Modeling of Correlated Resources Availability in Distributed Computing Systems

Bahman Javadi*, Kenan M. Matawie

School of Computing, Engineering and Mathematics,
Western Sydney University, Australia

Abstract

Volunteer computing systems are large-scale distributed systems with large number of heterogeneous and unreliable Internet-connected hosts. Volunteer computing resources are suitable mainly to run High-Throughput Computing (HTC) applications due to their unavailability rate and frequent churn. Although they provide Peta-scale computing power for many scientific projects across the globe, efficient usage of this platform for different types of applications still has not been investigated in depth. So, characterizing, analyzing and modeling such resources availability in volunteer computing is becoming essential and important for efficient application scheduling. In this paper, we focus on statistical modeling of volunteer resources, which exhibit non-random pattern in their availability time. The proposed models take into account the autocorrelation structure in individual and subset of hosts whose availability has temporal correlation. We applied our methodology on real traces from the SETI@home project with more than 230,000 hosts. We showed that Markovian arrival process and ARIMA time series can model the availability and unavailability intervals of volunteer resources with a reasonable to excellent level of accuracy.

Keywords: Volunteer Computing, Resource Availability, Statistical Modeling, Markov Model, Time Series

*Corresponding author. Telephone: +61-2-9685 9181; Fax: +61-2-9685 9245
Email address: `b.javadi@westernsydney.edu.au` (Bahman Javadi)

1. Introduction

Volunteer computing systems are large-scale distributed systems with large number of heterogeneous and unreliable Internet-connected hosts. These platforms provide more than 10 PetaFLOPS computing power to more than 70 scientific projects in different areas such as astronomy, physics, mathematics and chemistry [1, 2]. No doubt utilization of such resources is essential and growing as it provides immense computational power on the order of PetaFLOPS and storage on the order of PetaBytes at almost zero costs [3]. However, volunteer computing resources are only suitable to run High-Throughput Computing (HTC) applications due to their unavailability rate and frequent churn (several times a day). So characterizing, analyzing and modeling such resources availability in volunteer computing is an essential requirement to broaden the types of applications that can be executed in this system, which is the main goal of this paper.

The overall objective of the modeling is to use the free resources of volunteer computing systems for execution of scientific applications in form of many task computing workloads. Many task computing (MTC) is a new paradigm to bridge the gap between High-Throughput Computing (HTC) and High-Performance Computing (HPC) [4]. The structure of MTC applications can be considered as graphs of discrete tasks. These tasks can be different in terms of size, communication patterns and intensity. In most cases, the data dependencies among tasks are handled through file sharing that is a feasible way of tasks communication in volunteer computing platforms. However, in contrast to HTC applications, MTC applications have relatively short tasks (i.e., seconds to minutes long), so they need fast response time. Therefore, resources availability and their temporal structure is crucial and important for efficient scheduling of these applications.

In previous work, an analysis and methodology was proposed to form subsets of hosts with similar statistical properties that can be modeled with similar distribution functions [5]. This paper explained that about 21% of hosts exhibit *random* availability, which can be modeled with a few distinct distributions from different families. It was also shown how to apply the proposed models for stochastic scheduling of Bag-of Tasks (BoT) applications in a resource brokering context [6]. In this paper, we focus on statistical modeling of volunteer resources which exhibit *non-random* pattern in their availability time. To do this, we extended the existing methodology to further characterize, analyze and consider autocorrelation structure in modeling the subset of hosts whose availability has temporal correlation. Moreover, we are interested to find a host model as well as a system model to help the system to do more efficient task scheduling for MTC applications. In the host model, the behavior of a single host will be modeled, while for the system model, the behavior of all non-iid hosts will be modeled collectively. These two models can be adapted by the scheduler to optimize the specific criteria based on system and user requirements[1].

We applied the proposed methodology on real traces from the SETI@home project with more than 230,000 hosts. We selected various statistical models that have the ability to fit traces with temporal dependencies at both host and system levels. We conducted the model fitting and analyzed the model complexity and accuracy through state reduction. We propose a queuing simulation technique to evaluate quality of the modeling. The results show that Markovian arrival process and ARIMA time series can model the availability and unavailability intervals of volunteer resources with a good degree of accuracy.

The rest of this paper is organized as follows. Related work is described in Section 2. In Section 3, we present the detail of modeling workflow and real traces. Section 4 includes how statistical models are selected. The model fitting and analysis of the model parameters are presented in Section 5. In Section 6, the model evaluation through simulation experiments is discussed. Conclusions and future work are presented in Section 7.

2. Related work

This section describes the related work in modeling and analysis of availability in volunteer computing systems. There are several research on collecting of real availability traces in volunteer computing platforms. Most of these studies are focused on host availability [8, 9, 10], which is different from CPU availability considered in this paper. CPU availability is defined as the time when a host's CPU is available to run the application as a volunteer resource. In other words, host availability might be a misleading metric as a host can be available but not its CPU.

Moreover, some papers only focused on volunteer resources in the enterprise or university [11, 12] while we use real traces that includes hosts in the

[1] This paper is the extended version of [7]

enterprise, university and home. Some studies such as [13, 14] used availability traces of hundreds of hosts over a limited time period (e.g., a few weeks). In contrast, we study real traces of hundreds of thousands hosts over the period of 1.5 years.

There are many related work for availability modeling of volunteer systems, but most of them if not all did not take into account the temporal dependency of resource availability [15, 16, 17]. For instance, in [17], authors used the average availability as a distance metric to find cluster of hosts with the similar level of availability. So, the availability intervals were ignored as the goal was to find the correlated hosts. It has been shown that effective resource selection and scheduling is strongly depended on temporal structure of availability in such platforms [18, 19]. Hence, we propose statistical models considering the autocorrelation structure in subset of hosts whose availability has temporal correlation.

There are very limited work for availability modeling at the host level and most of them are related to the CPU load modeling [20, 21]. In [22], a forecasting approach based on vector autoregressive models and a tendency-based technique is proposed. In this approach, for each host, three different prediction will be examined and selected automatically and the prediction for the next hour will be provided. In contrast, we are looking at the modeling of the CPU availability and unavailability for the whole lifetime of the host using time series approaches.

In previous work, modeling and methodology to form subsets of hosts with purely random availability was introduced [6, 5]. Clustering technique was also used to form groups of hosts that can be modeled with similar distribution functions. It was revealed that cluster formation by static criteria such as host location, time zone and CPU speeds can not have the same results as clustering by availability distribution. In other words, there is no correlation between host location, time zone and CPU speeds of host with the length of availability intervals. In contrast, we consider statistical modeling of volunteer resources, which exhibit non-random pattern with temporal correlation in their availability time.

3. Modeling Methodology

In this section, we present the details of real traces as well as the modeling workflow used in this paper. The list of all abbreviations used in the paper is shown in Table 4.

3.1. *Availability Trace*

We used a real CPU availability trace from 230,000 hosts over the Internet between April 1, 2007 to January 1, 2009 [5]. The CPU availability is considered as a binary value indicating whether the CPU was free or not. The traces record the start and end time of CPU availability.

This trace is collected using BOINC server [23] from the SETI@home volunteer resources. BOINC is a middleware for volunteer computing and has been used in more than 60 projects such as SETI@home, Einstein@home and Rosetta@home with over one million hosts [1, 2]. The traces is application independent since they are in the level of BOINC client. In total, the traces captured 57,800 years of CPU time and 102,416,434 continues intervals of CPU availability. This trace is publicly available in the Failure Trace Archive (http://fta.scem.uws.edu.au/) [24].

3.2. *Modeling Workflow*

Previous work [6, 7] proposed a modeling workflow to model CPU availability and unavailability for large-scale distributed systems. Time series of availability and unavailability of each host in the system was used as shown in Figure 1. As you can see in this figure, A_x and U_y are random variables of availability and unavailability intervals, respectively. Different behaviors in these intervals in terms of randomness and periodicity were examined. For significant and accurate modeling, we need to capture and distinguish these different behaviors among available resources in the system. Therefore, a set of randomness tests were used to classify hosts whose availability is truly random [5, 6].

Randomness tests were applied on both variables A_x and U_y and when significant they were classified as *iid hosts*. This means that they have identical and independent distribution for availability and unavailability intervals. Otherwise they will be considered as *non-iid hosts*. Also, it was observed that 21% of total hosts in a large-scale volunteer computing systems have random availability. For iid hosts, clustering approach based on a distance metric that measures the difference between two distributions was applied, this resulted to six different clusters of hosts. The availability and unavailability intervals of these clusters can be accurately modeled by several distinct families such as Gamma and hyper-exponential distributions. For more detail about modeling of iid hosts, you can refer to [5, 6].

In this paper, we focus on CPU availability of non-iid hosts, which are the majority of the hosts in the volunteer computing system (about 79%). The

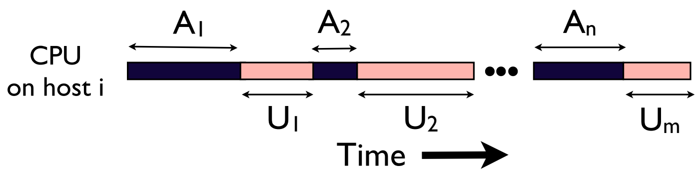


Figure 1: CPU availability and unavailability intervals on one host.

behavior of these hosts in contrast to iid hosts could be deterministic due to periodicity or trends, such as hour-in-day or day-in-week time effects as well as temporal correlation. To this end, we propose the statistical analysis and modeling of the availability and unavailability intervals for non-iid hosts.

4. Model Selection

For iid hosts the focus was on discovering a statistical model to fit the cumulative distribution function (CDF) of availability and unavailability intervals. This means that there is no dependencies in time series and the autocorrelation function (ACF) vanishes for all nonzero lags. It was clear that phase-type distribution such as hyper-exponential was an attractive model for distribution fitting of heavy-tail behavior in unavailability intervals. However, availability intervals have shorter tails and can be accurately modeled by simpler models such as Gamma distribution [5]. In contrast, non-iid hosts have some dependencies in their availability and unavailability intervals which cannot be captured by renewal models. Moreover, we are interested to find a host model as well as a system model to be used by the system for better task scheduling. In the host model, the behavior of a single host will be modeled, while for the system model, the behavior of all non-iid hosts will be modeled collectively. So the first step is to nominate a set of statistical models that can consider both CDF and ACF of the target random variables. To do that, we need to first inspect the characteristics of the non-iid hosts in terms of distribution as well as dependencies.

4.1. System Model

For the system model, we look at all the non-iid hosts collectively. Figure 2 shows the mass-count disparity of availability and unavailability intervals for non-iid hosts. From this figure we can clearly observe that about 20% of total availability is created by 90% of short availability intervals. So, the 10% of long availability interval contribute for the rest of 80% of total

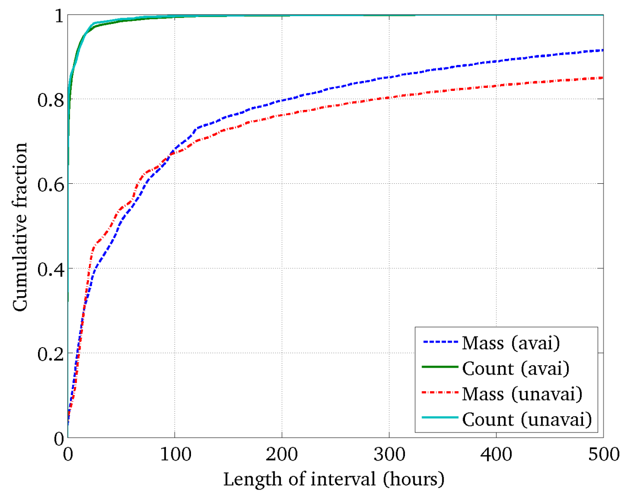


Figure 2: Mass-count for availability and unavailability intervals of non-iid hosts.

availability. This shows that availability has *long-tail* behavior. Based on this figure, unavailability distribution has a much heavier tail compared to availability. That means we need to focus on modeling of larger intervals as they have higher contribution.

The temporal dependencies of non-iid hosts can be presented by autocorrelation function. When the ACF of the time series decays slowly we have a long memory or long-range dependence (LRD) [25]. Also, any time series X has a long-range dependence property if the ACF satisfies the following condition:

$$R(k) \sim ck^{2H-2}, k \to \infty \tag{1}$$

where $R(k)$ is the autocorrelation of lag k, c is a constant and H is the Hurst parameter, which is referred to as *index of dependence*. The LRD property can be measured by the Hurst parameter which has a value between 0.5 and 1.0 ($0.5 \leq H \leq 1.0$). The higher value of H, the greater the degree of LRD, which indicates a time series with long-term positive autocorrelation. We used three well-known estimation methods, namely Aggregate Variance, R/S statistic and Periodogram [25] to calculate the Hurst parameter. The mean and standard deviation values of the H parameter are listed in Table 1 for non-iid hosts as well as the six clusters in the iid-hosts situation. The H values in Table 1 were also confirmed when using Wavelet-based and Wavelet lifting estimators with irregular sample and no missing intervals [41], [42].

Table 1: The Hurst parameter for availability and unavailability intervals.

Trace	Non-iid hosts	iid hosts					
		Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**	**Cluster 6**
Availability	0.71±0.04	0.61±0.04	0.61±0.03	0.63±0.09	0.65±0.03	0.61±0.04	0.63±0.04
Unavailability	0.70±0.05	0.55±0.00	0.54±0.02	0.64±0.05	0.60±0.02	0.56±0.02	0.58±0.03

While clusters of iid hosts show short to middle-range of dependencies, non-iid hosts have the LRD property as the H parameter is about 0.7 (highlighted in Table 1). We can also observe that in most cases, availability intervals have longer temporal dependencies compared to unavailability intervals.

Based on the observations of distribution and temporal dependency of non-iid hosts, we investigated and selected three statistical models for the fitting, which are briefly explained as follows. Markov modulated Poisson process (MMPP) is a doubly stochastic Poisson process where intensity is controlled by a finite state continues-time Markov chain (CTMC). MMPP(n) is parametrized with n Poisson arrival rates and one $n \times n$ matrix (Q) as n-state CTMC with infinitesimal generator. Therefore, the number of parameters for MMPP(n) is $n^2 + n$. MMPP has been widely used for modeling of network traffic and workload in distributed systems [26, 27]. Although MMPP has interesting features and can be used in tractable analytical model, it is not able to capture the LRD property [28]. Therefore, we selected another model called Markovian arrival processes (MAPs), a class of Markovian models developed by Neuts [29] that encompasses MMPP and phase-type distribution as special cases. A MAP(n) is defined by two $n \times n$ matrices where D_0 includes hidden transitions and elements and D_1 describes transitions rated between n states. The matrix $Q = D_0 + D_1$ is the transition rate matrix for a CTMC. The number of parameters in MAP(n) will be $2n^2 - n$. In the special case where D_1 is a diagonal matrix, the model is simply a MMPP(n).

The third model that we used to fit the (un)availability time series is Multifractal Wavelet Model (MWM) [30]. This model utilizes the power of multifractals as well as the efficiency of the wavelet transform to provide a flexible framework to capture behavior of positive LRD data. As (un)availability intervals are non-negative values this model can be used as an alternative to capture the behavior of non-iid hosts. The MWM works as a stochastic process via scaling techniques where at each scale a series of scaling coefficients and wavelet coefficient are generated to model the real traces. The MWM uses the symmetric beta distribution to fit wavelet coefficients to generate beta parameter $\vec{p}$. Finally, MWM produces the mean and variance of scaling coefficient (μ_c, σ_c) to complete the model parameters. So the number of parameters in the MWM is $m + 2$ where m is the number of factors in beta parameter $\vec{p}$. For more information about this model, you can refer to [30].

4.2. Host Model

Modeling availability intervals at a host (node) level requires flexible class of stochastic modeling that can be generalised with less conditions and softer model fitting that is in most situations difficult to achieve. Data characteristics of such nodes are important to achieving both suitability and generalisation of the model. As mentioned earlier, the availability/unavailability values of the series to model here is from a non-iid hosts and the observations are shown to be statistically dependent or related to each other. One of the best methods to model a series of such a historical data properties is Box-Jenkins method [31]. It uses an iterative approach of identifying a possible useful model from a general class of models. The suggested model is then checked and tested against the actual data in the series to determine the suitability and accuracy of the model. The model fits well if the residuals between the theoretical model and the actual data values are small, independent, and randomly distributed. When a particular model is not satisfactory, another model is suggested to improve on the original one. This process is repeated until a suitable model is achieved.

A general class of Box-Jenkis models for a stationary time series is the ARIMA, AutoRegressive Integrated Moving Average, models. A stationary time series is defined to be a kind of statistical equilibrium around a constant mean value, as well as a constant dispersion around that mean value (or series without trend, and its average value is not changing over time). There are different type of stationarity. A series is said to be stationary in the wide sense or weak sense if it has a fixed mean and constant variance, and it is strictly stationary if it has, in addition to fixed mean and constant variance, a constant autoregressive structure. ARIMA models for a time series which can be made to be stationary by differencing, if required, or in conjunction with some transformations such as logging if necessary. This can also be tested and confirmed using statistical tests for stationarity or trend-stationarity (also known as Unit Root test). The most popular and wildly applied stationarity tests are; Augmented Dicky-Fuller (ADF) [32], KPSS [33]and Ljung-Box (LBQ) [34].

Once the stationarity of a series is confirmed the selection of appropriate ARIMA model will depend on the best fit of the distributions of autocorrelation coefficients of the time series being fitted and the theoretical distributions for the various models. A general ARIMA class of models is formed from a combination of autoregressive (AR) and moving-average (MA) models as given in the equation below:

$$Y_t = \phi_0 + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ... + \phi_p Y_{t-p} + \epsilon_t - w_1 \epsilon_{t-1} - w_2 \epsilon_{t-2} - ... - w_q \epsilon_{t-q} \quad (2)$$

Where, Y_t = dependent vaiable at time t,
$\phi_0, \phi_1, ..., \phi_p$ = autoregressive coefficients,
$w_0, w_1, ..., w_q$ = moving average weights,
ϵ_t = Residual term representing the random events not explained by the model,
$\epsilon_{t-1}, \epsilon_{t-2}, ..., \epsilon_{t-q}$ = Previous values of residuals.

The first part of the above ARIMA equation is the AR model of order p, where p is the number of past observations to be considered for next observation in the model, and the second part is the MA of order q, where q is the number of past error terms to be considered in the model to represent the next observation. Moving-Average (MA) model is based on a linear combination of past errors, whereas autoregressive (AR) model express the series as a linear function of some number of actual past values of the series. ARIMA(p,d,q) models are determined by the above two values (p,q) and d, where d is the number differences needed for stationarity.

Analysis details for two examples with best suitable ARIMA models are given in Section 5.2.

5. Model Fitting

In this section, we provide the techniques and results of fitting the suggested statistical models for a single host as well as for the whole system.

5.1. System Model

For the system model, we utilized KPC-Toolbox [35] for fitting of MMPP and MAP models and a toolbox from the Digital Signal Processing (DSP) group at Rice University [36] for MWM model fitting. Both toolboxes are in Matlab, which are along with other standard Statistical Toolbox. We also implemented and modified some statistical functions ourselves.

KPC-Toolbox has a fitting technique that is more focused on higher-order correlations than higher order of moments. The toolbox automatically searches for the best order of MAP (i.e., n) that can accurately fit the trace using Bayesian Information Criterion (BIC) [35]. Given the order of target MAP, toolbox can generate the model that capture the the most essential characteristics of the trace. We used the *automatic* fitting for the MAP model while using two-state fitting for the MMPP model.

Similar to KPC-Toolbox, the MWM toolbox also uses an automatic modeling procedure for model fitting, which is based on creating a tree-like structure to generate a series of scaling coefficients and wavelet coefficients. The number of coefficients are dependent on the input trace and calculated by a recursive technique [30, 36]. So the number of factors in $\vec{p}$ will be obtained automatically based on the input trace (see Section 4).

The results of the fitting for availability intervals of non-iid hosts are plotted in Figure 3 as a complementary cumulative distribution function (CCDF) diagram. As one can see, all models seem to be a good fit especially for the end of the tail while MWM shows better fit throughout the whole distribution. MMPP has some discrepancies at the beginning of the tail as illustrated in this figure. In terms of ACF as depicted in Figure 4(a), MAP shows a close fit for the first 200 lags. The MWM also captured the autocorrelation with a reasonable accuracy for larger lags as shown in Figure 4(b). It was observed that the MAP model was not able to capture the autocorrelation feature for lags larger than 10^4, and as we expected MMPP failed to capture the LRD property of the non-iid hosts.

We conducted the same fitting for unavailability intervals of the non-iid hosts. The results of fitting in form of CCDF digram are plotted in Figure 5. This reveals that both MAP and MMPP are good fit in terms of distribution, but MWM can not model the beginning of the tail for the unavailability traces. The results of fitting for ACF depicted in Figure 6 show that MAP and MWM can expectedly fit the LRD property while MMPP failed to do so. Similar to the availability case, the MAP model can accurately fit the short lags of unavailability traces while MWM is able to fit the longer lags as illustrated in Figure 6(b).

As it can be seen in Figure 4(a) and Figure 6(a), the autocorrelation function of the availability traces decays much slower than the unavailability

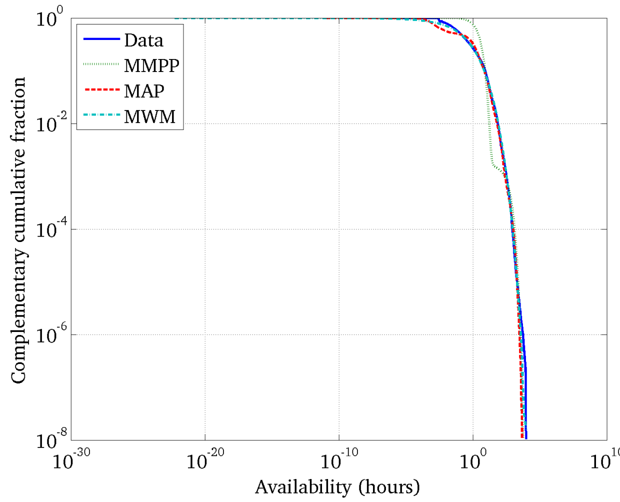
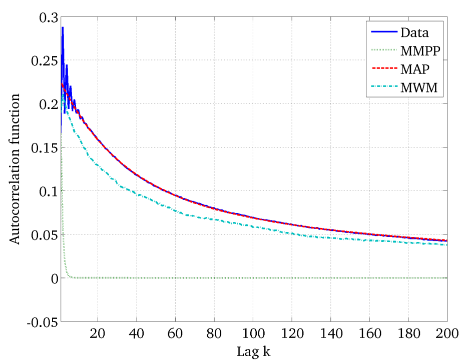
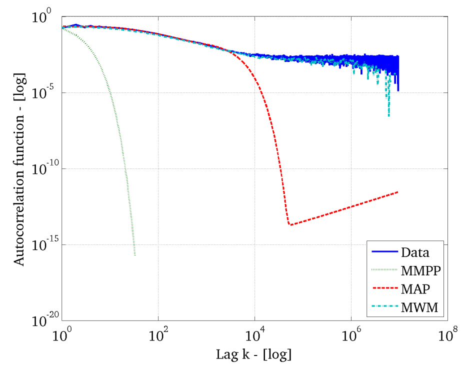


Figure 3: CCDF of the availability models and real traces.



(a) First 200 lags



(b) Full lags

Figure 4: Autocorrelation function for the availability models and real traces.

Table 2: Parameters for the fitted system models.

Model	**Availability Model**		**Unavailability Model**	
	No. of states/factors	No. of parameters	No. states/factors	No. of parameters
MMPP	2	6	2	6
MAP	16	$2 \times 16^2 - 16$	32	$2 \times 32^2 - 32$
MWM	22	24	22	24

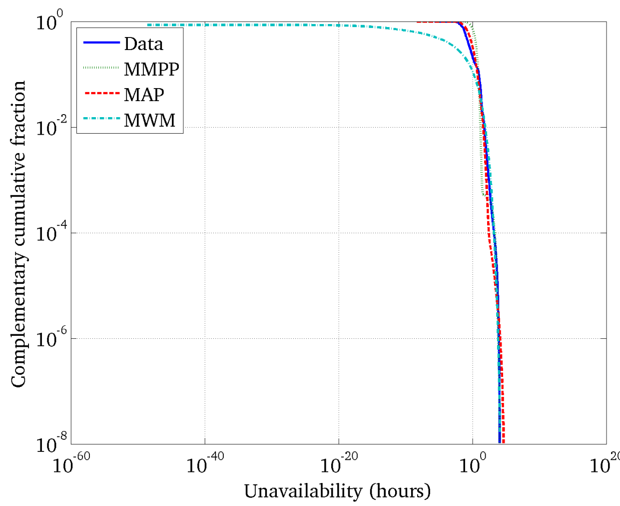
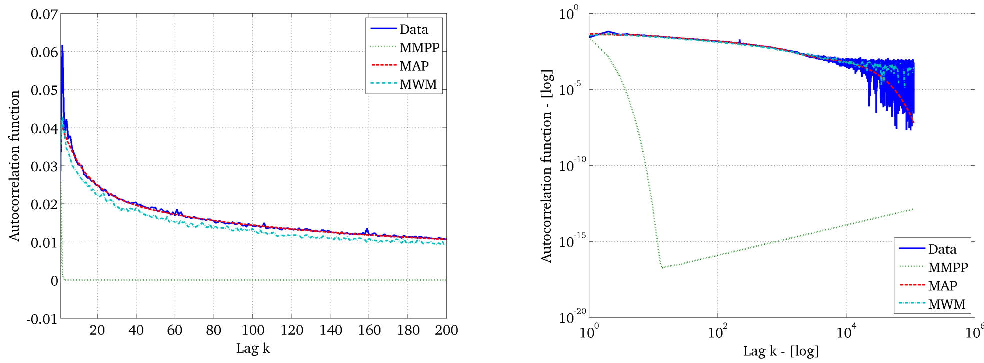


Figure 5: CCDF of the unavailability models and real traces.

traces for non-iid hosts, which means availability intervals have longer range of dependencies. This fact confirms the quantitative results using Hurst parameter given in Table 1.

As mentioned earlier, we used automatic fitting for both MAP and MWM while using two-state structure for the MMPP model. In the following, we will analyze the fitted models in terms of number of parameters. The number of states for the MAP and MMPP models as well as number of factors for the MWM model are listed in Table 2 for availability and unavailability intervals (refer to Section 4 about detail of parameters in these models). Based on this table, both MMPP and MWM have reasonable number of parameters. Although the MAP model showed a good fit for both availability and unavailability traces, but the number of states and number of parameters are very high and that might limit the application of the model. To address this issue, we consider the effect of state reductions by factor of 50% and 25% on the MAP model. To do this, we used the same fitting method with the number of states as a given value equals to half and quarter of the number of states listed in Table 2 for availability and unavailability intervals. We observed that the two new models have very similar behavior in terms of CCDF and ACF in compare with the original MAP model. For the sake of brevity, we used these models in the model evaluation in Section 6.



(a) First 200 lags

(b) Full lags

Figure 6: Autocorrelation function for the unavailability models and real traces.

5.2. Host Model

For the host model, we utilised R time series packages and Matlab toolboxes. All the host's availability and unavailability series were tested for stationarity using ADF, KPSS and LBQ tests as mentioned in Section 4.2. The test result showed that 15% of the non-iid hosts passed these tests; not to reject the stationary hypothesis at 0.05 level of significance, for both availability and unavailability. It was also noted that most of the series had few large interval values, to eradicate this and also have stronger stationarity we took the log (of base 10) for all of the raw data/series. This showed almost all the data, particularly the 15% mentioned above, were significantly stationary and it further qualified for better and smoother ARIMA Model.

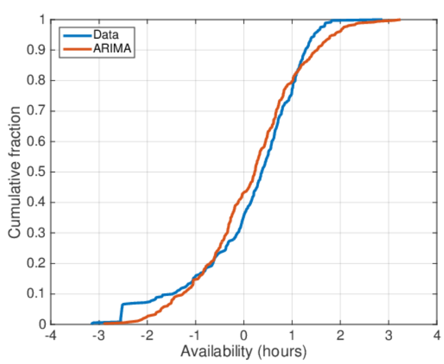
Auto ARIMA in R was used to generate the best fitted ARIMA model for the data, it also provided the estimates for the parameters (p,d,q), optimal lags, Akaie information criterion (AIC) [37], and log likelihood values. A large number of hosts was analysed, and as an example we randomly selected two hosts, Node1 and Node2. The stationary tests for these two nodes were strongly significant at less than 0.05 level of significance, for all the above mentioned three stationarity tests.

The Autocorrelation Function (ACF) for the first 200 lags for both samples are given in Figure 7(b)and Figure 8(b) for availability of Node1 and unavailability of Node2, respectively. These two graphs showed, as for most of the other hosts, the stationary process is decaying fast over time indicating short-memory dependency and good ARIMA fit (red color dots). It was

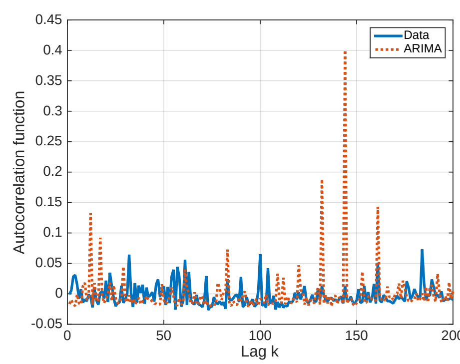
Table 3: Parameters for the fitted host models.

Host	Availability Model				Unavailability Model			
	Best ARIMA model	No. of parameters	Log Likelihood	AIC	Best ARIMA model	No. of parameters	Log Likelihood	AIC
Node1	(1,0,0)	2	-548.7	1103.5	(3,1,2)	7	-555.1	1122.1
Node2	(0,0,0)	1	-625.6	1253.2	(1,0,1)	3	-620.6	1247.3

expected that ARIMA models will fit availability data better than unavailability, as the last, usually is harder to fit due to long interval breaks and other properties mentioned earlier, see Figures 7(a) and 8(a) . In general ARIMA models fit the data structure very well and significant when the best selected ARIMA model was determined by the maximum log likelihood and minimum AIC values over all possible class of models. The best model for both nodes with their analysis details are reported in Table 3. Even though ARIMA models fit the availability data very well, we still couldn't more closely capture the upper long tail of some series as can be seen in these figures.



(a) CDF

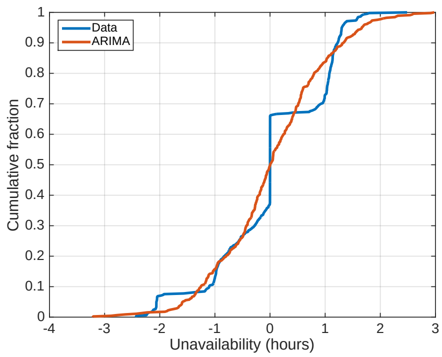


(b) ACF for the first 200 lags

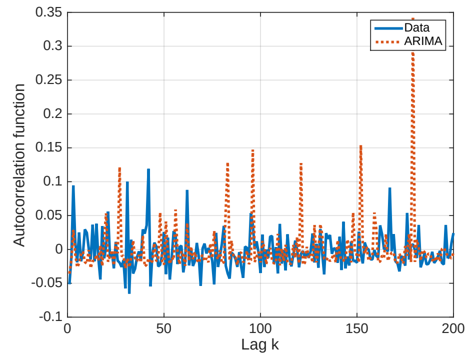
Figure 7: CDF and autocorrelation function for the availability model of Node1 (Log values)

6. Model Evaluation

In this section, we present the evaluation of the proposed models in the previous section. Goodness of fit (GoF) tests are the basic methods for



(a) CDF



(b) ACF or the first 200 lags

Figure 8: CDF and autocorrelation function for the unavailability model of Node2 (Log values)

evaluating the quality of fitting [38]. These tests include visual methods such as probability-probability (PP) plot as well as quantitative tests such as Kolmogrov-Smirnov (KS) and Anderson-Darling (AD) tests [5]. However, all of these tests are focus only on the distribution of the model (e.g., CDF) and do not consider any autocorrelation structure. In our modeling, we need to utilize a method that can combine both features (distribution and autocorrelation) and evaluate them at the same time.

Since our modeling technique is similar to modeling of inter-arrival time in network traffic and workload traces, we used the same approach for model evaluation. In this method, due to time-varying characteristics in the real traces, the quality of modeling will be evaluated by comparing the behavior of $Model/M/1$ queue versus $Trace/M/1$ queue [26, 39]. This means that they simulate two queues where the inter-arrival time to each queue will be generated by the fitted model or the real traces. They usually compare the queue length probabilities of two queues under different levels of utilizations. Since we model the process of (un)availability interval, we need to modify this queuing system for model evaluation appropriately.

As it is illustrated in Figure 1, a CPU on host i can serve as a queue to run incoming jobs. So if we consider jobs as the incoming requests then CPU works as a server with a *variable* service time. Hence, we can use $M/Model/1$ and $M/Trace/1$ queuing systems for model evaluation. We consider the exponential distribution for the inter-arrival time of input jobs to focus only on the target metric, which is the queue service time. But the question is how to generate the service times while we have two separate series for availability and unavailability for the server.

In order to generate service times for these queues, we proposed Algorithm 1. In this algorithm, the input variables are time series of availability and unavailability that can be fed from the real traces or fitted models. The output is the service time vectors with N elements. We first generate a job where its run time follows an exponential distribution with the given mean value of T (Line 6). If the job size is less than the current CPU availability time, then the job size will be the service time since the job doesn't experience any interruption due to CPU unavailability. (Line 27). Otherwise, we need to add one or more unavailability intervals into the service time until the job get served (Line 8,17). In other words, we need to add unavailability times to the service time as jobs need to wait until the next availability intervals to continue execution. In this algorithm, *residual* keeps track of the current value of the CPU availability. For the sake of simplicity, we ignore the CPU power heterogeneity in this algorithm. This assumption doesn't have any effect on the generality of this algorithm as we observed that CPU speed and availability time has no correlation [5].

6.1. Simulation Setup

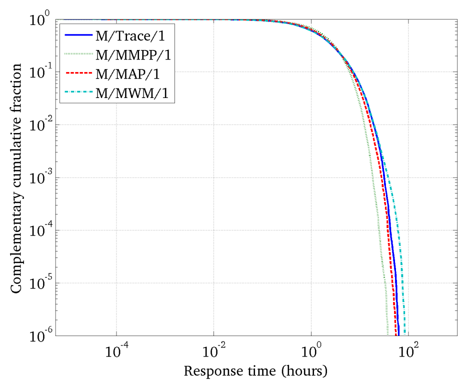
In order to simulate the two queuing systems, we implemented a discrete-event simulator using the Objective Modular Network Testbed in C++ (OMNeT++) [40]. This simulation environment is open-source, component-based and modular, which has been widely used for network simulations. We consider the response time of the queue as the performance metric for the model evaluation. It should be noted that we utilize the same model for availability and unavailability to generate the service times. We leave the other combinations such as the MAP model for availability and the MWM model for unavailability for the future work.

The simulator uses the service times generated by Algorithm 1 from the fitted models or the real traces. Based on the characterization of real BOINC projects, the average job size is about 1.2 hours [13], so we used the $T = 4320sec$ in this algorithm. We simulate each single queue with two different utilization values. To do this, we changed the input job rate in each queue to obtained different queue utilizations. We used 20% and 40% utilizations for the simulation experiments.

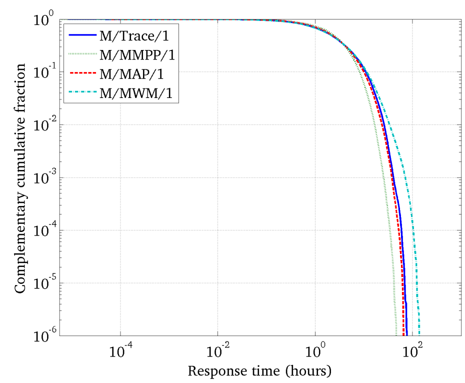
Algorithm 1: Service Time Generation

```
Input: Avai, Unavai, T, N
Output: Service.Time
residual = Avai(1);
i = 1, k = 1;
while sizeof(Service.Time) ≤ N do
    Service.Time(i) = 0;
    //job run time;
    job.size = exprnd(T);
    if job.size > residual then
        Service.Time(i) = Service.Time(i) + residual + Unavai(k);
        i = i + 1, k = k + 1;
        if k > sizeof(Avai) then
            return;
        //remaining time for job completion;
        job.size = job.size − residual;
        while job.size ≥ 0 do
            residual = Avai(k);
            if k > residual then
                Service.Time(i) = Service.Time(i) + Avai(k) + Unavai(k);
                job.size = job.size − Avai(k);
                i = i + 1, k = k + 1;
                if k > sizeof(Avai) then
                    return;
            else
                break;
        residual = Avai(k) − job.size;
    else
        residual = residual − job.size
    Service.Time(i) = Service.Time(i) + job.size;
    i = i + 1
```

For each simulation experiment, we used the batch means method to gather the statistics where we had 100 batches each of which with 20,000 jobs. The first and the last batch were ignored as warm-up and drain phases of the simulation. In our experiments the coefficient of variation of the results was very low (CV<0.01).

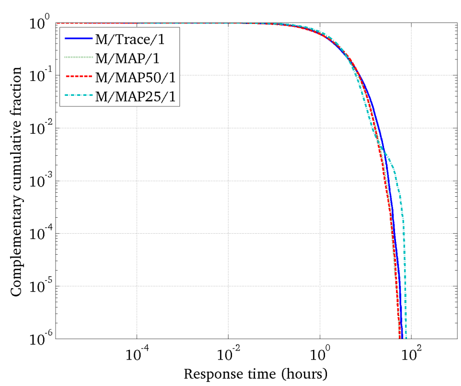


(a) 20% utilization

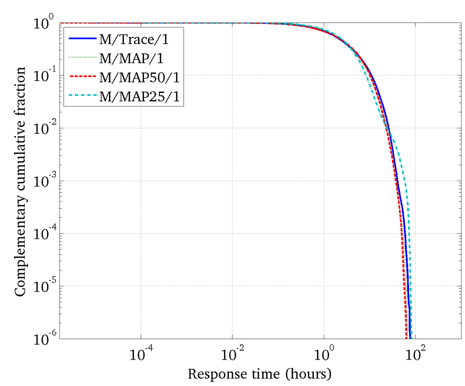


(b) 40% utilization

Figure 9: CCDF of the queue response time with different utilizations for the fitted models.

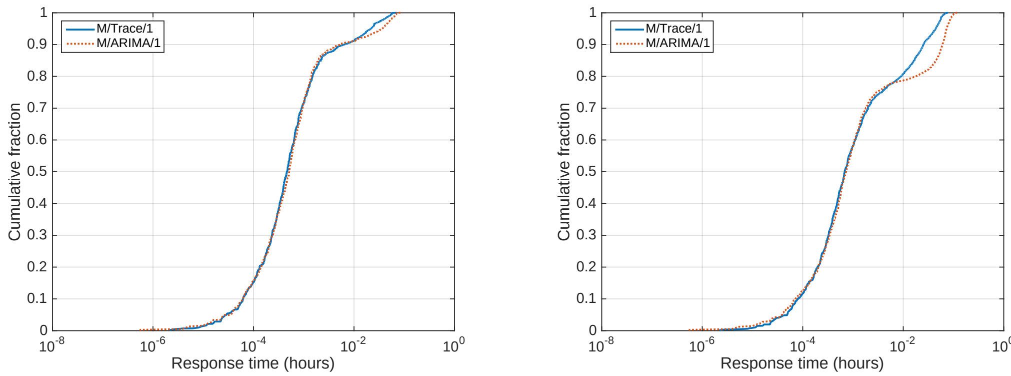


(a) 20% utilization



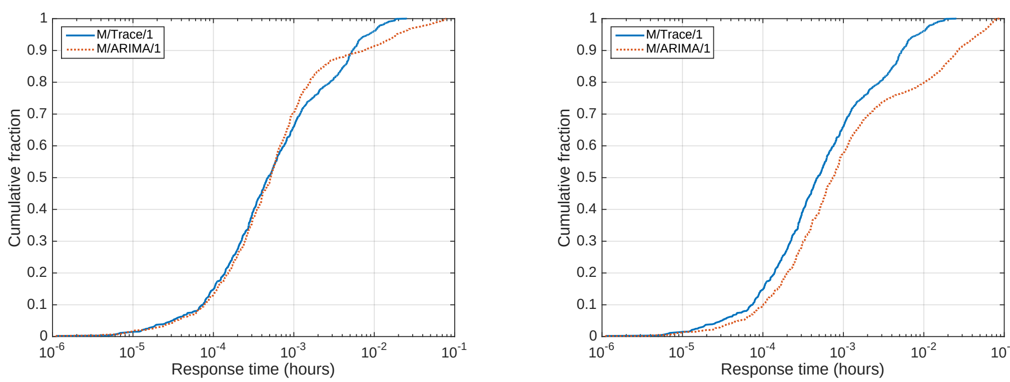
(b) 40% utilization

Figure 10: CCDF of the queue response time with different utilizations for the modified MAP models.



(a) 20% utilization (b) 40% utilization

Figure 11: CDF of the queue response time with different utilizations for the fitted models for Node1



(a) 20% utilization (b) 40% utilization

Figure 12: CDF of the queue response time with different utilizations for the fitted models for Node2

6.2. Results and Discussions

First, we present the results of the system model. The simulation results for the response time of the queues while using three different proposed system models as the service time are depicted in Figure 9. The CCDF of response time in four different queues for 20% and 40% utilizations are plotted in Figure 9(a) and 9(b), respectively. As can be seen in these figures, under lower utilization, both MAP and MWM are good matches while for higher utilization MAP model shows a closer fit to the real data. We can also observe that although MMPP model doesn't seem to be a good fit based on discussion in Section 5, it has a reasonable match especially for higher utilization queue.

These results reveal that the Markovian arrival process can model the availability and unavailability intervals of volunteer resources with a reasonable to excellent level of accuracy. The main advantage of MAPs is that they can be easily integrated within queuing systems or queuing networks, and then used in the evaluation of system performance. However, as we discussed at the end of Section 5, number of states in the MAP model could be a disadvantage for practical performance analysis. In the following, we analyze the result of state reduction for the proposed MAP model.

As mentioned before, we have reduced the number of states in the MAP model by a factor of 50% and 25%. To analyze the effect of this reduction on the quality of fitting, we used the same queuing systems with the new MAP models and repeat the experiments. The result of simulations under two different utilizations are plotted in Figure 10 where MAP, MAP50 and MAP25 refer to full state model, 50% and 25% of the full state model, respectively (see Table 2). As one can see, MAP50 shows a very close match with the original MAP as well as the real trace results. Moreover, MAP25 has a good fit near the end of tail while it has some discrepancies at the beginning of the tail. This result shows that we can use the simpler version of the MAP model with almost same accuracy.

The results of simulation for the host model based on ARIMA are presented in Figure 11 and Figure 12 for two different hosts. For these nodes, we plotted the CDF of response time in two different queues for 20% and 40% utilizations as we are interested to see the whole distribution. It can been seen from these figures, the models fit the body of the distribution with a good accuracy, however there are some differentiation on the tail of the distribution.

In Summary, MAP model is a good candidate to model the (un)availability traces with the LRD properties. While MMPP model is not able to capture the LRD properties in the (un)availability traces, it can be a reasonable candidate to simplify the performance modeling of the volunteer computing systems. Moreover, we observe that ARIMA model is able to capture a host behavior with its short rang dependency.

7. Conclusions

Statistical modeling of resource availability in volunteer computing systems is a key factor for efficient application scheduling in these platforms. We considered statistical modeling of volunteer resources, which exhibit non-random pattern in their availability time. The proposed models take into account the autocorrelation structure in subset of hosts whose availability has temporal correlation. We applied various statistical models namely MMPP, MAP, and MWM for the system level and ARIMA for the host model using real traces from the SETI@home project with more than 230,000 hosts. The results showed that for the system level, MAP model can fit the availability and unavailability intervals of volunteer resources with a good to excellent level of accuracy. It also shown that the reduction of states by factor of 50% in the MAP model doesn't have much effect on the quality of the model. We also observed that the MAP model can accurately fit the autocorrelation for limited number of lags (i.e., 200 lags), and this is sufficient-enough to apply the model for resource scheduling. We also observed that ARIMA is a reasonable model for a single host with short rang dependency and can capture the behavior of each host with a good degree of accuracy. In future work, we would like to generalize and apply our modeling to include the iid-hosts model for application scheduling on volunteer computing resources.

Acknowledgments

The authors would like to thank David Anderson for providing the availability traces and Giuliano Casale for useful discussions on using KPC-Toolbox.

References

[1] The berkeley open infrastructure for network computing, http://boinc.berkeley.edu/.

[2] D. P. Anderson, Volunteer computing: the ultimate cloud, Crossroads 16 (3) (2010) 7–10.

[3] D. Anderson, G. Fedak, The Computational and Storage Potential of Volunteer Computing, in: Proceedings of the IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06), 2006.

[4] I. Raicu, I. Foster, Y. Zhao, Many-task computing for grids and supercomputers, in: Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS 2008), 2008., 2008, pp. 1–11. doi:10.1109/MTAGS.2008.4777912.

[5] B. Javadi, D. Kondo, J. Vincent, D. Anderson, Mining for statistical availability models in large-scale distributed systems: An empirical study of SETI@home, in: 17th IEEE/ACM International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2009.

[6] B. Javadi, D. Kondo, J. Vincent, D. P. Anderson, Discovering statistical models of availability in large distributed systems: An empirical study of SETI@home, IEEE Transactions on Parallel and Distributed Systems 22 (11) (2011) 1896–1903.

[7] B. Javadi, K. Matawie, D. P. Anderson, Modeling and analysis of resources availability in volunteer computing systems, in: Performance Computing and Communications Conference (IPCCC), 2013 IEEE 32nd International, IEEE, 2013, pp. 1–9.

[8] R. Bhagwan, S. Savage, G. Voelker, Understanding Availability, in: In Proceedings of IPTPS'03, 2003.

[9] S. Saroiu, P. Gummadi, S. Gribble, A measurement study of peer-to-peer file sharing systems, in: Proceedinsg of MMCN, 2002.

[10] D. Stutzbach, R. Rejaie, Understanding churn in peer-to-peer networks, in: ICM'06, Rio de Janeiro, Brazil, 2006.
URL www.imconf.net/imc-2006/papers/p19-stutzbach2.pdf

[11] D. Kondo, M. Taufer, C. Brooks, H. Casanova, A. Chien, Characterizing and Evaluating Desktop Grids: An Empirical Study, in: Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'04), 2004.

[12] W. Bolosky, J. Douceur, D. Ely, M. Theimer, Feasibility of a Serverless Distributed file System Deployed on an Existing Set of Desktop PCs, in: Proceedings of SIGMETRICS, 2000.

[13] T. Estrada, K. Reed, M. Taufer, Modeling job lifespan delays in volunteer computing projects, in: Proceedings of the 9th IEEE International Symposium on Cluster Computing and Grid (CCGrid), 2009.

[14] M. Mutka, M. Livny, The available capacity of a privately owned workstation environment , Performance Evaluation 4 (12).

[15] E. M. Heien, D. P. Anderson, K. Hagihara, Computing low latency batches with unreliable workers in volunteer computing environments, Journal of Grid Computing 7 (4) (2009) 501–518.

[16] J. R. Douceur, Is remote host availability governed by a universal law?, SIGMETRICS Performance Evaluation Review 31 (3) (2003) 25–29.

[17] D. Kondo, A. Andrzejak, D. P. Anderson, On correlated availability in internet distributed systems, in: IEEE/ACM International Conference on Grid Computing (Grid), Tsukuba,Japan, 2008.

[18] A. Benoit, Y. Robert, A. L. Rosenberg, F. Vivien, Static strategies for worksharing with unrecoverable interruptions, in: IPDPS, 2009, pp. 1–12.

[19] J. D. Sonnek, M. Nathan, A. Chandra, J. B. Weissman, Reputation-based scheduling on unreliable distributed infrastructures, in: ICDCS, 2006, p. 30.

[20] J. Liang, J. Cao, J. Wang, Y. Xu, Long-term CPU load prediction, in: Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on, IEEE, 2011, pp. 23–26.

[21] Z. Li, C. Wang, H. Lv, T. Xu, Research on CPU workload prediction and balancing in cloud environment, International Journal of Hybrid Information Technology 8 (2) (2015) 159–172.

[22] N. C. Sekma, A. Elleuch, N. Dridi, Automated forecasting approach minimizing prediction errors of CPU availability in distributed computing systems, International Journal of Intelligent Systems and Applications 8 (9) (2016) 8–21.

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

[24] B. Javadi, D. Kondo, A. Iosup, D. Epema, The Failure Trace Archive: Enabling the comparison of failure measurements and models of distributed systems, Journal of Parallel and Distributed Computing 73 (8) (2013) 1208 – 1223. doi:http://dx.doi.org/10.1016/j.jpdc.2013.04.002.

[25] M. S. Taqqu, V. Teverovsky, W. Willinger, Estimators for long-range dependence: an empirical study, Fractals 3 (04) (1995) 785–798.

[26] G. Casale, E. Z. Zhang, E. Smirni, Trace data characterization and fitting for markov modeling, Performance Evaluation 67 (2) (2010) 61–79.

[27] H. Li, M. Muskulus, Analysis and modeling of job arrivals in a production grid, ACM SIGMETRICS Performance Evaluation Review 34 (4) (2007) 59–70.

[28] H. Li, Realistic workload modeling and its performance impacts in large-scale escience grids, IEEE Transactions on Parallel and Distributed Systems 21 (4) (2010) 480–493.

[29] M. F. Neuts, Structured stochastic matrices of M/G/1 type and their applications, Vol. 5, CRC Press, 1989.

[30] R. H. Riedi, M. S. Crouse, V. J. Ribeiro, R. G. Baraniuk, A multifractal wavelet model with application to network traffic, IEEE Transactions on Information Theory 45 (3) (1999) 992–1018.

[31] G. E. P. Box, G. M. JENKINS, Time Series Analysis Forecasting and Control, Holden-Day, San Francisco, USA, 1970.

[32] D. DICKEY, W. FULLER, Distribution of the Estimators for Autoregressive Time Series With a Unit Root, Journal of the American Statistical Association 74 (1979) 427–431.

[33] D. Kwiatkowski, P. Phillips, P. Schmidt, Y. Shin, Testing the null hypothesis of Stationary against the alternative of a unit root, Journal of Econometrics 54 (1992) 159–178.

[34] G. M. Ljung, G. E. P. Box, On a measure of lack of fit in time series models, Biometrika 65 (1978) 297–303.

[35] G. Casale, E. Z. Zhang, E. Smirni, KPC-toolbox: Simple yet effective trace fitting using markovian arrival processes, in: Fifth International Conference on Quantitative Evaluation of Systems (QEST'08), IEEE, 2008, pp. 83–92.

[36] The Digital Signal Processing (DSP) group at Rice University, http://dsp.rice.edu/.

[37] H. Akaike, A New Look at the Statistical Model Identification, IEEE Transactions on Automatic Control 19 (1974) 716–723.

[38] G. Cirrone, et al., A goodness-of-fit statistical toolkit, IEEE Transactions on Nuclear Science 51 (5) (2004) 2056–2063.

[39] G. Horváth, P. Buchholz, M. Telek, A MAP fitting approach with independent approximation of the inter-arrival time distribution and the lag correlation, in: Second International Conference on the Quantitative Evaluation of Systems, IEEE, 2005, pp. 124–133.

[40] A. Varga, R. Hornig, An overview of the OMNeT++ simulation environment, in: International Conference on Simulation Tools and Techniques for Commuications, Networks and Systems, Marseille, France, 2008.

[41] W. U. Liang, and D. I. N. G. Yiming, Wavelet-based estimator for the hurst parameters of fractional brownian sheet, Acta Mathematica Scientia 37.1 (2017): 205-222.

[42] M. I. Knight, G. P. Nason, M. A. Nunes, A wavelet lifting approach to long-memory estimation, Statistics and Computing 27.6 (2017): 1453-1471.

Appendix

Table 4: The list of abbreviations in the paper.

Abbreviation	Meaning
iid	independent and identically distributed
CDF	Cumulative Distribution Function
CCDF	Complementary Cumulative Distribution Function
ACF	Autocorrelation Function
LRD	Long-Range Dependence
MMPP	Markov Modulated Poisson Process
CTMC	Continues-Time Markov Chain
MAP	Markovian Arrival Processes
MWM	Multifractal Wavelet Model
ARIMA	AutoRegressive Integrated Moving Average
H	Hurst parameter
BIC	Bayesian Information Criterion
AIC	Akaie Information Criterion