Batch of Tasks Completion Time Estimation in a Desktop Grid

Evgeny Ivashko[0000−0001−9194−3976] (✉) and Valentina Litovchenko

[1] Institute of Applied Mathematical Research,
Karelian Research Centre of Russian Academy of Sciences, Petrozavodsk, Russia
[2] Petrozavodsk State University, Petrozavodsk, Russia
ivashko@krc.karelia.ru, va.lentina97@yandex.ru

Abstract. This paper describes a statistical approach used to estimate batch of tasks completion time in a Desktop Grid. The statistical approach based on Holt model is presented. The results of numerical experiments based on statistics of RakeSearch and LHC@home volunteer computing projects are given.

Keywords: Desktop Grid · BOINC · high-throughput computing · Holt model · confidence interval · completion time estimation.

1 Introduction and Related Works

Along with computing clusters and Grid systems, Desktop Grid systems keep their valuable place in high-performance computing infrastructure. Desktop Grid is a form of distributed high-throughput computing system which uses idle time of non-dedicated geographically distributed computing nodes connected over a low-speed (Internet or LAN) network. In general, Desktop Grids have a server-client architecture. Such the systems have huge computing potential exceeding 1 ExaFLOPS [20].

Desktop Grids have a number of peculiarities comparing to other computational systems:

- slow data transfer between server and a node (comparing to computing clusters and Grid systems); in general, no direct data transfer between nodes is allowed;
- limited computational capacity of separate nodes;
- high heterogeneity of software, hardware, or both;
- no information on a node state or completion level of a task is available;
- low reliability of computing nodes;
- dynamic set of nodes;
- lack of trust to computing nodes.

Because of these peculiarities Desktop Grids are significantly differ from computing clusters, Grid systems or cloud computing systems. So, it is necessary to develop special algorithms aimed at solving specific problems of Desktop Grid

management. One of these problems is batch of tasks completion time estimation. Many of Desktop Grid computational projects perform separate computational experiments, where each of the experiments consists of a number of tasks (batch of tasks).

Batch of tasks completion time estimation is a complicated and many-sided problem, caused by the intermittent resource availability. This problem relates to a single task or a batch of tasks completion time estimation problem as a part of a more complex scheduling problem.

For example, the paper [5] describes an approach to scheduling thousands of jobs with diverse requirements to heterogeneous grid resources, which include volunteer computers running BOINC software. A key component of this system provides a priori runtime estimation using machine learning with random forests.

Papers [12] and [13] deal with a novel solution for predicting the runtimes of parameter sweep jobs. In the infrastructure with a dynamic configuration, such as a Desktop Grid, resource uptime and application runtime are critical scheduling factors. The proposed scheduling mechanism maps job runtimes onto resource uptimes. It is based on runtime prediction and its application in the prediction-aware mapping of jobs onto available resources. The aim is to reduce the number of jobs prematurely interrupted due to insufficient resource availability. The parameter sweep prediction framework used to make the predictions is referred to as GIPSy (Grid Information Prediction System). A detailed comparison between the expected results, based on simulation analysis, and the final results is given. By comparing the results for different model building configurations an optimal configuration is found that produces reliable result independent of the chosen job type. Results are presented for a quantum physics problem and two simulated workloads represented by sleepjobs.

In the paper [11] authors try to estimate delays, which are part of the task lifespan, i.e., distribution, in-progress, and validation delays. In the paper, the authors evaluate the accuracy of several probabilistic methods to get the upper time bounds of these delays. An accurate prediction of job lifespan delays allows to make more accurate prediction of a batch of tasks time completion, provide more efficient resources use, higher project throughput, and lower job latency in Desktop Grid projects.

In the paper [15] a dynamic replication approach is proposed to reduce the time needed to complete a batch of tasks; a mathematical model and a strategy of dynamic replication at the tail stage are proposed.

In the paper [19] the authors formulate an analytical model that permits to compare different allocation policies. In particular, the authors study an allocation policy that aims at minimizing the average job completion time; it is shown that the proposed policy can reduce the average completion time by as much as 50% of the completion time required for uniform or linear allocation policies. In the paper [17] a Desktop Grid is dynamically augmented with an Infrastructure as a Service (IaaS) cloud to reduce the average batch of tasks completion time.

Batch of tasks completion time estimation can be performed using various Desktop Grid emulators and simulators. Anderson [3] proposes a BOINC client

emulator which is focused on scheduling strategies testing. The proposed emulator allows to trial different scheduling strategies in various usage scenarios, varying different hardware characteristics, defining client availability patterns, etc. The BOINC scheduling policies can be evaluated by several performance metrics. The results of emulations show which scheduling strategies are the most efficient. A part of chapter [8] is devoted to review and description of several Grid and Desktop Grid simulation tools: SimBA, SimBOINC, and EmBOINC software. The paper [6] also consider simulation of various computing systems focusing on the SimGrid simulator. The papers [9] and [10] by Estrada *et al.* are devoted to Desktop Grid emulation by EmBOINC, also paying attention to several simulation tools. The paper [18] describes the BOINC simulator SimBA.

This paper proposes statistical approach to batch of tasks completion time estimation. It is based on Holt's model and confidence intervals construction. The rest part of the paper is organized as follows. Section 2 briefly describes Desktop Grid and BOINC workflow. Section 3 describes mathematical model including formal problem definition, Holt's statistical model and confidence interval construction. Section 4 presents results of the numerical experiments. Finally, section 5 gives the final discussion.

2 Desktop Grid and BOINC

A Desktop Grid consists of a (large) number of computing nodes and a server which distributes tasks among the nodes. The workflow is as follows. A node asks the server for work; the server replies sending one or more tasks to the node. The node performs calculations and when it finishes it, it sends the result (which is a solution of a task or an error report) back to the server. The server collects the results, assimilates and validates them. The detailed description of the computational process in BOINC-based Desktop Grids is given in [1].

There is a number of middleware systems for Desktop Grid implementation, the paper [14] authors overviews and compares the most popular of them: BOINC, XtremWeb, OurGrid and HTCondor. They review scientific papers devoted to research of the factors that influence the performance of Desktop Grid (from the point of view of both server and the client).

The most popular Desktop Grid platform is BOINC (Berkeley Open Infrastructure for Network Computing). It is actively developing Open Source software with rich functionality; it is a universal platform for scientific project in different domains. Herewith, BOINC is simple in deployment, use, and management [2].

BOINC is based on server-client model. The client software exists for different hardware and software platforms. The server can have arbitrary number of clients. A server can host several different BOINC-projects; each client can be connected to several projects and servers. BOINC-project is identified by its URL. BOINC client can be flexibly tuned up to be unobtrusive to the computer owner.

3 Completion Time Estimation

A statistical approach to batch of tasks completion time estimation is based on studying of time series characteristics, describing the process of results retrieving. A time series is a series of data points listed in time order. More formal, a discrete-time time series is a set of observations x_t, each one being recorded at a specific time t; the set T of times at which observations are made is a discrete set.

3.1 Mathematical Model

We consider a BOINC-based Desktop Grid, consisting of a number of computing nodes. The configuration of the Desktop Grid is dynamic: the set of available nodes can change in time (the nodes can abandon the Desktop Grid and new nodes can appear), the nodes have different characteristics of availability and reliability, tasks can be lost because of missed deadlines or abandoned nodes. A computational experiment of N tasks takes place; we need to construct a forecast on completion time of the computational experiment.

To make a forecast one should determine a functional dependence reflecting to time series. This functional dependence is called a forecast model. The model should minimize absolute difference between forecasted and observed values for a specified horizon (look-ahead period). Based on forecast model, one should find out forecasted values and confidence interval.

Consider a cumulative process of results retrieving. This process is described by a time series

$$Z(t) = Z(t_1), Z(t_2), ..., Z(t_k). \tag{1}$$

The values of the process are observed at discrete time points $t = t_1 < t_2 < ... < t_k$ with non-uniform intervals between them. Note, that the process is steadily increasing, so $Z(t+1) > Z(t)\ \forall t$. At the point t_k (forecast point) one should estimate a time point t_p, at which observed value $Z(t_p)$ will exceed a specified value A (see fig. 1).

For convenience, turn to considering a process

$$Y_i = (Z(t))^{-1}, i = 1, ..., k, \tag{2}$$

which describes time points of i-th result receiving. Then at step k (forecast point) one should estimate the value of process Y_i at step A ($A > k$; $p = A - k$ – look-ahead period, see. fig. 2).

With a view to forecast, take the following assumptions. First, assume that there is a functional dependence between previous and future values of the process:

$$Z(t) = F\left(Z(t-1), Z(t-2), Z(t-3), ...\right) + \epsilon_t, \tag{3}$$

here ϵ_t – is a random error with a normal law of distribution. Second, this dependence is piecewise linear with up trend.

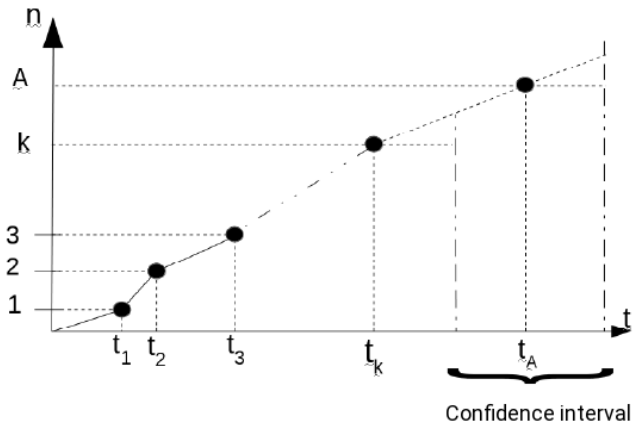

Fig. 1: Time series.

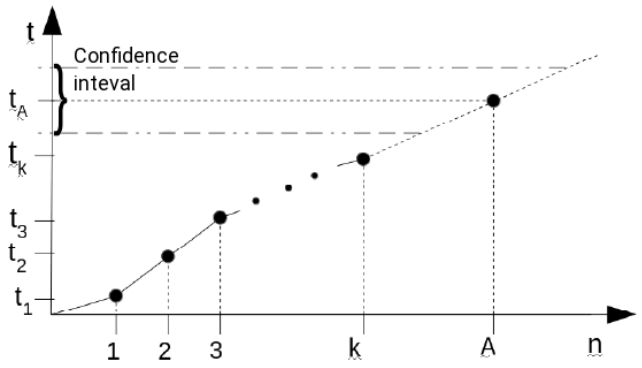

Fig. 2: Forecasted process.

From the point of view of Desktop Grid these assumptions mean the following. The observed process describes time points of new results receiving. So, it is strictly increasing (two results can not be received at the same moment). An angle of trend line describes performance of the Desktop Grid (the less the angle to x axis, the more performance has the computing system). The performance can vary, according to it changes the trend. We assume that change in performance is linear because non-linear changes usually related to BOINC-project start, computing competitions and so on. Such non-linear effects are limited to a transition period and subside quickly.

In this paper we consider Holt's linear trend method based on exponential smoothing as forecasting statistical model.

3.2 Forecasting Model

Exponential smoothing is a rule of thumb technique for smoothing time series data using the exponential window function with decreasing weights over time. Holt's forecasting model is extended simple exponential smoothing to allow forecasting of data with a trend. This model involves a forecast equation and two smoothing equations (one for the level and one for the trend) [7]:

$$\begin{aligned} Z_t &= L_t + \epsilon_t; \\ L_t &= \alpha Z_{t-1} + (1-\alpha)(L_{t-1} - T_{t-1}); \\ T_t &= \beta(L_{t-1} - L_{t-2}) + (1-\beta)T_{t-1}. \end{aligned} \tag{4}$$

Here

- Z_i – observed level value;
- L_i – smoothed level value;
- T_i – trend value,
- $0 \le \alpha, \beta \le 1$ – smoothing coefficients for level and trend accordingly.

Forecast on p steps is constructing with an assumption of keeping the trend by the following formula:

$$\overline{Z}_{t+p} = L_t + pT_t. \tag{5}$$

Forecast is linear and relies on the current trend.

3.3 Confidence Interval

Having right statistical model and keeping the trend, observed values and extrapolated point forecast are mismatching due to

1. inexact parameters of the model;
2. random error ϵ_t.

These errors can be shown as a forecast confidence interval. The formula to calculate a forecast confidence interval is following (see [7]):

$$\widehat{y}_{k+p} \pm t_\gamma \cdot S_y \cdot \sqrt{\frac{k+1}{2} + \frac{(k+p-\bar{t})^2}{\sum_{t=1}^{k}(t-\bar{t})^2}}, \tag{6}$$

here

- y_{k+p} – point forecast at the moment $k+p$, where k – number of observed values and p – look-ahead period;
- t_γ – value of Student's t-statistics;
- S_y^2 – mean-square distance between observed and forecasted values;
- $t = 1, 2, \cdots, k$ – process steps;
- $\bar{t} = \frac{k+1}{2}$ – mean step.

Mean-square distance between observed and forecasted values is defined by the following formula:

$$S_y^2 = \frac{\sum_{t=1}^{k}(y_t - \widehat{y}_t)^2}{k-1}, \quad (7)$$

where

- y_t – observed values;
- $\widehat{y}_t$ – forecasted values;
- k – number of values.

Therefore, confidence interval width depends on confidence level, look-ahead period, number of values and mean-square distance between observed and forecasted values.

4 Experiments Analysis

We performed set of numerical experiments to assess the approach. We used statistics of RakeSearch [16] and LHC@home [4] BOINC-project as input data. The available data of RakeSearch project contain information on workunit/result/host ids, results create/sent/received times, outcome, elapsed/cpu times; available data of LHC@home project contain information on workunit id and create time, results create/sent/received times, elapsed/cpu times. The summary on the number of records and time periods covering by input data are given in table 1. We also mention that the records of RakeSearch project data are generated by 1081 computing nodes (we do not have the same information for LHC@home project).

Table 1: Input data characterization.

	RakeSearch	LHC@home
Time period	06/09/2017 - 14/11/2017	13/07/2017 – 30/08/2017
Number of records	117 579	41 797

For the purposes of this research, the input data sets were reduced to timestamps of results receiving. The data sets are depicted on figures 3 and 4.

To estimate batch of tasks completion time we considered sequences of tasks with random length (from 100 to 1000 tasks) with random look-ahead periods (from 10 to 50 tasks). The experiments show good covering of real data by confidence intervals. The examples of batch of tasks completion time estimation (based on newer results of RakeSearch project) are given in figure 5 and 6.

In the figures the blue line shows results receiving time, solid line is an approximation according to Holt's model, arrow is the point forecast and the red line at the arrow is a confidence interval.

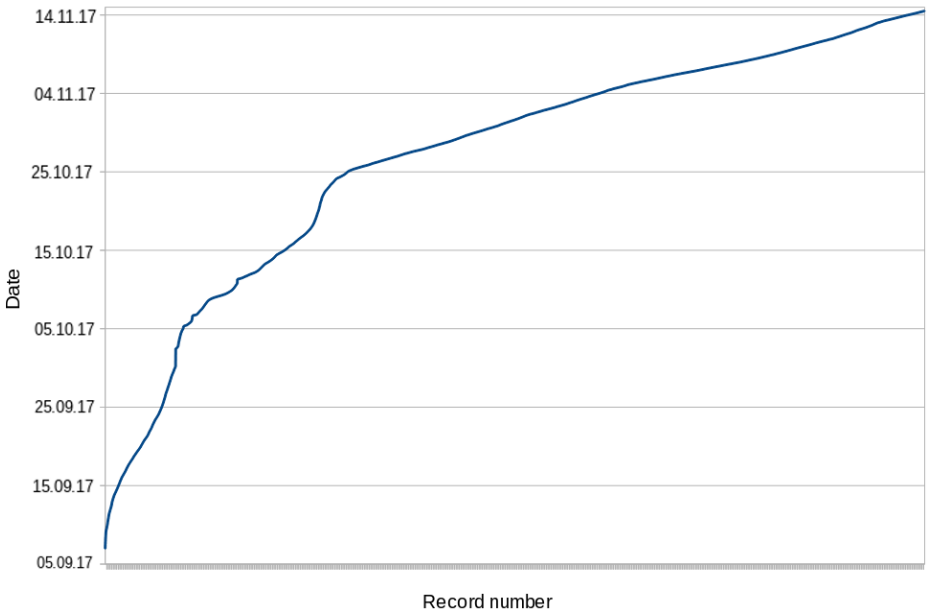

Fig. 3: Results receiving times of RakeSearch project.

5 Discussion and Conclusion

The paper presents statistical approach to a batch of tasks completion time estimation in a Desktop Grid. We used Holt's forecasting model, which is extended simple exponential smoothing with a trend.

We performed a number of numerical experiments on statistics of the BOINC-projects RakeSearch and LHC@home. The results of the experiments show good approximation of a point forecast to a real value at the same step. But in practice it is more important to have confidence interval, than a point forecast. So, we use confidence intervals also; the experiments show good covering of real data by confidence intervals. This shows that despite of high heterogeneity and big number of computing nodes (recall, that we used the statistics of RakeSearch project generated by 1081 computing nodes) and different tasks complexity, it is possible to get useful runtime estimation using simple statistical models.

There is a number of aspects that should be taken into account while implementing the described approach. First, approximation accuracy of Holt's model depends on two parameters: smoothing coefficients for level and trend. A procedure to tune-up the parameters is described, for example, in [7].

Second, in some cases it is reasonable to scale up time series when a long look-ahead period is given. Scaling up means summarizing several values of the time series into one single, describing the results received in the same hour, day or week. This allows to reduce computational overhead and increase accuracy

Fig. 4: Results receiving times of LHC@home project.

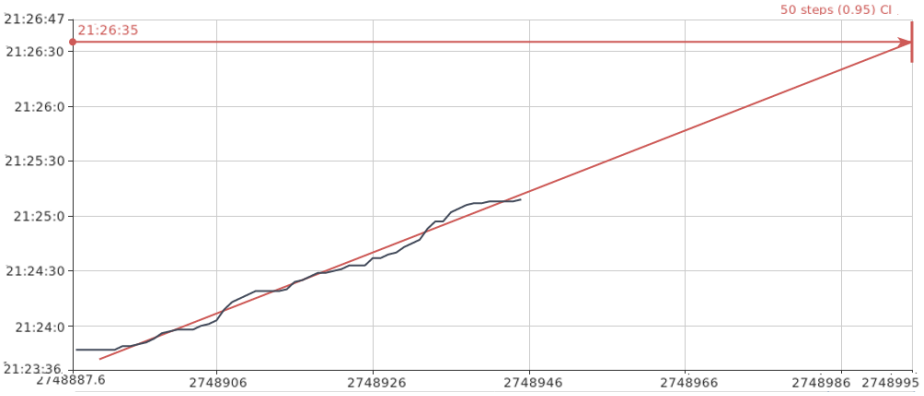

Fig. 5: Forecasting with $p = 50$ and confidence interval 0.95.

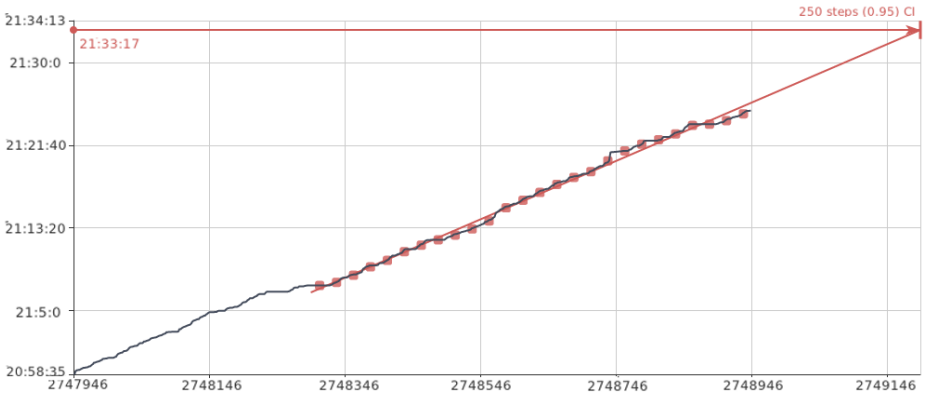

Fig. 6: Forecasting with $p = 250$, confidence interval 0.95.

while looking ahead long periods (thousands, tens of thousands steps) in large Desktop Grid projects.

Third, not all the values of the time series should be considered to perform forecasting. Only k latest values should be considered. The value k is selected to characterize the current trend and to minimize width of confidence interval, balancing mean-square distance between observed and forecasted values (which is increasing with higher k, in general) and Student's t-statistics (which is decreasing with higher k) according to formula 6.

Finally, fourth, in practice one is needed on-going update of the forecasted values according to new received observations. So, the model should be recalculated with every new observed value.

It also should be noted, that in this work computational complexity (size) and computing nodes performance are not considered. If in a sequence of tasks there is a subsequence in which all the tasks are more (or less) computationally complex than others, the statistical model will have a systematic error until this subsequence will be completed. Also, in practice it should be reasonable to take into account some kind of periodicity (for example, daily or weekly computing nodes periodicity).

All the mentioned above aspects will be taken into account to implement a BOINC-module of statistical batch of tasks completion estimation.

Acknowledgements

This work was supported by the Russian Foundation of Basic Research, projects 18-07-00628, 18-37-00094 and 16-47-100168.

References

1. Anderson, D.P., Christensen, C., Allen, B.: Designing a runtime system for volunteer computing. In: SC 2006 Conference, Proceedings of the ACM/IEEE. pp.

33–33 (Nov 2006). https://doi.org/10.1109/SC.2006.24
2. Anderson, D.P.: BOINC: A system for public-resource computing and storage. In: Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on. pp. 4–10. IEEE (2004)
3. Anderson, D.: Emulating volunteer computing scheduling policies. In: Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on. pp. 1839–1846. IEEE (2011)
4. Barranco, J., C.Y.C.D.e.a.: Lhc@home: a boinc-based volunteer computing infrastructure for physics studies at cern. Open Engineering **7(1)**, 379–393 (2017). https://doi.org/10.1515/eng-2017-0042
5. Bazinet, A.L., Cummings, M.P.: Computing the tree of life: Leveraging the power of desktop and service grids. In: 2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum. pp. 1896–1902 (May 2011). https://doi.org/10.1109/IPDPS.2011.344
6. Beaumont, O., Bobelin, L., Casanova, H., Clauss, P.N., Donassolo, B., Eyraud-Dubois, L., Genaud, S., Hunold, S., Legrand, A., Quinson, M., others: Towards Scalable, Accurate, and Usable Simulations of Distributed Applications and Systems. Research report rr-7761, Institut National de Recherche en Informatique et en Automatique (2011)
7. Brockwell, P.J., Davis., R.A.: Introduction to time series and forecasting. Springer-Verlag New York, Inc., second edn. (2002)
8. Estrada, T., Taufer, M.: Challenges in designing scheduling policies in volunteer computing. In: Cérin, C., Fedak, G. (eds.) Desktop Grid Computing, pp. 167–190. CRC Press (2012)
9. Estrada, T., Taufer, M., Anderson, D.: Performance Prediction and Analysis of BOINC Projects: An Empirical Study with EmBOINC. Journal of Grid Computing **7**(4), 537–554 (Aug 2009), 00017
10. Estrada, T., Taufer, M., Reed, K., Anderson, D.: EmBOINC: An emulator for performance analysis of BOINC projects. In: Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on. pp. 1–8. IEEE (2009)
11. Estrada, T., Taufer, M., Reed, K.: Modeling job lifespan delays in volunteer computing projects. In: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid. pp. 331–338. CCGRID '09, IEEE Computer Society, Washington, DC, USA (2009). https://doi.org/10.1109/CCGRID.2009.69, http://dx.doi.org/10.1109/CCGRID.2009.69
12. Hellinckx, P., Verboven, S., Arickx, F., Broeckhove, J.: Predicting parameter sweep jobs: From simulation to grid implementation. In: 2009 International Conference on Complex, Intelligent and Software Intensive Systems. pp. 402–408 (March 2009). https://doi.org/10.1109/CISIS.2009.86
13. Hellinckx, P., Verboven, S., Arickx, F., Broeckhove, J.: Runtime Prediction in Desktop Grid Scheduling, vol. 1 (01 2009)
14. Khan, M., Mahmood, T., Hyder, S.: Scheduling in desktop grid systems: Theoretical evaluation of policies and frameworks. International Journal of Advanced Computer Science and Applications **8**(1), 119–127 (2017)
15. Kolokoltsev, Y., Ivashko, E., Gershenson, C.: Improving "tail" computations in a boinc-based desktop grid. Open Engineering **7**(1), 371–378 (2017)
16. Manzyuk, M., Nikitina, N., Vatutin, E.: Employment of distributed computing to search and explore orthogonal diagonal latin squares of rank 9. In: In: Proceedings of the XI All-Russian research and practice conference "Digital techologies in education, science, society" (2017)

17. Reynolds, C.J., Winter, S., Terstyanszky, G.Z., Kiss, T., Greenwell, P., Acs, S., Kacsuk, P.: Scientific workflow makespan reduction through cloud augmented desktop grids. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science. pp. 18–23 (Nov 2011). https://doi.org/10.1109/CloudCom.2011.13
18. Taufer, M., Kerstens, A., Estrada, T., Flores, D., Teller, P.: SimBA: A discrete event simulator for performance prediction of volunteer computing projects. In: Principles of Advanced and Distributed Simulation, 21st International Workshop on. vol. 7, pp. 189–197 (2007)
19. Villela, D.: Minimizing the average completion time for concurrent grid applications. Journal of Grid Computing **8**(1), 47–59 (Mar 2010). https://doi.org/10.1007/s10723-009-9119-2, https://doi.org/10.1007/s10723-009-9119-2
20. Wu, W., Chen, G., Kan, W., Anderson, D., Grey, F.: Harness public computing resources for protein structure prediction computing [c]. In: The International Symposium on Grids and Clouds (ISGC). vol. 2013 (2013)