The Algorithm for Transferring a Large Number of Radionuclide Particles in a Parallel Model of Ocean Hydrodynamics

Vladimir Bibin[4,5(✉)], Rashit Ibrayev[1,2,3], Maxim Kaurkin[1,2,3]

[1]Institute of Numerical Mathematics RAS, Moscow, Russia,
ibrayev@mail.ru, kaurkinmn@gmail.com
[2] P.P. Shirshov Institute of Oceanology RAS, Moscow, Russia,
[3]Hydrometeorological Centre of Russia, Moscow, Russia,
[4]Bauman Moscow State Technical University, Moscow, Russia,
devbibinva@gmail.com
[5]Nuclear Safety Institute RAS, Moscow, Russia,

Abstract. The aim of the research is concerned with the description of algorithm of transferring a large, up to 10^6, number of radionuclide particles in a general circulation model of the ocean, INMIO. Much attention is paid to the functioning of the algorithm in conditions of the original model parallelism. The order of the information storage necessary in the course of model calculations is given in this paper. The important aspects of saving calculated results to external media are revealed. The algorithm of radionuclide particles decay is described. The results of the experiment obtained by calculation of the original model based on the configuration of the Laptev Sea are presented.

Keywords: Lagrangian model • Parallel computing • Particles transfer • Radioactive decay.

1 Introduction

In solutions of gas- and hydrodynamics problems, the two following approaches are traditionally used – the Lagrangian and the Eulerian methods. These methods are effective for various classes of problems and both of them possess a wide scope of application[1, 2].

The Lagrangian method got an impetus to development after its application in conjunction with dynamical system models which allowed obtaining fundamentally new results. With this approach, in order to solve nonlinear differential equations in general case, the phase space is introduced that coincides with the physical space of the particles being transported. From the utilization of the Lagrangian method, there emerge a number of additional opportunities for a detailed study of dynamic structures, for example, oceanic eddies.

Our interest in the problem of transport modeling with utilization of the Lagrangian method arises from the studies of radionuclide transfer in marine environment from potential sources located on the sea bottom. A number of studies are dedicated to the solution of this problem [3, 4]. The utilization of the Lagrangian method in solving the problem of radionuclide transfer possesses a number of evident advantages over the utilization of the Eulerian method, particularly in the description of the contamination consisting of various radionuclides, each having a different half-life, with the possibility of further formation of new radionuclides, with the possibility of the radionuclide transfer to contiguous environments (ice, atmosphere), with individual buoyancy characteristics, etc. The model of the Lagrangian transport (LT) can be divided into two classes: online models that synchronize with the hydrodynamic models and offline models in which the particles are transported with pre-calculated fields of flow. The following models should be mentioned as offline: TRACMASS [5, 6], Ariane [7], CMS [8]. In the study [9], an attempt of development of online Lagrangian transport model and the ocean circulation model has been made, however, this model is a prototype and currently works as an offline model. Also it should be noted that code of the model is not parallel. Latest review on the problem of Lagrangian analysis of ocean currents [10] mentions that the studies on particle trajectories calculation, in which instantaneous velocities and diffusion tensor are used, have not come to the authors' notice.

The aim of this research is to create an algorithm for solving the LT problem by three-dimensional ocean currents, which has the following features. First, the transport model should work in synchronism with the parallel model of ocean dynamics. Secondly, the number of particles being transported is sufficiently large, up to 1 million. Thirdly, the particles can have individual properties, for example, lifetime, buoyancy, etc.

The INMIO model [11] is used as the oceanic component of the online model. An essential feature of the INMIO software implementation is that it operates under the control of the computational modelling platform CMF2.0 [12], which is the development of the high-level driver idea. Originally CMF2.0 was developed to create online models of the Earth system and was limited by the coupler function that provides interprocessor exchanges in a completely parallel mode, multilevel interpolation of data and asynchronous work with the file system [12]. Later it turned out that the utilization of a computational modelling platform allows to effectively solve the problems of data assimilation [13], nesting [14], etc. While creating an online LT model and the INMIO ocean dynamics model, the tasks of information input/output, as well as interprocessor exchanges, were solved with the use of CMF2.0.

The plan of the article looks as follows. In Section 2, the algorithm for interprocessor exchange for Lagrangian particles and synchronization with the INMIO model are covered. Issues of information storage and output to external storage media are discussed in Sections 3 and 4. An example of the program work is given in Section 5. Section 6 describes the algorithm for radioactive decay. In Conclusion part we give conclusions and prospects for further development.

2 Interprocessor Exchanges in the Oceanic Component

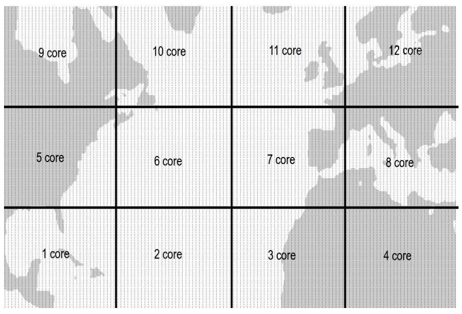
As already mentioned, the LT model is being developed to work in conjunction with the INMIO parallel model of ocean hydrodynamics. Therefore, the numerical scheme, the implementation of parallel computations, the input-output algorithms in the LT model must be coordinated with the INMIO model.

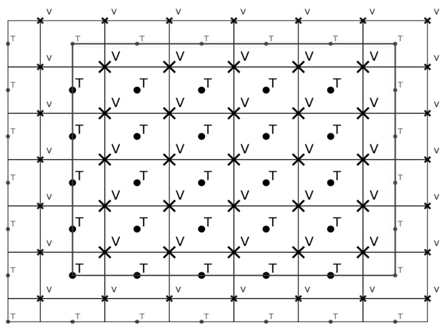
Offline LT models use Runge-Kutta schemes (4th and higher orders), since the discreteness of the data about flows is likely to be quite large, and to ensure the accuracy of the particle's trajectory reproduction, a high order of approximation is required. In the case when the online LT and flow dynamics model is used, the requirement for a high order of the time scheme becomes not rigorous, and one can use the Euler method (1st order scheme) without significant loss of accuracy. Nevertheless, the results of numerical experiments on the LT in a circular flow field show that, regardless of the scheme (Euler, Runge-Kutta), the time integration step $\Delta t^{\text{lagrange}}$ must be noticeably less than Δt^{ocean} in order to achieve an acceptable accuracy of reproduction of the circular particle's trajectory (not published , see also [10]). Therefore, we assume that

$$\Delta t^{\text{lagrange}} = \Delta t^{\text{ocean}} / p \tag{1}$$

Where p is integer and according to the results of test calculations $p \sim (10)$.

In the INMIO model of hydrodynamics, two-dimensional decomposition of the simulated space is accepted (Figure 1a), which provides parallel calculations on multiprocessor computers with distributed memory. The INMIO model, at each step in time, exchanges data of cells on the lateral faces of the calculated subregions, including the corner cells. Parallel system efficiency, including input-output algorithms and interpolation between different grids, was investigated in [15]. The LT model fill arrays for exchange with data about particles that left the processor's subregion and transfers these arrays to neighboring processors.



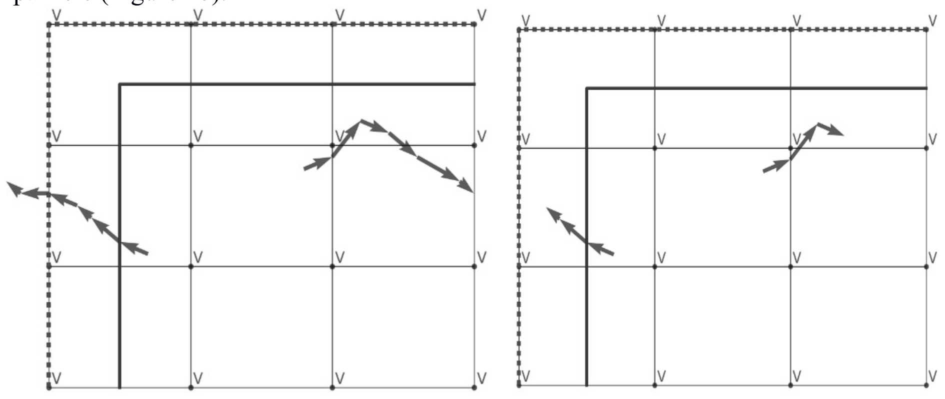


(a) (b)

Fig. 1. Two-dimensional domain decomposition and grid in the INMIO model (a), core subregion in the INMIO model (b). The large marks and font denote the nodes that belong to the considered subregion, the small ones are the nodes belonging to the neighboring subregions (b). The information from the neighboring sub-areas is transmitted to the considered sub-area during interprocessor exchanges.

Let us consider the particle's trajectory on the grid of the ocean dynamics model, Figure 2. Let the particle at the initial moment be in an oceanic cell $V_{\text{i,j,k}}$. In p steps in time, i.e. in Δt^{ocean} , the particle will move for a distance no greater than the size of the cell, because in the ocean model, the Courant-Friedrichs-

Lewy stability condition is fulfilled. If the particle is located close to the border-cell V, then it can cross the dashed line (Figure 2a). Note that beyond the dashed line, within the considered subregion, the velocities in the four nodes surrounding the particle are not determined. Accordingly, at the last few steps of the particle transfer, velocity at the particle location point cannot be found using the regular interpolation algorithm for the inner cells of the subregion. However, in $p/2$ steps in time, i.e. in $\Delta t^{ocean}/2$, the maximum distance that the particles can overcome is the half the cell $V_{i,j,k}$ and, correspondingly, the velocity at the particle location point can be found using the regular algorithm of velocity interpolation for the inner cells of the subdomain in the four nodes surrounding the particle (Figure 2b).



(a) (b)

Fig. 2. Trajectory of a particle for different integration intervals: (a) within the time Δt^{ocean} the particle passes a distance not exceeding the size of one cell; (b) within the time $\Delta t^{ocean}/2$ the particle passes a distance not exceeding the ½ size of the cell.

Thus, the following algorithm was implemented: the interprocessor exchanges are performed in time steps equal to $\Delta t^{ocean}/2$ and Δt^{ocean}. In these time intervals, the particles can overcome a distance of not more than ½ cells and, accordingly, cannot go beyond the borders of the coordinate grid, information about which is stored in the processor that is calculating the considered subregion. The array for sending to the area of the neighboring processor is formed from the particles that are outside the rectangular region, the boundary of which is marked by a solid line (Figure 1b, Figure 2).

3 Order of Information Storage

Since the developed model of Lagrange transfer should work together with the model of ocean dynamics, it is necessary to solve the problem dealt with the memory required for storing information about particles. There are two possible options: it is either to reserve memory for global arrays of Lagrangian particles on each calculated core of computational models of the ocean or to create local arrays about Lagrangian particles for each subarea. Obviously, this or that choice is determined by the number of particles. Practice of computations with the INMIO model on computers of parallel architecture with distributed memory shows that the optimum loading of computing cores is achieved when the domain is decomposed into local areas of 2000-4000 nodes in the horizontal plane. In this case, there is enough free memory on the compute node to accommodate the global arrays for 10^6 particles.

So, there is a vector $f\left(N^{\text{attribute}},\text{N}^{\text{lagrange}}\right)$ with the properties of particles where $N^{\text{attribute}}$ is the property of a particle , N^{lagrange} is the total number of particles, each particle $n^{\text{particle}} \in \left[1, N^{\text{lagrange}}\right]$

The vector of properties associated with the particle allows one to store all the necessary information in the course of calculations. It includes the following characteristics:

• 3 coordinates characterizing the current position of the particle in space;

• 3 coordinates of the cells in which the particle is currently located;

• particle state flag which in the case of finding a particle on a particular core takes the value 1 or 0.

On each of the cores three two-dimensional arrays are also allocated for each component of the coordinate for storing the particles' trajectory: $g\left(N^{\text{lagrange}},\text{N}^{\text{saved\_elements}}\right)$ where N^{lagrange} is the total number of particles, each particle $n^{\text{particle}} \in \left[1, N^{\text{lagrange}}\right]$ and $N^{\text{saved\_elements}}$ is the number of trajectory points to be stored.
After writing to the file considered arrays are reset and reused, which provides significant saving of RAM during the calculations.

4 Checkpoints and Output of Trajectories to External Storage Media

One of the main issues is the problem of output of the information received during calculations to external storage media. As it has already been mentioned, in the ocean model, the work with the file system is provided by means of the CMF2.0 coupler [12], the main tasks of which are to synchronize the interaction between various components of the model and to solve auxiliary problems, one of which is the output of the model to external storage media. In case when the I / O task associated with LT is performed by one of the oceanic cores, there is a substantial imbalance of the computational loads in the model in general: while the ocean master-core is in the process of writing to a file, other cores cannot start their work on the next oceanic step. Thus, the delegation of the input-output task to the core of the coupler is a necessity. Delegating the I / O task to the coupler provides an explicit division of responsibilities between the cores within the model: the oceanic cores directly calculate the physics of the processes of hydrodynamics, the cores of the coupler are in the role of the manager of calculations in the model and solve the auxiliary problems in obtaining the results.

The software code written in the coupler solves the problem of receiving information from the cores of other components and its output to a netCDF format file. However, this system has a number of limitations that make it advisable to write an additional module that solves the problem of writing data about the particle trajectories to a file. One of the restrictions is the presence of arrays that are registered in the coupler system and have dimensions corresponding to the size of the coordinate grid and cannot provide a unified reception of data on particles because their number for each experiment is variable.

To solve this problem, a software module Particle I / O was developed as a part of the CMF2.0 platform, the functionality of which allows one to save the trajectory of particles to a file with a user-defined frequency. By means of Particle I / O the information is collected from the cores that execute the ocean model code. The work of this module and the LT model are synchronized, which allows to exclude the possibility of deadlock and undefined behavior during the model calculations. Each ocean core sends data about the particles that are in the subdomain of the grid of this computational node at the time of saving to a file. Thus, an array containing information about the trajectory of all active particles is formed for subsequent recording to an external storage media. The Particle I/O module is an auxiliary tool in the work of CMF2.0 and can be deactivated for conducting experiments that do not require solving the LT problem.

5 Example of the Work

The computations were carried out on MVS-10P at Joint Supercomputer Center of the Russian Academy of Sciences (JSCC RAS). The current peak performance of this system, which includes 208 computing nodes based on RSC Tornado architecture with direct liquid cooling, Intel server boards, Intel Xeon E5-2690 processors and Intel Xeon Phi coprocessors, is 523.8 tflops.

To test the work of the LT module, an experiment was conducted using the Laptev Sea flow model. The INMIO model belongs to the class 3DPEM [11]. It solves the equations of three-dimensional thermohydrodynamics of the ocean by the finite volume method in Boussinesq approximations and hydrostatics on a grid of type B in vertical z-coordinates. The dimensions of the calculated area are 160x80x49, the time step is 2 minutes. Here it is more important for us to make sure that the LT model works correctly rather than to go into details of the problem statement.

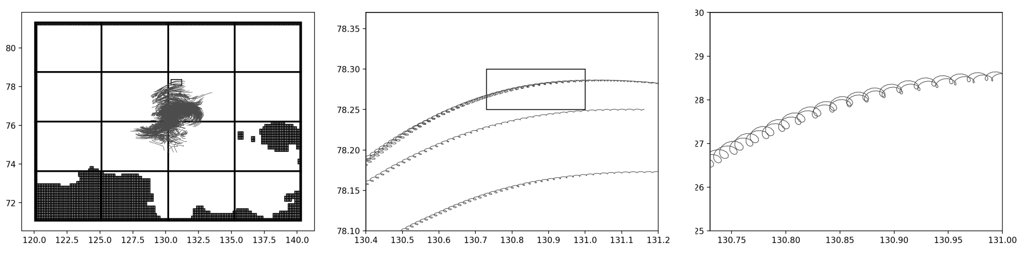
To set the horizontal components $\mathrm{x,y}$ of the initial coordinates , a random number generator is used which produces a set of numbers X where $\forall r \in \mathrm{X}, \mathrm{r} \in [0,1]$. Thus, the formulas, by which the horizontal components of the initial coordinates of an arbitrary particle are calculated, look as follows:

$$\mathrm{x} = \mathrm{x}_c + (r_1 - 0.5) * r_2 * R \tag{2}$$

$$\mathrm{y} = \mathrm{y}_c + (r_3 - 0.5) * r_4 * R \tag{3}$$

Where $r_1, \mathrm{r}_2, \mathrm{r}_3, \mathrm{r}_4 \in X$ and x_c, y_c, R are constants that set region of grid where particles are located.

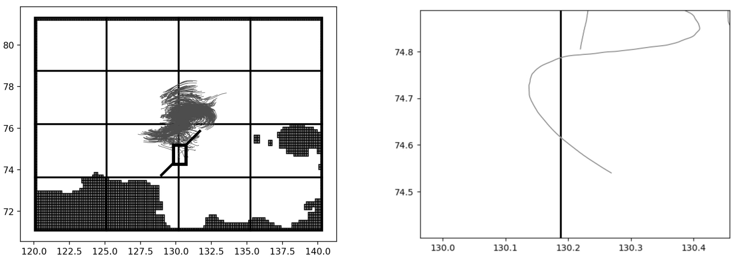
The vertical component z of the initial coordinates is initialized so that the particle is on the surface of the sea. At these initial coordinates, the calculation is made for a period of two model months, with the integration step Δt^{ocean} equal to 2 minutes and $\Delta t^{lagrange}$ equal to 1 minute, i.e. every minute the next coordinates of the particles are calculated. The current coordinates of the particles are stored to the file at a frequency of 1 hour of the model time.



(a) (b) (c)

Fig. 3 The trajectory of particles in the Laptev Sea current model. The considered region is zoomed; With rectangles in Figures a) and b) the areas are highlighted that are shown in Figures b) and c) respectively.

Figure 4 shows the trajectory of a particle crossing the subregion boundary. The trajectory does not undergo discontinuities and continues the expected dynamics of transferring from one subregion to another, which indicates the correctness of the interprocessor exchange of particles.



(a) (b)

Fig. 4. The trajectory of particles in the Laptev Sea current model. Interprocessor exchange. Figure b) that is highlighted in Figure a) with rectangle shows the trajectory of a particle that twice crossed the boundary between neighboring subdomains.

To test the model for successful parallelization of calculations, a series of experiments was performed for 10^6 particles uniformly distributed over the surface of the simulated space. The LT model program code was run on a different number of cores (Figure 5). As a result, the expected, close to linear, dependence of the execution time of one model step on the number of cores on which the experiment was run was obtained.

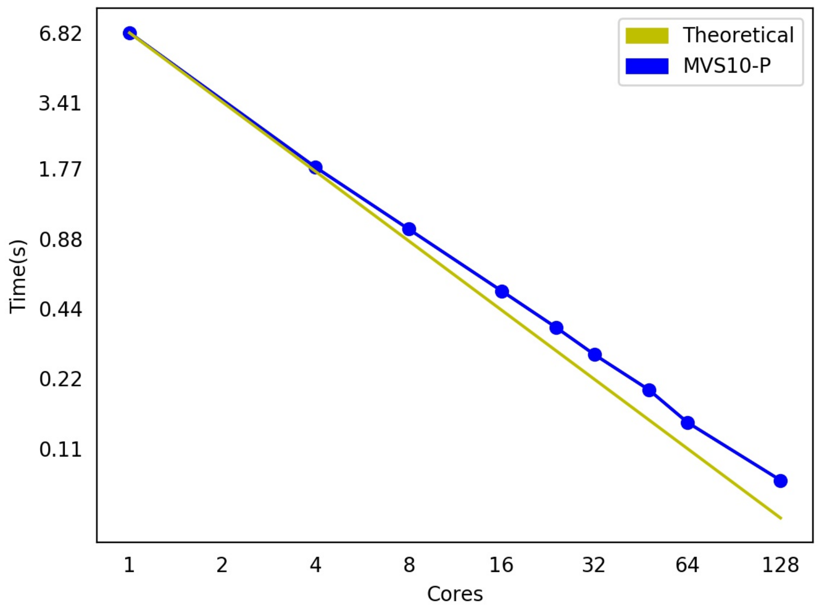


Fig. 5. Execution time of one LT model step in seconds depending on the cores number on the MVS10-P supercomputer. The X axis - number of cores used by the LT model, and on the Y - time, spending on the calculations performed by this model.

Thus, in this experiment, the time for calculating the Lagrangian transfer on the model day takes 60 seconds on 128 cores, if a single processor core were used, then its simulation would take about 4900 seconds, which is not acceptable.

The time for modelling general dynamics of the ocean is 22% of the total time, for 10^6 radionuclides transfer - 78% of the total time. These numbers are due to the fact that the simulated space is a shelf, and the number of calculated horizons was not so great, and consequently the number of nodes, in which the thermohydrodynamics equations were solving, was also significantly smaller than the number of particles.

6 Radioactive Decay and its Parallel Implementation

The description of radionuclides transfer requires the definition of specific particles' properties, such as the isotope name, its half-life, mass and etc. As a consequence, the LT model must classify each particle and associate it with a set of specific properties. As noted earlier, each core contains arrays that store the data about the current position, the accumulated trajectory and the state flag of each particle. To describe the characteristics of radionuclides, the following arrays are additionally introduced: $\mathrm{ParticleType}\left(N_p\right)$ where N_p is the dimension of the array, equal to the number of particles in the original $\mathrm{ParticleTypeDescription}\left(N_c\right)$, where N_c is the dimension of the array, equal to the number of particle types in the original model.

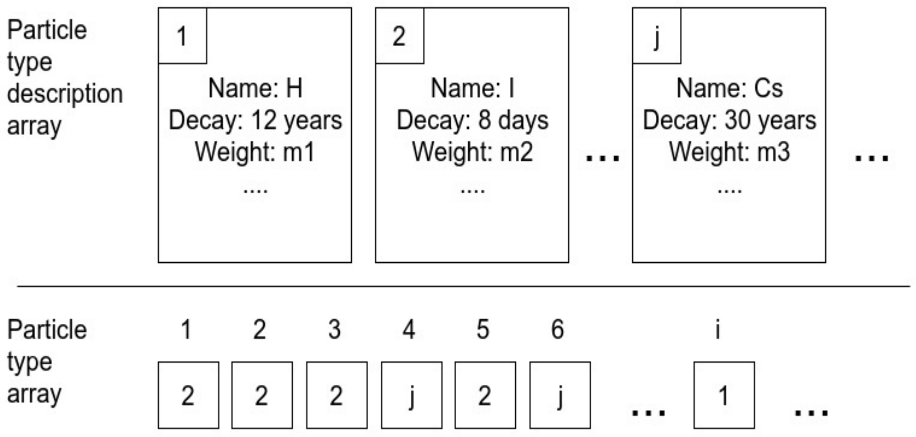


Fig. 6 Classification of particles.

The array $\mathrm{ParticleType}$ contains the indices of the array $\mathrm{ParticleTypeDescription}$ (Figure 6), which is essentially an example of relational database model [16], where the value in the array $\mathrm{ParticleType}$ is the key to the information stored in the $\mathrm{ParticleTypeDescription}$. Thus, each particle is associated with a set of defined specific properties and access to the set of these characteristics is guaranteed in the process of program execution.

The description of radioactive decay is an important part of the radionuclide transfer analysis, since each radionuclide particle can cease to exist at any time. The probability of this event is determined by the half-life. The algorithm describing radioactive decay and its implementation in the form of a program code must satisfy the following conditions:

- radioactive decay is described by fundamental laws;
- possible conversion of a single particle in a number of others;
- correct operation of the algorithm in parallel computing.

To solve this problem, the following algorithm was developed:

1. The cycle starts by all particles that are present in the model.
2. If the n-th particle is processed by the current processor, then go to the steps (3-7), otherwise, the particle is absent in this processor subdomain and does not need to be processed.
3. The particle's lifetime is increased by a model time step.

4. The moment of checking for the n-th particle's decay occurs when the following condition is fulfilled: $L(n) \bmod C(n) < dt$ where $L(n)$ is the current lifetime of the particle, $C(n)$ is the period of checking for decay, dt is the time step in the original model. The check period for decay is specified by the user.

5. If the moment for checking has come, i.e. the condition from block 4 has been fulfilled, a random number x from 0 to 1 is generated which has a uniform distribution over this interval.
6. If $x < 1 - 2^{(-C(n)/T(n))}$ where $T(n)$ is the half-life of the n-th particle, then it is considered that the particle has decayed and ceased to exist.
7. If the particle has decayed, certain procedures are performed to ensure the elimination of the n-th particle from the computational process.
8. End of the cycle for all particles in the model.

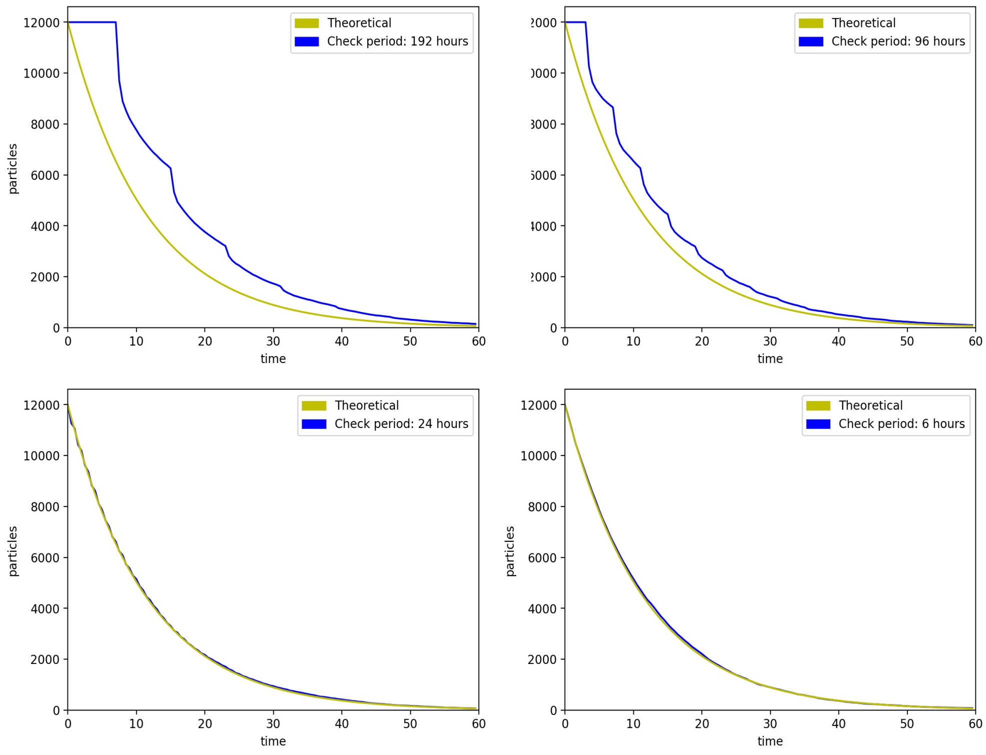


Fig. 7. Carrying out the experiment for different checking periods for decay for element I with a half-life of 8 days. Yellow curve shows the theoretical decay, blue - the result obtained in the course of the algorithm.

The described algorithm was tested in a numerical experiment on modelling the transport of Lagrangian particles for two months in the dynamics model of the Laptev Sea. Figure 7 shows the results of the work of the model, illustrating the process of radioactive decay in time.

The considered algorithm ensures correctness of the decay description regardless of the particle distribution over the processor subdomains of the coordinate grid. Thus, it is guaranteed that if the experiment is carried out on N processors, at the same time X_i is the number of particles with a half-life period T in the region of the i processor at the time step $t=0$, then at the time step $t=T$, in the model will remain $\frac{\sum_{i=1}^{N} X_i}{2}+\epsilon$ particles where ϵ is some error that sets a deflection from the theoretical value which is caused by the use of a random number generator.

Theoretical decay (Figure 7) was described by the formula: $N=N_0 e^{(-\lambda t)}$, where N_0 is the number of atoms in the initial time $t=0$.
The described algorithm gives the expected results and repeats the form of a theoretical curve (Figure 7). When the check period for decay decreases, the curve describing the dependence of the particles number on time tends to theoretical values, which also indicates the correctness of the algorithm.

7 Conclusion

This paper presents an algorithm for transferring a large, up to 10^6, number of particles of radionuclides. The algorithm makes it possible to obtain the trajectory of particles together with online LT model INMIO. The developed algorithm for interprocessor exchange which ensures the transfer of data on particles that left the subregion of one processor and moved to the subdomain of another, guarantees the possibility of moving particles across the entire grid of the ocean model.

The online model of LT and of ocean hydrodynamics minimizes the number of calls to external memory in comparison with similar offline models. Delegating the task of I/ O information to the LT model cores with the CMF2.0 coupler [12] eliminates the possibility of imbalance related to the operations of I/O during the calculations. The efficiency of the described algorithm is confirmed experimentally.

The work was carried out at the Nuclear Safety Institute of The Russian Academy of Sciences with the financial support of the project RNF No. 17-19-01674

References

1. Zhang, Z., Chen Q. Comparison of the Eulerian and Lagrangian methods for predicting particle transport in enclosed spaces // Atmospheric Environment, 41(25), 5236-5248. (2007)

2. Durst F., Milojevic D. Eulerian and Lagrangian predictions of particulate two-phase flows: a numerical study // Appl. Math. Modelling, Vol. 8, 101-115. (1984)
3. Bilashenko V.P., Vysotskii V.L., Kalantarov V.E., Kobrinskii M.N., Sarkisov A.A., Sotnikov V.A., Shvedov P.A., Ibraev R.A., Sarkisyan A.S. Prediction and Evaluation of the Radioecological Consequences of a Hypothetical Accident on the Sunken Nuclear Submarine B-159 in the Barents Sea Antipov // Atomic Energy. 2015. V. 119. № 2. P. 132-141.
4. Heldal H.E., F. Vikebo, and G.O. Johansen Dispersal of the Radionuclide Caesium-137 from Point Sources in the Barents and Norwegian Seas and Its Potential Contamination of the Arctic Marine Food Chain: Coupling Numerical Ocean Models with Geographical Fish Distribution. Environmental Pollution, 2013, vol. 180, 190–198 pp.
5. Döös, K., Kjellsson, J., and Jonsson, B. F. TRACMASS-A Lagrangian Trajectory Model, in Preventive Methods for Coastal Protection, Springer International Publishing, Heidelberg, p. 225–249, 2013.
6. Döös, K., Jönsson, B., and Kjellsson, J. Evaluation of oceanic and atmospheric trajectory schemes in the TRACMASS trajectory model v6.0, Geosci. Model Dev., 10, 1733–1749, https://doi.org/10.5194/gmd-10-1733-2017, 2017.
7. Blanke, B. and Raynaud, S. Kinematics of the Pacific Equatorial Undercurrent: An Eulerian and Lagrangian approach from GCM results, J. Phys. Oceanogr., 27, 1038–1053, 1997.
8. Paris, C. B., Helgers, J., van Sebille, E., and Srinivasan, A. Connectivity Modeling System: A probabilistic modeling tool for the mLange M., Sebille E., 2017ultiscale tracking of biotic and abiotic variability in the ocean, Environ. Modell. Softw., 42, 47–54, 2013
9. Lange M., Sebille E. Parcels v0.9: prototyping a Lagrangian ocean analysis framework for the petascale age, Geoscientific Model Development, 2017, 4175–4186 pp.
10. van Sebille E., S.M. Griffies, R. Abernathey, et al (total 35 authors) Lagrangian ocean analysis: Fundamentals and practices, Ocean Modelling, 2018, Vol. 121, 49–75 pp.
11. Ibrayev, R.A., Khabeev, R.N., Ushakov, K.V.: Eddy-resolving 1/10° Model of the World Ocean. Izvestiya. Atmospheric and Oceanic Physics 48(1), 37-46 (2012)
12. Kalmykov, V.V., Ibrayev, R.A.: A framework for the ocean-ice-atmosphere-land coupled modeling on massively-parallel architectures. Numer. Methods Program. 14(2), 88–95 (2013) (in Russian).
13. Kaurkin, M.N., Ibrayev, R.A., Belyaev, K.P.: Data assimilation ARGO data into the ocean dynamics model with high spatial resolution using Ensemble Optimal Interpolation (EnOI). Oceanology 56(6), 774–781 (2016)
14. Koromyslov A., Ibrayev R., Kaurkin M. The Technology of Nesting a Regional Ocean Model into a Global One Using a Computational Platform for Massively Parallel Computers CMF. In: V. Voevodin and S. Sobolev (Eds.): RuSCDays 2017, CCIS 793, pp. 241–250, 2017
15. Kalmykov, V. V., Ibrayev, R. A., Kaurkin, M. N., and Ushakov, K. V.: Compact Modeling Framework v3.0 for high-resolution global ocean-ice-atmosphere models, Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-294
16. Date, C. J., Introduction to Database Systems, 2003