Hybrid codes for atomistic simulations on the Desmos supercomputer: GPU-acceleration, scalability and parallel I/O

Nikolay Kondratyuk[123], Grigory Smirnov[123], and Vladimir Stegailov[123]

[1] Joint Institute for High Temperatures of RAS, Moscow, Russia
[2] Moscow Institute of Physics and Technology, Dolgoprudny, Russia
[3] National Research University Higher School of Economics, Moscow, Russia
`stegailov.vv@mipt.ru`

Abstract. In this paper, we compare different GPU accelerators and algorithms for classical molecular dynamics using LAMMPS and GROMACS codes. BigDFT is considered as an example of the modern *ab initio* code that implements the density functional theory algorithms in the wavelet basis and uses effectively GPU acceleration. Efficiency of distributed storage managed by the BeeGFS parallel file system is analysed with respect to saving of large molecular-dynamics trajectories. Results have been obtained using the Desmos supercomputer in JIHT RAS.

Keywords: molecular dynamics · density functional theory · GPU acceleration · strong scaling · parallel I/O

1 Introduction

Hybrid codes that use GPU acceleration provide an effective way to cost-effective and energy-effective calculations. GPU-based Summit and Sierra supercomputers are going to be, perhaps, the fastest supercomputers in 2018, each with about 150 PFlops of peak performance. Their development is a clear illustration of the success achieved by hybrid computing methods that have been growing intensively since 2007 when Nvidia introduced the CUDA framework for GPU programming. The complexity of hybrid parallelism and the diversity of GPU hardware motivate the detailed studies of the hardware-software matching and efficiency, especially for widely used popular software packages.

Desmos (see Figure 1) is a supercomputer targeted to MD calculations that has been installed in JIHT RAS in December 2016. Desmos is the first application of the Angara interconnect for a GPU-based MPP system [1]. In 2018 one half of the nodes have been upgraded to AMD FirePro S9150 accelerators.

Modern MPP systems can unite up to 10^5 nodes for solving one computational problem. For this purpose, MPI is the most widely used programming model. The architecture of the individual nodes can differ significantly and is usually selected (co-designed) for the main type of MPP system deployment. The most important component of MPP systems is the interconnect that properties stand behind the scalability of any MPI-based parallel algorithm. In this work, we describe performance results related to the Desmos supercomputer based on 1CPU+1GPU nodes connected by the Angara interconnect.

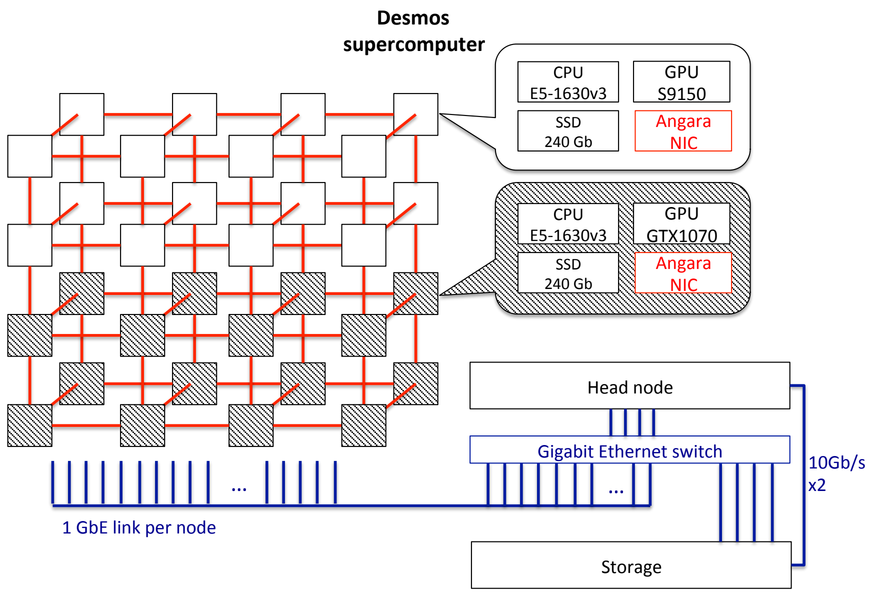


Fig. 1: The scheme of the Desmos supercomputer. 32 computational single-socket nodes based on Intel Xeon E5-1650v3 CPUs are connected by the Angara interconnect with torus topology (16 nodes are equipped with Nvidia GTX1070 GPUs and 16 nodes are equipped with AMD FirePro S9150 GPUs).

The Angara interconnect is a Russian-designed communication network with torus topology. The interconnect ASIC was developed by JSC NICEVT and manufactured by TSMC with the 65 nm process. The Angara architecture uses some principles of IBM Blue Gene L/P and Cray Seastar2/Seastar2+ torus interconnects. The torus interconnect developed by EXTOLL is a similar project [2]. The Angara chip supports deadlock-free adaptive routing based on bubble flow control [3], direction ordered routing [4,5] and initial and final hops for fault tolerance [4].

The results of the benchmarks confirmed the high efficiency of commodity GPU hardware for MD simulations [1]. The scaling tests for the electronic structure calculations also showed the high efficiency of the MPI-exchanges over the Angara network.

In the first part of the paper, we consider several GPU accelerators installed in the chassis of the Desmos supercomputer. We compare performance of different classical MD algorithms in LAMMPS and GROMACS. Additionally, we make the similar benchmarks on the IBM Minsky server and compare the results. In the second part of the paper, the performance of the BigDFT code is considered in the CPU-only variant and in the GPU-accelerated version using AMD FirePro S9150 accelerators installed in Desmos. In the third part of the paper, we consider the efficiency of distributed storage of Desmos nodes managed by the BeeGFS parallel file system for large MD trajectories storage.

2 Related work

Many algorithms have been rewritten and thoroughly optimized to use the GPU capabilities. However, the majority of them deploy only a fraction of the GPU theoretical performance even after careful tuning, e.g. see [6,7,8]. The sustained performance is usually limited by the memory-bound nature of the algorithms.

Among GPU-aware MD software one can point out GROMACS [9] as, perhaps, the most computationally efficient MD tool and LAMMPS [10] as one of the most versatile and flexible for MD models creation. Different GPU off-loading schemes were implemented in LAMMPS [11,12,13,14]. GROMACS provides a highly optimized GPU-scheme as well [15].

There are other ways to increase performance of individual nodes: using GPU accelerators with OpenCL, using Intel Xeon Phi accelerators or even using custom built chips like MDGRAPE [16] or ANTON [17]. Currently, general purpose Nvidia GPUs provide the most cost-effective way for MD calculations [18].

The main *ab initio* codes (VASP, Abinit, Quantum Espresso, CP2K) have recently implemented the off-loading scheme that use GPU-acceleration. However these implementations do not modify the main structure of the codes. That is why only limited speed-up of calculations can be achieved (limited in comparison with huge GPU peak performance). All the codes require GPU computing in double precision (here we can mention the success of TeraChem package [19] that illustrates the amazing perspectives of single precision GPU usage for quantum chemistry). BigDFT code is a rare example of the code that 1) has been developed with GPU acceleration from the very beginning and 2) uses mainly OpenCL for its GPU parts that widens the spectrum of the GPU hardware one could consider.

The ongoing increase of data that are generated by HPC calculations leads to the requirement for parallel file system for rapid I/O operations. However, benchmarking parallel file system is a complicated (and usually expensive!) task that is why accurate results of particular case studies are quite rare (see e.g. [20]).

3 Classical molecular dynamics with LAMMPS and GROMACS

The performance of the different node specifications is tested on the molecular dynamics benchmarks that represent the most common models of the matter.

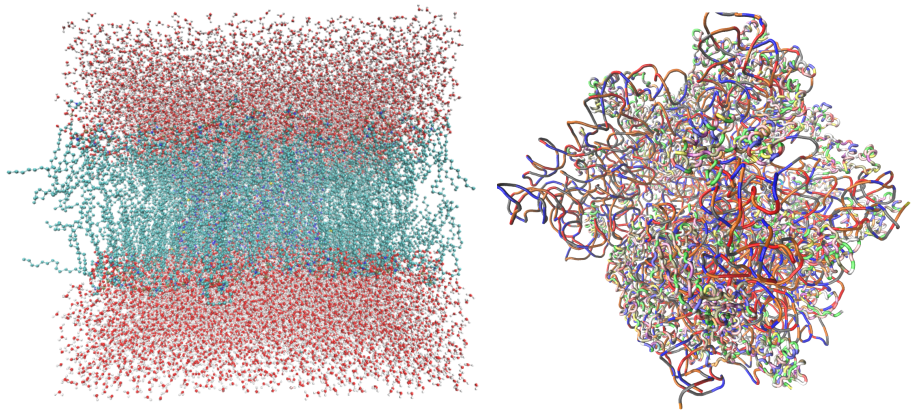


Fig. 2: The structure of the benchmark models MEM (left) and RIB (right, water molecules not shown).

The accuracy of the single precision of GPU is enough for such simulations in contrast to the quantum calculations that are described below. Therefore, the results presented in this section show the performance in the terms of single precision.

We use two wide-spread molecular dynamics programs: LAMMPS compiled with GPU package [21,22,23] and GROMACS which is highly optimized for the simulations of biological systems. In LAMMPS, the hydrocarbon liquid $C_{30}H_{62}$ [24] is used as a benchmark system. In GROMACS, the bio-membrane (MEM) and ribosome (RIB) systems [18] are considered (see Figure 2). In all cases, the performance is measured in nanoseconds of the simulated physical time per day.

Table 1 represents the tests performed on one Desmos node (Intel E5-1650v3 + Nvidia GTX 1070) as well as on the separate test nodes: the FirePro1 node (Intel E5-1650v3 + AMD FirePro S9150), the FirePro2 node (Intel E5-2620v4 + AMD FirePro S9150), the Tesla1 node (Intel E5-1650v3 + Nvidia Tesla V100), the Tesla2 node (Intel E5-2620v4 + Nvidia Tesla V100) and the Jupiter node at the supercomputer center of FEB RAS (2x IBM POWER8 + 2x Nvidia Tesla P100). The number of atoms in the model of hydrocarbon liquid in LAMMPS is about 700k. In the GROMACS benchmarks, MEM consists of 100k atoms and RIB is about 2M atoms.

For MEM test with LAMMPS, the Desmos node shows 2x better result than the FirePro1 node which has nearly the same peak floating point performance in single precision. The reason is that GPU package in LAMMPS is better optimized for CUDA than for OpenCL. The Tesla1 node exceeds the Desmos node due to the higher GPU performance.

LAMMPS can be also built with the GPU capabilities implemented with the KOKKOS library (this is an alternative for the GPU package). The KOKKOS library requires double precision. The advantage of the KOKKOS package is that

Table 1: The performance of different node types for the 3 benchmarks: $C_{30}H_{62}$ (LAMMPS), MEM and RIB (GROMACS).

Node	Desmos	FirePro1	FirePro2	Tesla1	Tesla2	Jupiter
CPU	E5-1650v3 6 cores 3.5 GHz	E5-1650v3 6 cores 3.5 GHz	E5-2620v4 8 cores 2.1 GHz	E5-1650v3 6 cores 3.5 GHz	E5-2620v4 8 cores 2.1 GHz	2xPOWER8 2x10 cores 2.86 GHz
GPU	GTX 1070 5.8 TF(sp) 256 GB/s	FPro S9150 5 TF(sp) 2.5 TF(dp) 320 GB/s	FPro S9150 5 TF(sp) 2.5 TF(dp) 320 GB/s	Tesla V100 14 TF(sp) 7 TF(dp) 900 GB/s	Tesla V100 14 TF(sp) 7 TF(dp) 900 GB/s	2xTesla P100 10 TF(sp) 5 TF(dp) 732 GB/s
Options	SLES11 gcc 4.8	Ubuntu 16.04 gcc 5.3				CentOS 7.3 xlc 13.1.5
	CUDA 8.0 ver.384.69	AMDGPU-PRO 17.40		CUDA 9.1 ver.384.98		CUDA 8.0 ver.384.59
LAMMPS (ns/day)						
$C_{30}H_{62}$	0.53	0.26		0.70		1.04 (0.52)
GROMACS (ns/day)						
MEM	35.5	27	23.8	50	44.4	44
RIB	2.1	2.4	2.5	3.5	3.2	4.7

the most of calculations are performed on GPU, hence it is not significant with respect to performance which kind of CPU is used as a host device. We varied the number of CPU cores used in the benchmark on the Jupiter node, the ideal choice is 2 cores (from different sockets) per 2 GPUs (Figure 3). The performance is two times less than in the GPU package case (see the value 0.52 ns/day in brackets in Table 1). The reason is that Tesla P100 has two times lower floating point peak performance in double precision than in single precision.

The MEM benchmark results on the Desmos and FirePro1, FirePro2 nodes show that GROMACS loses less in performance when it runs using OpenCL than LAMMPS. Despite lower peak floating point performance, the Tesla1 and Tesla2 nodes overpower one Jupiter node in the case of a smaller system (MEM). It is caused by the higher memory bandwidth for Tesla V100 than for Tesla P100.

The FirePro nodes win the Desmos node in the RIB benchmark which is big enough to use FirePro S9150 at the limit of its performance. The peak performance of Jupiter node (maximum among the systems considered) is visible on such a big model as well since it results in the fastest calculation speed.

4 Density functional theory with BigDFT

Kohn-Sham density functional theory (DFT) is a well-established method to calculate electronic structure of atomistic systems by reducing many-body Schrödinger equation to a one-body problem. This approach is relatively simple and accurate, so it became common in chemistry, physics and biology. However, algorithm is computationally demanding, modern supercomputer is required even

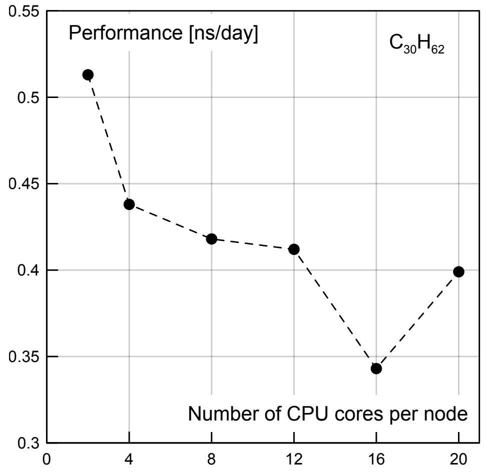


Fig. 3: The performance of the Jupiter node for $C_{30}H_{62}$ model using the KOKKOS package in LAMMPS (the runs are performed with two Tesla P100 GPUs). The best performance is achieved using two CPU cores for two GPUs.

for hundreds of atoms. Computational power of the CPUs stimulated the development of many free and commercial DFT codes in the past decades. Unfortunately, most of them do not have GPU-accelerated version or GPU support is limited (e.g. only single GPU) due to difficulties with many-cores hybrid parallelization.

One of the key properties of a DFT code is the form of basis set functions to expand the Kohn-Sham orbitals. The most popular choices are plane waves and Gaussian functions. Plane waves codes (e.g. ABINIT, CPMD, Quantum ESPRESSO, VASP) are mainly used for periodic and homogeneous systems, but they are less efficient for isolated systems, as high-energy cutoff and many plane waves are required for accurate expanding of localized functions. So Gaussian functions are preferred for electronic structure calculations of isolated molecules. Examples of such codes are GAMESS, GAUSSIAN, ORCA, TeraChem. Siesta is another package developed for periodic systems. Mixing of plane waves and localized orbitals in one basis is possible. Such a mixed Gaussian and plane wave method is implemented in CP2K and allows to perform efficient calculations of complex systems.

BigDFT [25,26] is a unique DFT code as it uses Daubechies wavelets basis for Kohn-Sham orbitals. Such method is well suited for GPU acceleration, not only single, but multiple cards. Both CUDA and OpenCL versions has been developed, the code is distributed under GNU GPL licence. Daubechies wavelets have all desired properties for effective parallelization, they form a systematic orthogonal and smooth basis, localized both in real and Fourier spaces. Some calculations in this basis can be done analytically, the other operations are based

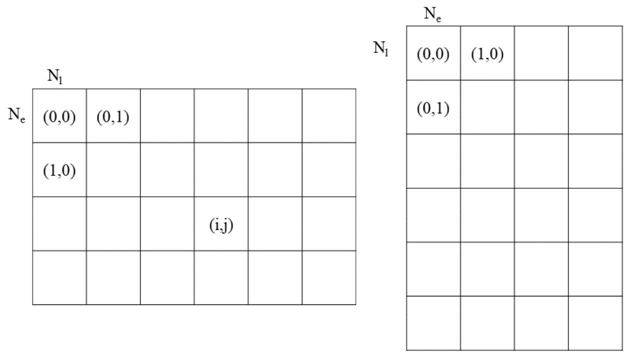


Fig. 4: One unidimensional convolution with transposition.

on convolutions with short-range filters, which can be effectively calculated on GPU.

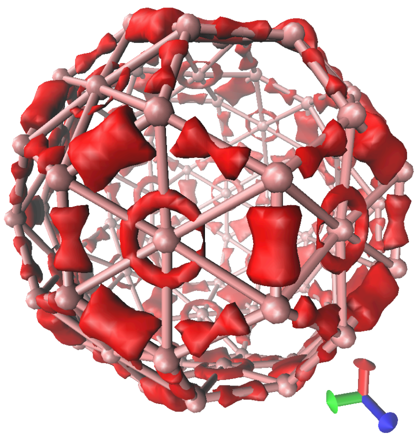
N independent convolutions should be computed. Each of the lines is split in chunks of size N_e, each multiprocessor of the GPU computes a group of N_l different chunks. After, $N_e N_l$ elements are copied in the corresponding part of the transposed output array. Three-dimensional convolutions can be expressed as a succession of three unidimensional convolution/transposition. Figure 4 shows the data distribution on the grid of blocks during the transposition. The GPU versions of the convolutions are about ten times faster than the CPU ones, though theoretical peak performance can not be achieved due to memory transfers. Some tricks are also used to reduce CPU-GPU and GPU-GPU communications.

In order to benchmark GPU-acceleration, we consider two models. One model represents a ball-like molecule of B_{80} (Figure 5). Another example represents a unit cell of f.c.c. Ag crystal with 4 atoms and a k-mesh of 15x15x15 (Figure 6).

Benchmark results show that GPU-acceleration can be quite substantial. Both models have quite good strong scaling for up to 16 nodes of Desmos equipped with FirePro accelerators.

One remarkable difference between the models is that for B_{80} the best speed corresponds to 1 MPI process per CPU. But in the case of the Ag crystal using multiple MPI processes with one GPU is beneficial. The subset of points on Figure 6 shows the decrease of the time-to-solution with increasing number of cores.

BigDFT code combines different parallel programming techniques: OpenMP, MPI and OpenCL/CUDA. At the same time, DFT algorithm has several strategies of parallel data mapping (parallelism for orbitals, for k-points, for basis functions). These two aspects make difficult the choice of the most efficient software-hardware combination of a particular problem. The benchmarks presented in this work is an attempt to pave the way in solving this complicated problem.



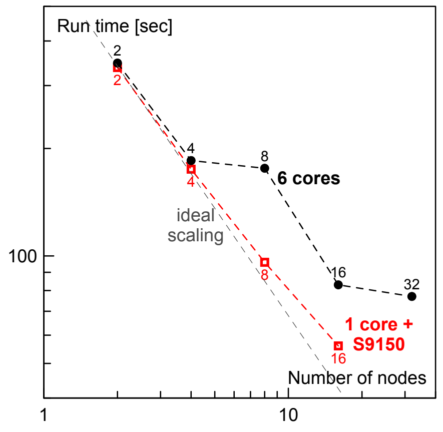
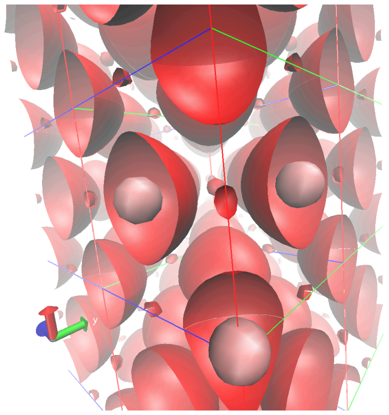


Fig. 5: The visual representation of the B_{80} molecule (red surfaces correspond to the electron density 0.15 e/A^3) and the strong scaling of the calculation of the electronic structure of this molecule for Desmos nodes w/ and w/o deploying the GPU (a grey dashed line shows ideal scaling, numbers of nodes are shown near symbols).



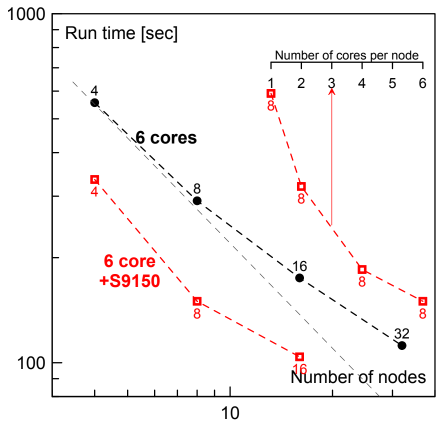


Fig. 6: The visual representation of the f.c.c. Ag crystal (red surfaces correspond to the electron density 0.009 e/A^3, several replicated unit cells are shown for demonstration of periodicity) and the strong scaling of the calculation of the electronic structure of this system for Desmos nodes w/ and w/o deploying the GPU (a grey dashed line shows ideal scaling, numbers of nodes are shown near symbols, the additional set of data shows the dependence on the number of cores per node that is the number of MPI processes using the same GPU simultaneously).

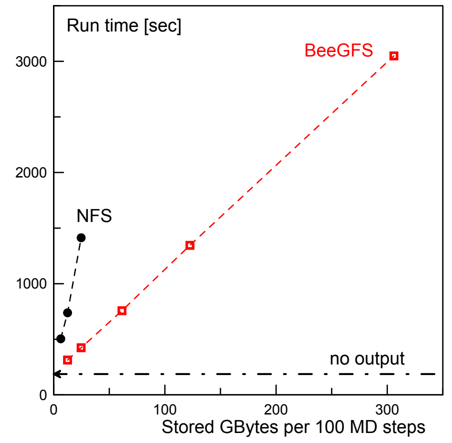


Fig. 7: Parallel output benchmarks based on the LAMMPS test model (16 million atoms, 2500 timesteps). The black circles show the case when the NFS-mounted /home folder from the head node is used for storage. The red squares represent the case when BeeGFS file system is mounted on all Desmos nodes and on the head node. This BeeGFS file system unites all 32 SSD disks in the Desmos nodes (with xfs file system).

5 Parallel file system benchmarks

Many HPC codes generate huge amounts of data. For example, for classical molecular dynamics (MD) the limits of the system size are trillions of atoms [27]. Desmos allows GPU-accelerated modelling of MD system with up to 100 millions atoms with GTX1070 nodes and up to 200 million atoms with FirePro S9150 equipped nodes. On-the-fly methods of data processing do help considerably, but can not substitute post-processing completely for such tasks as, for example, plastic deformation simulations. Another unavoidable requirement is the saving of control (or restart) points during (or at the end) of the calculation.

All 32 nodes of Desmos are equipped with fast XFS formatted SSD drives and the BeeGFS parallel file system has been installed in order to use all these disks as one distributed storage (all nodes boot over ethernet so these disks are used for storage only). It is important to emphasize that BeeGFS works over TCP/IP protocol and *the slow gigabit network*.

Two variants of the BeeGFS configuration are considered.

A. Each of 32 nodes is used both as a meta-data server and as a storage server.
B. Each of 32 nodes is used as a storage server only. A meta-data server runs on the head node (that hosts the /home folder on the SSD RAID1) and is associated with a dedicated SSD disk.

The standard Lennard-Jones benchmark is used with the LAMMPS molecular dynamics package (the benchmark is based on the replicated example “melt”).

LAMMPS has different variants for output large amounts of data. Here, we present the results obtained with MPI-IO.

Figure 7 presents the benchmarks for different frequencies of writing MD trajectory data. The NFS storage (/home folder of the head node) becomes prohibitively slow for higher writing frequencies. The distributed storage governed by BeeGFS demonstrates saturation and does not scale well but, at least, gives the possibility to store necessary data in a reasonable time. No difference between A and B configurations is found for these tests.

However, there is a considerable difference in speed of copying file from /mnt/beegfs to /home on the head node. For benchmarking, we use

```
dd iflag=direct,fullblock bs=10G \
                 if=/mnt/beegfs/dump.melt.mpiio of=/dev/null
```

that results in 200 MB/s reading speed for the A configuration and 315 MB/s reading speed for the B configuration. It shows that the configuration of the meta-data servers plays an important role and can significantly increase the reading speed from the distributed storage.

6 Conclusions

The representative set of CPU-GPU combinations have been benchmarked for classical MD GPU-algorithms implemented in LAMMPS and GROMACS. The results show that performance of the Desmos nodes is on par with other configurations (newer and/or more expensive). An important observation is that the GPU-oriented MD algorithm based on the KOKKOS library does not require a high performance CPU for getting maximum efficiency from the GPU.

The GPU-acceleration of the BigDFT code provides about 2x speed-up for the benchmarks considered. The benchmarks of an isolated DFT model and a periodic DFT model scale on Desmos well.

Distributed storage united by BeeGFS can improve performance in comparison with a local NFS storage for large MD data files even over slow gigabit ethernet network. The proper choice for the configuration of the meta-data servers can give 1.5x increase in the reading speed from the distributed storage.

Acknowledgments. The work is supported by the Russian Science Foundation (grant No. 14-50-00124). HSE and MIPT helped in accessing some hardware used in this study. We acknowledge Shared Resource Center of CC FEB RAS for the access to the Jupiter supercomputer (http://lits.ccfebras.ru). We are grateful to Nvidia for providing us with one Tesla V100 card for the benchmarks.

References

1. Vladimir Stegailov, Alexander Agarkov, Sergey Biryukov, Timur Ismagilov, Mikhail Khalilov, Nikolay Kondratyuk, Evgeny Kushtanov, Dmitry Makagon, Anatoly Mukosey, Alexander Semenov, Alexey Simonov, Alexey Timofeev, and Vyacheslav Vecher. Early performance evaluation of the hybrid cluster with torus

interconnect aimed at molecular-dynamics simulations. In Roman Wyrzykowski, Jack Dongarra, Ewa Deelman, and Konrad Karczewski, editors, *Parallel Processing and Applied Mathematics*, pages 327–336, Cham, 2018. Springer International Publishing.

2. S. Neuwirth, D. Frey, M. Nuessle, and U. Bruening. Scalable communication architecture for network-attached accelerators. In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, pages 627–638, Feb 2015.
3. V. Puente, R. Beivide, J. A. Gregorio, J. M. Prellezo, J. Duato, and C. Izu. Adaptive bubble router: a design to improve performance in torus networks. In *Proceedings of the 1999 International Conference on Parallel Processing*, pages 58–67, 1999.
4. Steven L. Scott and Gregory M. Thorson. The Cray T3E network: Adaptive routing in a high performance 3D torus. In *HOT Interconnects IV, Stanford University, August 15-16, 1996*, 1996.
5. N. R. Adiga, M. A. Blumrich, D. Chen, P. Coteus, A. Gara, M. E. Giampapa, P. Heidelberger, S. Singh, B. D. Steinmacher-Burow, T. Takken, M. Tsao, and P. Vranas. Blue Gene/L torus interconnection network. *IBM J. Res. Dev.*, 49(2):265–276, March 2005.
6. G. S. Smirnov and V. V. Stegailov. Efficiency of classical molecular dynamics algorithms on supercomputers. *Mathematical Models and Computer Simulations*, 8(6):734–743, 2016.
7. Vladimir V. Stegailov, Nikita D. Orekhov, and Grigory S. Smirnov. *Parallel Computing Technologies: 13th International Conference, PaCT 2015, Petrozavodsk, Russia, August 31-September 4, 2015, Proceedings*, chapter HPC Hardware Efficiency for Quantum and Classical Molecular Dynamics, pages 469–473. Springer International Publishing, Cham, 2015.
8. Krzysztof Rojek, Roman Wyrzykowski, and Lukasz Kuczynski. Systematic adaptation of stencil-based 3D MPDATA to GPU architectures. *Concurrency and Computation: Practice and Experience*, pages n/a–n/a, 2016.
9. H.J.C. Berendsen, D. van der Spoel, and R. van Drunen. Gromacs: A message-passing parallel molecular dynamics implementation. *Computer Physics Communications*, 91(13):43 – 56, 1995.
10. Steve Plimpton. Fast Parallel Algorithms for Short-Range Molecular Dynamics. *J. Comput. Phys.*, 117(1):1–19, March 1995.
11. C. R. Trott, L. Winterfeld, and P. S. Crozier. General-purpose molecular dynamics simulations on GPU-based clusters. *ArXiv e-prints*, September 2010.
12. W. Michael Brown, Peng Wang, Steven J. Plimpton, and Arnold N. Tharrington. Implementing molecular dynamics on hybrid high performance computers — short range forces. *Computer Physics Communications*, 182(4):898 – 911, 2011.
13. W. Michael Brown, Axel Kohlmeyer, Steven J. Plimpton, and Arnold N. Tharrington. Implementing molecular dynamics on hybrid high performance computers — Particle-particle particle-mesh. *Computer Physics Communications*, 183(3):449 – 459, 2012.
14. H. Carter Edwards, Christian R. Trott, and Daniel Sunderland. Kokkos: Enabling manycore performance portability through polymorphic memory access patterns. *Journal of Parallel and Distributed Computing*, 74(12):3202 – 3216, 2014. Domain-Specific Languages and High-Level Frameworks for High-Performance Computing.
15. Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C. Smith, Berk Hess, and Erik Lindahl. Gromacs: High performance molecular simu-

lations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, 12:19 – 25, 2015.
16. Itta Ohmura, Gentaro Morimoto, Yousuke Ohno, Aki Hasegawa, and Makoto Taiji. MDGRAPE-4: a special-purpose computer system for molecular dynamics simulations. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 372(2021), 2014.
17. Stefano Piana, John L Klepeis, and David E Shaw. Assessing the accuracy of physical models used in protein-folding simulations: quantitative evidence from long molecular dynamics simulations. *Current Opinion in Structural Biology*, 24(0):98 – 105, 2014.
18. Carsten Kutzner, Szilard Pall, Martin Fechner, Ansgar Esztermann, Bert L. de Groot, and Helmut Grubmuller. Best bang for your buck: Gpu nodes for gromacs biomolecular simulations. *Journal of Computational Chemistry*, 36(26):1990–2008, 2015.
19. Nathan Luehr, Ivan S. Ufimtsev, and Todd J. Martnez. Dynamic precision for electron repulsion integral evaluation on graphical processing units (gpus). *Journal of Chemical Theory and Computation*, 7(4):949–954, 2011. PMID: 26606344.
20. Nicholas Mills, F. Alex Feltus, and Walter B. Ligon III. Maximizing the performance of scientific data transfer by optimizing the interface between parallel file systems and advanced research networks. *Future Generation Computer Systems*, 79(Part 1):190 – 198, 2018.
21. S. J. Plimpton A. N. Tharrington W. M. Brown, P. Wang. Implementing molecular dynamics on hybrid high performance computers - short range forces. *Comp. Phys. Comm.*, 182:898–911, 2011.
22. S. J. Plimpton A. N. Tharrington W. M. Brown, A. Kohlmeyer. Implementing molecular dynamics on hybrid high performance computers - particle-particle particle-mesh. *Comp. Phys. Comm.*, 183:449–459, 2012.
23. Y. Masako W. M. Brown. Implementing molecular dynamics on hybrid high performance computers three-body potentials. *Comp. Phys. Comm.*, 184:2785–2793, 2013.
24. Nikolay D. Kondratyuk, Genri E. Norman, and Vladimir V. Stegailov. Self-consistent molecular dynamics calculation of diffusion in higher n-alkanes. *J. Chem. Phys.*, 145(20):204504, 2016.
25. Luigi Genovese, Alexey Neelov, Stefan Goedecker, Thierry Deutsch, Seyed Alireza Ghasemi, Alexander Willand, Damien Caliste, Oded Zilberberg, Mark Rayson, Anders Bergman, and Reinhold Schneider. Daubechies wavelets as a basis set for density functional pseudopotential calculations. *J.Chem.Phys.*, 129(1):014109, 2008.
26. Luigi Genovese, Matthieu Ospici, Thierry Deutsch, Jean-François Méhaut, Alexey Neelov, and Stefan Goedecker. Density functional theory calculation on many-cores hybrid central processing unit-graphic processing unit architectures. *J.Chem.Phys.*, 131(3):034103, 2009.
27. Wolfgang Eckhardt, Alexander Heinecke, Reinhold Bader, Matthias Brehm, Nicolay Hammer, Herbert Huber, Hans-Georg Kleinhenz, Jadran Vrabec, Hans Hasse, Martin Horsch, Martin Bernreuther, Colin W. Glass, Christoph Niethammer, Arndt Bode, and Hans-Joachim Bungartz. 591 TFLOPS multi-trillion particles simulation on SuperMUC. In Julian Martin Kunkel, Thomas Ludwig, and Hans Werner Meuer, editors, *Supercomputing*, volume 7905 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg, 2013.