ISSN 1995-0802, Lobachevskii Journal of Mathematics, 2019, Vol. 40, No. 5, pp. 566–573. © Pleiades Publishing, Ltd., 2019.

RANS/ILES Method Optimization for Effective Calculations on Supercomuter

G. I. Savin[1*], **L. A. Benderskiy**[1**], **D. A. Lyubimov**[1***], and **A. A. Rybakov**[1****]

(Submitted by A. M. Elizarov)

[1]*Joint Supercomputer Center, Russian Academy of Sciences, Moscow, 119334 Russia*

Received February 3, 2019; revised February 24, 2019; accepted February 24, 2019

Abstract—The article discusses the use of the RANS/ILES method for modeling gas-dynamic processes in a combustion chamber, described using an axisymmetric block-structured computational grid. For these calculations, the approaches of the linked border conditions and the fragmentation of the computational grid are used, which allows to effectively calculate this problem on a supercomputer, achieving acceleration by more than two orders of magnitude, using a total of 32 computing nodes.

DOI: 10.1134/S1995080219050172

Keywords and phrases: *computation fluid dynamics, RANS/ILES, calculation grid, grid crushing, linked border conditions.*

1. INTRODUCTION

The complex geometry of the combustion chamber and the different scale of the flame tube and fuel supply devices requires the use of computational grid with a very large number of cells for adequately describe all the features of the flow, even when using high-resolution methods. On the other hand, the geometry of the front device has a structure that is periodic in the azimuthal direction. This formally allows us to calculate only the sector of the flame tube with the conditions of periodicity on its lateral sides. However, it is known that separated flows in annular diffusers can have azimuthal inhomogeneity with axisymmetric distribution of flow parameters at the inlet [1]. This effect can only be detected when calculating the entire flame tube but not its sector.

Numeric simulation of combustion chamber sector require to solve a number of tasks on scaling the calculations. It is necessary to take into account the condition of periodicity in grid decomposition. This article is aimed at describing the mechanisms of grid decomposition with such boundary conditions and testing the decomposition method.

2. NUMERICAL METHOD

The RANS/ILES (Reynolds Averaged Navier–Stokes—RANS, Implicit Large Eddy Simulation—ILES) method is based on Navier–Stokes equations describing the flow of a compressible gas. The transport equation for turbulence model is written in conservative form for curvilinear coordinate system. The grid lines coincide with the boundaries of computational domain, the nozzle surface. Hybrid RANS/ILES method [2, 3] is used to solve equations. The RANS method is used near walls and ILES is used in the rest of the computational domain. Scheme viscosity plays a role of a subgrid scale (SGS) model. The Roe method was applied to calculate non-viscous flux on cell faces. The high resolution

[*]E-mail: savin@jscc.ru
[**]E-mail: leosun.ben@gmail.com
[***]E-mail: dalyubimov@ya.ru
[****]E-mail: rybakov.aax@gmail.com

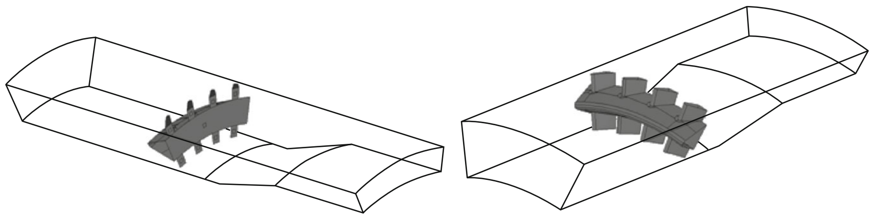

Fig. 1. The general view of the computational model for sector of flame tube.

of this method is provided by using a monotone difference scheme MP9 [4] with upwind 9th-order approximation to calculate flow parameters on cell faces. This approach has been successfully used in [5]. Diffusion fluxes are calculated on cell faces with second-order approximation by central differences. The time discretization is made with second order by implicit scheme and with integration by double time method. The Spalart–Allmaras turbulence model is used in RANS region. The WENO-5 scheme [4] is used to calculate convective flows on cell faces in the difference analog of turbulence model equation. In LES region, the Spalart–Allmaras turbulence model is modified so that the turbulent viscosity is equated to zero. This is achieved by changing the distance in dissipative term of turbulence model equation. The modified distance $\tilde{d}$ is calculated by the formula

$$\tilde{d} = \begin{cases} d, & d \leq C_{ILES}\Delta_{MAX}, \\ 0, & d > C_{ILES}\Delta_{MAX}, \end{cases} \tag{1}$$

where d is the distance from the wall to the cell center, Δ_{MAX} is the maximum size of the cell, C_{ILES} is constant defining position of transition from RANS to ILES and equal 0.65.

3. COMPUTATIONAL GRIDS AND SIMULATION PARAMETERS

The computational grid of the full-size flame tube is obtained by cloning and rotating the grid of the flame tube sector. The size of the grid for the sector is 9.4×10^6 cells, for a full-size flame tube there is 94×10^6 cells. The supply of passive impurity, simulating fuel, is made through the holes on the stabilizer, as well as through the holes on the ends of the pylons located above and below the stabilizer. The general view of the front device, as well as a grid on its surface, are shown in Figs. 1–3.

Total temperature and pressure is 300 K and 10^5 Pa at the inlet of the flame tube. The twist angle of flow is 40°. Static pressure at the end of flame tube is 89560 Pa. The velocity of flow between pylons compose 100–120 m/s the inlet velocity is 32 m/s for this pressure ratio. The velocity vector and the total temperature is fixed at the hole on pylons end. Velocity of passive impurity on pylons end is 32 m/s and twist angle is 40°. Velocity of passive impurity on front device is 42 m/s and angle is a normal to the end of front device. The total temperature of passive impurity is 300 K.

In Fig. 4 shows the fields of instantaneous distribution of axial velocity in sections passing through the holes on the hole on the stabilizer. The essentially unsteady nature of the flow is seen. The zone with a reverse flow is formed behind the stabilizer. The flow is accelerated above and below the stabilizer due to section load. The same can be seen in the Fig. 5 where the isosurfaces of the concentration of a passive impurity are shown. Due to the recirculation zone formation the jet of passive impurity from the holes on the stabilizer propagates almost along the surface of the front device.

The calculation results show a slight difference in the numerical simulation of the flame tube for sector of 36° and full 360°. The recovery coefficient of the total pressure in the outlet of flame tube is 6% and 6.13%. The difference in the averaged flow parameters in the outlet is small and amounts to less than 1% for the distribution of the total and static pressure and about 13% for the distribution of the concentration of a passive impurity. The greatest differences are observed in the distribution of the pulsating characteristics of the flow, so the pressure pulsations differ by 5–17%, the energy of turbulence by 5–12%. In this case, the level of pulsations is about 0.2% and 14–15% of the values of the averaged pressure and velocity in this section. The rate of increase in the mixing completeness whole combustion chamber is slightly lower (Fig. 6), however, the η level at the exit of the flame tube is almost the same for both cases. The mixing completeness is determined by the ratio $\eta = \frac{1-C_{\max}}{1-C_{av}}$, where $C_{\max}$ is

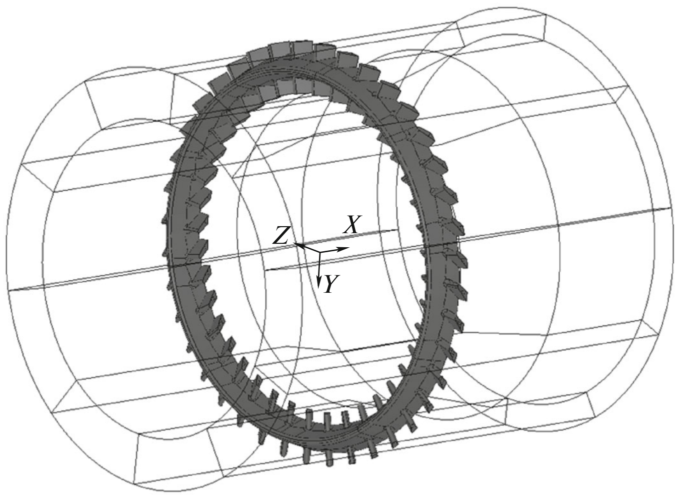

Fig. 2. General view of the computational model for full geometry of flame tube.

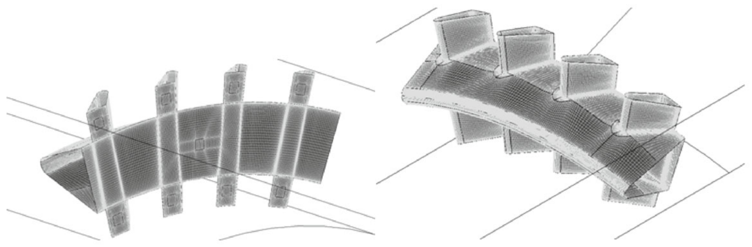

Fig. 3. The computational grid on front device and pylons.

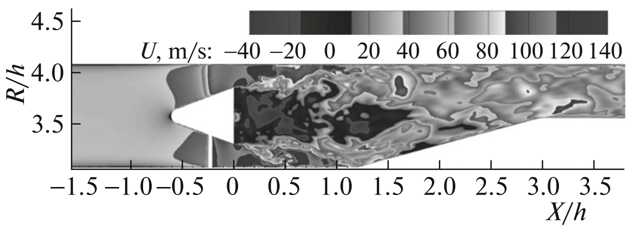

Fig. 4. Instantaneous axial velocity at longitudinal cut of flame tube (h is inlet height of flame tube).

maximum concentration in the section and C_{av} is averaging in cross-section value of passive impurity concentration.

The calculation results show a slight difference in the numerical simulation of the flame tube for sector of 36° and full 360°.

4. REALIZATION OF LINKED BORDER CONDITIONS

The RANS/ILES method works with block-structured grids [6, 7], which consist of separate blocks interconnected by interfaces. Each block of the computational grid is an ordered three-dimensional array of cells, which can be accessed by indices. This greatly simplifies the work with the grid within one block. With each cell a set of gas-dynamic parameters is associated. Each block of the computational grid has its own three-dimensional curvilinear coordinate system, the axes of which are directed along the edges of the block. Interfaces describing the contact of blocks with each other contain information about the coordinates of the touch of blocks areas (in terms of nodes of the computational grid), as well as information about the matching of coordinate systems of the related blocks. Some faces of the grid blocks do not touch other blocks, but are the borders of the computational domain. In this case, it is

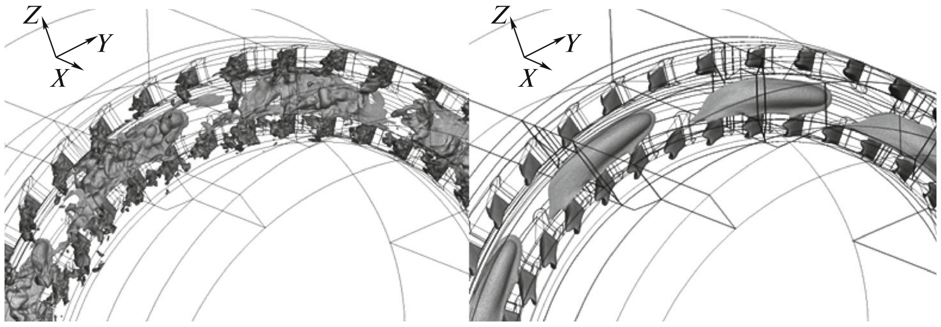

Fig. 5. Instantaneous isosurface and time average isosurface of passive impurity for concentration 0.1.

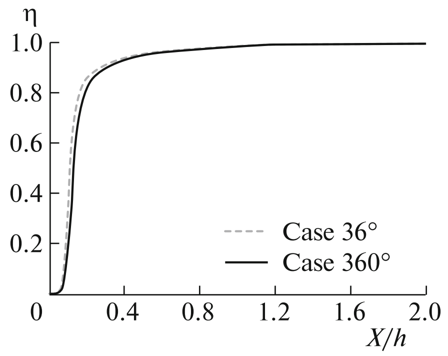

Fig. 6. Distribution of the completeness of mixing along the length of the combustion chamber.

necessary to describe the border conditions. This is done with the help of special objects that describe the border conditions on a given rectangular section of the block boundary [8].

When carrying out calculations, calculation grids consisting of identical sectors are often used. That is, in essence, a grid with a smaller number of cells, duplicated several times and rotated relative to the original one around a given straight line. To obtain a solution to the gas-dynamic problem, it is sufficient to perform calculations for only one such sector. However, during the calculations it is necessary to observe additional conditions. When calculating a separate sector, it is necessary to take into account that the output flows at the individual block borders must coincide with the input flows at the borders of other blocks, and these borders are spaced apart. Such border conditions will be called linked. To increase the efficiency of the calculations using the RANS/ILES method, this type of border conditions was implemented.

One of the main operations performed on the linked border conditions is to determine whether they correspond to each other. With automatic marking of the linked border conditions of the selected sector of the computational grid, a compliance check is performed, with which pairs of the linked border conditions are formed. Two types of search for connections between border conditions are implemented: border conditions combined by parallel transfer and rotation around a given axis. To determine whether the two border conditions are consistent with each other, it is needed to combine them with the help of a given type of movement so that their centers coincide, and then check whether their corner nodes coincide (Fig. 7).

Checking the coincidence of the corner nodes of two potentially linked border conditions is determined by considering four possible cases, taking into account the renaming of vertices: the quadrangle $A_0B_0C_0D_0$ can coincide with one of the quadrangles $ABCD$, $DABC$, $CDAB$, $BCDA$ (Fig. 8).

The order of checking the correspondence of the corner nodes $A_0B_0C_0D_0$ and $ABCD$ of the two border conditions is presented in Fig. 9. In addition to the fact that the corner nodes of a pair of border conditions coincide, a correspondence between their coordinate systems is also established. This allows for any cell of a single border condition to find the corresponding cell from the associated border condition. All further transformations of the associated border conditions (renaming, splitting the border condition or parent blocks) are performed only simultaneously.

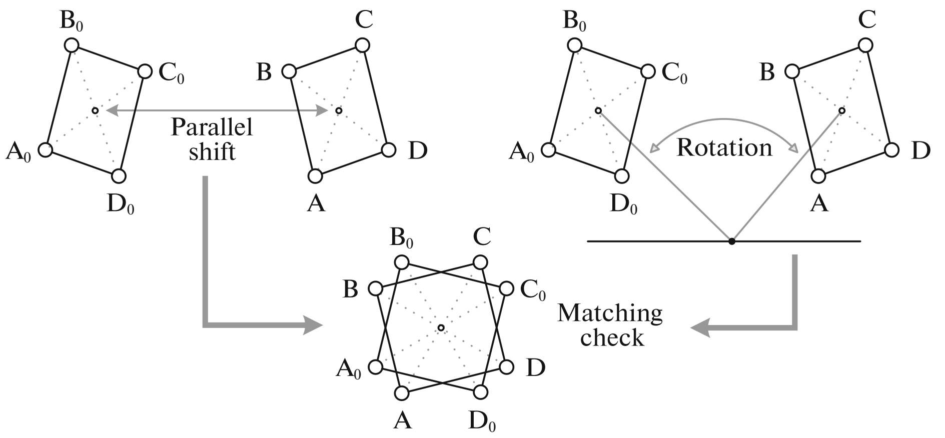

Fig. 7. Connection of linked border conditions to each other using parallel shift and rotation around a straight line.

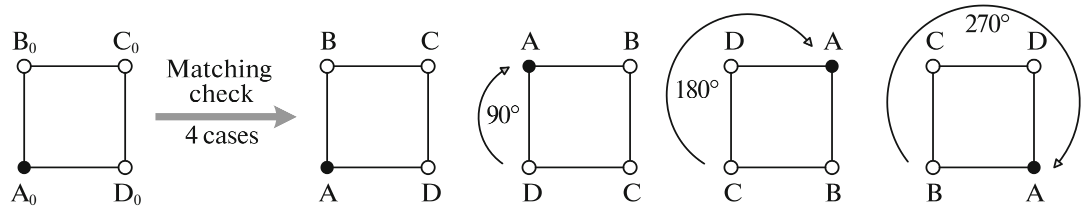

Fig. 8. Four cases for checking the coincidence of the corner points of the border conditions.

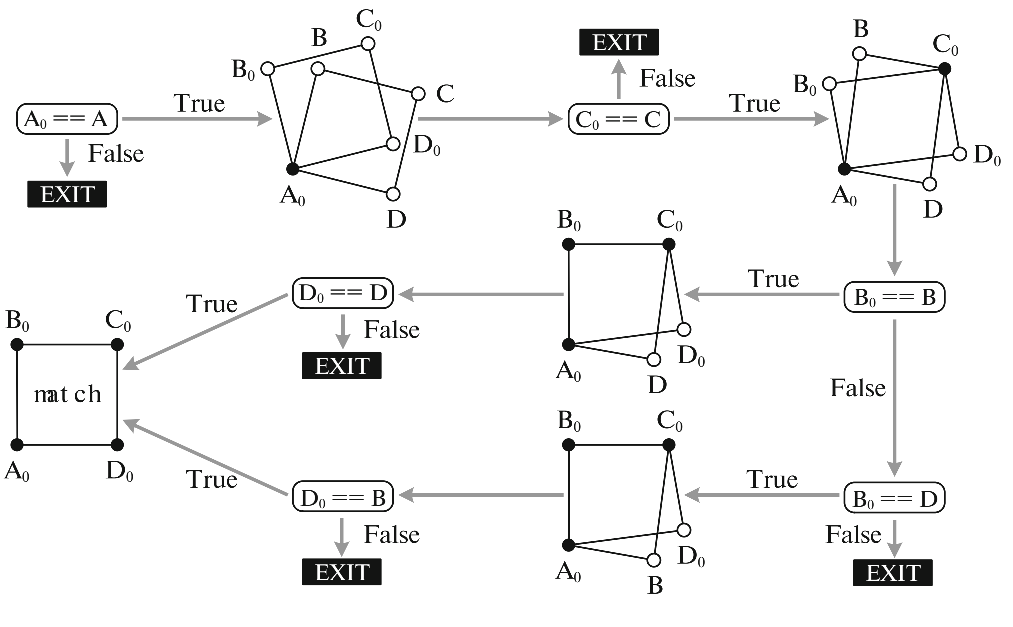

Fig. 9. Checking the coincidence of the corner points of the border conditions.

5. SCALABILITY OF CALCULATIONS INVESTIGATION WHEN CUTTING THE COMPUTATIONAL GRID

In this section, we describe the experimental launches and analyze the scalability results obtained using the mechanisms of coupled border conditions and fragmentation of the computational grid.

For testing, we used a computational grid describing the flame tube and containing 94 million cells (we will call it a complete grid). The mechanism of linked border conditions allowed us to replace the

Table 1. Characteristics of computational grids

Grid description	Blocks/scopes	Interfaces	Border conditions	Linked border cond.
360 deg. (94 mln cells)	300	1796	1643	0
36 deg. (10.7 mln cells)	30	152	204	14
36 deg., cut for 16 proc. (10% dev.)	39	224	218	14
36 deg., cut for 32 proc. (10% dev.)	54	340	248	14
36 deg., cut for 64 proc. (10% dev.)	170	982	429	23
36 deg., cut for 64 proc. (1% dev.)	383	2356	682	34

calculations on this complete grid with calculations made on a sector representing the 1/10 part of the original grid (36 degrees relative to the OX axis). The computational grid describing one sector contains 10.7 million cells, which is 9 times smaller than the initial grid. At the same time, the calculation time is also reduced by 9 times.

The experiments were performed on a segment of the MVS-10P supercomputer located at the JSCC RAS. This segment consists of computing nodes based on Intel Xeon Phi Knights Landing 7290 microprocessors [9]. From 1 to 32 computing nodes were used for launches.

First, calculations were carried out on a complete grid. This grid contains 300 calculation blocks, calculations on it are scaled to 32 nodes linearly with MPI [10] (the execution time of calculations at 32 nodes is less than the calculation time on one node by almost 32 times).

The computational grid describing one sector contains only 30 nodes, so the calculations on it are much worse scaled. To analyze the scalability, crushing of this grid was performed by dividing the largest block in half. In this algorithm, at each iteration, a block is selected that contains the largest number of cells and is divided in half. The algorithm finishes its work when the resulting blocks can be distributed between the specified number of processes in such a way that the maximum deviation from the average value does not exceed the specified threshold. The Table 1 shows examples of crushing, for which calculations were subsequently made.

In Fig. 10 adjacency graphs of computational grids blocks obtained as a result of crushing are presented. The nodes of the adjacency graph of the computational grid blocks are the centers of the grid blocks. Two nodes of the graph are connected by an edge if the two corresponding blocks are adjacent.

Figure 11 shows the data of calculation runs performed on the same sector (small grid) with various parameters of the grid crushing. It can be seen that the acceleration of calculations on the initial grid reaches approximately 4–5, and then regardless of the number of nodes used do not change. This indicates that there is a large block in the grid, so the processing speed of the entire grid is limited from below by the processing speed of this block by one process. When cutting the grid for distribution by 16 processes (with a maximum deviation of 10%), the scaling of the calculations improves, but with an increase in the number of computing nodes to 16 or more, the same effect is observed. When cutting the grid for distribution to 32 or 64 processes (with a maximum deviation of 10%), the results of scaling are much better. The dependence of the acceleration on the number of nodes used becomes almost linear, although with a large number of computational nodes, some fluctuations are observed.

Also in Fig. 11 the data of the launches performed on the grid, which is fragmented very finely. The grid was cutted for distribution to 64 processes, while the maximum allowable deviation from the average computational load was taken as 1%. The graph shows a very serious drawdown of scaling in the range from 10 to 20 computing nodes. This is due to the fact that unnecessarily fine crushing leads to the appearance of a huge number of computational blocks and interfaces between them. In some cases, blocks can be distributed between processes so that the amount of data involved in interprocess exchanges between them, becomes critical. In such situations it is necessary to take into account the factor of interprocess exchanges or not to resort to excessive fragmentation of the grid.

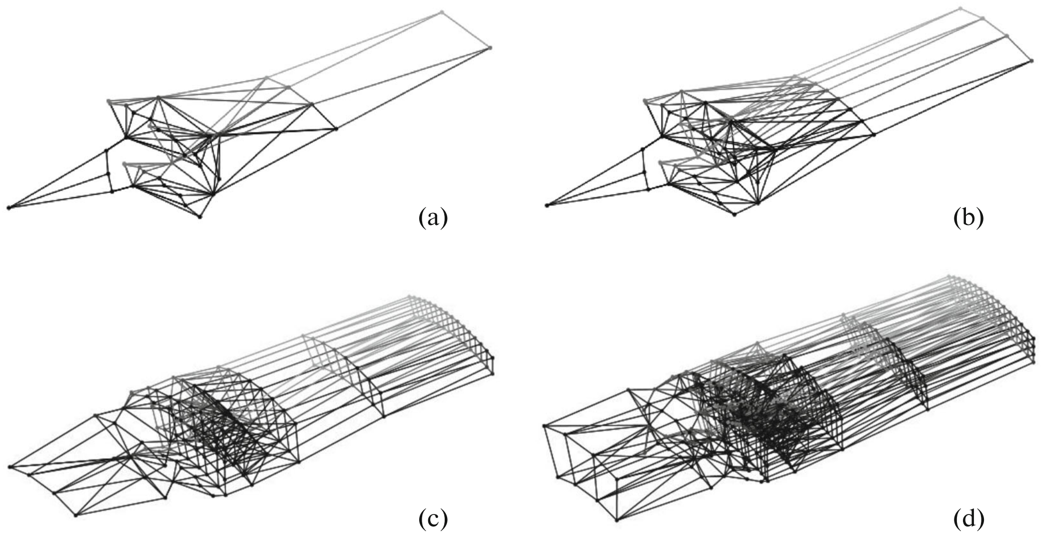

Fig. 10. Borders adjacency graphs for different cases of computational grid cutting: (a) cutting for 16 processes (10% deviation), (b) cutting for 32 processes (10% deviation), (c) cutting for 64 processes (10% deviaton), (d) cutting for 64 processes (1% deviation).

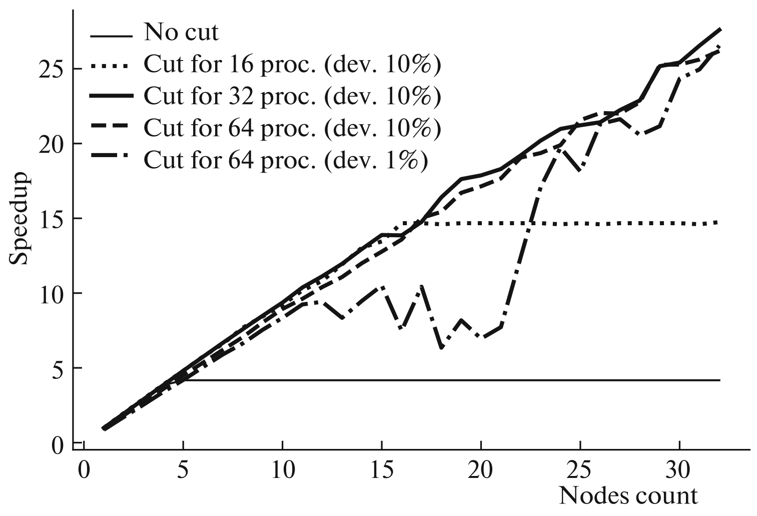

Fig. 11. Graph scaling calculations when using different options for cutting the grid.

6. CONCLUSION

The results of practical experiments have shown that the mechanisms for preparing the block-structured computational grid for the RANS/ILES method help to achieve multiple acceleration of calculations when performing calculations on a supercomputer. In particular, the calculations on the grid considered in the article, containing 94 million cells, were replaced by calculations on a smaller grid using the mechanism of linked border conditions. This small grid was fragmented to achieve linear scaling when running on 32 nodes of the supercomputer. The total decrease in the counting time achieved using the two mechanisms described above exceeded 2 hundred times when using 32 compute nodes.

FUNDING

The work has been done at the Joint Supercomputer Center of the Russian Academy of Sciences as part of the state assignment for the topic 0065-2019-0016 (reg. no. AAAA-A19-119011590098-8). The supercomputer MVS-10P, located at the JSCC RAS, was used for calculations during the research.

REFERENCES

1. D. A. Lyubimov, “The use of the hybrid RANS/ILES approach for the investigation of three-dimensional separated turbulent flows in curvilinear diffusers,” High Temp. **48**, 261–271 (2010).
2. D. A. Lyubimov, “Development and application of a high-resolution technique for jet flow computation using large eddy simulation,” High Temp. **50**, 420–436 (2012).
3. L. A. Benderskii, D. A. Lyubimov, A. O. Chestnykh, B. M. Shabanov, and A. A. Rybakov, “The use of the RANS/ILES method to study the influence of coflow wind on the flow in a hot, nonisobaric, supersonic airdrome jet during its interaction with the jet blast deflector,” High Temp. **56**, 247–254 (2018).
4. A. Suresh and H. T. Huynh, “Accurate monotonicity-preserving schemes with Runge–Kutta time stepping,” J. Comput. Phys. **136**, 83–99 (1997).
5. F. F. Grinstein, L. G. Margolin, and W. J. Rider, *Implicit Large Eddy Simulation: Computing Turbulent Fluid Dynamics* (Cambridge Univ. Press, Cambridge, 2007).
6. M. Farrashkhalvat and J. P. Miles, *Basic Structured Grid Generation, With an Introduction to Unstructured Grid Generation* (Butterworth-Heinemann, London, 2003).
7. V. Liseikin, *Grid Generation Methods* (Springer, Netherlands, 2010).
8. A. A. Rybakov, “Inner respresentation and crossprocess exchange mechanism for block-structured grid for supercomputer calculations,” Program. Sist.: Teor. Prilozh. **32** (8), 121–134 (2017).
9. J. Jeffers, J. Reinders, and A. Sodani, *Intel Xeon Phi Processor High Performance Programming, Knights Landing Edition* (Morgan Kaufmann, 2016).
10. M. Queen, *Parallel Programming in C with MPI and OpenMP* (McGraw-Hill, New York, 2004).