Survey on Volunteer Computing Frameworks for Scientific Applications

Shreesha Aithal 1, Naveen M 2, Vyaas Shenoy 3 and Suhas Shrinivasan 4

1,2,3 Information Science and Engineering, National Institute of Engineering
4 Computer Science and Engineering, National Institute of Engineering

Abstract - Scientific applications require immense computing power to produce quantifiable results. Computing power is a measure of the resource capacity offered by a computer. Scientific applications generally utilize supercomputers since they have very high computing power. Supercomputers are expensive and are not accessible to the general public. This is a problem for scientific application developers without access to supercomputers. The advent of volunteer computing is a solution for scientific application developers without access to supercomputers. Volunteer computing involves the use of a cluster of general purpose computers which have volunteered their computing power, resources and services to the scientific application.

Keywords—Distributed computing, volunteer computing, grid computing, BOINC, scientific application, framework, Akka.io, Apache Airavata.

I. INTRODUCTION

The world is filled with immense problems of great complexity. Research scientists work on solving or coming closer to the solution to these problems. Quite a lot of mathematical findings, biomedical research, biological advancement, findings in engineering, etc. have the use of computers in them. Greater the complexity of the problem, greater is the need for computing power. Supercomputers are generally used for scientific applications and experimentation on tasks of great complexity. Supercomputers are expensive, need special storing conditions and are not accessible to everybody. They are restricted only to universities and private organizations.

Volunteer computing is a type of computing where machines distributed over a network, owned by the general public solicit a small part of their resources and computing power to help provide the necessary resources for a scientific application. This reduces the cost of the infrastructure to a minimum and provides results like that of a single supercomputer.

There are several frameworks available in the market. BOINC, Akka.io and Apache Airavata are distributed volunteer computing frameworks having unique models to distribute, organize and maintain tasks involved in a scientific application.

II. BOINC

Stands for Berkeley Open Infrastructure for Network Computing[1]. It is a system for Public-Resource Computing and Storage. It makes it easier for scientific application developers to create public-resource computing projects.

There is a large amount of computing power is distributed in millions of personal computers belonging to the general public. Public-Resource computing uses these resources to do scientific supercomputing. SETI@home, Predictor@home, Folding@home, Climate@home and CERN projects all use BOINC to do scientific computation. BOINC is open source and is available at http://boinc.berkeley.edu.

A BOINC project corresponds to an organization that does public-resource computing. A single master URL is the home page of the website and also serves as a directory of scheduling servers. The server complex of BOINC is centered around a relational database which stores information about the application, results, platforms, versions, accounts and so on. Server processes
are run by a set of daemon processes and web services. Scheduling servers handle RPC from the clients. They issue work and handle the reports of tasks that are completed from the clients. The Data servers handle file uploads using a certificate based mechanism to ensure that only legitimate files are uploaded and which have a prescribed limit on the file size. File downloads are handled by standard HTTP.

Participants who wish to contribute to an ongoing project need to fill up a form at the BOINC website and download the BOINC client application. This client application runs in several modes. One as a screensaver that displays some graphics to the user when the system is not in use. It can also run as a windows service that can run even when there are no users logged on and log results onto a database and as a UNIX command line program that communicates through stdin, stdout and stderr and can be run from a cron job or startup file. The BOINC core client program implements a ‘local scheduling policy’. This policy decides when to get work and from what project, and what tasks to execute at a given point.

III. AKKA.IO

Akka\textsuperscript{[2]} is a software framework that makes it easy to build correct, scalable, fault-tolerant and concurrent applications. It uses the Actor model to raise the abstraction level. Supports applications to be written in Java and Scala. Akka is open source and is available at http://akka.io/downloads.

Actors are objects. They encapsulate state and behavior. They communicate by sending and receiving messages which are placed in the recipient’s mailbox. Actors the most stringent form of object-oriented programming. It is better to view them as persons. While modeling a solution with actors, visualize a group of people, assign subtasks to them, arrange their functions into an organizational structure and think about how to escalate failure. An Actor contains a State, Behaviour, a Mailbox and Children.

Actors in Akka.io form a hierarchy. One actor which looks after some activity in the program can split up into smaller units. It starts child actors for this purpose and supervises them. Supervision is done based on a certain policy to make sure work is done in the most efficient way possible and concurrency is maintained. Each actor has only one supervisor and can have multiple children. The children actors report to the supervisor actor in a recursive fashion. The reporting can include things such as errors, faults, results of some computation and so on.

IV. APACHE AIRAVATA

Apache Airavata\textsuperscript{[3]} is a software framework used to create, organize, run and monitor distributed applications and workflows. It is used on resources ranging from local resources to computational grids and the cloud. It builds on general concepts of service-oriented computing, distributed messaging and workflow composition and orchestration.

Airavata aspires to serve long running applications and workflows on distributed computational resources. The architecture is designed to be a software framework comprising of modular components. It provides desktop tools and browser based web-interfaces to manage applications, workflows and generated data. Server side tools for registering and managing scientific applications on computational resources, GUI for constructing, executing, controlling, managing and reusing of scientific workflows, interfacing and interoperability with various third party data, workflows and provenance management tools.

Airavata includes modules for Generic Application Factory Service - GFac, XBaya workflow suite, an interpreted workflow enactment server, information and data registry API and the WS-Messenger. All of these modules are developed and packaged as a standalone component and as well as integrated into the Airavata Suite.
V. CONCLUSION

The following three distributed application frameworks enable scientific application developers to write applications, get results and advance on their research without the use of expensive supercomputers or access to private organizations and universities. The reliable network of volunteers in the network can provide resources and computing power whose magnitude is comparable to that of a supercomputer. The three frameworks use different models each having a different approach to distributing tasks amongst volunteer machines. Each of the models have been highlighted in the survey.

REFERENCES