How to Get Volunteers for Web-based Metacomputing

Francisco Soares, Luís M. Silva, João Gabriel Silva

Departamento de Engenharia Informática
Universidade de Coimbra - POLO II
Vila Franca - 3030 Coimbra
PORTUGAL
Email: luis@dei.uc.pt
Tel: +351-39-790090
Fax: +351-39-701266

Abstract
Web-based parallel computing is a promising idea. However, its success depends on the good willing of Internet users. We have developed a project for Internet-based parallel computing, called JET, and we found this is an important problem that needs to be solved. Some features should be provided to achieve the enthusiasm of Internet users that contribute the CPU idle cycles of their machines. A possible solution is to provide a Web-page with statistics about the computational power offered by each individual user or institution.

In this paper, we present the design of JET-STATS, a server of statistics that was developed in our project. The server was implemented inside a HTTP demon and several optimization techniques have been used to obtain efficiency. It provides the statistical information in a Web page with a fancy interface. This server was implemented in a modular way and it can be easily adapted to any other system that uses the idea of Web-based parallel computing.

Keywords: Metacomputing, Web-based Computing, Java, World-Wide-Web.

1. Introduction

The popularity of the World Wide Web has lead to an impressive increase in the number of Internet hosts. According to the statistics presented in [1] in January of 1998 there were about 29,6 millions of computers connected to the Internet. This mass of processors connected together represent a very powerful parallel supercomputer with an incredible computational power. Most of these machines are only used for small interactive tasks, like the reading of electronic mail, the editing of files or just the browsing of Web pages and most of them remain idle in a significant part of the time [2]. Thus, it seems insightful to apply this vast computing resource for solving some problems of cryptography, mathematics and computational science.

In fact, there are some quite interesting projects that use the computational power of machines that are connected to the Internet. Examples include the Bovine project for cracking RC5 and DES encryption keys [3], the GIMPS project to find Mersenne Primes [4], the RSA Factoring-By-Web effort that tries to crack RSA codes [5], the experimental study presented in [6], the example described in [7] and the announced project -SETI@home- that will try to use the spare power of Internet-connected computers in the Search for Extra-Terrestrial Intelligence (SETI) [8].
These examples demonstrate that thousands of computers can be harnessed as an Internet-based supercomputer to solve long-running problems using nothing but spare CPU time. None of these projects use Java; client programs are written in traditional languages like C and assembly. The download of code is done manually by the user that is also responsible by its installation and configuration. In most of the cases the communication over the Internet is simply based on electronic mail.

Although Java is still a very young language it is claimed to be the language for the Internet [Javasoft]. In our opinion, Java can bring some important advantages when used in this kind of Internet-based number-crunching computing: it solves the architectural heterogeneity of Internet hosts ensuring the portability of code, it is easy to install in Internet nodes by using downloaded Java applets and it has built-in security mechanisms. Besides, Java applications are easier to develop and maintain than other languages.

The goal of our project, called JET, is to provide an infrastructure to support the execution of parallel applications over the Internet. There are other on-going projects that pursue the same goal, so most of the ideas presented in this paper can also be implemented in other systems.

In this paper we describe the module of statistics of the JET system. It provides up-to-date information about the CPU time volunteered by individual users and/or institutions that participate in a global JET computation. All the statistics are organized in rankings and can be seen in a Web page by any Internet user. This module of statistics is an important motivation for the users that participate in a Web-based computation and it is an effective way to seize new volunteers that offer their CPU idle cycles.

The rest of the paper is organized as follows: the next section presents a brief overview of JET. Section 3 presents an overview of the module of statistics of JET. Section 4 describes the related work while section 5 concludes the paper.

2. A General Overview of JET

This section presents a brief overview of the JET system. The interested reader is referred to [10] for more details about the system. JET is a software infrastructure that supports parallel processing of CPU-intensive problems that can programmed in the Master/Worker paradigm. There is a Master process that is responsible for the decomposition of the problem into small and independent tasks. The tasks are distributed among the Worker processes that execute a quite simple cycle: receive a task, compute it and send the result back to the master. The Master is responsible for gathering the partial results and to merge them into the problem solution. Since every task is independent from each other, there is no need for communication between worker processes.

The Worker processes execute as Java applets inside a Web browser. The user that wants to volunteer his spare CPU cycles to a JET computation just need to access a Web page by using a Java-enabled browser. Then, she just have to click somewhere inside the page and one Worker Applet is downloaded to the client machine. This Applet will communicate with a JET Master that executes on the same remote machine where the Web page came from.
The communication between the worker applets and the JET Master is done through UDP sockets. This class of sockets provides higher scalability and consume less resources than TCP sockets. The UDP protocol does not guarantee the delivery of messages but the communication layer of JET implements a reliable service that insures sequenced and error-free message delivery. The library keeps a time-out mechanism for every socket connection in order to detect the failure or an withdraw of some worker applet.

The volunteer machines may join and leave the computation at any instant of time. Thereby, the execution environment is completely dynamic. The JET system provides some mechanisms to tolerate the frequent changes on the parallel virtual machine and include support for dynamic task distribution.

The JET system provides some internal mechanisms to tolerate the high latency of the communication over the Internet. Those techniques are based on the prefetching of tasks by the remote machines and the asynchronous flush of output results back to the JET Master. There are some internal threads that perform the communication in a concurrent way with the normal execution of the application processes.

The number of machines that can join a JET computation is surely unpredictable but the system should be able to manage hundreds or thousands of clients. To assure a scalable execution we depart from the single-server approach and the forthcoming version of JET relies in a hierarchical structure of servers, as represented in Figure 1.

![Diagram](image.png)

Figure 1: The Structure of the JET virtual machine.
The JET system includes some fault-tolerance mechanisms. Task reconfiguration is used to tolerate the loss of worker applet. The resiliency of the Master processes is achieved through the use of checkpointing and logging techniques. The checkpointing mechanism is necessary to assure the continuity of the application when there is a failure or a preventive shutdown of a JET Master or the main Server. The critical state of the application is saved periodically in stable storage in some portable format that allows its resumption later on in the same or in a different machine.

The JET machine must keep the volunteers interested to participate in a world-wide computation. The best way to motivate people and increase their enthusiasm is to maintain statistics about their participation. The current status of the JET parallel virtual machine may be accessed through a Web page that contains the module of statistics, where any user can see its contribution in terms of a world-wide ranking. Information like the list of volunteer machines, their contribution time and performance will be automatically published in a Web page. This would motivate people to be organized in teams and compete with other institutions and Universities. More details about this module of statistics will be presented in section 3.

In order to evaluate the feasibility of JET we ported a set of complex applications to the first prototype of JET: a program to find Mersenne primes [11], another one to crack the Secure Sockets Layer [12], a third application from a group of Physics of our University that calculates some Feynman path integrals using Monte Carlo techniques, and finally we have ported a RCS (64-bit key) [13] client to JET.

3. The JET Statistical Module (JET-STATS)

There are some quite good examples of number-crunching applications that have been executing over the Internet. These examples demonstrate that thousands computers can be harnessed as an Internet-based supercomputer to solve long-running problems using nothing but spare CPU time. What these applications have in common? First of all they try to solve some problem that is of public-interest.

The Bovine cooperative effort is probably the best good example where tens of thousands of users feel motivated to participate in a world-wide computation. However, the success of the Bovine project is also directly related with the fact it provides a Web-based interface with the usage statistics of the machines that contribute their cycles to the computations. The information about the work made by individual users, teams and countries is raising an healthy competition between them. Some users are organized in teams and contribute their CPU cycles with the desire to see his team grow in the rankings of statistics.

If the JET machine wants to seize volunteers to participate in a Web-based global computation it must keep a database with statistics that can be accessed through the Web. Information about the participating users and their machines, the contribution time and the corresponding performance could then be consulted by any Internet user.
The statistical information is completely independent from the underlying application: the jobs that are executed and who computed them is the basic data exchanged between the two modules (JET-Master and JET-STATS).

The JET Statistical Server (JET-STATS) has been built to serve not only as a collector of information but also to have many other important statistical features. We tried to implement a server of statistics with a flexible structure capable of collecting information, processing it in many different ways and also publishing it into the Web.

We have chosen to use a file-based approach instead of using a real database, although it would be easier to implement. However we decided to avoid all the unnecessary overhead that result from the time spent with the accesses to the database and a file-based approach provides a more faster implementation.

Another relevant thing to notice is that we have implemented a HTTP server in Java and all the mechanisms that will be explained next were implemented inside this HTTP server by Java Threads.

As was described before, the JET computation is based on Java applets. When a new worker wants to join the JET Machine, she has to fill in a form with her email or her statistical i.d. When a user enters for the first time, the statistical server gives her an unique internal i.d, that can be used as an identification in future computations and to search for the user’s information in the visual statistical module. If nothing is filled, the user is considered anonymous and all the user’s statistical data is not taken into account. The machine’s information is collected automatically using some applet facilities, without any intervention from the user.

The communication between the JET Master and the JET-STATS module is done through a TCP socket. This socket is open until the end of the computation and, in case of failure of one of the parts, it is somehow restored. The use of a permanent communication channel is justified by the flow of data that is almost permanent between the two modules. The information is received from the workers by the JET Master and all the relevant information of statistics is filtered and sent to the statistical server. The JET Master always identifies a worker by an internal id (worker_id) that is also sent with the statistics data.

To complement the efficient implementation of the server we have developed a Web interface with a high quality visual module. We developed a wide range of options and we integrated them in an attractive environment. The statistical visual module receives the data through HTTP requests. These requests are made to the “normal” HTTP server of the statistical machine or to the adapted JET HTTP server running on a different port. The first ones get the files already made (top rank pages) and the last ones get the dynamically generated statistical pages.

Everyone can access the statistical information, not only the users that have contributed to the computation. This is a normal procedure because, if we want to use this module as a global marketing unit, everyone is a welcome future potential volunteer.
The development of Web pages involves some techniques that can be used when we want to do something more than "click the link". The major visual challenge of this module is that it must be work fine in any browser or size of the monitor that is used by the end user. Some JavaScript features are a good solution for these cases because they are recognised by the majority of the browsers. The dynamically replacement of images and the usage of persistent data between queries are some of the benefits that we gained by using it. All the frame techniques that were used work in all types of monitors, to assure that the statistical information can be correctly seen by any type of display. In Figure 2 we can see a global view of the visual module.

Figure 2: Global view of the JET Statistical Visual Module.

There are three different frames with specific functions. The left frame one is where the user selects the data to be displayed. The scrollbar at the bottom of the page where are displayed the update times of the statistical server. The central frame is where all the statistical data is displayed.
4. Related Work

The internal structure of the JET-STATS is not based on a single concept or language approach. By the contrary, it uses a wide range of techniques to provide an efficient implementation of the server and a fancy graphical interface to the end-user. Most of the examples with statistical information showed in the Internet are based on CGI queries and database approaches. Our server combines the power of the Java language with some memory and file-based techniques to speed up the queries.

The Bovine Project [3] is currently the best example that provides a Web page with complete statistical information. Everyday, thousands of computers are cracking keys with the main objective of getting better in the world-wide rankings. This project has shown that statistic information is a powerful feature to get volunteers for a Web-based global computation. The GIMPS project [4] also offer some statistic information, but the information provided is not so appealing, thereby limiting the interest from possible volunteers.

WebSideStory, Inc. developed a Java visitor tracker for the pages in the Internet [14]. In that site we can get obtain a ranking of the mostly accessed Web-pages including a wide range of statistics that are updated dynamically by a Java applet. That server is not related to Web-based distributed computing, though.

There are several projects like JET for Web-based metacomputing: examples include like the Legion project [15], Globus [16], Globe [17], Web Computer [18], Charlotte [19], ATLAS [20], ParaWeb [21], Popcorn [22], JavaParty [23], Albatross [24] and the Javelin project [25]. The goal of all these projects is to provide a unified high-performance virtual machine that may span across several world-wide institutions and may consist of several workstations, personal computers and parallel machines.

To the best of our knowledge none of these projects provide a server of statistics like JET-STATS, but all of them could perfectly take advantage of our server since it was implemented in a modular way.

5 - Conclusions

The JET-STATS server is an useful complement to the JET Web-based parallel machine. In many of the cases it will be the decisive factor to persuade some volunteers to give their CPU cycles to a global computation. Our server is able to collect information from the master server, process it and display it in run-time in a Web page, taking advantage of optimization techniques to speedup all these features. As a complement to the statistics server, we have also developed a visual module, that shows the statistic data organized in some different rankings in a very fancy way.
With all the information provided by JET-STATS we can easily implement the concept of CPU market proposed in [22], where there are buyers and sellers of CPU cycles that execute global applications. The payment between buyers and sellers could then be made through e-cash, software or access to computer resources.

References