National Research University Higher School of Economics

as a manuscript

Choi Yea Rem

Parallel matrix multiplication and matrix exponent algorithms based on asynchronous data transfers between several graphics accelerators, and their application to solve the non-stationary Schrödinger equation

Dissertation summary
for the purpose of obtaining academic degree
Doctor of Philosophy in Applied Mathematics HSE

Academic supervisor:
Doctor of Physical and Mathematical Sciences
Professor
Vladimir Stegailov

Moscow — 2024

Relevance of the topic

The development of the supercomputers hardware base, in general, is ahead of the parallel algorithms development that can most effectively solve mathematical modeling problems on new types of computing equipment [1]. One of the most actively developing types of high-performance servers are ones with several (4, 6, 8, 16) GPU accelerators connected by fast communication channels. The first server of this type, DGX-1, with NVLink interconnect was released in 2016 by Nvidia; currently, similar servers were started to be produced by AMD (with Infinity Fabric interconnect) and Intel (with Xe Link interconnect). Using such servers with several GPU accelerators as a single computing tool is not a trivial task and requires the development of parallel algorithms with an optimal structure of data exchanges between GPU accelerators. This can be achieved by overlapping data exchanges and calculations (the "overlapping computation and communication" principle). The relevance of the dissertation work lies in the development of similar algorithms for matrix multiplication and for calculating the matrix exponent. In addition, the work demonstrates how, based on the developed computational algorithms, it is possible to use the colossal computing performance of servers with several GPU accelerators for the molecular dynamics mathematical modeling of the simplest molecular ion H_2^+ based on solving the non-stationary Schrödinger equation in a formulation with a minimum number of simplifying assumptions, which opens up the possibility of modeling in the hitherto insufficiently studied area of nonadiabatic molecular dynamics.

Formulation of the problem

The goal of the work is to develop a parallel algorithm for the matrix multiplication and the matrix exponent on several graphics accelerators using high-performance communication links asynchronously to achieve maximum performance, and its application to solve the non-stationary Schrödinger equation.

To achieve the set goal, it was necessary to solve the following **tasks**:

1. Development of an asynchronous parallel program for an efficient matrix multiplication algorithm that performs calculations on several GPUs, and the program performance study.
2. Derivation of a theoretical model to set the optimal tile size, in which the program will be executed with the best performance.

3. Development and research of a program for the matrix exponential algorithm on GPUs based on the use of the developed matrix multiplication algorithm.
4. Solution of the time-dependent Schrödinger equation using the resources of high computing power GPUs and analysis of the wave function behavior in a time-varying potential for a molecular H_2^+ ion.

Degree of the research topic development

A review of the literature shows that the first published matrix multiplication algorithm that implemented the principle of asynchronous data exchanges between GPUs to improve performance was the BLASX algorithm described in the paper by Wang et al [2]. Somewhat later, the PaRSEC [3] software framework appeared, aimed at systems with IBM Power processors that have fast NVLink communication channels between the CPU and GPU, due to which the task of matrix multiplication can be effectively divided between the CPU and GPU. Nvidia provides a closed-source cuBLAS-XT solution for calculating matrix multiplication on multiple GPUs and CPUs simultaneously (cuBLAS-XT is used as a reference in this work, and the proposed matrix multiplication algorithm for multiple GPUs has been shown to provide higher performance). One of the most universal frameworks for matrix multiplication, combining various types of parallel algorithms, is the COSMA project [4]. Methods for dividing calculations in COSMA are based on optimal combinations of matrix dimensions, the number of processors and memory sizes, when choosing which the key characteristic is the optimality of providing a read-write operation through the combinatorial model of the "red-blue game with pebbles". Within the framework of the COSMA project, the Tiled-MM project is being developed, which is the closest analogue of the parallel matrix multiplication algorithm proposed in this dissertation work (however, Tiled-MM was published much later than the first article with the results of this dissertation).

The matrix product algorithm is used in various computational problems, for example, in the algorithm for calculating the matrix exponent. The problem of calculating the matrix exponent is a computationally very complex task, therefore, to solve it, various kinds of simplifications are used that are associated with the special characteristics of the matrices. On the other hand, the computing power of supercomputers is growing quite rapidly these days. As a result, it is no longer something completely impossible to consider the problem of calculating the matrix exponential in the most general case. There are quite a lot of options for methods

for finding the matrix exponent. They can be divided into the following categories: representation as a sum of a series, solution of differential equations, polynomial methods, various types of matrix decompositions (including spectral) or separation methods [5; 6]. As the authors of the cited reviews note, it is difficult to say which of the presented methods is "best". In the case of normal matrices, many problems automatically disappear due to the properties of normality, but in an arbitrary case it is necessary to deal with rounding errors.

Solving the time-dependent Schrödinger equation (TDSE) requires numerous matrix exponential calculations, making the problem difficult to handle in the general case. Solutions to particular TDSE problems can be found in various works. For example, the work of Lugovskoy and Bray considers the almost sudden perturbation of a quantum system by an ultrashort pulse [7]: the analytical theory is compared with the numerical solution of TDSE. Various He ionization scenarios were considered by numerically solving one-dimensional TDSE in the work of Yu and Madsen [8; 9]. Nonadiabatic quantum dynamics of molecular ions H_2^+ and HD^+ excited by single laser pulses linearly polarized along the molecular axis was studied within the framework of a three-dimensional model in the work of Paramonov et al. [10]. The interaction of strong laser fields with He, H_2^+ and H_2 was modeled based on the TDSE solution in Majorosi et al. [11]

Main provisions submitted for defense:

1. The matrix multiplication algorithm has been developed that uses asynchronous data exchanges between several GPUs within one server.
2. The theoretical model has been constructed to determine the optimal tile size for the proposed matrix multiplication algorithm and verified by testing on the servers with different connections between GPUs.
3. The algorithm has been developed for calculating the matrix exponent based on the proposed matrix multiplication algorithm, using several GPUs.
4. Described that the numerical solution of the time-depended 1D Schrödinger equation using the proposed algorithm for calculating the matrix exponential makes it possible to illustrate nonadiabatic transitions in models of bi-nuclear molecules with one electron.

Scientific novety: The parallel algorithm for matrix multiplication for several GPUs has been developed, the efficiency of which is higher than that of analogues included in standard libraries (cuBLASXt). An analytical model of the algorithm performance is proposed. The predictive power of the model is verified by

numerical experiments on various hybrid platforms. Based on the developed matrix multiplication algorithm, the algorithm for calculating the matrix exponent has been developed. Using that a program was created to solve the time-dependent Schrödinger equation and calculations of the molecular hydrogen ion were carried out. The possibility of describing the process of nonadiabatic transition of the electronic subsystem from the ground to the excited state is shown.

Methodology and research methods

Programming tools (Nvidia CUDA SDK, AMD ROCm), optimization methods, profiling tools (Nsight, rocprof), programming-related methods for program composition, and external mathematical libraries were used. The model of the optimal adjustable parameter of the algorithm (tile size) was obtained analytically. Difference methods were applied to construct an algorithm for solving the time-dependent Schrödinger equation.

Main results of the study

Asynchronous multi-GPU algorithm for matrix multiplication and matrix exponent

A general matrix product algorithm program has been developed

$$C = \alpha AB + \beta C, \tag{1}$$

which uses only GPUs for calculations and data storage. The algorithm provides the possibility to store all data of matrices as in the memory of one device, providing the rest of the GPU with data for calculations, as in different GPUs, reducing the load on the memory access process.

The general scheme of the algorithm is as follows:

– devices storing matrices A and B send bands of A_i and B_i to other GPUs,
– GPUs perform matrix multiplication with α using the obtained data, and calculate tiles C'_{ij}, which are then assembled into a band C'_i,
– a device in which the matrix C is stored, collects the stripes C'_i and sums the resulting matrix with βC.

A more detailed diagram of the algorithm's operation is shown in Fig. 1.

The algorithm running on accelerators that perform the calculations has the following form (see. Alg. 1):

A detailed algorithm for working in GPUs is given in Alg. 2.

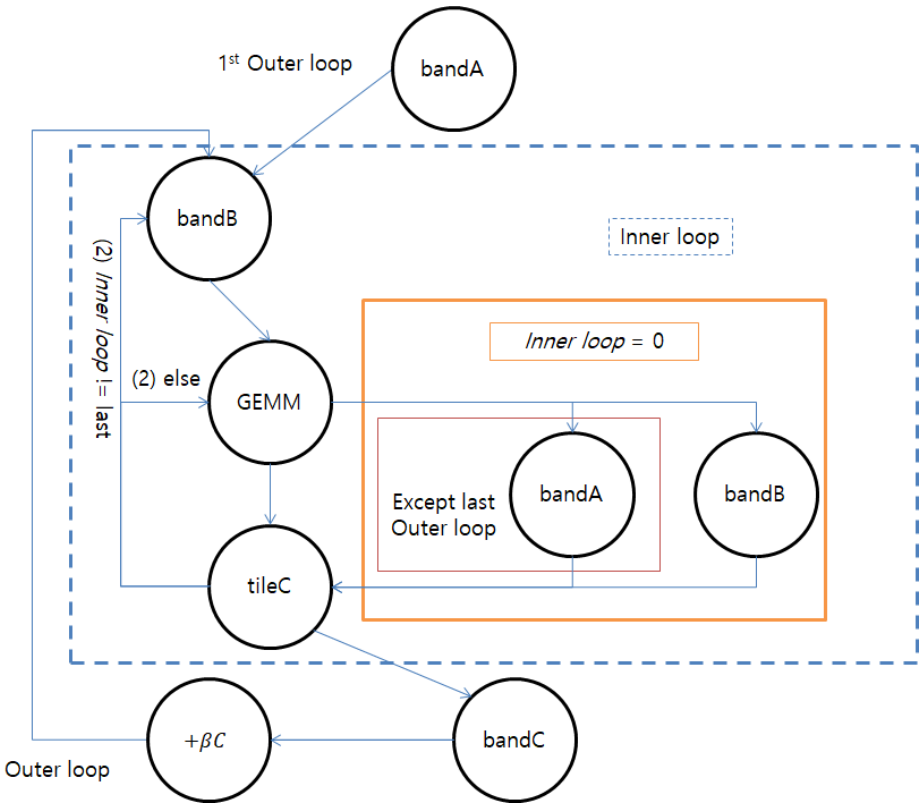

Figure 1 — Graphic diagram of the algorithm. The entire algorithm works within the outer loop. Data transfer commands may be called in a different order if full data usage is verified before overwriting.

Algorithm 1 Schematic flow of the Multi-GPU GEMM algorithm in an arbitrary worker GPU, m is the number of bands in a row (column).

```
for outerloop = 0 to m/NumOfGPUs do
  receive (bandA);
  for innerloop = 0 to m do
    recieve (bandB);
    GEMM (bandA, bandB, alpha, tileC);
    write_TileToBand (tileC, bandC);
  end for
  send (bandC);
end for
```

In general, the exponent of the matrix A is represented as a Taylor series expansion

$$\exp A = \sum_{i=0}^{\infty} a_i = \sum_{i=0}^{\infty} \frac{1}{k!} A^k = I + \frac{1}{1!}A + \frac{1}{2!}A^2 + \frac{1}{3!}A^3 + \ldots, \qquad (2)$$

Algorithm 2 Complete Multi-GPU GEMM algorithm taking into account command changes and data transfer operations for different devices.

```
if (device ≠ device_withA) then
  receive (bandA);
end if
for outerloop = 0 to m/NumOfGPUs do
  for innerloop = 0 to m do
    if (innerloop < m - 1) and (device ≠ device_withB) then
      recieve (bandB);
    end if
    GEMM (bandA, bandB, alpha, tileC);
    if (innerloop = 0) then
      if (device ≠ device_withB) then
        recieve (bandB);
      end if
      if (outerloop < m/NumOfGPUs - 1) and (device ≠ device_withA) then
        Receive (bandA);
      end if
    end if
  end for
  send (bandC);
  if (device == device_withC) then
    AddbetaC (C, beta, bandC);
  end if
end for
```

where I is the identity matrix. Each k-th element of a_k is expressed as

$$a_k = \frac{1}{k} A a_{k-1}, \tag{3}$$

that is, to find the k-th term, the algorithm (1) can be applied with matrices A and a_{k-1} and constants $\alpha = 1/k, \beta = 0$.

In complex space, the multiplication of two complex matrices can be split into four separate matrix multiplications with real numbers

$$\begin{cases} Re(C) = \alpha(Re(A) * Re(B) - Im(A) * Im(B)) + \beta Re(C), \\ Im(C) = \alpha(Re(A) * Im(B) + Im(A) * Re(B)) + \beta Im(C). \end{cases} \tag{4}$$

In the matrix exponential algorithm, $\beta = 0$ (3), therefore, at each iteration it is simply necessary to find the products of real and imaginary parts with the coefficient α and separately perform the operations of adding or subtracting matrices. To maximize memory utilization, whenever possible, each real and imaginary part of the matrices is stored in separate graphics accelerators as separate matrices.

Theoretical model for finding the optimal tile size

The arithmetic intensity has been determined for the case of square matrices, in which the calculation speed will be limited by the computing capacity of the accelerators, and not by the memory bandwidth

$$\begin{cases} BW_{math}/BW_{mem} = k_{BW}, \\ Intensity = \frac{N_i^2 N}{2(N_i N + N_i N + N_i^2)} = \frac{N_i N}{4N + 2N_i} > k_{BW}. \end{cases} \tag{5}$$

where BW is the processing capacity of the processor (math) or memory bandwidth (mem), respectively. From this we obtained the condition for the tile size

$$N_i > 4k_{BW}N/(N - 2k_{BW}),\ T_{math} > T_{mem}. \tag{6}$$

The work considers fairly large matrices that require large computing power. In this case, the time of data transfer between devices is obtained as

$$T_{transfer} = 4N_i N/BW_{transfer}. \tag{7}$$

If the task depends on the computing power of GPUs, for tile sizes

$$N_i > 2(Num_{GPUs} - 1)BW_{math}/BW_{transfer}, \tag{8}$$

the calculation speed should be greater than the data delivery speed, and no waiting should occur. Here Num_{GPUs} is the number of graphics accelerators used in the calculation. The calculation time depends linearly on the tile size, so for optimal calculation the following conditions are imposed:

$$\begin{cases} N_i > 4k_{BW}N/(N - 2k_{BW}), & N > 2k_{BW}, \\ N_i > 2(Num_{GPUs} - 1)BW_{math}/BW_{transfer}, \\ N_i \to \min. \end{cases} \tag{9}$$

The developed optimal tile size model was analyzed and observed with a profiler. The predicted tile sizes were validated for a square matrix multiplication algorithm that distributes tasks evenly across accelerators.

Computational experiments on different hardware environments

Results were collected on platforms with 4 V100s linked by NVLink 2.0, 8 A100 connected by NVLink 3.0, 4 GTX 1070 by PCIe 3.0, and 4 RX 6900 XT by PCIe 4.0.

On platforms with four V100 and four GTX1070, satisfaction of the optimal parameter model were shown.

When working with consumer-level graphics accelerators (AMD RX 6900 XT), performance losses and delays between data transfers were noticed (see Fig. 2).

Using an example with significantly influencing factors, such as uneven balancing of the computing load on each GPU, which may arise depending on the considered parameters of the problem, the possible deviation of the empirically optimal parameters from the predicted ones is demonstrated for a server based on

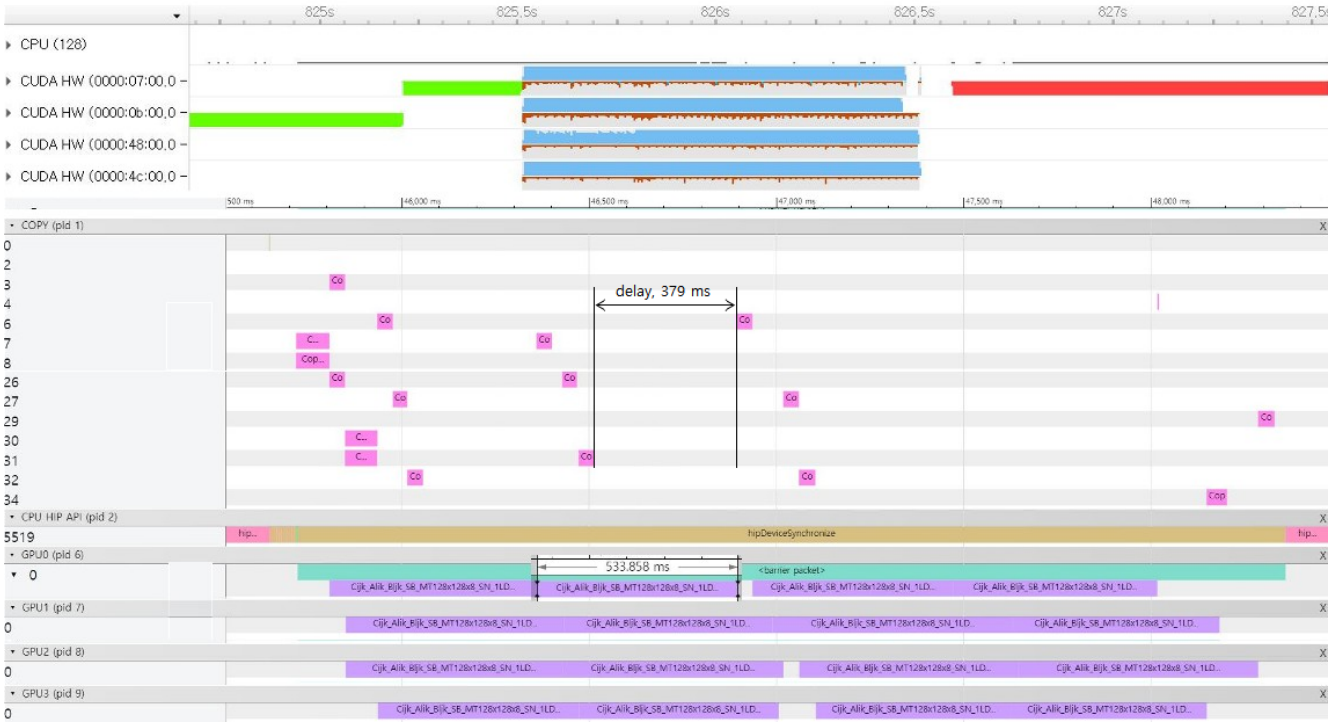

Figure 2 — Multi-GPU GEMM performance profiles on 4 A100 GPUs without tensor cores (above) and 4 RX 6900 XT GPUs (below). The number of elements ($N = 32768$) in a row (column) of matrices and the size of tiles ($N_i = 1024$) in the case of the A100 GPU and ($N_i = 8192$) in the case of the RX 6900 XT GPU. Matrices A, B, and C are stored in devices 2, 1, 0, respectively. Green columns (above, below omitted) are the data transfer from the host to the GPUs for calculations, red columns (above, below omitted) are the resulting data transfer from the GPUs to the host, blue columns (above) and purple columns (below) are calculations in the GPU, and the brown columns (above) and pink columns (below) are data transfer operations between peer-to-peer accelerators.

eight A100 with NVLink. Using the same platform as an example, we show how the use of tensor cores changes the balance between communication and computation (see Fig. 3).

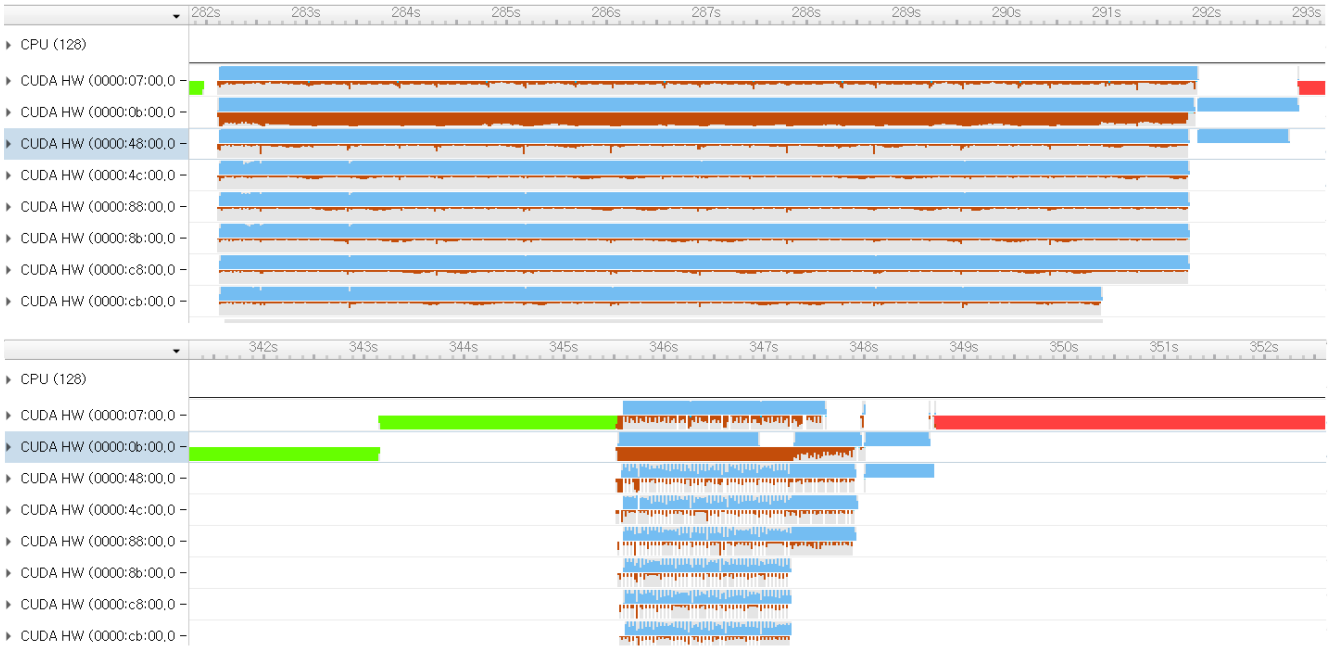

Figure 3 — Profiles of Multi-GPU GEMM work on 8 A100 GPUs without tensor cores (above) and with tensor cores (below). The number of elements ($N = 90000$) in a row (column) of matrices, the size of tiles ($N_i = 1024$) in the case without tensor cores and ($N_i = 4096$) in the case with tensor cores. Matrices A, B, and C are stored in devices 2, 1, 0, respectively. Green columns are the transfer of data from the host to the accelerators for calculations, red columns are the transfer of the resulting data from the accelerators to the host, blue columns are calculations in the GPU and brown columns are data transfer operations between peer-to-peer accelerators.

Using the developed algorithm to solve the 1D time-dependent Schrödinger equation

The algorithm for solving one-dimensional time-dependent Schrödinger equation has been developed as follows (Alg. 3).

The solution has been found for the stationary case of the molecular hydrogen ion model (two protons and one electron) with a soft-core Coulomb potential, see Fig. 4.

$$V(r) = \frac{-Z}{\sqrt{r^2 + a}}. \tag{10}$$

Algorithm 3 Scheme of the algorithm for solving the time-dependent Schrödinger equation executed on GPUs in one time iteration, ψ is the wave function, H is the generated Hamiltonian matrix, $\exp A$ is the resulting matrix exponential. All operations are performed separately for real and imaginary matrices. Communication between graphics accelerators is not shown.

receive (potential V);
build Hamiltonian matrix ($H = T + V$), $A = iH\Delta t$;
$\exp A = I + A$;
for $k = 2$ to rank **do**
  $\alpha = 1/k$;
  Multi-GPU GEMM ($A^k = \alpha A A^{k-1}$);
  $\exp A = \exp A + A^k$;
end for
GEMV ($\psi_j = \exp A\ \psi_{j-1}$);
send (ψ_j); //for output in file

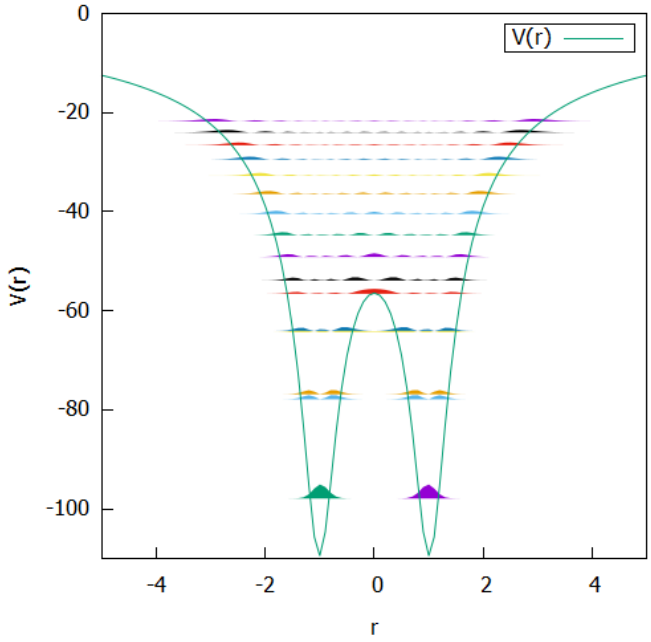

Figure 4 — An example of the electron density distribution $|\psi_n(r)|^2$ for several bound states at the corresponding positions of energy levels in the soft-core Coulomb potential of two ions (10) ($Z = 30, a = 0.1$).

To simulate ion vibration, we implemented the movement of ion centers.

$$V(r,t) = \frac{-Z}{\sqrt{(r - \alpha \sin(\beta t))^2 + a}}, \tag{11}$$

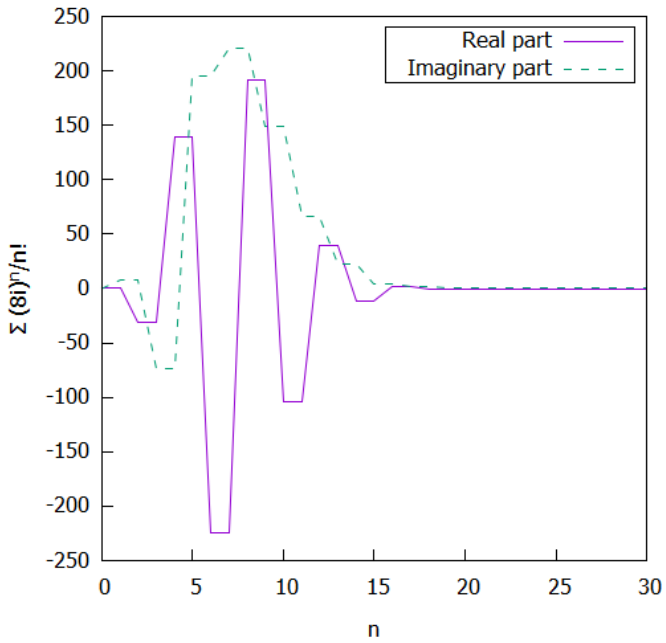

Figure 5 — The behavior of the function $\sum_n \frac{(8i)^n}{n!}$, illustrating the typical convergence of elements of the matrix exponent.

where Z is the strength of the Coulomb interaction of the soft core, a is the softening parameter [8], α and β are some constants.

When calculating the complex matrix exponential we should pay attention to the increasing error effect (Fig. 5). To eliminate this problem, it is necessary to maintain the ratio of time and distance steps $\Delta r \sim \Delta t^2$.

The results of experiments with stationary and sinuously moving ions (potential centers) of the molecular hydrogen ion were obtained. In the considered non-stationary case a non-adiabatic transition of the electronic subsystem from the ground to the excited state was observed (Fig. 6).

Summary

The main results of the work are as follows.

1. The asynchronous Multi-GPU GEMM algorithm has been developed for several GPUs using only devices for computation and communication without accessing the memory of the central processor.
2. The matrix exponential algorithm has been developed for real and complex number versions based on the Multi-GPU GEMM algorithm.
3. The theoretical model is derived that predicts the optimal variable tile size parameter for which the Multi-GPU GEMM algorithm shows the best performance. It has been established that the optimal size depends on platform-dependent parameters.

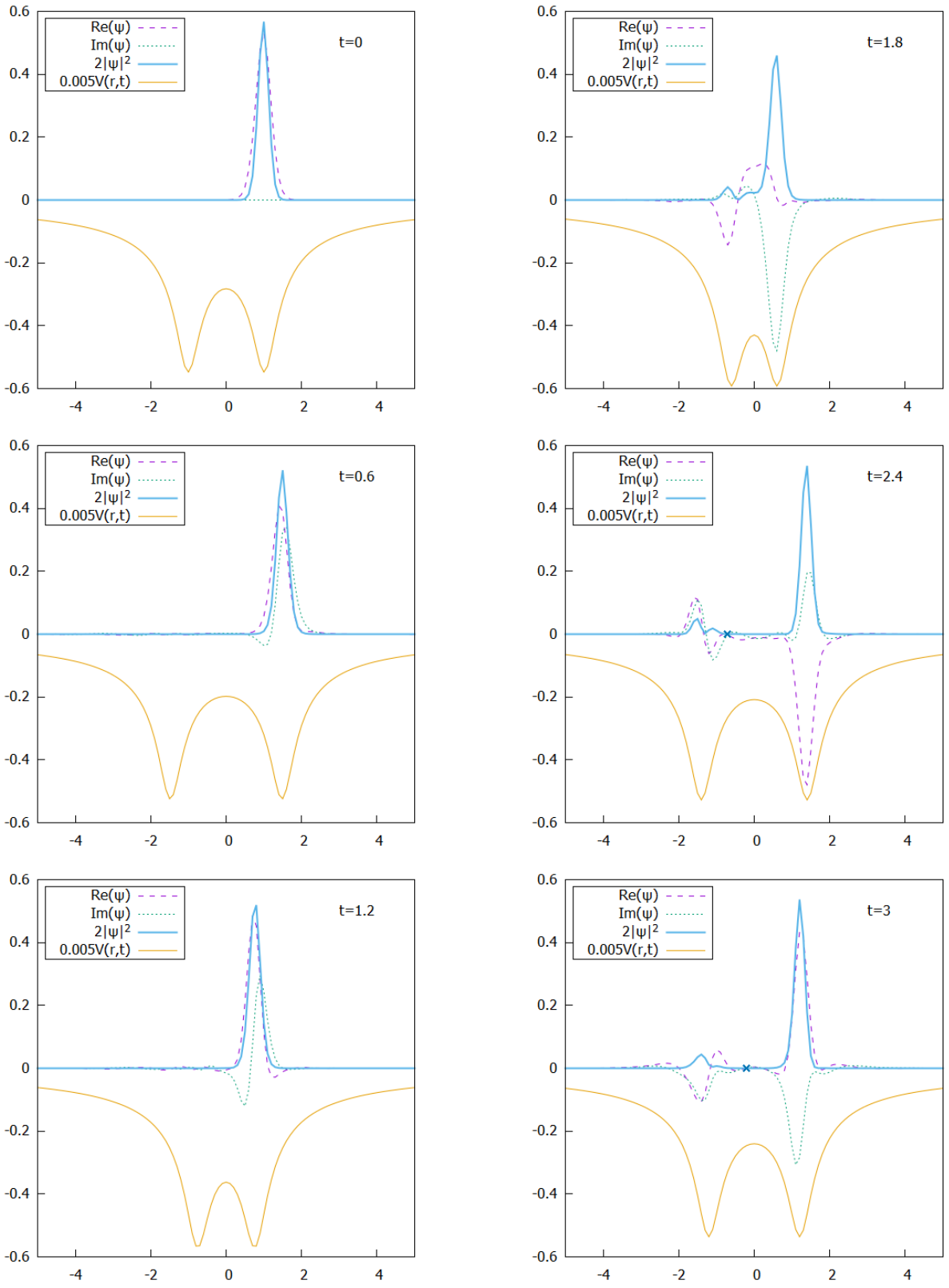

Figure 6 — Behavior of the complex wave function (ψ) in a sinuously moving double-well soft-core Coulomb potential (11) (initial distance between wells is 2, $Z = 30, a = 0.1$, $\Delta t = 0.002$, $\Delta r = 0.1$, $k = 10$, $\alpha = 0.5$ and $\beta = 3$). The dots indicate the nodes of the wave function that arise after excitation from the ground state.

4. The HIP version of the Multi-GPU GEMM algorithm has been implemented to work with AMD GPUs.
5. The Multi-GPU GEMM algorithm has been successfully launched and tested on different computing platforms. High performance has been achieved, over 80% of peak performance for server-level accelerators and approximately 40% for consumer-level accelerators.
6. The predicted values of the varied parameters by the proposed model were verified using empirical results. Factors influencing possible discrepancies for real cases of working with the algorithm are described.
7. A parallel algorithm has been developed on several graphics accelerators for solving the one-dimensional time-dependent Schrödinger equation based on the algorithms described above.
8. Nonadiabatic energy transfer from moving nuclei to single-electron excitation of a hydrogen ion was simulated H_2^+.

Approbation of work.

The results of the work were reported at the following conferences.

1. 2020 Global Smart Industry Conference (GloSIC), Russia, Chelyabinsk, 17-19 November, 2020. «Matrix-Matrix Multiplication Using Multiple GPUs Connected by Nvlink».
2. 20-th Mathematical modeling and supercomputer technologies (MMST2020), within the International Congress Russian Supercomputing Days Lobachevsky State University of Nizhny Novgorod, Russia, Nizhny Novgorod, 23-25 November, 2020. «Алгоритм матричного умножения для нескольких GPU, объединенных высокоскоростными каналами связи».
3. 9-th "Distributed Computing and Grid Technologies in Science and Education"(GRID'2021), Russia, Dubna, 5-9 July, 2021. «Overlapping Computation and Communication in Matrix-Matrix Multiplication Algorithm for Multiple GPUs».
4. Parallel computational technologies (PCT) 2022, Russia, Dubna, 29-31 March, 2022. «The Tuning of Matrix-Matrix Multiplication Algorithm for Several GPUs Connected by Fast Communication Links».
5. «Extreme-scale big data analytics and scientific computing on heterogeneous platforms», Lake Como School of Advanced Studies, Como,

Italy, 26-30 September, 2022. «Multi-GPU GEMM algorithm: maximizing efficiency on different platforms».

6. 22-th Mathematical modeling and supercomputer technologies (MMST 2022), within the International Congress Russian Supercomputing Days Lobachevsky State University of Nizhny Novgorod, Nizhny Novgorod, 14-17 November, 2022. «Multi-GPU GEMM algorithm performance analysis for Nvidia and AMD GPUs connected by NVLink and PCIe».
7. The Conference on Computational Physics (CCP2023) – 34th International Union of Pure and Applied Physics (IUPAP) Conference on Computational Physics, Kobe International Conference Center, Kobe, Japan, 4-8 August, 2023. «Multi-GPU GEMM for 1D Time-Dependent Schrodinger Equation».
8. «Russian Supercomputing Days 2023», Russia, Moscow, 25-26 September, 2023. «GPU-accelerated matrix exponent for solving 1D time-dependent Schrodinger equation».

Personal contribution.

Under the supervision of the supervisor at the overall picture and structure of the work, the applicant compiled all the given algorithms and wrote the corresponding programs. The applicant conducted computational experiments on different platforms, access to which was provided by the supervisor. With the help of the supervisor, the results of numerical experiments were analyzed, and ideas were jointly developed to eliminate vulnerabilities and modify the program for the better, which were then implemented into the code by the applicant. The applicant personally gave all presentations at conferences of various formats. The applicant provided the data presented in the publications, and wrote the main part of the first article, and most, including the main part, of the remaining articles, which were then examined and corrected by the scientific supervisor.

Publications The main results on the topic of the dissertation are presented in 4 printed publications, 4 — in periodical scientific journals indexed by Web of Science and Scopus.

Author's publications on the topic of the dissertation

1. *Choi, Y. R.* Matrix-Matrix Multiplication Using Multiple GPUs Connected by NVLink [Текст] / Y. R. Choi, V. Nikolskiy, V. Stegailov // 2020 Global Smart Industry Conference (GloSIC). — IEEE. 2020. — С. 354—361.
2. *Choi, Y. R.* Multi-GPU GEMM Algorithm Performance Analysis for Nvidia and AMD GPUs Connected by NVLink and PCIe [Текст] / Y. R. Choi, V. Stegailov // Mathematical Modeling and Supercomputer Technologies: 22nd International Conference, MMST 2022, Nizhny Novgorod, Russia, November 14–17, 2022, Revised Selected Papers. — Springer. 2022. — С. 281—292.
3. *Choi, Y. R.* Tuning of a Matrix-Matrix Multiplication Algorithm for Several GPUs Connected by Fast Communication Links [Текст] / Y. R. Choi, V. Nikolskiy, V. Stegailov // International Conference on Parallel Computational Technologies. — Springer. 2022. — С. 158—171.
4. *Choi, Y. R.* GPU-Accelerated Matrix Exponent for Solving 1D Time-Dependent Schrödinger Equation [Текст] / Y. R. Choi, V. Stegailov // Supercomputing / под ред. V. Voevodin [и др.]. — Cham : Springer Nature Switzerland, 2023. — С. 100—113.

References

1. *Schulthess, T. C.* Programming revisited [Текст] / T. C. Schulthess // Nature Physics. — 2015. — Т. 11, № 5. — С. 369—373.
2. BLASX: A high performance level-3 BLAS library for heterogeneous multi-GPU computing [Текст] / L. Wang [и др.] // Proceedings of the 2016 International Conference on Supercomputing. — 2016. — С. 1—11.
3. Generic matrix multiplication for multi-GPU accelerated distributed-memory platforms over PaRSEC [Текст] / T. Herault [и др.] // 2019 IEEE/ACM 10th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems (ScalA). — IEEE. 2019. — С. 33—41.

4. Red-blue pebbling revisited: near optimal parallel matrix-matrix multiplication [Текст] / G. Kwasniewski [и др.] // Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. — 2019. — С. 1—22.

5. *Moler, C.* Nineteen dubious ways to compute the exponential of a matrix [Текст] / C. Moler, C. Van Loan // SIAM review. — 1978. — Т. 20, № 4. — С. 801—836.

6. *Moler, C.* Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later [Текст] / C. Moler, C. Van Loan // SIAM review. — 2003. — Т. 45, № 1. — С. 3—49.

7. *Lugovskoy, A.* Almost sudden perturbation of a quantum system with ultrashort electric pulses [Текст] / A. Lugovskoy, I. Bray // Physical Review A. — 2008. — Т. 77, № 2. — С. 023420.

8. *Yu, C.* Sequential and nonsequential double ionization of helium by intense XUV laser pulses: Revealing ac Stark shifts from joint energy spectra [Текст] / C. Yu, L. B. Madsen // Physical Review A. — 2016. — Т. 94, № 5. — С. 053424.

9. *Yu, C.* Above-threshold ionization of helium in the long-wavelength regime: Examining the single-active-electron approximation and the two-electron strong-field approximation [Текст] / C. Yu, L. B. Madsen // Physical Review A. — 2017. — Т. 95, № 6. — С. 063407.

10. Quantum dynamics, isotope effects, and power spectra of H_2^+ and HD^+ excited to the continuum by strong one-cycle laser pulses: Three-dimensional non-Born-Oppenheimer simulations [Текст] / G. K. Paramonov [и др.] // Physical Review A. — 2018. — Т. 98, № 6. — С. 063431.

11. Density-based one-dimensional model potentials for strong-field simulations in He, H_2^+, and H_2 [Текст] / S. Majorosi [и др.] // Physical Review A. — 2020. — Т. 101, № 2. — С. 023405.