Performance Modeling and Prediction of Real Application Workload in a Volunteer-based System

PAWEL CZARNUL
pczarnul@eti.pg.gda.pl

MARIUSZ MATUSZEK
mrm@eti.pg.gda.pl

Gdansk University of Technology
Department of Computer Architecture, Faculty of ETI
11/12 Narutowicza Street, 80-233 Gdansk, Poland

Abstract: The goal of this paper is to present a model that predicts the real workload placed on a volunteer based system by an application, with incorporation of not only performance but also availability of volunteers. The application consists of multiple data packets that need to be processed. Knowing the computational workload demand of a single data packet we show how to estimate the application workload in a volunteer based system. Furthermore, we show how real life metrics obtained from the BOINC volunteer system can be applied to the proposed model.

Key–Words: volunteer computing, parallel computations, prediction of application execution workload, reliability, BOINC

1 Introduction

Parallel computing has become widespread with a wide availability of computing devices among consumers. This ranges from HPC clusters such as Tianhe-2 with over 3 million cores and 37 PFlop/s performance, through volunteer computing based systems such as BOINC and Comcute [1], parallel processing within application servers such as Java EE, multiple servers and client endpoints on Internet based applications up to multithreaded processing on modern CPUs and accelerators such as GPUs or Intel Xeon Phi.

In terms of large computational power, two types of systems have emerged as alternatives with particular advantages and shortcomings as shown in Table 1:

1. cluster systems – for tightly coupled parallel applications with possibly dense communication, that can be realized by low latency high bandwidth interconnects such as Infiniband,

2. volunteer based systems – for low cost computing, best suited for either embarrassingly parallel codes or simulations of disjoint and multiple test cases on multiple machines of possibly geographically distributed volunteers.

The goal of this paper is to present an analytical model that predicts the expected workload of an application in a volunteer based system. In a controlled cluster environment, the execution workload correlates strongly with time and would be expressed mainly by processing times of individual data packets and communication. In a volunteer based system we take into account also availability of volunteers and potential failures of data processing that requires redistribution and rerunning of data processing.

The work is organized as follows. Section 2 presents related work and motivations of this work compared to the existing results. Section 3.1 outlines the proposed analytical model with discussing distribution of volunteer performance and availability. Section 3.2 presents application of BOINC data to the analytical model with prediction of real application workload in Section 3.3. Finally, conclusions and future work are presented in Section 4.

2 Related Work

Building HPC cluster-based systems with fast networks such as Infiniband is suitable for demanding applications [2] with dense communication such as modeling physical phenomena [3], numerical computations [4], image processing [5] etc. Drawbacks of clusters include high costs of purchase, very large running and maintenance cost (power consumption of

ISBN: 978-960-474-381-0

Table 1: Cluster vs volunteer computing

Feature	Cluster Computing	Volunteer Computing
best suited applications	demanding compute-intensive applications	mainly embarrassingly parallel applications/ without dense communication
participating parties	dedicated users	independent volunteers
reliability	reasonably high but with issues for large scale clusters[1]	reasonably low – requires replication
safety for client	high	lower, requires sandboxing for safety on the client side
cost of purchase	high	low for project owner
running and maintenance costs	high	low for project owner

over 17MW for Tianhe-2[2]).

In distributed systems, there are several possibilities for performing computations. It is possible to integrate clusters into grid [6, 7] systems for distributed processing and allowing users access these using Web-based interfaces. Cloud computing [8] can be used to rent infrastructure including clusters that can be used for data processing. Paper [9] presents how to make use of Google App Engine for parameter study problems using Task Queue API.

Volunteer computing is mostly suited for embarrassingly parallel computations and applications in which various instances can be launched with different input data sets. There exist several implementations of such systems and approaches including:

1. BOINC[3] [10] – the leading solution for volunteer-based projects which allows definition of several projects, client-side settings referring to usage of particular resources. Projects such as Search for Extraterrestrial Intelligence[4] (SETI@home), biomedical research[5] (GPUGRID.net), Collatz Conjecture[6] and Protein Optimization with Energy Methods[7] (POEM@home) are among the most popular ones on the platform.

2. Comcute[8] [1] – Comcute was designed as a distributed volunteer-based project but with two fundamental differences compared to BOINC:

   - ability to perform computations within Web browsers of volunteers instead of desktop applications. On one hand, this assures secure processing from the client point of view as computations are performed within a sandbox. On the other hand, technologies allowed on the client side are limited to those available within the browser such as: Javascript, Java applets, Flash, Silverlight etc.
   - more complex processing on the server side where a group of so-called W servers is elected for management of computations of an individual project to ensure scalability and reliability of computations. W servers partition data, either before computations or on-the-fly, distribute data packets among distribution servers S and gather and integrate results. S servers are contacted directly by volunteers. Comcute allows definition of trade-off between performance and reliability by assignment of proper numbers of servers to particular projects.

3. WeevilScout prototype framework [11] adopts a similar approach to Comcute for processing data on the client-side within a browser. However, compared to Comcute, WeevilScout focuses only on Javascript.

There are several models available for modeling various aspects of volunteer computing. Paper [12]

[2] http://top500.org/system/177999
[3] http://boinc.berkeley.edu/
[4] http://setiathome.berkeley.edu/
[5] http://www.gpugrid.net/
[6] http://boinc.thesonntags.com/collatz/
[7] http://boinc.fzk.de/poem/
[8] http://comcute.eti.pg.gda.pl/en/

discusses availability of various types of computers that are available for computations. Percentages are given for the following categories of users: student, home, business, public computers. Average durations of the following period types are given for the categories: available, unavailable (powered on, off).

Work [13] models phases related to job lifespan in volunteer computing including the following phases: distribution, in progress (computations) and validation. Projects such as P@H CHARMM, P@H MFold, WCG FightAIDS@Home, WCG Genome Comparison are analyzed. Statistical methods are used and functions are used for modeling particular phases with best distributions determined: Weibull for distribution, mixture of Gaussian distributions for in progress and validation phases.

Paper [14] presents a model of Internet volunteer hosts including distribution of the numbers of cores and available memory over successive years. Additionally, histograms with performance, disk space, cache sizes are given for particular years. Statistical models of availability for the SETI@home project are discussed in [15].

Paper [16] discusses an environment for emulation of scheduling policies for volunteer computing including: job dispatch decisions (which jobs to send), job fetching (from the client perspective) and job scheduling on the client side.

Paper [17] presents efficiency and scalability of task distribution in BOINC that allows achieving over 23 million tasks per day.

On the other hand, work [18] studies social aspects of participation in volunteer-based computing projects and relation to contribution to the project. The paper finds that factors such as enjoyment and reputation do not have significant impact on contribution levels. On the other hand, enhancement and team affiliation were found to have statistically significant impact on contribution.

The authors of work [19] experimented with video consoles such as Playstation devices as additional devices that could be used for computations in volunteer based project. Although in the test Playstation machines were outperformed by CPUs tested, the devices were able to complete the required tasks successfully.

3 Proposed Model of Volunteer Computing

Compared to the existing models, this paper proposes an analytical approach for modeling application workload in a volunteer-based system. It is assumed that the application requires to process d data packets of same size. Contrary to the well expected execution time of a data packet in a fully reliable and uniform cluster environment, in a volunteer environment variance in computing power must be taken into account due to its impact on the processing time of the data packet and associated timeouts. Once a timeout is assumed, the server node will redistribute the data packet to another volunteer. This results in some lost work and increased total workload on volunteers' computers.

3.1 Theoretical Model

The proposed performance model of a volunteer based system needs to consider the following specific factors:

1. heterogeneity of the environment, especially in terms of various:
   - hardware specifications related to computing resources,
   - software specifications and incompatibility, e.g. the browser not capable of rendering web sites using specific technologies when web-based voluntary model is considered,
2. network contention and bottlenecks on the server size including the capability of project servers to handle a large number of clients,
3. reliability of computations considering:
   - failures of equipment including issues with:
     - hardware failure of end computers,
     - network failures including availability of particular ISPs,
     - software reliability such as OSes,
   - usage patterns of distinct users i.e. how much time the typical user spends on a web page and particular probabilities of leaving the page after the given time (in a web-based voluntary model) [20], or computer usage patterns in BOINC-like scenarios,
   - power outages.

All these factors compounded influence the availability of volunteer nodes, which in turn influences the reliability and performance of the whole computing environment, which are addressed in more detail below.

ISBN: 978-960-474-381-0

3.1.1 Reliability Model

The reliability model of voluntary computations should ideally consider the following factors:

- internal faults:
  - hardware,
  - application software,
  - framework software,
  - operating system,
- communication faults:
  - network outages on site (net hw failures, power loss),
  - network outages (provider),
- power faults:
  - public utility grid outages, (will affect at least local connectivity too).

3.1.2 Task Execution Probability

In the context of voluntary computing a single task execution is deemed to be complete when the computation coordination center was able to successfully upload a task/data to the volunteer node V and retrieve a result. The probability of a successful task execution will be a product of individual probabilities related to reliability factors listed Section in 3.1.1 and the probability of successfully computation finalization given the individual data packet processing demand and the average volunteer computing capabilities.

Once all reliability factors are considered, a probability distribution describing the average computing capabilities of a V node can be given. The model described in this chapter proposes to use a sum of Gaussians probability distribution to describe distribution of processing power in V nodes, but using other probability density function, perhaps better fit to a particular data, does not invalidate the approach chosen.

In addition, the model presented here should consider different sociological and technological approaches e.g. Boinc's [10] dedicated client compared to browser based computations in Comcute [1, 21]. These result in different probability distributions of volunteer availabilities and capabilities.

The placement of an average execution demand of a single job (data unit) of a given task with relation to the characteristics of a voluntary model employed (as shown in Fig. 1) will have a significant impact on the task execution probability. This should be evident from looking at the illustration above and will be addressed in a more formal way in the next sections.

3.1.3 Estimated Processing Time of a Single Data Unit

The average processing time of a single data unit D belonging to job J run on a volunteer node with average computing power C_A can be estimated as:

$$TP_J(D) = \frac{C_J(D)}{C_A} \quad (1)$$

where:

D	-	a data unit,
$C_J(D)$	-	application dependent translation quantifying the computational requirements to process a single data packet (in agreed upon units, e.g. MFlops),
C_A	-	the average computing power of a volunteer node (e.g. in MFlops/s).

For jobs, where data processing time $TP(D)$ is orders of magnitude larger than communication time needed to transfer data D the model may ignore the communications. In all other cases the communication time $TC(D)$ of data and results should be considered. Equation (1) then becomes:

$$TP_J(D) = TC_N^P(D) + \frac{C_J(D)}{C_A} + TC_N^P(R) \quad (2)$$

where:

D	-	a data unit,
R	-	a result,
$C_J(D)$	-	application dependent constant quantifying the computational requirements to process a single data packet (in agreed upon units, e.g. MFlops),
C_A	-	the average computing power of a volunteer node (e.g. in MFlops/s).

However, it should be noted, that volunteers are connected to the Internet by a wide range of networking technologies, with different bandwidths and latencies, therefore the $TC()$ function will show a large variance and should be considered as an average of the whole set.

Because many jobs may be running in parallel in the same environment, the resource allocation coefficient η_{Ra}^J needs to be applied:

$$TP_J(D) = TC_N^P(D) + \frac{C_J(D)}{\eta_{Ra}^J C_A} + TC_N^P(R) \quad (3)$$

where:

η_{Ra}^J	-	processing resource allocation coefficient for a given job J.

ISBN: 978-960-474-381-0

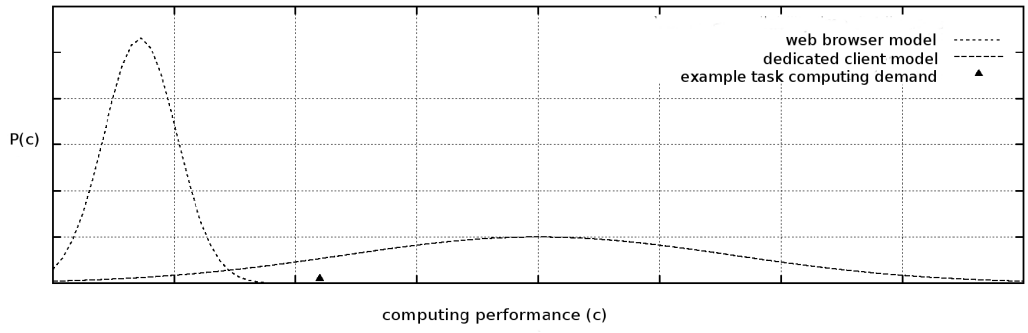

Figure 1: Two volunteer models of node computing performance distributions in relation to each other and to the example average execution demand of a single data set in a task

3.1.4 Estimation of Work Required for Single Data Unit Execution

In this case we assume that volunteers join the project and request data packets. Each volunteer fetches a data packet and starts processing it. Depending on the computing power of the volunteer computer, this will influence the processing time for the data packet. From the point of view of the server, though, we impose a timeout (e.g. 24 hours) for results of this particular data packet to come back. If the results have not come back (because the computing power of the volunteer computer is too small), the server assumes that the data packet needs to be fetched by another volunteer.

Once an average work required for execution of a single data unit of a job J is known it can be related to the characteristics of a voluntary computing environment shown in Fig. 1. For instance, in [13], a mixture of Gaussian distributions is suggested. We follow this approach and use it for estimation of real workload imposed by the application on the system.

We assume that $G(x)$ is a probability distribution of computational power of volunteers' computers. In this case we model it by combining three Gaussian distributions, each with a mean μ_i and variance ${\sigma_i}^2$, described by the following formula:

$$G_i(x) = \frac{1}{\sigma_i\sqrt{2\pi}} e^{\frac{-(x-\mu_i)^2}{2\sigma_i^2}} \tag{4}$$

with

$$G(x) = \sum_{i=1}^{3} \omega_i G_i(x). \tag{5}$$

Then the question which must be addressed is: what is the probability of a successful processing of a single data unit D given execution environment described by G?

Let us assume the following notation:

WP_J – work required for completion of a single data unit of job J,

WT_J – total estimated work required for completion of a single data unit of a job J assuming timeouts and retries resulted from inability of some volunteers to complete within the given time frame,

Ω – a probability *sample space* containing all possible outcomes of a task execution attempt, $\Omega = \{f, s\}$,

f – a failed outcome of attempt at task execution resulting from lack of results after timeout,

s – a successful outcome of attempt at task execution,

$P(f)$ – probability of a failed attempt at task execution,

$P(s)$ – probability of a successful attempt at task execution,

W_{a_f} – expected average work elapsed until timeout occurred with no result from a volunteer.

A failure of a task is guaranteed when the volunteer does not have sufficient computing power to complete the task within the imposed time frame:

$$P(f) = \int_0^{WP_J} G(x)dx \tag{6}$$

Execution of a single data unit may either succeed on a first try, or it may fail and require further attempts at execution, which may also succeed or fail. Therefore, the estimated work required for a data packet can be broken into two components:

1. execution succeeded – with probability $P(s)$,

ISBN: 978-960-474-381-0

2. execution failed – with probability $P(f)$ and processing needs to be restarted.

This observation allows to express the formula for estimated total work required to process a single data unit:

$$WT = P(s)WP + P(f)(W_{a_f} + WT) \quad (7)$$

which can be further expressed as:

$$\begin{aligned} WT &= \frac{P(s)WP + P(f)W_{a_f}}{P(s)} \\ &= WP + W_{a_f}\frac{P(f)}{P(s)} \end{aligned} \quad (8)$$

where the expected mean work (lost) to failure W_{a_f} is equal to the expected value of a volunteer computing power probability distribution in a given execution environment.

It should be noted, that the approach remains valid regardless of the probability density function used. It is a subject of an ongoing research, whether the best fit will be obtained by a single Gaussian, a sum of Gaussians, by a Weibull distribution etc. In [13], a mixture of Gaussians is preferred.

3.2 Model Parametrization with Real-life Data

Once the theoretical model was constructed, it was interesting to find its parameters based on real-life data from a volunteer based computing environment. Because of abundance of published data, the BOINC project was an obvious choice, although, once more data becomes available, it will be interesting to apply the same procedure to the Comcute system and compare the results. As there is a fundamental difference in approach between the two environments, in which BOINC volunteers use a dedicated client and Comcute computations are browser-based, it is expected, that the model parameters of the two approaches will be quite different. This comparison is planned as a future work.

Because the proposed model is probabilistic and attempts to describe the characteristics of a large scale volunteer system, it was decided to use aggregate data from all BOINC projects, as it is most immune to shifts in popularity of various projects, and represents the total computational potential of volunteers. It was also considered to be the most fair approach, as it does not favour any particular project.

Furthermore, it was observed, that while the available statistics list almost three million individual volunteers[9], the majority of the entries are inactive and of historical interest only. Since our goal was to find parameters of a live, active system, it was decided to filter the data and base further calculations only on well established, active volunteers. The filtering took into account the daily, weekly and monthly credit fields of each user. A volunteer was considered long term established active, when all three credit fields had a non-zero value. It was discovered, that such users can be located within first 200000 entries of statistics. For an extra margin, 300000 entries were downloaded for further filtering. The process was repeated at approximately monthly interval and is summarized in Table 2. The sum of total credits contributed was calculated using either a sum of daily credits listed for a given volunteer, or a sum of their recent average credit. The first approach is more of a snapshot from performance of a given day, while the latter is an averaged daily performance over a period of time, and gives more realistic estimate of a volunteer performance over longer time. While the total observation period is not yet long enough to formulate any long-term trend observations, it is nevertheless interesting to see the short term trends in data, as charted in Figure 2.

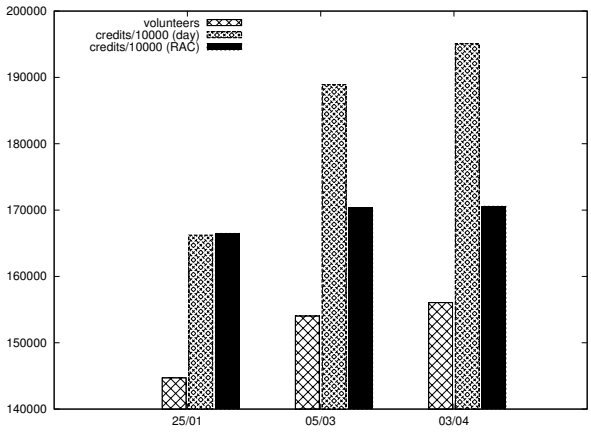

Figure 2: A histogram of a number of stable active volunteers and their contributed computing power (credits) using two estimates: daily and recent average credit values

Once the active volunteers credit data was extracted, it was further binned into groups of adjacent values, to smoothen the data. The raw results from January data set are shown in Figure 3 and the same results smoothed with bin width 10 are shown in Figure 4. Both figures show only the beginning of the computational power density distribution. The actual data continues along the x axis with a very long tail

[9] http://boincstats.com – total number of users: 2 776 744 (April, 2014).

Table 2: Summary of BOINC data downloads

date	Jan 25	Mar 05	Apr 03
raw data points	300 000	300 000	300 000
data points considered	144 701	154 027	156 054
total credits contributed (day)	1 662 334 118	1 889 065 086	1 950 694 941
total credits contributed (RAC)	1 664 835 650	1 703 564 542	1 705 687 752

of fairly small densities, with y values in the order of 10^{-6}.

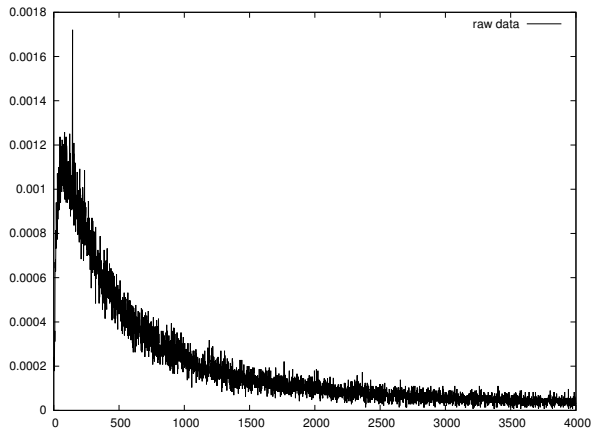

Figure 3: A volunteer computational power density distribution - raw data (Jan 25 set)

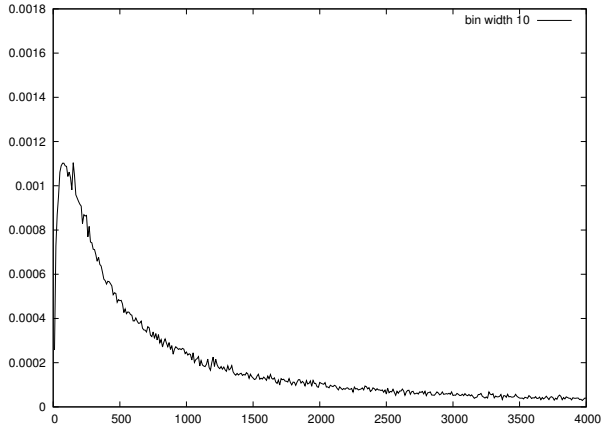

Figure 4: A volunteer computational power density distribution - smoothed data (Jan 25 set)

To determine the final parameters of the volunteer based distributed system model described earlier, a sum of Gaussian distributions (n=3) was fit to the raw data, as well as to the smoothed data. The resulting parameters are presented in Table 3 and the fit obtained is shown in Figures 5 and 6 respectively.

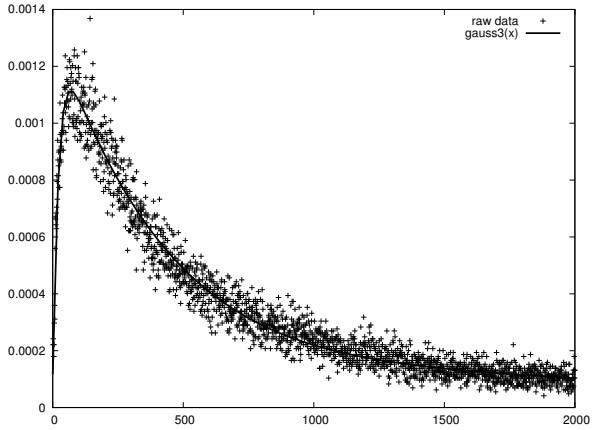

Figure 5: A sum of Gaussians fit to raw computational power density data - (Jan 25 set)

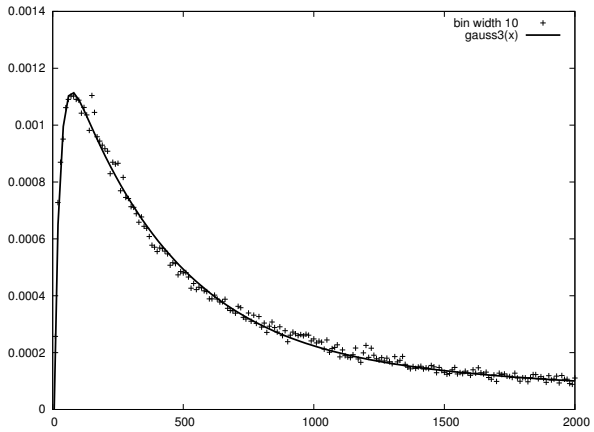

Figure 6: A sum of Gaussians fit to smoothed (bin size 10) computational power density data - (Jan 25 set)

3.3 Prediction of Real Application Workload in BOINC

To predict a performance of an application in BOINC, measured as total work needed to complete the execution, we assume a benchmark application, which represents a workload of a 100000 data packets, where calculations of each single data packet require a computational job WP_J equivalent to 200 BOINC credits. Given these assumptions, a probability of failed computation of a single data packet in a volunteer en-

Table 3: Model parameters for January measurement set

data set	ω_0	ω_1	ω_2	σ_0	μ_0	σ_1	μ_1	σ_2	μ_2
raw	-224.59	46.3393	96.4946	85.2384	-313.443	998.829	-2375.89	6268.57	-15954.3
smooth	-1528.32	3239.64	131.152	94.071	-383.305	1470.99	-5362.61	6845.21	-18229.7

vironment described by parameters given in Table 3 must be calculated. We can do it as follows:

$$P(f)^{raw}_{Jan\ 25} = \sum_{i=1}^{3} \int_0^{200} \omega_i G_i(x)dx = 0.1927032$$

$$W_{a_f} \cdot P(f)^{raw}_{Jan\ 25} = \int_0^{200} x \cdot G(x)dx = 19.886711$$

Finally, from Equation 8, $WT^{raw} = 227.997748265$. For smoothed data we use the same formulas.

$$P(f)^{smoothed}_{Jan\ 25} = \sum_{i=1}^{3} \int_0^{200} \omega_i G_i(x)dx = 0.1868745$$

$$W_{a_f} \cdot P(f)^{smoothed}_{Jan\ 25} = \int_0^{200} x \cdot G(x)dx = 19.913226$$

Finally, from Equation 8, $WT^{smoothed} = 227.768118691$

A total work estimate for our benchmark application can be expressed as a multiplication of a single data packet work estimate by a total number of packets in benchmark.

4 Conclusions and Future Work

In this paper, we have presented an analytical model for assessment of real application workload in a volunteer based system, in which a certain distribution of volunteer computers' performance is given. We have shown how to extract model parameters from statistics gathered from the BOINC system. We assume that some data packets cannot be completed by some less powerful volunteers in the given timeframe, which results in the need for resending the work to others. We estimate real work that needs to be performed to complete the task taking probabilities of such repetitions into consideration.

In the future, following our previous work for cluster-based systems [22], we plan to extend this approach with consideration of power consumption, especially prediction of application execution time with specific power consumption constraints on the devices being used for computations.

Acknowledgments: The work was performed within grant "Modeling efficiency, reliability and power consumption of multilevel parallel HPC systems using CPUs and GPUs" sponsored by and covered by funds from the National Science Center in Poland based on decision no DEC-2012/07/B/ST6/01516.

References:

[1] Czarnul, P., Kuchta, J., Matuszek, M.: Parallel computations in the volunteer based comcute system. In Springer-Verlag, ed.: Proc. of Parallel Processing and Applied Mathematics (PPAM). Volume LNCS 8384., Poland (2013) in press.

[2] Majid, Z.A., Mehrkanoon, S., Othman, K.I., Suleiman, M.: Implementation of mpi environment for solving large systems of odes using block method. WSEAS Trans. Math. **9** (2010) 801–810

[3] Czarnul, P., Grzeda, K.: Parallel Simulations of Electrophysiological Phenomena in Myocardium on Large 32 and 64-bit Linux Clusters. In: 11th European PVM/MPI Users Group Meeting Budapest, Hungary, September 19 - 22, 2004. Proceedings. (Volume 3241/2004.)

[4] Karniadakis, G.M., Kirby, R.M.: Parallel Scientific Computing in C++ and MPI. Cambridge University Press, New York, NY, USA (2003)

[5] Czarnul, P., Ciereszko, A., Fraczak, M.: Towards efficient parallel image processing on cluster grids using gimp. In Bubak, M., van Albada, G.D., Sloot, P.M.A., Dongarra, J., eds.: International Conference on Computational Science. Volume 3037 of Lecture Notes in Computer Science., Springer (2004) 451–458

[6] Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications **15** (2001) 200–222

[7] Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Inte-

gration. In: Open Grid Service Infrastructure WG. (2002)

[8] Bellenger, D., Bertram, J., Budina, A., Koschel, A., Pfänder, B., Serowy, C., Astrova, I., Grivas, S.G., Schaaf, M.: Scaling in cloud environments. In: Proceedings of the 15th WSEAS International Conference on Computers, Stevens Point, Wisconsin, USA, World Scientific and Engineering Academy and Society (WSEAS) (2011) 145–150

[9] Malawski, M., Kuzniar, M., Wojcik, P., Bubak, M.: How to use google app engine for free computing. Internet Computing, IEEE **17** (2013) 50–59

[10] Anderson, D.P.: Boinc: A system for public-resource computing and storage. In: Proceedings of 5th IEEE/ACM International Workshop on Grid Computing, Pittsburgh, USA (2004)

[11] Cushing, R., Putra, G., Koulouzis, S., Belloum, A., Bubak, M., de Laat, C.: Distributed computing on an ensemble of browsers. Internet Computing, IEEE **17** (2013) 54–61

[12] Toth, D., Finkel, D.: Characterizing resource availability for volunteer computing and its impact on task distribution methods. In: Proceedings of the 6th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems. SEPADS'07, Stevens Point, Wisconsin, USA, World Scientific and Engineering Academy and Society (WSEAS) (2007) 107–114

[13] Estrada, T., Taufer, M., Reed, K.: Modeling job lifespan delays in volunteer computing projects. In: Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on. (2009) 331–338

[14] Heien, E.M., Kondo, D., Anderson, D.P.: A correlated resource model of internet end hosts. IEEE Trans. Parallel Distrib. Syst. **23** (2012) 977–984

[15] Javadi, B., Kondo, D., Vincent, J.M., Anderson, D.P.: Discovering statistical models of availability in large distributed systems: An empirical study of seti@home. IEEE Trans. Parallel Distrib. Syst. **22** (2011) 1896–1903

[16] Anderson, D.P.: Emulating volunteer computing scheduling policies. In: Proceedings of the 2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and PhD Forum. IPDPSW '11, Washington, DC, USA, IEEE Computer Society (2011) 1839–1846

[17] Anderson, D.P., Korpela, E., Walton, R.: High-performance task distribution for volunteer computing. In: Proceedings of the First International Conference on e-Science and Grid Computing. E-SCIENCE '05, Washington, DC, USA, IEEE Computer Society (2005) 196–203

[18] Nov, O., Anderson, D., Arazy, O.: Volunteer computing: A model of the factors determining contribution to community-based scientific research. In: Proceedings of the 19th International Conference on World Wide Web. WWW '10, New York, NY, USA, ACM (2010) 741–750

[19] Toth, D.: Volunteer computing with video game consoles. In: Proceedings of the 6th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems. SEPADS'07, Stevens Point, Wisconsin, USA, World Scientific and Engineering Academy and Society (WSEAS) (2007) 102–106

[20] Nielsen, J.: How long do users stay on web pages? (2011) Nielsen Norman Group, http://www.nngroup.com/articles/how-long-do-users-stay-on-web-pages/.

[21] Balicki, J., Krawczyk, H., Nawarecki, E., eds.: Grid and Volunteer Computing. Gdansk University of Technology, Faculty of Electronics, Telecommunication and Informatics Press, Gdansk (2012) ISBN: 978-83-60779-17-0.

[22] Czarnul, P., Rosciszewski, P.: Optimization of execution time under power consumption constraints in a heterogeneous parallel system with gpus and cpus. In Chatterjee, M., Cao, J.N., Kothapalli, K., Rajsbaum, S., eds.: ICDCN. Volume 8314 of Lecture Notes in Computer Science., Springer (2014) 66–80