Architecture and Functionality of the Collective Operations Subnet of the Angara Interconnect

Alexey Simonov[1](✉) and Oleg Brekhov[2](✉)

[1] JSC "NICEVT", Varshavskoe Shosse 125, Moscow, Russia
alexey.s.simonov@rambler.ru

[2] National Research University for Aeronautical Engineering, Volokolamkoe Shosse 4, Moscow, Russia
obrekhov@mail.ru

http://www.nicevt.ru

Abstract. The Angara interconnect developed by JSC "NICEVT" is designed to connect the nodes of supercomputers and computing clusters. The paper describes the main architectural solutions, algorithms and functionality of the collective operations subnet of the Angara interconnect and presents the forecast of its characteristics based on the simulation modeling and actual operation. The proposed solutions allow bringing the time complexity of the collective operations execution to the theoretical limit for the kD-torus topology network.

Keywords: Angara interconnect · Multiprocessor computing system · Supercomputer

1 Introduction

Multiprocessor computing systems (hereinafter referred to as MCS) are important for solving application tasks aimed at increasing the scientific and technical potential of the economy and strengthening the country's defense capability. After NEC SX-6 Earth Simulator [1] appeared, it became clear how the characteristics and capabilities of the interconnect are important for ensuring high scalability of the MCS performance in solving computationally complex problems, primarily computer simulation, processing of large data arrays and forecasting.

JSC "NICEVT" started the development of the first-generation Angara high-speed interconnect (hereinafter referred to as Angara interconnect) in 2006. The operation principles and technical appearance of the Angara interconnect were formed basing on the analysis of the world experience in creating custom-made interconnect solutions for the highest performance range supercomputers, primarily the IBM BlueGene series [2–4] and CRAY SeaStar/Gemini [5–7], as well

JSC "NICEVT".

V. M. Vishnevskiy et al. (Eds.): DCCN 2020, LNCS 12563, pp. 209–219, 2020.
https://doi.org/10.1007/978-3-030-66471-8_17

as on the results of a number of studies conducted at JSC “NICEVT” using simulation modeling tools.

Angara interconnect is a Direct Network that supports topologies from 1D-mesh to 4D-torus and allows to create MCS of up 32K nodes. Its first production samples were presented in 2013.

During the development of the Angara interconnect the emphasis was placed on ensuring high scalability of the MCS performance in solving computationally complex problems. Algorithms for solving these problems, as a rule, are based on numerical methods with spatial decomposition of the computational domain, which require the execution of collective operations and boundary condition exchange operations after every iteration between the computational nodes of MCS that are involved in solving the problem.

Collective operations are among the main primitives of the interaction of computational processes in most parallel programming standards oriented to distributed memory computer systems (MPI, SHMEM, PGAS-languages—UPC, X10), and they can make up a significant part of communication exchanges. Such operations primarily include:

- broadcasting a packet from one node of MCS to other nodes allocated to the application (broadcast);
- collecting information from the nodes allocated to the application into one node and performing reduction (reduce).

This paper presents the main architectural solutions, algorithms and functionality of the collective operations subnet of the Angara interconnect and forecast of its characteristics based on the simulation modelling and actual operation.

2 Architecture of the Collective Operations Subnet of the Angara Interconnect

A trivial way to implement collective operations at the hardware level is to adequately replace them with many point-to-point operations, in which each broadcast is replaced with many writes to the memory coming from the root node to all other nodes of MCS allocated to the application, and each operation of collection is replaced by many operations of reading data from the MCS nodes memory to the root node.

This approach has undoubted advantages - simplicity of implementation and predictability of the result, and it can be used to build small MCS up to 100–200 nodes, for which, due to the small number of nodes, broadcast traffic will not have a significant impact on the load on the interconnect, and due to the small diameter of the network and the number of hops, the communication delay will be negligible.

For the medium and large size MCS the situation is different. In this case a large share of duplicate traffic can significantly affect the load of the interconnect, which, together with a large diameter, can negatively affect the latency and bandwidth of the interconnect and will lead to a significant deterioration in MCS performance [8]. That is why the hardware support of collective operations will positively affect the MCS performance scalability [9].

Obviously, a significant increase in the duration of collective operations due to an increase in the number of MCS nodes allocated to the application is associated with two factors - increased traffic due to an increase in the number of packets in the network and an increase in the number of hops due to an increase in the diameter of the network. In this regard, the problem of reducing the number of packets when performing collective operations for medium and large size MCS is very urgent.

The analysis of the trajectories of packets on the network in the case of implementation of the collective operations as a set of simple read and write operations in the memory of a remote node showed that in most cases the trajectories overlap each other. Considering the fact that the proposed solution requires its own routing algorithm that allows duplication and multiplication of packets when passing through transit nodes, it is advisable to implement hardware support for collective operations within a dedicated collective operations virtual subnet based on two virtual channels. A virtual subnet has the topology of a tree built in a torus (developers of the SMPO-10G interconnect came to a similar solution). There are two directions of movement in the tree: from root to leaves and from leaves to root. Each direction has its own virtual channel, VcDown – from root to leaves, and VcUp – from leaves to root. There may be transit nodes in the system; neither injection nor ejection of traffic occurs in these nodes.

When the broadcast operation is performed, each node, when receiving a packet from a node located higher in the tree, sends it to all nodes located lower in the tree. When a packet is injected into the network not from the root node, first a broadcast request is generated, which is sent to the root via the VcUp virtual channel, after which the broadcast operation itself is performed from the root via the VcDown virtual channels.

When the reduce (or all reduce) operation is performed, the node waits for packets from the node's processor if the node isn't transit, and from all the nodes located lower in the tree, performs the commutative associative binary operation indicated in the packet and sends the finished result up to the root. The current implementation supports the operations of maximum, minimum and sum of integers. The reduce operation ends with a hardware sending (without ejection) of the result to the given node using point-to-point operation (broadcast for all reduce). In order to determine direction to and from the root a routing table for collective operations subnet is set at each node. The table form is shown in Table 1.

The routing table in addition to dir-bits indicating the direction of the packet distribution along the subnet down the tree, has isRoot field that determines whether the node is root, toRoot field that indicates the direction to the root node, PE field that determines whether this node is transit or not and dirSum field which stores the number of directions to the leaves of the tree.

Table 1. Form of collective operation subnet routing table

TreeId	isRoot	toRoot	PE	dirSum	dir-bits			
					+X	−X	+Y	−Y
0 × 00	0	+Y	1	3	1	1	0	1
0 × 01	0	+X	0	2	0	0	1	1
...								
0 × 0F								

To set the correct tree, the following criteria must be met:

a) there is only one root;
b) if at some node the direction is set down the tree, then in this direction there should be a node that belongs to the tree in which the direction up the tree is set opposite to the given;
c) directions to nodes down the tree can only be:

– directions next to the direction, specified by the toRoot field, according to the direction order;
– the direction opposite to the direction, specified by the toRoot field.

Since many tasks can run simultaneously on the MCS, the collective operations virtual subnet allows to build up to 16 intersecting trees. At the same time, one task can use several trees. Each tree has its own TreeId identifier. For each TreeId identifier there is a corresponding routing table entry that determines routing direction. A subnet supports up to 16 different reduce packets for each TreeId. Each reduce performing on this tree has its own reduceId identifier (from 0 to 15).

The generalized algorithms of the virtual channels VcDown and VcUp of the collective operation subnet are shown in Fig. 1 and 2. The algorithm presented in Fig. 1 works as follows. From the header fleet received for routing, the TreeId field is selected, according to which the corresponding line is searched in the routing table of the collective operations subnet. If the line is found, the PE field in the routing table is checked. If it is 1, i.e. the node is not transit; the packet is ejected into the node.

Next, the reading and execution of the bitwise conjunction operation of dir-bits of the packet header fleet with dir-bits from the routing table is performed. The resulting bit vector is used to perform a loop search over directions (+X, −X, +Y, −Y . . .) and duplicate the packet into those for which the corresponding bit of the resulting vector is in state 1.

The algorithm presented in Fig. 2 works as follows. The packet type is checked after checking the TreeId identifier. If this is a broadcast packet the field isRoot of the routing table is checked to see if this node is root. If this is a root node the package is ejected into the node and distributed similar to previous algorithm. Otherwise, the toRoot field of the routing table is read and the packet is sent in the direction to the root node of the tree.

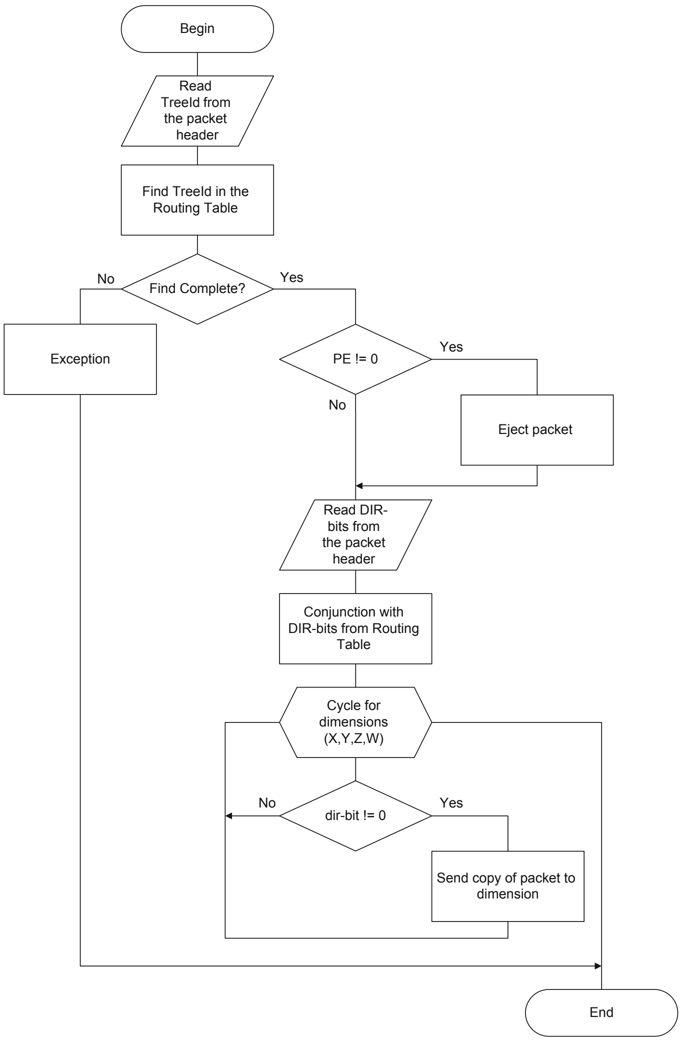
Begin
Read TreeId from the packet header
Find TreeId in the Routing Table
Find Complete?
No
Yes
Exception
PE != 0
Yes
No
Eject packet
Read DIR-bits from the packet header
Conjunction with DIR-bits from Routing Table
Cycle for dimensions (X,Y,Z,W)
dir-bit != 0
No
Yes
Send copy of packet to dimension
End

Fig. 1. The algorithm of the virtual channel VcDown of the collective operations subnet of the network with kD-torus topology

Reduce type packets that have got into the virtual channel VcUp are processed as follows. Data is extracted from the package and stored in a special table in the line corresponding to the value of the reduceId field in the package, after which the received packets counter allocated for this reduceId is increased by 1. If packets came from all directions, i.e. the value of received packets counter has become equal to the dirSum value of the routing table, the operation is performed, the packet is sent up the tree to the root node and the received packets counter is reset.

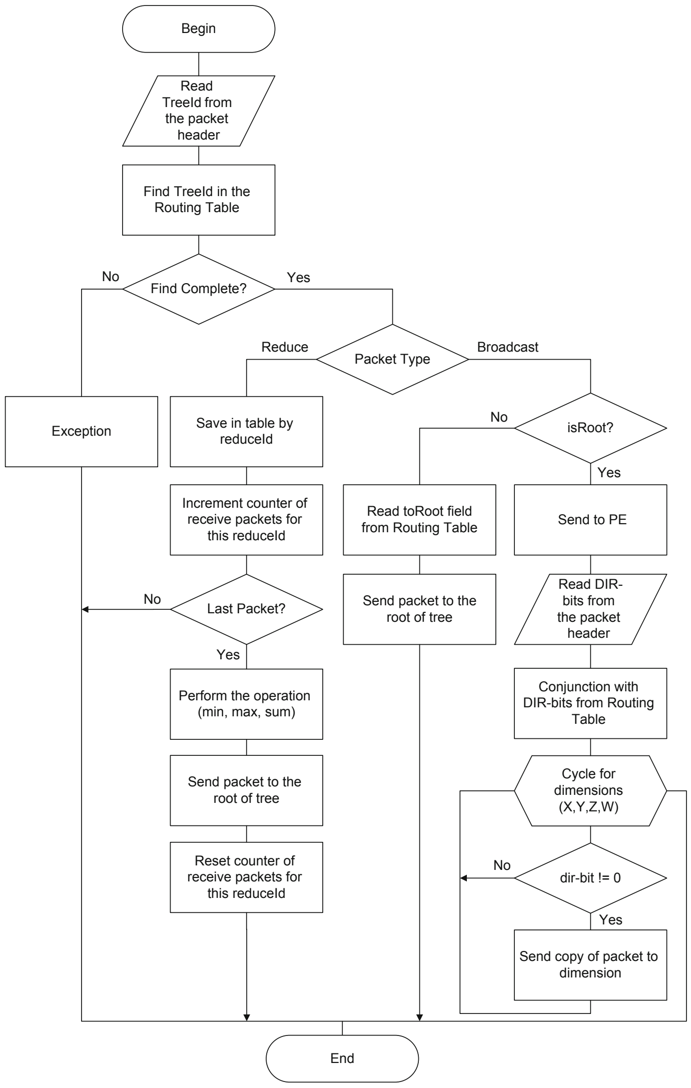
Begin
Read TreeId from the packet header
Find TreeId in the Routing Table
Find Complete?
No
Yes
Packet Type
Reduce
Broadcast
Exception
Save in table by reduceId
Increment counter of receive packets for this reduceId
Last Packet?
No
Yes
Perform the operation (min, max, sum)
Send packet to the root of tree
Reset counter of receive packets for this reduceId
isRoot?
No
Yes
Read toRoot field from Routing Table
Send packet to the root of tree
Send to PE
Read DIR-bits from the packet header
Conjunction with DIR-bits from Routing Table
Cycle for dimensions (X,Y,Z,W)
dir-bit != 0
No
Yes
Send copy of packet to dimension
End

Fig. 2. The algorithm of the virtual channel VcUp of the collective operations subnet of the network with kD-torus topology

From the point of view of the application programmer, the basic versions of collective operations are one-way asynchronous operations. The processor is not blocked after sending the message, and the result is written into memory without the active participation of the receiving party, which allows overlapping computation and communication. Synchronization mechanisms based on collective operations are implemented in order to determine if the collective operation is completed and the result is available to the computational nodes.

Considering the specificity of toroidal networks, the distribution of nodes by tasks is usually carried out on the principle of minimizing the distance between them. As a result, the nodes allocated to the task in the structure of the kD-torus are generally limited to a rectangular region. Since it is advisable to use the principle of minimizing the distances to the most distant nodes of the tree when choosing the root node, it is obvious that it is advisable to choose the node located in the geometric center of the rectangular area of the MCS nodes allocated to the task as the root node.

Two mechanisms are used in collective operations virtual subnet to prevent deadlocks caused by different tasks trees overlapping:

- bubble routing;
- various trees are constructed according to the X, Y, Z, W direction order.

3 Characteristics of the Collective Operations Subnet of the Angara Interconnect

To confirm the efficiency of proposed algorithms time complexity should be estimated. Diameter of a network is the shortest distance between the two most distant nodes in it. In general, for an equilateral kD-torus with even number of nodes diameter equals

$$r = \frac{k}{2}\sqrt[k]{\omega} \tag{1}$$

where r - is the network diameter;

k - is a number of dimensions in torus;

ω – is the total number of MCS nodes.

For an arbitrary 4D-torus networks r gives the lower-bound estimate.

When using VCT routing method, network message delivery time is determined by the communication delay and the time, required to inject the message of a given size into the communication channel (link) with a certain bandwidth, that is

$$T_d = t^0 + (l-1)t^1 + \frac{Q}{V_L} \tag{2}$$

where T_d - is the network message delivery time;

t^0 – is the communication latency between two adjacent nodes;
l – is the route length (number of hops);
t^1 – is the communication latency of a hop;
Q – is the message size;
V_L – is the communication channel bandwidth.

If collectives are implemented using point-to-point operations, the time complexity estimation for the broadcast and reduce algorithms will heavily depend on the host interface bandwidth of the root node

$$T(A^1_\omega) = T(A^\omega_1) = (\omega - 1)(t^1 + \frac{Q}{V_P}) + (\frac{k}{2}\sqrt[k]{\omega} - 1)t^1 + t^0 \tag{3}$$

where A^1_ω – is a broadcast algorithm;

$T(A^1_\omega)$ – is an estimation of the time complexity of the broadcast algorithm;
A^ω_1 – reduce algorithm;
$T(A^\omega_1)$ - is an estimation of the time complexity of the reduce algorithm;
V_P – host processor interface bandwidth.

Considering that in the VcDown and VcUp virtual channels algorithms the multiplication of packets during broadcast execution and reduction during reduce execution are performed at all nodes in the tree, the proposed method can significantly reduce both the number of packets and the total execution time. As a result, the estimate of time complexity will be close to the theoretical limit as it is determined by the distance to the most distant nodes. Applied to kD-torus network

$$T(A^1_\omega) = T(A^\omega_1) = t^0 + (\frac{k}{2}\sqrt[k]{\omega} - 1)t^1 + \frac{Q}{V_L} \tag{4}$$

Collective operations execution time were estimated to verify the above relation using simulation model. The tests consisted of sending one packet of 32 flits (256 bytes) using the broadcast operation for a different number of nodes of the simulated network.

Tables 2, 3 present the results obtained under the following initial conditions: $t^0 = 700$ ns; $t^1 = 80$ ns; $Q = 256$ bytes; $V_L = 6$ GB/s; $V_P = 8$ GB/s.

Table 2. Comparison of the estimated execution time of the broadcast operation depending on the number of computational nodes for the 3D-torus network

Parameter	Number of nodes of MCS			
	8	64	512	4096
3D-torus, point-to-point collective operations				
Analytical calculation, s	1,64	8,16	58,81	461,18
Simulation modelling results, s	2,18	6,07	40,16	312,64
Divergence, %	24,5%	25,6%	31,7%	32,2%
3D-torus, proposed method				
Analytical calculation, s	0,9	1,14	1,62	2,58
Simulation modelling results, s	0,96	1,20	1,76	2,80
Divergence, %	5,9%	4,8%	7,8%	7,8%

The above results indicate an acceptable value of the inaccuracy in estimating the time complexity of collective operations execution using expression 4. Expression 3, unfortunately, gives a higher inaccuracy, which, according to the author, is caused by incomplete consideration of certain aspects of the network.

The results obtained using the simulation model made it possible to preliminarily confirm the hypothesis that the proposed method for making a collective operations subnet gives a significant gain in comparison with the use point-to-point collective operations. At the same time, its hardware implementation in comparison with the software implementation also provides a significant gain.

Table 3. Comparison of the estimated execution time of the broadcast operation depending on the number of computational nodes for the 4D-torus network

Parameter	Number of nodes of MCS		
	16	256	4096
4D-torus, point-to-point collective operations			
Analytical calculation, s	2,62	29,82	460,54
Simulation modelling results, s	2,90	21,78	312,65
Divergence, %	9,6%	26,9%	32,1%
4D-torus, proposed method			
Analytical calculation, s	0,98	1,3	1,94
Simulation modelling results, s	1,04	1,44	2,16
Divergence, %	5,5%	9,5%	10,0%

4 Angara Interconnect Design Versions

Currently there are two version of the Angara interconnect:

– switchless version – full-height full-length PCI Express card (see. Fig. 3) that allows to connect up 32K computing nodes with an $8 \times 16 \times 16 \times 16$ 4D-torus topology;
– switch version – 19" 24-port switch and low-profile PCI Express card (half-height full-length) (see. Fig. 4). This version allows to connect up to 2048 computing nodes with 2D-torus topology by connecting up to 256 switches.

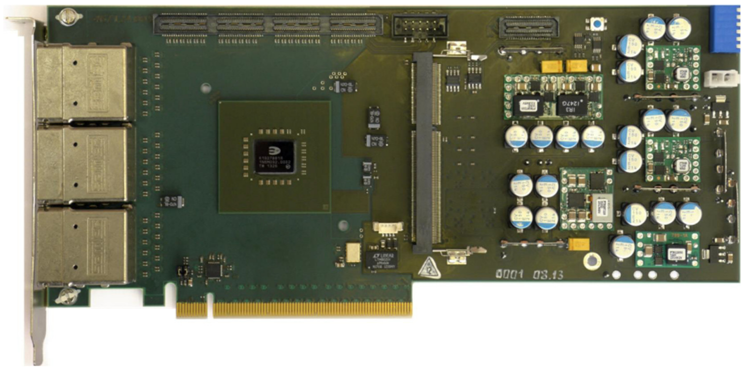

Fig. 3. Switchless version of the Angara interconnect

There are several MCS with high performance scalability based on the Angara interconnect in the Russian Academy of Sciences institutions, research institutes and industrial enterprises.

Fig. 4. The Angara interconnect 24-port switch (a) and low-profile PCI Express card (b)

The Angara interconnect allows achieving good performance scalability of MCS both on evaluation tests and applied computer simulation tasks (see. Fig. 5) [10–16].

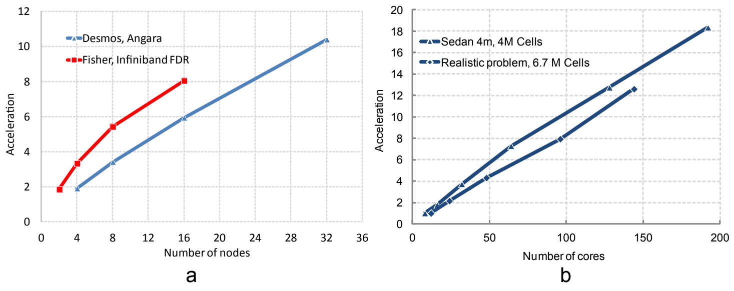

Fig. 5. MCS performance scalability (a – VASP - software package for quantum-chemical calculations, Desmos supercomputer; b – ANSYS FLUENT, Angara-K1 cluster)

5 Conclusion

1. The paper proposes architectural and algorithmic solutions for the collective operations subnet for a kD-torus topology network.
2. The analytical assessment of the developed algorithms time complexity is presented in the paper.
3. A comparison of the analytical assessment with the simulation modeling results was performed, which showed a 10% divergence (for proposed method).
4. The analysis of the developed algorithms was performed and the preliminary confirmation of the hypothesis that the proposed architectural and algorithmic solutions allows bringing the time complexity of the collective operations execution to the theoretical limit for the kD-torus topology network was obtained using the simulation model.

The proposed architectural and algorithmic solutions have a positive effect on the scalability of MVS performance in solving computationally complex problems, primarily computer simulation, processing large data arrays, planning and forecasting.

References

1. Habata, S., et al.: The earth simulator system. NEC Res. Dev. **44**(1), 21–26 (2003)
2. Gara, A.: Overview of the Blue Gene/L system architecture. IBM J. Res. Dev. **49**, 195–212 (2005). http://rsim.cs.illinois.edu/arch/qual_papers/systems/19.pdf
3. Almasi, G., Asaad, S., Bellofatto, R.E., et al.: Overview of the IBM Blue Gene/P project. IBM J. Res. Dev. **52**(1–2), 199–220 (2008). http://scc.acad.bg/ncsa/documentation/team.pdf
4. Chen, D., Eisley, N., Heidelberger, P., et al.: The IBM BlueGene/Q interconnection fabric. IEEE Micro **32**(1), 32–43 (2012). https://www.researchgate.net/publication/220290398_The_IBM_blue_geneQ_interconnection_fabric
5. Abts, D., Storm, C.R.: The Cray XT4 and Seastar 3-D torus interconnect (2010). https://static.googleusercontent.com/media/research.google.com/ru//pubs/archive/36896.pdf
6. Alam, S.R.: Cray XT4: an early evaluation for petascale scientific simulation. In: Proceedings of the 2007 ACM/IEEE Conference on Supercomputing, pp. 1–12. IEEE (2007). https://doi.org/10.1145/1362622.1362675
7. Alverson, R., Roweth, D., Kaplan, L.: The Gemini system interconnect. 2010 IEEE 18th Annual Symposium on High Performance Interconnects (HOTI), pp. 83–87. IEEE (2010). https://doi.org/10.1109/HOTI.2010.23
8. Bala, V., Bruck, J., Cypher, R., et al.: CCL: A Portable and Tunable Collective Communication Library for Scalable Parallel Computers. In: Proceedings of the Parallel Processing Symposium, pp. 835–844 (1994). ISBN 0-8186-5620-6
9. Almási, G., Dózsa, G., Erway, C.C., Steinmacher-Burow, B.: Efficient implementation of allreduce on BlueGene/L collective network. In: Di Martino, B., Kranzlmüller, D., Dongarra, J. (eds.) EuroPVM/MPI 2005. LNCS, vol. 3666, pp. 57–66. Springer, Heidelberg (2005). https://doi.org/10.1007/11557265_12
10. Mukosey, A., Simonov, A., Semenov, A.: Extended routing table generation algorithm for the angara interconnect. In: Voevodin, V., Sobolev, S. (eds.) Extended Routing Table Generation Algorithm for the Angara Interconnect. CCIS, vol. 1129, pp. 573–583. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36592-9_47
11. Stegailov, V., et al.: Angara interconnect makes GPU-based Desmos supercomputer an efficient tool for molecular dynamics calculations. Int. J. High Perform. Comput. Appl. **33**, 507–521 (2019)
12. Stegailov, V., Smirnov, G., Vecher, V.: VASP hits the memory wall: processors efficiency comparison. Concurr. Comput. Pract. Exp., e5136 (2019). https://doi.org/10.1002/cpe.5136
13. Polyakov, S., Podryga, V., Puzyrkov, D.: High performance computing in multiscale problems of gas dynamics. Lobachevskii J. Math. **39**(9), 1239–1250 (2018)
14. Ostroumova, G., Orekhov, N., Stegailov, V.: Reactive molecular-dynamics study of onion-like carbon nanoparticle formation. Diamond Related Mater. **94**, 14–20 (2019)
15. Tolstykh, M., Goyman, G., Fadeev, R., Shashkin, V.: Structure and algorithms of SL-AV atmosphere model parallel program complex. Lobachevskii J. Math. **39**(4), 587–595 (2018). https://doi.org/10.1134/S1995080218040145
16. Akimov, V., Silaev, D., Aksenov, A., Zhluktov, S., Savitskiy, D., Simonov, A.: FlowVision scalability on supercomputers with angara interconnect. Lobachevskii J. Math. **39**(9), 1159–1169 (2018). https://doi.org/10.1134/S1995080218090081