Modern Ways to Organize Computations in Cloud Environment

Tamara Bardadym
Department of Intelligent Information Technologies
V.M. Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine
Kyiv, Ukraine
0000-0001-8657-8687

Vasyl Gorbachuk
Department of Intelligent Information Technologies
V.M. Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine
Kyiv, Ukraine
0000-0001-5619-6979

Oleksandr Lefterov
Department of Intelligent Information Technologies
V.M. Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine
Kyiv, Ukraine
0000-0002-1475-1281

Sergiy Osypenko
Department of Intelligent Information Technologies
V.M. Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine
Kyiv, Ukraine
0000-0002-1903-8207

***Abstract*—A short review of modern tools and capacities to organize large-scale computations is presented based on the author's experience related with reproducibility of results obtained by researchers in the process of their remote team work. The work assumes proper access to large and heterogeneous data sets stored, application of specific programming tools available, software transfer between different environments, efficient organization of cloud computing.**

Keywords—cloud service; containerized application; isolated software environment; reproducibility of calculations

I. Introduction

The main goal of this paper is to describe our experience in developing applied analytical system for biomedical computations based on certain main principles of organization and existing modern computing tools [1−2]. The principles include the reproducibility of research results related with big data [3−4]. Similarly to experiments in physics, biology, chemistry, numerical experiments in cybernetics and informatics may be reproduced. In the context of computing, reproducibility means reconstruction of all conditions for the numerical experiments including the software used. In addition, reproducibility of big data becomes the requirement for biomedical numerical experiments.

II. Containerisation and Other Tools

Modern software tools used in biomedical studies are quite often distributed in the form of packages of the R software environment. There are more than 10,000 packages available in this environment, a special place among which is the Bioconductor package [5], it contains about 2000 bioinformatics computing methods. There are also R-packages where of high-performance, parallel and distributed computing is implemented [6]. This allows to effectively use the R in cluster environments and on grid platforms. The use of these approaches involves configuring the computing environment, installing additional software. Moreover, to ensure reproducibility of calculations, it is necessary to be able to provide a computing environment identical to the one in which the results were obtained.

The other approach to the implementation of high-performance computing in the study of the human genome is implemented in the Cancer Genomics Cloud (CGC) cloud service [7]. It is a specialized cloud platform that provides free access to genetic, medical databases, in particular – The Cancer Genome Atlas (TCGA) [8], and more than 450 public applications designed to analyze data on this topic. It is possible to expand this list with the own applications, data sets, research results (currently there are more than one million on this service), to involve other researchers in projects. The main difference of this approach is that the cloud service does not require the installation of additional software. All the software needed (for example, the language environment R or Python with all the necessary packages) is isolated in Docker containers [9], so, every software application is nothing else than a collection of containers. Such a software application can be described as an analysis scheme (workflow). The sequence of steps (Docker processes), input and output parameters, requirements for the number of resources required for the operation of such a scheme are described using the Common Workflow Language (CWL, [10]). This approach with the help of special tools (for example, cwltool and visual and code editor for Common Workflow Language – Rabix composer, see [11]) can be used to obtain reproducible biomedical calculations both in the cloud environment and on a personal computer (for example, for its testing).

The popularity of the containerized application approach in bioinformatic computing is indicated by the fact that the Bioconductor package is available in a containerized form as a service on Amazon Web Services and CGC (where it can be used for interactive analysis [7]). The containerized web version of R-Studio with the R environment, which includes the Bioconductor package, can be downloaded from the Docker Hub resource [12].

III. Containerisation. Our experience

When creating the analytical system of biomedical calculations [1] we had a need to test the developed software on real data. Based on the approaches presented in [13-14], optimization models and methods for solving problems of

DOI: https://doi.org/10.54381/pci2023.04

constructing linear classifiers have been developed. In particular, the problem of constructing classifiers for linearly indivisible sets was formulated as a problem of minimizing the band of incorrect classification of training sample points. This model belongs to the class of optimization problems of non-convex programming and is multi-extreme. Various formulations of this problem are offered, approaches to construction of approximate decisions and calculation of estimations of optimum values are considered. To solve these optimization problems, methods of non-smooth optimization, namely *r*-algorithms of N.Z. Shor [15-16] and exact penalty functions [17-18] were used. When creating appropriate software, modern libraries of linear algebra, similar to [19-21] should be used to speed up arithmetic operations. So, the combination of algorithms based on non-smooth optimization methods and the use of modern libraries of linear algebra was implemented in the developed software module NonSmoothSVC.

To test the abilities of the new classifier NonSmoothSVC a comparison with existing tools was made. The methods integrated into the library scikit-learn [22] were chosen, namely Linear SVC, NuSVC, Ada Boost. The two last methods are non-linear classifiers, they were chosen to get additional information concerning advantages of different methods for different problems. The need to test new software on real data forced us to locate the software module NonSmoothSVC into a containerized application (using Docker technology [9]) for use on a personal computer, as well as on a cluster, grid, and cloud environment.

Computational experiments have demonstrated that on some data sets the NonSmoothSVC has qualitative advantages over other methods involved in the comparison, but is inferior in speed. Particularly, on linearly separable samples the NonSmoothSVC gained an advantage over the LinearSVC in the number of cases with better classification accuracy. On the unbalanced samples, the NonSmoothSVC software slightly outperformed the LinearSVC software in the number of cases with better classification accuracy on average, but demonstrated an advantage in some parts of the classification accuracy scale. Full description of numerical experiments and the results of testing can be found in the report (in Ukrainian) at http://moderninform.icybcluster.org.ua/ais/.

Thanks to the containerized form, the developed software can become publicly available tool and application of this and other services in the problems of constructing optimized linear classifiers using modern libraries of linear algebra. In the presence of technical possibilities, parallelization on microprocessor networks looks promising. This approach is especially recommended in the case of large data samples, when the dimension of the feature space is tens of thousands.

In terms of using cluster technologies, creating environments separate for each user and maintaining them in a conflict-free state is quite a burdensome task (unless you use special software configuration tools). Most of the libraries and applications used in biomedical computing do not provide efficient use of parallel multithreaded computing with multi-core processors. Nevertheless, our first numerical experiments on OpenStack [23] test deployment and comparison of virtual and real cluster environments look to be perspective. In the paper [24] we give a detailed description of test deployment of OpenStack to create a scalable computing environment for reproducible scientific computing using modern technological solutions, which can be applied to both cloud (OpenStack, AWS, Google) and cluster platforms (Slurm).The structure of the created test containerized (using Singularity technology) biomedical application which contains modern software and libraries and can be used in conventional and cloud virtual cluster environments is briefly described. The results of a comparative test of this application in the virtual cluster environment Slurm under the control of OpenStack and in the node of cluster SKIT-4.5 in the V.M. Glushkov Institute of Cybernetics of the NAS of Ukraine are given. Information on solving the problem of finding the optimal in terms of saving resources scaling parameters for the developed application in two comparable cluster environments is given.

Some features of the use of these cluster environments are clarified, in particular, a comparison of the dependence of the application speed on the number of parallel processes for two cluster environments is presented. Empirical data illustrate the nature of the load on the OpenStack server and the use of RAM on the number of parallel processes. Possibilities of portability between the specified cluster environments, scaling of calculations and maintenance of reproducibility of calculations for the offered test application are demonstrated. The advantages of using OpenStack technology for scientific biomedical calculations are pointed out. The described example of test deployment and use of OpenStack gives an idea of the requirements for the necessary technical base to ensure the reproducibility of scientific biomedical calculations in cloud and cluster environments.

IV. Technologies That Ensure the Reproducibility of Scientific Calculations

Taking into account the peculiarities of biomedical computing, reproducibility and their horizontal scaling (the ability to increase the number of identical computing units to solve one problem) can be achieved through the use of containerized applications, software pipeline computing and parameterization of software environment.

Technologies of containerization of software applications. Due to the containerization of biomedical applications (Docker, Singularity containerization technology) the following can be achieved: reproducibility of the conditions in which the calculations took place (invariability of software including software and libraries), the possibility of horizontal scaling provided the use of "stunning" model of parallelism in cluster (Singularity) and cloud (using Docker) calculations.

Technologies of software pipelining of calculations. Software pipeline allows you to organize flow calculations (calculations in which the inputs and outputs of processes are interconnected). Thanks to the use of tools for automation of flow calculations (workflow engine) such as CWL (Common Workflow Language), GWL (Guix Workflow Language), Snakemake, Nextflow, it is possible to present a specific calculation in the form of a task (text file, as usual, in YAML format or JSON), the results of which can be reproduced [24]. In addition, there are tools that allow you to create / display such tasks in the form of a graph of processes and data flows. An example of such a tool is RABIX (Reproducible Analyzes for Bioinformatics) – a graphical editor for CWL. Some pipeline tools also use containerization (for example, CWL) – such tasks can be performed both on a personal computer and in a cloud environment. An important feature of streaming automation tools is that the task description syntax allows you to specify the scale of the calculations, indicating the number of resources required. Seven Bridges' product, Cancer

Genomics Cloud [7] is an example of a cloud software platform for performing reproducible biomedical computations using containerization and pipelining.

Technologies for parametrization of software environment. Parametrization of the software environment allows you to reproduce, if necessary, an identical computing environment. GNU Guix, Conda, Bioconda are examples of tools that allow you to create an isolated software environment for individual users in a cluster [25].

V. Conclusion

The paper has briefly described the first-hand experience in complex organization of biomedical computations to provide:

- reproducibility of results when any researcher can verify the conclusions and technical features of new results using precisely described containerized environment and software used;
- proper access to real data placed in cloud environment;
- application of cloud services and specific programming tools developed for the class of problems considered;
- option of using computational tools for different classes of problems;
- an opportunity of using the tools developed at technical devices of various classes from a personal computer to a powerful cluster.

Cloud and cluster technologies have enabled use of containerization, and other modern computing tools. The first-hand experience and practice would be helpful in efficient organization of cloud computing.

References

[1] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] T. A. Bardadym, V. M. Gorbachuk, N. A. Novoselova, S. P. Osypenko, and V. Yu Skobtsov, “Intelligent analytical system as a tool to ensure the reproducibility of biomedical calculations,” Artificial Intelligence, 2020, №3, pp. 67–81.

[3] T. A. Bardadym, V. M. Gorbachuk, N. A. Novoselova, S. P. Osypenko, V. Yu Skobtsov, and I. E. Tom, “On Biomedical Computations in Cluster and Cloud Environment,” Cybernetics and Computer Technologies, 2021, №2, pp. 76–84 https://doi.org/10.34229/2707-451X.21.2.8.

[4] J. Ioannidis, “Why Most Published Research Findings Are False,” PLoS Medicine, 2005, 2 (8), p.e124. https://doi.org/10.1371/journal.pmed.0020124 (accessed June 30, 2023).

[5] M. Baker, "Reproducibility crisis?", Nature, 2016, vol. 26, №533, pp. 353-366.

[6] Bioconductor. Open source software for bioinformatics. Available at: https://www.bioconductor.org (accessed June 30, 2023).

[7] CRAN Task View: High-Performance and Parallel Computing with R. Available at: https://cran.r-project.org/web/views/HighPerformanceComputing.html (accessed June 30, 2023).

[8] The Cancer Genomics Cloud. Available at: http://www.cancergenomicscloud.org (accessed June 30, 2023).

[9] The Cancer Genome Atlas (TCGA). Available at: http://www.cancer.gov/aboutnci/organization/ccg/research/structuralgenomics/tcga (accessed June 30, 2023).

[10] Tools for creation of isolated Linux-containers. Available at: http://www.docker.com/ (accessed June 30, 2023).

[11] Common Workflow Language. Available at: https://www.commonwl.org/ (accessed June 30, 2023).

[12] Rabix composer. Available at: https://github.com/rabix/composer/ (accessed June 30, 2023).

[13] Docker containers for Bioconductor. Available at: https://hub.docker.com/r/bioconductor/bioconductor_docker/ (accessed June 30, 2023).

[14] Yu. I. Zhuravlev, Yu. P. Laptin, A. P. Vinogradov, N. G. Zhurbenko, O. P. Lykhovyd, and O. A. Berezovskyi, “Linear classifiers and selection of informative features,” Pattern Recognition and Image Analysis, 2017, vol. 27, №3, pp. 426–432.

[15] Yu. P. Laptin, Yu. I. Zhuravlev, and A. P. Vinogradov, “Comparison of some approaches to classification problems, and possibilities to construct optimal solutions efficiently,” Pattern Recognition and Image Analysis, 2014, 24 (2), pp. 189–195.

[16] N. Z. Shor, “Nondifferentiable Optimization and Polynomial Problems,” London, KluwerAcad. Publ., 1998. 381 p.

[17] N. Z. Shor, and N.G. Zhurbenko, “A minimization method using space dilation in the direction of the difference of two successive gradients,” Cybernetics, 1971, №3, pp. 51–59 (in Russian).

[18] Yu. P. Laptin, and T. A. Bardadym, “Problems of determining the coefficients of exact penalty functions,” Cybernetics and systems analysis, 2019, №3, pp. 64–79 (in Russian).

[19] Yu. P. Laptin, and T. A. Bardadym, “On approximate calculation of the coefficients of exact penalty functions,” Mathematical and computer modeling. Series: physical and mathematical sciences, 2019, Iss. 19, pp. 54-60 (in Russian).

[20] Chang C.-C., Lin C.-J. LIBSVM - A Library for Support Vector Machines. Available at: https://www.csie.ntu.edu.tw/~cjlin/libsvm/ (accessed June 30, 2023).

[21] BLAS (Basic Linear Algebra Subprograms). Available at: http://www.netlib.org/blas/ (accessed June 30, 2023).

[22] LAPACK – Linear Algebra PACKage. Available at: http://www.netlib.org/lapack/ (accessed June 30, 2023).

[23] Free software machine learning library for the Python programming language. Available at: https://scikit-learn.org/stable/index.html/ (accessed June 30, 2023).

[24] O. Sefraoui, M. Aissaoui, and M. Eleuldj, “OpenStack: toward an open-source solution for cloud computing,” International Journal of Computer Applications, 2012, vol.55, №3, pp. 38–42.

[25] T. O. Bardadym, O. V. Lefterov, and S. P. Osypenko, “Experience of OpenStack test deployment and comparison of virtual and real cluster Environments,” Cybernetics and Computer Technologies, 2021, №3, pp. 74–85 (in Ukrainian) https://doi.org/10.34229/2707-451X.21.3.7