engineering proceedings

MDPI

Proceeding Paper

Evaluation of Computer Technologies for Calculation of Exact Approximations of Statistics Probability Distributions †

Andrey Melnikov [1], Ilya Levin [2], Aleksey Dordopulo [2,*] and Lyubov Slasten [2]

[1] "Computational Solutions" JSC, Varshavskoe Highway 125/17, Moscow 119991, Russia; niimvs@yandex.ru
[2] Supercomputers and Neurocomputers Research Center, Italyansky Lane, 106, Taganrog 347900, Russia; levin@superevm.ru (I.L.); lmslasten@yandex.ru (L.S.)
* Correspondence: dordopulo@superevm.ru; Tel.: +7-8634-612111
† Presented at the 15th International Conference "Intelligent Systems" (INTELS'22), Moscow, Russia, 14–16 December 2022.

Abstract: This paper is devoted to the evaluation of the hardware resources of computer systems for solving a computationally expensive problem such as the calculation of the probability distributions of statistics by the second multiplicity method based on Δ-exact approximations of samples with a size of 320–1280 characters and an alphabet power of 128–256 characters. The accuracy is $\Delta = 10^{-5}$ and the total solution time should not exceed 30 days or 2.592×10^6 seconds for 24/7 computing. Owing to the use of the properties of the second multiplicity method, the computational complexity of the calculations can be brought within the range of 9.68×10^{22}–1.60×10^{52} operations with the number of tested vectors at 6.50×10^{23}–1.39×10^{50}. The solution of this problem for the specified parameters of samples during the given time requires hardware resources which cannot be provided by modern computer technology such as processors, graphics accelerators and programmable logic integrated circuits. Therefore, in this paper, we analyze the possibilities of promising quantum and photon technologies for solving the problem with the given parameters. The main advantage of quantum computer systems is the high speed of calculations for all possible parameter values. However, quantum acceleration will not be achieved to calculate the probability distributions of statistics due to the need to check all the obtained solutions. Here, the number of obtained solutions corresponds to the dimension of the problem. In addition, due to the current development level of quantum hardware components, it is impossible to create and use 120 qubit quantum computers for the solution of the considered problem. Photon computers can provide high computation speeds at low power consumptions and require the smallest number of nodes to solve the considered problem. However, unsolved problems with the physical implementation of efficient memory elements and the lack of available hardware components make the use of photon computer technologies impossible for calculating the probability distributions of statistics in the near future (5–7 years). Therefore, it is most reasonable to use hybrid computer systems containing nodes of different architectures. To solve the problem on various hardware platforms (general purpose processors, GPUs and FPGAs) and configurations of hybrid computer systems, we suggest to use the architecture-independent high-level programming language SET@L. The language combines the representation of calculations as sets and collections (based on the alternative set theory of P. Vopenka), the absolutely parallel form of the problem represented as an information graph and the paradigm of aspect-oriented programming.

Keywords: probability; statistic; exact distribution; exact approximation; algorithmic complexity; quantum calculations; photonic technologies; architecture-independent programming; Set@l language

check for updates

Citation: Melnikov, A.; Levin, I.; Dordopulo, A.; Slasten, L. Evaluation of Computer Technologies for Calculation of Exact Approximations of Statistics Probability Distributions. *Eng. Proc.* **2023**, *33*, 40. https://doi.org/10.3390/engproc2023033040

Academic Editors: Askhat Diveev, Ivan Zelinka, Arutun Avetisyan and Alexander Ilin

Published: 21 June 2023

1. Introduction

In general, criteria that minimize the number of false decisions concerning the truth of tested hypotheses are used to solve application problems that require statistical processing of texts as meaningful character sequences. The criteria based on the exact distributions

of reference statistics [1] have the greatest relative efficiency, but the calculation of exact distributions is a computationally laborious task [1,2], depending on the power of the alphabet N and the sample size n (the length of the text sequence).

We can reduce the computational complexity of the problem if we use exact approximations (limit distributions) instead of exact distributions which minimally reduce the efficiency of the used criteria [1,2]. As exact approximations of distributions of reference statistics, we use Δ-exact distributions [2] which differ from the initial ones by an arbitrarily small Δ. To calculate exact distributions, there are methods such as the calculation method of the exact distributions statistics of the Kolmogorov–Smirnov type [3,4], the well-known classical Monte Carlo method [5], etc. The most preferable is the second multiplicity method [2], which provides calculations of the exact approximations of distributions for the maximum values of the samples' parameters with the same resource. The second multiplicity method, based on solving systems of linear equations, has polynomial complexity, but its computational complexity for real application problems is still quite large [2], so calculation of exact approximations in a reasonable time using modern computational means is difficult. An evaluation of the required computing resources and the problem solution time by means of modern processors, graphics accelerators and FPGAs for the distribution parameters required in practice is given in [2]. In this paper, we consider an evaluation of the required resources and possible problem solution times for calculating exact approximations of probability distributions of statistics by means of promising computer technologies: quantum and photon computer systems.

2. Setting of the Problem

We consider probability distributions of statistics for an alphabet $A_N = \{a_1, \ldots, a_N\}$ with a power $\mid A_N \mid = N$ and its sample n. To calculate the exact approximations of statistics probability distributions $P_\Delta\{S_{N,n} \geq c\}$, which differ from the exact distributions $P_T\{S_{N,n} \geq c\}$ by a specified arbitrary small value Δ

$$\mid P_T\{S_{N,n} \geq c\} - P_\Delta\{S_{N,n} \geq c\} \mid \leq \Delta, \tag{1}$$

we use the second multiplicity method (SMM) based on the solution of a system of linear equations

$$\begin{cases} \mu_0^{(v)} + \mu_1^{(v)} + \ldots + \mu_n^{(v)} = N, \\ 1 \cdot \mu_1^{(v)} + 2 \cdot \mu_2^{(v)} + \ldots + n \cdot \mu_n^{(v)} = n \end{cases} \tag{2}$$

where $\mu_i^{(v)}$ is the number of characters that occurred j times in the sample v of the alphabet A_N and n is the size of the sample v.

The SMM [2] is based on sequential enumeration of the vectors $\mu^{(v)} = \{\mu_0^{(v)}, \mu_1^{(v)}, \ldots, \mu_n^{(v)}\}$ and determination of whether each $\mu^{(v)}$ is a solution of the system of linear Equations (2) or not. A detailed theoretical review of the use and implementation of the SMM is given in [2]. The SMM algorithm's complexity is defined by

$$C_{MVK}(P_T\{S_{N,n} \geq c\}) = L_{\mu(N,n,r)} \times 5 \cdot r + K_{\mu(N,n,r)} \cdot (5 \cdot (r+1) + 2(N+r) + 3) + 2 \cdot K_{\mu(N,n,r)} \cdot \log_2 K_{\mu(N,n,r)} + 2 \cdot K_{\mu(N,n,r)}, \tag{3}$$

where $L_{\mu(N,n,r)}$ is the reduced number of tested vectors of possible solution candidates of (2) with the limitations $\{r \leq n \mid \forall i = \overline{r+1, n},\ \mu_i^{(v)} = 0\}$ and $K_{\mu(N,n,r)}$ is the number of non-negative integer solutions of (2) with the limitations r. According to [2], the main complexity of (3) depends on the first term of the polynomial

$$L_{\mu(N,n,r)} = (N+1)^{min([n/N,r])+1} \cdot \frac{(min([n,N],r))!}{(n + min([n,N],r))!} \cdot \frac{(n+r)!}{r!} \tag{4}$$

It follows from (3) and (4) that the algorithmic complexity of calculating the exact approximations $C_{MVK}(P_\Delta\{S_{N,n} \geq c\})$ is a polynomial which depends on both the parameters

of the sample N and n, and on the limitation parameter r, which also functionally depends on the sample's parameters and the accuracy Δ, i.e., $r = m(N, n, \Delta)$.

The most laborious part of the SMM is the procedure of calculating and testing solution candidate vectors. For practical problems, the power of the alphabet N belongs in the range from 128 to 256, and the sample sizes are in the range from 320 to 1280, when the required total solution time does not exceed 30 days. Therefore, to evaluate the algorithmic complexity [2], we specified the following samples as (the alphabet power, the sample size): (256, 1280), (128, 640), (128, 320) and (192, 320) with the accuracy $\Delta = 10^{-5}$. The algorithmic complexity and the required performance of the task with these sample parameters are given in Table 1.

According to Table 1, the computational complexity is in the range from 9.68×10^{22} to 1.60×10^{52} operations, the average complexity is about 4.55×10^{25} operations, and it is necessary to test from 6.50×10^{23} to 1.39×10^{50} vectors and to obtain from 4.67×10^{12} to 5.60×10^{25} solutions.

The number of tested variables in (2) does not exceed $(r + 1)$, i.e., it does not exceed 24 for the parameters given in Table 1. Accordingly, $\mu^{(v)} = \{\mu_0^{(v)}, \mu_1^{(v)}, \ldots, \mu_n^{(23)}\}$, and all other variables are equal to zero: $\{\mu_j^{(v)} = 0 | j = 24, \ldots, n\}$. The values of $\mu_i^{(v)}$ are integers and belong to the range $\{0 \le \mu_j^{(v)} \le 23 | i = \overline{0, 23}\}$. One $\mu_j^{(v)}$ is no less than 5 bits ($2^4 < 23 < 2^5$), and the whole vector $\mu^{(v)}$, which contains 24 $\mu_i^{(v)}$, is no less than $5 \times 24 = 120$ bit.

The important property of the method is its parallelization by data because any μ^i and μ^j can be tested independently and concurrently when $i \neq j$.

Table 1. Characteristics of the calculation method of exact approximations for various parameters of samples.

№	Parameters of the Sample N/n	Limitation Parameter r	Reduced Number of Tests $L_{\mu(N,n,r)}$	Number of Solutions $K_{\mu(N,n,r)}$	Total Complexity $C_{MVK}(P_T\{S_{N,n} \ge c\})$	Required Performance for Calculation for 30 days (op/s)
1	256/1280	23	1.39×10^{50}	5.60×10^{25}	1.60×10^{52}	6.15×10^{45}
2	128/640	22	2.67×10^{27}	1.76×10^{20}	2.94×10^{29}	1.13×10^{23}
3	192/320	14	6.50×10^{23}	4.67×10^{12}	4.55×10^{25}	1.75×10^{19}
4	128/320	16	1.21×10^{21}	2.23×10^{13}	9.68×10^{22}	3.72×10^{16}

3. Use of General Purpose Processors to Calculate Exact Approximations of Distributions

The ×86 processors with the traditional von Neumann architecture and 32-bit and 64-bit data processing compose the main and most widespread general purpose computing architecture for the design of cluster multiprocessor high-performance systems (MHPS) [6].

For calculating exact approximations, the performance P_{CPU} of a CPU-based MHPS with unlimited scalability is

$$P_{CPU} = N_{CPU} \cdot K_{CPU} \cdot H_{1\_CPU}, \tag{5}$$

where N_{CPU} is the number of processors; K_{CPU} is the number of provided parallel threads; and H_{1_CPU} is the frequency of one processor.

To estimate the number of processors needed to solve the problem with time constraints, let us consider a hypothetical CPU that supports eight parallel threads at 3000 MHz, i.e., $K_{CPU} = 8$ and $H_{1\_CPU} = 3 \times 10^9$. Then, to achieve the minimal performance (according to Table 1) of $P_{CPU} \ge 1.75 \times 10^{19}$ op/s, it is necessary to use

$$N_{CPU} \ge \frac{1.75 \times 10^{19}}{K_{CPU} \cdot H_{1\_CPU}} = \frac{1.75 \times 10^{19}}{8 \times 3.0 \times 10^9} = 7.29 \times 10^8,$$

i.e., about 729 million hypothetical processors. The obtained number exceeds the number of cores (not the number of processors!) of the most high-performance supercomputers in the world (such as Fugaku [6,7] with 7 million cores and Sunway TaihuLight [6,7] with 10 million cores, included in TOP500 [6]) by 1.5–2 decimal orders.

To calculate exact approximations for all values of the samples' parameters $\{N = \overline{2,256},\ n = \overline{1,5N}\}$, the number of required nodes is

$$N_{CPU} \geq \frac{6.15 \times 10^{45}}{K_{CPU} \cdot H_{1\_CPU}} = \frac{6.15 \times 10^{45}}{8 \times 3.0 \times 10^{9}} = 5.56 \times 10^{35},$$

which cannot be achieved if we take into account the modern level of processor architectures and technologies in cluster MHPS designs. The performance of modern CPUs is insufficient for the solution of the considered problem. Furthermore, calculations of exact approximations cannot be provided if a computer system is based only on modern CPUs because hundreds of million CPUs are needed to solve the problem even in its minimal form.

4. Use of Graphics Accelerator Technology to Calculate Exact Approximations of Distributions

The development of GPU (Graphic Processing Unit) technology [7], originally designed to calculate 3D graphics in real time, has led to their application in high-performance computing. Modern graphics accelerators, containing thousands of special purpose cores, provide a high degree of parallelism and can perform tasks in a multithread parallel mode. For example, the standard GEFORCE RTX-3090 game graphics card contains 10,496 NVIDIA CUDA cores operating at 1.70 GHz (1.7×10^9 op/s) [8]. We can roughly define the performance of a GPU-based computer system P_{GPU} as the following product

$$P_{GPU} = N_{GPU} \cdot K_{CP} \cdot H_{CP}, \tag{6}$$

where N_{GPU} is the number of graphics accelerators GPU in the system; K_{CP} is the number of cores of each accelerator; and H_{CP} is the clock frequency.

To provide $P_{GPU} \geq 1.75 \times 10^{19}$, the system based on a GEFORCE RTX-3090 must contain no less than

$$N_{RTX3090} \geq \frac{1.75 \times 10^{19}}{K_{RTX} \times H_{RTX}} = \frac{1.75 \times 10^{19}}{1.05 \times 10^{4} \times 1.7 \times 10^{9}} = 9.80 \times 10^{5}$$

nodes, and the system based on an NVIDIA Quadro K6000 [9] with the performance

$$K_{K6000} \cdot H_{K6000} = 16.3 \times 10^{12} \text{ op/s}$$

of one videocard, must contain no less than

$$N_{K6000} \geq \frac{1.75 \times 10^{19}}{K_{K6000} \cdot H_{K6000}} = \frac{1.75 \times 10^{19}}{1.63 \times 10^{13}} = 1.07 \times 10^{6}$$

nodes. Despite the difference in the properties of the considered graphics accelerators, the number required for calculation of exact approximations is estimated at about one million, which is impossible with the current level of development of graphics accelerator architectures and design technologies of GPU-based computer systems. Thus, due to the insufficient performance of modern graphics accelerators, it is impossible to use them as the only basis to calculate exact approximations.

5. Use of Parallel Pipeline FPGA Technologies to Calculate Exact Approximations of Distributions

The FPGA (Field Programmable Gate Array) technology combines the capabilities of parallel and pipeline computing. In contrast to computer systems with a fixed architecture

designed on a CPU and GPU basis, it allows reconfiguration [10] of the architecture of an FPGA-based computer system to the architecture of the problem to be solved. Taking into account information dependencies in the structure of the application [10], it is possible to provide high real performance for labor-intensive problems in various fields of science and technology [11]. For example, the computational block (CB) of the latest Seguin system based on the UltraScale+ FPGAs (3U height, 96 XCVU9P-1FLGC2104E chips designed using 16 nm technology) achieves the performance of 240 Tflops (2.4×10^{14} op/s) [12].

For the problem of calculating exact approximations, the performance of an FPGA-based reconfigurable multiprocessor computer system P_{FPGA} can be roughly estimated as the product of the number of computational blocks (CB) N_{CB_FPGA} in the system and the performance of one CB P_{CB_FPGA}.

$$N_{CB\_FPGA} \geq \frac{1.75 \times 10^{19}}{P_{CB\_FPGA}} = \frac{1.75 \times 10^{19}}{2.4 \times 10^{14}} = 7.29 \times 10^{4}. \quad (7)$$

To calculate exact approximations for all values of the considered parameters of the samples $\{N = \overline{2,256}, n = \overline{1,5N}\}$, it is necessary to use

$$N_{CB\_FPGA} \geq \frac{6.15 \times 10^{45}}{P_{CB\_FPGA}} = \frac{6.15 \times 10^{45}}{2.4 \times 10^{14}} = 2.56 \times 10^{31}$$

computational blocks. Parallel pipeline FPGA technologies with the real performance of up to 10^{14} operations per second for one computational block have the greatest potential for implementation of the second multiplicity method for the largest values of the sample parameters. However, a computer system, which is based only on FPGA technologies and implements the considered problem, cannot be easily designed because it requires a very large number of computational blocks.

According to the analysis of the capabilities of computer technologies to calculate exact approximations of distributions, none of the existing computer technologies can provide the required computing resources. To solve this computationally expensive problem during the specified time, it is necessary to analyze the capabilities of promising computer technologies such as quantum and photon computers.

6. Use of Quantum Computer Technologies to Calculate Exact Approximations of Distributions

Quantum computer technologies were proposed in the 1980s by a number of famous scientists, such as Richard F. Feynman [13], Paul Benioff [14], K.A. Valiev and A.A. Kokin [15] and Yu. I. Manin [16]. The main idea of quantum computing is the following: a quantum system of q concurrently operating qubits has 2^q linearly independent states. According to the principle of quantum superposition, the state space of such a quantum register is a 2^qD Hilbert space [17]. One operation on a group of q qubits is calculated immediately for all its possible values, unlike a group of classical bits, when only one current value exists. This ensures the maximum possible parallelization of data calculations and an increase in performance to 2^q, which is called "quantum acceleration". Any object with two quantum states, such as polarization states of photons, electronic states of isolated atoms or ions, spin states of atomic nuclei, etc., can be a physical system implementing qubits. The structure of the quantum computer proposed by Russian scientists K.A. Valiev and A.A. Kokin [15] is shown in Figure 1. According to the principle formulated by R. Feynman [13], for any algorithm, it is possible to obtain an implementation on a quantum system, which will be no worse than its implementation on the classical von Neumann system. At the same time, it is shown [16] that "quantum acceleration" is not possible for any algorithm and for an arbitrary algorithm, the possibility of quantum acceleration is not guaranteed. A feature of quantum calculations is the probabilistic nature of the result of calculations, i.e., the result is true only with some probability.

Quantum computers were designed at Harvard University (51 qubit system) [18] and at the Polytechnic School of the University of Paris-Saclay (70 qubit system) [19]. The American company IonQ [20] announced the first commercial quantum computer based on ion traps, which contains 160 qubits, but for quantum operations only 79 qubits are used, and only 11 qubits are used for implementations of arbitrary quantum algorithms. In 2020, IBM [21] introduced the most powerful 64 qubit quantum computer. In Russia, the design of a general purpose quantum computer of 100 qubits is expected by 2024 [19]. At present, a quantum computer capable of operating with 120 bit data does not exist in the world.

Let us evaluate the possibility of calculating exact approximations of distributions on a quantum computer. The structure of the vector testing procedure of the considered problem [2] corresponds to the structure of the quantum computer (Figure 1), which allows us to expect a significant effect when solving the problem on quantum computers. According to the minimum evaluation, 120 bits are required to place the test vector candidate $\mu^{(v)}$; therefore, the quantum computer must contain 120 qubits operating concurrently. Let us suppose that a quantum computer operating with 120 qubits (let us call it QC-120) exists, and the problem of valid results is solved. If exact approximations are calculated on QC-120, we obtain all $\{\mu^{(1)}, \mu^{(2)}, \ldots, \mu^{(2^{120})}\}$ for the 120-bit test vector $\mu^{(v)} = \{\mu_0^{(v)}, \mu_1^{(v)}, \ldots, \mu_{23}^{(v)}\}$, i.e., $(2^{120} \cong 1.3292 \times 10^{36})$ possible solutions. Then, it is necessary to select $K_\mu(N, n, r)$ real solutions. For $N = 256$ and $n = 1280$, we obtain $K_\mu(256, 1280, 23) = 5.60075 \times 10^{25}$ from Table 1.

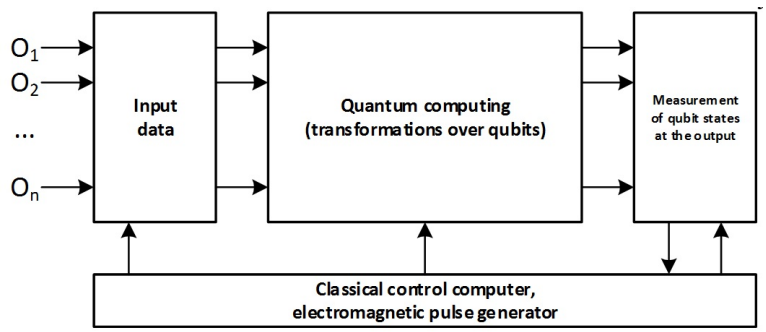

Figure 1. The structure of the quantum computer.

Thus, if we calculate the values of i-th possible solution, we obtain the SLE state $\{\underbrace{\mu^{(i)}, \mu^{(i)}, \ldots, \mu^{(i)}}_{2^{120}}\}$. To calculate the $(i+1)$-th possible solution, we obtain the SLE state $\{\mu^{(1)}, \mu^{(2)}, \ldots, \mu^{(2^{120})}\}$ and read the $(i+1)$-th possible solution. To test all possible solutions and to obtain $K_\mu(N, n, r)$ real solutions, we need 2^{120} accesses to QC-120, which drastically reduces the effect of concurrent calculations of 2^{120} possible solutions.

The need to check all the obtained solutions, the number of which corresponds to the dimension of the problem, does not allow achieving quantum acceleration and is a significant and fundamental limitation of the use of promising quantum computer technologies for calculating accurate approximations of distributions. The lack of a technical and technological base not only in the Russian Federation, but also in the world, is an additional, technological limitation that does not allow creation of a quantum computer system operating 120 qubits required to solve the problem of calculating exact approximations of statistical probability distributions. Therefore, the prospect of using quantum computer systems at the existing level of development of science and technology is quite modest for the considered problem, despite the potentially high performance. Perhaps the given evaluations for the considered problem can be revised with the development of quantum computing.

7. Use of Photon Computers to Calculate Exact Approximations of Distributions

Another relevant area for the development of promising computer facilities based on new physical principles is the design of computer systems that use the effects of interactions of coherent light waves generated by laser radiation and its carriers, i.e., photons [22,23].

The structure of the photon computer developed at the All-Russian Research Institute of Experimental Physics (Sarov) [22] is shown in Figure 2. The photon computer (Figure 2) consists of four large units: a unit for converting a task into a program for the photon computer (UCT), a laser radiation source (LRS), an input–output unit (IOU) and a photon processor (PP). In turn, the photon processor contains four processor elements which comprise arithmetic logic devices (ALU), control devices (CD) and switches (SW). The units of the photon computer are interconnected by electronic and optical channels, and the components of the processor elements are connected only by optical channels.

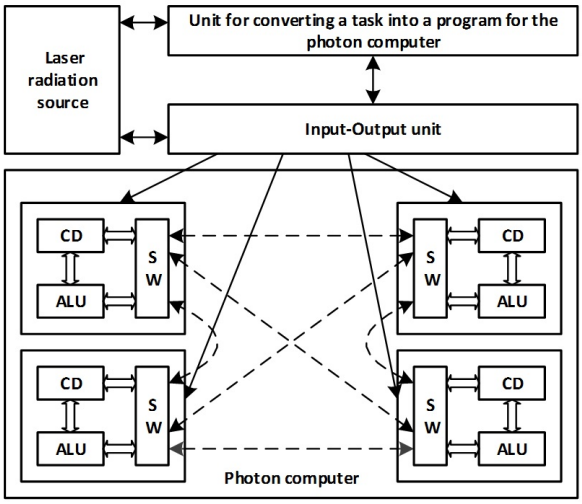

Figure 2. The structure of the photon computer.

The UCT transfers the photon program to the LRS which generates laser radiation and supplies it to the IOU, where it is divided into light rays according to the number of digits simultaneously supplied to the PE. The IOU generates a photon program and transmits it to a photon processor where calculations are performed by the processor elements. Light beams interact within the photon processor, and optical delay lines [23] perform synchronization. Within the photon processor, the PEs can be connected by optical channels to a multiprocessor environment of any topology [23]. A low power consumption and a high performance are the important advantages of photon computer systems.

It has been shown that the performance of a photon computing node can reach up to 10^{18} op/s per 100 W of power consumption [24].

To achieve the required performance of the photon computer P_{PHOTON}, it is necessary to use N_{PH_ND} nodes with the performance $P_{PH\_ND} = 10^{18}$ op/s each:

$$P_{PHOTON} = N_{PH\_ND} \times P_{PH\_ND}. \tag{8}$$

The performance level $P_{PHOTON} \leq 1.75 \times 10^{19}$ for calculating exact approximations (the parameters of the sample №3 in Table 1) is provided by

$$N_{PH\_ND} \geq 1.75 \times 10^{19} / P_{PH\_ND} = 1.75 \times 10^{19} / 1.0 \times 10^{18} \cong 17.5$$

nodes, i.e., no less than 18 photon computing nodes. To achieve the performance for calculating exact approximations of the whole spectrum of the parameters of the considered samples $\{N = \overline{2, 256},\ n = \overline{1, 5N}\}$, it is necessary to use no less than

$$N_{PH\_ND} \geq 6.15 \times 10^{45} / P_{PH\_ND} = 6.15 \times 10^{45} / 1.0 \times 10^{18} = 6.15 \times 10^{27}$$

computing nodes. This is much less than in all technologies that were reviewed earlier.

Unlike quantum computer systems, there is no available information about existing prototypes of digital photon computer systems. According to most works [22–24], scientific teams simulate the functioning of individual nodes and then evaluate the possible parameters of the calculator. There are works of academician V. A. Soifer [25–27] and some other scientists in the field of analog photonics. There, each computing node is created for a task with certain parameters, and the technology of developing and creating an analog photon computing node for an arbitrary task requires a whole cycle of research and development work. Therefore, there are no active prototypes of photon computer systems, technological bases and/or commercially available components today. Due to the current development level of science and technology, in the next 5–7 years it is impossible to consider the use of photon computer systems to solve the problem of calculating exact approximations of statistical probability distributions, although potentially such computer systems will have very high performances and require the smallest number of low-power computing nodes. Perhaps, owing to the development of photon computer technologies, it will be possible to revise and improve the presented evaluations for the considered task in the next few years.

8. Architecture-Independent Representation of the Exact Approximation Calculation Problem for Hybrid Computer Systems

According to the reviewed computer technologies and their current level of development, a possible solution is to design hybrid or heterogeneous computer systems containing the computing nodes of modern architectures (such as general purpose processors, graphics accelerators and FPGAs) capable of solving the problem with the given parameters in a reasonable time. Since the calculation of distributions is carried out by one and the same method for all parameters of samples, the possibility of programming different computer architectures in a single loop, which is provided in the architecture-independent programming paradigm, is especially important for such a system [28]. Taking into account the possible use of promising architectures of quantum computers and/or photon computing nodes, we consider an architecture-independent representation of the problem of calculating exact approximations for hybrid computer systems as especially significant.

Such capabilities are provided by the architecture-independent aspect-oriented language Set@l, which allows the developer to focus their attention on parallelizing methods when solving a problem, and not on their technical implementation in the selected computer architecture. The Set@l language reduces the problem of software transfer between different configurations and architectures of the HCS to the creation of aspects (descriptions) of technical means, which describe the key points of parallelization of the method on these technical means. At the same time, the source program implementing the data processing method remains unchanged [28].

The language Set@l is based on the paradigm of aspect-oriented programming (AOP) [29], according to which the algorithm of an application problem and the peculiarities of its implementation are described in the form of separate program modules. The Set@l program represents the information graph of the computational problem in the form of sets, whose partitioning and typing specify different parallelization options and other aspects of the implementation of the algorithm. Unlike other programming languages based on the set theory, for example, the languages SETL, SETL2 and SETLX, the Set@l language uses typing of sets according to different criteria and allows operations with fuzzy collections in accordance with the alternative set theory [30].

The Set@l language provides separate descriptions of the algorithm and the peculiarities of its implementation on a computer system by the use of the AOP paradigm. According to this paradigm, the cross-cutting concern of the program, which causes the negative effects of code tangling and scattering, is presented in the form of separate program modules (aspects). The source code, implementing the main functionality of the program, contains the user's markup, which determines its interaction with aspects during translation or execution. Analyzing the markup and source code, the preprocessor trans-

lator forms a new executable cross-cutting concern program. As a rule, the use of AOP technology simplifies the development and further support of software and increases the adaptability of programs to various modifications.

To implicitly describe various methods of parallelizing algorithms, the language of architecture-independent programming Set@l provides classification of collections by the type of parallelism of their elements during processing. When the parallelism nature of the set's elements is clearly defined, the types "par" (parallel-independent processing), "seq" (sequential processing), "pipe" (pipeline processing) and "conc" (parallel-dependent processing by iterations) are used. However, in some aspects of the program, the type of parallelism of a number of sets cannot be determined uniquely, since the architecture of the computer system on which the algorithm will be implemented is unknown. In this case, a special type of "imp" (implicit) is used, and the typing of collections is refined in other aspects of the program using special syntactic structures. If the aspects of the Set@l program do not change the algorithm in the process of its adaptation to the computer system's architecture, then the solution to the problem can be presented according to the classical Cantor–Bolzano set theory. However, in some cases, it is reasonable to modify the algorithm in accordance with the peculiarities of its implementation on the computer system with a certain architecture. In such cases, some collections are fuzzy and are not sets, so they cannot be specified as objects of the classical set theory. Using the Set@l language, we can describe different implementations of the same algorithm in a single-aspect-oriented program. To do this, classification of collections by the definiteness of their elements is performed according to the alternative set theory of P. Vopenka. The type "set" (set) corresponds to the classical clearly distinguished collection of elements, the type "semi-set" (sm) corresponds to a fuzzy collection and the type "class" (cls) corresponds to a set of objects, the type and partition of which cannot be uniquely determined at a given level of abstraction. An example of a computational problem that requires the Set@l fuzzy collections for its description is the Jacobi SLAE algorithm [31]. Unlike other approaches to parallel programming of high-performance computer systems, the Set@l programming language specifies not only boundary cases of the algorithm implementation, but also a family of intermediate options. They cannot be distinguished from the point of view of procedural programming but provide continuity of the description of the calculation model. Owing to the use of the Set@l language, it is possible to synthesize many variants of the problem solution and to switch between its elements depending on the architecture and configuration of the computer system. The program of fast Fourier transform illustrates this possibility. According to the available memory of the computer system, complex coefficients W are calculated in advance and loaded from memory or calculated with the help of basic and auxiliary components [32].

Thus, the algorithm of the problem in the Set@l language is presented as an architecture-independent source code by means of aspect-oriented programming, set theoretical code representation and relational calculus. The peculiarities of the algorithm's implementation are represented as the separate aspects that define the division into subsets and typing of the main collections of the program. The program can be quickly ported between computer systems with different architectures and adapted to any changes in hardware resources due to the use of aspects of the processing method, architecture and configuration. Owing to the suggested approach to parallel programming of high-performance computer systems in the Set@l language, there are new prospects for the development of architecture-independent and resource-independent software.

The language Set@l has been successfully used to solve algorithmically complex and resource-intensive problems, such as solving SLAE by the Gaussian [28] and Jacobi [31] methods and implementing the fast Fourier transform algorithm [32] and others with the same structure.

Therefore, the use of the Set@l language for the calculating exact approximations of distributions on a hybrid computer system will significantly decrease the programming

complexity of computing nodes of various architectures such as general purpose processors, graphics accelerators, FPGAs, quantum computers and/or photon computing nodes.

9. Conclusions

In this paper, we have analyzed the use of promising computer technologies to solve the computationally expensive problem of calculating Δ-exact approximations of statistical probability distributions by the second multiplicity method based on solving the SLAE of the second multiplicity. The method has polynomial complexity and allows parallelization into fragments by data. We have evaluated the possibilities of promising computer technologies based on quantum and photon computers for calculating distributions by the second multiplicity method. An analysis of the capabilities of quantum and photon computer technologies shows that they have great potential for solving this problem. However, at present, these technologies cannot provide a solution to the problem of calculating exact approximations of sample distributions with an alphabet power less than 256, a size less than 1280 characters and an accuracy of $\Delta = 10^{-5}$.

We have performed a theoretical analysis of quantum computer systems and their applicability for the problem, which showed that quantum acceleration is impossible. Thus, the current level of development of quantum computers is insufficient, and there are also no algorithms and criteria for choosing a suitable solution from a huge number of obtained solutions to the problem. In addition, the level of development of photon computers does not allow creating a computer with the required number of computing nodes.

Author Contributions: Conceptualization, A.M., I.L. and A.D.; methodology, A.M. and I.L.; validation, A.M., A.D. and L.S.; formal analysis, I.L.; investigation, A.M.; resources, A.M.; writing—original draft preparation, A.D. and L.S.; writing—review and editing, A.D. and L.S.; supervision, A.M.; project administration, I.L.; funding acquisition, I.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Russian Foundation for Basic Research, project number 20-07-00545.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Melnikov, A. Complexity of calculating exact probability distributions of symmetric additive separable statistics and application of limit distributions. *Proc. TUSUR* **2017**, *20*, 126–130. [CrossRef]
2. Melnikov, A.; Levin, I.; Dordopulo, A.; Pisarenko, I. Analysis of advanced computer technologies for calculation of exact approximations of statistics probability distributions. *Izvestia SFedU* **2021**, 7, 6–19.
3. Timonin, V.; Chernomordik, O. Method of calculating the exact distribution of statistics of the Kolmogorov–Smirnov type under Lehman alternatives. *Probab. Theory Its Appl.* **1985**, *30*, 572–573.
4. Tyannikova, N.; Timonin, V. Method of calculation of exact distributions of Kolmogorov–Smirnov type statistics in case of violation of homogeneity and independence of analyzed samples. *Sci. Educ.* **2014**, 11, 227–237.
5. Cramer, G. *Mathematical Methods of Statistics*; Princeton University Press: Princeton, NJ, USA, 1946.
6. TOP500. 2022. Available online: https://www.top500.org (accessed on 21 September 2022).
7. Antonov, A.; Afanasiev, I.; Voevodin, V. High-performance computer platforms: Current status and trends. *Numer. Methods Program.* **2021**, *22*, 135–177.
8. GeForce RTX 3090 NVIDIA. 2022. Available online: https://3dnews.ru/1021405/obzor-videokarti-nvidia-geforce-rtx-3090 (accessed on 21 September 2022).
9. NVIDIA Quadro RTX 6000. 2022. Available online: https://3dnews.ru/1027931/nvidia-obyavila-o-dostupnosti-sverhmoshchnoy-videokartirtx-a6000-s-48-gbayt-gddr6-i-tsenoy-5500 (accessed on 21 September 2022).
10. Levin, I.; Fedorov, A.; Doronchenko, Y.; Raskladkin, M.; Guzik, V.; Kalyaev, I. *Reconfigurable Computer Systems*; SFedU Publishing: Taganrog, Russia, 2016.

11. Levin, I.I.; Dordopulo, A.I.; Fedorov, A.M.; Kalyaev, I.A. Reconfigurable computer systems: From the first FPGAs towards liquid cooling systems. *Supercomput. Front. Innov.* **2016**, *3*, 22–40.
12. Levin, I.; Fedorov, A.; Doronchenko, Y.; Raskladkin, M. Promising high-performance reconfigurable computers with immersion cooling. In *Supercomputer Technologies*; SFedU Publishing: Taganrog, Russia, 2020; pp. 29–34.
13. Feynman, R. Simulating physics with computers. *Int. J. Theor. Phys.* **1982**, *21*, 467–488. [CrossRef]
14. Benioff, P. Quantum mechanical hamiltonian models of turing machines. *J. Stat. Phys.* **1982**, *29*, 515–546. [CrossRef]
15. Valiev, K.; Kokin, A. *Quantum Computers: Hopes and Reality*; RHD: Izhevsk, Russia, 2004.
16. Manin, Y.I. *Computable and Non-Computable*; Soviet Radio: Moscow, Russia, 1980.
17. Moren, K. *Hilbert Space Methods*; Mir: Moscow, Russia, 1965.
18. Physicists from Russia and the United States Have Created the First 51-Qubit Quantum Computer. 2022. Available online: https://ria.ru/20170714/1498476410.html? (accessed on 2 May 2022).
19. How Russia Will Spend 15 Billion to Create a Quantum Computer. 2022. Available online: https://www.cnews.ru/articles/2020-01-29_kak_rossiya_potratit_15_mlrd_na_sozdanie?ysclid=liy7s0gbbk340669962 (accessed on 2 May 2023).
20. IonQ Announced the Creation of the World's Most Powerful Quantum Computer. 2022. Available online: https://3dnews.ru/1060908/kompaniya-ion-q-soobshchila-o-sozdanii-samogo-moshchnogo-v-mire-kvantovogo-kompyutera? (accessed on 2 May 2022).
21. IBM Demonstrated a Quantum Computer with "Quantum Volume" of 64. 2022. Available online: https://habr.com/ru/company/raiffeisenbank/news/t/516062/? (accessed on 2 May 2022).
22. Stepanenko, S. Photon computer. Design principles. Parameters evaluation. *Proc. RAS* **2017**, *476*, 389–394.
23. Stepanenko, S. Photon computer: Structure and algorithms, parameters evaluation. *Photonics* **2017**, *7*, 72–83. [CrossRef]
24. Stepanenko, S. Interference logic elements. *Proc. RAS Math. Inform. Control Process.* **2020**, *493*, 64–69.
25. Soifer, V.A.; Kharitonov, S.I.; Khonina, S.N.; Strelkov, Y.S.; Porfirev, A.P. Spiral caustics of vortex beams. *Photonics* **2021**, *8*, 24. [CrossRef]
26. Kashapov, A.I.; Doskolovich, L.L.; Bezus, E.A.; Bykov, D.A.; Soifer, V.A. Spatial differentiation of optical beams using a resonant metal-dielectric-metal structure. *J. Opt.* **2021**, *23*, 023501. [CrossRef]
27. Golovastikov, N.V.; Doskolovich, L.L.; Bezus, E.A.; Bykov, D.A.; Soifer, V.A. An Optical Differentiator Based on a Three-Layer Structure with a W-Shaped Refractive Index Profile. *J. Exp. Theor. Phys.* **2018**, *127*, 202–209. [CrossRef]
28. Levin, I.; Dordopulo, A.; Pisarenko, I.; Melnikov, A. Approach to architecture-independent programming of computer systems in aspect-oriented SET@L language. *Izvestiya SFedU* **2018**, *3*, 46–58.
29. Safonov, V. *Using Aspect-Oriented Programming for Trustworthy Software Development*; John Wiley & Sons.: New York, NY, USA, 2008.
30. Vopenka, P. *The Alternative Set Theory: A New Look at Infinity*; Institute of Mathematics Publishing: Novosibirsk, Russia, 2004.
31. Levin, I.; Dordopulo, A.; Pisarenko, I.; Melnikov, A. Description of Jacobi algorithm for solution of linear equation system in architecture-independent SET@L programmning language. *Izvestiya SFedU* **2018**, *5*, 34–48.
32. Levin, I.; Dordopulo, A.; Pisarenko, I.; Melnikov, A. Architecture-independent program of Fast Fourier transform in SET@L programming language. *Her. RSREU* **2019**, *68*, 28–36. [CrossRef]