

Разработка эффективного высокопроизводительного вычисления для решения уравнения Пуассона

Аннотация. В данной статье рассматривается разработка эффективного высокопроизводительного вычисления для решения уравнения Пуассона. В работе был предложен подход параллельного программирования для численного решения двумерного уравнения Пуассона с использованием явной разностной схемы для одного компьютера. Дифференциальное уравнение в частных производных было заменено его конечно-разностным аналогом. В качестве технологии для распараллеливания был выбран на программно-аппаратной архитектуре (CUDA). Численные эксперименты проводятся для проверки и сопоставления эффективности предложенного подхода распараллеливания. Результаты сравниваются с результатами, полученными на основе моделирования с использованием центральных процессоров (CPU) и на графическом процессоре (GPU), который на основании сравнительного анализа показал эффективность параллельного численного алгоритма. Производительность алгоритмов была проиллюстрирована на архитектурах NVIDIA GTX 1050 и с последовательной реализацией Core(TM) i5-7500H.

Ключевые слова: CPU, GPU, уравнение Пуассона, CUDA, параллельная вычисления.

DOI doi.org/10.32523/2616-7263-2022-139-2-143-151

Введение

Одной из основных задач научных вычислений является разработка алгоритмов и вычислительных структур, оптимально использующих доступное оборудование. При применении гибридных суперкомпьютеров сложность в процессе выполнения кода еще больше усложняется, когда центральный процессор (CPU) интегрируется со вспомогательными ускоряющими вычислительными устройствами (Russkov и др., 2021, [1]).

Многоядерные алгоритмы на основе процессора остаются проблемой до тех пор, пока они не станут достаточно быстрыми в течение определенного периода времени. Многие крупные научные задачи требуют вычислительных затрат, а для этого необходимо использовать различные методы параллельной обработки. Для параллельной обработки используются модели Compute Unified Architecture (CUDA) вычислений на графических процессорах NVIDIA. GPU изначально были процессорами, которые ускоряли обработки графики. Универсальный вычислительный графический процессор широко используется в качестве платформы с параллелизмом для выполнения программ (Jodra и др., 2017, [2]). Для научных вычислений графические процессоры (GPU) стали важным ресурсом сокращения времени вычислений. Благодаря архитектуре CUDA [6], предлагаемой NVIDIA, GPU используется в качестве инструмента для обеспечения высокой производительности и системы параллельного программирования (Jiaquan и др., 2021, [3]). Зачастую в науке и технологиях для высокопроизводительных вычислений в качестве инструментов используются: квазитрехдиагональная система линейных уравнений (Li и др., 2016, [4]), однофазный поток (Dominguez и др., 2014, [5]), термический состав с химическими реакциями (Imankulov и др., 2018, [7]), проблема с добычей нефти (Imankulov и др., 2021, [8]), акустические

волны (Altybay и др., 2020, [9]), крупномасштабное моделирование в реальном времени клеточной модели Поттса (Tapia and D'Souza, 2009, [10]), поиск кратчайших уникальных подстрок с несоответствием (Schultz и др., 2021, [11]), первичная разведка залежей углеводородов.

При моделировании задач используется уравнение Пуассона: в уравнении Навье-Стокса несжимаемой жидкости для определения давления получают уравнение Пуассона; зажигание дуги переменного тока на холодных электродах в аргоне атмосферного давления (Santos и др., 2021, [12]); начальный слой и квазинейтральный предел трехмерных моделей электродиффузии (Wang and Jiang, 2021, [13]); обратные задачи оценки момента в размерности 2 (Leblond и Pozzi, 2021, [14]); физико-совместимые методы дискретизации (Palha и др., 2014, [15]); моделирование ветра; моделирование пространственного заряда.

Целью данной работы является параллельная реализация конечного разностного метода для решения двумерного уравнения Пуассона с использованием технология GPU-CUDA. В данной статье предложен подход к параллельному программированию для персональных компьютеров.

Статья структурирована следующим образом: в Разделе 2 представлен обзор теории и численной реализации двумерного уравнения Пуассона. Детали применяемых механизмов распараллеливания описаны в Разделе 3. Результаты, полученные с помощью технологий GPU-CUDA, представлены в Разделе 4, а в разделе 5 приводятся обобщающие выводы.

1. Постановка задачи

Численное решение уравнения Пуассона рассматривается в некоторой прямоугольной области:

$$\nabla^2 U = \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = -f(x, y), \quad (x, y) \in [0; l] \times [0; l], \quad (1.1)$$

граничные условия Дирихле:

$$u(x, 0) = 0, u(x, l) = 0, \quad x \in [0; l], \quad (1.2)$$

$$u(0, y) = 1, u(l, y) = 1, \quad y \in [0; l], \quad (1.3)$$

Теперь вводится прямоугольная сетка с шагами h_1 и h_2 по переменным x и y соответственно:

$$\Omega_{h_1, h_2} = \{x_i = ih_1, i = \overline{0, N}; y_j = jh_2, j = \overline{0, M}\}, \quad (1.4)$$

где $h_1 = l/(N-1)$, $h_2 = l/(M-1)$.

Методом конечных разностей аппроксимируем выражения (2.1)-(2.3). По каждому направлению для простоты положим $N := N = M$ и обозначим $h := h_1 = h_2$.

$$\frac{u_{i+1,j}^k - 2u_{i,j}^{k+1} + u_{i-1,j}^k}{h^2} + \frac{u_{i,j+1}^k - 2u_{i,j}^{k+1} + u_{i,j-1}^k}{h^2} = f_{i,j}^k. \quad (1.5)$$

с граничными условиями

$$u_{0,j}^k = 1, u_{N,j}^k = 1, u_{i,0}^k = 0, u_{i,N}^k = 0, \quad (1.6)$$

для $(i, j) \in \overline{0, N} \times \overline{0, N}$.

2. Модель параллельных вычислений

Сегодня невозможно представить высокопроизводительные вычисления без параллелизации, так как самые мощные вычислительные системы имеют сотни и тысячи процессоров, работающих одновременно в тесном взаимодействии, то есть параллельно. При параллельном вычислении программа делится на множество подпрограмм, поэтому все они выполняются параллельно для вычисления требуемых значений.

Реализация GPU-CUDA

Подход CUDA предоставляет небольшой набор расширений для стандартных языков программирования, таких как C, которые обеспечивают прямую реализацию параллельных алгоритмов. Когда приложения используют CPU и GPU, CUDA также поддерживает гетерогенные вычисления. Программа CUDA вызывает параллельные функции. Параллельные функции также называются ядрами, которые выполняются почти во всех параллельных потоках. Разработчик либо компилятор соединяет эти потоки в блоки потоков и сетки блоков потоков. Применяемый для сброса регистров, вызовов функций и автоматических переменных массивов C в модификации параллельного программирования, CUDA имеет отдельное пространство памяти. В данной работе разработанный подход CUDA был основан на методе Якоби. В алгоритме 1 показан алгоритм решения задачи (1.1)-(1.3).

Алгоритм 1: Реализация двумерного уравнения с использованием CPU

```
compute initial function matrix  $u_{prev}$ 
from initial condition (2.2) we get  $u \leftarrow u_{prev}$ 
do
call function P2D( $u_{prev}$ ,  $u$ ,  $f$ ,  $N$ ,  $h$ )
calculate matrix  $u$ 
swap ( $u_{prev}$ ,  $u$ )
while ( $k < nIterations$ )
```

Для начала необходимо выполнение расчетов размера блока в строгом соответствии с размером матрицы и номерами шагов прямого и обратного решения. В алгоритме 2 показан алгоритм реализации двумерного уравнения с использованием GPU.

Алгоритм 2: Реализация двумерного уравнения с использованием GPU

```
dim3 dimBlock(BLOCK_SIZE_X, BLOCK_SIZE_Y);
dim3 dimGrid(N / dimBlock.x + ((N % dimBlock.x) ? 1 : 0), N / dimBlock.y + ((N % dimBlock.y) ? 1 : 0));
do {
P2D <<<dimGrid, dimBlock >>> (<math>d_{u_{prev}}</math>, <math>d_u</math>, <math>dev_f</math>, N, h);
float* pingPong <math>\leftarrow d_{t_{prev}}</math>;
<math>d_{t_{prev}} \leftarrow d_t</math>;
<math>d_t \leftarrow pingPong</math>;
k++;
} while (k < nIterations);
```

Таким образом, рассчитываем массив d_u и расчетные данные d_u с устройства на хост копируем с помощью `cudaMemcpy(d_u, u, sizeof(float) * N * N, cudaMemcpyHostToDevice);`

3. Численные результаты

В данном разделе продемонстрированы результаты, полученные на настольном компьютере с конфигурацией 640 ядра GeForce GTX 1050, NVIDIA GPU и с CPU Intel(R) Core(TM) i5-7500, 3.40GHz, RAM 16Gb. Значения параметров моделирования даны в виде: $\Delta x = \Delta y = 1/N$, а относительная погрешность 10^{-6} . В результате использования явного метода (1.5) была получена сеточная функция $u_{i,j}^k$. Для получения более реалистичных данных были протестированы семь случаев с разной площадью рассматриваемой области: 32×32 , 64×64 , 128×128 , 256×256 , 512×512 , 1024×1024 , 2048×2048 . В таблице 1 указывается время выполнения в секундах для последовательного подхода (в ЦП) и параллельного подхода CUDA (в графическом процессоре) для задач (1.5) и (1.6).

На рисунке 1 показан графический результат, график построен с помощью программы Tесplot. Результаты программ приведены в таблице 1 и при увеличении размера сетки вычисление на графическом процессоре производится быстрее чем на центральном процессоре Это видно на рисунке 2 и 3.

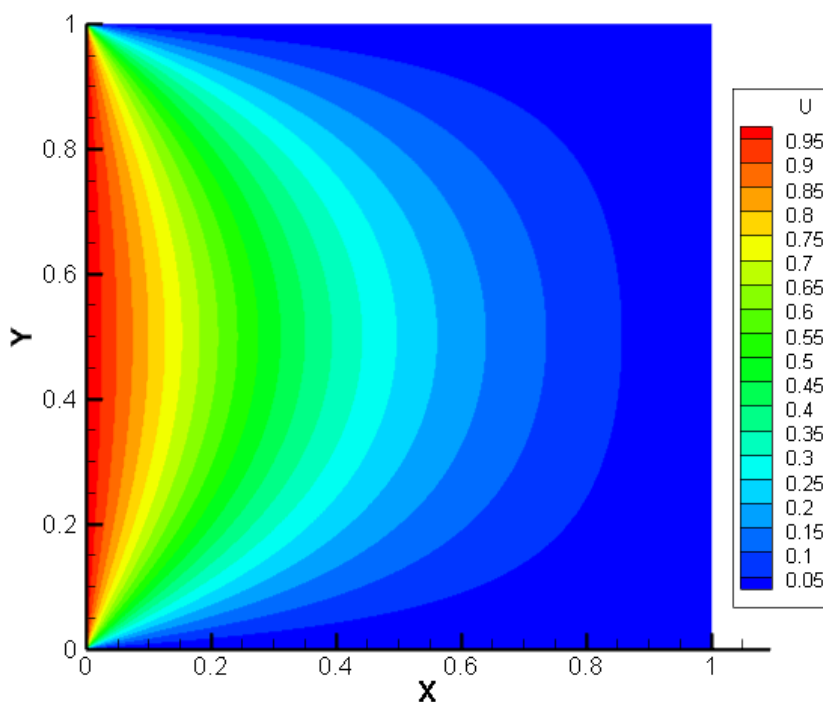


Рисунок 1. График результата с помощью программы Tесplot

Таблица 1. Execution time (seconds)

Mesh size	CPU	GPU	Speed-up
32x32	0.035	0.2160385	0,162008
64x64	0.497	1.0914834	0,455344
128x128	6.871	7.5760875	0,906933
256x256	91.917	72.6604385	1,265021
512x512	1292.31	842.4293213	1,534028
1024x1024	16771.3	10528.823242	1,592894
2048x2048	202957	125857.4296875	1,612595

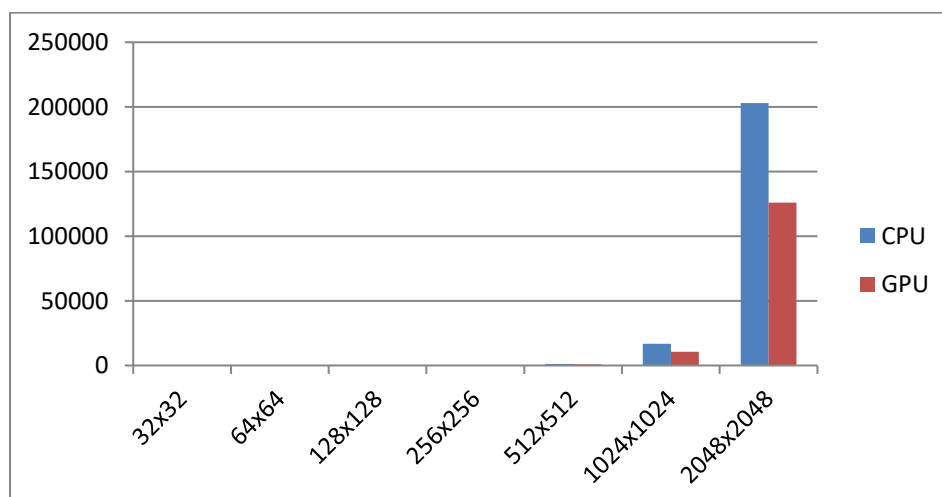


Рисунок 2. Время выполнения программ

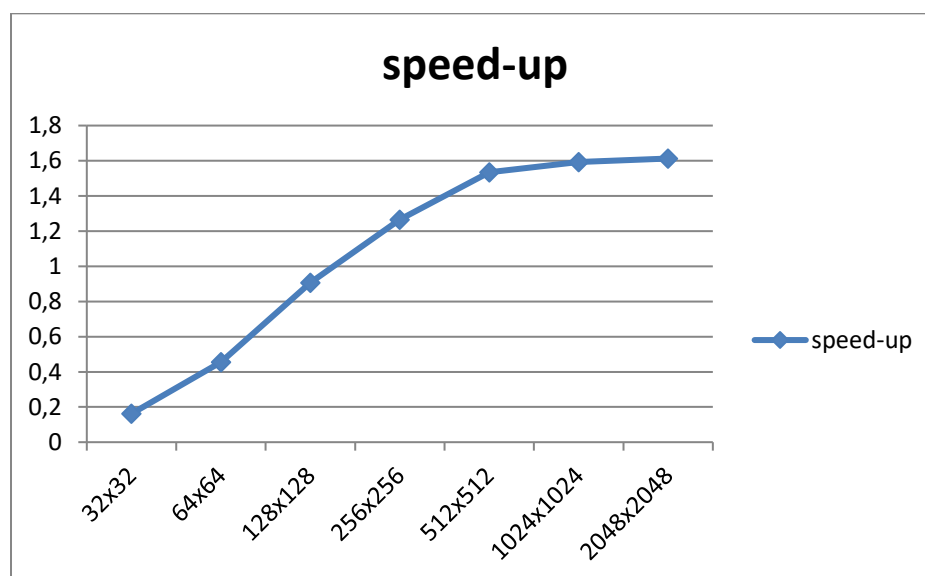


Рисунок 3. Ускорение программ

Замечено, что для большего количества данных и количества пространственных узлов в пространстве решений достигаются более высокие значения ускорения. Кроме того, следует отметить, что из-за процесса распараллеливания как на CPU, так и на GPU потери точности не произошло.

Заключение и будущая работа

Таким образом, в настоящем исследовании было продемонстрировано численное решение двумерного уравнения Пуассона эллиптического типа в прямоугольной области на основе явной конечно-разностной схемы. Было выполнено распараллеливание краевой задачи Дирихле для уравнения эллиптического типа. Все численные расчеты показали высокую производительность в сравнении с экспериментальными данными. В будущем планируется усовершенствовать данную работу и применить реализованный подход для GPU.

Список литературы

1. [Russkov, A](#), [Chulkevich, R](#), [Shchur, LN](#), Algorithm for replica redistribution in an implementation of the population annealing method on a hybrid supercomputer architecture. *COMPUTER PHYSICS COMMUNICATIONS*, vol.261, APR 2021.
2. Jodra, J. L., Gurrutxaga, I., Muguerza, J., & Yera, A. (2017). Solving Poisson's equation using FFT in a GPU cluster. *Journal of Parallel and Distributed Computing*, 102, 28–36.
3. [Jiaquan Gao](#), [Qi Chen](#), [Guixia He](#). A thread-adaptive sparse approximate inverse preconditioning algorithm on multi-GPUs. *PARALLEL COMPUTING*. [Volume 101](#), April 2021, 102724.
4. Li, K., Yang, W., & Li, K. (2016). A Hybrid Parallel Solving Algorithm on GPU for Quasi-Tridiagonal System of Linear Equations. *IEEE Transactions on Parallel and Distributed Systems*, 27(10), p. 2795–2808.
5. [Dominguez, D.S.](#), [Orellana, E.V.](#), [Santos, B.P.](#), [Iglesias, S.M.](#), [Comparing parallel technologies based on GPU and CPU in numerically solving single phase flow problems](#). 11th World Congress on Computational Mechanics, WCCM 2014, 5th European Conference on Computational Mechanics, ECCM 2014 and 6th European Conference on Computational Fluid Dynamics, ECFD 2014, c. 5122-5128
6. NVIDIA E. *CUDA C Programming Guide*, v1.0 (2007). URL <https://developer.nvidia.com/content/cuda-10>.
<https://www.sciencedirect.com/science/article/pii/S0167819120301083?via%3Dihub#b17>
6. NVIDIA E. *CUDA C Programming Guide*, v1.0 (2007). URL <https://developer.nvidia.com/content/cuda-10>.
<https://www.sciencedirect.com/science/article/pii/S0167819120301083?via%3Dihub#b17>
7. T.S. Imankulov, D.Zh. Akhmed-Zaki, B.S. Daribayev, D.V. Lebedev, K.A. Aidarov, Y.S. Nurakhov, O.N. Turar. Intellectual System for Analyzing Thermal Compositional Modeling with Chemical Reactions. ECMOR 2018 - 16th European Conference on the Mathematics of Oil Recovery, Barcelona, Spain. 3-6 September, – 2018.
8. T. Imankulov, B. Daribayev, S. Mukhambetzhonov. Comparative analysis of parallel algorithms for solving oil recovery problem using CUDA and OpenCL. *International Journal of Nonlinear Analysis and Applications*. - 2021. - 12(1). - p 351–364
9. [Altybay, A.](#), [Ruzhansky, M.](#), [Tokmagambetov, N.](#) A parallel hybrid implementation of the 2d acoustic wave equation. [International Journal of Nonlinear Sciences and Numerical Simulation](#), 2020, 21(7-8), p. 821–827.
10. Tapia, J. J., & D'Souza, R. (2009). Data-parallel algorithms for large-scale real-time simulation of the cellular potts model on graphics processing units. 2009 IEEE International Conference on Systems, Man and Cybernetics.
11. [Schultz, Daniel W.](#), [Xu, Bojian](#). [Parallel Methods for Finding k-Mismatch Shortest Unique Substrings Using GPU](#). *IEEE-ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS* Volume: 8 Issue: 1 Pages: 386-395 Published: JAN 1 2021.
12. D F N [Santos](#), M [Lisnyak](#), NA [Almeida](#), L G [Benilova](#), M S [Benilov](#), [Numerical investigation of AC arc ignition on cold electrodes in atmospheric-pressure argon](#), *Journal of Physics D: Applied*

[Physics, Volume 54, Number 19, Published 25 February 2021.](#)

12. D F N Santos, M Lisnyak, NA Almeida, L G Benilova, M S Benilov, [Numerical investigation of AC arc ignition on cold electrodes in atmospheric-pressure argon, Journal of Physics D: Applied Physics, Volume 54, Number 19, Published 25 February 2021.](#)

13. Shu Wang, Limin Jiang Quasi-neutral limit and the initial layer problem of the electro-diffusion model arising in electro-hydrodynamics. [Nonlinear Analysis: Real World Applications vol. 59](#), June 2021, 103266

14. Leblond, J., Pozzi, E., Solutions to inverse moment estimation problems in dimension 2, using best constrained approximation. [Journal Of Approximation Theory, Volume 264](#), April 2021, 105520105520

15. Palha, A., Rebelo, P. P., Hiemstra, R., Kreeft, J., & Gerritsma, M. (2014). Physics-compatible discretization techniques on single and dual grids, with application to the Poisson equation of volume forms. [Journal of Computational Physics](#), 257, 1394–1422.

М.Ж. Сақыпбекова

Әл-Фараби атындағы Қазақ ұлттық университеті, Алматы, Қазақстан

Пуассон теңдеуін шешу үшін тиімді жоғары өнімді есептеулерді жасау

Аңдатпа. Мақалада Пуассон теңдеуін шешу үшін тиімді жоғары өнімді есептеулерді әзірлеу қарастырылады. Екі өлшемді Пуассон теңдеуінің сандық шешімін бір компьютер үшін айқын айырмашылық сұлбасын қолдану арқылы параллельді бағдарламалау тәсілі ұсынылды. Дербес дифференциалдық теңдеу өзінің соңғы айырмашылығымен ауыстырылды. Параллельдеу технологиясы ретінде бағдарламалық-аппараттық архитектура (CUDA) таңдалды. Ұсынылған параллелизация тәсілінің тиімділігін тексеру және салыстыру үшін сандық эксперименттер жүргізіледі. Нәтижелер орталық процессорлар (CPU) және графикалық өңдеу блогы (GPU) арқылы модельдеу негізінде алынған нәтижелермен салыстырылады, олар салыстырмалы талдау негізінде параллельді сандық алгоритмнің тиімділігін көрсетті. Алгоритмдердің өнімділігі NVIDIA GTX 1050 архитектураларында және Core(TM) i5-7500H дәйекті орындалуымен суреттелген.

Кілт сөздер: CPU, GPU, Пуассон теңдеуі, CUDA, параллельді есептеу.

M. Zh. Sakypbekova

Al-Farabi Kazakh National University, Almaty, Kazakhstan

Development of an effective high-performance computation for solving the Poisson equation

Abstract. The article discusses the development of efficient high-performance computing for solving the Poisson equation. The article proposes a parallel programming approach for the numerical solution of the two-dimensional Poisson equation using an explicit difference scheme for one computer. The partial differential equation has been replaced by its finite difference counterpart. As technology for parallelization was chosen from the software-hardware architecture (CUDA). Numerical experiments were carried out to test and compare the effectiveness of the proposed parallelization approach. The results compared with the results obtained based on simulations using central processing units (CPU) and on a graphical processing unit (GPU), which, based on a comparative analysis, showed the effectiveness of a parallel numerical algorithm. The performance of the algorithms was illustrated on the NVIDIA GTX 1050 architectures and with the sequential implementation of the Core (TM) i5-7500H.

Keywords: CPU, GPU, Poisson equation, CUDA, parallel computing.

References

1. [Russkov, A, Chulkevich, R, Shchur, LN](#), Algorithm for replica redistribution in an implementation of the population annealing method on a hybrid supercomputer architecture. *COMPUTER PHYSICS COMMUNICATIONS*, vol.261, APR 2021.
2. Jodra, J. L., Gurrutxaga, I., Muguerza, J., & Yera, A. (2017). Solving Poisson's equation using FFT in a GPU cluster. *Journal of Parallel and Distributed Computing*, 102, 28–36.
3. [Jiaquan Gao, Qi Chen, Guixia He](#). A thread-adaptive sparse approximate inverse preconditioning algorithm on multi-GPUs. *PARALLEL COMPUTING*. [Volume 101](#), April 2021, 102724
4. Li, K., Yang, W., & Li, K. (2016). A Hybrid Parallel Solving Algorithm on GPU for Quasi-Tridiagonal System of Linear Equations. *IEEE Transactions on Parallel and Distributed Systems*, 27(10), p. 2795–2808.
5. [Dominguez, D.S., Orellana, E.V., Santos, B.P., Iglesias, S.M., Comparing parallel technologies based on GPU and CPU in numerically solving single phase flow problems](#). 11th World Congress on Computational Mechanics, WCCM 2014, 5th European Conference on Computational Mechanics, ECCM 2014 and 6th European Conference on Computational Fluid Dynamics, ECFD 2014, c. 5122-5128
6. NVIDIA E. CUDA C Programming Guide, v1.0 (2007). URL <https://developer.nvidia.com/content/cuda-10>.
<https://www.sciencedirect.com/science/article/pii/S0167819120301083?via%3Dihub#b17>
7. T.S. Imankulov, D.Zh. Akhmed-Zaki, B.S. Daribayev, D.V. Lebedev, K.A. Aidarov, Y.S. Nurakhov, O.N. Turar. Intellectual System for Analyzing Thermal Compositional Modeling with Chemical Reactions. ECMOR 2018 - 16th European Conference on the Mathematics of Oil Recovery, Barcelona, Spain. 3-6 September, – 2018.
8. T. Imankulov, B. Daribayev, S. Mukhambetzhonov. Comparative analysis of parallel algorithms for solving oil recovery problem using CUDA and OpenCL. *International Journal of Nonlinear Analysis and Applications*. - 2021. - 12(1). - p 351–364
9. [Altybay, A., Ruzhansky, M., Tokmagambetov, N.](#) A parallel hybrid implementation of the 2d acoustic wave equation. [International Journal of Nonlinear Sciences and Numerical Simulation](#), 2020, 21(7-8), p. 821–827.
10. Tapia, J. J., & D'Souza, R. (2009). Data-parallel algorithms for large-scale real-time simulation of the cellular potts model on graphics processing units. 2009 IEEE International Conference on Systems, Man and Cybernetics.
11. [Schultz, Daniel W., Xu, Bojian. Parallel Methods for Finding k-Mismatch Shortest Unique Substrings Using GPU. IEEE-ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS](#) Volume: 8 Issue: 1 Pages: 386-395 Published: JAN 1 2021.
12. D F N [Santos](#), M [Lisnyak](#), NA [Almeida](#), L G [Benilova](#), M S [Benilov](#), [Numerical investigation of AC arc ignition on cold electrodes in atmospheric-pressure argon](#), [Journal of Physics D: Applied Physics](#), [Volume 54](#), [Number 19](#), [Published 25 February 2021](#).
13. [Shu Wang, Limin Jiang](#) Quasi-neutral limit and the initial layer problem of the electro-diffusion model arising in electro-hydrodynamics. [Nonlinear Analysis: Real World Applications](#) vol. 59, June 2021, 103266
14. Leblond, J., [Pozzi, E.](#), Solutions to inverse moment estimation problems in dimension 2, using best constrained approximation. *Journal Of Approximation Theory*, [Volume 264](#), April 2021, 105520105520
15. Palha, A., Rebelo, P. P., Hiemstra, R., Kreeft, J., & Gerritsma, M. (2014). Physics-compatible discretization techniques on single and dual grids, with application to the Poisson equation of volume forms. *Journal of Computational Physics*, 257, 1394–1422.

Сведения об авторе:

Сақыпбекова М.Ж. – старший преподаватель кафедры искусственного интеллекта и Big Data Казахского национального университета им. аль-Фараби, пр. аль-Фараби, 71, Алматы, Казахстан.

Sakupbekova M.Zh. - Senior Lecturer of the Department of Artificial Intelligence and Big Data of the Kazakh National University. al-Farabi, al-Farabi avenue 71, Almaty, Kazakhstan.