Volunteer computing: the ultimate cloud

David P. Anderson
Space Sciences Laboratory
University of California, Berkeley

High-performance computing

Computers continue to get faster exponentially, but the computational demands of science are growing even faster. Extreme requirements arise in several areas:

- Physical simulation. Scientists use computers to simulate physical reality at many levels of scale: molecule, organism, ecosystem, planet, galaxy, universe. The models are typically chaotic, and studying the distribution of outcomes requires lots of simulation runs with perturbed initial conditions.
- Compute-intensive analysis of large data. Modern instruments (optical and radio telescopes, gene sequencers, gravitational wave detectors, particle colliders) produce huge amounts of data, which in many cases requires compute-intensive analysis.
- Biology-inspired algorithms such as genetic and flocking algorithms for function optimization.

These areas engender computational tasks that would take hundreds or thousands of years to complete on a single PC. Reducing this to a feasible interval – days or weeks – requires **high-performance computing (HPC)**. One approach is to build an extremely fast computer – a **supercomputer**. However, in the areas listed above, the rate of job completion, rather than the turnaround time of individual jobs, is the important performance metric. This subset of HPC is called **high-throughput computing**.

To achieve high throughput, the use of **distributed computing**, in which jobs are run on networked computers, is often more cost-effective than supercomputing. There are many approaches to distributed computing:

- **Cluster computing**, which uses dedicated computers in a single location.
- **Desktop grid computing**, in which desktop PCs within an organization (such as a department or university) are used as a computing resource. Jobs are run at low priority, or while the PCs are not being otherwise used.
- **Grid computing**, in which separate organizations agree to share their computing resources (supercomputers, clusters, and/or desktop grids).
- **Cloud computing**, in which a company sells access to computers on a pay-as-you-go basis.
- **Volunteer computing**, which is like desktop grid computing except that the computing resources are volunteered by the public.

Each of these paradigms has an associated resource pool: the computers in a machine room, the computers owned by a university, the computers owned by a cloud provider. In the case of volunteer computing, the resource pool is the set of all privately-owned PCs in the world. This pool is interesting for several reasons:

- It dwarfs the other pools: the number of privately-owned PCs is currently 1 billion and is

projected to grow to 2 billion by 2015.

- The pool is self-financing, self-updating and self-maintaining: people buy new PCs, upgrade system software, maintain their computers, and pay their electric bills.
- Consumer PCs, not special-purpose computers, are state of the art. Consumer markets drive research and development. For example, the fastest processors today are GPUs developed for computer games. Traditional HPC is scrambling to use GPUs, but there are already 100 million GPUs in the public pool, and tens of thousands are already being used for volunteer computing.

The history of volunteer computing

In the mid-1990s, as consumer PCs became powerful and millions of them were connected to the Internet, the idea of using them for distributed computing arose. The first two projects were launched in 1996 and 1997: **GIMPS** finds prime numbers of a particular type, and **distributed.net** breaks cryptosystems via brute-force search of the key space. They attracted tens of thousands of volunteers and demonstrated the feasibility of volunteer computing.

In 1999 two new projects were launched: **SETI@home**, from U.C. Berkeley, analyzes data from the Arecibo radio telescope, looking for synthetic signals from space. **Folding@home**, from Stanford, studies how proteins are formed from gene sequences. These projects received significant media coverage and moved volunteer computing into the awareness of the global public.

These projects all developed their own **middleware**: the application-independent machinery for distributing jobs to volunteer computers and for running jobs unobtrusively on these computers, as well as web interfaces by which volunteers could register, communicate with other volunteers, and track their progress. Few scientists had the resources or skills to develop such software, and so for several years there were no new projects.

In 2002, with funding from the National Science Foundation, the BOINC project was established to develop general-purpose middleware for volunteer computing, making it easier and cheaper for scientists to use volunteer computing. The first BOINC-based projects launched in 2004, and today there about 60 such projects, in a wide range of scientific areas. Some of the larger projects include Milkyway@home (from Rensellaer Polytechnic Institute; studies galactic structure), Einstein@home (from Univ. of Wisconsin and Max Planck Institute; searches for gravitational waves), Rosetta@home (from Univ. of Washington; studies proteins of biomedical importance), ClimatePrediction.net (from Oxford University; studies long-term climate change), and IBM World Community Grid (operated by IBM; hosts 5-10 humanitarian applications from various academic institutions).

Evaluating volunteer computing

Volunteer computing can be compared with other high-performance computing paradigms in several dimensions:

Performance: about 900,000 computers are actively participating in volunteer computing. Together they supply about 10 PetaFLOPS (trillion floating-point operations per second) of computing power; the fraction supplied by GPUs is about 70% and growing. For comparison, the fastest supercomputer supplies about 1.4 PetaFLOPS, and the largest grids number in the tens of thousands of hosts. So in

terms of throughput, volunteer computing is competitive with other paradigms, and it has the near-term potential to greatly surpass them: if participation increases to 4 million computers, each with a 1 TeraFLOPS GPU (the speed of current high-end models) and computing 25% of the time, the result will be 1 ExaFLOPS of computing power; other paradigms are projected to reach this level only in a decade or more. Actually, since 4 million PCs is only 0.4% of the resource pool, the near-term potential of volunteer computing goes well beyond Exa-scale.

Cost effectiveness: for scientists, volunteer computing is cheaper than other paradigms – often dramatically so. A medium-scale project (10,000 computers, 100 TeraFLOPS) can be run using a single server computer and one or two staff – roughly $200,000 per year. An equivalent CPU cluster costs at least an order of magnitude more. Cloud computing is even more expensive. For example, Amazon Elastic Computing Cloud instances provide 2 GigaFLOPS and cost $2.40 per day. To attain 100 TeraFLOPS, 50,000 instances would be needed, costing $43.8 million per year. (However, studies suggest that cloud computing is cost-effective for hosting volunteer computing project servers.)

Resource allocation policy and public outreach: In traditional HPC paradigms, resources are allocated by bureaucracies: funding agencies, institutions, and committees. The public, although it pays for the resources, has no direct voice in their allocation, and doesn't know how they're being used. In volunteer computing, the public has direct control over how resources are allocated, and knows what they're being used for. As a result, public awareness of science is increased, and research projects that are outside of the current academic mainstream can potentially get significant computing resources.

Scientific adoption: volunteer computing has not yet been widely adopted: 60 research groups are currently using volunteer computing, while perhaps a hundred times that many could benefit from it. Cluster and grid computing are much more widely used by scientists. The HPC community, on whom scientists rely for guidance, has ignored volunteer computing, perhaps because it offers neither control nor funding. In addition, although BOINC has reduced the barrier to entry, few research groups have the resources and skills needed to operate a project. The most promising solution to this is to create **umbrella projects** serving multiple scientists and operated at a higher organizational level (for example, at the level of a university).

Energy efficiency: the FLOP/Watt ratio of a PC is lower than that of a supercomputer, and it is tempting to conclude that volunteer computing is less energy-efficient than supercomputing. However, this is not necessarily the case. In cold climates, for example, energy used by a PC may replace energy used by a space heater, to which the PC is thermodynamically equivalent. No study has been done taking such factors into account.

The BOINC project/volunteer model

The BOINC software consists of two parts:

- Server software that is used by to create **projects**. Anyone – academic researchers, hobbyists, malicious hackers – can create a project. Projects are independent; each one operates its own server and provides its own web site. BOINC has no centralized component other than a web site from which its software can be downloaded.
- Client software that volunteers install and run on their computers. The client software is available for all major platforms, including Windows, Linux, and Mac OS X.

Having installed the client program, volunteers can then **attach** it to any set of projects, and for each project can assign a **resource share** that determines how the computer's resources are divided among the projects.

The choice of projects is up to the volunteer. Attaching to a project allows it to run arbitrary executables on one's computer, and BOINC provides only limited (account-based) sandboxing. So the volunteer must assess the project's authenticity, its technical competence, and its scientific merit. The ownership of intellectual property resulting from the project may also be a factor.

BOINC encourages volunteers to participate in multiple projects simultaneously. By doing so, they avoid having their computer go idle if one project is down. Multiple attachment also helps projects whose supply of work is sporadic.

More generally, by making it easy to join and leave projects, BOINC encourages volunteers to occasionally evaluate the set of available projects, and to devote their computing resources to that projects that, in their view, are doing the most important and best research.

BOINC does accounting if **credit** – a numerical measure of a volunteer's contribution to a project. The accumulation of a large amount of credit in a particular project can be a disincentive to try other projects. To combat this, BOINC provides a cross-project notion of identity (based on the volunteer's email address). Each project exports its credit statistics as XML files, and various third-party credit statistics sites import these files and display **cross-project credit**: that is, the volunteer's total credit across all projects.

Even with the modest number (60) of current projects, the process of locating them, reading their web sites, and attaching to a chosen set is a tedious process, and will become infeasible if the number of projects grows to hundreds or thousands.

BOINC provides a framework for dealing with this problem. A level of indirection can be placed between client and projects. Instead of being attached directly to projects, the client can be attached to a web service called an **account manager**. The client periodically communicates with the account manager, passing it account credentials and receiving a list of projects to attach to.

This framework has been used by third-party developers to create "one-stop shopping" web sites, where volunteers can read summaries of all existing BOINC projects and can attach to a set of them by checking boxes. The framework could also be used for delegation of project selection, analogous to mutual funds. For example, volunteers wanting to support cancer research could attach to an American Cancer Society account manager. ACS experts would select a dynamic weighted "portfolio" of meritorious cancer-related volunteer projects.

Human factors in volunteer computing

All HPC paradigms involve human factors, but in volunteer computing these factors are particularly crucial and complex. To begin with, why do people volunteer? This question is currently being studied rigorously. Evidence suggests that there are several motivational factors:

- **Support for scientific goals**: some volunteers want to further a particular research goal (such as

curing diseases, finding extraterrestrial life, or predicting climate change).
- **Community**: some volunteers enjoy participating in the on-line communities and social networks that form, through message boards and other web features, around volunteer computing projects.
- **Credit**: some volunteers are interested in the performance of computer systems, and use volunteer computing to quantify and publicize the performance of their computers.

There have been attempts to commercialize volunteer computing by paying participants, directly or via a lottery, and reselling the computing power. These efforts have failed because the potential buyers, such as pharmaceutical companies, are unwilling to have their data on computers outside of their control.

To attract and retain volunteers, a project must perform a variety of human functions. It must develop web content describing its research goals, methods, and credentials; it must provide volunteers with periodic updates (via web or email) on its scientific progress; it must manage the moderation of its web site's message boards to ensure that they remain positive and useful; it must publicize itself by whatever media are available (mass media, alumni magazines, blogs, social networking sites).

Volunteers must trust projects, but projects cannot trust volunteers. From a project's perspective, volunteers are effectively anonymous. If a volunteer behaves maliciously, for example by intentionally falisfying computational results, the project has no way to identify and punish the offender. In other HPC paradigms, such offenders can be identified and disciplined or fired.

Technical factors in volunteer computing

Volunteer computing poses a number of technical problems. For the most part, these problems are addressed by BOINC, and scientists need not be concerned with them. The problems include:

Heterogeneity: the volunteer computer population is extremely diverse in terms of hardware (processor type and speed, RAM, disk space), software (operating system and version) and networking (bandwidth, proxies, firewalls). BOINC provides scheduling mechanisms that assign jobs to the hosts that can best handle them. However, projects still generally need to compile applications for several platforms (Windows 32 and 64 bit, Mac OS X, Linux 32 and 64 bit, various GPU platforms). This difficulty may soon be reduced by running applications in virtual machines.

Sporadic availability and churn: volunteer computers are not dedicated. The time intervals when a computer is on, and when BOINC is allowed to compute, are sporadic and generally unpredictable. BOINC tracks these factors and uses them in estimating job completion times. In addition, computers are constantly joining and leaving the pool of a given project. BOINC must address the fact that computers with many jobs in progress may disappear forever.

Result validation: because volunteer computers are anonymous and untrusted, BOINC cannot assume that job results are correct, or that the claimed credit is accurate. One general way of dealing with is replication: that is: send a copy of each job to multiple computers; compare the results; accept the result if the replicas agree; otherwise issue additional replicas. This is complicated by the fact that different computers often do floating-point calculations differently, so that there is no unique correct result. BOINC addresses this with a mechanism called **homogeneous redundancy** that sends instances of a

given job to numerically identical computers. In addition, redundancy has the drawback that it reduces throughput by at least 50%. To address this, BOINC has a mechanism called **adaptive replication** that identifies trustworthy hosts and replicates their jobs only occasionally.

Scalability: large volunteer projects can involve a million hosts and millions of jobs processed per day. This is beyond the capabilities of grid and cluster systems. BOINC addresses this using an efficient server architecture that can be distributed across multiple machines. The server is based on a relational database, so BOINC leverages advances in scalability and availability of database systems. The communication architecture uses exponential backoff after failures, so that the rate of client requests remains bounded even when a server comes up after a long outage.

Security: volunteer computing poses a variety of security challenges. What if hackers break into a project server and use it to distribute malware to the attached computers? BOINC prevents this by requiring that executables be digitally signed using a secure, offline signing computer. What if hackers create a fraudulent project that poses as academic research while in fact stealing volunteers' private data? This is partly addressed by account-based sandboxing: applications are run under an unprivileged user account and typically have no access to files other than their own input and outputs. In the future, stronger sandboxing may be possible using virtual machine technology.

The future of volunteer computing

Volunteer computing has demonstrated its potential for high-throughput scientific computing. However, only a small fraction of this potential has been realized. Moving forward will require progress in three areas:

Increased participation: the volunteer population has remained around 500,000 for several years. Can it be grown by an order magnitude or two? Several factors could help: a) a dramatic scientific breakthrough such as the discovery of a cancer treatment or a new astronomical phenomenon; b) the effective use of social networks like Facebook; c) bundling of BOINC by computer manufacturers or software vendors (currently, Folding@Home is bundled with the Sony Playstation 3 and with ATI GPU drivers).

Increased scientific adoption: the set of volunteer projects is small and fairly stagnant. Several factors might change this: a) the creation of "umbrella projects" by universities and other institutions; b support for higher-level computing models such as workflow management systems and MapReduce; c) promotion of volunteer computing by scientific funding agencies; d) increased acceptance of volunteer computing by the HPC and computer science communities.

Tracking technology: today, the bulk of the world's computing power is in desktop and laptop PCs, but in a decade or two it may shift to energy-efficient mobile devices. Such devices, while docked, could be used for volunteer computing.

If these challenges are addressed, and volunteer computing experiences explosive growth, there will be thousands of projects. At this point volunteers can no longer be expected to evaluate all projects, and new allocation mechanisms will be needed: for example, the "mutual fund" idea mentioned above, or something analogous to decision markets, in which individuals are rewarded for participating in new projects that later produce significant results; such "expert investors" would steer the market as a whole.