UDC 004.75

Valerii Demchyk, Vitalii Tyzun,
Alexander Korochkin, Olga Rusanova.

THE APPLICATION OF WCF TECHNOLOGY TO INCREASE THE EFFICIENCY OF PARALLEL COMPUTING IN CLOUD DISTRIBUTED COMPUTER SYSTEMS

The article considers the effectiveness of WCF technology usage for the organization of parallel computing in cloud distributed computer systems (DCS). The results of a comparative study of the effectiveness of this technology compared to MPI technology are demonstrated on the example of solving real mathematical problems in real cloud DCS.

Keywords: distributed computing, distributed computer system, cloud computer system, node, parallelism, granularity, acceleration coefficient, WCF, TPL, MPI, OpenMP.

Fig.: 2. Bibl.: 7.

Relevance of the research topic. The positive dynamic of the development of informatics and computer technologies can be observed throughout the entire existence of this industry. Hardware and software development are inseparable linked, as more advanced algorithms and software systems require more powerful hardware support. At first, the main approach to increasing hardware power was to improve the frequency and electrical characteristics of the main components of computer logic elements, as well as increasing the number of these same elements per unit space. However, over time, this path was almost exhausted [1], as almost the limit characteristics at the physicochemical level for silicon elements of the chips were achieved. In this regard, the vector of further development of computer systems performance has shifted towards the simultaneous use of several cores or processors within one parallel system, and then several such systems within one distributed system [2].

Modern high-level computer systems, including public user and mobile platforms, are currently built on the principles of parallel computing systems. System-level software (operating systems) is also based on the principles of time division and support for multicore and/or multiprocessor. Moreover, we can say that the most effective solution of the problem of organizing the parallel operation on the hardware and software levels is one of the key requirements for modern operating systems. However, due to the reasons explained above, even parallelism is not enough for modern computer systems to work effectively. Therefore, the most demanding calculations, and especially those to which critical duration conditions are put forward, are performed only on distributed systems, among which the most popular are cluster systems [3].

Formulation of the problem. Despite the potentially infinite possibilities of increasing the speed of programs in parallel and distributed systems, in practice they are limited from a mathematical point of view, because there are a huge number of algorithms that cannot have a perfect parallelization implementation. This leads to the fact that when these implementations are performed on real parallel and distributed systems, the relative efficiency of the components of the computing elements of these systems is not maximum, because pauses and delays appear in the work of each of them. Therefore, by a detailed analysis of the above factors, we can conclude that in solving modern science-intensive problems or the organization of distributed queuing servers based on parallel and distributed computer systems, the main impact is made not only by powerful hardware but also software that is responsible for functioning of systems at different levels, from operational and communicative to applied [4].

However, at the same time, the technologies that are now most commonly used to organize computing in distributed computing systems are mostly strictly tied to the C++ programming language (such as the Intel Threading Building Block) or are also based on outdated approaches. and paradigms (like Message Passing Interface, MPI).

Analysis of recent research and publications. The analysis showed that currently all modern technologies of computing in distributed systems adopted a model of interaction of parallel processes, which is based on low-level messaging. An advantage of this model in systems with local memory is its low level. The programmer is provided with the maximum means of organization and control over communications and program execution, on the basis of which it is already possible to build higher-level models for systems with local memory. The program in this case in its structure and logic is as close as possible to the system.

However, this is the main disadvantage – the program reflects primarily the structure of the system, rather than the problems and logic of the target task. In addition, program code with low-level primitives is usually more voluminous. Also the use of some low-level primitives can lead to the deadlocks described above, starvation of flows, and lack of guarantees of progress.

Study [5] presents a model of higher-level messaging in a distributed system, but it also remains tied to the MPI, being a superstructure over it, which leads to the following disadvantages.

Selection of unexplored parts of the general problem. MPI remains the main generally accepted standard of distributed computing technologies. Despite the high speed of programs organized by using this technology, and the general convenience of its use, its main problem is its obsolescence, which is only growing from year to year. Among the key points that indicate its obsolescence, there are three, each of which follows from the previous one:

1. Limited target languages. MPI technology was developed for the C and Fortran languages. With the loss of these languages, developers are forced to develop using MPI in combination with C++. However, due to the problem of backward compatibility, MPI has not been translated to fully support all tools that appeared in C++ compared to C, so it remains necessary to operate only with those C++ primitives that are compatible with the C language.

2. Focusing on procedural programming. MPI technology was developed at a time when the main paradigm used in application programming was procedural. However, currently much more complex paradigms are considered to be generally accepted – object-oriented, functional, reactive.

3. Low level of abstraction, the need to allocate a significant intermediate level in the software package to organize the interaction between the layer of business logic and the layer of communication, interfaces conversion, and so on. For example, in MPI, the maximum abstraction that can be sent is a predefined structure.

A relatively new WCF (Windows Communication Foundation) technology can be offered as a high-level alternative to MPI technology. From the very beginning, WCF technology was developed to solve the problems described above, which was eventually done. In addition, its application allows you to combine computers with different architectures without being distracted by compatibility issues, because, like all technologies based on the .NET Framework, it works within the CLR (Common Language Runtime) virtual machine. It is based on the concept of deferred evaluation and calling remote methods, which allows you to organize much more flexible interaction than conventional messaging technologies. In this case, the parallelism within each end machine can be implemented through the accompanying built-in .NET Framework library TPL (Thread Parallel Library).

The only and main disadvantage that can be identified in WCF technology, compared to OpenMP and MPI, is the greater complexity and volume of program code.

The research objective. The objective is to test the hypothesis that the use of WCF technology leads to an increase in the acceleration coefficient of parallel computing in distributed computer systems when solving practical problems, compared with the acceleration coefficients obtained when solving the corresponding problems in similar systems, but using MPI technology. To do this, it is necessary to develop a software package to solve several classic tasks for high-load computing in distributed computer networks, one using the connection of MPI + OpenMP technology, the other – WCF technology (with TPL for node-level parallelism), and conduct comparative testing of developed programs in distributed computer systems. In particular, basing on previous studies [6-7], it is advisable to use fine-grained parallelism in both programs. In C#, its support tools are embedded in the TPL, and to provide similar support for fine-grained parallelism in programs implemented by MPI, we will combine them with OpenMP technology.

Test results. The programs were tested on a distributed computer system hosted by the Google Cloud Console service, which consisted of four nodes, each containing one virtual Intel 4-core Intel SkyLake processor.

Software: Windows Server 2012 R2 Datacenter x64, .NET Framework 4.7, OpenMP 3.0, Microsoft MPI 9.0.1.

Multiplication of two square matrices with dimension of N rows * N columns (since parallel execution of this task involves insignificant data exchange between nodes, only sending it at the beginning and collection of results) and calculation with given accuracy of trigonometric function of hyperbolic arctangent through the calculation of the convergence of the series (because the parallel performance of this task involves a significant exchange of data (numbers of current iterations) between the nodes of the system) were chosen as the target tasks for testing the hypothesis.

The graph shown in Figure 1 shows the dependence of the program acceleration coefficient on the dimension (N) of the matrices and the means by which the program was organized.

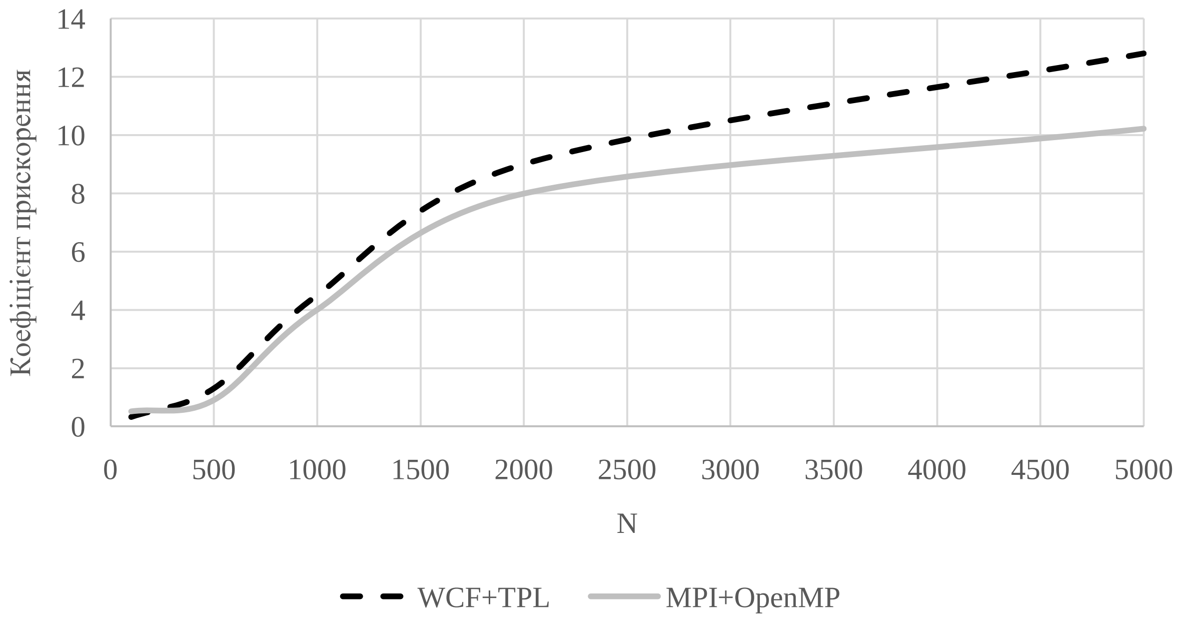

Fig. 1. Graph of the acceleration coefficient of the multiplication program of two square matrices depending on the dimension (N) and the means of organization

The graph shown in Figure 2 shows the dependence of the program acceleration coefficient on the accuracy (ε) and the means by which the program was organized.

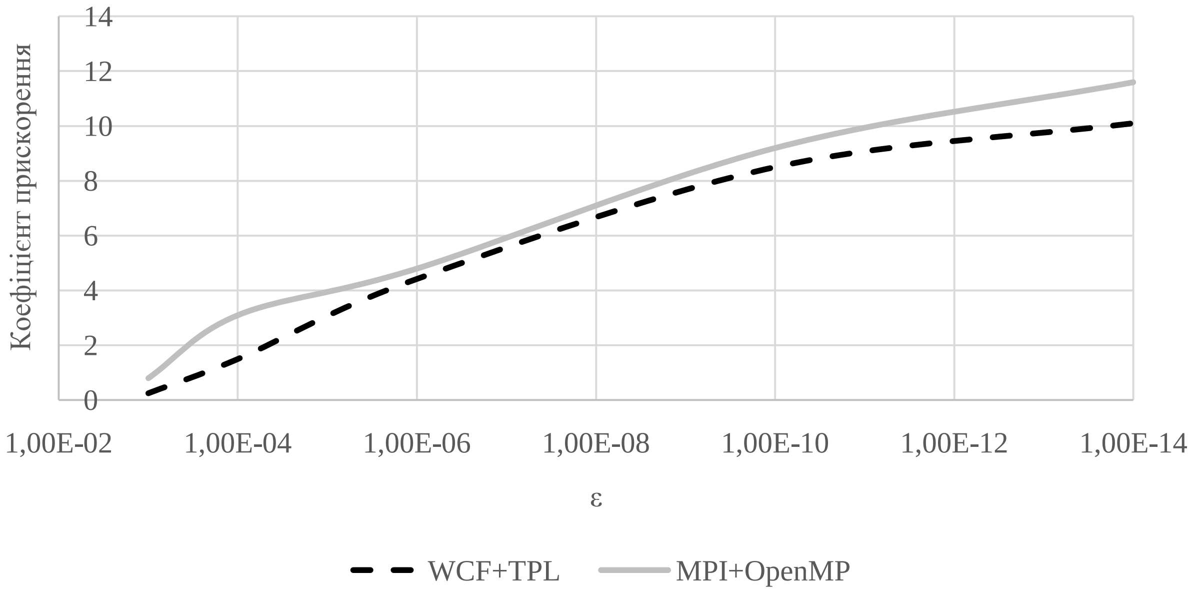

Fig. 2. Graph of the dependence of the acceleration coefficient of the program of finding the convergence of the series depending on the accuracy (ε) and the means of organization

Conclusions. The test results showed a significant overall efficiency of the use of cloud distributed computer systems in solving these problems, both in the case of organizing their solution by means of WCF, and in the case of organizing their solution by means of MPI. It was possible to achieve the maximum value of the acceleration coefficient for the problem of multiplication of matrices in 12.6 when using WCF and TPL technologies and for the problem of calculating the sum of the decomposition of a function in a series of 11.8 when using MPI-OpenMP technologies.

Based on the graphs shown above, we can also say the following:

1. Solving small problems in distributed computer systems is not justified, the acceleration coefficients in this case are less than 1. This means that solving the problem within one machine would be more appropriate than distributing it among all machines in the system. This can be explained by significant sending costs, as network time is up to 10,000 times more expensive than CPU time.

2. By using WCF technology, it was possible to achieve a higher acceleration coefficient for the problem of matrix multiplication than when using MPI technology. On average, WCF technology increased the acceleration coefficient by up to 15% compared to MPI technology.

3. At the same time, when using WCF technology to solve the problem of calculating the value of the sum of the decomposition of a function in a series, there was a decrease in average to 8% of the acceleration coefficient, compared with the use of MPI technology.

It follows from paragraphs 2 and 3 that for tasks that do not require intensive data exchange between nodes of the distributed system in the process of work, it is more appropriate to organize the solution of these problems by means of WCF and TPL. And for tasks that in the process of the work demand intensive communication between nodes of system it is better to choose MPI technology. Therefore, it can be argued that MPI has more optimized for computational tasks (compared to WCF) communication tools between nodes, network protocols, and so on. And in WCF the computational process and parallelism at the node level is more optimized.

Thus, the hypothesis of a more efficient solution of parallel problems in distributed computer systems in the case of its organization by WCF, rather than MPI, is only partially confirmed and it is valid only for a class of problems that do not require significant communication in direct computing.

REFERENCES

1. Sutter, H.: The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software / H. Sutter // Dr. Dobby’s Journal. – 2005. –Vol. 30, no. 3.

2. Martonosi, Margaret. Parallelism, heterogeneity, communication: Emerging challenges for performance analysis / Margaret Martonosi // Performance Analysis of Systems and Software (ISPASS), 2012 IEEE International Symposium on IEEE. – Washington, DC, USA: IEEE Computer Society, 2012. – P. 124.

3. Tanenbaum, Andrew S. Distributed Systems: Principles and Paradigms (2nd Edition) / Andrew S. Tanenbaum, Maarten van Steen. – Upper Saddle River, NJ, USA: Prentice-Hall, Inc. – 2006.

4. Software as a service for data scientists / Bryce Allen, John Bresnahan, Lisa Chilers [et al.] // Commun. ACM. – 2012. – Vol. 55, no. 2. – P. 81-88.

5. Стіренко С. Г. Організації паралельний обчислювальних процесів в кластерних системах [Текст] / С.Г. Стіренко, – К.: «Три К», 2014. – 196 с.

6. Демчик В. В. Дослідження ефективності дрібнозернистого паралелізму в багатоядерних комп'ютерних системах / В. В. Демчик, О. В. Корочкін, О. В. Русанова // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка : зб. наук. праць. – К. : Век+, 2018. – № 66. – С. 56 – 61.

7. Демчик В. В. Застосування дрібнозернистого паралелізму для підвищення ефективності паралельних та розподілених обчислень / В. В. Демчик, О. В. Корочкін // Безпека. Відмовостійкість. Інтелект. Збірник праць міжнародної науково-практичної конференції ICSFTI2018. Київ, Україна, 10-12 травня 2018 р. / КПІ ім. Ігоря Сікорського –К. : КПІ ім. Ігоря Сікорського, Вид–во «Політехніка», 2018. – С. 362 – 368.

AUTHORS

Valerii Demchyk – student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: moshel.valerii@gmail.com

Vitalii Tyzun – Master of Science, graduated in 2019 from the Faculty of Informatics and Computer Science of the National Technical University of Ukraine " Igor Sikorsky Kyiv Polytechnic Institute" with a degree in "Software Engineering".

E-mail: vitaliy.tyzun@gmail.com

Aleksandr Korochkin (supervisor) – docent, candidate of Technical Sciences, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: avcora@gmail.com

Olga Rusanova (supervisor) – docent, candidate of Technical Sciences, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: olga.rusanova.v@gmail.com