

The Computational and Storage Potential of Volunteer Computing

David P. Anderson¹
Gilles Fedak²

¹ *Space Sciences Laboratory, U.C. Berkeley, davea@ssl.berkeley.edu*
² *INRIA, fedak@lri.fr*

Abstract

Volunteer computing uses Internet-connected computers, volunteered by their owners, as a source of computing power and storage. This paper studies the potential capacity of volunteer computing. We analyzed measurements of about 200,000 hosts participating in a volunteer computing project. These measurements include processing power, memory, disk space, and network throughput, as well as host usage factors, user-specified limits on resource usage, and host churn. We show that volunteer computing can support applications that are data-intensive or have high RAM or storage requirements.

1. Introduction

Volunteer computing (sometimes called “peer-to-peer computing” or “global computing”) uses large numbers of computers, volunteered by members of the general public, for large-scale distributed computing. Volunteer computing is being used in high-energy physics, molecular biology, medicine, astrophysics, climate study, and other areas. These projects have attained unprecedented computing power. For example, SETI@home has sustained a processing rate of about 100 TeraFLOPS for several years.

Most existing volunteer computing projects are throughput-oriented (i.e. they have minimal latency constraints), and have relatively small memory, disk, and network bandwidth requirements. To what extent is volunteer computing useful for more demanding applications? To shed light on this question, we have studied the resources and the factors that limit their use.

Our primary source for this study is a database of host information collected by the SETI@home project [3]. The current version of SETI@home is based on BOINC (Berkeley Open Infrastructure for Network Computing), a middleware system for volunteer

computing [4]. BOINC facilitates the creation of volunteer computing projects; about 20 BOINC-based projects currently exist.

Volunteers participate by running a BOINC client program on their computers. They can “attach” each host to any set of projects, and can control the **resource share** devoted to each project. They can specify **preferences** that limit when and how BOINC can use their computer resources. For example, they can specify that no computation should be done when the computer is in use (as evidenced by keyboard or mouse input).

The BOINC client periodically measures the hardware characteristics of the host. It also measures usage data such as the fraction of time the host is powered on, the fraction of time it has a network connection, and so on.

The BOINC client periodically contacts a scheduling server at each of the projects to which is attached, reporting the host’s hardware and usage characteristics. The scheduling server replies with a set of instructions for doing work: downloading executable files and input files, running the applications against the input files, and uploading the resulting output files. BOINC does not support peer-to-peer communication, and does not support task migration between hosts.

The data in this paper reflects a snapshot of the SETI@home database taken on November 18, 2005. The data reflects the population of hosts that were actively participating at that time - i.e. those that had successfully completed work within the past week.

2. Resource measurements

2.1) CPU performance

The BOINC client periodically executes the Whetstone [7] and Dhrystone [16] benchmarks. The

results are interpreted as floating-point and integer operations per second, respectively.

Of the participating hosts, 25% have 2 or more CPUs, and 2% have 4 or more. A multiprocessor machine with N CPUs typically has lower performance than N times the speed of a single CPU. The difference is especially great for multi-core architectures, such as Intel “hyperthreaded” CPUs, which share a single floating-point unit between cores. To reflect this, BOINC benchmarks all CPUs simultaneously.

The results are shown in Figures 1 and 2.

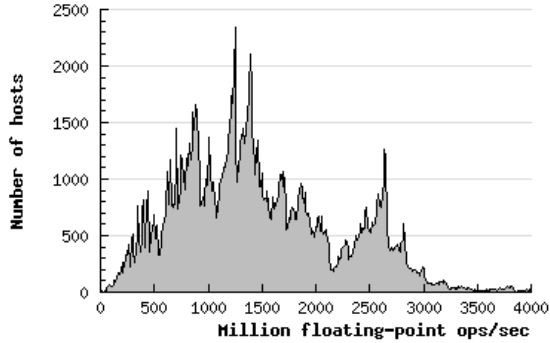


Figure 1: Floating-point computing power

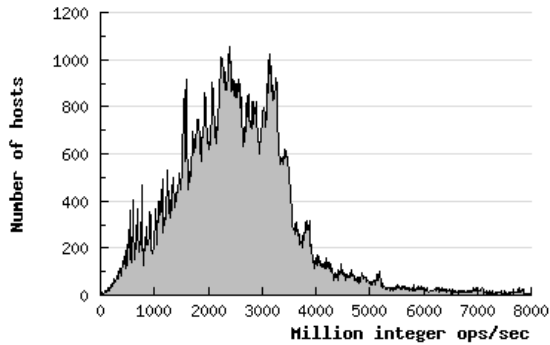


Figure 2: Integer computing power

The participating hosts run a variety of processor types and operating systems, listed in Tables 1 and 2. Microsoft Windows, which accounts for 86.8% of the hosts and 90% of the FLOPS, is subdivided into versions.

CPU type	Number of hosts	GFLOPS per host	GFLOPS total
Intel	141,887	1.543	218,932
AMD	64,921	1.698	110,236
SPARC	590	0.906	535
PowerPC	9,005	1.079	9,716
Others	1,148	1.609	1,847
Total	217,551	1.568	341,120

Table 1: Processor type breakdown

Operating system	Number of hosts	GFLOPS per host	GFLOPS total
Linux	18,227	1.230	22,419
SunOS	715	1.536	1,098
Darwin	9,006	1.079	9,717
Windows total	189,010	1.626	307,330
XP	148,106	1.672	247,633
2000	27,734	1.299	36,026
NT	782	0.782	612
98	3,550	0.697	2,474
2003	7,715	2.526	19,488
95	34	0.402	14
Millennium	980	0.806	790
Longhorn	109	1.940	211
Others	593	1.223	725
Total	217,551	1.568	341,120

Table 2: Operating system breakdown

2.2) Memory

The BOINC client measures and reports the amount of physical memory (RAM) and swap space. Averages are 806 MB RAM and 1.78 GB swap.

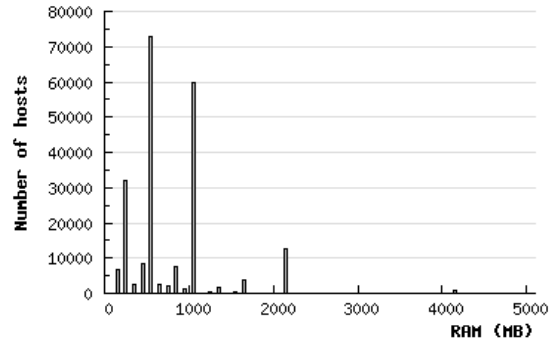


Figure 3: RAM distribution

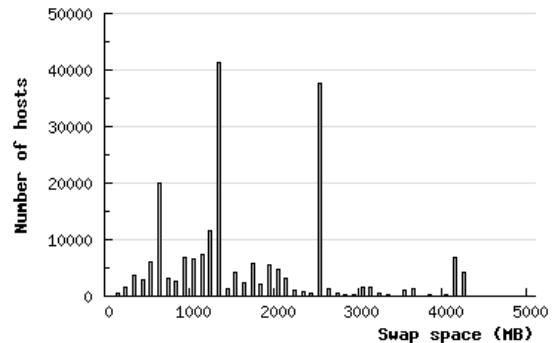


Figure 4: Swap space distribution

BOINC doesn't measure the usage of RAM or swap space by other applications.

2.3) Network throughput

The BOINC client measures throughput during periods when file transfers are in progress (many transfers may be active simultaneously) and it maintains an exponentially weighted average of these values. These measurements reflect several factors: the network bandwidth between the host and the data server, the speed with which the BOINC client transfers data, and the speed of the data server (SETI@home's data server is slow and often overloaded).

Average throughput is 289 Kbps, and the distribution is shown in Figure 5.

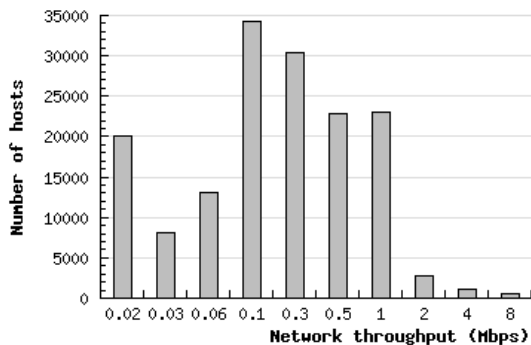


Figure 5: Network throughput distribution

2.4) Disk space

The BOINC client measures the amount of total and free disk space on the volume where it is installed. Averages are 60 GB and 35 GB respectively. The total free space is 7.74 Petabytes. The distributions are shown in Figures 5 and 6.

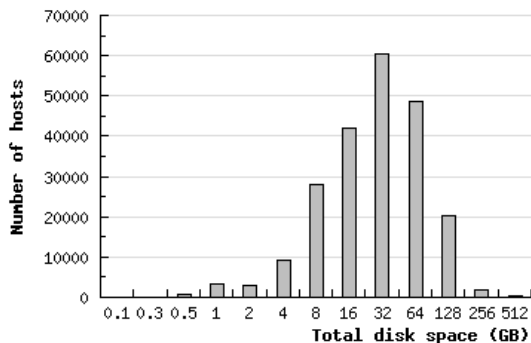


Figure 6: Number of hosts with given amounts of total disk space

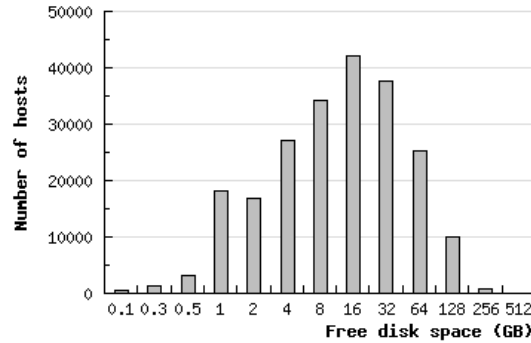


Figure 7: Number of hosts with given amounts of free disk space

BOINC doesn't measure space on volumes other than the one on which it is installed, so it may underestimate available disk space on some machines. It may overestimate disk space in situations where several hosts share a single network-accessible volume, and run BOINC from that volume.

2.5) Combinations of resources

Hardware resources are useless in isolation. Disk space is useful only if there is network bandwidth available to access it, and CPU power is useful only if there is memory in which to execute. Figures 8 through 11 show various combinations of resources. Each graph shows the amount of one resource given that a second resource (shown on the X axis) exceeds a value. Figures 8 and 11 are relevant to applications with large storage and memory requirements. Figure 10 is relevant to applications involving data storage and retrieval.

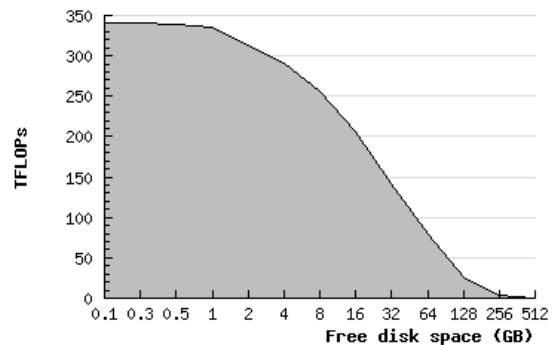


Figure 8: Computing power versus free disk space

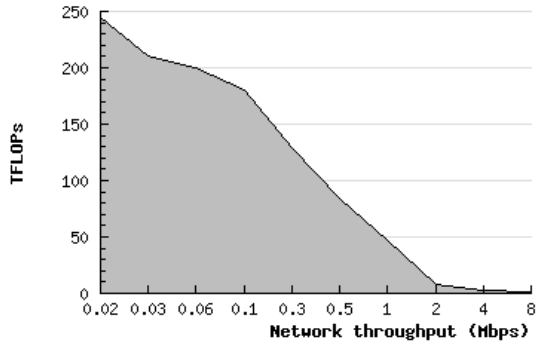


Figure 9: Computing power versus network throughput

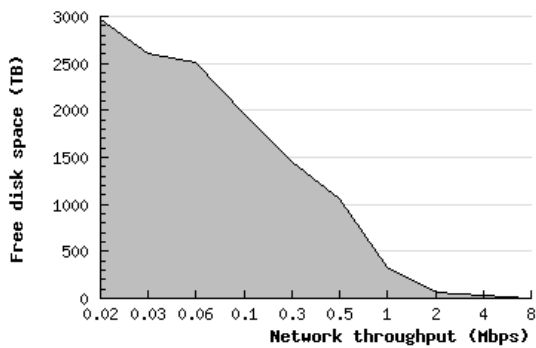


Figure 10: Free disk space versus network throughput

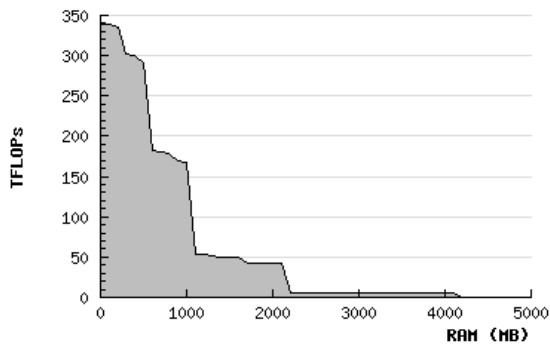


Figure 11: Computing power versus RAM size

2.6) Host location

BOINC participants, during the registration process, can specify their country. In this way hosts are associated with countries. The breakdown is shown in Table 3.

Country	#hosts	GFLOPS per host	Free disk	Thruput
USA	83,225	1.53	41.99	216.28
Germany	24,851	1.58	26.80	128.86
UK	16,998	1.79	41.36	161.54
Canada	9,475	1.54	38.24	301.01
Japan	6,348	1.48	37.49	209.81
France	6,126	1.60	28.25	142.03
Australia	5,524	1.59	34.00	174.15
Netherlands	4,628	1.60	27.24	151.13
Italy	4,586	1.77	31.07	106.54
Spain	4,528	1.59	30.90	88.84

Table 3: Breakdown by country

BOINC doesn't verify that participants are actually from the country they indicate. However, the breakdown roughly agrees with data collected via the BOINC client, which obtains and reports the time zone (offset from Greenwich Mean Time) of the host, and reports it to the scheduling server. The distribution of time zones is shown in Figure 12.

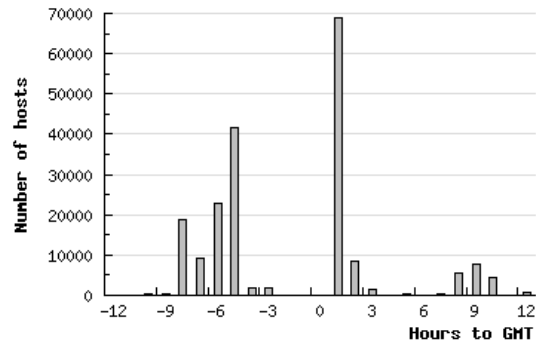


Figure 12: Time zone distribution

Participants can specify whether hosts are at home, school, or work. We call this the **venue**. If they have multiple hosts, they can assign them separate venues, and can define a different set of preferences (see Section 5) to each venue. For example, hosts at work might be configured to run BOINC applications only at night.

The breakdown among venues is shown in Table 4.

Venue	Number of hosts	GFLOPS per host	Disk GB per host	Average thrupt
Home	141,336	1.55	61.78	156.63
Work	39,015	1.68	50.67	219.72
None	28,728	1.53	60.94	212.75
---	8,472	1.45	50.54	204.78

Table 4: Breakdown by venue

3) Participation

3.1) Number of hosts

The dominant factor in a volunteer computing project’s capacity is the number of participating hosts. This depends a variety of factors: the merit and public appeal of the application, ability to get media coverage and other public relations activity, the incentives provided to users, and so on [6].

We expect that the number of hosts participating in volunteer computing will increase significantly, and that there will be many projects with hundreds of thousands of hosts. Currently, on the order of 1 million hosts participate – a few hundred thousand each for BOINC-based projects, GIMPS, distributed.net, Folding@home, Grid.org and World Community Grid. There are, according to current research, about 1 billion PCs in the world [9], so only about 0.1 percent of these participate. As volunteer projects appear in a wider range of areas, and are publicized and marketed more systematically, this could increase by one or two orders of magnitude.

3.2) Host churn

A volunteer computing project’s pool of hosts is dynamic: hosts are continually arriving and leaving. In addition, users occasionally reset the BOINC client on a given host, which has the effect of creating a new host.

We measured host “lifetime”: the interval from creation to last communication for hosts that had not communicated in at least one month (this underestimates lifetime because it omits active hosts). The average host lifetime is 89.5 days, and the distribution is shown in Figure 13.

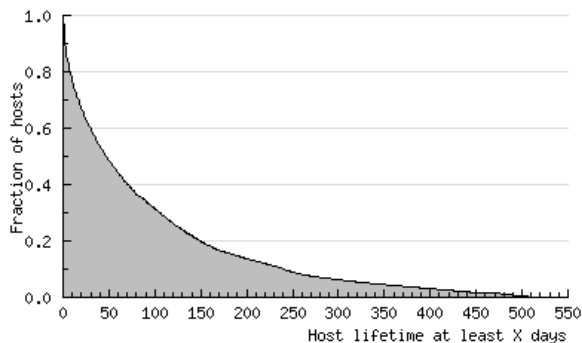


Figure 13: Host lifetime distribution

Host churn is of critical importance to applications that rely on their persistence on hosts. Examples

include long-term storage applications like Oceanstore [REF] and applications that do extremely long computations (such as Climateprediction.net, whose tasks take several months of CPU time on a typical host [6]).

The average number of active hosts is the average arrival rate times the average lifetime. The arrival rate can change over time. The arrival history for SETI@home is shown in Figure 14. Spikes in the graph correspond to public-relations events; gaps correspond to server outages.

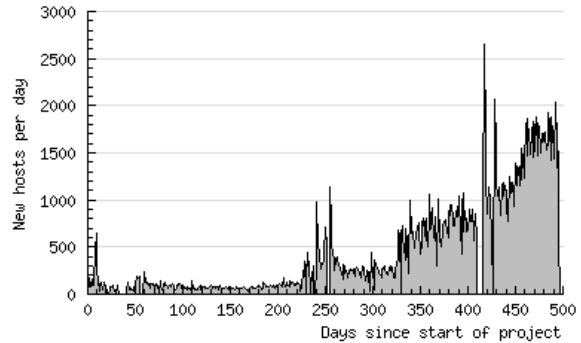


Table 14: Host arrival history

3.3) Number of hosts per user

We analyzed the number of hosts per user (see Table 5 and Figure 15). The top three users had 2179, 1738 and 1491 hosts. Most users have a single host, but most hosts belong to a user with multiple hosts.

hosts/user	#users	#hosts	percentage total hosts
1	83,870	83,870	38.6%
2-10	32,257	97,770	44.9%
11-100	1,217	25,620	11.7%
101-1000	27	4,884	2.2%
1000+	3	5,408	2.4%

Table 5: Number of hosts per user

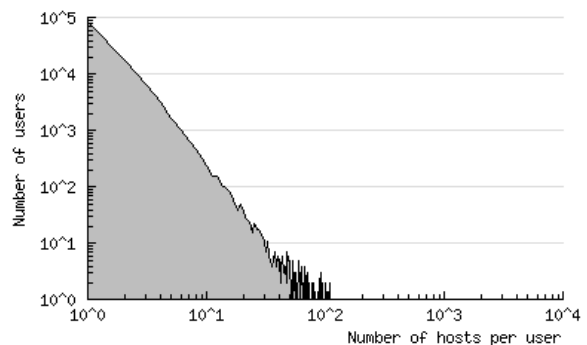


Figure 15: Number of hosts per user

4) Usage statistics

The BOINC client measures several aspects of host usage. The fraction of real time during which the BOINC client is running on the host is called the **on-fraction**. On most hosts, this is about the same as the fraction of time the host is powered on, since BOINC starts automatically at boot-up and runs in the background all the time. The mean on-fraction is 0.807.

The fraction (of the time that BOINC is running) that a physical network connection exists is called the **connected-fraction**. For hosts with LAN and DSL connections, this is about 1. For hosts with telephone-based (ISDN or modem) or wireless connections, it may be lower. The mean connected-fraction is 0.831.

There may be periods when, although BOINC is running, it is not allowed to execute applications or transfer files. This occurs when the host is in use (mouse/keyboard input) and user preferences are to not run when in use, the time of day is outside a user-specified range, or the user has explicitly suspended BOINC activity (via a command in the BOINC graphical interface). The fraction (of the time that BOINC is running) when it is allowed to compute and communicate is called the **active-fraction**. The average active-fraction is 0.842.

Not all a host's CPU time is available to BOINC: other CPU-intensive programs may run on some hosts. BOINC does not directly measure CPU load. However, it does maintain, for each project, the **CPU efficiency**, defined as average number of CPU seconds accumulated by that project's applications per second of wall time during which they are runnable. This reflects CPU usage by non-BOINC applications, and I/O activity by the BOINC application.

In the case of SETI@home, which does very little I/O, CPU efficiency reflects primarily non-BOINC CPU load. The average CPU efficiency is 0.899.

5) User preferences

BOINC allows participants to specify (through a web-based interface) various **preferences** that limit how and when BOINC uses their hardware resources. These preferences include:

Run if user active: whether BOINC should be active even if there has been user input (mouse or keyboard) in the last three minutes. The default is No. 71.9% selected Yes.

Limited computing hours: a range of hours during which BOINC may compute or communicate. 3.3% of

users specified such a range, and the average duration was 12.41 hours.

Limited communication hours: a range of hours during which BOINC may do network communication. 0.8% of users specified such a range, and the average duration was 12.18 hours.

Confirm before connecting: whether BOINC should get user permission (via a dialog) before doing network communication. This is primarily relevant to modem users. The default is No. 8.4% selected Yes.

Hangup if dialed: whether BOINC should hang up a modem connection when it no longer needs it. The default is No. 60% selected Yes.

Minimum connection interval: a target minimum time between network communication. This has two purposes: 1) it lets modem users (who often pay a fee per connection) concentrate communication into infrequent bursts; 2) if a host is sporadically connected (e.g. a laptop) it lets the user make sure that enough work is fetched to keep the host busy. The default is 0.1 days. The average setting is 0.69 days.

Disk access interval: a minimum time between disk accesses. This is relevant primarily to laptops with a low-power mode in which the disk doesn't rotate. The default is 60 seconds; the average setting is 78.9 seconds.

Disk maximum used: the maximum amount of disk space used by BOINC. The default is 100 GB. The average setting is 63.6 GB.

Disk maximum percent used: the maximum percentage of total disk space used by BOINC. The default is 50%. The average setting is 42.6%.

Disk minimum free: the minimum amount of free disk space. (When free space falls below this level, BOINC deletes files). The default is 0.1 GB. The average setting is 0.97 GB.

In addition to these preferences, which apply to all projects to which a host is attached, users can specify a per-project **resource share** that determines how bottleneck resources are allocated (a project's share of a bottleneck resources is its resource share relative to other projects contending for that resource). 16.8% of SETI@home users participate in other BOINC projects, and the average resource share of SETI@home (over all users) is 0.917.

6) Analysis

6.1) Total processing capacity

Because anonymously volunteered computers can't be trusted, many volunteer computing projects use redundant computing to minimize the effect of

malicious or malfunctioning hosts. In this technique, each job is executed twice on hosts belonging to different volunteers. If the results agree within application-defined thresholds, they are accepted; otherwise a third instance is executed, and so on. Redundant computing decreases effective computing power by a factor of slightly more than two.

Combining the various factors we have presented, and assuming that the factors are statistically independent, we have the following expression for the total computing power X available to a project:

$$X = X_{\text{arrival}} * X_{\text{life}} * X_{\text{ncpus}} * X_{\text{flops}} * X_{\text{eff}} * X_{\text{onfrac}} * X_{\text{active}} * X_{\text{redundancy}} * X_{\text{share}}$$

Where X_{arrival} is the average arrival rate of hosts, X_{life} is the average lifetime of hosts, X_{ncpus} is the average number of CPUs per host, X_{flops} is the average FLOPS per CPU, X_{eff} is the average CPU efficiency, X_{onfrac} is the average on-fraction, X_{active} is the average active-fraction, $X_{\text{redundancy}}$ is the reciprocal of the average redundancy, and X_{share} is the average resource share (relative to other CPU-intensive projects).

For applications that use large amounts of RAM or disk, this estimate must be scaled by the factors described in sections X and Y. Analogous expressions estimate the limits of storage capacity and network transfer.

In the case of SETI@home, the product of the first 4 factors (i.e. the hardware resource) is about 341 TeraFLOPS. The product of the remaining 5 factors is 0.28. Thus SETI@home, during the study period, had a processing rate of 95.5 TeraFLOPS.

6.2) Data-intensive applications

To what extent can volunteer computing handle data-intensive tasks? Foster and Iamnitchi discuss this question [10], and point out that while SETI@home processes about 25 KB of data per CPU hour, some physics applications have a much higher ratio. They cite applications that process 60 MB and 660 MB per CPU hour.

To study this question, we define the **data processing rate** R of an application to be the average number of bytes of data it processes using 3.6e11 floating-point operations (i.e. one hour of CPU time on a 1 GFLOPS computer). We assume that a client is able to do both computation and communication nearly all the time (the BOINC client overlaps these activities, and network communication takes little CPU time).

Suppose a 1 GFLOPS computer has a 1 Mbps network connection. Then it can download 450 MB per

hour. If it runs an application for which $R=450\text{MB}$, then both network and CPU are saturated (i.e. busy all the time). If $R < 450\text{MB}$, the network is not saturated; if $R > 450\text{MB}$, the CPU is not saturated (of course, the excess CPU time could be used by a less data-intensive project).

This critical value of R varies with the host; it will be smaller if the host has a faster CPU or a slower network connection. For a given value of R , some fraction of hosts will be network-saturated and won't be able to devote all their CPU time to the application. Figure 15 illustrates this effect, showing the computing power available to applications with various values of R . The shaded line shows the fraction of hosts whose CPUs are not saturated at the given data rate.

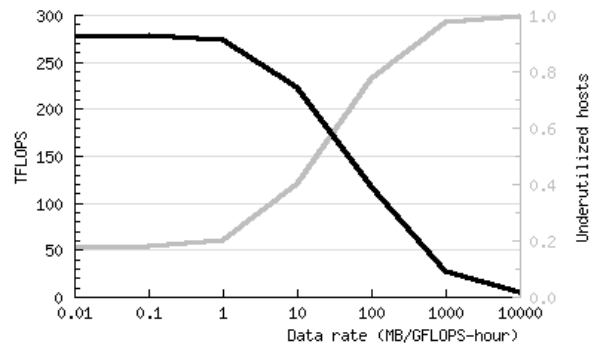


Figure 15: Computing power versus data rate

It can be seen that considerable processing power is available even to applications with $R = 100\text{MB}$ or 1GB . In other words, volunteer computing can potentially handle data-intensive applications.

This analysis omits some important factors: saturating lots of client network connections could swamp the outgoing server links, ISP backbone networks, and shared incoming links. Solving these problems is an entire research agenda; we believe that an approach based on gleaning unused network bandwidth could yield progress.

7) Related work

Sarmenta [15] originally articulated the idea of volunteer computing, and he also explored many of its technical aspects. The Entropia [5] and XtremWeb [11] projects have studied the speedup of specific applications in the context of volunteer computing.

The Condor project [14] and others [2] have explored using the idle time of organizational workstations to do parallel computing. Acharya and Setia [1] studied the availability of idle RAM on

workstation pools. Eggert and Touch [8] studied operating mechanisms for idle resource use.

Workstation cycle-stealing (and Grid computing in general) differs fundamentally from volunteer computing. It generally requires that parallel tasks run simultaneously, so that they may communicate; this in turn requires the ability to migrate running tasks. Resources are trusted, so that validation techniques like redundant computing are not needed. Workstations can be contacted dynamically (in BOINC, all communication is client-initiated, so that firewalls and NATs can be traversed).

8) Conclusion

We have presented an analysis of the hardware resources present in a typical volunteer computing project (SETI@home), and have described the other factors that affect the computing power and storage capacity available to the project. On the basis of this data, we conclude that this host pool – and hence volunteer computing in general – can support applications that are much more data-intensive than SETI@home, or that have much higher RAM and disk requirements.

A snapshot of the SETI@home host database is made every day, and we have an archive of these over a period of about 10 months. This could provide interesting data about the change in resources over time. We made some initial studies, but the results were inconclusive; early participants had mostly fast machines on fast networks. Many later participants had lower-end machines. Thus, some performance measures actually declined over the interval.

We thank Rom Walton, Matt Lebofsky, and many volunteer programmers for their help in collecting performance data. This work was supported by the National Science Foundation under grants SCI-0221529 and SCI-0438443.

References

- [1] A. Acharya and S. Setia. “Availability and Utility of Idle Memory in Workstation Clusters”. SIGMETRICS 99. Atlanta, May 1-4 1999.
- [2] A. Acharya, G. Edjlali, and J. Saltz. “The Utility of Exploiting Idle Workstations for Parallel Computation”. SIGMETRICS 97. Seattle, June 15-18 1997.
- [3] D.P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, D. Werthimer. “SETI@home: An Experiment in Public-Resource Computing”. *Communications of the ACM*, 45(11), November 2002, 56-61.
- [4] D.P. Anderson. “BOINC: A System for Public-Resource Computing and Storage”. 5th IEEE/ACM International Workshop on Grid Computing, pp. 365-372, Nov. 8 2004, Pittsburgh, PA.
- [5] A. Chien, B. Calder, S. Elbert, and K. Bhatia. “Entropy: architecture and performance of an enterprise desktop grid system”. *J. Parallel Distrib. Comput.* 63(2003) 597-610.
- [6] C. Christensen, T. Aina and D. Stainforth. “The Challenge of Volunteer Computing With Lengthy Climate Model Simulation”. To appear in 1st IEEE International Conference on e-Science and Grid Computing, Melbourne, Dec 5-8 2005.
- [7] H.J. Curnow and B.A. Wichmann. “A synthetic benchmark”. *The Computer Journal* 19(1), pp. 43-49. 1976.
- [8] L. Eggert and J. Touch. “Idle-time Scheduling with Preemption Intervals”. 20th ACM Symposium on Operating Systems Principles, Oct. 23-26 2005, Brighton, UK.
- [9] eTForecasts. <http://www.etforecasts.com>
- [10] I. Foster and A. Iamnitchi. “On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing”. 2nd International Workshop on Peer-to-Peer Systems. Berkeley, February 20-21 2003.
- [11] C. Germain, V. Neri, G. Fedak and F. Cappello. “XtremWeb: Building an Experimental Platform for Global Computing”. First IEEE/ACM International Workshop on Grid Computing. December 17-20, 2000, Bangalore, India
- [12] Internet Systems Consortium Internet Domain Survey, July 2005. <http://www.isc.org/>
- [13] J. Kubiawicz, et al. OceanStore: An Architecture for Global-Scale Persistent Storage. ASPLOS, December 2000.
- [14] M.J. Litzkow, M. Livny, M.W. Mutka. “Condor - A Hunter of Idle Workstations”. *Proceedings of the 8th International Conference of Distributed Computing Systems*, pages 104-111, June, 1988.
- [15] L.F.G. Sarmata, “Bayanihan: Web-Based Volunteer Computing Using Java”. *Lecture Notes in Computer Science* 1368, Springer-Verlag, 1998. pp. 444-461.
- [16] R. P. Weicker. “Dhrystone: A Synthetic Systems Programming Benchmark”. *Communications of the ACM* 27 (10), October 1984, pp. 1013-1030.