 System under study

We performed molecular dynamics (MD) simulations with Gromacs 4.5.5 [1] to investigate the structural orientation of imidazole based room temperature ionic liquids (RTILs) at charged electrodes. Graphene layers were chosen to model realistic electrodes. The initial structures for the electrodes have been generated with the help of of `ase.structure`[1]. A 3-4 armchair nanoribbon resulted in a graphene sheet of 6 nm × 6 nm area. Five graphene sheets represent one electrode. The distance between the two electrodes is 10 nm, resulting in a accessible volume of 360 nm^3 for the RTILs. We added a vacuum slab by prolonging the box size in z-direction to 33 nm, thereby following the Berkowitz rule

$$\Delta z_{\text{vacuum}} \geq 3 \cdot \max(x, y)$$

to obtain correct electrostatics simulations in slab geometry (see Figure 1).[2] Packmol [3] was used to generate the slab configurations. The resulting configurations consist of 45000 to 50000 atoms per simulation box.

For analysis of the domain decomposition we used BMI BF_4.

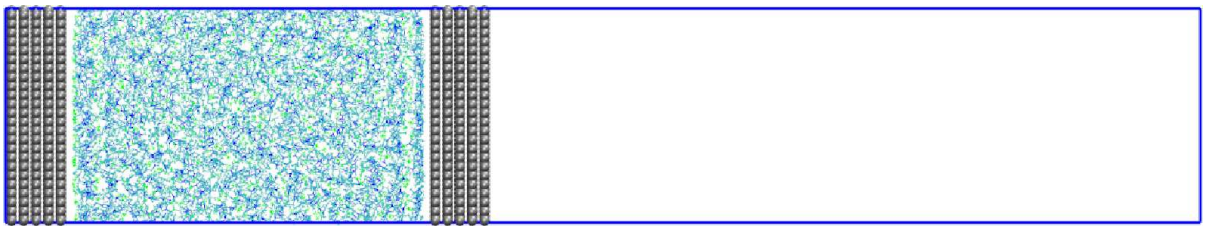

Figure 1: Molecular snapshot of zx-plane of the simulation box of BMI BF_4 in slab geometry. Simulation box is marked as a blue rectangular. The electrodes (graphene sheets) are represented by the grey color. Cations and anions are represented by VMD standard colors.

[1] A tool of the Atomic Simulation Environment (ASE), that is the common part of the simulation tools developed at CAMd. `https://wiki.fysik.dtu.dk/ase/epydoc/ase.structure-module.html`

3 What is told by Gromacs mdrun help

To do: Add explanations

When mdrun is started using MPI with more than 1 node, parallelization is used. By default domain decomposition is used, unless the -pd option is set, which selects particle decomposition.

With domain decomposition, the spatial decomposition can be set with option -dd. By default mdrun selects a good decomposition. The user only needs to change this when the system is very inhomogeneous. Dynamic load balancing is set with the option -dlb, which can give a significant performance improvement, especially for inhomogeneous systems. The only disadvantage of dynamic load balancing is that runs are no longer binary reproducible, but in most cases this is not important. By default the dynamic load balancing is automatically turned on when the measured performance loss due to load imbalance is 5% or more. At low parallelization these are the only important options for domain decomposition. At high parallelization the options in the next two sections could be important for increasing the performace.

When PME is used with domain decomposition, separate nodes can be assigned to do only the PME mesh calculation; this is computationally more efficient starting at about 12 nodes. The number of PME nodes is set with option -npme, this can not be more than half of the nodes. By default mdrun makes a guess for the number of PME nodes when the number of nodes is larger than 11 or performance wise not compatible with the PME grid x dimension. But the user should optimize npme. Performance statistics on this issue are written at the end of the log file. For good load balancing at high parallelization, the PME grid x and y dimensions should be divisible by the number of PME nodes (the simulation will run correctly also when this is not the case).

This section lists all options that affect the domain decomposition. Option -rdd can be used to set the required maximum distance for inter charge-group bonded interactions. Communication for two-body bonded interactions below the non-bonded cut-off distance always comes for free with the non-bonded communication. Atoms beyond the non-bonded cut-off are only communicated when they have missing bonded interactions; this means that the extra cost is minor and nearly indepedent of the value of -rdd. With dynamic load balancing option -rdd also sets the lower limit for the domain decomposition cell sizes. By default -rdd is determined by mdrun based on the initial coordinates. The chosen value will be a balance between interaction range and communication cost. When inter

charge-group bonded interactions are beyond the bonded cut-off distance, mdrun terminates with an error message. For pair interactions and tabulated bonds that do not generate exclusions, this check can be turned off with the option -noddcheck. When constraints are present, option -rcon influences the cell size limit as well. Atoms connected by NC constraints, where NC is the LINCS order plus 1, should not be beyond the smallest cell size. A error message is generated when this happens and the user should change the decomposition or decrease the LINCS order and increase the number of LINCS iterations. By default mdrun estimates the minimum cell size required for P-LINCS in a conservative fashion. For high parallelization it can be useful to set the distance required for P-LINCS with the option -rcon. The -dds option sets the minimum allowed x, y and/or z scaling of the cells with dynamic load balancing. mdrun will ensure that the cells can scale down by at least this factor. This option is used for the automated spatial decomposition (when not using -dd) as well as for determining the number of grid pulses, which in turn sets the minimum allowed cell size. Under certain circumstances the value of -dds might need to be adjusted to account for high or low spatial inhomogeneity of the system.

The option -gcom can be used to only do global communication every n steps. This can improve performance for highly parallel simulations where this global communication step becomes the bottleneck. For a global thermostat and/or barostat the temperature and/or pressure will also only be updated every -gcom steps. By default it is set to the minimum of nstcalcenergy and nstlist.

4 Performance test

The following quotation gives the output for initializing the domain decomposition on 32 nodes for the above specified test system. `mdrun_mpi -maxh $MAXH -deffnm $tpr -dlb auto`

Code 1: Output for initializing the domain decomposition for 32 nodes

```
Initializing Domain Decomposition on 32 nodes
Dynamic load balancing: auto
Will sort the charge groups at every domain (re)decomposition

NOTE: Periodic molecules: can not easily determine the required minimum bonded
    cut-off, using half the non-bonded cut-off

Minimum cell size due to bonded interactions: 0.650 nm
Guess for relative PME load: 0.23
Will use 24 particle-particle and 8 PME only nodes
This is a guess, check the performance at the end of the log file
Using 8 separate PME nodes
Scaling the initial minimum size with 1/0.8 (option -dds) = 1.25
Optimizing the DD grid for 24 cells with a minimum initial size of 0.812 nm
Ewald_geometry=3dc: assuming inhomogeneous particle distribution in z, will not
     decompose in z.
The maximum allowed number of cells is: X 7 Y 7 Z 1
Domain decomposition grid 6 x 4 x 1, separate PME nodes 8
PME domain decomposition: 8 x 1 x 1
Interleaving PP and PME nodes
This is a particle-particle only node
```

As it is suggested in this output, we should check the performance given in the end of the log-file after the simulation finished. Especially the following note should be taken into account.

Code 2: Output after automatic domain decomposition for 32 nodes

```
NOTE: 13.4 % performance was lost because the PME nodes
      had less work to do than the PP nodes.
      You might want to decrease the number of PME nodes
      or decrease the cut-off and the grid spacing.
```

While trying to decrease the performance loss using own specifications for domain decomposition `-dd` and number of PME nodes `-npme`, we also played with the total number of nodes. Table 1 summarizes the performance for several amounts of nodes and domain decompositions.

Indeed decreasing the number of PME nodes also decreases the performance loss. The best performance is reached with `-dd 9 9 1` and `-npme 15` while using 96 nodes. The performance scaling from 32 to 96 nodes is linear.

Table 1: Domain decomposition and performance of a test system on HECToR (1048 BMI BF_4 ionpairs in slab geometry with 10 graphene sheets modelling electrodes, approx 45000 atoms). (*) Performance lost due to load imbalance in the domain decomposition, e.g. some nodes being idle.

Total number of nodes	-dd X Y Z	-npme	performance ns/day	performance hour/ns	performance lost (*)
32	auto 6 4 1	auto 8	5.061	4.742	13.4 %
64	auto (48)	auto 16	error		
64	8 6 1	16	9.325	2.574	11.6 %
64	7 7 1	15	9.836	2.440	10.2 %
64	9 6 1	10	9.945	2.413	5.3 %
96	9 9 1	15	14.470	1.659	5.2 %
96	12 7 1	12	error		

5 Error messages

When initializing the domain decomposition by Gromacs, the following error message might occur.

Code 3: No domain decomposition for xx nodes that is compatible with the given box

```
Initializing Domain Decomposition on 64 nodes
Dynamic load balancing: auto
Will sort the charge groups at every domain (re)decomposition

NOTE: Periodic molecules: can not easily determine the required minimum bonded
    cut-off, using half the non-bonded cut-off

Minimum cell size due to bonded interactions: 0.650 nm
Guess for relative PME load: 0.23
Will use 48 particle-particle and 16 PME only nodes
This is a guess, check the performance at the end of the log file
Using 16 separate PME nodes
Scaling the initial minimum size with 1/0.8 (option -dds) = 1.25
Optimizing the DD grid for 48 cells with a minimum initial size of 0.812 nm
Ewald_geometry=3dc: assuming inhomogeneous particle distribution in z, will not
     decompose in z.
The maximum allowed number of cells is: X 7 Y 7 Z 1

-------------------------------------------------------

Program mdrun_mpi, VERSION 4.5.5
Source code file: domdec.c, line: 6436
```

```
20
21 Fatal error:
22 There is no domain decomposition for 48 nodes that is compatible with the given
       box and a minimum cell size of 0.8125 nm
23 Change the number of nodes or mdrun option -rdd or -dds
24 Look in the log file for details on the domain decomposition
25 For more information and tips for troubleshooting, please check the GROMACS
26 website at http://www.gromacs.org/Documentation/Errors
```

Gromacs was not able to find a suitable domain decomposition for 64 nodes. I suggest the problem lying in the additional scaling factor of 1.25 for the minimum cell size. This is a constraint to prevent the Gromacs tool from doing something to wrong. It is necessary to try self adjusted domain decomposition and PME node assigning.

In the case of the following error message the cell size limit is really violated. This message is a hard limit for any parallelization.

Code 4: Initial cell size smaller than cell size limit

```
1  mdrun_mpi -maxh $MAXH -deffnm $tpr -dlb auto -dd 12 7 1 -npme 12
2
3  #++++++++++++++++++++++++++++++++++++++++++++++++
4
5  Initializing Domain Decomposition on 96 nodes
6  Dynamic load balancing: auto
7  Will sort the charge groups at every domain (re)decomposition
8
9  NOTE: Periodic molecules: can not easily determine the required minimum bonded
       cut-off, using half the non-bonded cut-off
10
11 Minimum cell size due to bonded interactions: 0.650 nm
12 ERROR: The initial cell size (0.491903) is smaller than the cell size limit
       (0.650000)
13
14 ------------------------------------------------------------
15 Program mdrun_mpi, VERSION 4.5.5
16 Source code file: domdec.c, line: 6413
17
18 Fatal error:
19 The initial cell size (0.491903) is smaller than the cell size limit (0.650000)
       , change options -dd, -rdd or -rcon, see the log file for details
20 For more information and tips for troubleshooting, please check the GROMACS
21 website at http://www.gromacs.org/Documentation/Errors
```

6 How to obtain the best domain decomposition

In our case we have the following constraints:

- Box size in x and y dimension: $= 5.99\,\text{nm}$
- Minimum cell size due to bonded interactions: $= 0.65\,\text{nm}$
- Number of nodes to use: multiples of 32 due to the HECToR policy

  > [aprun -N 32 / mppnppn=32] Specifies the number of distributed memory parallel tasks per node. There is a choice of 1-32 for phase 3 (XE6) nodes. As you are charged per node on HECToR the most economic choice is always to run with "fully-packed" nodes if possible, i.e. -N 32. Running "unpacked" or "underpopulated" (i.e. not using all the cores on a node) is useful if you need large amounts of memory per parallel task or you are using more than one shared-memory thread per parallel task. When running "unpacked", please keep in mind that your budget gets charged for all cores of each node, even if some of the other cores are idle.

Let's do some maths:

- Maximum number of domains for x-axis:

$$dx = 5.99/0.65 \approx 9$$

- Maximum number of domains for y-axis:

$$dy = 5.99/0.65 \approx 9$$

- Maximum number of domains for z-axis:

$$dz = 1$$

  due to slab configuration

- Maximum amount of PP nodes:

$$nPP = dx \times dy \times dz = 9 \times 9 \times 1 = 81$$

- Maximum amount of PME nodes per total number of nodes:

$$nPME < \frac{8}{32} \cdot nNodes = 0.25 \cdot nNodes$$

(Taking the initial domain decomposition of 8 PME nodes and $6 \times 4 \times 1 = 24$ PP nodes into account, and the performance loss of 13.4 %).

Now we need to find a good choice for the number of nodes for PP and PME with

$$nPP + nPME = nNodes$$

and

$$nNodes = 32 \cdot u \qquad\qquad u \in \mathbb{N}$$

.

$$nPP + 0.25\, nNodes > nNodes$$

$$81 + 0.25 \cdot 32 \cdot u > 32 \cdot u$$

$$u < \frac{81}{0.75 \cdot 32} \approx 3.375$$

Thus we obtain $u = 3$, a total number of nodes of 96 split into 81 PP nodes and 15 PME nodes. This will be the node composition with the highest possible performance.

```
mdrun_mpi -maxh $MAXH -deffnm $tpr -dlb auto -dd 9 9 1 -npme 15
```

28

References

[1] B. Hess, C. Kutzner, D. van der Spoel, and E. Lindahl. Gromacs 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation. *Journal of Chemical Theory and Computation*, 4(3):435–447, March 2008.

[2] I.C. Yeh and M.L. Berkowitz. Ewald summation for systems with slab geometry. *Journal of Chemical Physics*, 111:3155–3162, 1999.

[3] L. Martínez, R. Andrade, E. G. Birgin, and J. M. Martínez. Packmol: A package for building initial configurations for molecular dynamics simulations. *Journal of Computational Chemistry*, 30(13):2157–2164, October 2009.