A Correlated Resource Model of Internet End Hosts

Eric M. Heien, Derrick Kondo, *Member, IEEE,* and David P. Anderson, *Member, IEEE*

Abstract—Understanding and modeling resources of Internet end hosts is essential for the design of desktop software and Internet-distributed applications. In this paper we develop a correlated resource model of Internet end hosts based on real trace data taken from several volunteer computing projects, including SETI@home. This data covers a 5-year period with statistics for 6.7 million hosts. Our resource model is based on statistical analysis of host computational power, memory, and storage as well as how these resources change over time and the correlations among them. We find that resources with few discrete values (core count, memory) are well-modeled by approximations governing the change of relative resource quantities over time. Resources with a continuous range of values are well-modeled by correlated log-normal distributions (cache, processor speed and available disk space). We validate and show the utility of the model by applying it to a resource allocation problem for Internet-distributed applications, and compare it to other models. We also make our trace data and tool for automatically generating realistic Internet end hosts publicly available.

Index Terms—resource model, host model, Internet end host, resource scheduling, Internet computing, volunteer computing

1 Introduction

While the Internet plays a vital role in society, relatively little is known about Internet end hosts and in particular their hardware resources. Obtaining detailed data about hardware resources of Internet hosts at a large-scale is difficult. The diversity of host ownership and privacy concerns often preclude the collection of hardware measurements across many hosts. Internet safeguards such as firewalls make access to end hosts almost impossible and ISPs are reluctant to collect or release data about their end hosts.

Nevertheless, the characteristics and models of Internet end hosts are essential for the design and implementation of any desktop software or Internet-distributed application. Such software or applications include but are not limited to operating systems, web browsers, peer-to-peer (P2P), gaming, multi-media and word-processing applications. These types of models are also needed for Internet-computing research. For instance, works such as [1], [2], [3] developed algorithms for scheduling or resource discovery for distributed applications run across Internet hosts. Assumptions were made about the distribution of hardware resources of these Internet hosts, and the performance of such algorithms is arguably tied to the assumed distributions. Realistic models of Internet resources derived systematically from real-world data are needed to quantify and understand the performance of these algorithms under a range of scenarios.

Our goal in this study is to characterize and model resources of Internet end hosts. Our approach for data collection is to use hardware statistics and measurements from various Internet computing projects, including SETI@home [4], [5], Einstein@home [6], [7], World Community Grid [8], Rosetta@home [9] and Climate Prediction [10].

These are among the largest volunteer computing projects in the world, aggregating millions of volunteered hosts for distributed computation. Using the BOINC framework deployed on these projects, we retrieved hardware data over a 5-year period with statistics for 6.7 million hosts. Our approach for modeling is to investigate statistically the distribution, correlation, and evolution of resources. Our main contributions are:

1) We characterize and statistically model hardware resources of Internet hosts, including the number of cores, host memory, processor cache, floating point/integer speed and disk space. Our model captures the resource mixture across hosts and how it evolves over time. Our model also captures the correlation of resources (for instance memory and number of cores) within individual hosts.
2) We evaluate the utility of our model and show its accuracy in the context of a resource allocation problem involving Internet-distributed computing applications. We demonstrate the improved accuracy of our model compared to other models.
3) We describe the general implications for system design in light our model. We investigate what our model predicts for future hosts and put bounds on the host resources that will be available.
4) We make our data and tool for automated model generation publicly available. This online tool generates realistic sets of Internet hosts using the parameters in this paper or user specified parameters.

The paper is structured as follows. In Section 2 we discuss related work and contrast our contribution. We

- *E.M. Heien and D. Kondo are with INRIA Grenoble Rhone-Alpes, France. E-mail: {eric.heien, derrick.kondo}@inria.fr*
- *D. P. Anderson is with UCBerkeley. E-mail: davea@ssl.berkeley.edu*

then discuss the application context and describe the data collection methodology in Section 3. We introduce details of the model and describe how the resources are modeled in Section 4. We validate the model using statistical techniques and show how it generates realistic sets of hosts for simulations in the supplemental materials (Appendix 5). To demonstrate the effectiveness of our model compared to other methods we perform simulations in the supplemental materials (Appendix 6). Finally, we offer some discussion related to our model and predictions in Section 5 and conclude in Section 6.

2 Related Work

The branches of work related to this paper include Internet network modeling, peer-to-peer (P2P) modeling, desktop benchmarking, and Grid resource modeling.

With respect to Internet network measurement and modeling [11], [12], [13], previous studies tend to focus exclusively on the network of end hosts, and not their hardware resources. Several works such as [14], [15], [16] model specifically residential networks, but omit hardware measurements or models. Also, the scale of those measurements are relatively small on the order of thousands of hosts monitored on the order of months (versus millions of hosts on the order of years). P2P research [17], [18] has focused primarily on application-level network traffic, topology, and its dynamics. Again, hardware measurements and models are missing.

For desktop benchmarking there are a handful of programs such as XBench [19], PassMark [20] and LMBench [21]. However, these benchmarks are generally designed for a particular operating system and set of tests - often oriented towards game graphics performance - making it difficult to compare across platforms. These benchmarks are also generally run only once on a system, limiting their usefulness in predicting how total resource composition changes over time. In contrast, our work uses benchmarks run on a variety of platforms over a long time period.

Some previous works investigated modeling clusters or computational Grids [22], [23], [24]. These works differ from ours in terms of the resource focus of the model, the host heterogeneity and the evolution and correlation of resources over time. Also, most Grid resource models are based on data from many years ago and may no longer be valid for present configurations.

The closest work described in [25], [26] gives a general characterization of Internet host resources. However, neither statistical models nor the evolution and dynamics of Internet resources are investigated. These works focus on handling heterogeneous host resources rather than the specific modeling of the resources.

3 Data Collection Method

While there are an infinite range of host resources to monitor and model, we select only those that are the most relevant for Internet-distributed computing. One class of Internet-distributed computing is distributed peer-to-peer (P2P) file sharing [17], [18], [27]. Another important class is volunteer distributed computing. As of January 2011, volunteer computing provides over 4 PetaFLOPS of computing power [4], [28] for over 68 applications from a wide range of scientific domains (including climate prediction, protein folding, and gravitational physics). These projects have produced hundreds of scientific results [29] published in the world's most prestigious conferences and journals. We use these types of applications to drive what we model.

The hosts in this study were measured using the BOINC (Berkeley Open Infrastructure for Network Computing) [30] client software, and participated in one or more volunteer computing projects between January 1, 2006 and January 1, 2011. These projects include SETI@home, Einstein@home, World Community Grid, Rosetta@home and Climate Prediction (brief descriptions of these are in Table 1). These projects provide a good approximation of types of hosts likely to be available for large-scale Internet computing applications. The model developed in this paper uses host data from January 1, 2006 to January 1, 2010. We validate this model by predicting host composition until January 1, 2011.

In BOINC projects, hosts perform work in a master-worker style computing environment where the host is the worker and the project server is the master. Host resource measurements occur every time the host contacts the server; this allows the server to allocate the appropriate work for the available host resources. The host resource measurements are recorded on the server and periodically written to publicly available files.

4 Modeling

In this section we discuss the model of host resources - how it is defined and how we model the host resources and their change over time. For details of the model measurements we refer readers to Appendix 1 in the supplemental materials. Section 4.1 provides a statistical overview of the host resources over time. Since resources may be correlated we begin the model building process by examining correlation between resources in Section 4.2. In Sections 4.3 through 4.8 we perform detailed analysis of each resource and build a predictive correlated model of host cores, cache, memory, computing speed and disk storage. We also provide supplemental analyses of processor and OS composition in Appendix 3 and we briefly examine host GPU resources in Appendix 4.

4.1 Host Overview

First we present an overview of the active hosts and their resources. Figure 1 shows a probability density function (PDF) and cumulative distribution function (CDF) of host lifetimes, where the lifetime is defined as the time between the first and last connection of the host to the server. Using a maximum likelihood of fit estimation we find the host lifetime distribution fits well to a Weibull

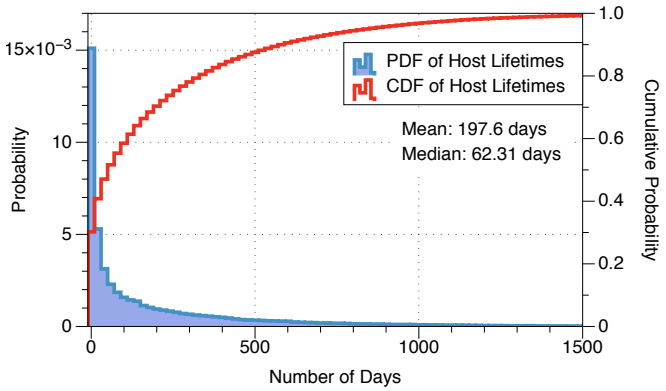


Fig. 1. Host lifetime distribution over all projects.

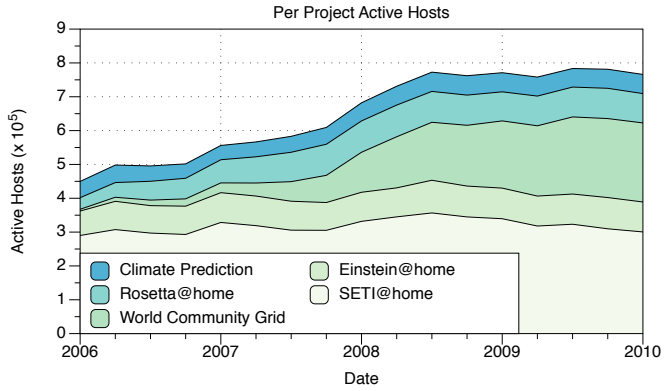


Fig. 2. Active host distribution across projects.

distribution with parameters $k = 0.561, \lambda = 119.7$, which indicates that hosts have a decreasing dropout rate.

Figure 2 shows the change in active hosts over time, broken down among the five projects. SETI@home shows a relatively constant number of active hosts over the measurement period, while other projects showed slight growth. The largest growth comes from the World Community Grid project which switched to BOINC from another computing platform between 2007-2008. Overall, the active host data set size grows from 450,000 hosts in 2006 to nearly 800,000 hosts by 2010.

Table 1 shows further details about each project and the total number of hosts recorded by the project. Our model utilizes data from over 6.7 million hosts, split among projects in a range of 36% from SETI@home to 6% from Climate Prediction. These projects comprise over two-thirds of publicly available host data.

TABLE 1
Project host statistics.

Project Name	Description	Total Hosts
SETI@home	Radio signal analysis	2,426,535
Einstein@home	Gravitational wave search	1,879,216
World Community Grid	Disease-related computation	1,102,539
Rosetta@home	Protein structure prediction	930,744
Climate Prediction	Long-term climate prediction	398,430
Total		6,737,464

Some host data values may be questionable due to storage/transmission errors or modification of the client resource checking function. In this work, we discard hosts which report more than 128 cores, 128 GB memory, 64 MB cache, 10^5 Whetstone MIPs, 10^5 Dhrystone MIPs, 16 TB available disk space, or negative values for any resource. Based on these criteria we discard 26,190 hosts (0.39% of total). For further details on host resources and how they changed over time, we refer the reader to Appendix 2 in the supplementary materials.

Since the goal of this paper is to understand the composition and dynamics of a wide range of Internet hosts, we feel it is reasonable to use the combined data set for analysis in the rest of the paper (unless otherwise noted). This smooths the differences in project hosts and gives a better picture of overall host resource characteristics.

4.2 Resource Correlations

To guide us in creating the model of host resources, we first examine the correlations between different resources. All resources will tend to improve together as technology advances over time. Also, users will tend to purchase systems with correlated resource characteristics, for example, a system with many cutting edge cores will also tend to have a greater amount of memory. Therefore our model should include these correlations to realistically capture the characteristics of hosts.

Visual inspection of the data shows linear correlation between certain resources. In the supplemental file, Figure 12 shows graphs of the normalized coefficient of correlation (the Pearson correlation coefficient) between resources at different times. Several things are apparent from this analysis. First, the number of cores increases in correlation with other resources over time, except for per-core-memory. This is because initially most hosts are single core machines, meaning the single core will be a poor predictor of other resources. As multicore machines become common, the presence of multiple cores should be better correlated with progress in other resources.

Per-core-memory has different correlations with different resources, but we find that this correlation tends to stay fairly constant over time. For example, floating point performance is slightly correlated with per-core-memory at a constant level ($r \approx 0.3$) during the entire period, though the correlation with total memory rises from $r \approx 0.3$ to $r \approx 0.55$. Similarly, integer performance correlation with per-core-memory is mostly constant ($r \approx 0.3$) though the correlation with memory rises from $r \approx 0.25$ to $r \approx 0.55$. However, per-core-memory has virtually no correlation with the number of cores.

Cache correlation is uneven through the data period because the BOINC client measured cache on only a few OSes prior to April 2008. We therefore only use the correlation values after this in building our model. From April 2008, cache stays constantly correlated with performance around $r \approx 0.4$. It has weak correlation with disk space, and little correlation with per-core-memory.

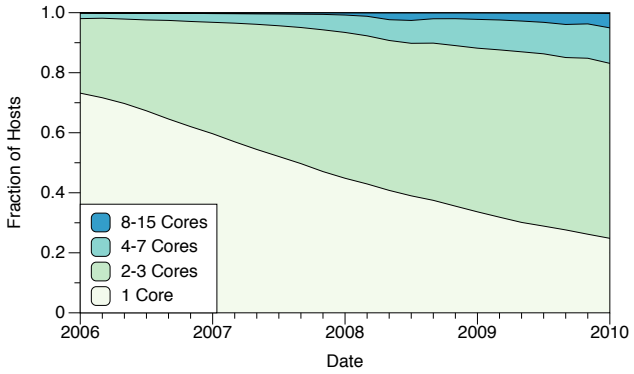


Fig. 3. Fraction of hosts with different numbers of cores.

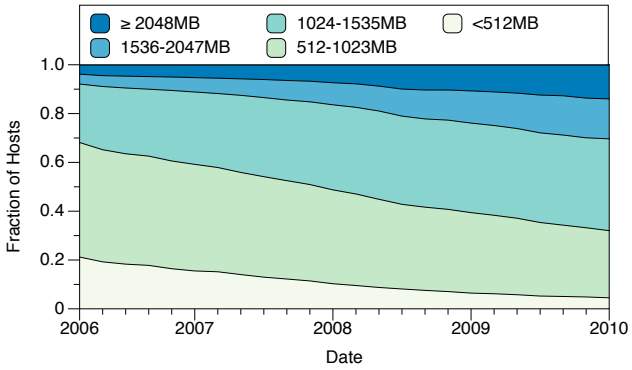


Fig. 4. Fraction of hosts with different per-core-memory.

Performance of integer and floating point benchmarks are also well correlated with each other to a slightly increasing degree ($r \approx 0.7$ to $r \approx 0.8$). This is likely due to advances in processor technology and speed which tend to improve both floating point and integer performance. Unsurprisingly, available disk space and total disk space are well correlated at a constant level ($r \approx 0.75$). Free disk space is slightly correlated with other resources with r values ranging from $r \approx 0.1$ to $r \approx 0.4$. Total disk space is only slightly better correlated with other resources.

4.3 Modeling Multicore

In recent years processor manufacturers have started increasing the number of cores on a processor rather than exclusively increasing the speed of the individual cores. This trend is seen in Figure 3, which shows the fraction of hosts with different numbers of cores over time. In 2006, the ratio of 1 core machines to 2 core machines was 2.9 to 1, however, by 2010 the ratio inverted to 1 to 2.3 and 17% of hosts had 4 or more cores.

Since the number of cores on a host is a discrete value, we are limited in the types of probability distributions we can use. For the model of multicore on a host, we use a discrete probability distribution where the number of cores must be a power of 2. Although there are systems available with non-power-of-two core counts, we ignore them since they comprise less than 0.3% of hosts in our data set. As processors with more cores are introduced to the marketplace, their number will increase relative to processors with fewer cores then decrease relative to processors with even more cores. To model this, we examine the history of the ratio of 1, 2, 4, 8 and 16 core hosts to each other since 2006.

We first attempted to fit core counts to a truncated discrete log-normal distribution but found poor fit. In the supplemental file, Figure 10 shows a logarithmic plot of the core ratios from 2006-2010. To avoid noisiness from few samples, we remove parts of the data where less than 0.3% of the hosts were of the specified core count (before June 2008 for 16 core hosts). The black lines show the actual ratios from the data set and the red dashed lines show the best fit from 2008 to 2010. For example, in 2006 there were roughly 14.5 2-core hosts for every 4-core host, but by 2010 this ratio had dropped to 4.9 2-core hosts for every 4-core host. We found that the logarithm of the relative fractions of each of these is well-modeled by the function $a(year - 2006) + b$. The values of a and b which best fit the data from 2008-2010 are shown in Table 2. The calculated r values show that the fitted curve has a reasonably good match with the data. Therefore, we model the logarithm of the number of cores in a host as a ratio governed by a linear function.

4.4 Modeling Memory

The available memory per host is also increasing over time as shown in Figure 9 in the supplemental material. However, the correlation results in Figure 12 indicate some correlation ($r \approx 0.45$ to $r \approx 0.7$) between the number of cores and amount of memory. Rather than trying to model host memory as a function of the cores, we instead model per-core-memory and multiply the results by the number of cores. This makes intuitive sense - a host with 512 MB of RAM is more likely to have 1 core rather than 8 cores (which would be only 64 MB of RAM per core). This is also supported by the correlation analysis in Section 4.2, which showed that although the total memory is correlated with the number of cores, the amount of per-core-memory has nearly zero correlation and can therefore be generated independently.

First we examine the per-core-memory and how it changes over time. In the supplemental material, Figure 11 shows distributions of per-core-memory at three points in time. This figure shows a clear trend of per-core-memory increasing over time. The fraction of hosts with 256MB or less per core drops from 21% to 4% of the total from 2006 to 2010, while the fraction of hosts with 1024MB per core rises from 23% to 37% and hosts with 2048MB per core rise from 3% to 12%. Over 90% of the per-core-memory values are in the set of (256, 512, 768, 1024, 1536, 2048) MB. To simplify the model, we use these values to calculate the memory on a host.

Figure 4 shows the fraction of hosts with different amounts of memory per core and how this changes over time. Similar to multicore counts, we find that the ratios

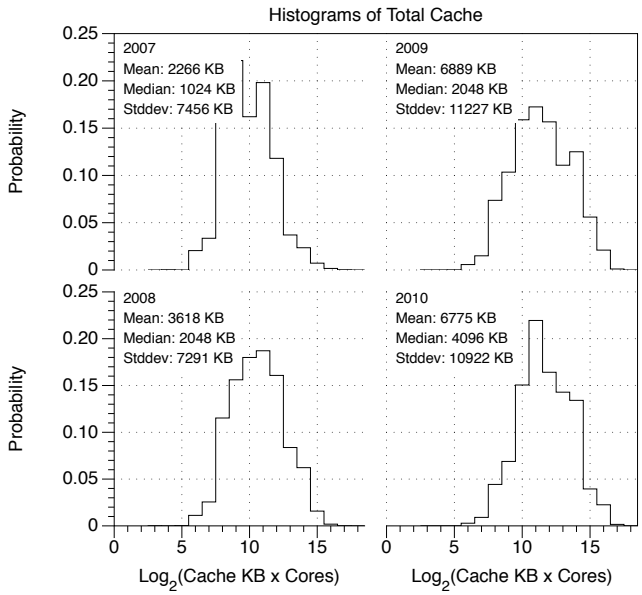


Fig. 5. Histograms of total cache.

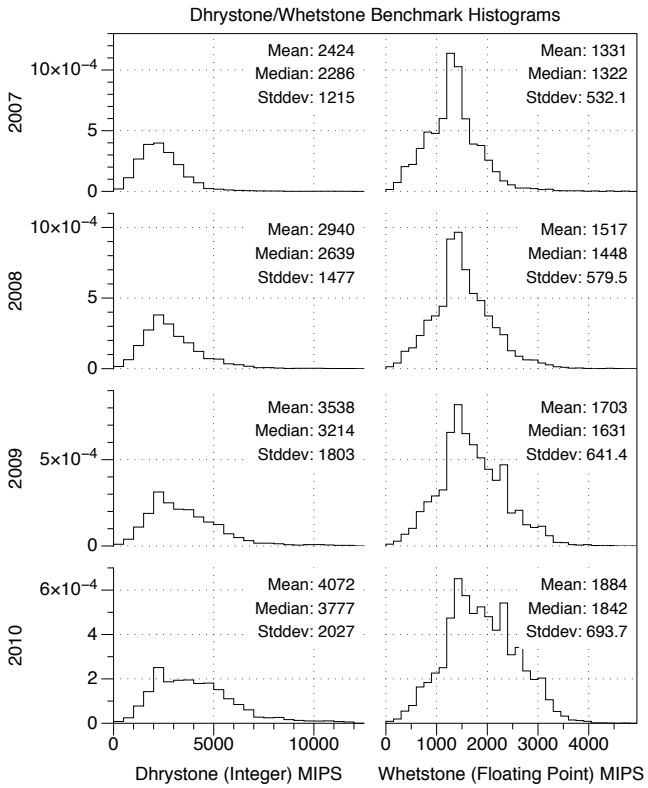


Fig. 6. Histograms of benchmark performance over time.

of host per-core-memory are best modeled by a linear fit $a(year-2006)+b$ of the logarithm of the ratio. The values for this fit are given in Table 2. The correlation coefficient r indicates the values match the data very well except for small memory amounts. It is worth noting that we discard some intermediate per-core-memory values (e.g., 1280MB, 1792MB, etc). The accuracy of the model could therefore be improved by including these values, though at a cost of increased complexity.

4.5 Modeling Processor Cache

Figure 5 shows histograms of the base-2 logarithm of host total cache sizes at four time periods. Although these appear to be best fit by a log normal distribution, we used the Kolmogorov-Smirnov (KS) test [31] to confirm which distribution is best. This test calculates a p-value to determine if the empirical data did not come from a proposed distribution. The KS-test is sensitive to slight discrepancies in large data sets, so to calculate p-values we took the average p-value of 100 KS tests each using a randomly selected subset of 20 values (since the cache sizes are discrete values, large subset samples will significantly degrade the KS-test). This subsampling method is a standard method also used in [32], [33]. We compared our data to 7 distributions - normal, log-normal, exponential, Weibull, Pareto, gamma and log-gamma. The results of this show that the log-normal distribution fits the total cache data best with average p-values ranging from 0.23 to 0.37 at different times in the data set. A gamma distribution also fits reasonably well (p-values from 0.07 to 0.34).

After confirming that a log-normal distribution is most suitable, we model the parameters of this distribution using data starting from April 2008. The base-2 logarithm mean value of the distribution is well-modeled by a linear function of the form $a(year - 2006) + b$ with $a = 0.0426$ and $b = 12.70$ with $r = 0.3171$. The σ parameter can be modeled with a similar equation having $a = 0.4143$ and $b = 25.92$ with $r = 0.5362$.

4.6 Modeling Processor Speed

Next we develop a model for host computational speed in terms of Dhrystone and Whetstone benchmark performance. Figure 6 shows probability density functions of the Dhrystone and Whetstone MIPS performance at four times in the data set. The data for both benchmarks appears to fit a normal distribution, but we again use the KS-test (with size 50 sample subsets) to measure the fit of different distributions. In this case, the Dhrystone speeds actually fit a log-normal distribution (p-values from 0.29 to 0.52) better than a normal distribution (p-values from 0.15 to 0.33). The Whetstone speeds also fit a log-normal distribution (p-values from 0.20 to 0.40) better than a normal distribution (p-values from 0.13 to 0.40). Therefore, we use log-normal distributions to model processor integer and floating point speeds.

Again, we must model the change in the distribution parameters over time. Using the same strategy as for cache, we model the mean and variance of Dhrystone and Whetstone MIPS with parameters shown in Table 2.

4.7 Modeling Available Disk Space

Finally we develop the model for available disk space on a host. Figure 7 shows the probability density functions

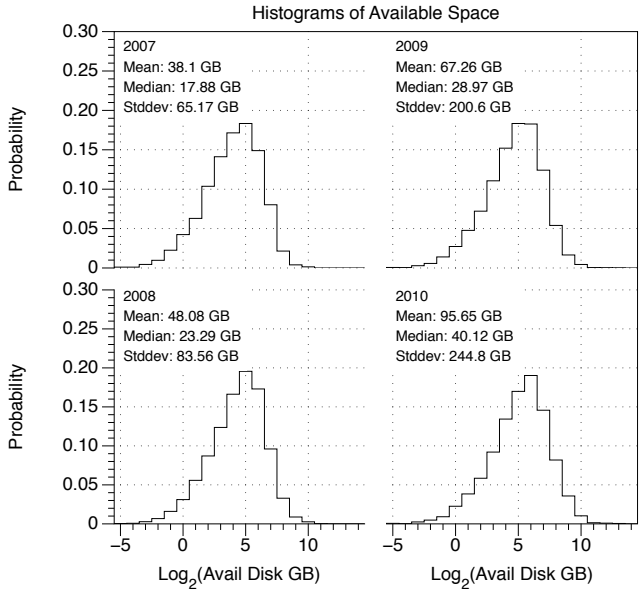


Fig. 7. Histograms of available disk space over time.

of the logarithm of available disk space on active hosts at four times. These distributions are smooth and appear to fit well to a normal distribution meaning disk space should fit to a log-normal distribution.

Using the KS-test again (with size 50 samples due to the continuous nature of the distribution) we find that log-normal is the best fitting distribution for available disk space with p values from 0.2 to 0.51. Therefore we model available disk space as an independent log-normal distribution with mean and variance calculated using the exponential law with values from Table 2. It is worth noting why we chose to model available disk space rather than total disk space. The main reasons are: 1) the distribution of total disk space is highly irregular and difficult to model and 2) applications using Internet computing resources will generally be restricted by available disk space rather than total space.

4.8 Modeling Resource Correlation

In the previous sections we developed models for each resource based exclusively on analysis of that resource data. However, as shown previously there are correlations of varying degrees between resources that should be accounted for in the model. Here we explain how these correlations are added to the model.

To properly capture the resource correlations found in Section 4.2, we create correlated statistics using a common method involving the Cholesky decomposition. Using a line of best fit on the correlation plots in Figure 12, we capture the time dependence of the correlation (if any). We first take a matrix R of the correlation coefficients between per-core-memory, total cache, Dhrystone performance, Whetstone performance and free disk space from Figure 12 and using $t = (year - 2006)$ to model time variance in the correlations. Since cache measurements are sparse prior to April 2008, we use only the data from after April 2008 to estimate cache correlation with other resources. We ignore time variance if the yearly change in correlation was less than 0.02.

$$R = \begin{bmatrix} 1 & 0.005 + 0.032t & 0.303 & 0.309 & 0.123 \\ - & 1 & 0.442 & 0.436 & 0.267 \\ - & - & 1 & 0.840 & 0.195 + 0.031t \\ - & - & - & 1 & 0.136 + 0.041t \\ - & - & - & - & 1 \end{bmatrix}$$

The matrix is symmetric ($R_{ij} = R_{ji}$) so for clarity we replace duplicate entries with a dash. We apply the Cholesky decomposition to get matrix U. For example with $year = 2011$ the decomposition becomes:

$$U = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0.166 & 0.986 & 0 & 0 & 0 \\ 0.303 & 0.397 & 0.866 & 0 & 0 \\ 0.309 & 0.391 & 0.689 & 0.535 & 0 \\ 0.123 & 0.251 & 0.244 & 0.074 & 0.925 \end{bmatrix}$$

We take a vector V of five values randomly selected from a normal distribution with mean 0 and variance 1. $V_C = VU$ gives a vector of five normally distributed random values correlated by the values in R. $V_C[1]$ is converted from a normal distribution to a uniform distribution and used to select the per-core-memory, while $V_C[2]$ through $V_C[5]$ are renormalized to the predicted mean and variance of the cache, benchmarks and disk space. Using this method we are able to generate hosts with similar resource correlations as in the actual data.

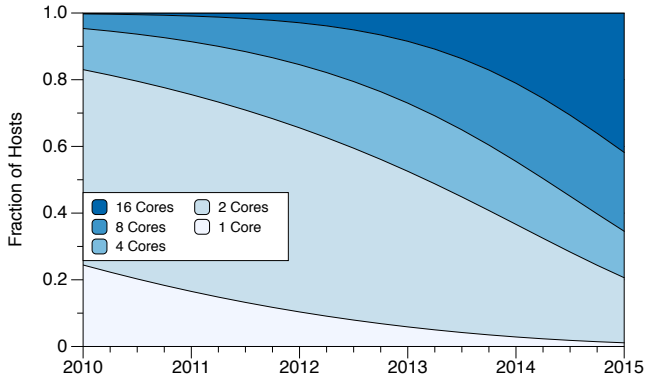
5 Discussion

For any sort of modeling, the question of model accuracy arises, especially for predictive models. Due to space limitations, we include the analysis of model accuracy in the supplemental materials (Appendix 5 and Appendix 6) Naturally it is difficult to verify the correctness of a predictive model before the actual data is available. In this section we discuss model based prediction and possible issues with the future accuracy of our model.

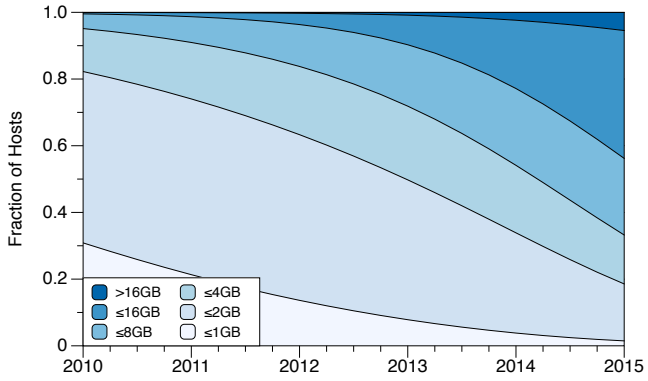
5.1 Model Based Prediction

Given the equations of resource ratios from Section 4 we can make predictions about how the host resource composition will change in the future. Note this is not just a prediction of new technology, which can be estimated from manufacturer roadmaps, but rather a prediction of total active host composition, with both newer and older machines, as well as their resource correlations.

Figure 8(a) shows the predicted distribution of host cores until 2015. There are several notable aspects of this prediction. First, the number of single core hosts decreases to a negligible fraction ($\approx 1\%$), as one would expect due to part failure and decreasing usefulness of older machines. Second, there are still many 2 core hosts which comprise roughly 20% of the total by 2015. However, the largest number of hosts (42%) have 16 or more cores by this time. We also notice that the proportion



(a) Predicted Future Multicore Distribution



(b) Predicted Future Host Total Memory Distribution

Fig. 8. Model based predictions up to 2015.

of 4 and 8 core hosts remains roughly constant through time. Based on our model we predict the mean number of cores per host in 2015 to be 9.5 - significantly higher than the number of 4.8 obtained by extrapolating the mean value from Figure 9.

Figure 8(b) shows the predicted total host memory distributions until 2015. This prediction indicates an average of 9.7 GB per host by 2015 - close to the value of 10.6 GB found by extrapolating values in Figure 9. Using the values from Table 2 we predict the (mean, standard deviation) of cache KB as (8680, 29011), Dhrystone MIPS as (9266, 4697), Whetstone MIPS as (3083, 810) and available disk space GB as (425.5, 1039.6) in 2015.

5.2 Model Accuracy

In Section 5.2 we validated our model using standard statistical methods. However, it is also worth discussing our model in regards to anticipated trends such as the shift to energy efficient portable devices, many-core processors, etc. to guess how valid it will remain in the future. We cannot predict the influence or capabilities of disruptive technologies such as DNA or quantum computing so we ignore these.

It is unclear how far consumer demand will be able to drive research, development and manufacturing of faster and better components. Supercomputing and similar high performance applications will always demand more memory and speed to perform high resolution simulations, analyze larger data sets at faster speeds, etc. In this way, our model should fit well to Grids and similar large scale computing resources. However, home consumers are tending to move to portable devices which put greater value on minimizing power usage and cost than maximizing processing speed or capacity. In regards to modeling the computational and storage potential of these devices our model may overestimate future resource capacity.

In regards to GPU based computing, it is unclear how fast consumer based resources will improve. In this case, improvements in speed and memory tend to be driven by gaming applications. Increased speed is always beneficial to gaming realism so it is reasonable to expect this to increase, but given limited display resolutions there is less benefit to increasing available video memory. This is borne out in our GPU analysis in Section Appendix 4 where we find a much slower increase in video memory compared to main host memory.

There do not appear to be insurmountable problems in the near future with regards to increasing cores, memory or disk space beyond the issues of economic feasibility. The improvement of individual core processing speeds may slow down or even reverse as focus moves to increasing the number of cores and improving battery life. In the future our model may be improved by measuring total processing capability (speed x number of cores) rather than modeling each resource separately. Required memory and disk space should continue to rise at roughly the same rate due to the increasing requirements of multimedia and similar data.

6 Conclusion

Resource models of Internet end hosts are critical for the design and implementation of desktop software and Internet-distributed applications. In this paper we derive such a resource model using hardware traces of 6.7 million hosts on the Internet from several volunteer computing projects.

Our main contributions are as follows:

1) We determine a statistical model of the hardware resources of Internet hosts, namely, the number of cores, memory, cache, floating point and integer speeds, and available disk space. This model captures:
   a) the correlations among resources and the change in correlation over time
   b) the evolution of resources over time (in particular, trends in the fraction of hosts with a certain number of cores or memory)

   Table 2 shows a condensed version of the model developed in this paper. This includes the resources described by the model, how they are derived and the a and b values used to generate them.
2) We evaluate model accuracy in a resource allocation problem for Internet-distributed applications.

TABLE 2
Summary of Model Parameters.

Resource	Value	Method	a	b
Cores	Log_2(1:2 Cores)	Relative Ratio	-0.5757	1.041
	Log_2(2:4 Cores)	Relative Ratio	-0.3522	3.656
	Log_2(4:8 Cores)	Relative Ratio	-0.4520	3.305
	Log_2(8:16 Cores)	Relative Ratio	-0.9751	7.954
Mem/Core	Log_2(256:512MB)	Relative Ratio	0.1540	-5.703
	Log_2(512:768MB)	Relative Ratio	-0.3276	-0.9939
	Log_2(768MB:1GB)	Relative Ratio	-0.3265	2.651
	Log_2(1GB:1.5GB)	Relative Ratio	3.98	-0.1367
	Log_2(1.5GB:2GB)	Relative Ratio	1.51	-0.0925
	Log_2(2GB:4GB)	Relative Ratio	0.0156	0.2142
Cache	Log_2(Mean KB)	Lognormal Dist.	0.0426	12.70
	Log_2(Variance)	Lognormal Dist.	0.4143	25.92
Dhrystone	Log_2(Mean MIPS)	Lognormal Dist.	0.2242	11.16
	Log_2(Variance)	Lognormal Dist.	0.4217	20.60
Whetstone	Log_2(Mean MIPS)	Lognormal Dist.	0.1378	10.35
	Log_2(Variance)	Lognormal Dist.	0.0737	18.66
Disk Space	Log_2(Mean GB)	Lognormal Dist.	0.4179	4.972
	Log_2(Variance)	Lognormal Dist.	0.6915	13.82

Compared with naive models and Grid resource models, our model is up to 130% more accurate.

3) Our resource trace data, and tools for automated model generation are available publicly at: http://fta.inria.fr/res/

There are several ways our model could be expanded. First, the model of resources could be tied to models of network topology, traffic, or host availability, which would be useful for testing scheduling algorithms. Resources could be added by improving the measurements in BOINC, as well as other CPU benchmarks. Alternative models for resources could also be investigated, such as economic models which estimate host characteristics based on consumer spending habits and expected component costs. Finally, the use of GPUs for high performance computing is becoming common, so with more data an accurate GPU model could be developed.

ACKNOWLEDGMENTS

This work has been supported in part by the ANR project Clouds@home (ANR-09-JCJC-0056-01). Many thanks to Rom Walton and Jean-Marc Vincent for advice.

REFERENCES

[1] I. Al-Azzoni and D. G. Down, "Dynamic scheduling for heterogeneous desktop grids," *Journal of Parallel and Distributed Computing*, vol. 70, no. 12, pp. 1231–1240, 2010.
[2] C. Anglano and M. Canonico, "Scheduling algorithms for multiple bag-of-task applications on desktop grids: A knowledge-free approach," in *IPDPS*, 2008, pp. 1–8.
[3] D. Zhou and V. M. Lo, "Wavegrid: a scalable fast-turnaround heterogeneous peer-based desktop grid system," in *IPDPS*, 2006.
[4] D. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer, "Seti@home: an experiment in public-resource computing," *Communications of the ACM*, vol. 45, no. 11, Nov 2002.
[5] "Seti@home website." [Online]. Available: http://setiathome.berkeley.edu/
[6] B. P. Abbott and et al., "Einstein@home search for periodic gravitational waves in early s5 ligo data," *Physical Review D*, vol. 80, p. 42003, Aug 2009.
[7] "Einstein@home website." [Online]. Available: http://einstein.phys.uwm.edu/
[8] "World community grid website." [Online]. Available: http://www.worldcommunitygrid.org/
[9] "Rosetta@home website." [Online]. Available: http://boinc.bakerlab.org/rosetta/
[10] "Climate prediction website." [Online]. Available: http://climateprediction.net/
[11] S. Floyd and E. Kohler, "Internet research needs better models," *Computer Communication Review*, vol. 33, no. 1, pp. 29–34, 2003.
[12] "The Cooperative Association for Internet Data Analysis," http://www.caida.org.
[13] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *SIGCOMM*, 1999, pp. 251–262.
[14] C. R. S. Jr. and G. F. Riley, "Neti@home: A distributed approach to collecting end-to-end network performance measurements," in *PAM*, 2004, pp. 168–174.
[15] Y. Shavitt and E. Shir, "Dimes: let the internet measure itself," *Computer Communication Review*, vol. 35, no. 5, pp. 71–74, 2005.
[16] M. Dischinger, A. Haeberlen, P. K. Gummadi, and S. Saroiu, "Characterizing residential broadband networks," in *Internet Measurement Comference*, 2007, pp. 43–56.
[17] S. Saroiu, P. Gummadi, and S. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Proceedinsg of MMCN*, January 2002.
[18] J. Chu, K. Labonte, and B. Levine, "Availability and locality measurements of peer-to-peer file systems," in *Proceedings of ITCom: Scalability and Traffic Control in IP Networks*, July 2003.
[19] "Xbench," http://www.xbench.com/.
[20] "PassMark," http://www.passmark.com/.
[21] "LMBench - Tools For Performance Analysis," http://www.bitmover.com/lmbench.
[22] Y.-S. Kee, H. Casanova, and A. Chien, "Realistic modeling and synthesis of resources for computational grids," *SC '04: Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, Nov 2004. [Online]. Available: http://portal.acm.org/citation.cfm?id=1048933.1049999
[23] A. Sulistio, U. Cibej, S. Venugopal, B. Robic, and R. Buyya, "A toolkit for modelling and simulating data grids: an extension to gridsim," *Concurrency and Computation: Practice & Experience*, vol. 20, no. 13, Sep 2008.
[24] D. Lu and P. A. Dinda, "Synthesizing realistic computational grids," in *SC*, 2003, p. 16.
[25] D. Anderson and G. Fedak, "The computational and storage potential of volunteer computing," *Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on*, vol. 1, pp. 73– 80, 2006.
[26] D. Anderson and K. Reed, "Celebrating diversity in volunteer computing," *System Sciences, 2009. HICSS '09. 42nd Hawaii International Conference on*, pp. 1 – 8, 2009.
[27] R. Bhagwan, S. Savage, and G. Voelker, "Understanding Availability," in *Proceedings of IPTPS'03*, 2003.
[28] S. Larson, C. Snow, M. Shirts, V. Pande, and V. Pande, "Folding@ home and genome@ home," *Using Distributed Computing to Tackle Previously Intractable Problems in Computational Biology*, 2004.
[29] "BOINC Papers," http://boinc.berkeley.edu/trac/wiki/BoincPapers.
[30] D. Anderson, "Boinc: a system for public-resource computing and storage," *Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on*, pp. 4– 10, 2004.
[31] Chakravarti, Laha, and Roy, *Handbook of Methods of Applied Statistics, Volume I*. John Wiley and Sons, 1967.
[32] B. Javadi, D. Kondo, J. Vincent, and D. Anderson, "Mining for statistical models of availability in large-scale distributed systems: An empirical study of seti@home," *Modeling, Analysis & Simulation of Computer and Telecommunication Systems, 2009. MASCOTS '09. IEEE International Symposium on*, pp. 1 – 10, 2009.
[33] D. Nurmi, J. Brevik, and R. Wolski, "Modeling machine availability in enterprise and wide-area distributed computing environments," *Lecture Notes in Computer Science*, vol. 3648, p. 432, 2005.
[34] R. Weicker, "Dhrystone: a synthetic systems programming benchmark," *Communications of the ACM*, vol. 27, no. 10, Oct 1984. [Online]. Available: http://portal.acm.org/citation.cfm?id=358274.358283
[35] H. J. Curnow, B. A. Wichmann, and T. Si, "A synthetic benchmark," *The Computer Journal*, vol. 19, pp. 43–49, 1976.
[36] P. Douglas, "A theory of production," *The American Economic Review*, Jan 1928. [Online]. Available: http://www.jstor.org/stable/1811556