CyberBridges Fellowship Progress Report

Project: Performance Evaluation of Large-Scale Scientific Applications on Grid Computing Environments with PRIME

Reporting Period: Period Ending December 1, 2009

Principal Authors: Miguel A. Erazo

 Roberto Pereira

Institution: Florida International University

1. **Introduction**

The **goal** of this project is to study the energy consumption of distributed and parallel simulations on a grid computing environment using the PRIME network simulator. Since we cannot measure the energy signature of an application which runs on a grid environment directly, we plan to approximate it estimating the power footprint of each application across different computing resources. Specifically, we estimate the energy consumption of the **CPU, memory, disk,** and **network** following a similar technique to that of [1]. That is, having from the technical specification sheets the power that each computing resource consumes and getting from the experiments the time that the application uses that resource; we can compute the total energy consumed by the application in the experimentation time. It is important to remark that each resource may have more than one state and consequently more than one power level. Consequently, for a more detailed study it is necessary, although not always possible, to take into account the many states a computing resource is along the experimentation time.

1. **Work completed during this time period**

During this time period a number of activities were performed that can be classified into two broad areas: *Installation* and *Methodology*. Installation activities are those conducted to install, test and schedule experiments using PRIME on Teragrid while Methodology ones aim to find a suitable approach to approximate the energy consumption of network models on PRIME.

PRIME network simulator was installed in the Teragrid environment, specifically Abe and Lincoln. In order to verify that PRIME was running correctly, network models have been run using the PBS scheduler provided by Teragrid. During the installation process, we studied and tested many tools provided by Teragrid that will help achieve our objectives: Perfsuite, psrun, psprocess.

PerfSuite is a collection of tools, utilities, and libraries for software performance analysis. This tool uses the Performance Application Programming Interface (PAPI), which provides a consistent interface and methodology for use of the ***performance counter hardware*** found in most major microprocessors. PAPI enables engineers to see, in near real time, the relation between software performance and processor events. These counters exist as a small set of registers that count *events*, occurrences of specific signals related to the processor's function. Teragrid has some Persuite-enabled machines which enable monitoring of hardware events, including **Abe** and **QueenBee**. psrun is a PerfSuite command-line utility that can be used to gather hardware performance information on an **unmodified** executable. We used this utility to gather information about PRIME running on **Abe**, **Lincoln, and QueenBee**. Running this command using an executable as a parameter creates an xml file with performance information related specifically to the specified application. psprocess is a PerfSuite command-line utility that can be used to post-process the results of a performance analysis experiment. The execution of this command causes a number of derived metrics to be generated based on the ones measured when running psrun. It is important to remark that, depending in the hardware architecture, only some events captured at processor-level are available.

As pointed out before, we use a similar technique as that described in [1] to estimate the energy consumption. This technique consists in measuring the time that an application, i.e. PRIME, uses each computing resource and then derive the energy consumption by extracting from the specifications the power signature of each these resources. The expression below shows how we compute the energy consumption on per-application basis.

$$E\_{Total}= E\_{CPU}+E\_{Mem}+E\_{Disk}+E\_{Net}=\sum\_{i\in Resources}^{}\left(\sum\_{j\in States\_{i}}^{}\left(P\_{ij}\*f\_{ij}\right)\right)\*T (1)$$

Where:

*Resource*: set of computing resources taken into account for energy consumption

*Statesi*= Set of states of computing resource i

pij= Power of resource i in state j

fij= Percentage of time that resource i is in state j during experimentation time

*T* = Experimentation time (wall-clock time)

We devise two methods for measuring CPU time: 1) Using performance counters in conjunction with Perfsuite, and 2) using the ***pidstat*** command which is part of the **SYSSTAT** set of utilities. The SYSSTAT utilities are a collection of performance monitoring tools for Linux that include: sar, sadf, mpstat, iostat, pidstat and sa tools. The pidstat command is used for monitoring **individual tasks** currently being managed by the Linux kernel. However, the **-d** option of the pidstat command is not available in Teragrid because it can only be used for kernels 2.6.20 and later only, and the kernel installed in Abe and QueenBee are **2.6.18-92.1.10.el5\_lustre.1.6.6smp-perfctr** and **2.6.9-55.0.9.EL\_lustre.1.6.4custom** respectively. Thus, we plan to use Perfsuite for measuring CPU time. We can precisely measure the time that our application, namely PRIME, is using the CPU. According to the documentation, the correct measure is ***CPU time*** (seconds). This measure is generated from the use of psprocess and is computed as:

$$CPU seconds= \frac{cycleCounter}{clockSpeed} (2)$$

Where:

*cycleCounter* = Number of CPU cycles our application has used

*clockSpeed* = Number of clock cycles per second

Following the previously described method we can get with high precision the CPU time and using expression (1) for the CPU and taking into account two states for CPU we get:

$$E\_{CPU}=\left(P\_{active}\*f\_{active}+P\_{idle}\left(1-f\_{active}\right)\right)\*T (3)$$

In case that data/instructions are not found in L1 and L2 caches, the main memory is accessed. Also, when this happens, not only a **line** is fetched from the memory but also more data from the memory is loaded by the pre-fetcher. Thus, if we have the total number of accesses to RAM and the time each access occupied the memory then we can compute the memory time. The PAPI event **PAPI\_PRF\_DM** (Data prefetch cache misses) **is not available** in the infrastructure provided by Abe in Teragrid. Thus, as an approximation, we compute the memory time taking into account the number of accesses due to **L2 cache misses** only. For the main memory we will consider the *Total Access Time*, which is composed by the *Address Transport Time*, the *Data Access Time* and the *Data Transport Time*. Since the memory controller and the data array operate at the same clock, the Address Transport Time is 1 clock cycle (equivalent to the cycle time, 3ns for a DDR clock of 667MHz).

The Data Access Time is given by the formula:

$Data Access Time= t\_{RCD}+\left(\left(CL-1\right)\*Cycle Time\right)+ t\_{AC} (4)$

Where:

***tRCD*: the RAS to CAS delay obtained from the memory datasheet.**

***CL*: is the CAS Latency obtained from the memory datasheet.**

***Cycle Time*: is the time it takes for a complete clock cycle.**

***tAC*: is the Access Timewhich is not given in the memory datasheet but it can be approximated and taken to be 1ns.**

**The Data Transport Time is given by the formula:**

$$Data Transport Time= t\_{DPD}+BMM\*Cycle Time (5)$$

**Where:**

***tDPD*: the Data Propagation delay and is equal to the Cycle time minus the Access Time.**

***BMM*: the number of consecutive access that will be performed in burst mode, 3 in this case.**

**For the Hard disk drive we will use the Internal Sustained Transfer Rate because we will be using relatively large files (more than the maximum sector capacity of 512kb) and we will assume that these files are not fragmented. Since this rate varies depending on what sector the files are written to or read from, which is not practically feasible to determine, we will use the average Internal Sustained Transfer Rate obtained from the datasheet of the device.**

1. **Work to finish the project**

The following activities remain to be done:

* Find a suitable methodology for approximating the energy consumption of the network.
* Pick a network model to be used for the experiments.
* Run the experiments on Teragrid.
* Process results.
* Compose the paper.
1. **References**

[1] Kansal, A., and Zhao, F. "Fine-grained energy profiling for power-aware application design", In Workshop on Hot Topics in Measurement and Modeling of Computer Systems (2008)

[2] PAPI: <http://icl.cs.utk.edu/papi/>