HAL
open science

Mining for Availability Models in Large-Scale Distributed Systems:A Case Study of SETI@home

Bahman Javadi, Derrick Kondo, Jean-Marc Vincent, David P. Anderson

To cite this version:

Bahman Javadi, Derrick Kondo, Jean-Marc Vincent, David P. Anderson. Mining for Availability Models in Large-Scale Distributed Systems:A Case Study of SETI@home. [Research Report] 2009. inria-00375624

HAL Id: inria-00375624

https://inria.hal.science/inria-00375624v1

Submitted on 15 Apr 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Mining for Availability Models in Large-Scale Distributed Systems: A Case Study of SETI@home

Bahman Javadi[1], Derrick Kondo[1], Jean-Marc Vincent[2], David P. Anderson[3]
[1]INRIA, France, [2]University of Joseph Fourier, France, [3]UC Berkeley, USA

Abstract

In the age of cloud, Grid, P2P, and volunteer distributed computing, large-scale systems with tens of thousands of unreliable hosts are increasingly common. Invariably, these systems are composed of heterogeneous hosts whose individual availability often exhibit different statistical properties (for example stationary versus non-stationary behaviour) and fit different models (for example Exponential, Weibull, or Pareto probability distributions). In this paper, we describe an effective method for discovering subsets of hosts whose availability have similar statistical properties and can be modelled with similar probability distributions. We apply this method with about 230,000 host availability traces obtained from a real large-scale Internet-distributed system, namely SETI@home. We find that about 34% of hosts exhibit availability that is a truly random process, and that these hosts can often be modelled accurately with a few distinct distributions from different families. We believe that this characterization is fundamental in the design of stochastic scheduling algorithms across large-scale systems where host availability is uncertain.

1 Introduction

With rapid advances in networking technology and dramatic decreases in the cost of commodity computing components, large-scale distributed computing platforms with tens or hundreds of thousands of *unreliable* and *heterogeneous* hosts are common. The uncertainty of host availability in P2P, cloud, or Grid systems can be due to the host usage patterns of users, or faulty hardware or software. Clearly, the dynamics of usage, and hardware and software stacks are often heterogeneous, spanning a wide spectrum of patterns and configurations. At same time, within this spectrum, subsets of hosts with homogeneous properties can exist.

So one could also expect that the statistical properties (stationary versus non-stationary behavior for example) and models of host availability (Exponential, Weibull, or Pareto for example) to be heterogeneous in a similar way. That is, host subsets with common statistical properties and availability models can exist, but differ greatly in comparison to other subsets.

The goal of our work is to be able to discover host subsets with similar statistical properties and availability models within a large-scale distributed system. In particular, the main contributions are as follows:

- *Methodology.* Our approach is to use tests for randomness to identify hosts whose availability is independent and identically distributed (iid). For these hosts, we use clustering methods with distance metrics that compare probability distributions to identify host subsets with similar availability models. We then apply parameter estimation for each host subset to identify the model parameters.
- *Modelling.* We apply this method on one of the largest Internet-distributed systems in the world, namely SETI@home. We take availability traces from about 230,000 hosts in SETI@home, and identify host subsets with matching statistical properties and availability models. We find that a significant fraction (34%) of hosts exhibit iid availability, and a few distributions from several distinct families (in particular Gamma, Weibull, and Log-Normal) can accurately model the availability of host subsets.

These results are essential for the design of stochastic scheduling algorithms for large-scale systems where availability is uncertain.

In the next section, we describe the application context that is used to guide our modelling. In Section 3, we define availability and describe how our availability measurements were gathered. In Section 4, we contrast this measurement method and our modelling approach to related work. In Section 6, we determine which hosts exhibit random, independent, and stationary availability. In Section 7, we describe the clustering of hosts by their availability distribution. In Section 8, we describe the implications of our results for scheduling, and in the last section, we summarize our findings.

2 Application Context

We describe here the context of our application, which is used to guide what we model. Our modelling is conducted in the context of volunteer distributed computing, which uses the free resources of Internet-distributed hosts for large-scale computation and storage. Currently this platform is limited to embarrassingly parallel applications where the main performance metric is throughput. One of the main research goals in this area is to broaden the types of applications that can effectively leverage this platform.

We focus on the problem of scheduling applications needing fast response time (instead of high throughput). This class of applications includes those with batches of compute-intensive tasks that must be returned as soon as possible, and also those whose tasks are structured as a directed acyclic graph (DAG).

For these applications, the temporal structure of availability and resource selection according to this structure is critical for their performance. The temporal structure of availability is important because many applications cannot easily or frequently checkpoint due to complexities of its state or checkpointing overheads.

Resource selection is important because in our experience and discussions with application scientists the maximum number of tasks in a batch for fast turnaround is needed is much lower than the number active and available hosts at a given time point (hundreds of thousands). This same is true for the maximum number of tasks available in a DAG application during its execution.

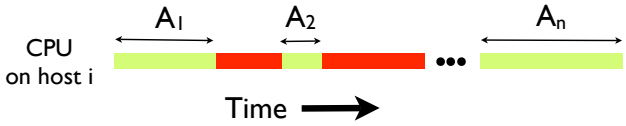


Figure 1. CPU availability intervals on one host

Thus our modelling focuses on the interval lengths of continuous periods of CPU availability for individual hosts. We term this continuous interval to be an **availability interval** (see Figure 1). The lengths of the availability intervals of a particular host over time is the process that we model.

Using these models, a scheduler should then be able to compute accurately the probability of successful task execution in some time frame, given a task's execution time. Moreover, the scheduler could compute the probability of success when task replication is used.

3 Measurement Method for CPU Availability

BOINC [5, 2] is a middleware for volunteer distributed computing. It is the underlying software infrastructure for projects such as SETI@home, and runs across over 1 million hosts over the Internet.

We instrumented the BOINC client to collect CPU availability traces from about 230,000 hosts over the Internet between April 1, 2007 to January 1, 2009. We define CPU availability to be a binary value indicating whether the CPU was free or not. The traces record the time when CPU starts to be free and stops.

Our traces were collected using the BOINC server for SETI@home. The traces are not application dependent nor specific to SETI@home because they are recorded at the level of the BOINC client (versus the application it executes). In total, our traces capture about 57,800 years of CPU time and 102,416,434 continuous intervals of CPU availability.

Artifacts due to a measurement method or middleware itself are almost inevitable. In our case, artifacts resulted from a benchmark run periodically every five days by the BOINC client. As a result, availability intervals longer than five days were truncated prematurely and separated by about a one minute period of unavailability before the next availability interval. Histograms and mass-count graphs showed that these 5-day intervals were outliers and significantly skewed the availability interval mass distribution. As such, we preprocessed the traces to remove these gaps after five day intervals artificially introduced by our middleware. This minimized the effect of these anomalies in the modelling process.

4 Related Work

This work differs from related in terms of what is measured and what is modelled. In terms of measurements, this work differs from others by the types of resources measured (home versus only within an enterprise or university), the scale and duration (hundreds of thousands over 1.5 years versus hundreds over weeks), and the type of measurement (CPU availability versus host availability). Specifically, the studies [12, 6] focus on host in the enterprise or university setting and may have different dynamics compare to hosts primarily on residential broadband networks. Our study includes both enterprise and university hosts, in addition to hosts at home.

Also, other studies focus on hundreds of hosts over a limited time span [12, 7, 15, 1]. Thus their measurements are of limited breadth and could be biased to a particular platform. The limited number of samples per host prevents accurate modelling at the resource level.

Furthermore, while many (P2P) studies of availability exist [16, 4, 14], these studies focus on host availability, i.e., a binary value indicating whether a host is reachable. By contrast, we measure CPU availability as defined in Section 3. This is different as a host can clearly be available but not its CPU. Nevertheless, host availability is subsumed by CPU availability.

In terms of modelling, most related works [7, 18, 11] focus on modelling the system as a whole instead of individual resources, or the modelling does not capture the temporal structure of availability [3, 11, 13]. Yet this is essential for effective resource selection and scheduling. For instance, in [7], the authors find that availability in wide-area distributed systems can be best modelled with a Weibull or hyperexponential distribution. However, these models consider the system as a whole and are not accurate for individual resources. Even the authors note that their best fitted global distribution is not adequate for modelling individual hosts.

For another instance, in [11], the author models host availability as a fraction of time. However, the temporal structure of availability, which is critical to computational tasks for example, is not captured in the model. A host with that is 99% available but in intervals of 1 millisecond may be of little use to distributed applications.

For a final instance, in our own past work in [13], we conducted a general characterization of SETI@home. However, we ignored availability intervals by taking averages to represent each host's availability. So we used a different data representation with different distance metrics when clustering. Moreover, the purpose was different because we intentionally focused on identifying *correlated* patterns instead of random ones. As such, we did not build statistical models of availability intervals, and did not address issues such as partitioning hosts according to randomness and probability distributions.

5 Experimental Setup

We conduct all of our statistical analysis below using Matlab 2009a on a 32-bit on a Xeon 1.6GHz server with about 8.3GB of RAM. We use when possible standard tools provided by the Statistical Toolbox. Otherwise, we implement or modify statistical functions ourselves.

6 Randomness Testing

A fraction of hosts are likely to exhibit availability intervals that contain trends, periodicity, autocorrelation, or nonstationarity. For these hosts, modelling their availability using a probability distribution is difficult given the change of its statistical properties over time.

Therefore, as the preliminary phase before data analysis and modeling we apply randomness tests to determine which hosts have truly random availability intervals. There are several randomness tests which are divided in two general categories: parametric and nonparametric. Parametric tests are usually utilized when there is an assumption about the distribution of data. As we cannot make any assumption for the underlying distribution of the data set, we adopted non-parametric randomness tests. We conducted four well-known non-parametric tests, namely the runs test, runs up/down test, autocorrelation test and Kendall-tau test [17, 10]. For all tests, the availability intervals of each hosts was imported as a given sequence or time series.

We describe intuitively what they measure. The runs test compares each interval to the mean. The runs up/down test compare each interval to the previous interval to determine trends. The autocorrelation test discovers repeated patterns that differ by some lag in time. The Kendall-tau test compares a interval with all previous ones.

Since the hypothesis for all these tests are based on the normal distribution, we must make sure that there are enough samples for each host, i.e., at least 30 samples, according to [8] and personal communication with a statistician. About 20% of hosts do not have enough samples because of a limited duration of trace measurement. (Measurement for a host began only after the user downloaded and installed the instrumented BOINC client. So some hosts may have only a few measurements because they began using the instrumented BOINC client only moments before we collected and ended the trace measurements.) So we ignored these hosts in our statistical analysis. Finally, we applied all four test on 168751 hosts with a significance level of 0.05.

As there is no perfect test for randomness, we decide to apply all tests and to consider only those hosts that pass all four tests to be conservative. Table 1 shows the fraction of hosts that pass each and all four tests.

While iid hosts may not be in the majority (i.e., 34%), together they still form a platform with significant computing power. In total, volunteer computing systems, which now include GPU's and Cell processors of the Sony PS3, provide a sustained 6.5 PetaFLOPS. The hosts with iid availability thus contribute about 2.2 PetaFLOPS, which is significant.

7 Clustering by Probability Distribution

For hosts whose availability is truly random, our approach was to first inspect the distribution using histograms and the mass-count disparity (see Figures 2(a) and 2(b)). We observe significant right skew. For example, the shortest 80% of the availability intervals contribute only about 10% of the total fraction of availability. The remaining longest 20% of intervals contribute about 90% of the total fraction

Test	Runs std	Runs up/down	ACF	Kendall	All
# of hosts	101649	144656	109138	101462	**57757**
Fraction	0.602	0.857	0.647	0.601	**0.342**

Table 1. Result of Randomness Tests

of availability. The implication for modelling is that we should focus on the larger availability intervals.

But which hosts can be modelled by which distributions with which parameters?

7.0.1 Clustering Background

We use two standard clustering methods, in particular k-means and hierarchical clustering, to cluster hosts by their distribution. K-means randomly selects k cluster centers, groups each point to the nearest cluster center, and repeats until convergence. The advantage is that it is extremely fast. The disadvantages are that it requires k to be specified a priori, the clustering results depend in part on the cluster centers chosen initially, and the algorithm may not converge.

Hierarchical clustering iteratively joins together the two sub-clusters that are closest together in terms of the average all-pairs distance. It starts with the individual data points working its way up to a single cluster. The advantage of this method is that one can visualize the distance of each sub-cluster at different levels in the resulting dendrogram, and that it is guaranteed to converge. The disadvantage is that we found the method to be 150 times slower than k-means.

7.0.2 Distance Metrics for Clustering

We tested several distance metrics for clustering, each metric measures the distance between two CDFs [9]. Intuitively, Kolmogorov-Smirnov determines the maximum absolute distance between the two curves. Kuiper computes the maximum distance above and below of two CDFs. Cramer-von Mises finds the difference between the area of under the two CDFs. Anderson-Darling is similar to Cramer-von Mises, but has more weight on the tail.

Using these distance metrics is challenging when the number of samples is too low or too high. In the former case, we do not have enough data to have confidence in the result. In the latter case, the metric will be too sensitive. A model by definition is only an approximation. When the number of samples is exceedingly high, the distance reported by those metrics will be exceedingly high as well. This is because the distance is often a multiplicative factor of the number of samples.

Our situation is complicated even further by the fact that we have a different number of samples per host. Ideally, when computing each distance metric we should have the same number of samples. Otherwise, the distance between two hosts with 1000 samples will be incomparable to the distance between two hosts with 100 samples.

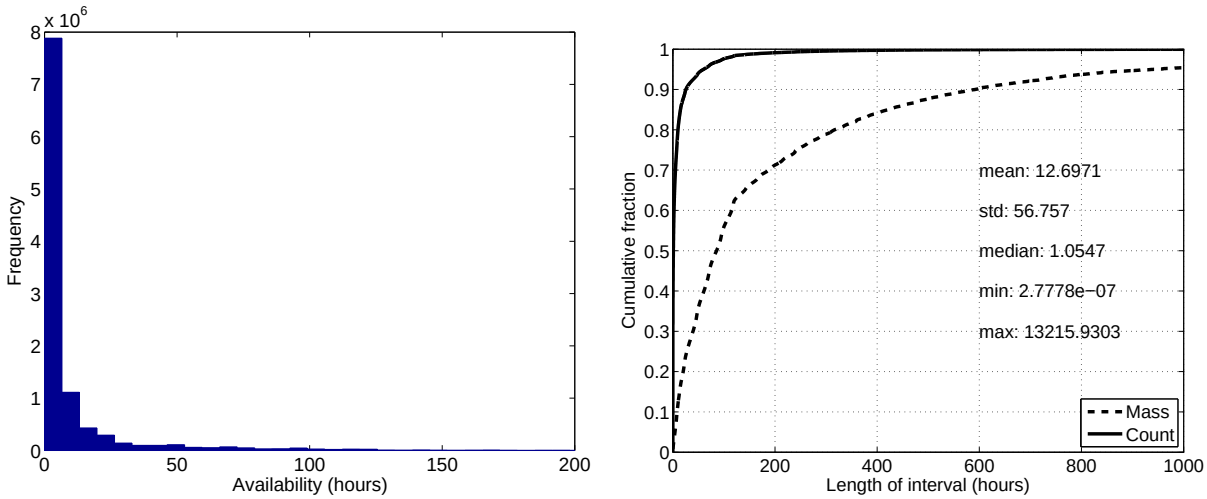
Our solution is to select a fixed number of intervals from each host *at random*. This method was used also in [7], where the authors had the same issue. We also discussed this issues with a statistician, who confirmed that this is an acceptable approach.

However, if we choose the fixed number to be too high, it will excluded too large of a portion of hosts from our clustering. If we choose the number to be too low, we will not have statistical confidence in the result. We choose the fixed number to be 30 intervals as discussed in Section 6. The test statistics corresponding to each distance metric is normally distributed, and so with 30 intervals, one can compute p-values with enough accuracy.

We performed extensive experiments to compare the distance metrics for several cases including for different subsets of iid hosts. We also conducted positive and negative control experiments where the cluster distributions were generated and known a priori. The following conclusions were been found to be consistent across all the cases considered. We found out that Kolmogorov-Smirnov and Kuiper are not very sensitive in terms of distance as they are only concerned with extreme bounds. Cramer-von Mises and Anderson-Darling revealed that they are good candidates of clustering, but Anderson-Darling is engaged with high time and memory complexity. Moreover, Cramer-von Mises is advantageous for right-skewed distributions [9]. Thus we used the Cramer-von Mises as the distance metric when clustering.

7.0.3 Cluster Results and Justification

We found that optimal number cluster was 6. We justify this number of clusters through several means. First, we observed the dendrogram as a result of hierarchical clustering for a random subset of hosts (due to memory consumption and scaling issues). As it can be seen in Figure 3(a) the tree is an unbalanced tree where the height of tree shows the distance between different (sub)clusters. The good separation of hosts in this figure confirmed the advantage of Cramer-von Mises as the distance metric (in comparison to the result with other metrics). The number of distinct groups in this dendrogram where the distance threshold is set to one



(a) Histogram

(b) Mass-count

Figure 2. Distribution of Availability Intervals

reveals that number of clusters should be between 5 to 10.

Second, using the result of hierarchical clustering as a bootstrap, we run k-means clustering *for all iid hosts* and then compute the within-group and between-group distances for various values of k (see Figure 3(b)). We observe that the maximum ratio of inter-cluster over intra-cluster distance is with six clusters. Specifically, for k of 6, the inter-cluster distance between all clusters was maximized relative to other values of k. Also, for k of 6, the distance between all clusters is roughly equal (about 5).

Third, we plotted the EDF corresponding to each cluster for a range of k. In this way, we can observe convergence or or divergence of clusters during their formation.

Fourth, we plotted the EDF corresponding to each cluster. Figure 4 shows good separation of these plots.

7.1 Parameter Estimation for Each Cluster

After cluster discovery, we conduct parameter fitting for various distributions, including the Exponential, Weibull, Log-normal, Gamma, and Pareto. Parameter fitting was conducted using maximum likelihood estimation (MLE). Intuitively, MLE maximizes the log likelihood function that the samples resulted from a distribution with certain parameters.

We also considered using moment matching instead of MLE. However, because the overall distribution is heavily right-skewed, we were concerned that moment matching, which is relatively much faster then MLE, would not produce the most accurate results. Moment matching is also

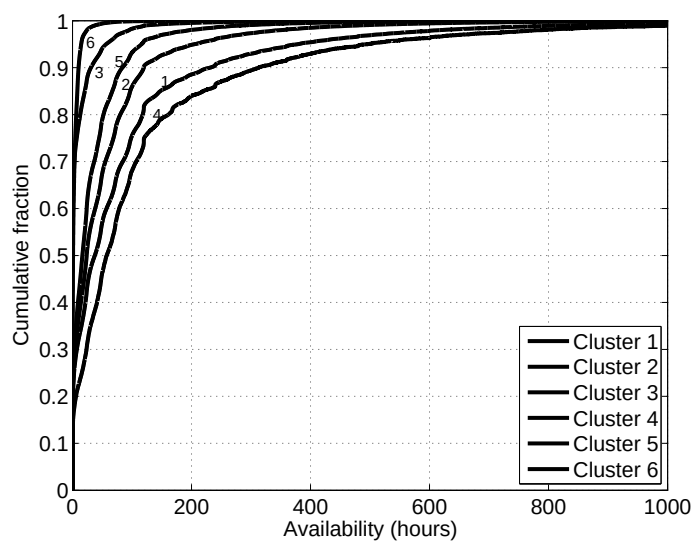
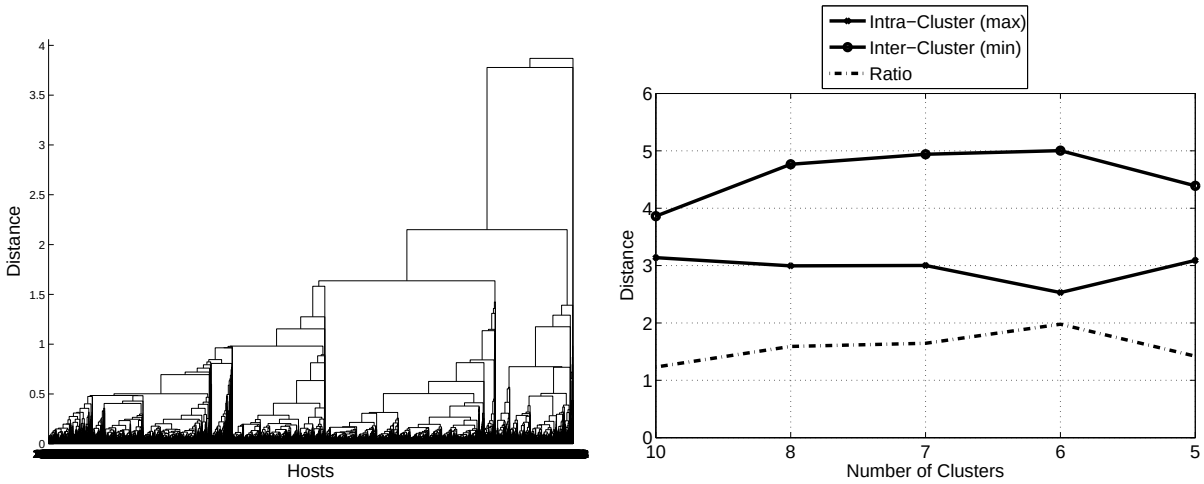


Figure 4. EDF of clusters



(a) Dendrogram of hierarchical clustering for a random subset of hosts

(b) Comparison of distances in clusters

Figure 3. Cluster Comparison

very sensitive to outliers [10].

We measured the goodness of fit (GOF) of the resulting distributions using standard probability-probability (PP) plots as a visual method and also quantitative metrics, i.e., Kolmogorov-Smirnov (KS) and Anderson-Darling (AD) tests. The graphical results which are depicted in Figure 5 revealed that in the most cases the Gamma distribution is a good fit. Also, Exponential distribution has some close fits, specially in cluster 4. These results could allow an approximation that would result in a simple analytical model. Based on the third column of PP-plots, it is obvious that the Pareto is completely far from our underlying distribution, so we do not have a heavy-tailed distribution. However, as in some plots Log-normal and Weibull have a close match, some cluster distributions are likely to be long-tailed.

To be more quantitative, we also reported the p-values of two goodness-of-fit tests. We randomly select a subsample of 30 of each data set and compute the p-values iteratively for 1000 times and finally obtain the average p-value. This method is similar to the one used by the authors in [7], and was suggested to us by a statistician.

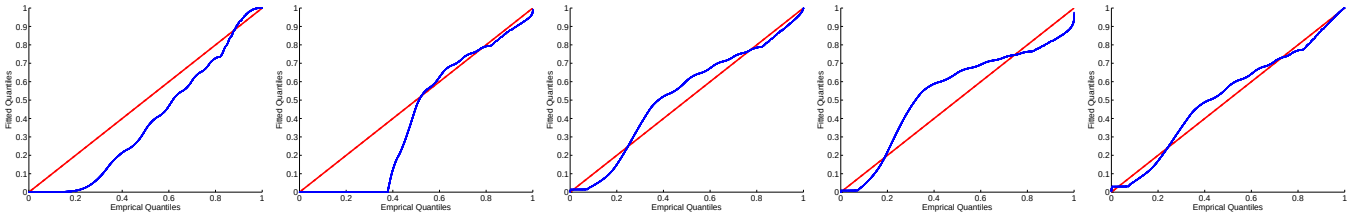
The results of GOF tests are listed in Table 2 where in the each row the best fit is highlighted. These quantitative results strongly confirm the graphical result of the PP-plots. (In the PP-plots, the closer the plots are to the line $y = x$, the better the fit.) We find that the best-fitting distributions of each cluster to differ significantly from the overall distribution over all iid hosts. Thus, clustering was essential for accurate host availability modelling.

To be more precise, Table 3 lists some properties of iid hosts and clusters as well. The first point is the existing heterogeneity in the clusters in terms of members and percentage of total availability. The biggest cluster (i.e, cluster 6) that includes more than 50% of total iid hosts, only contribute 20% of the total availability and also, cluster 3 that is behind cluster 6 and 3 in number of hosts has the highest availability contribution.

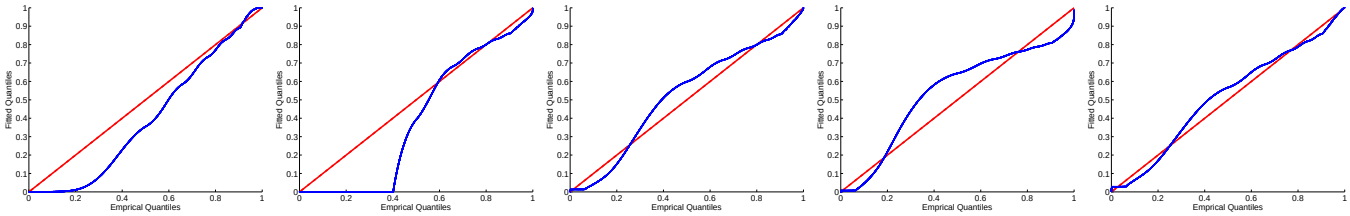
One possible question about the distribution fitting results is why four clusters that follow the Gamma distributions are not in one cluster. The answer is in the last column of Table 3 where the shape and scale parameters of fitting are reported (For Log-normal distribution, the numbers are μ and σ, respectively). These values revealed that all four clusters have almost same shape (see Figure 4) but at a different scale which are very far from each other.

Another question that may be raised here is whether or not we are able to model all hosts as a single distribution, i.e., Gamma distribution with fixed parameters. The answer could is no; the shape and scale parameters for the Gamma distribution fitting for all iid hosts are 0.2442 and 51.9864, respectively, which does not reflect the properties of different distributions listed in Table 3.

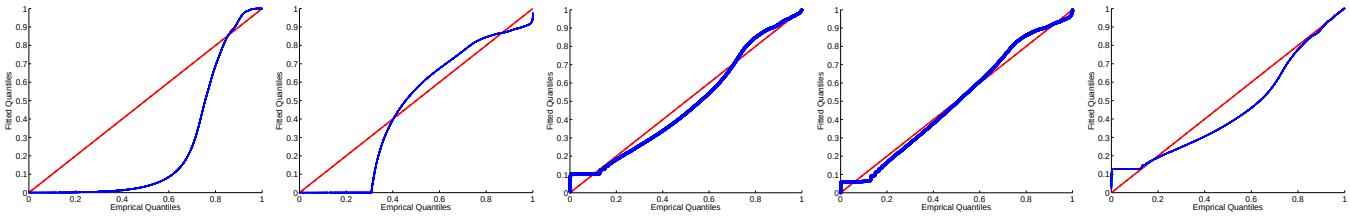
However, based on the graphical and quantitative p-value results, we could say that the Gamma is a good fit for all clusters, so we can use a unique distribution but with different parameters (at least different scales) to model the availability of different hosts. It is worth nothing that the Gamma distribution has very interesting properties in terms of flexibility and generalization and can be easily used to propose an analytical Markov model as well [10].



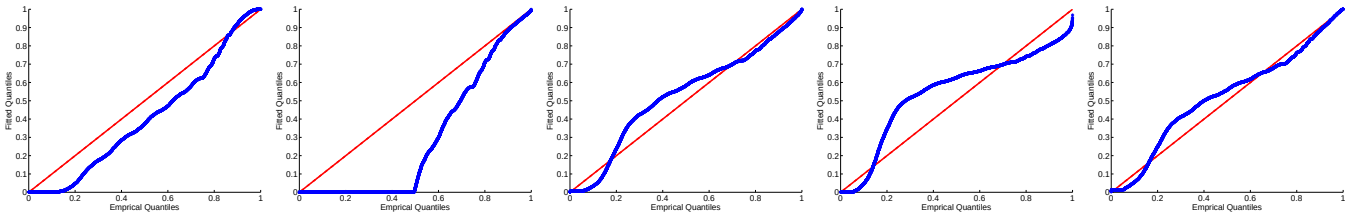
(a) Cluster 1



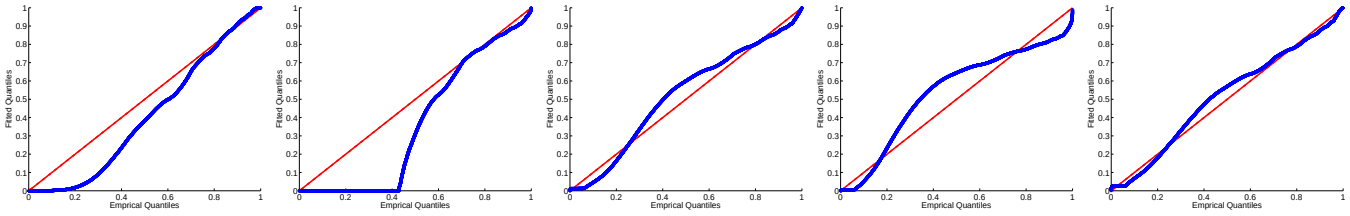
(b) Cluster 2



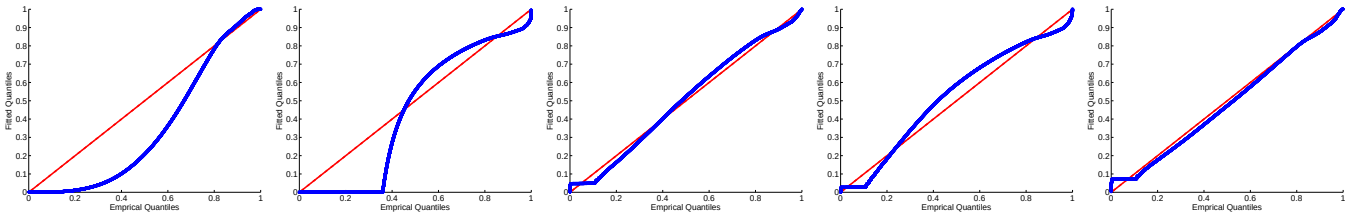
(c) Cluster 3



(d) Cluster 4



(e) Cluster 5

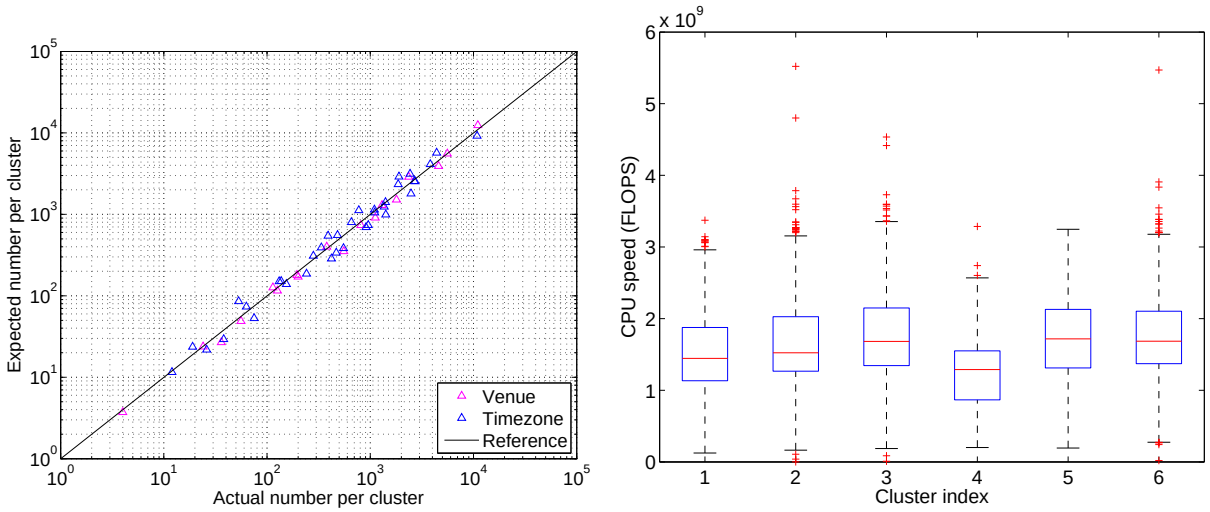


Data sets	Exponential		Pareto		Weibull		Log-Normal		Gamma	
	AD	*KS*	*AD*	*KS*	*AD*	*KS*	*AD*	*KS*	*AD*	*KS*
All iid hosts	0.004	0.000	0.061	0.013	**0.581**	**0.494**	0.568	0.397	0.431	0.359
Cluster 1	0.155	0.071	0.029	0.008	0.466	0.243	0.275	0.116	**0.548**	**0.336**
Cluster 2	0.188	0.091	0.020	0.004	0.471	0.259	0.299	0.128	**0.565**	**0.384**
Cluster 3	0.002	0.000	0.068	0.023	0.485	0.380	**0.556**	**0.409**	0.372	0.241
Cluster 4	0.264	0.163	0.002	0.000	0.484	0.242	0.224	0.075	**0.514**	**0.276**
Cluster 5	0.204	0.098	0.013	0.002	0.498	0.296	0.314	0.153	**0.563**	**0.389**
Cluster 6	0.059	0.016	0.033	0.009	**0.570**	**0.439**	0.485	0.328	0.538	0.467

Table 2. P-value results from GOF tests for all iid hosts and six clusters

Clusters	# of hosts	% of total avail.	Best fit	Parameters	
				shape	*scale*
All iid hosts	57757	1.0	Weibull	0.3787	3.0932
Cluster 1	3606	0.16	Gamma	0.3131	289.9017
Cluster 2	9321	0.35	Gamma	0.3372	161.8350
Cluster 3	13256	0.22	Log-Normal	-0.8937	3.2098
Cluster 4	275	0.01	Gamma	0.3739	329.6922
Cluster 5	1753	0.05	Gamma	0.3624	95.6827
Cluster 6	29546	0.20	Weibull	0.4651	1.8461

Table 3. Some properties of clusters and iid hosts



(a) By venue or timezone

(b) By CPU speed

Figure 6. Clustering by other criteria

7.2 Significance of Clustering Criteria

Our results in the previous section beg the following question: could the same clusters have been found using some other static criteria? In this section we considering cluster formation by static criteria, namely host venue, timezone, and clock rates. (Note that a small fraction of hosts did not have this information specified, and were thus excluded from this analysis.)

We define host venue to be whether a host is used at home, school, or work. This is specified by the user to the BOINC client (though this is not required). Roughly, 23900 are at home, 5500 are at work, and 772 are at school. If our clustering results correspond to those categories, we would expect the distribution of host venue across clusters to be even more skewed.

To measure this, we computed the expected number of hosts of a particular venue for each one of the six clusters identified in Section 7.0.3. The expected number is computed using the global percentage of home, work, and school hosts. The we counted the actual number of hosts of each venue in each cluster.

Figure 6(a) shows the expected number versus the actual number for each venue type over each cluster. The results for larger cluster sizes are more significant. If the clusters correspond to host venues, we would expect large deviations from the $y = x$ line. However, this is not the case as the expected and actual values are similar. Thus, the same clusters would not have resulted from the host venue.

We conducted the same comparison using host timezones. We took counted the number of hosts in each cluster for each of the six largest timezones (in terms of hosts). These timezones in corresponded to Central Europe (17,000 hosts), Eastern North America (11,003 hosts), Central North America (6,077 hosts)), Western North America (4,900 hosts), Western Europe (4280 hosts), and Eastern Asia (2396 hosts).

We then compared this with the expected number, given the number of hosts in total for each of the six timezones. We find again that the corresponding points in Figure 6(a) are close the the line $y = x$. This indicates that the clusters are not a direct result of timezones alone.

We also investigated the relationship between CPU speeds (in FLOPS) and the clusters. Figure 6(b) shows a box-and-whisker plot of the CPU speeds for each cluster. The box represents the inter-quartile range. As this box for each cluster appears in similar ranges, we conclude that the clusters could not have been formed using CPU speeds alone, and that there is little correlation between CPU speeds and the length of intervals.

Nonetheless, one would like an intuition to explain why there was a formation of six clusters. We do not have a precise answer to this question. It could depend very much on the behavior of the user, which is hard to measure. For future work, we could like to conduct a survey of volunteers or to monitor their host processes to classify them by how they use their computer (for example, for gaming, word processing, web surfing). This might shed light on the cause of these clusters.

8 Scheduling Implications

Our study has pinpointed the exact mixture of iid versus non-iid hosts, and when iid, the distribution that best models it and its parameters. The difference between the global distribution for all iid hosts, and the distributions for each individual cluster impacts scheduling accuracy greatly. For example, if one uses the Weibull distribution as the global model, the probability of a host to successfully complete a 24 hour task is less than 20%. By contrast, for hosts in cluster 4, the probability is about 70%. These differences are magnified multiplicatively when computing the probability of task completion with replication or for a series of tasks with dependencies.

Moreover, in terms of multi-job scheduling, an online scheduler could create a multi-level priority queue and channel jobs to hosts depending on their model requirements. For example, in typical volunteer computing systems, there are a mixture of high-throughput and low-latency jobs. In practice, we have observed that the total sum of availability across hosts in large numbers to be relatively constant and tends towards a Normal distribution. High-throughput jobs could then be sent to non-iid hosts as their lack of randomness likely will not affect the aggregate system throughput as a whole.

On the other hand, the jobs that need low response time, such as batches of low-latency tasks or DAG's, can be scheduled on hosts that exhibit iid availability, and the probability of successful task completion can be computed with the discovered model.

9 Conclusions

We considered the problem of discovering availability models for host subsets from a large-scale distributed system. Our specific contributions were the following:

- *With respect to methodology.*
  - We detected and determined the cause of outliers in the measurements. We minimize their effect during the preprocessing phase of the measurements. This is important for others to use the trace data set effectively.
  - We described a new method for partitioning hosts into subsets that facilitates the design of stochas-

tic scheduling algorithms. We use randomness tests to identify hosts with iid availability.
  - For iid hosts, we use clustering based on distance metrics that measure the difference between two probability distributions. We find that the Cramer-von Mises metric is the most sensitive and computationally efficient compared to others.
  - In the process, we describe how we deal with with large and variable sample sizes and the hypersensitivity of statistical tests by taking a fixed number (30) of random samples from each host.
- *With respect to modelling.* We apply the methodology for the one of the largest Internet-distributed platforms on the planet, namely SETI@home.
  - We find that about 34% of hosts have truly random availability intervals.
  - These iid hosts have availability that can be modelled using 6 distributions, in particular, the Gamma, Weibull, and Log-Normal distributions. About 51% of the total availability comes from hosts whose availability follows the Gamma distribution. This is useful as this distribution is short-tailed and can be used easily to construct analytical Markov models.
  - The family of Gamma distributions (with different scales) could be used to model all iid hosts.
  - The discovered distributions from the same family differ greatly by scale, which explains their separation after clustering. The distribution for all hosts with iid availability also differs significant in scale from any of the distributions corresponding to each cluster.

Acknowledgements

We thank Eric Heien for useful discussions. This work has been supported in part by the ANR project SPADES (08-ANR-SEGI-025).

References

[1] A. Acharya, G. Edjlali, and J. Saltz. The Utility of Exploiting Idle Workstations for Parallel Computation. In *Proceedings of the 1997 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 225–234, 1997.

[2] 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.

[3] D. Anderson and 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] R. Bhagwan, S. Savage, and G. Voelker. Understanding Availability. In *Proceedings of IPTPS'03*, 2003.

[5] The berkeley open infrastructure for network computing. `http://boinc.berkeley.edu/`.

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

[7] J. Brevik, D. Nurmi, and R. Wolski. Quantifying Machine Availability in Networked and Desktop Grid Systems. Technical Report CS2003-37, Dept. of Computer Science and Engineering, University of California at Santa Barbara, November 2003.

[8] G. Casella and R. Berger. *Statistical Inference*. Duxbury, 2002.

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

[10] Feitelson. D. *Workload Modeling for Computer Systems Performance Evaluation*. 2009.

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

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

[13] Derrick Kondo, Artur Andrzejak, and David P. Anderson. On correlated availability in internet distributed systems. In *IEEE/ACM International Conference on Grid Computing (Grid)*, Tsukuba,Japan, 2008.

[14] D. Long, A. Muir, and R. Golding. A Longitudinal Survey of Internet Host Reliability. In *14th Symposium on Reliable Distributed Systems*, pages 2–9, 1995.

[15] M. Mutka and M. Livny. The available capacity of a privately owned workstation environment . *Performance Evaluation*, 4(12), July 1991.

[16] S. Saroiu, P.K. Gummadi, and S.D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedinsg of MMCN*, January 2002.

[17] Ying Wang. Nonparametric tests for randomness. Research report, UIUC, May 2003.

[18] J. Wingstrom and H. Casanova. Probabilistic allocation of tasks on desktop grids. In *IPDPS'08*, pages 1–8, 2008.