Performance of ProtoMol

Thierry Matthey
matthey@ii.uib.no

June 24, 2005

Abstract

A collection of performance tests of ProtoMol during its development and a note about how to tune the performance of force algorithms.

1 Performance: ProtoMol 1.8.3 vs. ProtoMol 2.0.2

Ver.	Machine		Compiler	Vacuum			PBC		
				Cell Size					
				10	5	$3\frac{1}{3}$	10	5	$3\frac{1}{3}$
2.0.2	Dual AMD Opteron	lcsl	gcc 3.2.3	47.77	38.37	42.65	83.50	57.51	61.98
1.8.3	Dual AMD Opteron	lcsl	gcc 3.2.3	49.66	41.72	46.73	99.05	74.47	95.90
2.0.2	Dual AMD Opteron	lcsl	gcc 3.2.3	341.98	274.11	303.87	645.56	417.45	428.67
1.8.3	Dual AMD Opteron	lcsl	gcc 3.2.3	350.59	296.86	329.28	848.47	555.22	665.11

Table 1: BProtoMol 1.8.3 vs 2.0.2: BPTI 14281 Atoms / ApoA1 92224 Atoms, 100 Steps.

2 Performance: ProtoMol 1.8.3

Machine		Compiler	Vacuum			PBC		
			Cell Size					
			10	5	$3\frac{1}{3}$	10	5	$3\frac{1}{3}$
Dual AMD Opteron 248 2.2GHz	le04	Intel 8.0	57.6068	46.53375	46.78541	104.51191	75.19006	84.90123
Dual AMD Opteron 248 2.2GHz	le04	gcc 3.2.2	42.23086	34.98725	36.11871	78.57884	61.0053	69.73622
Pentium(R) 4 2.53GHz	romhegg	Intel 8.0	82.61629	64.27086	63.91178	131.30994	93.24453	103.0066
Pentium(R) 4 2.53GHz	romhegg	gcc 3.2.2	113.0993	94.08331	86.44412	182.61486	134.46937	146.3645
P4+ v2.2 1.6GHz (*)	blade	xlC 6						
P4+ v2.2 1.6GHz	blade	gcc 3.2.3	114.22825	98.49399	98.75213	172.38679	135.47089	145.3059
P4 1.3Ghz	tre	xlC 6	124.77146	104.73123	108.48935	199.67187	153.70434	170.16276
Dual AMD Opteron 2.0Ghz (*)	c0-0	pathCC 1.2						
Dual AMD Opteron 2.0Ghz	c0-0	gcc 3.2.3	50.73854	42.55911	43.812	96.66381	73.69411	82.10057
MIPS R14000 1.4	gridur	MIPSpro 7.4	126.04032	106.59266	118.52494	198.53973	163.74625	196.26711

Table 2: BProtoMol 1.8.3: BPTI 14281 Atoms, 100 Steps.

3 Performance: ProtoMol 2000.03.10 vs. NAMD 2.2

Runs were performed on a SGI Onyx2 (250MHz r10000). The times represent the wall time [s] for 10 steps, sequential. For every step ProtoMol uses plain Ewald summation, where NAMD2 uses PME.

3.1 ApoA1, 92224 atoms

ProtoMol		NAMD2	
Periodic boundary conditions			
Plain Ewald	2152.45	PME	712.96
Cutoff	407.94	Cutoff	277.45
Normal boundary conditions			
Cutoff	304.86	Cutoff	220.87
Full Coulomb	18662.70	Full Coulomb	–
Full Coulomb & VdW	–	Full Coulomb & VdW	–

Table 3: ApoA1, 92224 atoms, cutoff of 12A.

3.2 BPTI, 14281 atoms

ProtoMol		NAMD2	
Periodic boundary conditions			
Plain Ewald	209.38	PME	57.10
Cutoff	33.19	Cutoff	22.53
Normal boundary conditions			
Cutoff	25.36	Cutoff	20.00
Full Coulomb	455.03	Full Coulomb	–
Full Coulomb & VdW	349.43	Full Coulomb & VdW	351.49

Table 4: BPTI, 14281 atoms, cutoff of 10A.

4 Tuning ProtoMol

Since ProtoMol provides wide variety of different force algorithms, which can be combined arbitrary, it may not always be simple to find the an appropriate parametrization and force definition.

4.1 Direct methods

For the direct methods – all pair-wise interactions are considered – on may change the `blocksize` in order to tweak the block size of the sub matrices and improve the cache hits. The direct methods do compute sub matrix by sub matrix and do not depend an the cell list algorithms, i.e., `cellsize` has no influence. Note that changing the block size does change the order of computing the interactions, e.g., is the block size $> N$ (number of atoms), the algorithms does a simple 2-nested loop over all atoms.

4.2 Cutoff methods

The cutoff methods use the cell list algorithm (see Fig.1) in order to retrieve all neighbor particles in $O(N)$. The Cell list algorithm requires a cell. The algorithm keeps track of all particles residing in the cell and can therefore retrieve them in $O(N)$. It finds all neighbors (candidates) of all particles

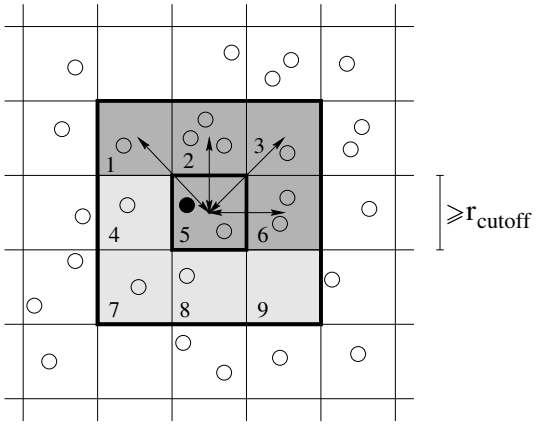

Figure 1: Partition of cells, where cell no. 5 is the origin and 1-4, 6-9 the neighbor cells. Cell no. 6, 3, 2 and 1 describe the interaction path.

of the original cell, which defines the minimal dimension of a single which are equal or less distant than the actual cutoff. A cell size bigger than the cutoff will result with more pairs which are more distant than the cutoff, where a smaller cell size will increase the total number of cells to cover the system, decrease the number of particles per cell and make the retrieval of candidates more accurate. The cutoff methods themself have a cutoff at which distance particles further away as neglected. The cutoff methods normally offers different switching function to achieve certain properties, i.e., C^2-continues at the cutoff point. The cell size should be chosen dependent on the cutoff's of all forces, also considering those coming from switching functions and fast electrostatic forces. In case of vacuum, the cell dimensions will be defined exactly the given cell size, where as for periodic boundary conditions the cell dimensions are chosen such that the number of cells in the simulation box is maximal, but with at least dimensions defined by the cell size. For small periodic systems the actual cell dimensions may be significant lager than the cell size to fit the system box. Choosing the cell size as the maximum of all cutoff is for most systems a good starting point. One

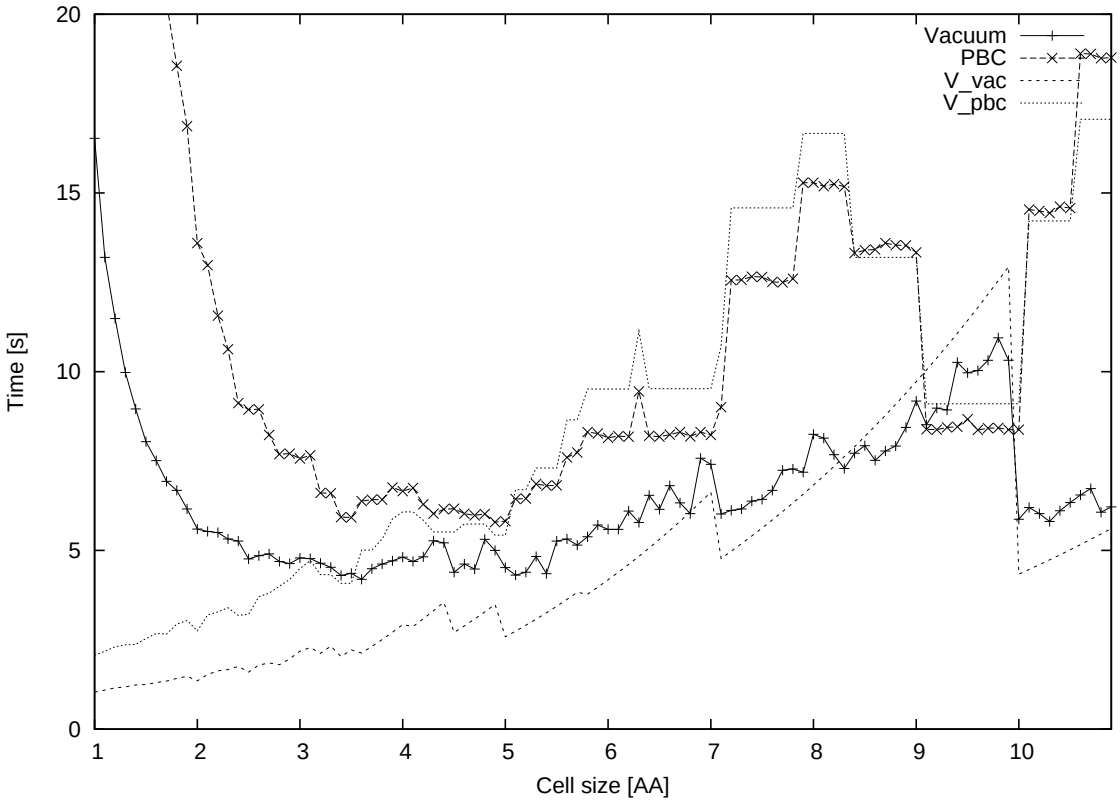

Figure 2: BPTI 14281 Atoms, run with different cell sizes, vacuum and PBC, cutoff 10[Å].

may also try out the cell sizes such that the maximal cutoff is a multiple. The smaller the cell size the less particles reside in a single cell, and increasing the overhead of the cell list algorithm. On the other hand, a cell size which is smaller (a multiple) than the cutoff enables the cell list algorithm to decrease the domain (volume) to pick pair-wise candidates to interact. Fig.2 gives a rough picture how performance may change with a different cell size. For small cell sizes (< 3) the overhead of the cell list is significant, especially for vacuum where the total number of cells variate more often. The volume to pick possible candidates is small and close to a sphere. For cell sizes > 3 and < 5 one reaches best performance, especially for cell size $3\frac{1}{3}$ and 5. For lager cell sizes (> 5) the performance follows more or less the volume curve, but for cell size 10 the volumes drops and the performance gets better again. One can also see that with periodic boundary conditions the performance is more stair alike due to the fact that the real cell dimensions are always a multiple of the system dimensions, but at least the cell size. The saw effect for vacuum is due to constant number of total cells until the cells a big enough to cover the system with less cells.

4.3 Fast electrostatic methods

Three main fast electrostatic force mtheods are supported:

- Ewald, $O(N^{\frac{3}{2}})$
- Particle-Mesh-Ewald, $O(N \log(N))$
- Multi Grid, $O(N)$

Please consult their tutorials.

4.3.1 Evaluation of two potentials simultaneously

One on the simples and most effective performance tuning is to combine two potentials evaluated by a cutoff or direct method. ProtoMol offers to combine the pair-wise potentials to be compute at the same time, which for big systems convergences to a improvement of close to 2. This idea does also apply for the fast electrostatic forces, which have a pair-wise part. For the exmaple BPTI with 14281 atoms the total force evaluatation takes 9.1s for 10 steps:

```
Force LennardJones
  -switchingFunction C2
  -algorithm NonbondedCutoff
  -switchon 1        # C2 swf switch on
  -cutoff 8          # C2 swf cutoff
  -cutoff 8          # algorithm cutoff
Force Coulomb
  -switchingFunction Shift
  -algorithm NonbondedCutoff
  -cutoff 10         # shift swf cutoff
  -cutoff 10         # algorithm cutoff
```

Where combinding reduces the total force evaluation to 5.8s:

```
Force LennardJones Coulomb
  -switchingFunction C2
  -switchingFunction Shift
  -algorithm NonbondedCutoff
  -switchon 1        # C2 swf switch on
  -cutoff 8          # C2 swf cutoff
  -cutoff 10         # shift swf cutoff
  -cutoff 10         # algorithm cutoff
```

4.3.2 Look-up tables

One may also consider to replace computational expensive pair-wise potentials by look-up tables, but at a price of more memory consumption, lower accuracy and increase of cache misses.