Check for updates

International Journal of
HIGH PERFORMANCE
COMPUTING APPLICATIONS

PPAM 2017 Special Issue

The International Journal of High
Performance Computing Applications
2019, Vol. 33(3) 507–521
© The Author(s) 2019
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/1094342019826667
journals.sagepub.com/home/hpc
SAGE

Angara interconnect makes GPU-based Desmos supercomputer an efficient tool for molecular dynamics calculations

Vladimir Stegailov[1,2], Ekaterina Dlinnova[2], Timur Ismagilov[3], Mikhail Khalilov[2], Nikolay Kondratyuk[1,2], Dmitry Makagon[3], Alexander Semenov[2,3], Alexei Simonov[3], Grigory Smirnov[1,2] and Alexey Timofeev[1,2]

Abstract

In this article, we describe the Desmos supercomputer that consists of 32 hybrid nodes connected by a low-latency high-bandwidth Angara interconnect with torus topology. This supercomputer is aimed at cost-effective classical molecular dynamics calculations. Desmos serves as a test bed for the Angara interconnect that supports 3-D and 4-D torus network topologies and verifies its ability to unite massively parallel programming systems speeding-up effectively message-passing interface (MPI)-based applications. We describe the Angara interconnect presenting typical MPI benchmarks. Desmos benchmarks results for GROMACS, LAMMPS, VASP and CP2K are compared with the data for other high-performance computing (HPC) systems. Also, we consider the job scheduling statistics for several months of Desmos deployment.

Keywords

HPC interconnect, scalability, efficiency, GPU-accelerated software, atomistic simulations

1. Introduction

Rapid development of parallel computational methods and supercomputer hardware provide great benefits for atomistic simulation methods. At the moment, these mathematical models and computational codes are not only the tools of fundamental research but the more and more intensively used instruments for diverse applied problems (Heinecke et al., 2015). For classical molecular dynamics (MD), the limits of the system size and the simulated time are trillions of atoms (Eckhardt et al., 2013) and milliseconds (Piana et al., 2014) (i.e. 10^9 steps with a typical MD step of 1 femtosecond).

There are two main ways of hardware acceleration for MD calculations (as, in fact, for many other algorithms of mathematical modelling and simulation).

The first possibility consists in the increase of the computing capabilities of individual nodes of massively parallel programming (MPP) systems. Multi-CPU and multi-core shared-memory node architectures provide essential acceleration. However, the scalability of shared memory systems is limited by their cost and speed limitations of DRAM access for multi-socket and/or multi-core nodes. It is the development of GPGPU that boosts the performance of shared-memory systems.

The Nvidia CUDA technology was introduced in 2007 and provided a convenient technique for GPU programming. 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 Smirnov and Stegailov, 2016; Stegailov et al., 2016; Rojek et al., 2016). The sustained performance is usually limited by the memory-bound nature of the algorithms.

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 (Ohmura et al., 2014) or ANTON (Piana et al., 2014). Currently, general-purpose Nvidia GPUs are considered as the most cost-effective hardware for MD calculations (Kutzner et al., 2015).

[1] Joint Institute for High Temperatures of Russian Academy of Sciences, Moscow, Russia
[2] National Research University Higher School of Economics, Moscow, Russia
[3] JSC NICEVT, Moscow, Russia

Corresponding author:
Vladimir Stegailov, Joint Institute for High Temperatures of Russian Academy of Sciences, Izhorskaya St. 13 Bldg 2, Moscow 125412, Russia.
Email: v.stegailov@hse.ru

The second possibility for MD calculations acceleration is the use of distributed memory MPP systems. For MD calculations, domain decomposition is a natural technique to distribute both computational load and data across nodes of MPP systems (Begau and Sutmann, 2015).

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.

Torus topologies of the interconnect have several attractive aspects in comparison with fat-tree topologies. In 1990s, the development of MPP systems had its peak during the remarkable success of Cray T3E systems based on the 3D torus interconnect topology (Scott et al., 1996) that was the first supercomputer that provided 1 TFlops of sustained performance. In June 1998, Cray T3E occupied 4 of top 5 records of the Top 500 list. In 2004, after several years of the dominance of Beowulf clusters, a custom-built torus interconnect appeared in the IBM BlueGene/L supercomputer (Adiga et al., 2005). Subsequent supercomputers of Cray and IBM had torus interconnects as well (with the exception of the latest Cray XC series based on the Aries interconnect with the Dragonfly topology). Fujitsu designed K Computer based on the Tofu torus interconnect (Ajima et al., 2012). The AURORA Booster and GREEN ICE Booster supercomputers are based on the EXTOLL torus interconnect (Neuwirth et al., 2015).

The development of a new type of supercomputer oriented interconnect hardware is a complex endeavour requiring simultaneous development of the corresponding software stack. This software stack should enable correct and efficient operation of all the software packages of end users. From the economical point of view, the usage of two main types of presently commercially available interconnects (Mellanox Infiniband and Intel Omni-Path) is always cheaper than the development of a new technology. Due to these complexity of development and economic considerations, the number of supercomputer interconnect types is quite limited. Among the novel types of interconnects in addition to those listed above, we can mention, for example, the bull eXascale Interconnect (BXI) (Derradji et al., 2015) and the TaihuLight interconnects (Lin et al., 2017).

In this work, we describe the Desmos supercomputer that is the first public high-performance computing (HPC) system based on the Angara interconnect. Desmos is based on inexpensive 1CPU + 1GPU nodes, the performance of which has been chosen with respect to MD calculations (Stegailov et al., 2017). The capabilities of the Angara interconnect are illustrated in this work using Desmos results both for synthetic benchmarks and for real-life models. Also, we discuss the statistics of the supercomputer deployment using novel graphical representations.

2. Related work

Among the references to the recent developments of original types of supercomputer interconnects in Russia, we can mention the MVS-Express interconnect based on the PCIe bus (Elizarov et al., 2012), the field-programmable gate array (FPGA) prototypes of the SKIF-Aurora (Adamovich et al., 2010) and Pautina (Klimov et al., 2015) torus interconnects. Up to this moment, the Angara interconnect has been evolving all the way from the FPGA prototype (Korzh et al., 2010; Mukosey et al., 2015) to the ASIC-based card (Agarkov et al., 2016).

Torus topology is believed to be beneficial for strong scaling of many parallel algorithms. However, the accurate data that verify this assumption are quite rare. Probably, the most extensive work was done by Fabiano Corsetti who compared torus and fat-tree topologies using the SIESTA electronic structure code and six large-scale supercomputers belonging to the PRACE Tier-0 network (Corsetti et al., 2014). The author concluded that machines implementing torus topologies demonstrated a better scalability to large system sizes than those with fat-tree topologies. The comparison of the benchmark data for the CP2K electronic structure code showed a similar trend (Stegailov et al., 2016).

The performance analysis of algorithms and tuning for hybrid-based computing environments attracts considerable attention in various application fields (e.g. see D'Amore et al., 2015; Laccetti et al., 2014; Montella et al., 2017; Rojek et al, 2016).

Among GPU-aware MD software one can point out GROMACS (Berendsen et al., 1995) as, perhaps, the most computationally efficient MD tool and LAMMPS (Plimpton, 1995) as one of the most versatile and flexible for MD models creation. Different GPU off-loading schemes were implemented in LAMMPS (Brown et al., 2011; Brown et al., 2012; Edwards et al., 2014; Trott et al., 2010). GROMACS provides a highly optimized GPU-scheme as well (Abraham et al., 2015).

MD benchmarks are among the most popular tests for supercomputers. For example, ApoA1 (protein in water) is a widespread model, and we can find the corresponding benchmark data for Cray XK6 (Sun et al., 2012), for IBM BlueGene/P and BlueGene/Q systems (Kumar et al., 2013). The AURORA Booster supercomputer was benchmarked using LAMMPS (Neuwirth et al., 2015).

Job scheduling determines the efficiency of practical deployment of a supercomputer and is a very important topic in the context of HPC systems optimization studies (see, e.g. Gómez-Martín et al., 2016). Everyday usage of supercomputer centres shows the need for the separation of the cloud-like jobs (that do not require a high-bandwidth low-latency interconnect between nodes) from the multi-node parallel jobs (Caruso et al., 2005; Murli et al., 2017). Such a separation is a way for increasing efficiency of supercomputer deployment (Kraemer et al., 2016). There are attempts of statistical analysis of supercomputers

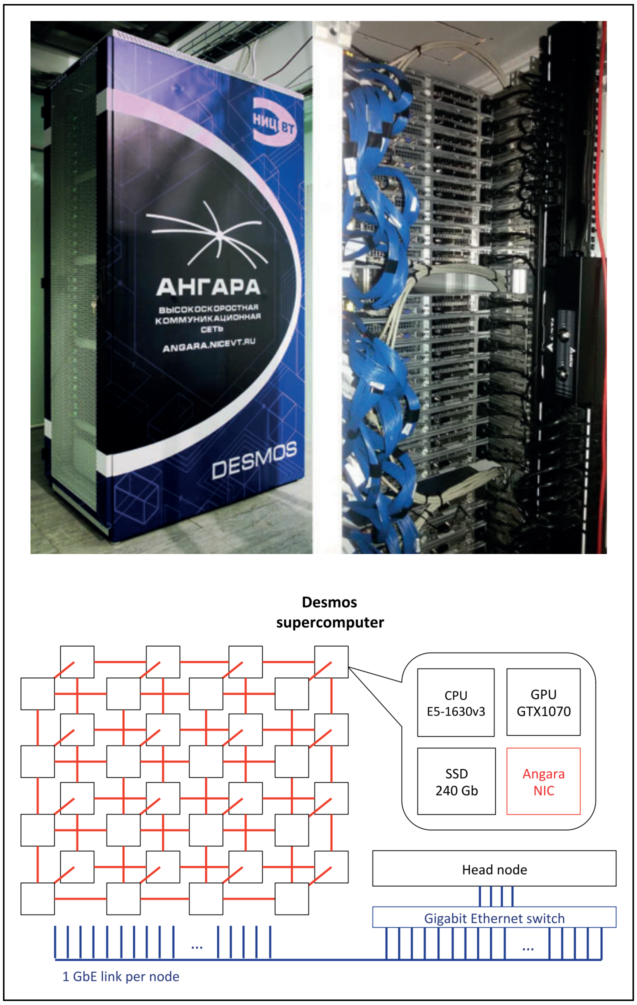


Figure 1. The Desmos supercomputer: the photos and the scheme.

operation (e.g. see Barone et al., 2017; Mamaeva and Voevodin, 2017).

3. The Desmos supercomputer

Figure 1 shows the photo and the scheme of the Desmos supercomputer that resides in one standard rack. The hardware components have been selected to maximize the number of nodes and the efficiency of a one node for MD workloads. Each node consists of Supermicro SuperServer 1018GR-T, Intel Xeon E5-1650v3 (6 cores, 3.5 GHz) and Nvidia GeForce 1070 (8 GB GDDR5) and has DDR4-2133 (16 GB) operating at 1866 MHz.

The Nvidia GeForce 1070 cards have no error-correcting code memory in contrast to professional accelerators. For this reason, it was necessary to make sure that there is no hardware memory errors in each GPU. Testing of each GPU was performed using MemtestG80 (Haque and Pande, 2010) during more than 4 h for each card. No errors were detected for 32 cards considered.

The cooling of the GPU cards was a special question. We used ASUS GeForce GTX 1070 8 GB Turbo Edition. Each card was partially disassembled prior to installation into 1U chassis. The plastic cover and the dual-ball bearing fan were removed that made the card suitable for horizontal air flow cooling inside chassis.

The nodes are connected by Gigabit Ethernet and Angara interconnect (in the 4D-torus $4 \times 2 \times 2 \times 2$, copper Samtec cables). Due to budget limitations, we did not use all possible ports for the full 4D-torus topology. The currently implemented topology is 4D-torus $4 \times 2 \times 2 \times 2$ $(X \times Y \times Z \times K)$ but each node along Y, Z, K dimensions is connected to another node by one link only.

There is a front-end node with the same configuration as all 32 computing nodes of the supercomputer (the front-end node is connected to GigE only). The supercomputer is running under SLES 11 SP4 with Angara MPI (based on MPICH 3.0.4).

The supercomputer energy consumption is 6.5 kW in the idle state and 14.4 kW under full load.

4. The Angara interconnect

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 (Neuwirth et al., 2015).

The Angara interconnect supports 4D-torus network topology. Deadlock-free deterministic routing is performed according to the bubble rule while preserving the order of directions $+X + Y + Z + W - X - Y - Z - W$ (Adiga et al., 2005; Puente et al., 1999; Scott et al., 1996). Five virtual channels are used: two channels for deterministic routing (blocking request channel and non-blocking response channel); a separate virtual channel is used for adaptive routing (with the ability to switch to a non-blocking deterministic channel in case of potential deadlock); two more virtual channels are used to send messages over a virtual subnet for collective operations. The collective operations (broadcast and reduce) support is implemented using a virtual subnet with tree topology, which is applied to multidimensional torus topology.

The Angara chip supports simultaneous operations with multiple threads/processes of one task; it is implemented as several injection channels available for use by independent packet buffers. The fault-tolerant packet transmission is supported at the data link layer. There is also a mechanism for bypassing the failed links and nodes by reorganizing the routing tables and using non-standard first/last steps of the package route (Scott et al., 1996).

The network adapter supports RDMA read and write operations and atomic operations. The adapter supports two types of atomic operations: addition and XOR. Each node has a dedicated memory region available for remote access from other nodes to support OpenSHMEM and PGAS.

Table 1. The comparison of the Desmos supercomputer with the Lomonosov-2 supercomputer used as a reference system for MPI benchmarks.

Cluster	Desmos	Lomonosov-2
Chassis	SuperServer 1018GR-T	T-Platforms A-Class
Processor	E5-1650v3 (6c, 3.5 GHz, 15 MB cache)	E5-2697v3 (14c, 3.3 GHz Turbo, 35 MB cache)
GPU	Nvidia GeForce GTX 1070	NVidia Tesla K40M
Memory	DDR4 16 GB (1866 MHz)	DDR4 64 GB (2133 MHz)
Number of nodes	32	1 698
Interconnect	Angara 4D-torus 4 × 2 × 2 × 2	Infiniband 4 × FDR Flattened Butterfly
Operating system	SLES 11 SP4	CentOS 7.1.150
Compiler	GNU CC 4.8.3	Intel Parallel Studio XE 2015
MPI	Angara MPI (based on MPICH 3.0.4)	Open MPI 2.1.1

FDR: Fourteen Data Rate.

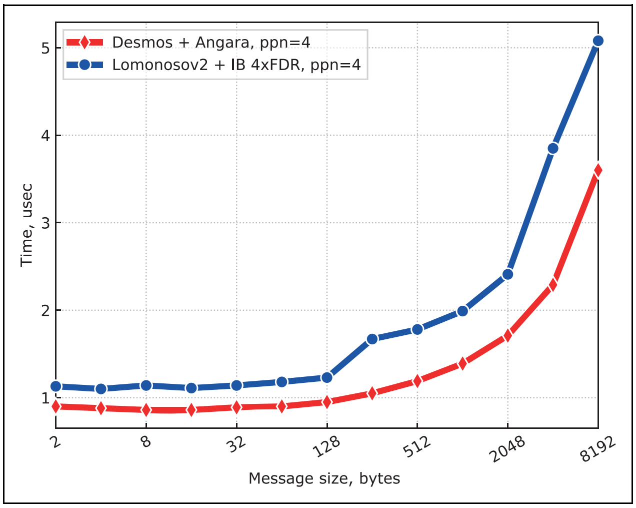


Figure 2. The OSU latency between two adjacent nodes.

Multiple programming models are supported, including MPI and OpenMP.

The network adapter extension card can be connected to the card in adjacent nodes by up to six cables (or up to eighr with an extension card). The following topologies are supported: a ring, 2-D, 3-D and 4-D tori.

To provide more insights into Angara communication behaviour, we present a performance evaluation comparison of the Desmos supercomputer and the Lomonosov-2 supercomputer with the Mellanox Infiniband 4× Fourteen Data Rate (FDR) interconnect (with the Flattened Butterfly topology). FDR InfiniBand (14 Gb/s data rate per lane) is the generation of InfiniBand technology developed and specified by the InfiniBand Trade Association in 2011. Table 1 compares these systems. Both systems are equipped with single socket Haswell CPUs, but the processors' characteristics differ very much.

OpenMPI on Lomonosov-2 is used with the Yalla point-to-point message layer, which reduces overhead by cutting through communication layers and using MXM (Mellanox Messaging Accelerator) directly. No other tuning of MPI on Lomonosov-2 is used.

Figure 2 shows the latency results obtained by the OSU Micro-Benchmarks test. The Angara hardware implementation is optimized for high performance computing applications, the latency of a hop (the router latency and the link latency) is extremely low (129 ns). This fact and high frequency of CPUs of the Desmos supercomputer lead to 0.85 μs OSU latency for the 16 B message size, and this latency is better than the corresponding values for 4× FDR interconnect for all small message sizes. The Angara interconnect bandwidth for large messages is the subject of the non-disclosure agreement.

We use Intel MPI Benchmarks to evaluate MPI_Alltoall operation time for different number of nodes of the Desmos and the Lomonosov-2 supercomputers. The Angara superiority for small messages explains better results for MPI_Alltoall with 16 B messages (Figure 3). For large messages (256 KB) the RDMA Angara operation used in the MPI library implementation as well as the MPI_Alltoall collective operation are better tuned on Desmos for 16 and 32 nodes in comparison with the Lomonosov-2 supercomputer (Figure 3).

We use NAS Parallel Benchmarks NPB 3.3.1 (LU, MG, CG, FT, IS tests, class C) for the comprehensive application-level performance evaluation of the Desmos cluster. FT and IS are communication bound kernels that depend on the all-to-all communication pattern. For 16 and 32 nodes FT and IS performance are better on Desmos than on Lomonosov-2 due to the showed collective operations performance (Figure 4).

CG, MG and LU tests results are in Figure 5. MG is a memory bound test, performance results on the Lomonosov-2 supercomputer outperform Desmos results because memory frequency on the Desmos cluster is slower than on the Lomonosov-2 supercomputer. CG is a memory and communication bound test, the results are the same for 1, 2 and 4 nodes on both supercomputers. For 8, 16 and 32 nodes there is superlinear speed-up on both supercomputers, it seems that the problem size (class C) is too small. LU is a compute bound test, and the results are similar on the Desmos and the Lomonosov-2 supercomputers.

So we can see that the low-latency Angara interconnect and the RDMA engine successfully perform all-to-all

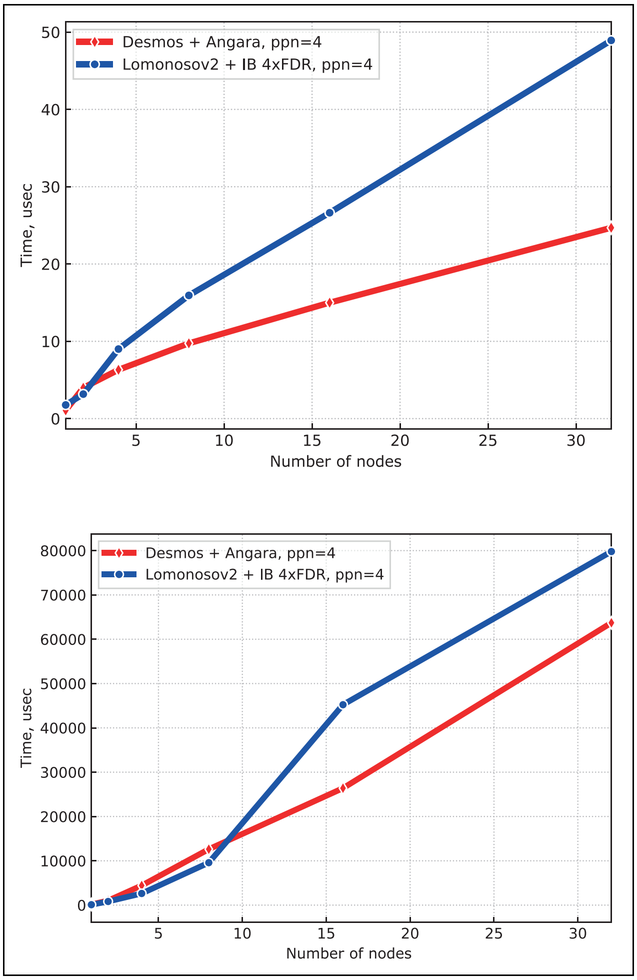


Figure 3. Times for MPI_Alltoall with four processes per node and message sizes 16 B and 256 kB.

collective operations and allow the Desmos supercomputer to outperform the Lomonosov-2 supercomputer on communication bound benchmarks for the corresponding number of cores used.

5. Mapping of MPI processes for Angara interconnect

The algorithm for optimizing the mapping of MPI processes, based on the partitioning of communication pattern graph (Hoefler and Snir, 2011), is adapted for the usage with the Angara interconnect. The main idea of the method for optimizing the process mapping is to split the communication pattern graph into the disjoint subsets of the intensively exchanging processes and to bind these subsets to nodes/cores connected by the fastest communication channels. The partition is performed first for inter-node level and then for intra-node level.

For the prototype dual-socket supercomputer with Angara (located in JSC NICEVT), the algorithm allows to reduce up to 25% of the execution time on the NPB LU and SP benchmarks (Figure 6). Most of the optimization is related to the transfer of calculations to the first socket, and it is not the effect of topology optimization.

The repartition of the communication graph for NAS Parallel Benchmarks (LU, SP, FT) and GROMACS benchmarks does not show a significant improvement on a small scale (e.g. for the 32 single-socket compute nodes in the case of the Desmos supercomputer). The proposed method shows minor improvements with a small number of processors and a symmetric pattern of the benchmark.

A theoretical model for the dependence of the application execution acceleration on the supercomputer size and topology is constructed to assess the effectiveness of this optimization method for supercomputers of bigger size and different topologies. In this way, the dependence of the relative decrease in execution time on the number of computational nodes with selected topology can be roughly estimated (without consideration of bandwidth, congestion, etc.). This model is based on the following assumptions:

- The simplest zero-message transfer can be described by the formula

$$t(0) = t_{\text{injection}} + n_{\text{hops}} \cdot t_{\text{hop}} + t_{\text{ejection}}$$

where $t_{\text{injection}}$ and t_{ejection} are the packet handling times during sending and reception by the Angara adapter, n_{hops} is the number of links in packet transfer route, t_{hop} is time for packet processing on each hop (Rauber and Runger, 2013). According to the *osu_latency* test from the OSU micro-benchmarks, $t_{\text{injection}} + t_{\text{ejection}} = 850$ ns and $t_{\text{hop}} = 129$ ns for the Desmos supercomputer.

- Each process node is associated with a process that exchanges messages in the pair with another random process.
- The distances (in hops) between the processes are described by the standard normal distribution. The average number of hops between two random processes $N_{\text{hops}}^{\text{average}} = d_{\text{net}}/2$, where d_{net} is the diameter of network.
- In an optimized case, the exchanging pairs of processes would be mapped either to the same node or to the nodes connected directly.
- The dependence of the decrease Δt in execution time t_{sum} on the number of computational nodes with selected topology is calculated by the formula

$$\frac{\Delta t}{t_{\text{sum}}} = \frac{p_{\text{mpi}}}{100\%} \cdot \frac{t_{\text{hop}} \cdot (d_{\text{net}} - 1)}{t_{\text{ejection}} + t_{\text{injection}} + t_{\text{hop}} \cdot d_{\text{net}}}$$

It is worth nothing that the gain from optimization is estimated at a fixed parameter p_{mpi} for different message sizes.

Figure 7 shows the results of the process mapping implementation algorithm of this model for different topologies and the comparison of experimental results of the NAS parallel benchmarks (NPB) and GROMACS benchmarks optimization using 4-32 Desmos nodes with the proposed

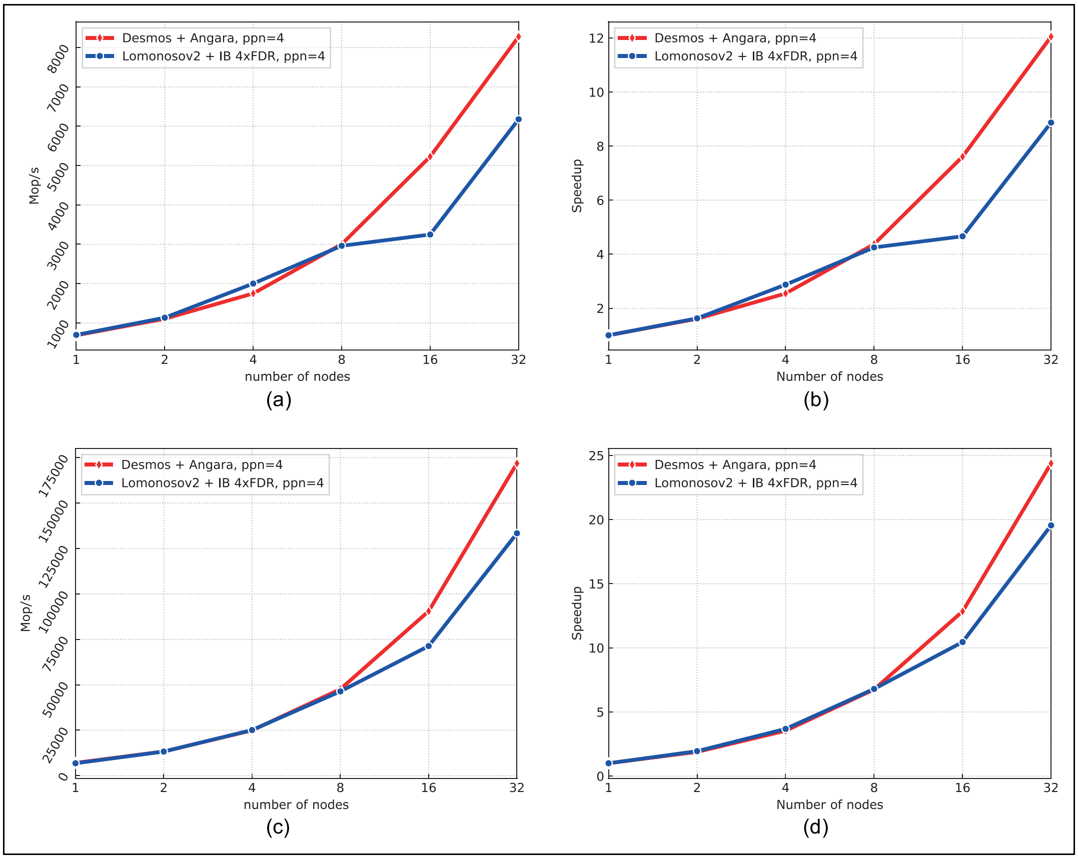


Figure 4. NAS Parallel Benchmarks on Desmos and Lomonoso v2. Four processes per node are used. IS test: performance (a), speedup (b). FT test: performance (c), speedup (d).

theoretical model for 2-512 similar nodes of a hypothetical supercomputer.

The percentage of message transmission time p_{mpi} in total running time can be taken close to 30% for a usual MD problem. $p_{\mathrm{mpi}} \approx 5\%$ for NPB LU benchmark, which leads us to an estimation of the maximum effect of optimization at the level of 1%. The effect of mapping of MPI processes of MD applications is expected to be close to 2% for 32 single-socket nodes on 4-D-torus cluster with Angara interconnect comparable in performance with Desmos and is close to 5% for a system with 256 nodes, according to the theoretical model considered. The effect is greater for less connected topologies. For example, the obtained gain is greater than 10% even for a 64 node supercomputer with 2D-torus topology.

This estimates justify the efficiency of possible topology-aware MPI-mapping algorithms to be developed in the future for supercomputers based on the Angara interconnect.

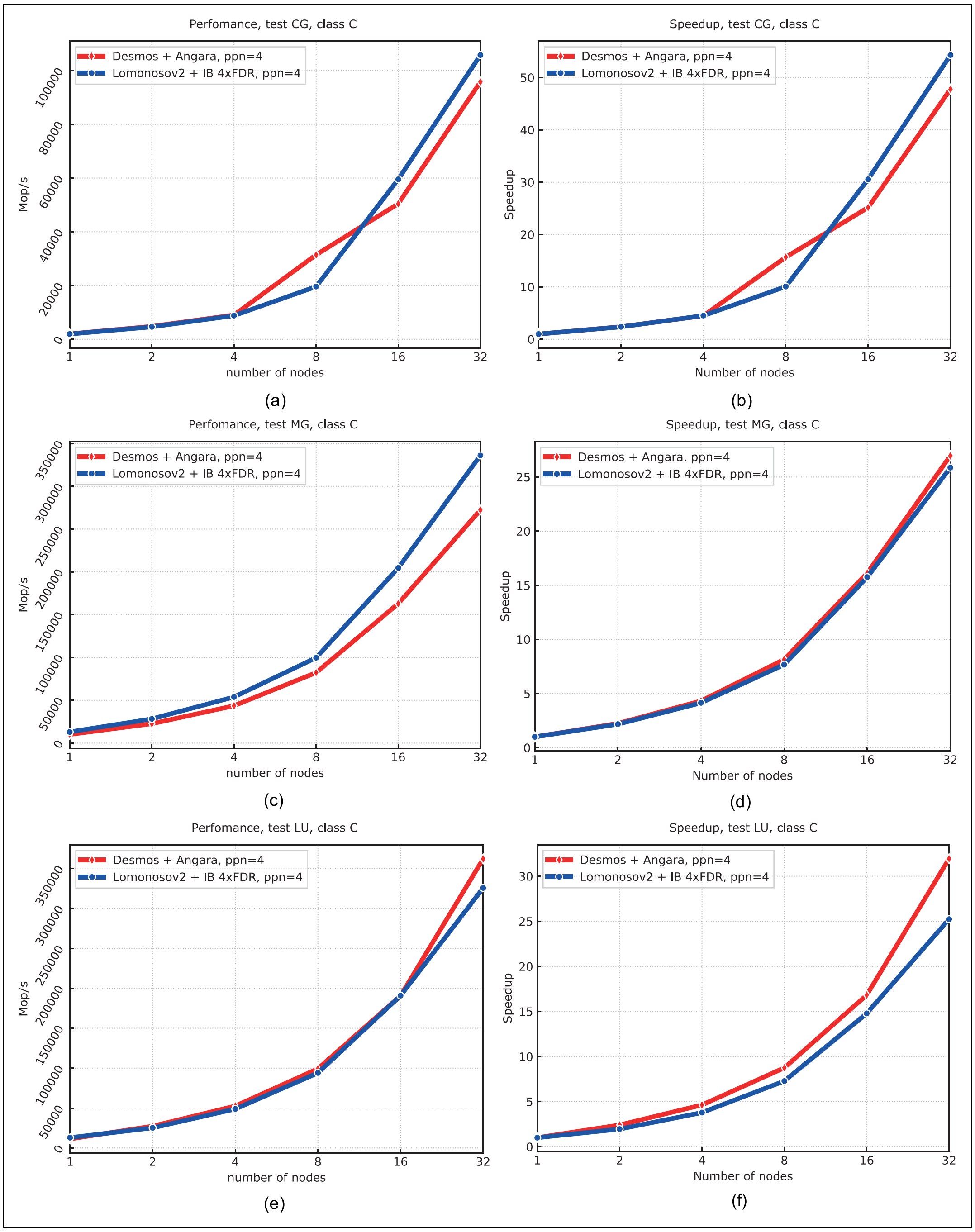
6. Classical MD benchmarks

Nowadays, there is no novelty in the partial use of single precision in MD calculations with consumer-grade GPUs. The results of such projects as Folding@Home confirmed the broad applicability of this approach. Recent developments of optimized MD algorithms include the validation of the single precision solver (e.g. Höhnerbach et al., 2016). In this study, we do not consider the questions of accuracy and limit ourselves to the benchmarks of computational efficiency.

The first set of benchmark data is shown on Figure 8. It illustrates the efficiency of LAMMPS, LAMMPS with the GPU package (with mixed precision) or LAMMPS with USER-INTEL package running on one computational node for different numbers of atoms in the MD model (Kondratyuk et al., 2016). The USER-INTEL package provides SIMD-optimized versions of certain interatomic potentials in LAMMPS. As expected, the CPU + GPU pair shows maximum performance for sufficiently large system sizes. The results for the Desmos node are compared with the results for a two-socket node with 14-core Haswell CPUs (the MVS1P5 supercomputer).

We see that for the simple Lennard-Jones (LJ) liquid benchmark the GPU-version of LAMMPS on one Desmos node provides 10–15% higher times-to-solution than the SIMD-optimized LAMMPS on 2 × 14-core Haswell CPUs. However, for the liquid $C_{30}H_{62}$ oil benchmark, the GPU-version of LAMMPS is faster starting from 100,000 atoms per node. For comparison, we present the results for the same benchmark on the professional grade Nvidia Tesla K80.

An apolipoprotein A1 (ApoA1) in water MD model is a popular benchmark system with approximately



(a)

(b)

(c)

(d)

(e)

(f)

Figure 5. NAS Parallel Benchmarks on Desmos and Lomonosov-2. Four processes per node are used. CG test: performance (a), speedup (b). MG test: performance (c), speedup (d). LU test: performance (e), speedup (f).

100,000 atoms. Figure 9 shows the weak scaling results for the LJ system, for the $C_{30}H_{62}$ oil model and for the ApoA1 system. *Weak scaling* is the standard approach to benchmark the supercomputer interconnect increasing proportionally the systems size of a model considered and the number of the nodes deployed. We see that weak scaling becomes better for larger models. Data for the case of reduced DRAM bandwidth are shown with dashed lines (when two of four memory channels of CPUs are populated).

The time-to-solution (τ) criterion leads us to the evident choice of a time for one MD integration step (for the whole MD model or per atom) or a time for one iteration during electronic structure calculation as one parameter for the performance metric. The second parameter should characterize the hardware. Usually the number of some abstract processing elements (e.g. cores) is considered. However, although this metric serves well in the *weak* and *strong* scaling benchmarks for the given system, it does not allow to compare essentially different hardware. To overcome

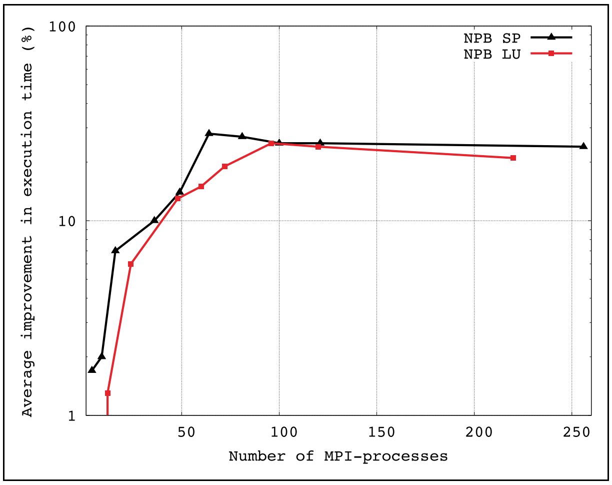


Figure 6. The results of the process mapping optimization of NAS Parallel Benchmarks SP and LU on the prototype dual-socket supercomputer with the Angara interconnect.

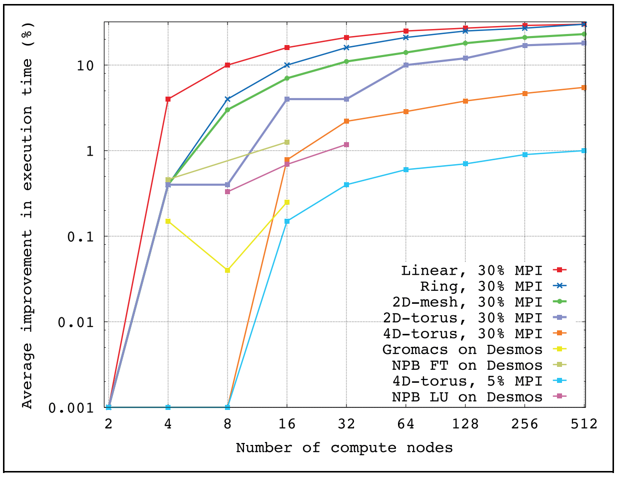


Figure 7. The theoretical model results and the benchmark results of process mapping optimization on the Desmos supercomputer.

this problem, we consider the total peak performance R_{peak} as a second parameter for the metric that put on equal footing all HPC hardware under consideration (Stegailov et al., 2016).

Formally, we can express this metric as

$$\tau = \tau\left(R_{\text{peak}}(N_{\text{nodes}})\right)$$

with the ideal scaling being $\tau = A/R_{\text{peak}}$, where A is a constant.

Since the Desmos supercomputer rely heavily on single-precision performance, we need to devise some reasonable conversion of SP Flops to DP Flops. Here we take $R_{\text{peak}}^{\text{node}} = 6 \times 3.5 \times 16 + 0.5 \times R_{\text{peak}}^{\text{GPU-SP}} = 336 + 2892 = 3228$ GFlops.

The performance of different supercomputers based on ApoA1 test is shown in Figure 10 in terms of seconds per 1 atom for 1 MD step and the declared peak performance.

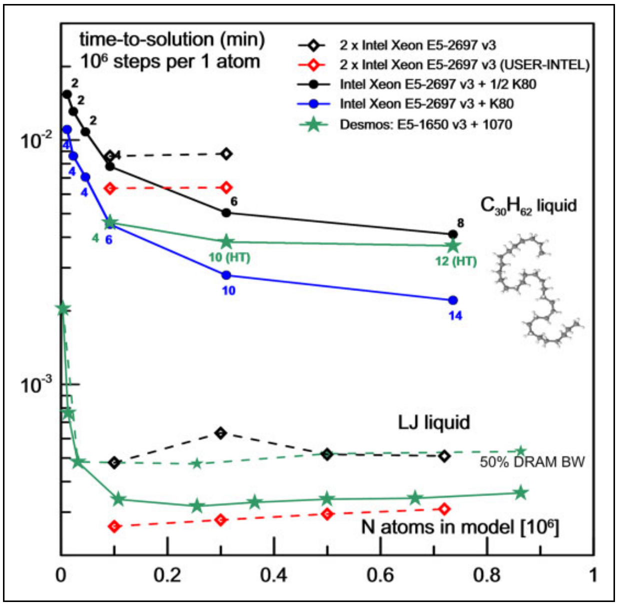


Figure 8. Time per atom per timestep for LJ liquid and C 30 H 62 oil LAMMPS MD models of different size and different hardware combinations (the numbers near symbols show the number of MPI threads used). LJ: Lennard-Jones; MD: molecular dynamics.

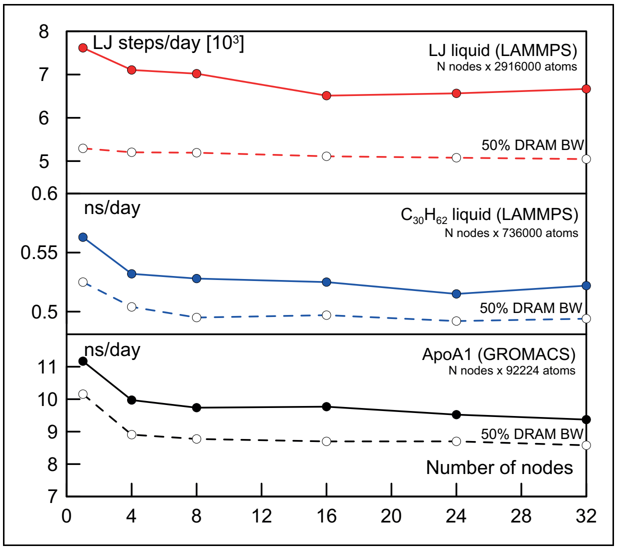


Figure 9. Weak scaling benchmarks for the Desmos supercomputer and three different MD models. MD: molecular dynamics.

The dotted line shows ideal scalability with performance 0.1 MFlops/atom/step.

Despite different architectures and MD software, these data can be used to analyse the efficiency of supercomputer systems for biomolecular MD studies. The black and green lines with dots show the performance of the CPU supercomputer with Intel Xeon nodes (Smirnov and Stegailov, 2016). The new Desmos supercomputer (the purple filled stars) with GROMACS shows better scalability then Cray XK6 (Sun et al., 2012) and the K Computer with NAMD. It shows better efficiency than

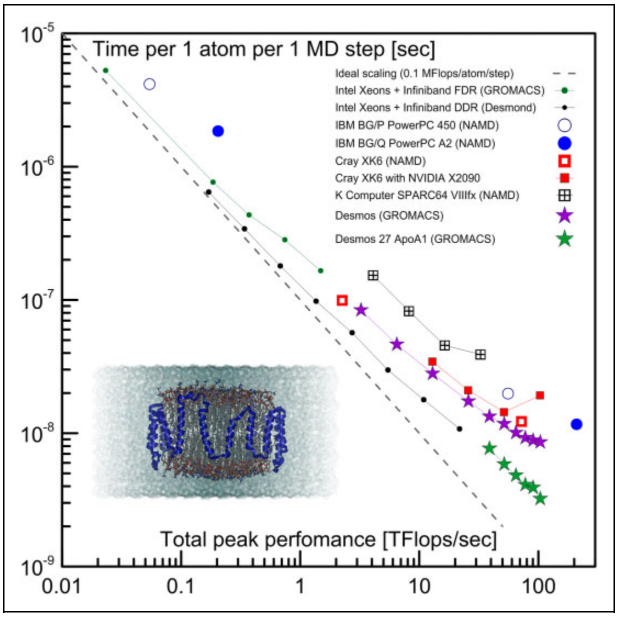


Figure 10. ApoA1 benchmark results obtained with GROMACS for the Desmos cluster (including the 3 × 3 × 3 replicated model). For comparison we show the data for the same benchmark but obtained with NAMD for different supercomputers: Cray XK6 (Sun et al., 2012), IBM BlueGene/P and BlueGene/Q (Kumar et al., 2013), K Computer (Akinori Yonezawa, 2012).

the benchmarks for BlueGene systems (Kumar et al., 2013).

The green stars show the 27 times replicated ApoA1 system benchmark for the Desmos cluster. These data show the continuous scaling with larger number atoms per node. It indicates the possibility for adding more nodes to Desmos without noticeable efficiency degradation.

The work by Kutzner et al. (2015) gives very instructive guidelines for achieving the best performance for the minimal price in 2015. Authors compared different configurations of clusters using two biological benchmarks: the membrane channel protein embedded in a lipid bilayer surrounded by water (MEM, ~100,000 atoms) and the bacterial ribosome in water with ions (RIB, ~2,000,000 atoms). The GROMACS package was used for all tests. We follow the same protocol but use longer MD simulations than in the study by Kutzner et al. (2015) to obtain the best performance.

The cost of each node in Figure 11 consists of the price of the computational resources with corresponding infrastructure excluding the costs of interconnect. The prices of single nodes are estimated using prices form the website ThinkMate.com at the end of November 2017.

A Desmos node without GPU costs about US$2600, a Desmos node with one GTX 1070 costs US$3100. An IRUS17 node with two Intel Xeon E5-2698 v4 costs effectively US$11000 (IRUS17 consists of dual-node blades in the enclosure), An IRUS17 node with two Intel Xeon E5-2699 v4 costs US$13000. The labels on Figure 11 show the amount of nodes. The cost is multiplied by the corresponding number of nodes.

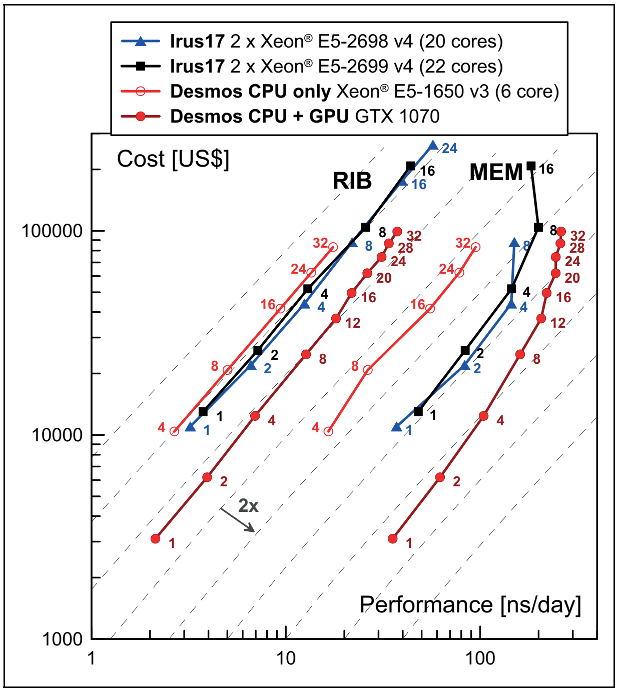


Figure 11. Comparison of the supercomputers Desmos and IRUS17 on two biomolecular benchmarks (RIB – 2,000,000 atoms, MEM – 82,000 atoms, see Kutzner et al., 2015).

We see that Desmos is ahead of IRUS17 for these benchmarks both in terms of maximum attainable speed of calculation (ns/day) and in terms of cost-efficiency.

7. Benchmarks with the electronic structure codes VASP and CP2K

The main computing power of Desmos cluster consists in GPU and is aimed at classical MD calculations. Each node has only one 6-core processor. However, it is interesting methodically to produce scaling tests for density functional theory (DFT) calculations of electronic structure. DFT calculations are highly dependent on the speed of collective all-to-all exchanges in contrast to classical MD models. We use two of the most used DFT packages VASP (Kresse and Hafner, 1993, 1994; Kresse and Furthmuller, 1996a, 1996b;) and CP2 K (Hutter et al., 2014).

The GaAs crystal model consisted of 80 atoms is used for the test in VASP (Cytowski, 2013). The results of tests are shown on Figure 12. The time for one iteration of self-consistent electronic density calculation is shown depending on the peak performance. The data for different clusters are presented: MVS10P and MVS1P5 supercomputers of Joint Supercomputer Centre of Russian Academy of Sciences and Boreasz IBM 775 of Warsaw University. The numbers near the symbols are the numbers of nodes for each test. The lower of two points showing the computing time on 32 Desmos nodes corresponds to additional parallelization over k-points.

For the CP2K benchmarks the standard water model is used. The number of molecules is varied from 32 to 1024

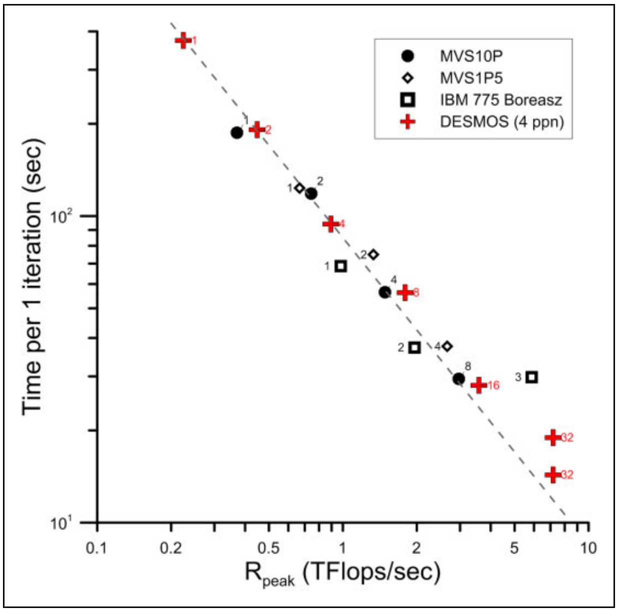


Figure 12. The scaling of the GaAs crystal model in VASP.

(benchmarks with larger systems are not possible due to limitations of DRAM on Desmos). The results of the benchmarks are shown on Figure 13.

The obtained results show that the Angara network very effectively unites the supercomputer nodes together. The supercomputers that are used for comparison have two-socket nodes (IBM 775 has even four-socket nodes). Nevertheless, MPI-exchanges over the Angara network in terms of resulting performance give the same result as MPI-exchanges in shared memory.

8. Statistical data of Desmos deployment

The batch system for user jobs scheduling of Desmos is based on Slurm. Slurm is an open-source workload manager designed for Linux clusters of all sizes (Yoo et al., 2003). Slurm has the following main functions

- it allocates exclusive and/or non-exclusive access to resources (Compute Nodes) to users for some duration of time so they can perform work,
- it provides a framework for starting, executing, and monitoring work (normally a parallel job) on the set of allocated nodes,
- it arbitrates conflicting requests for resources by managing a queue of pending work.

SlurmDB daemon stores data into a MySQL database. The SlurmDB daemon runs on the management node. In September 2017, the SlurmDB database has been activated on Desmos that has given us the possibility of detailed analysis of the supercomputer load statistics. The default Slurm tool *sreport* has quite a limited functionality. That is why we use SQL queries for accessing the SlurmDB for statistical analysis.

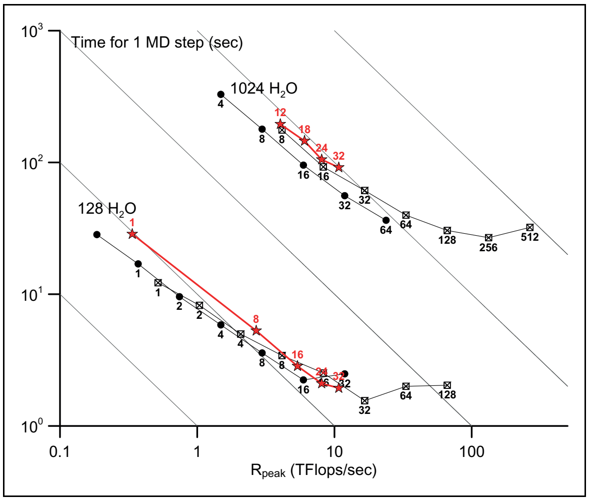


Figure 13. The scaling of the water models in CP2K.

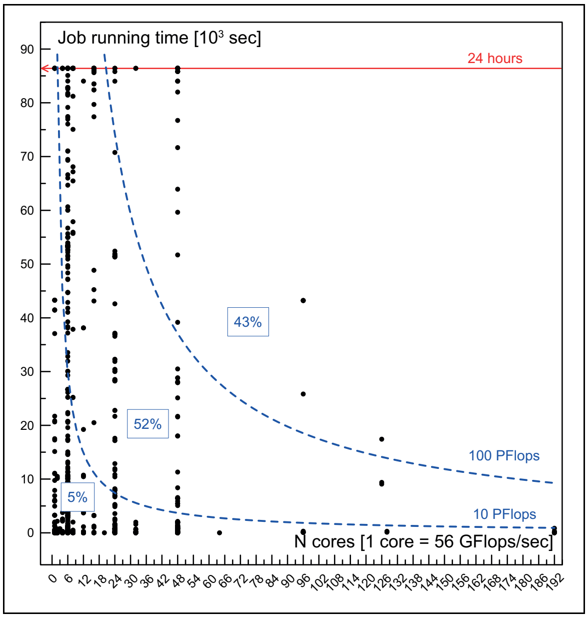


Figure 14. Job running time versus job size. Each point corresponds to one job.

The analysis of user jobs statistics has many aspects. Here, we propose a simple graphical representation that illustrates the distribution of jobs (i) by computational time and job size and (ii) by computational time and waiting time.

Figure 14 presents the distribution of jobs over the number of cores used and over running time t_R. GPU floating point performance is not taken into account in drawing the iso-levels of constant total peak number of floating-point operations $R_{\text{peak}} \times t_R$.

Parallel algorithms can be solved either slowly on a modest number of cores (nodes) or quickly if their parallel scalability justifies using a large number of processing elements efficiently. Two iso-levels separates three regions of total number floating-point operations corresponding to individual jobs: less than 10 PFlops, between

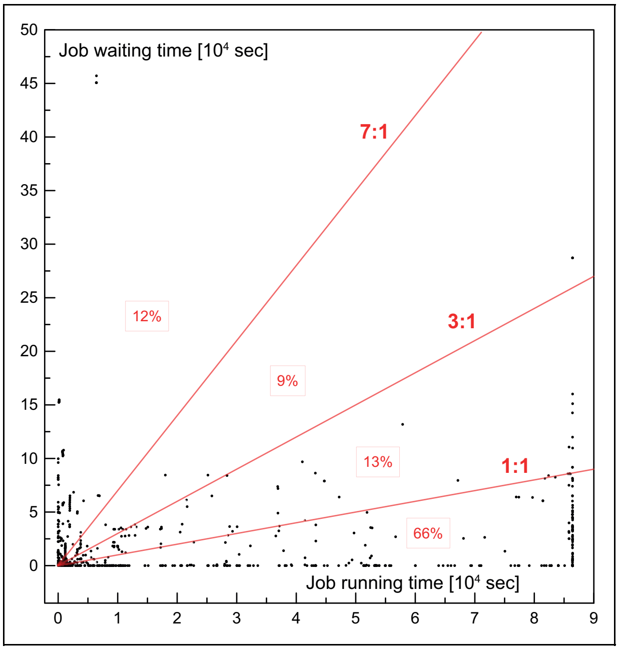


Figure 15. Job waiting time versus job running time. Each point on the figure corresponds to one job. The percent values shown in the red boxes correspond to the share of each region in the total workload of the supercomputer.

10 and 100 PFlops and above 100 PFlops. The percent values shown in the blue boxes correspond to the share of each region in the total workload of Desmos since the beginning of SlurmDB logging. We see that the major part of all the jobs executed on Desmos have been essentially supercomputer-type jobs.

At the same time, we see that there are jobs that were executed on six cores or less, that is, on a single node. This type of jobs can be easily moved away from the supercomputer either to the cloud or to a personal workstation.

The efficiency of the supercomputer job scheduling policy can be evaluated by such type of graphs. The more points we see on the right side of the graph, the more efficient is the end-user collective deployment of the supercomputer. Users should be motivated to use scalable codes and to choose larger number of nodes for speeding up calculations. The following Slurm batch system partitions have been created on the supercomputer Desmos:

- test: max time = 10 min,
- max1n: max time = 1440 min, min/max number of nodes = 1, nodes 25–32 only,
- max8n: max time = 1440 min, min/max number of nodes = 4/8,
- max16n: max time = 720 min, min/max number of nodes = 4/16,
- max32n: max time = 360 min, min/max number of nodes = 4/32.

This policy motivates users to deploy higher numbers of nodes. Also, it prevents overloading the supercomputer with small one-node or two-node jobs.

Another aspect of job scheduling is the waiting time for the job to start calculation after being submitted to the job queue. Figure 15 shows the correlation of the job running time t_R to the job waiting time t_W (that is the time between the moment of the job submission into the batch queue and the moment of the job execution). Three levels of ratios t_W/t_R are shown on Figure 15. We see that the majority of jobs (66%) fall to the category with $t_W < t_R$. Obviously, the jobs with $t_W > t_R$ should be regarded as a signature of the inefficient deployment policy of the supercomputer.

9. Conclusions

In the article, we have described the Desmos supercomputer that is the first public HPC system based on the Angara interconnect and targeted to MD calculations.

The benchmarks of the Angara interconnect shows its applicability for building HPC systems. The results obtained by the OSU Micro-Benchmarks and Intel MPI Benchmarks tests show that Angara is on par with the Infiniband FDR and in some cases demonstrate better results (e.g. for the OSU latency between two adjacent nodes).

A simple model has been presented for estimating a possible gain from the MPI processes mapping optimization for the Angara interconnect with torus topology. According to this model, the topology-aware mapping would give no substantial benefits for the 32-node Desmos supercomputer but is expected to speed up calculations for Angara-based systems with more than 256 nodes.

Desmos shows very efficient strong scaling for such popular codes for molecular modelling as GROMACS, LAMMPS, VASP and CP2K that confirms the maturity of the Angara interconnect and its software ecosystem.

The job accounting statistic of the Desmos supercomputer has been reviewed. Two methods of quantitative efficiency monitoring and analysis have been proposed.

Acknowledgements

The work with the Desmos supercomputer was performed at the Supercomputer Centre of JIHT RAS. The authors thank for the access to the other supercomputers the Lomonosov Moscow State University Supercomputing Center, the Joint Supercomputer Centre of Russian Academy of Sciences and the Shared Resource Center ‘Far Eastern Computing Resource’ IACP FEB RAS. The authors are grateful to Dr Maciej Cytowski and Dr Jacek Piechota (ICM, University of Warsaw) for the data on the VASP benchmark (Cytowski, 2013). The comments, corrections and suggestions of anonymous reviewers of this article are gratefully acknowledged.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The JIHT team has been supported by the Russian Science Foundation (grant no. 14-50-00124). The work of the JIHT team included the preliminary tests for the co-design of the Desmos architecture, tuning of the codes performance and benchmarking.

The work of the HSE team has been prepared within the framework of the HSE University Basic Research Program and funded by the Russian Academic Excellence Project '5–100'. The work included the development of the topology-aware MPI process-placement algorithm for Desmos and the analysis of statistics of its deployment.

The NICEVT team developed the Angara interconnect and its low-level software stack, built the Desmos supercomputer and prepared it to general deployment.

ORCID iD

Vladimir Stegailov https://orcid.org/0000-0002-5349-3991

References

Abraham MJ, Murtola T, Schulz R, et al. (2015) GROMACS: high performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* 12: 19–25. Available at: http://www.sciencedirect.com/science/article/pii/S2352711015000059.

Adamovich IA, Klimov AV, Klimov YA, et al. (2010) Thoughts on the development of SKIF-Aurora supercomputer interconnect. *Programmnye Sistemy: Teoriya i Prilozheniya* 1(3): 107–123. Available at: http://psta.psiras.ru/read/psta2010_3_107-123.pdf.

Adiga NR, Blumrich MA, Chen D, et al. (2005) Blue Gene/L torus interconnection network. *IBM Journal of Research and Development* 49(2.3): 265–276.

Agarkov AA, Ismagilov TF, Makagon DV, et al. (2016) Performance evaluation of the Angara interconnect. In: *Proceedings of the International Conference "Russian Supercomputing Days"*. pp. 626–639. Available at: http://dx.doi.org/10.14529/cmse150304 (accessed 7 February 2019).

Ajima Y, Inoue T, Hiramoto S, et al. (2012) The Tofu interconnect. *IEEE Micro* 32(1): 21–31.

Akinori Yonezawa (2012) "Concurrent objects, agents and HPC," AICS RIKEN, 2012. Available at: http://www.slideshare.net/AkinoriYonezawa/ageretalk (accessed 7 February 2019).

Barone GB, Boccia V, Bottalico D, et al. (2017) An approach to forecast queue time in adaptive scheduling: how to mediate system efficiency and users satisfaction. *International Journal of Parallel Programming* 45(5): 1164–1193.

Begau C and Sutmann G (2015) Adaptive dynamic load-balancing with irregular domain decomposition for particle simulations. *Computer Physics Communications* 190: 51–61. Available at: http://www.sciencedirect.com/science/article/pii/S0010465515000181.

Berendsen H, van der Spoel D and van Drunen R (1995) GROMACS: a message-passing parallel molecular dynamics implementation. *Computer Physics Communications* 91(13): 43–56. Available at: http://www.sciencedirect.com/science/article/pii/001046559500042E.

Brown WM, Kohlmeyer A, Plimpton SJ, et al. (2012) Implementing molecular dynamics on hybrid high performance computers — Particle-particle particle-mesh. *Computer Physics Communications* 183(3): 449–459. Available at: http://www.sciencedirect.com/science/article/pii/S0010465511003444.

Brown WM, Wang P, Plimpton SJ, et al. (2011) Implementing molecular dynamics on hybrid high performance computers – short range forces. *Computer Physics Communications* 182(4): 898–911. Available at: http://www.sciencedirect.com/science/article/pii/S0010465510005102.

Caruso P, Laccetti G and Lapegna M (2005) A performance contract system in a grid enabling, component based programming environment. In: Sloot PMA, Hoekstra AG, Priol T, et al. (eds.), *Advances in Grid Computing - EGC 2005*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 982–992. ISBN 978-3-540-32036-4.

Corsetti F (2014) Performance analysis of electronic structure codes on HPC systems: a case study of SIESTA. *PLOS ONE* 9(4): 1–8. Available at: http://dx.doi.org/10.1371%2Fjournal.pone.0095390.

Cytowski M (2013) Best Practice Guide – IBM Power 775. In: *PRACE*. Available at: http://www.prace-ri.eu/IMG/pdf/Best-Practice-Guide-IBM-Power-775.pdf (accessed 7 February 2019).

D'Amore L, Laccetti G, Romano D, et al. (2015) Towards a parallel component in a GPUCUDA environment: a case study with the L-BGFS Harwell routine. *International Journal of Computer Mathematics* 92(1): 59–76. Available at: https://doi.org/10.1080/00207160.2014.899589.

Derradji S, Palfer-Sollier T, Panziera J, et al. (2015) The BXI interconnect architecture. In: *2015 IEEE 23 rd annual symposium on high-performance interconnects*, Santa Clara, CA, USA, 26–28 August 2015, pp. 18–25. Chair Luca Valcarengh: IEEE. DOI: 10.1109/HOTI.2015.15.

Eckhardt W, Heinecke A, Bader R, et al. (2013) 591 TFlops multitrillion particles simulation on SuperMUC. In: Kunkel JM, Ludwig T and Meuer HW (eds) *International Supercomputing Conference ISC 2013*. Leipzig, Germany, 16–20 June 2013, pp. 1–12. Springer.

Eckhardt W, et al. (2013) 591 TFLOPS Multi-trillion Particles Simulation on SuperMUC. In: Kunkel JM, Ludwig T and Meuer HW (eds) *Supercomputing. ISC 2013. Lecture Notes in Computer Science, vol 7905. Springer, Berlin, Heidelberg 28th International Supercomputing Conference, ISC 2013*, Leipzig, Germany, June 16–20, 2013.

Edwards HC, Trott CR and Sunderland D (2014) Kokkos: enabling manycore performance portability through polymorphic memory access patterns. *Journal of Parallel and Distributed Computing* 74(12): 3202–3216. Available at: http://www.sciencedirect.com/science/article/pii/S0743731514001257 (accessed 7 February 2019). Domain-Specific

Languages and High-Level Frameworks for High-Performance Computing.

Elizarov GS, Gorbunov VS, Levin VK, et al. (2012) Communication fabric MVS-Express. *Vychisl Metody Programm* 13(3): 103–109. Available at: http://num-meth.srcc.msu.ru/english/zhurnal/tom_2012/v13r214.html.

Gómez-Martín C, Vega-Rodríguez MA and González-Sánchez JL (2016) Fattened backfilling: an improved strategy for job scheduling in parallel systems. *Journal of Parallel and Distributed Computing* 97: 69–77.

Höhnerbach M, Ismail AE and Bientinesi P (2016) The vectorization of the Tersoff multi-body potential: an exercise in performance portability. In: *Proceedings of the international conference for high performance computing, networking, storage and analysis*. article No. 7, Salt Lake City, Utah, 13–8 November 2016, p. 7. NJ, USA: IEEE Press.

Haque IS and Pande VS (2010) Hard data on soft errors: a large-scale assessment of real-world error rates in GPGPU. In: Manish P and Rajkumar B (eds) *Proceedings of the 2010 10th IEEE/ACM international conference on cluster, cloud and grid computing, CCGRID '10*. Washington, DC, USA, pp. 691–696. IEEE Computer Society. ISBN: 978-0-7695-4039-9, DOI:10.1109/CCGRID.2010.84.

Heinecke A, Eckhardt W, Horsch M, et al. (2015) *Supercomputing for Molecular Dynamics Simulations*. Cham: Springer.

Hoefler T and Snir M (2011) Generic topology mapping strategies for large-scale parallel architectures. In: *Proceedings of the international conference on supercomputing, ICS '11*, Tucson, USA, 31 May–04 June 2011, pp. 75–84. NY, USA: ACM. Available at: http://doi.acm.org/10.1145/1995896.1995909. ISBN 978-1-4503-0102-2, DOI:10.1145/1995896.1995909.

Hutter J, Iannuzzi M, Schiffmann F, et al. (2014) CP2K: atomistic simulations of condensed matter systems. *Wiley Interdisciplinary Reviews: Computational Molecular Science* 4(1): 15–25. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/wcms.1159.

Klimov YA, Shvorin AB, Khrenov AY, et al. (2015) Pautina: the high performance interconnect. *Programmnye Sistemy: Teoriya i Prilozheniya* 6(1): 109–120. Available at: http://psta.psiras.ru/read/psta2015_1_109-120.pdf.

Kondratyuk ND, Norman GE and Stegailov VV (2016) Selfconsistent molecular dynamics calculation of diffusion in higher n-alkanes. *The Journal of Chemical Physics* 145(20): 204504.

Korzh AA, Makagon DV, Borodin AA, et al. (2010) Russian 3D-torus interconnect with globally addressable memory support. *Vestnik YuUrGU Ser Mat Model Progr* (6): 41–53. Available at: http://mmp.vestnik.susu.ru/article/en/107 (accessed 7 February 2019).

Kraemer A, Maziero C, Richard O, et al. (2016) Reducing the number of response time SLO violations by a cloud-HPC convergence scheduler. In: Essaaidi M and Zbakh M (eds) *2016 2nd international conference on cloud computing technologies and applications (CloudTech)*. Marrakech, Morocco, 24–26 May 2016, pp. 293–300. IEEE.

Kresse G and Furthmüller J (1996a) Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set. *Phys Rev B* 54: 11169–11186.

Kresse G and Furthmuller J (1996b) Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. *Comput Mater Sci* 6(1): 15–50. Available at: http://www.sciencedirect.com/science/article/pii/0927025696000080 (accessed 7 February 2019).

Kresse G and Hafner J (1993) Ab initio molecular dynamics for liquid metals. *Phys Rev B* 47: 558–561.

Kresse G and Hafner J (1994) Ab initio molecular-dynamics simulation of the liquid-metal-amorphous-semiconductor transition in germanium. *Phys Rev B* 49: 14251–14269.

Kumar S, Sun Y and Kalé LV (2013) Acceleration of an asynchronous message driven programming paradigm on IBM Blue Gene/Q. In: *2013 IEEE 27th international symposium on parallel and distributed processing*, Boston, USA, 20–24 May 2013, pp. 689–699. DOI: 10.1109/IPDPS. 2013.83. IEEE.

Kutzner C, Páll S, Fechner M, et al. (2015) Best bang for your buck: GPU nodes for GROMACS biomolecular simulations. *Journal of Computational Chemistry* 36(26): 1990–2008.

Laccetti G, Lapegna M, Mele V, et al. (2014) A study on adaptive algorithms for numerical quadrature on heterogeneous GPU and multicore based systems. In: Wyrzykowski R, Dongarra J and Karczewski K, et al. (eds.), *Parallel Processing and Applied Mathematics*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 704–713. ISBN 978-3-642-55224-3.

Lin H, Tang X, Yu B, et al. (2017) Scalable graph traversal on Sunway TaihuLight with ten million cores. In: *2017 IEEE International parallel and distributed processing symposium (IPDPS)*, Orlando, FL, USA, 29 May–2 June 2017, pp. 635–645. IEEE. DOI:10.1109/IPDPS.2017.53.

Mamaeva AA and Voevodin VV (2017) Methods for statistical analysis of large supercomputer job flow. In: Voevodin VI V (ed) *Proceeding of international conference Russian supercomputing days*. Moscow, Russia, 25–26 September 2017, pp. 788–799. MSU.

Montella R, Giunta G, Laccetti G, et al. (2017) On the virtualization of CUDA based GPU remoting on ARM and x86 machines in the GVirtuS framework. *International Journal of Parallel Programming* 45(5): 1142–1163.

Mukosey AV, Semenov AS and Simonov AS (2015) Simulation of collective operations hardware support for Angara interconnect. *Vestn YuUrGU Ser Vych Matem Inform* 4(3): 40–55.

Murli A, Boccia V, Carracciuolo L, et al. (2017) Monitoring and migration of a PETSc-based parallel application for medical imaging in a grid computing PSE. In: Gaffney PW and Pool JCT (eds.), *Grid-Based Problem Solving Environments*. Boston, MA: Springer US, pp. 421–432. ISBN 978-0-387-73659-4.

Neuwirth S, Frey D, Nuessle M, et al. (2015) Scalable communication architecture for network-attached accelerators. In: *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, pp. 627–638. IEEE.

Ohmura I, Morimoto G, Ohno Y, et al. (2014) 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), article no. 20130387, pp. 2–16. Available at: http://rsta.royalsocietypublishing.org/content/372/2021/20130387.full.pdf. DOI: 10.1098/rsta.2013.0387 (accessed 7 February 2019).

Piana S, Klepeis JL and Shaw DE (2014) 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.

Plimpton S (1995) Fast parallel algorithms for short-range molecular dynamics. *J Comput Phys* 117(1): 1–19.

Puente V, Beivide R, Gregorio JA, et al. (1999) Adaptive bubble router: a design to improve performance in torus networks. In: Panda D and Shiratori N (eds) *1999 International conference on parallel processing, 1999. Proceedings*. Aizu-Wakamatsu, Japan, 24 September 1999, pp. 58–67. IEEE.

Rauber T and Runger G (2013) *Parallel Programming for Multicore and Cluster Systems*. Cham: Springer.

Rojek K, Wyrzykowski R and Kuczynski L (2016) Systematic adaptation of stencil-based 3D MPDATA to GPU architectures. *Concurrency and Computation: Practice and Experience* 29(9): e3970. DOI: 10.1002/cpe.3970.

Scott SL, et al. (1996) The cray T3E network: adaptive routing in a high performance 3D torus. In: *Proceedings of HOT Interconnects IV*, Stanford University, 15–16 August 1996. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.126.3882 (accessed 7 February, 2019).

Smirnov GS and Stegailov VV (2016) Efficiency of classical molecular dynamics algorithms on supercomputers. *Mathematical Models and Computer Simulations* 8(6): 734–743.

Stegailov V, Agarkov A, Biryukov S, et al. (2017) Early performance evaluation of the hybrid cluster with torus interconnect aimed at molecular-dynamics simulations. In: Wyrzykowski R, Dongarra J, Deelman E, et al. (eds.), *Parallel Processing and Applied Mathematics*. Cham: Springer International Publishing, pp. 327–336. ISBN 978-3-319-78024-5.

Stegailov VV, Orekhov ND and Smirnov GS. (2015) HPC Hardware Efficiency for Quantum and Classical Molecular Dynamics. In: Malyshkin V (ed) *Parallel Computing Technologies. PaCT 2015. Lecture Notes in Computer Science*. vol 9251, Petrozavodsk, Russia, 31 August–4 September, 2015, pp. 469–473. Cham: Springer.

Sun Y, Zheng G, Mei C, et al. (2012) Optimizing fine-grained communication in a biomolecular simulation application on Cray XK6. In: *Proceedings of the international conference on high performance computing, networking, storage and analysis. SC '12*, Los Alamitos, CA, USA, 10–16 November 2012, pp. 55:1–55:11. Available at: https://dl.acm.org/citation.cfm?id=2389071 (accessed 7 February 2019). CA, USA: IEEE Computer Society Press. ISBN 978-1-4673-0804-5.

Trott CR, Winterfeld L and Crozier PS (2010) General-purpose molecular dynamics simulations on GPU-based clusters. *ArXiv e-prints*. Available at: https://arxiv.org/abs/1009.4330 (accessed 7 february 2019).

Yoo AB, Jette MA and Grondona M (2003) Slurm: simple linux utility for resource management. In: *Workshop on Job Scheduling Strategies for Parallel Processing*. WA, USA, 24 June 2003, pp. 44–60. Berlin, Heidelberg: Springer.

Author biographies

Vladimir Stegailov graduated from the Moscow Institute of Physics and Technology in 2004, received a PhD degree in 2005 and a DrSc degree in 2012. He is Head of Department at the Joint Institute for High Temperatures of Russian Academy of Sciences, Moscow, Russia. He is a professor of the Higher School of Economics and a leading researcher in the International Laboratory for Supercomputer Atomistic Modelling and Multi-scale Analysis. He is a professor in the Moscow Institute of Physics and Technology where he leads the Laboratory of Supercomputer Methods in Condensed Matter Physics. His research interests include high-performance computing and atomistic and multiscale modelling and simulation.

Ekaterina Dlinnova received her Master's degree in Applied Mathematics and Informatics from the Moscow Institute of Electronics and Mathematics of the Higher School of Economics in 2017. Currently, she is a PhD student of the Doctoral School of Computer Science of HSE and a research assistant at the International Laboratory for Supercomputer Atomistic Modelling and Multi-scale Analysis of HSE. Her research interests include the optimization of supercomputer task scheduling and user policies.

Timur Ismagilov is a senior researcher at JSC NICEVT, Moscow, Russia. His research interests include high performance computing, interconnects, scalability of applications performance, classes of functions with a dominating mixed modulus of smoothness, embedding theorems. He graduated from the Moscow State University in 2011 and received a PhD degree in 2017.

Mikhail Khalilov is a software engineer at JSC NICEVT and a junior researcher at the International Laboratory for Supercomputer Atomistic Modelling and Multi-scale Analysis of the Higher School of Economics. He received his Bachelor's degree in Applied Mathematics from the Moscow Institute of Electronics and Mathematics of the Higher School of Economics in 2018. Currently, he is a Master student in MIEM HSE. His research interests include high-performance computing, interconnects and networking.

Nikolay Kondratyuk is a junior researcher at the Joint Institute for High Temperatures Russian Academy of Sciences, Moscow, Russia, as well as at the International Laboratory for Supercomputer Atomistic Modelling and Multi-scale Analysis of HSE and at the Laboratory of Supercomputer Methods in Condensed Matter Physics of MIPT. His research interests include molecular dynamics calculations of soft matter properties and using high-performance computing in these studies. He received his Master's degrees in Applied Physics and Mathematics from the Moscow Institute of Physics and Technology in 2016. Currently, he is a PhD student in MIPT.

Dmitry Makagon received his diploma in Applied Mathematics from the National Research University of Electronic Technology (MIET), Moscow, in 2006. Since then, he has been working in the field of high performance computing. Currently, he is Head of Department at JSC NICEVT and a senior architect of the Angara high speed interconnect. His research interests include parallel computing, hardware acceleration of algorithms and architecture of interconnection networks.

Alexander Semenov is a head of department of JSC NICEVT and a senior architect of application and system software in the Angara project. In 2018, he joined the International Laboratory for Supercomputer Atomistic Modelling and Multi-scale Analysis of the Higher School of Economics as a leading researcher. His research interests include high-performance computing, interconnects, scalability of applications performance and graph algorithms. He graduated from the Moscow State University in 2004 and received a PhD degree in 2010.

Alexei Simonov is the first deputy director of JSC NICEVT. His research interests include high performance computing, interconnects and scalability of applications performance. He graduated from the Tula State University in 1994 and received a PhD degree in 2001.

Grigory Smirnov is a head of laboratory at the Joint Institute for High Temperatures of Russian Academy of Sciences, Moscow, Russia. He received his Master's degree in Applied Mathematics and Physics from the Moscow Institute of Physics and Technology in 2013 and then received a PhD degree in 2017. He is a leading researcher at the International Laboratory for Supercomputer Atomistic Modelling and Multi-scale Analysis of HSE and at the Laboratory of Supercomputer Methods in Condensed Matter Physics of MIPT. His main interests are in the field of high-performance computing, in particular in the classical and ab initio molecular dynamics simulations of molecular solids.

Alexey Timofeev is a deputy director for research at the Joint Institute for High Temperatures of Russian Academy of Sciences, Moscow, Russia. He graduated from the Moscow Institute of Physics and Technology in 2010 and received a PhD degree in 2011. He is an assistant professor at the Moscow Institute of Physics and Technology and at the Higher School of Economics. Also he is a leading researcher at the International Laboratory for Supercomputer Atomistic Modelling and Multi-scale Analysis of HSE and at the Laboratory of Supercomputer Methods in Condensed Matter Physics of MIPT. His current research interests include high-performance computing in materials science and plasma studies.