Parallel implementation of a numerical method for solving transport equations for the mesoscale meteorological model TSUNM3

A V Starchenko, E A Danilkin, S A Prohanov and D V Leshchinskiy

Tomsk State University, Tomsk, Russian Federation, 634050

E-mail: starch@math.tsu.ru, danilkine@gmail.com, sergey@prohanov.com, 360flip182@gmail.com

Abstract. A parallel algorithm for numerical solution of a generalized unsteady 3-dimensional advection-diffusion equation for a mesoscale meteorological model TSUNM3 is considered. Efficiency of parallel implementation of the solver with Message Passing Interface (MPI), Open Multi-Processing (OpenMP), Open Accelerators (OpenACC), and NVidia Compute Unified Device Architecture (CUDA) technologies was compared. The efficiency of the MPI-parallelized code of TSUNM3 model with a two-dimensional domain decomposition strategy applied is not less than 75% on 256–512 processes on Cyberia supercomputer of Tomsk State University.

1. Introduction
Mathematical modeling is widely applied to studying the environment along with experimental investigations. It allows studying and forecasting the development of atmospheric, hydrological, and geological processes on the Earth without actively influencing them. Inhomogeneous unsteady three-dimensional advection-diffusion equations with source terms are the basis of the majority of the mathematical models of continuum mechanics, hydrometeorology, and ecology. These equations describe distributions of density, velocity components, temperature, concentration, and turbulence characteristics. Modern mathematical models are the systems of several advection-diffusion equations with an algebraic closure model and boundary conditions.

Finite difference, finite volume, and finite element methods are applied to solve these systems of equations. Very fine meshes of several millions of nodes are used in these computations. Moreover, in predictive modeling of atmospheric and hydrological processes the solution should be provided in a very short time. Modern multi-core and multi-processor computational systems with graphic processor units (GPUs) can significantly speed up these computations [1, 2].

The aim of this work was to make computational experiments on a hybrid computational system and define the technology (MPI, OpenMP, OpenACC, CUDA) which is the most efficient for building parallel programs for solving a system of inhomogeneous unsteady three-dimensional advection-diffusion equations. The results obtained were applied to developing a parallel version of the high-resolution mesoscale meteorological model TSUNM3, which is being developed in Tomsk State University (TSU) for predicting hazardous weather phenomena and air quality in a city [3].
2. Mathematical formulation of a problem and a numerical method

2.1. Mesoscale meteorological model TSUNM3

The mesoscale meteorological model TSUNM3 (Tomsk State University Nonhydrostatic Mesoscale Meteorology Model) predicts the values of wind velocity components, temperature, and humidity in the atmospheric boundary layer on 50 vertical layers (up to 10 km from the surface). The size of the area of research is 200x200 km (mesh step is 2 km) with nested area of 50x50 km with the city of Tomsk in the center (mesh step for the nested area is 1 km). The model is initialized with the numerical forecast obtained with the SL-AV operational global scale model [4] of the Hydrometeorological Research Center of the Russian Federation. The main features of TSUNM3 model are:

- a nonhydrostatic approximation of the momentum equations and a quasi-steady approximation of the continuity equation [3];
- a coordinate system with variable resolution in vertical that follows the shape of the surface [5, 6];
- nested computational areas with one-way interaction [7];
- radiation conditions for the components of horizontal velocity, temperature, and humidity at the lateral boundaries [7, 8] that account for the spatial and temporal tendencies in the values of the dependent variables that are generated by the model of a larger scale (SL-AV model of Hydrometcenter of Russia [4]);
- predictive model for the soil temperature that is based on a heat-transfer equation and a diagnostic relation for the humidity of the surface layer of the soil [6, 7];
- surface heat fluxes are computed based on the Monin-Obukhov similarity theory [9];
- shortwave and longwave radiation with the influence of the cloudiness is considered [10, 11];
- WSM6 [12] microphysics scheme, which considers formation of the raindrops, clouds, snow, ice crystals, ice pellets from atmospheric moisture;
- turbulent structure of the atmospheric boundary layer is modeled based on the transport equation for the turbulent energy and algebraic relations for the turbulence length scale and turbulent diffusion [13].

Mathematical formulation of the TSUNM3 model includes 11 unsteady inhomogeneous three-dimensional advection-diffusion equations (equations for 3 components of velocity vector, temperature, turbulent energy, humidity, raindrops, snowflakes, cloud moisture, ice crystals, ice pellets) and algebraic relations, which are computed independently for each (x,y).

2.2. Mathematical formulation of the problem

Generalized advection-diffusion equation

\[
\frac{\partial \rho \Phi}{\partial t} + \frac{\partial \rho U \Phi}{\partial x} + \frac{\partial \rho V \Phi}{\partial y} + \frac{\partial \rho W \Phi}{\partial z} = \frac{\partial}{\partial x} \left( K_{xy} \frac{\partial \Phi}{\partial y} \right) + \frac{\partial}{\partial y} \left( K_{xy} \frac{\partial \Phi}{\partial x} \right) + \frac{\partial}{\partial z} \left( K_{z} \frac{\partial \Phi}{\partial z} \right) + S_{\Phi} \tag{1}
\]

is considered.

Here \( t, x, y, z \) are the time and spatial coordinates (\( X \) is the eastward axis, \( Y \) is the northward, and \( Z \) is the upward axis); \( U, V, W \) are the components of the velocity vector; \( K_{xy}, K_{z} \) are the turbulent exchange coefficients; \( S_{\Phi} \) is the source term. Velocity components, potential temperature of the air, absolute humidity, concentration of the raindrops, cloud moisture, snow, ice crystals, or ice pellets \( (U, V, W, \Theta, k, q_{rain}, q_{cloud}, q_{snow}, q_{ice}, q_{grapel}) \) can be considered as \( \Phi \), which also can be equal to 1 in the continuity equation.
The transformation
\[ x' = x, \; y' = y, \; z' = \frac{z - h(x, y)}{H - h(x, y)} \]  
\[ (2) \]
is applied to the equation (1) to account for the influence of the Earth’s surface elevation \( h(x, y) \).

The third-type boundary conditions (3) are used on the surface:
\[ z = h(x, y) : K_z \frac{\partial \Phi}{\partial z} = \alpha \Phi - \Phi_0 \]  
\[ (3) \]

The second-type boundary conditions (4) are used at the upper boundary:
\[ z = H : K_z \frac{\partial \Phi}{\partial z} = \gamma_\Phi \]  
\[ (4) \]

Radiation boundary conditions [8]:
\[ x = -L; x = L : \frac{\partial \Phi}{\partial t} + C_\text{omega} \frac{\partial \Phi}{\partial x} = \frac{\partial \Phi_\text{omega}}{\partial t} + C_\text{omega} \frac{\partial \Phi_\text{omega}}{\partial x} \]  
\[ (5) \]
\[ y = -L; y = L : \frac{\partial \Phi}{\partial t} + C_\text{omega} \frac{\partial \Phi}{\partial y} = \frac{\partial \Phi_\text{omega}}{\partial t} + C_\text{omega} \frac{\partial \Phi_\text{omega}}{\partial y} \]  
\[ (6) \]
are used at the lateral boundaries.

Subscript «\text{S}» indicates the parameters that are defined from the global scale model [4]; \( C^x_\Phi, C^y_\Phi \) are the phase velocity components [8], \( 2L \) is the dimension of the area in horizontal.

The values of \( \Phi_\text{omega} \) was interpolated to the fine mesh, which was build for the mesoscale meteorological model, from the coarser mesh of a global model and used as initial conditions for the equation (1).

2.3. Numerical method
A structured mesh that is uniform in X and Y directions and refining towards the Earth’s surface in Z direction was used to solve equations (1)–(6). Differential equations were approximated with the finite volume method, which is the second-order accurate in space [14]. Semi-implicit Adams–Bashforth and Crank–Nicolson approximations were used to ensure the second-order accuracy in time. Each term of the equation (1) was integrated by each finite volume to obtain the discrete equation:

\[
\frac{\rho_p^{m+1} \Phi_p^{m+1} - \rho_p^m \Phi_p^m}{\tau} \bigg|_{Vol_p} + \\
\frac{3}{2} \int \int \int_{Vol_p} \left[ \frac{\partial \rho U \Phi}{\partial x} + \frac{\partial \rho V \Phi}{\partial y} + \frac{\partial \rho W \Phi}{\partial z} - \frac{\partial}{\partial x} \left( K_{\omega x} \frac{\partial \Phi}{\partial x} \right) - \frac{\partial}{\partial y} \left( K_{\omega y} \frac{\partial \Phi}{\partial y} \right) - S_\Phi \right]^{m} dxdydz - \\
\frac{1}{2} \int \int \int_{Vol_p} \left[ \frac{\partial \rho U \Phi}{\partial x} + \frac{\partial \rho V \Phi}{\partial y} + \frac{\partial \rho W \Phi}{\partial z} - \frac{\partial}{\partial x} \left( K_{\omega x} \frac{\partial \Phi}{\partial x} \right) - \frac{\partial}{\partial y} \left( K_{\omega y} \frac{\partial \Phi}{\partial y} \right) - S_\Phi \right]^{m-1} dxdydz - \\
\frac{1}{2} \int \int \int_{Vol_p} \left[ \frac{\partial}{\partial z} \left( K_z \frac{\partial \Phi}{\partial z} \right) \right]^{m+1} dxdydz - \frac{1}{2} \int \int \int_{Vol_p} \left[ \frac{\partial}{\partial z} \left( K_z \frac{\partial \Phi}{\partial z} \right) \right]^{m} dxdydz = 0
\]

Here \( m \) is a number of a time step; \( (x_p, y_p, z_p) \) are the coordinates of the center of a finite volume \( Vol_p \). The value of the time step \( \tau \) was chosen from the stability condition for the finite-difference scheme [15]. Implicit approximation of diffusion transport in vertical direction, which is very
significant in the atmospheric boundary layer, reduces the limitation to the time step. Approximating integrals in (7) with simple interpolation formulas gives SLAE (8):

\[-c_{i,j,k} \Phi^{m+1}_{i,j,k} + e_{i,j,k} \Phi^{m+1}_{i,j,k} - d_{i,j,k} \Phi^{m+1}_{i,j,k+1} = \frac{3}{2} \left( ap_{i,j,k} \Phi^{m}_{i,j,k} + ae_{i,j,k} \Phi^{m}_{i+1,j,k} + an_{i,j,k} \Phi^{m}_{i,j+1,k} + at_{i,j,k} \Phi^{m}_{i,j,k-1} \right) +
\]
\[+ \left( aw_{i,j,k} \Phi^{m}_{i-j-1,k} + as_{i,j,k} \Phi^{m}_{i-1,j-1,k} + ab_{i,j,k} \Phi^{m}_{i,j-1,k-1} \right) +
\]
\[-\frac{1}{2} \left( ap_{i,j,k} \Phi^{m-1}_{i,j,k} + ae_{i,j,k} \Phi^{m-1}_{i+1,j,k} + an_{i,j,k} \Phi^{m-1}_{i,j+1,k} + at_{i,j,k} \Phi^{m-1}_{i,j,k-1} \right) +
\]
\[+ \left( aw_{i,j,k} \Phi^{m-1}_{i-j-1,k} + as_{i,j,k} \Phi^{m-1}_{i-1,j-1,k} + ab_{i,j,k} \Phi^{m-1}_{i,j-1,k-1} \right) +
\]
\[+ c_{i,j,k} \left( \Phi^{m}_{i,k-1,j} - \Phi^{m}_{i,k,j} \right) + d_{i,j,k} \left( \Phi^{m}_{i,k+1,j} - \Phi^{m}_{i,k,j} \right),
\]
\[i = 1, N_x; j = 1, N_y; k = 1, N_z; m = 1, 2, 3, \ldots \]

Here \( \{ \Phi^{m+1}_{i,j,k} \} \) is an unknown scalar function; \( ap_{i,j,k}, ae_{i,j,k}, aw_{i,j,k}, as_{i,j,k}, at_{i,j,k}, ab_{i,j,k} \) are the coefficients of the scheme. Convective terms of the equation (7) were approximated with the monotonic linear upwind scheme MLU of van Leer [14].

The difference scheme built is a cost-effective and allows applying the tridiagonal matrix algorithm (TDMA) to define values of \( \{ \Phi^{m+1}_{i,j,k} \} \).

The runtime of the program was estimated for the case of admixture transport from the constant emission point source in the area of 50 (2L) to 50 (2L) to 0.6 (H) km. The source was located in 0.015L km in X-direction, 0.015L km in Y-direction, and 0.0101H km in Z-direction from the center of the domain. Velocity components were considered constant: \( U=1 \) m/s, \( V=1 \) m/s, and \( W=0 \) m/s. Diffusion coefficients were also considered constant \( K_{xy}=K_z=100 \) m^2/s. Time step \( \tau \) was set to 6 s and the modeling was finished at \( Time_{Fin} = 5000 \tau \). The mesh of 256x256x32 nodes was used in computations. The accuracy of the numerical solution was controlled with the analytical solution of the equation considered [16].

3. Parallel implementations of the numerical method

After the sequential program was verified and tested it was parallelized with OpenMP, MPI, OpenACC, and CUDA technologies. Computations were run on both shared-memory system Thor and the supercomputer Cyberia of TSU. The average runtime of the sequential program was 644.5 s on Thor server and 1915.9 s on a single node of Cyberia supercomputer of TSU.

Shared-memory system Thor is a dual processor (2xIntel® Xeon® Silver 4214 (2.20GHz)) server with 192Gb RAM and 2 GPUs NVidia RTX2080 Ti that is used for running TSUNM3 model. The characteristics of the node of Cyberia supercomputer are: 48Gb RAM, 2xIntel® Xeon® X5670 (2.93GHz).

3.1. Message Passing Interface technology

MPI is a message-passing library specification for distributed memory systems. At the same time MPI functions can be implemented on the shared memory system, which extends possible applications of the technology. MPI includes functions for explicit distribution of computations among processes and organization of data exchange.

In this work two-dimensional (along X-axis and Y-axis) domain decomposition was chosen as the main parallelization strategy with MPI technology (figure 1). The values of the discrete function \( \{ \Phi^{m}_{i,j,k} \} \) belong to different subdomains were distributed among processes and the next time step values \( \{ \Phi^{m+1}_{i,j,k} \} \) were computed with the TDMA algorithm.
Figure 1. Two-dimensional decomposition of the domain along X and Y axes and a scheme of data transfers between subdomains

Because of the stencil of the semi-implicit difference scheme (8), the values of the grid function \( \{ \Phi_{n+1}^{i,j,k} \} \) from the adjacent process are required to calculate the next time step values at the near-boundary nodes of each subdomain. Therefore, ‘ghost cells’ (blank circles in figure 1) were created on each process to store data from an adjacent process values of the function \( \{ \Phi_{n+1}^{i,j,k} \} \) from the adjacent subdomains and data exchange between subdomains was organized. This approach ensures the uniformity of the computations in all nodes of a subdomain [17].

Figure 2. MPI-program speedup (left) and OpenMP-program speedup (right)
MPI functions for creating two-dimensional Cartesian virtual topology, creating datatypes, and organizing effective communication between processes were used in this work. The values of the discrete function were sent to the adjacent subdomains by blocking point-to-point function MPI_SENDRECV().

Speedup of the MPI-program on different number of processes is shown in table 1 and figure 2 (on the left). The results of the computations made on the shared-memory system Thor show that the speedup increment slightly slowed down when 24 processes were used due to the increase in memory load: memory bandwidth per active core decreases when all available physical cores involved. The speedup of the computations performed on Cyberia supercomputer is almost linear. MPI program ran on all available cores of the dual processor server provided the solution in 36.7 s. The same program ran on 128 cores of the Cyberia supercomputer required 16.3 s.

3.2. **Open Multi-Processing technology**

OpenMP API is a tool for developing parallel programs for multiprocessor (multicore) shared-memory computing systems. The main task of a programmer who chose this parallelizing technology is distributing computations among threads working in parallel with shared data. Compiler directives, library routines, and environmental variables are the work-sharing instruments of OpenMP [18]. In the case considered special attention was paid to parallelizing the code of the main computational block of the algorithm, where values of the function \( \{ \Phi^{k+1}_P \} \) for each time step are computed. The computational core of the algorithm consists of loops on i and j that go over all the grid nodes in the XY-plane. For each node (i, j) in the XY-plane the numerical solution of the equation (5) was found nodes of a vertical mesh line with the TDMA. To parallelize the sequential algorithm developed it is sufficient to introduce ‘omp parallel for’ directive, which runs the following block of code in parallel, right before the loops on i and j.

| Processes   | 1   | 2   | 4   | 8   | 12  | 16  | 24  | 32  | 64  | 128 |
|-------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| MPI – Thor  | 1.0 | 2.0 | 3.9 | 7.3 | -   | 13.7| 17.6| -   | -   | -   |
| MPI – Cyberia| 1.0 | 2.0 | 3.9 | 7.8 | -   | 15.0| 23.5| 31.0| 60.3| 117.7|
| OpenMP – Thor| 1.0 | 2.0 | 3.8 | 7.1 | -   | 13.5| 16.5| -   | -   | -   |
| OpenMP – Cyberia | 1.0 | 2.0 | 3.8 | 7.4 | 10.5| -   | -   | -   | -   | -   |

Results of the computations with OpenMP-parallelized program are shown in the table 1 and figure 2 (on the right). Deviation from linear acceleration appears when more than 16 threads on Thor server used (figure 2) due to the increase in memory load as it was for MPI-parallelized code. Only 12 physical cores with shared memory are available on the node of Cyberia supercomputer. The speedup of the parallel program ran on the node of Cyberia is equal to 10.

3.3. **Open Accelerators technology**

OpenACC technology is used for parallel programming for heterogeneous computing systems with both CPUs and GPUs. The architecture of a GPU that were initially specialized for highly parallel computations is significantly different from the architecture of general-purpose CPU.

OpenACC technology as well as OpenMP implies introducing compiler directives to indicate blocks of a sequential code that should be performed in parallel on a GPU. It also has instruments for splitting variables to shared and private. Special attention is also paid to parallelizing loops to run cycling operations on the GPU.

The execution time of the OpenACC-parallelized program on NVidia RTX2080 Ti GPU is about 20.4 s. This is significantly less than the execution of the sequential code and comparable to the
execution time of the MPI-parallelized program on 24 processes on shared-memory Thor server or 128 processes on Cyberia supercomputer.

3.4. CUDA
CUDA is a parallel computing platform developed by NVidia Corporation to perform parallel computing on Graphic Processor Units (GPUs). Parallelizing strategy for the algorithm considered was based on two-dimensional decomposition of the domain. 256x256 threads were used in computations and each thread independently computed Nz values in the nodes of a vertical line (i,j). CUDA kernel was computed for each time step and then synchronization of threads was performed. CUDA program solved the task considered in 16.6 s which is the best result on Thor server among parallelization technologies considered.

4. Parallel implementation of the mesoscale meteorological model TSUNM3 with MPI technology
The release version of TSUNM3 model is parallelized with MPI technology. Speedup and efficiency of the parallel program modeling the mesoscale meteorological processes took place during 2 days on the meshes of 50x50x50, 98x98x50, and 194x194x50 nodes are shown on figure 3.

The size of the mesh chosen for the computations was defined by the size of the area modeled. In the first case it was an area with a city with population less than a million people in the center (the step of the mesh in horizontal directions was 1–2 km, the size of the area is 50–100 km). In the second case the area of modeling was an industrial area with many factories and a megalopolis in the center of the area (the size of the area is about 100–200 km). In the third case it was a region (the size is up to 400 km). Computations with the mesoscale meteorological model for the meshes chosen were performed on Cyberia supercomputer of TSU.

From figure 3 is clear that the parallel algorithm developed is efficient and well scalable for large enough number of processes (up to 256) and meshes with large number of nodes. The results obtained show that the software developed fulfills requirements for operational weather forecasting computations that the forecast for a day should be done in less than an hour of computing on a supercomputer.

![Figure 3. Speedup and efficiency of the parallel program for computing mesoscale meteorological processes over the airport](image-url)
Table 2. Speedup and efficiency of the parallel program for meshes with different number of nodes

| Processes | speedup | 4 | 9 | 16 | 36 | 64 | 144 | 256 | 576 |
|-----------|---------|---|---|----|----|----|-----|-----|-----|
| 50x50x50  | efficiency | 3.9 | 8.5 | 14.8 | 31.6 | 53.9 | 102.4 | 145.5 | - |
| 98x98x50  | efficiency | 3.9 | 8.7 | 15.3 | 33.0 | 57.9 | 121.5 | 199.6 | 319.7 |
| 194x194x50 | efficiency | 4.0 | 8.9 | 15.6 | 33.7 | 60.2 | 130.6 | 226.4 | 447.8 |

5. Conclusions
The results of the computations made for solving the three-dimensional admixture transfer equations with the semi-implicit method on the mesh of more than 2 million cells showed that the speedup and efficiency of the OpenMP program are almost equal to ones of the MPI program. MPI-parallelized program with two-dimensional domain decomposition strategy applied on all the cores of the dual processor server provided solution in 36.7 s. The same code on 128 processes of Cyberia supercomputer provided solution in 16.3 s. The execution time of the sequential program on these computing systems was 644.5 s (Thor) and 1915.9 s (Cyberia).

CUDA-parallelized program ran on the NVidia RTX2080Ti GPU provided solution in 16.6 s. OpenACC-parallelized program on the same GPU required 20.4 s. Programs parallelized with MPI and CUDA technologies speed up computations on the shared-memory system by 18 and 38 respectively.

The fastest computation on Cyberia supercomputer took 16.3 s and the speedup of the program was the largest (117) with the efficiency of the program on the mesh chosen equal to 90 %.

Two-dimensional latitude-longitude decomposition approach with the method chosen is a basis of an effective and well-scalable MPI-parallelization of the solution of TSUNM3 model equations. Experimental computations made on Cyberia supercomputer of TSU showed that the parallel algorithm developed fulfills the requirement for operational computations for forecasting the local weather conditions.

Acknowledgments
The work was supported by the Russian Science Foundation (project no. 19-71-20042).

References
[1] Powers J G, Klemp J B, Skamarock W C, Davis C A, Dudhia J, Gill D O, Coen J L, Gochis D J, Ahmadov R, Peckham S E, Grell G A, Michalakes J, Trahan S, Benjamin S G, Alexander C R, Dimego G J, Wang W, Schwartz C S, Romine G S, Liu Z, Snyder C, Chen F, Barlage M J, Yu W and Duda M G 2017 The Weather Research and Forecasting Model: Overview, System Efforts and Future Directions Bull. Amer. Meteor. Soc. 98 1717–1737
[2] Ridwan R, Kistijantoro A I, Kudsy M and Gunawan D 2015 Performance evaluation of hybrid parallel computing for WRF model with CUDA and OpenMP 3rd International Conference on Information and Communication Technology (ICOICT) (Nusa Dua) pp 425-430
[3] Starchenko A V, Bart A A, Bogoslovskiy N N, Danilkin E A and Terenteva M A 2014 Mathematical modelling of atmospheric processes above an industrial centre Proceedings of SPIE 9292 88H.
[4] Tolstykh M A, Fadeev R Yu, Shashkin V V, Goyman G S, Zaripov R B, Kiktev D B, Makhnorylova S V, Mizyak V G and Rogutov V S 2018 Multiscale Global Atmosphere Model SL-AV: the Results of Medium-range Weather Forecasts Russ. Meteorol.Hydrol. 43 773–779
[5] Starchenko A V, Kuzhevskaya I V, Kizhner L I, Barashkova N K, Volkova M A and Bart A A 2019 Evaluation of the TSUNM3 high-resolution mesoscale NWP model *Atmospheric and oceanic optics* **32**(01) 57-61

[6] Penenko V V and Aloyan A E 1985 *Models and methods for environmental protection tasks* (Moscow: Science) p 256.

[7] Kunz R and Moussiopoulos N 1995 Simulation of the wind field in Athens using refined boundary conditions *Atmospheric Environment* **29**(24) 3575-3591

[8] Carpenter K 1982 Note on the Paper « Radiation Condition for the Lateral Boundaries of Limited-Area Numerical Models» by Miller M. and Thorpe A. (Vol. 107. pp. 615–628) *Journal of Royal Meteorology Society* **108** 717-719

[9] Nieuwstadt F T and Van Dop H 1984 *Atmospheric Turbulence and Air Pollution Modelling* (Dordrecht, Holland: Springer) p 358

[10] Avissar R and Mahrer Y 1988 Mapping Frost-Sensitive Areas with a Three-Dimensional Local-Scale Numerical Model. Part I. Physical and Numerical Aspects *Journal of Applied Meteorology* **27** 400-413

[11] Stephens G 1978 Radiation Profiles in Extended Water Clouds. Part II: Parameterization Schemes *Journal of the Atmospheric Sciences* **35** 2123–2132

[12] Hong S.-Y and Lim J.-O J 2006 The WRF single-moment 6-class microphysics scheme (WSM6) *J. of Korean meteorological society* **42**(2) 129-151

[13] Belikov D A and Starchenko A V 2007 Numerical model of turbulent pollutant transport in the atmospheric boundary layer *Atmospheric and oceanic optics* **20**(8) 607-612

[14] Van Leer B 1974 Towards the ultimate conservative difference scheme: II. Monotonicity and conservation combined in a second order scheme *Journal of Computational Physics* **14** 361

[15] Xu M, Bao J, Warner T T and Stensrud D J 2001 Effect of Time Step Size in MM5 Simulations of a Mesoscale Convective System *Mon. Wea. Rev.* **129** 502-516

[16] Marchuk G I 1986 *Mathematical models in Environmental Problems* (Moscow: Elsevier Science) p 220

[17] Danilkin E A and Starchenko A V 2010 High Performance Computation for Large Eddy Simulation *LNCS series* **6083** 163

[18] Quinn M J 2004 *Parallel programming in C with MPI and OpenMP* (Singapore: McGraw-Hill Education) p 530