Performance estimation of a BOINC-based Desktop Grid for large-scale molecular docking

Natalia Nikitina[1][0000−0002−0538−2939]✉, Maxim Manzyuk[2][0000−0002−6628−0119], Črtomir Podlipnik[3][0000−0002−8429−0273], and Marko Jukić[4,5][0000−0001−6083−5024]

[1] Institute of Applied Mathematical Research, Karelian Research Center of the Russian Academy of Sciences, 185910, Petrozavodsk, Russia
✉nikitina@krc.karelia.ru
[2] Internet portal BOINC.Ru, Moscow, Russia
[3] Faculty of Chemistry and Chemical Technology, University of Ljubljana, 1000 Ljubljana, Slovenia
[4] Chemistry and Chemical Engineering, University of Maribor, 2000 Maribor, Slovenia
[5] Faculty of Mathematics, Natural Sciences and Information Technologies, University of Primorska, 6000 Koper, Slovenia

Abstract. This paper addresses the performance evaluation of a heterogeneous distributed computing environment (Desktop Grid) for large-scale molecular docking and scoring. At the initial stage of a distributed computing project based on a Desktop Grid, the set of computational nodes is dynamically growing. With peculiarities of Desktop Grids such as heterogeneity and unreliability of the nodes, this factor imposes difficulties on the task scheduling and algorithm scaling. We analyze a Desktop Grid performance, propose the efficiency metrics for this stage and provide statistics and analysis of the initial stage of a volunteer computing project named SiDock@home that is performing large-scale in silico medicinal chemistry experiments.

Keywords: Distributed computing · Volunteer computing · Desktop Grid · Task scheduling · BOINC · Virtual screening · Molecular docking.

1 Introduction

Desktop Grids are an essential tool for performing computationally-demanding scientific research. They combine non-dedicated geographically distributed computing resources (typically, desktop computers) connected to the central server by the Internet or a local access network. The nodes perform computations for the Desktop Grid in their idle time. The resources are usually provided either by the volunteer community or by individuals or organizations related to the performed research. Many of the world's leading research institutions run large-scale computational projects based on Desktop Grids (Washington University: Rosetta@home [18], Folding@home [21]; CERN: LHC@home [14]; University of Oxford: Climateprediction.net [5], and many others).

Unlike computational clusters and supercomputers, Desktop Grids are devoid of characteristics such as the high-speed interconnection between computational nodes, homogeneity, reliability, and availability of nodes during defined periods of time. These disadvantages restrict the class of computational problems that one can solve on Desktop Grids to the type of bag-of-tasks problems.

Nonetheless, Desktop Grids serve as an affordable, quickly deployable tool for solving urgent large-scale computational problems. An essential feature of the Desktop Grid technology is the ability to attract a large number of inexpensive computing resources for a temporary or long-term mission, providing a quick response to emerging scientific problems.

A recent example of employing Desktop Grids at the very early stages of solving urgent scientific problems is the fight against novel coronavirus disease at the beginning of 2020. The structure of the key SARS-CoV-2 spike protein (Sprot; key role in pathogenesis) was accurately predicted by the Rosetta@home project [15] several weeks before its description using cryo-electron microscopy [23].

Such an outrunning work allowed to speed up the research process that many academic groups performed those days: design of novel vaccines and antiviral drugs. The project also works on protein research in connection with other diseases and provides a web interface for submitting own jobs related to the drug search. It allows different scientists to employ computational resources for a total of 644 teraflops (as measured by the BOINC platform, [2]).

The Folding@home project is also carrying out the development of an antiviral agent against SARS-CoV-2. In early 2020, more than 700,000 new participants joined the project to find a coronavirus cure. As a result, the performance of Folding@home exceeded 2.4 exaflops, which made the project the first world's exascale system, more powerful than the Top500 supercomputers altogether [12].

The World Community Grid, another large-scale volunteer computing project, has also joined the search for a cure against the coronavirus. Apart from that, the project searches for cures against AIDS, cancer, malaria and other diseases. Scientific results to date include important advances in cancer treatment as well as non-medical discoveries. The computing resources of the project are about 565 teraflops [3].

It is natural that existing supercomputers have also been employed to fight against SARS-CoV-2 from the very early stages COVID-19 pandemic onset (see, e.g., [4] for a detailed overview). There are nation-wide and cross-nation research initiatives that provide scientists with supercomputer resources. To name a few, there is the world's leading supercomputer as of TOP-500 Fugaku (Japan), the COVID-19 High-Performance Computing Consortium (USA), the Exscalate4CoV project (EU), the Good Hope Net project (international), the ARCHER National Supercomputing Service (UK) and many others.

There have been, as well, supercomputers from non-scientific fields of applications that provided their resources to research projects. For instance, the La Liga football association (Spain) and Petrobras oil company (Brazil) directed parts of their supercomputers' capacities to the Folding@home project.

However, not every computationally-intensive problem is well designed for a supercomputing environment. The problems of bag-of-tasks type do not, as a rule, utilize the high-speed interconnection of supercomputer nodes. It causes under-utilization of supercomputer resources. At the same time, the resources are shared among multiple intensively computational applications. Setting and maintaining a supercomputer implies high costs. For these reasons, research groups (especially of a small/medium size) typically do not have immediate, on-demand access to supercomputing resources. Desktop Grid systems serve as a computational tool that can complement and when needed, substitute traditional high-performance systems.

When planning and organizing the workflow, time and budget of a research project, one needs to evaluate available computational resources. In the case of Desktop Grids, this evaluation is additionally complicated by such intrinsic features of Desktop Grids as heterogeneity and irregular availability of the computing nodes. In this paper, we aim to summarise and discuss the essential performance characteristics of a Desktop Grid on an example of a computational problem of large-scale molecular docking.

The remainder of the paper has the following structure. Section 2 introduces the BOINC middleware for Desktop Grids. In Section 3, we describe the drug discovery project COVID.SI. In Section 4, we report on the creation and first months of a BOINC-based extension of COVID.SI: SiDock@home. In Section 5, we analyze and discuss performance metrics. In Section 6, we provide directions for future work. Finally, Section 7 concludes the paper.

2 BOINC middleware

To organise and manage Desktop Grid-based distributed computations, a number of software platforms are used. The most popular platform among them is BOINC (Berkeley Open Infrastructure for Desktop Computing) [1]. Among the 157 active largest projects on volunteer computing, 89 are based on BOINC [10]; that is, BOINC can be considered a *de-facto* standard for the operation of volunteer computing projects. The BOINC platform is an actively developing Open Source software and provides rich functionality.

BOINC has a server-client architecture. The server generates a large number of tasks that are mutually independent parts of a computationally-intensive problem. When a client computer is idle, it requests work from the server, receives tasks, and independently processes them. Upon finishing, it reports results back to the server. The results are then stored in the database for further usage.

The owners of BOINC client computers independently determine their contribution to the computing process: the amount of resources provided and the time of their availability. Thus, they create uncertainty in the project's operation, particularly in the estimation of the project's performance. To unify the client computers and to optimise work distribution, BOINC implements a Whetstone benchmark [9]. It estimates the peak performance of a host basing on a set of floating-point operations. This benchmark considers only CPU capacity

and does not address the usage of disc, memory and other resources that a task may require. However, this basic estimate allows to roughly measure how many flops could have been done in a task runtime at a specified host and distribute new tasks accordingly.

3 The citizen science project COVID.SI

Since the beginning of the COVID-19 pandemics in late 2019-early 2020, many new and existing citizen science projects have attracted the public's attention at analyzing viral life-cycle and working on potential therapeutic targets. They have been based on well-established scientific methods and libraries, as well as novel and experimental ones. Due to the world-wide character of the COVID-19 pandemic and the public's high interest in citizen science, such projects gathered large amounts of traction and computational resources. It helped many research teams boost their research, developing drugs and vaccines, sharing the results with the scientific community and broadening the fundamental scientific disciplines.

In March 2020, a Slovenian research group led by Dr. Črtomir Podlipnik and Dr. Marko Jukić initiated a citizen science project "*Citizen science and the fight against the coronavirus*" (COVID.SI) [13] in the field of drug design and medicinal chemistry. The project is aimed at drug discovery, and first of all, against coronavirus infection, using highthroughput virtual screening (HTVS) [16]. To achieve the goal, the authors performed molecular docking using a library of ten million of small molecules against multiple potential therapeutic targets.

At the initial stage of the project, molecular docking was performed using an open-source software RxDock [19]. Later, the authors' team implemented an independent fork CmDock [7], aimed at the open-science approach, utilisation of graphics processing units (GPU) and introduction of modern docking analysis tools as well as improved docking algorithms.

The problem of the target-based virtual drug screening consists of a large number of mutually independent computational tasks (subsets of small molecules to study against the specified target), has the bag-of-tasks type and can be efficiently implemented in a Desktop Grid environment. At the same time, the scale of the proposed solution requires a significant amount of computational resources. These factors together made volunteer computing an efficient tool for the project like COVID.SI.

In the first stage of the project development, the authors implemented an original middleware for volunteer computing. Following the call for joining the computations, volunteers provided about 400 computers in the following ten months, and all together, they finished more than 60 projects; in other words, they performed 600 million independent docking experiments.

The workflow was established, the optimal docking parameters were calculated, and the first results were obtained and analyzed. Basing on the first results, the authors estimated the project's performance as more than a year to screen the new one billion library against one target, multiple-fold longer than

desired for an efficient drug discovery process aimed at biological evaluation of hit compounds. To complement and scale the available computational resources, we created a new SiDock@home, the BOINC-based extension of COVID.SI that organically grew into a sizable, independent and competent research project for general drug design. In the next section, we describe the project setup in more detail.

4 The project SiDock@home

The project SiDock@home [20] was created based on the BOINC middleware. The project's server part was deployed in an Ubuntu 18.04 LTS-based machine under system configuration of 2 Xeon 6140 cores, 8 Gb RAM, 32 Gb SSD and 512 Gb HDD. The molecular docking application CmDock [7] was adapted for execution in the BOINC environment using the wrapper program [24] for Windows, Linux and MacOS 64-bit operating systems.

The project was announced to the public on October 23, 2020. Since its beginning, the project has attracted the interest of the BOINC volunteer community. Consequently, in the testing phase of two months, 240 participants joined the project and provided the computational resources of more than 500 computers. Such a capacity allowed us to perform virtual screening (VS) on a part of the library, to elaborate on the optimal parameters of task distribution and results processing, and to evaluate the developed CmDock software in a heterogeneous distributed environment.

The project's workflow is constituted of mutually independent tasks, each of which performs molecular docking of 2 000 (this parameter can be tuned depending on the target) small molecules to a specified target. Such a division of the small molecules library allows us to have an average runtime of about 1-2 hours on a desktop computer, which follows the common practice of volunteer computing.

Input data of a task contains a package of small molecule models. Output data contains the docking results and the docking log. The latter is used to validate the results automatically and achieve a quorum. Basing on the results of the testing phase, we set the initial replication level to two instances of a task and the quorum to two results.

Such a setup allowed us to perform HTVS on the complete initial library for four targets in five months. In the next section, we discuss the performance metrics obtained based on gathered statistics of project operation.

5 Performance metrics of a Desktop Grid

5.1 Performance bounds

First of all, let us estimate the performance bounds of the considered Desktop Grid. Knowledge of performance bounds allows one to plan the server's capacity accordingly and evaluate the workflow scale for the next period of time. A deeper

analysis of the performance bounds dynamics may allow detecting irregularities in the development of the Desktop Grid, such as, for instance, an unusually high number of technical errors when attaching new computers.

In this work, we consider a combination of the four variants of the performance upper bound of the Desktop Grid:

1. P_{reg}, peak performance of all computers registered in the project up to the moment. This value is gradually increasing and reflects the total initial interest of the community in the project. More specifically, this characteristic allows us to estimate how efficient would the Desktop Grid be if every computer remained active and devoted to the project at its full capacity.
2. P_{credit}, peak performance of the computers registered in the project and awarded with any positive amount of BOINC credit. This value is strongly connected with P_{reg}, but restricts the summation to only those computers that have successfully completed at least one task of the project. The difference between the two values represents the amount of registered computational resources that could not be used for the project, most probably because of technical problems.
3. P_{active}, peak performance of the computers currently active and awarded with any positive amount of BOINC credit. This value apparently does not exceed P_{credit} and can increase or decrease in different periods of time. It reflects the total current interest of the community in the project.
4. P_{avail}, peak performance actually available to BOINC tasks of the project, taking into account availability periods of the computers and the user-imposed restrictions on resources usage by BOINC. Together with P_{active}, this value indicates the amount of computational resources the participants devote to the project. Overall, P_{avail} is the smallest value among the four and can be considered as the performance upper bound when planning workflow at a short distance.

To calculate the bounds values, we use the following host characteristics calculated by the BOINC middleware and stored in a database:

- `create_time`: timestamp of the moment the host registered to the project;
- `rpc_time`: timestamp of the last RPC of the host to the server;
- `total_credit`: the total BOINC credit earned by the host;
- `on_frac`: the fraction of total time the host runs the BOINC client;
- `active_frac`: of the time the host runs the BOINC client, the fraction it is enabled to use CPU;
- `p_ncpus`: the number of available CPU cores;
- `p_fpops`: estimate peak performance of a single CPU core, flops.

Note that the value `p_ncpus` can be edited by the host's owner. In practice, it is usually done to increase the number of received tasks. To estimate the performance bounds, we adjusted the apparent outliers. Specifically, we did not consider hosts with p_ncpus ≥ 1000 and manually corrected several values of typical configuration computers. The value `p_fpops` is estimated by the BOINC benchmark.

Let H be the set of database entries of all project's hosts. For each $h \in H$, let us denote h.field the corresponding value of the `field` at entry h. $I(x)$ is the indicator of an event x. Using the listed database fields, one can calculate the average performance at any specified interval of time $[t, t+1]$ as follows:

$$P_{reg}(t) = \sum_{h \in H} h.\text{p\_ncpus} \cdot h.\text{p\_fpops} \cdot I(h.\text{create\_time} \le t) \quad (1)$$

$$P_{credit}(t) = \sum_{h \in H} h.\text{p\_ncpus} \cdot h.\text{p\_fpops} \cdot \\ \cdot I(h.\text{create\_time} \le t) \cdot I(h.\text{total\_credit} > 0) \quad (2)$$

$$P_{active}(t) = \sum_{h \in H} h.\text{p\_ncpus} \cdot h.\text{p\_fpops} \cdot \\ \cdot I(h.\text{create\_time} \le t) \cdot I(h.\text{total\_credit} > 0) \cdot I(h.\text{rpc\_time} \ge t+1) \quad (3)$$

$$P_{avail}(t) = \sum_{h \in H} h.\text{p\_ncpus} \cdot h.\text{p\_fpops} \cdot \\ \cdot I(h.\text{create\_time} \le t) \cdot I(h.\text{total\_credit} > 0) \cdot I(h.\text{rpc\_time} \ge t+1) \cdot \\ \cdot h.\text{on\_frac} \cdot h.\text{active\_frac} \quad (4)$$

We discretise the time interval of the project's operation by days. In Fig. 1, we provide dynamics of performance bounds during the testing phase of the project. One can observe that, despite the community's rising interest in the project, the difference between potential and actual performance bounds gradually increases, which means the rate of computers leaving the project also increases. According to the practice of the project's operation and discussions on the project's website, the most common reasons for that are technical issues preventing the client computers from stable work on project's tasks, and the temporary loss of interest of the participants due to the testing character of the project's first stage.

We present the entire dynamics of performance bounds of the project in Fig. 2. The data shows that since the testing phase, the peak performance of all registered computers has shifted from tens to hundred of teraflops. Two time intervals of the fastest increase in P_{reg}, in the beginning of February 2021 and the middle of March 2021, correspond to the two competitions organized by the BOINC community. However, the increase in P_{credit} was smoother. P_{active} and P_{avail} increased only by a factor of two.

5.2 Load of a Desktop Grid

While the previously discussed peak performance is a theoretical estimate, it is also important to evaluate the actual performance attained by the project. In

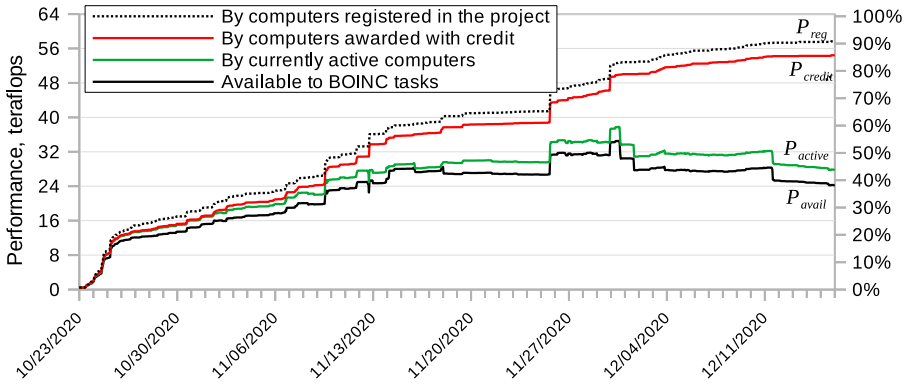

Fig. 1. Performance dynamics of the project's Desktop Grid in the testing phase.

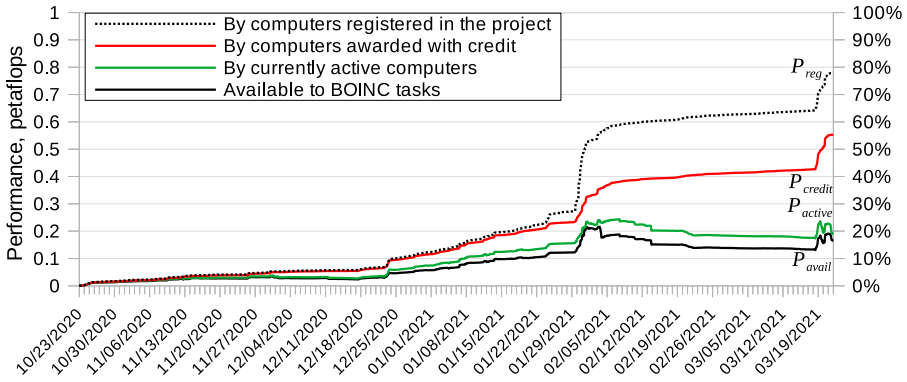

Fig. 2. Performance dynamics of the project's Desktop Grid in 5.5 months.

this subsection, we consider the actual load of the Desktop Grid by the project's task runtimes on host computers.

Given a BOINC task τ, let L be the estimated load this task imposed on a host computer. We propose to calculate it using the following result characteristics calculated by the BOINC middleware and stored in a database:

- `received_time`: timestamp of the moment the result was received by the server. We use it to discretise the workflow by days;
- `elapsed_time`: actual time of the task execution on the host computer;
- `flops_estimate`: estimated peak flops of the host computer.

The total load BOINC tasks imposed in a day t can be calculated as

$$L(t) = \sum_{\tau} \frac{\tau.\text{elapsed\_time}}{86\,400} \cdot \tau.\text{flops\_estimate}. \tag{5}$$

In Fig. 3, we provide the load dynamics of the project's Desktop Grid in comparison with the performance bounds obtained in the previous subsection.

We observe that the actual load varies in between 10–40% of the theoretical peak performance of active computers. This is partly explained by the background character of the Desktop Grid computations that, by definition, use only idle time of a computer. The value is also influenced by the presence of other BOINC projects at a host computer. Finally, the positivity of the load value corresponds to the normal operation of the Desktop Grid, while periods of zero value indicate the pauses in the server work or a temporary absence of tasks.

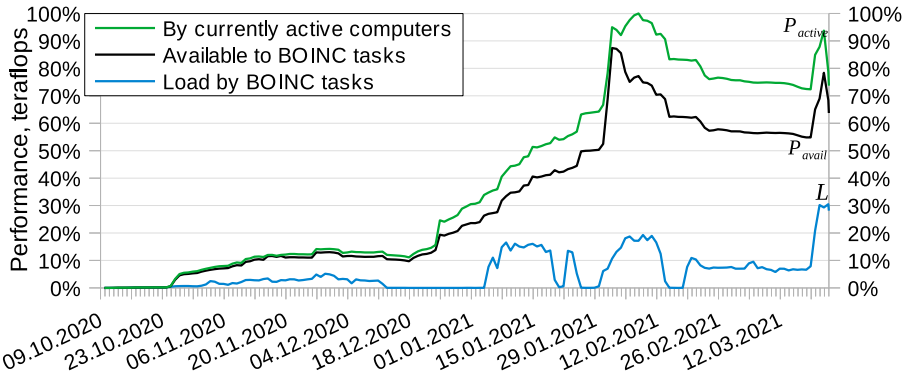

Fig. 3. Load dynamics of the project's Desktop Grid.

5.3 Number of active threads in a Desktop Grid

The project's performance can also be estimated in terms of active threads: the amount of work in CPU days per day of a Desktop Grid operation. The estimated number of active threads in a day t can be calculated as

$$T(t) = \sum_{\tau} \frac{\tau.\text{elapsed\_time}}{86\,400}. \tag{6}$$

In general, the actual performance strongly depends on the specifics of the solved computationally-intensive problem. For this reason, it is complicated to compare Desktop Grids and traditional high-performance computing systems. However, the number of active threads allows us to find the first approximation of a Desktop Grid's scale.

Fig. 4 presents the dynamics of $T(t)$. The average number of active threads during non-zero workload periods is 3 766, comparable with supercomputers in the last 10% of Top-50 supercomputers of Russia and CIS [22]. The practice has shown that for short periods of high load, such as BOINC competitions, $T(t)$ value reaches 18 208, comparable with a significantly more powerful supercomputer available to us on demand.

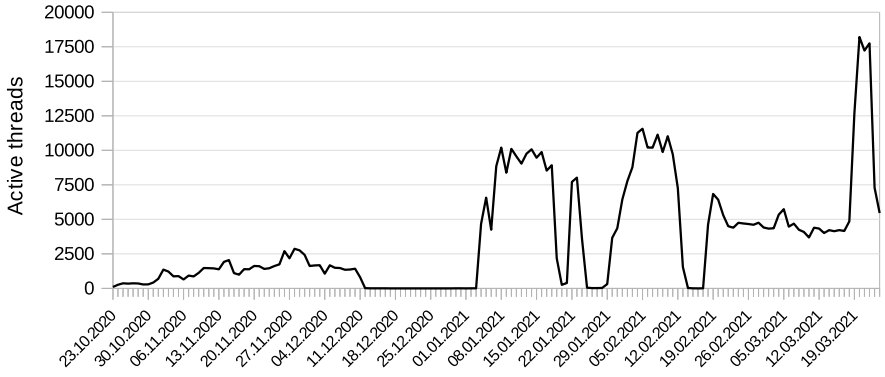

Fig. 4. Dynamics of the number of active threads.

5.4 Credit-awarded work of a Desktop Grid

In the previous subsections, we considered the Desktop Grid's total load imposed by all tasks with a positive runtime. However, some of the tasks end up with errors of different types or may not be needed because, at the moment of their return to the server, the quorum has been already met. Now, let us consider only the tasks with a positive credit, or, in other words, the tasks that performed efficient work according to the logic of the solved problem.

BOINC maintains the record of the performed work in the form of credit [8] which, in the most common case, is calculated as follows. Each host $h \in H$ is assigned a value a_h, a peak performance of its CPU flops, estimated with an internal BOINC benchmark. When a task τ has been executed on host h, BOINC registers the elapsed time $T_{h\tau}$. The amount of credit the host would get for the task is $C_{h\tau} = T_{h\tau} \cdot a_h \cdot CS$. If the result passes a validity check on the server and the quorum has been met, the host is awarded $C_{h\tau}$ (or an appropriately adjusted value if quorum exceeds 1).

Here, $CS = \frac{200}{86\,400} \times 10^9$ *(a Cobblestone)* is a constant unifying the effective work of heterogeneous computers with a reference one that would do one gigaflop/s based on the Whetstone benchmark and receive 200 credits a day.

To compare the scale of the Desktop Grid with traditional high-performance systems, one may need to adapt an existing benchmark such as LINPACK [11]. Let us illustrate a variant of such adaptation on an example of a sample desktop computer available 24/7, AMD Ryzen 9 3900X 12-Core Processor, earning 22 500 BOINC credits a day and producing 500 gigaflops/s according to the LINPACK benchmark.

In Fig. 5, we provide the dynamics of the total credit-awarded work of the project SiDock@home in teraflops estimated by the internal BOINC benchmark and by the LINPACK benchmark of the sample computer. Again, as in Subsection 5.3, the average performance of the considered Desktop Grid is comparable with the 10% tail of Top-50 supercomputers of Russia and CIS.

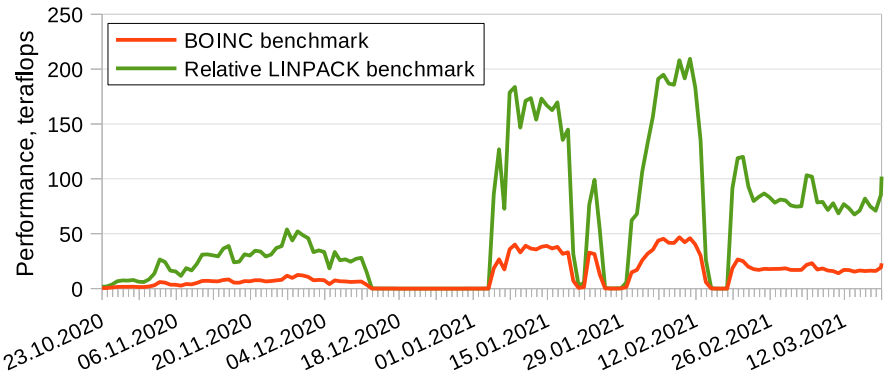

Fig. 5. Credit-awarded work dynamics of the project's Desktop Grid.

We observe that the performance estimates by the two benchmarks differ several-fold. Additional work is required to appropriately adapt the LINPACK benchmark for evaluating the performance of a Desktop Grid in comparison with supercomputers.

6 Future work

In the future, we aim to work on the research of Desktop Grids' performance in three main directions.

Firstly, we will develop and implement a prognosis system for a Desktop Grid. Such a system will estimate the time to complete the project or a separate computational experiment considering the performance dynamics. A possible application of such a prognosis is the long-term planning of research.

Secondly, we will develop new methods for optimising a Desktop Grid utility, targeting the fastest discovery of practically valuable results such as, for example, chemically diverse hits of virtual screening [17].

Thirdly, we will implement the developed methods in the long-running drug discovery project SiDock@home to increase its utility. We will also share the results in the form of open-source software.

7 Conclusion

In general, it is complicated to compare Desktop Grids and traditional high-performance computing systems. The actual performance of any system strongly depends on the specifics of the solved computationally-intensive problem; moreover, Desktop Grids' peculiar features impose additional difficulties. In this work, we propose several performance metrics for Desktop Grids. We formalize them for a BOINC-based volunteer computing project, illustrate them on the statistics of the project SiDock@home and discuss their practical application scope.

The results show that the actual efficient load of a Desktop Grid of a volunteer computing project is several-fold lower than its potential performance. It is partly explained by the background character of the Desktop Grid computations, but yet leaves open the problem of optimizing the workload.

The performance bounds based on BOINC benchmarks represent the relative scale of the BOINC project among the others regarding community involvement and theoretical peak performance. Such metrics may be helpful when planning the further development of a project. For instance, one may decide between joining an existing umbrella project or running a separate one, extending the server hardware base, targeting new client platforms or processor types.

The proposed metrics can be used for a detailed investigation of the performance and efficiency of a Desktop Grid. We have discussed the possible applications of the metrics in a Desktop Grid-based project of any scale.

Analysis of the Desktop Grid-based project SiDock@home has shown the applicability of the proposed metrics and their combinations for quantitative and qualitative conclusions about the project's performance.

Funding

This work was partly supported by the Slovenian Ministry of Science and Education infrastructure project grant HPC-RIVR and by the Slovenian Research Agency (ARRS) programme P2-0046.

Acknowledgements

The first initial library (one billion of compounds) was prepared with the generous help of Microsoft that donated computational resources in the Azure cloud platform [6]. We all from COVID.SI are grateful and looking forward to future collaborations.

We wholeheartedly thank all BOINC participants for their contributions.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Anderson, D.P.: BOINC: a platform for volunteer computing. Journal of Grid Computing pp. 1–24 (2019)
2. BOINCstats. rosetta@home. https://www.boincstats.com/stats/14/project/detail (2020), [Online; accessed 25-March-2021]
3. BOINCstats. World Community Grid. https://www.boincstats.com/stats/15/project/detail (2020), [Online; accessed 25-March-2021]

4. By Editorial Team: The history of supercomputing vs. COVID-19. https://www.hpcwire.com/2021/03/09/the-history-of-supercomputing-vs-covid-19 (2021), [Online; accessed 18-March-2021]
5. climateprediction.net — the world's largest climate modelling experiment for the 21st century. https://www.climateprediction.net (2020), [Online; accessed 25-March-2021]
6. Cloud Computing Services — Microsoft Azure. https://azure.microsoft.com/en-us/ (2020), [Online; accessed 25-March-2021]
7. CmDock. https://gitlab.com/Jukic/cmdock (2020), [Online; accessed 25-March-2021]
8. CreditNew – BOINC. https://boinc.berkeley.edu/trac/wiki/CreditNew (2020), [Online; accessed 25-March-2021]
9. Curnow, H.J., Wichmann, B.A.: A synthetic benchmark. The Computer Journal **19**(1), 43–49 (1976)
10. Distributed Computing – Computing Platforms. http://distributedcomputing.info/platforms.html (2020), [Online; accessed 25-March-2021]
11. Dongarra, J.J., Luszczek, P., Petitet, A.: The LINPACK benchmark: past, present and future. Concurrency and Computation: Practice and Experience **15**(9), 803–820 (2003). https://doi.org/https://doi.org/10.1002/cpe.728, https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.728
12. Folding@home – fighting disease with a world wide distributed super computer. https://foldingathome.org (2020), [Online; accessed 25-March-2021]
13. Home – COVID.SI. https://covid.si/en (2020), [Online; accessed 25-March-2021]
14. home — lhc@home. https://lhcathome.web.cern.ch (2020), [Online; accessed 25-March-2021]
15. Institute for Protein design, University of Washington: Rosetta's role in fighting coronavirus. https://www.ipd.uw.edu/2020/02/rosettas-role-in-fighting-coronavirus (2020), [Online; accessed 25-March-2021]
16. Jukič, M., Janežič, D., Bren, U.: Ensemble docking coupled to linear interaction energy calculations for identification of coronavirus main protease (3clpro) non-covalent small-molecule inhibitors. Molecules **25**(24), 5808 (2020)
17. Nikitina, N., Ivashko, E., Tchernykh, A.: Congestion game scheduling for virtual drug screening optimization. Journal of computer-aided molecular design **32**(2), 363–374 (2018)
18. Rosetta@home. https://boinc.bakerlab.org (2020), [Online; accessed 25-March-2021]
19. Ruiz-Carmona, S., Alvarez-Garcia, D., Foloppe, N., Garmendia-Doval, A.B., Juhos, S., Schmidtke, P., Barril, X., Hubbard, R.E., Morley, S.D.: rDock: A fast, versatile and open source program for docking ligands to proteins and nucleic acids. PLoS computational biology **10**(4), e1003571 (2014)
20. SiDock@home. https://sidock.si/sidock (2020), [Online; accessed 25-March-2021]
21. Together we are powerful – folding@home. https://foldingathome.org (2020), [Online; accessed 25-March-2021]
22. Top50 — Supercomputers [in Russian]. http://top50.supercomputers.ru/list (2020), [Online; accessed 25-March-2021]
23. Wrapp, D., Wang, N., Corbett, K.S., Goldsmith, J.A., Hsieh, C.L., Abiona, O., Graham, B.S., McLellan, J.S.: Cryo-em structure of the 2019-ncov spike in the prefusion conformation. Science **367**(6483), 1260–1263 (2020)
24. WrapperApp – BOINC. https://boinc.berkeley.edu/trac/wiki/WrapperApp (2020), [Online; accessed 25-March-2021]