

Group Matrix Calculations for Computational Grid Data Integrity

Chad Mano, Aaron Striegel
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46530 USA
Email: {cmano, striegel}@nd.edu

Abstract—The abstract goes here.

I. INTRODUCTION

Computational grids are an effective means of efficiently processing a computationally intensive data set. Traditionally, research institutions had to enlist the power of a supercomputer in order to perform calculations for complex simulations or other research activities. Grid technology is able to produce a system with supercomputer level processing power that is comprised of consumer grade computer systems. Grid systems range from those privately owned and administered to those comprised of volunteer cpu cycles located throughout the internet.

Volunteer systems are generally a type of peer-to-peer network. (By some strict definitions these systems are not considered grids. However, we use a very broad and generic meaning of the word include all systems with distributed parallel computation to be in the class of grids). In these systems users are able to donate their unused CPU cycles to some project. In order to gain participants these systems rely on either the altruistic beliefs of a user, or a reward system which pays users with notoriety or perhaps monetarily. Seti@Home [1] and Folding@Home [2] are to popular examples of a volunteer types of system. A *consumer grid* is envisioned which will allow a anyone to submit their computations to a worldwide system of volunteer PCs [3].

There are a number of security concerns that must be dealt with in the world of grids, especially when dealing with a volunteer grid which is comprised of unknown hosts. An interesting paradox of the grid is that we enlist it for the sole purpose of taking some input and offering a result, yet we cannot trust the result returned. Results may be invalid due to either some type of system error or possibly even to a malicious host. A rewards based system inherently provides the motivation for cheaters to participate. In this case a user may look to obtain credit for work that is not actually completed. The cheater simply obtains a work unit and then submits a fabricated result since the real result requires cpu time. Malicious users are a threat to all systems. A malicious user is someone whose goal is to deflate the integrity of the computation for some purpose other than gaining the unwarranted rewards. However, no matter the underlying reason, error producing systems, cheaters, and malicious users are all

cheaters from our perspective. So, regardless of the source, faulty results much be identified and removed in order to preserve the integrity of the calculations.

An audit system must be used to validate results and eliminate cheaters. This paper presents a new method of auditing results from a distributed and untrusted computational nodes. There are four main goals of this study. First, to minimize the amount of computational redundancy while while maintaining the integrity of a highly redundant system. Second, to reduce the computational load on the main server. Third, to quickly and easily identify cheaters. Finally, to protect against a variety of attacks including collusion among cheaters.

The organization of this paper is as follows. Section II describes current methods of validating data and users. Section III gives a detailed framework of our method. Section IV presents comparisons between our method and others, gives results of simulations of a P2P grid system, and offers a probability analysis to determine data integrity levels. The final sections discuss future directions of this work and a summary of this paper.

II. CURRENT METHODS

Traditional means of auditing rely on a voting systems where a job is sent to multiple nodes for processing and the returned results are compared with the most commonly occurring results being deemed to be correct. Bolle presented the notion of ringers which is a set of data with know valid results that act as a kind of bait to catch a cheater. If someone returns incorrect results of set of ringer input then they are identified as a cheater. Szajda, et al, present a method of validating results in sequential computations that requires less processing than the traditional voting system method. Each of these methods will need to be explained in detail, especially the final method. Need to get references of these and a few other methods.

III. GROUP MATRIX COMPUTATION AND AUDIT

We now describe our method of computing and auditing distributed computational tasks. The work of Szajda, et al, in [4] as pertaining to sequential computations is similar to this work. Our approach differs in that computational requirements on the leader for verification is reduced as is the need for

completely trusted leader nodes. These differences lead to varying results which will be discussed in section IV.

As described by Szajda, the Great Internet Mersenne Prime Search (GIPMS) [5], is a prime (pun intended) example of a sequential task used in distributed computations. Sequential computations are those where the intermediate results are dependent on previous computations of the task. This is in contrast to non-sequential tasks where a node calculates results of many independent inputs on a single function. See [5] for a descriptive example of the sequential nature of GIMPS.

As stated previously, one of the goals of this research is to reduce the computational load on the main server. Assuming that the main server is the only completely trustworthy entity in the system, it is this server that typically audits results [?]. In a typical n -way redundant system, the server would obtain n results and check the similarity of the results in order to determine the probable correct answer based on votes. For a value $n = 2$ this is fairly simple, however, as n increases, the computational requirements increase also. For large datasets this can become quite significant.

It would behoove a system to take advantage of the distributed computational power of the grid to somehow audit the system. However, as we've learned from Enron[6], trusting an untrustworthy system to not only calculate results, but then to audit the results is not a good way to do business. While a pure audit by the grid is not feasible, we show how we can use the grid to reduce the amount of computational burden placed on the server while maintaining the benefits of highly redundant computations. In so doing we are actually able to reduce the computational load on the grid itself. This is another one of the four goals mentioned earlier.

The final two goals of identifying cheaters and protecting against attacks are somewhat obvious and do not warrant an argument as to why they are important.

Our system, Group Matrix Computation and Audit (GMCA), is comprised of three types of systems. First is the *coordinator* which is the main server that administers the entire system. The *coordinator* is the only completely trusted system in the entire network. Second are *managers* which are established nodes in the system that have reached some level of trust based on past performance. Finally, *workers* are nodes that have not reached the trust level to become a *managers*.

A. Process

This section describes the overall sequence of processing a group of sequential tasks. The description here is the basic method, modifications may be made later and discussed.

- 1) The *coordinator* creates a group by selecting one *manager* and assigning $m - 1$ *workers* to it. Note that other *managers* may be assigned also, but they act only as a worker. The result is a group of m nodes because the *manager* will also participate in the processing.
- 2) A group of m tasks, one of which is specified as task T , is assigned to a group G_1 . Another group of m tasks, including task T , is assigned to a group G_2 . The tasks

assigned to each group are unique save the single task T .

- 3) The *manager* divides the tasks into m segments.

IV. RESULTS AND ANALYSIS

V. FUTURE WORK

VI. SUMMARY

REFERENCES

- [1] "The Search for Extraterrestrial Intelligence Project," University of California, Berkley. [Online]. Available: <http://setiathome.berkeley.edu>
- [2] "The Folding@Home Project," Stanford University. [Online]. Available: <http://folding.stanford.edu>
- [3] Editorial Board, "Grid computing 101: what's all the fuss about?" *IEEE IT Professional*, vol. 6, no. 2, pp. 25-33, March-April 2004.
- [4] D. Szajda, B. G. Lawson, and J. Owen, "Hardening functions for large scale distributed computations." in *IEEE Symposium on Security and Privacy*, 2003, pp. 216-224.
- [5] "The Great Internet Mersenne Prime Search (GIMPS)." [Online]. Available: <http://www.mersenne.org/prime.htm>
- [6] "Explaining the Enron Bankruptcy," Associated Press. [Online]. Available: <http://archives.cnn.com/2002/US/01/12/enron.qanda.focus>